Twin Builder Help

ANSYS, Inc.
Southpointe
2600 Ansys Drive
Canonsburg, PA 15317
ansysinfo@ansys.com
https://www.ansys.com
(T) 724-746-3304
(F) 724-514-9494

Release 2024 R2
July 2024

ANSYS, Inc. and ANSYS Europe, Ltd. are UL registered ISO 9001:2015 companies.

Copyright and Trademark Information

© 1986-2024 ANSYS, Inc. Unauthorized use, distribution or duplication is prohibited.

ANSYS, Ansys Workbench, AUTODYN, CFX, FLUENT and any and all ANSYS, Inc. brand, product, service and feature names, logos and slogans are registered trademarks or trademarks of ANSYS, Inc. or its subsidiaries located in the United States or other countries. ICEM CFD is a trademark used by ANSYS, Inc. under license. All other brand, product, service and feature names or trademarks are the property of their respective owners. FLEXlm and FLEXnet are trademarks of Flexera Software LLC.

Disclaimer Notice

THIS ANSYS SOFTWARE PRODUCT AND PROGRAM DOCUMENTATION INCLUDE TRADE SECRETS AND ARE CONFIDENTIAL AND PROPRIETARY PRODUCTS OF ANSYS, INC., ITS SUBSIDIARIES, OR LICENSORS. The software products and documentation are furnished by ANSYS, Inc., its subsidiaries, or affiliates under a software license agreement that contains provisions concerning non-disclosure, copying, length and nature of use, compliance with exporting laws, warranties, disclaimers, limitations of liability, and remedies, and other provisions. The software products and documentation may be used, disclosed, transferred, or copied only in accordance with the terms and conditions of that software license agreement.

ANSYS, Inc. and ANSYS Europe, Ltd. are UL registered ISO 9001: 2015 companies.

U.S. Government Rights

For U.S. Government users, except as specifically granted by the ANSYS, Inc. software license agreement, the use, duplication, or disclosure by the United States Government is subject to restrictions stated in the ANSYS, Inc. software license agreement and FAR 12.212 (for non-DOD licenses).

Third-Party Software

See the legal information in the product help files for the complete Legal Notice for Ansys proprietary software and third-party software. If you are unable to access the Legal Notice, contact ANSYS, Inc.

Table of Contents

Table of Contents **Contents-1**

1 - Welcome to Twin Builder™ Help **1-1**

Twin Builder User Interface Quick Links 1-1

Projects Quick Links 1-2

Designs Quick Links 1-2

Components Quick Links 1-3

Analysis Quick Links 1-3

Working with Legacy Files Quick Links 1-4

Optimetrics Quick Links 1-4

Results Quick Links 1-5

Scripting Quick Links 1-5

Twin Builder Getting Started Guides 1-6

2 - Getting Started **2-1**

Launching Twin Builder 2-1

Welcome to the Ansys Electronics Desktop 2-3

Launching Ansys Electronics Desktop 2-5

Installing PyAEDT (Beta) 2-7

System Requirements 2-9

Twin Builder Student Limitations 2-9

The Twin Builder Desktop 2-10

The Project Manager Window 2-12

Setting the Project Tree to Expand Automatically 2-15

Viewing Twin Builder Design Details 2-15

The Definitions Folder 2-16

The Component Libraries Window 2-16

The Components Tab 2-18

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Simplorer Element Library Symbols 2-19
Using the Favorites and Most Recently Used Lists 2-20
Help for Components 2-21
Loading Component Example Projects 2-21
Placing Components on a Schematic 2-21
The Search Tab 2-21
Properties Window 2-22
Param Values Tab (Properties Window) 2-24
General Tab (Properties Window) 2-25
Symbol Tab (Properties Window) 2-25
Quantities Tab (Properties Window) 2-27
Signals Tab (Properties Window) 2-27
Showing and Hiding the Properties Window 2-27
Launching Help from the Properties Window 2-27
The Properties Dialog Box 2-28
On-sheet Component Properties Dialog Box 2-28
Design Properties Dialog Box 2-29
Parameter Defaults Tab (Design Properties Dialog Box) 2-29
Parameter Values Tab (Properties Dialog Box) 2-30
General Tab (Properties Dialog Box) 2-33
Symbol Tab (Properties Dialog Box) 2-33
Quantities Tab (Properties Dialog Box) 2-33
Signals Tab (Properties Dialog Box) 2-33
Array/Record Element Values Dialog Box 2-34
Property Displays Tab (Properties Dialog Box) 2-36
Add/Edit Property Dialog Boxes 2-37
The Message Manager 2-38
Hiding the Message Manager Window Until Messages Appear 2-39

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Clearing Messages ... 2-39
Showing Message Details ... 2-40
The Progress Window ... 2-40
The Design Area ... 2-41
Schematic Editor Window ... 2-41
Symbol Editor Window ... 2-43
Netlist Editor Window ... 2-43
Model Editor Window ... 2-44
Script Editor Window ... 2-45
Report Window ... 2-45
Window Layouts ... 2-46
The Status Bar ... 2-47
Desktop Menus ... 2-47
Working with the Menu Bar ... 2-48
External User Tools ... 2-49
Shortcut Keys ... 2-50
Twin Builder Desktop shortcut keys ... 2-50
Choosing the View Navigation Options ... 2-52
Editing Keyboard Shortcuts ... 2-54
Undo and Redo Commands ... 2-55
Choosing a Color Scheme [Beta] ... 2-55
Working with Ribbons ... 2-57
Running Twin Builder from a Command Line ... 2-60
Examples and Further Explanations of -batchoptions Use ... 2-67
For -batchoptions Use: Project Directory and Lib Paths ... 2-69
For -batchoptions Use: TempDirectory ... 2-70
For -batchoptions Use: Various Desktop Settings ... 2-70
Obtaining Information about the Software and Release ... 2-74

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Debug Logging ... 2-74
3 - Help Menu ... **3-1**
Finding Information in the Help ... 3-2
Using the Search Function in the Help ... 3-2
Performing a Basic Search ... 3-3
Getting Help from Ansys Technical Support ... 3-7
The Ansys Product Improvement Program ... 3-8
How to Participate ... 3-8
How the Program Works ... 3-8
Data We Collect ... 3-8
Data We Do Not Collect ... 3-9
Opting Out of the Program ... 3-9
The ANSYS, Inc., Privacy Policy ... 3-9
Frequently Asked Questions ... 3-9
Conventions Used in the Help ... 3-11
What's New in this Release ... 3-13
4 - Working with Twin Builder Projects ... **4-1**
Setting Options for Twin Builder ... 4-2
Setting General Options ... 4-3
General Options: Desktop Configuration ... 4-3
Distributing Optimetrics Variations ... 4-4
General Options: Project ... 4-6
General Options: Miscellaneous ... 4-7
General Options: User Interface ... 4-7
General Options: Directories ... 4-10
General Options: Desktop Performance Options ... 4-11
General Options: Default Units ... 4-12
General Options: Remote Analysis ... 4-13

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

General Options: Component Libraries Options 4-14
Options: Optimetrics Options 4-15
Setting Twin Builder Options 4-15
Twin Builder Options: General Options 4-16
Twin Builder Options: Port Options 4-16
Conservative Domain Port Source Components 4-18
Conservative Domain Port Impedance Components 4-20
Quantity Domain Port Source Components 4-24
Twin Builder Options: C-Model Options 4-26
Twin Builder Options: Spice Compiler Options Tab 4-27
Twin Builder Options: VHDLCompiler Options Tab 4-27
Twin Builder Options: Modelica Compiler Options 4-28
Twin Builder Options: Python Model Options 4-29
Model Editor Settings 4-30
Setting HPC and Analysis Options 4-30
Configurations Tab 4-31
Options Tab 4-32
Licensing Settings Tool 4-34
Exporting Options Files 4-36
Setting Options via Configuration Files 4-37
Setting or Removing Option Values in Configuration Files: UpdateRegistry Command 4-41
Example Uses for Export Options Features 4-45
User Options and the Update Registry Tool 4-48
Getting a Value from a Specific Configuration File 4-48
Getting a Value Using Precedence Rules 4-49
Example of Removing a Host Dependent User Option Setting 4-49
Example Adding a Host-Independent User Option Setting 4-49

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Setting the Temporary Directory 4-50
Temporary Directory Configuration File Format 4-52
UNC Example 4-53
Single Quote Example 4-53
Setting the Temporary Directory using the GUI 4-54
Setting the Temporary Directory From the Command Line 4-54
Setting or Removing Temporary Directory Values in Configuration Files: UpdateRegistry Command 4-55
Batchoptions Command Line Examples 4-55
Batchoptions File Format 4-55
Example -BatchOptions with -Remote 4-56
Example -Batchsolve for Local 4-57
Batchoptions and Analysis Configurations in the Registry 4-58
How Analysis Configurations are Stored in the Registry 4-58
Using HPC and Analysis Options for Configurations 4-59
Batch Options 4-59
Setting Analysis Configurations Using the User Interface 4-62
Twin Builder File Types 4-62
Setting Up a Twin Builder Design 4-63
Creating Projects 4-64
Opening Existing Projects 4-64
Opening Recent Projects 4-65
Opening Example Projects 4-66
Translating Legacy Simplorer Projects and Schematics 4-67
When Component Terminals Do Not Match the Model Terminals 4-67
Importing ANF Design Data 4-68
Closing Projects 4-69
Saving Projects 4-69

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Saving a New Project ... 4-70
Saving a Copy of a Project ... 4-71
Saving Project Data Automatically ... 4-71
Recovering Project Data in an Autosave File ... 4-72
Archiving Projects ... 4-73
Restoring Archives ... 4-77
Renaming a Project ... 4-78
Deleting Projects ... 4-78
Removing Unused Definitions from a Project ... 4-79
Datasets ... 4-79
Datasets Dialog Box ... 4-80
Adding Datasets ... 4-81
Adding Datasets Manually ... 4-81
Dataset Preview Plot Properties ... 4-82
Importing Datasets ... 4-82
Import Data from Generic Text Files ... 4-84
Editing Datasets ... 4-86
Cloning Datasets ... 4-86
Exporting Datasets ... 4-87
Removing Datasets ... 4-87
Using SheetScan ... 4-88
SheetScan Toolbars ... 4-89
SheetScan Settings ... 4-89
The Curve Values Window ... 4-90
Loading a Datasheet Picture into SheetScan ... 4-90
Deleting a Datasheet Picture ... 4-91
Defining a SheetScan Coordinate System ... 4-91
Defining a Characteristic Curve in SheetScan ... 4-92

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Selecting a SheetScan Characteristic Curve ... 4-92
Changing Characteristic Curve Settings ... 4-92
Editing a SheetScan Characteristic Curve ... 4-93
Deleting a SheetScan Characteristic Curve ... 4-93
Checking for Monotonicity in X ... 4-93
Importing Characteristic Data into SheetScan ... 4-94
Exporting SheetScan Data ... 4-94
Undoing Commands ... 4-95
Redoing Commands ... 4-96
Inserting a Documentation File ... 4-96
Importing and Updating Twin Builder Models and Components ... 4-97
Importing Twin Builder Models ... 4-97
Handling of Incompatible Definition Names during Simulation Model Import ... 4-103
Related Topic ... 4-103
Using the Compile & Update Project Command ... 4-103
Saving Uncompiled Model Changes ... 4-105
Exporting a VHDL-AMS Model Description ... 4-107
Setting Read Only Designs ... 4-109
Printing ... 4-109
Adding and Saving Project Notes ... 4-110
Event Callbacks ... 4-111
Using Legacy DSO Licenses for Parametric Analysis ... 4-114
Working with PExprt Designs ... 4-115
Opening an Existing Design Using PExprt ... 4-115
Creating a New Design using PExprt ... 4-116
Using the Material Editor ... 4-118
Editing an Existing Material ... 4-118
Creating a New Material ... 4-118

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The View/Edit Material Dialog Box ... 4-119
Properties of the Material List ... 4-120
Specifying Thermal Quadratic Parameters ... 4-121
Material Properties ... 4-121
Specifying Thermal Modifiers ... 4-123
Calculating the Properties for a Permanent Magnet ... 4-127
5 - Minerva Remote Storage Environment ... 5-1
6 - Working with Libraries ... 6-1
Introduction: Components and Component Libraries ... 6-1
Component Definition Overview ... 6-1
Parameter Data and Dependent Data ... 6-2
Twin Builder Netlist ... 6-3
Library Concepts ... 6-3
Library Directories ... 6-4
The VHDL-AMS work Library in Twin Builder ... 6-4
Specifying the Location of Library Files ... 6-4
Managing Library Aliases ... 6-5
Components and Library Search Precedence ... 6-6
Updating Project Definitions from Library Definitions ... 6-6
Model and Library Synchronization ... 6-7
Synchronizing Project Definitions ... 6-7
Automatic Library Synchronization ... 6-8
On-Demand Library Synchronization ... 6-8
Translating Legacy Libraries ... 6-9
Editing Translated Components, Symbols, and Models ... 6-9
Ansys Customer Portal Extension Library ... 6-9
Managing Library Contents ... 6-10
Adding Definitions to Libraries via the Project Manager ... 6-11

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Exporting Definitions to User Libraries via the Project Manager 6-11
Exporting and Importing Definition Archives 6-11
Exporting Definition Archives 6-12
Importing Definition Archives 6-13
Editing Materials Libraries 6-15
Editing Scripts Libraries 6-15
Editing Component Libraries 6-16
Exporting Hierarchical Components 6-17
Editing Model Libraries 6-18
Editing Packages Libraries 6-19
Editing Symbol Libraries 6-20
Twin Builder Modelica Connector Library 6-20
Structure of Connector Library 6-21
Examples 6-22
RLC Individual 6-22
RLC Integrated 6-24
Simplified Motor Driven Power Train_conservative 6-25
Managing Library Files 6-28
Exporting a Model as an FMU 6-29
Exporting a Twin Model as a Twin 6-31
Recompiling, Encoding, and Licensing Libraries 6-32
Using the Password Manager 6-34
Adding an Encryption Resource 6-35
Deleting an Encryption Resource 6-35
Encryption Settings Dialog Box 6-36
Time License Settings 6-36
Using the Component Editor 6-37
Editing Components 6-37

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Editing an Existing Component ... 6-38
Adding a New Component ... 6-38
The Edit Component Dialog Box ... 6-39
General Tab (Edit Component) ... 6-39
Miscellaneous Tab (Edit Component) ... 6-41
Terminals Tab (Edit Component) ... 6-41
Simulation Models Tab (Edit Component) ... 6-43
The Properties Dialog Box (Edit Component) ... 6-43
Parameter Defaults Tab (Edit Component) ... 6-43
Properties Tab (Edit Component) ... 6-44
Quantities Tab (Edit Component) ... 6-45
Signals Tab (Edit Component) ... 6-47
Adding and Editing Properties (Edit Component) ... 6-47
Removing Properties (Edit Component) ... 6-48
The Component Netlist String Property ... 6-48
Netlist String Syntax Reference ... 6-48
Using the Symbol Editor ... 6-52
Creating and Editing Symbols ... 6-52
Editing an Existing Symbol ... 6-52
Creating a New Symbol ... 6-53
The Symbol Editor ... 6-53
Symbol Editor Shortcut Menus ... 6-54
Symbol Graphic Object Levels ... 6-55
Arranging Symbol Elements ... 6-56
The Draw Menu ... 6-56
Arc ... 6-57
Circle ... 6-57
Line ... 6-57

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Polygon 6-58
Rectangle 6-58
Text 6-59
Image 6-59
Pin 6-60
Rotate 6-62
Align Horizontal 6-62
Align Vertical 6-63
Flip Vertical 6-63
Flip Horizontal 6-63
Bring to Front 6-63
Send to Back 6-63
The Symbol Menu 6-64
Update Project 6-64
Set Name 6-64
Symbol Property Display Setup 6-64
Pin List (Symbol) 6-65
Grid Setup (Symbol) 6-65
Import File (Symbol) 6-66
Export File (Symbol) 6-67
Edit Component (Symbol) 6-67
Animate 6-67
List Symbol Graphic Elements 6-67
Editing Pin Properties 6-68
The Pin List Dialog Box 6-68
Using Pin Labels 6-69
Changing Pin Properties 6-69
Using the Script Editor 6-70

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Starting the Script Editor ... 6-70
Script Editor Menu ... 6-70
Script Editor Ribbon ... 6-71
Using the C-Model Editor ... 6-72
Adding a C-Model ... 6-73
Adding Analysis Types, Nodes, States, Variables, and Models to a C-Model ... 6-74
Adding Analysis Types ... 6-75
Adding Terminal Nodes ... 6-76
Editing a Terminal Node ... 6-77
Adding Quantity Nodes ... 6-78
Editing a Quantity Node ... 6-79
Adding Internal State Definitions ... 6-79
Editing Internal State Definitions ... 6-80
Declaring Variables ... 6-80
Editing Variable Declarations ... 6-81
Adding Models ... 6-81
Implementing the Initialize, Simulate, Validate, and Close Functions ... 6-82
Setting Matrix Entries ... 6-82
Initializing Variables ... 6-84
Setting the Values of Quantity Nodes ... 6-85
Checking Syntax (C-Model Editor) ... 6-85
Building a C-Model Component ... 6-86
C-Model Editor File Settings ... 6-86
Source File Settings Tab ... 6-87
Model Update Settings Tab ... 6-87
Configuring Build Settings ... 6-87
Compiling and Updating a Project to Include the C-Model ... 6-89
Debugging a C-Model Created with the C-Model Editor ... 6-89

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Revising an Existing C-Model with the C-Model Editor 6-89
Starting the Revision Process 6-90
Revising a C-Model 6-90
Implementing the Changes 6-90
C-Model Editor Menus and Windows 6-91
C-Model Editor Main Menu 6-91
C-Editor Standard Functions 6-92
Main C-Editor Window 6-92
C-Editor Message Window 6-92
Insert Data Menu 6-93
Simulator Data Menu 6-93
Callback Functions Menu 6-93
Sample C-Model Source Code 6-94
RLC Low-Pass Filter Sample Code 6-94
Linear Characteristic Model Sample Code 6-103
Using the VHDL-AMS Model Editor 6-107
Adding a VHDL-AMS Model 6-108
Editing a VHDL-AMS Model 6-108
Editing Libraries, Packages, and Models 6-109
Adding Terminal, Quantity, or Signal Ports to an Entity Description 6-109
Adding Generics to an Entity Description 6-110
Editing an Entity 6-111
Renaming an Entity 6-111
Adding an Architecture 6-111
Editing an Architecture Declaration 6-112
Renaming an Architecture Declaration 6-113
Checking Syntax (VHDL Model Editor) 6-113
Instantiating a Predefined Component in an Architecture Description 6-114

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Updating a Project to Add a VHDL-AMS Model and Component 6-114
Encrypting or Encoding a VHDL-AMS Model 6-115
Using IEEE Standard Encryption for VHDL-AMS Models in Twin Builder 6-116
Twin Builder Encryption Public Key 6-116
Twin Builder Command-line Encryption Tool 6-116
Decryption in Twin Builder 6-117
Revising an Existing VHDL Model 6-117
VHDL-AMS Model Editor Menus and Windows 6-117
VHDL Model Editor Main Menu 6-118
VHDL-AMS Edit Model Dialog Box 6-118
VHDL Model Editor Standard Toolbar 6-120
VHDL-AMS Editor Window 6-120
VHDL-AMS Editor Message Window 6-121
Using the SML Model Editor 6-121
Adding an SML Model 6-122
SML Editor Window 6-122
Editing an External Model 6-123
Inserting a Dataset in the SML Editor 6-124
Dataset Parameters Tabs 6-125
Checking Syntax (SML, and SPICE Model Editors) 6-126
Encrypting or Encoding an SML or SPICE Model 6-126
Using the SPICE Model Editor 6-127
Adding a SPICE Model 6-128
SPICE Editor Window 6-128
Using the Package Editor 6-129
Adding a Package 6-129
Editing a Package 6-130
Package Editor Window 6-130

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Checking Package Syntax ... 6-131
Encrypting or Encoding a VHDL-AMS Package ... 6-131
Using the Modelica Model Editor ... 6-133
Modelica Models in Twin Builder ... 6-133
Compiling and Updating a Project to Add a Modelica Model and Component ... 6-133
Use External Solver ... 6-134
Initialization of Input Values ... 6-135
Check Object Syntax ... 6-135
Check Object Semantics ... 6-135
Selecting Component Interface ... 6-136
Exporting Modelica Text As ... 6-138
Time License Settings for Modelica ... 6-139
Encrypting or Encoding a Modelica Model ... 6-140
7 - Device Characterization Wizard ... 7-1
Characterize Device ... 7-1
Characterization Flow for IGBT and Power MOSFET ... 7-2
Component Information ... 7-5
Nominal Working Point Values ... 7-14
Nominal Working Point Values for IGBTs ... 7-14
Vendor-Specific Guidelines for Nominal Working Point Values ... 7-16
Nominal Working Point Values for Power MOSFETs ... 7-17
Breakthrough Values ... 7-19
Test Circuit Conditions ... 7-22
Vendor-Specific Guidelines for Basic Dynamic IGBT Half-Bridge Test Circuit Condition ... 7-25
Transfer Characteristics ... 7-31
Boundary Conditions for Transfer Characteristic Data ... 7-33
Fitting Ranges for Transfer Characteristic ... 7-33

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Fitting Characteristic Order for Transfer Characteristic 7-34
Fitting Transfer Characteristic Data 7-34
Output Characteristic 7-34
Boundary Conditions for Output Characteristic Data 7-36
Fitting Ranges for Output Characteristic 7-37
Fitting Characteristic Order for Output Characteristic 7-37
Fitting Output Characteristic Data 7-38
Freewheeling Diode Characteristic 7-39
Boundary Conditions for Freewheeling Diode Characteristic 7-41
Fitting Ranges for Freewheeling Diode Characteristic 7-41
Fitting Characteristic Order for Freewheeling Diode Characteristic 7-41
Fitting Freewheeling Diode Characteristic Data 7-41
Including Thermal Effects in the IGBT Model 7-42
IGBT Thermal Model 7-42
IGBT Thermal Fitting Mode 7-43
Transient Thermal Impedance 7-44
Fraction Coefficients Mode 7-45
Thermal Network Topology 7-46
Partial Fraction Model 7-46
Continued Fraction Model 7-46
IGBT Heatsink Parameters 7-47
Fitting Characteristic Data Curves 7-47
Freewheeling Diode Thermal Model 7-47
Energy Characteristics (Average IGBT and Average Power MOSFET) 7-49
Energy Characteristic Default Row Description 7-51
Fitting Types of Static Characteristics 7-52
Dynamic Model Input (Basic and Advanced Dynamic IGBT and Basic Dynamic Power MOSFET) 7-52

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

QRR 7-56
TxG, TxI, TxV 7-60
Dynamic Parameter Validation (Basic and Advanced Dynamic IGBT and Basic Dynamic Power MOSFET) 7-62
Validation (Average IGBT and Average Power MOSFET) 7-64
Model Parameters 7-66
Adding/Deleting Characteristic Data Curves 7-69
Plotting Characteristics Data 7-70
Selecting a Characteristic Curve 7-71
Working with Characteristic Data Curves 7-72
Fitting Data to the Model 7-73
Validate 7-74
Show Result 7-75
Show Log 7-76
Fitting Info 7-76
Start Fitting 7-79
Import Model 7-80
Save Model 7-81
Close the Device Characterization Wizard 7-81
Tutorial for Characterizing a Basic Dynamic IGBT 7-82
Preliminary Considerations using Basic Dynamic IGBT Data Sheet Information 7-82
Using the Device Characterization Wizard for a Basic Dynamic IGBT 7-83
Advanced Settings 7-88
Advanced Settings for Characterizing a Basic Dynamic IGBT 7-89
Tutorial for Characterizing a Power MOSFET (Basic Dynamic Model) 7-98
Preliminary Considerations Using Power MOSFET Data Sheet Information 7-99
Using the Device Characterization Wizard for a Power MOSFET 7-99
Advanced Settings for Characterizing a Power Mosfet 7-107

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Tutorial for Characterizing a Power MOSFET (Average) 7-116
Preliminary Considerations Using Power MOSFET (Average) Data Sheet Information 7-117
Using the Device Characterization Wizard for a Power MOSFET (Average) 7-117
Tutorial for Characterizing a Power Diode 7-122
Preliminary Considerations using Power Diode Data Sheet Information 7-123
Using the Device Characterization Wizard for a Power Diode 7-123
Advanced Settings for Characterizing a Power Diode 7-130
Tutorial for Characterizing a Thyristor 7-136
Preliminary Considerations using Thyristor Data Sheet Information 7-137
Using the Device Characterization Wizard for a Thyristor 7-137
8 - Power Module Characterization Wizard 8-1
Using System Library Power Power Module Models 8-1
Modifying System Library Power Module Models 8-3
Creating a Power Module Behavioral Model 8-3
Creating a New Power Module Model Library 8-4
Adding a New Behavioral DC/DC Converter Model 8-4
Defining a Static Converter Model 8-5
Generating VHDL-AMS Code and Importing the Model into Twin Builder 8-8
Defining the Converter Model Dynamic Behavior 8-9
Defining Event Driven Behavior 8-11
Enabling Current Sharing and Thermal Modeling 8-12
Modeling of DC-DC Converters Based on Hybrid Wiener-Hammerstein Structure 8-14
Proposed Hybrid Wiener-Hammerstein Structure 8-15
Detailed Wiener-Hammerstein Structure 8-16
Event-driven Behavior 8-21
9 - Component Dialog Wizard 9-1
Creating a New Component Dialog Box Using the Component Dialog Wizard 9-2

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding Static and Control Elements to a Component Dialog Box 9-2
Changing the Name of a Component Dialog Box 9-3
Adding Groups to a Component Dialog Box 9-4
Adding Statics to a Component Dialog Box 9-5
Adding Text Edits to a Component Dialog Box 9-7
Adding Combo Boxes to a Component Dialog Box 9-10
Adding Check Boxes to a Component Dialog Box 9-15
Adding Radio Buttons to a Component Dialog Box 9-21
Adding List Boxes to a Component Dialog Box 9-23
Determining a Non-conservative Input Node's Value Using Free Values 9-26
Aligning and Sizing Component Dialog Box Elements 9-29
Component Dialog Box Grid 9-29
Component Dialog Box Lock Controls 9-29
Component Dialog Wizard Size Controls 9-32
Setting the Tab Order of a Component Dialog Box 9-35
Setting the Display Behavior of Component Dialog Box Elements 9-37
Component Dialog Box Display Behaviors 9-37
Lock Behavior 9-37
Relative Behavior 9-39
Default Behavior 9-40
Selecting the Dialog Box Display Behavior of a Dialog Element 9-41
Selecting Dialog Box Resize Behavior 9-41
Setting Dialog Box Element Conditions for Display 9-43
Testing a Component Dialog Box 9-46
Using Show Component Dialog 9-46
Using Check Component Dialog 9-47
Saving and Closing a Component Dialog Box 9-48
Using a Component Dialog Box 9-49

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Revising a Component Dialog Box 9-49
Component Dialog Wizard Window Items 9-49
Component Dialog Wizard Menu Bar 9-49
Wizard Menu (Component Dialog Wizard) 9-49
Edit Menu (Component Dialog Wizard) 9-50
Layout Menu (Component Dialog Wizard) 9-50
Controls Menu (Component Dialog Wizard) 9-51
Page Menu (Component Dialog Wizard) 9-51
Component Dialog Wizard Standard Toolbar 9-51
Component Dialog Wizard Layout Toolbar 9-52
10 - Working with Variables 10-1
Variable Types 10-1
Defining a Project Variable 10-4
Viewing and Editing Project Variables 10-6
Deleting Project Variables 10-6
Defining Local (Design) Variables 10-8
Adding a Local Variable at the Design Level 10-8
Adding a Local Variable from a Component 10-10
Adding an Array of Values for a Local Variable 10-11
Deleting Local Variables 10-12
Defining an Expression 10-13
Operator Precedence 10-13
Range Functions 10-14
Using Piecewise Linear Functions in Expressions 10-14
Using Standard Mathematical Functions in Expressions 10-16
Constants Tab (Project Variables Dialog Box) 10-17
Reserved Variable Names 10-17
Defining Parameter Defaults 10-19

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Defining a New Parameter Default by Starting at the Design Level 10-19
Assigning Variables 10-19
Choosing a Variable to Optimize 10-20
Including a Variable in a Sensitivity Analysis 10-21
Choosing a Variable to Tune 10-22
Including a Variable in a Statistical Analysis 10-23
Assigning Parameters to Component Quantities 10-24
11 - Schematic Editor 11-1
Setting Schematic Editor Options 11-1
Schematic Editor Options: General Tab 11-1
Schematic Editor Options: Twin Builder Schematic Tab 11-2
Schematic Editor Options: Wiring Tab 11-2
Schematic Editor Options: Multiple Placement Tab 11-3
Schematic Editor Options: Fonts Tab 11-3
Schematic Editor Options: Colors Tab 11-3
Schematic Editor Options: Symbol Editor Tab 11-4
Starting the Schematic Editor 11-4
The Schematic Editor Window 11-4
Schematic Editor Shortcut Keys 11-5
Grid Setup 11-6
Schematic Page Setup 11-8
Page Borders Tab 11-8
Title Block Tab 11-9
Page Properties and Display Tab 11-10
Zooming and Panning the View 11-11
Zooming In and Out on the Window Contents 11-11
Zooming In on a Rectangular Area 11-12
Restoring a Previous Zoom View 11-12

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Fitting the Drawing to the Window ... 11-12
Fitting the Drawing Border to the Window ... 11-12
Panning the View ... 11-12
Redrawing a View ... 11-13
Closing the Schematic Editor ... 11-13
Creating Simulation Models Using the Schematic Editor ... 11-13
Selecting and Placing Components ... 11-14
Editing Component Parameters ... 11-16
The Parameters Dialog Box Output/Display Tab ... 11-18
Editing Component Properties ... 11-19
Copying and Pasting Properties ... 11-20
To Copy and Paste Selected Properties for Components ... 11-21
To Copy and Paste Common Properties for Primitive Drawing Elements ... 11-23
Favorites and Most Recently Used Components ... 11-25
Operations on Components and Design Objects in the Schematic Editor ... 11-25
Characteristics in Simulation Models ... 11-29
Characteristics in Component Dialog Boxes ... 11-29
Using Pins ... 11-30
Displaying and Hiding Pins ... 11-31
Properties Dialog (Show Pins) ... 11-33
Special Component Dialog (Show Pins) ... 11-34
Names at Pins ... 11-34
Adjusting Symbols and Pins ... 11-34
Connecting Components ... 11-36
Selecting a Wire ... 11-38
Displaying Wire Properties ... 11-38
Checking Connectivity ... 11-39
Removing Unconnected Wires ... 11-39

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Nets, Buses, and Bundles 11-40
Drawing Bus Entry Objects 11-42
Net and Bus Auto-Naming 11-44
Placing Ports 11-45
Interface Ports information 11-46
Placing Interface Ports 11-47
Adding Interface Ports at Pins 11-48
Renaming an Interface Port 11-48
Setting Interface Port Properties 11-49
Global Ports 11-51
Renaming a Global Port 11-52
Page Connector 11-53
Adding Page Connectors at Pins 11-54
Ground Ports 11-55
Selecting a Ground Symbol 11-56
Adding Ground Connectors at Pins 11-57
How Ports Affect Node Names 11-57
Changing Node Names 11-58
The Effect of Deleting Ports 11-59
Creating Twin Models 11-59
Creating and Exporting Co-Simulation FMU Models 11-61
Compile an SML Model from a Schematic 11-65
Finding Elements in the Schematic Editor 11-66
Selecting Elements 11-69
Sorting Components 11-70
Listing Design Elements 11-71
Components Tab 11-73
Graphics Tab 11-73

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ports Tab 11-74
Nets Tab 11-75
Editing Operations 11-75
Adding Primitive Drawing Elements 11-76
Properties of Drawing Elements 11-77
Drawing an Arc 11-77
Drawing a Circle 11-78
Drawing a Line 11-78
Drawing a Polygon 11-78
Drawing a Rectangle 11-79
Adding Text to a Schematic 11-79
Adding an Image to a Schematic 11-80
Drawing Operations 11-80
Adding Reports to a Schematic 11-82
Modifying an On-sheet Report 11-83
Opening an On-sheet report in a New Window 11-84
Setting Up Multi-Page Schematics 11-85
Setting up Hierarchical Designs 11-87
The Bottom-Up Approach 11-87
The Top-Down 11-88
Adding a Subcircuit to a Twin Builder Design 11-89
Adding a Subcircuit 11-90
Wiring Window 11-90
Creating a Subcircuit from a Selection Area 11-92
Copying an Existing Subcircuit within a Design 11-96
Editing Subcircuit Pin Locations 11-97
Moving Between Designs in a Schematic Hierarchy 11-99
Copying a Twin Builder Design into Another Design 11-99

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Working with Design Configurations 11-100
Saving a Design Model Configuration 11-101
Loading a Design Model Configuration 11-101
Editing a Design Model Configuration File 11-102
Design Settings 11-103
Exporting a Schematic 11-105
Printing a Schematic 11-106
12 - Diagram Editor **12-1**
Using the Diagram Editor to Create a Model 12-2
Zooming and Panning the View 12-5
Zooming In and Out on the Window Contents 12-5
Zooming In on a Rectangular Area 12-6
Restoring a Previous Zoom View 12-6
Fitting the Drawing Border to the Window 12-6
Panning the View 12-6
Redrawing a View 12-7
Operations in the Diagram Editor 12-7
Component Libraries 12-8
Project Package 12-9
Variables & Declarations 12-10
Adding a Variable 12-10
Editing a Variable 12-13
Removing a Variable 12-13
Creating a Variable by Dragging a Type Object 12-13
Creating a Variable by Editing the Component Parameter Properties 12-15
Initial Equation 12-16
Adding an Initial Equation 12-16
Editing an Initial Equation 12-18

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Removing an Initial Equation ... 12-18
Equations ... 12-19
Adding an Equation ... 12-19
Editing an Equation ... 12-21
Removing an Equation ... 12-21
Reverting Diagram Editor Changes ... 12-22
Editing Properties ... 12-22
Editing Attributes / Hierarchy ... 12-23
Add Modifier ... 12-24
Add Modifiers ... 12-24
Changing Class ... 12-29
Diagram Object Information ... 12-29
Show Documentation ... 12-29
Show Modelica Text ... 12-29
Deleting a Diagram Editor ... 12-30
Diagram Editor Settings ... 12-30
Updating Modelica Text from Diagram ... 12-33
Updating Modelica Diagram from Text ... 12-34
Updating Modelica Library Definitions ... 12-34
Edit Model Information ... 12-35
Extend a Modelica Model or Diagram ... 12-38
13 - Product Coupling ... 13-1
Simulink® Subcircuits ... 13-3
Program Requirements for MATLAB®/Simulink® Subcircuits ... 13-3
Add Simulink Component ... 13-5
Creating and Linking a MATLAB/Simulink Model ... 13-7
Twin Builder/Simulink Co-Simulation ... 13-9

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Using MathWorks™ Simulink Coder® to Export a Simulink Model to a Twin Builder C-Model DLL ... 13-9
Requirements for Model Creation ... 13-10
Template Files ... 13-11
Workflow ... 13-12
Target ... 13-13
Twin Builder C-Model Target with Dynamic Memory Allocation ... 13-15
Using MathWorks™ Simulink Coder® to Export a Simulink Model as an FMU ... 13-19
Requirements for Model Creation ... 13-20
FMU Target by Ansys Twin Builder ... 13-20
Adding a Dynamic Coupling Component Subcircuit ... 13-22
Python Component Coupling ... 13-27
System Requirements ... 13-27
Overview ... 13-27
Add a New Python Component ... 13-30
Edit or Update a Python Component ... 13-31
Options ... 13-31
Maxwell Coupling Components ... 13-32
Maxwell Transient Cosimulation Component ... 13-32
Transient Cosimulation Component Program Requirements ... 13-32
Transient Cosimulation Component Interface ... 13-33
Maxwell 2D and 3D Finite Element Model Prerequisites ... 13-35
Adding a Maxwell Transient Cosimulation Component ... 13-35
Preparing a Maxwell 2D or 3D Transient model for a link to Twin Builder ... 13-37
How to Use a Maxwell 2D or 3D Finite Element Model in Twin Builder ... 13-39
Continuing a Cosimulation with Maxwell ... 13-39
Maxwell Equivalent Circuit Component ... 13-41
Maxwell Equivalent Circuit Component Program Requirements ... 13-41

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Maxwell Equivalent Circuit Component Interface Concept 13-42
Adding a Maxwell Equivalent Circuit Component 13-42
Preparing a Maxwell Equivalent Circuit Model for Twin Builder 13-44
Linear Motion Equivalent Circuits 13-47
Rotational Motion Equivalent Circuits 13-47
Matrix Equivalent Circuits 13-48
Transformer Equivalent Circuits 13-49
Lookup Table Equivalent Circuits 13-50
How to Use a Maxwell Equivalent Circuit Model in Twin Builder 13-50
Exporting a Maxwell Equivalent Circuit Model 13-51
Maxwell Dynamic Magnetostatic Coupling Component 13-52
Program Requirements for Maxwell Dynamic Magnetostatic Coupling 13-52
Maxwell Dynamic Magnetostatic Coupling Component Interface Concept 13-52
Overview of Maxwell Dynamic Magnetostatic Coupling 13-52
Adding a Maxwell Dynamic Magnetostatic Coupling Component 13-53
Maxwell Dynamic Electrostatic Coupling Components 13-55
Program Requirements for Maxwell Dynamic Electrostatic Coupling 13-55
Maxwell Dynamic Electrostatic Coupling Component Interface Concept 13-56
Overview of Maxwell Dynamic Electrostatic Coupling 13-56
Adding a Maxwell Dynamic Electrostatic Coupling Component 13-56
Maxwell Dynamic Eddy Current Coupling Components 13-58
Program Requirements for Maxwell Dynamic Eddy Current Coupling 13-59
Overview of Maxwell Dynamic Eddy Current Coupling 13-59
Adding a Maxwell Eddy Current Dynamic Coupling Component 13-60
Maxwell Excitations Component 13-62
Adding an Excitations Component 13-62
Sending Data to Maxwell 13-64
RMxprt Dynamic Coupling Component 13-65

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Program Requirements for RMxprt Dynamic 13-65
Overview of RMxprt Dynamic Coupling 13-65
Mechanical Pins for RMxprt Dynamic Coupling Components 13-66
Electrical Pins for RMxprt Dynamic Coupling Components 13-66
Adding an RMxprt Dynamic Coupling Component in Twin Builder 13-66
Example of an RMxprt Dynamic Model in Twin Builder 13-68
Q3D Dynamic Coupling Components 13-68
Q3D Equivalent Circuit Component 13-68
Program Requirements for Q3D Equivalent Circuit Component Coupling 13-68
Q3D Dynamic Equivalent Circuit Coupling Component Interface Concept 13-69
Overview of Q3D Dynamic Equivalent Circuit Component Coupling 13-69
Adding a Q3D Equivalent Circuit Coupling Component 13-70
Q3D State Space Components 13-71
Program Requirements for Q3D State Space Component Coupling 13-72
Overview of Q3D State Space Coupling 13-72
Adding a Q3D State Space Coupling Component 13-73
PExprt Static Coupling Components 13-75
Program Requirements for PExprt Static Component Coupling 13-75
Overview of PExprt Static Component Coupling 13-75
Ansys Mechanical Components 13-76
Program Requirements for Ansys Mechanical Component Coupling 13-76
Ansys Mechanical Component Interface Concept 13-76
Overview of Ansys Mechanical Component Coupling Workflow 13-77
Adding an Ansys Mechanical Component 13-78
Ansys Mechanical Link Example 13-80
File Link Dialog Box 13-81
Icepak Components 13-83
Program Requirements for Ansys Icepak Component Coupling 13-83

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ansys Icepak Component Interface Concept 13-83
Overview of Ansys Icepak Component Coupling Workflow 13-89
Adding an Ansys Icepak Component Subcircuit 13-89
State-Space Components 13-97
Program Requirements for State-Space Components 13-97
Overview of State-Space Component Coupling 13-97
Generate State-Space Component Dialog Box 13-98
Adding a State-Space Component 13-100
Viewing and Editing State-Space Component Matrix Values 13-102
Changing or Moving Coupling Model Files 13-102
SIwave Components 13-103
Program Requirements for SIwave Component Coupling 13-103
Overview of SIwave Component Coupling 13-104
HFSS Components 13-104
Program Requirements for HFSS Component Coupling 13-105
Overview of HFSS Component Coupling 13-105
Adding an HFSS Dynamic Coupling Component in Twin Builder 13-106
Push Excitations for HFSS and SIwave Components 13-107
ANSYS RBD Components 13-108
Ansys Fluent LTI Components 13-109
Program Requirements for Ansys Fluent LTI Component Coupling 13-109
Overview of Ansys Fluent LTI Component Coupling 13-109
Motor-CAD ROM Components 13-111
ProgramRequirementsforMotorCADROMCoupling 13-111
Overview of Motor-CAD ROM Components 13-111
Add a New Motor-CAD ROM Component 13-112
Edit or Update a Motor-CAD ROM Component 13-114
Ansys SCADE Components 13-114

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Program Requirements for SCADE Component Coupling 13-114
FMU Components 13-114
Program Requirements for FMU Component Coupling 13-115
Overview of FMU Coupling with Twin Builder 13-115
Stand Alone ROM Viewer 13-123
Introduction 13-123
Using Stand Alone ROM Viewer 13-123
Input Data 13-123
Launching Stand Alone ROM Viewer 13-124
Graphic Interface Organization 13-125
Opening a Case (Project Toolbar) 13-126
Consuming a Static ROM 13-126
Consuming a Dynamic ROM 13-127
Opening One Snapshot File 13-128
Opening a Folder of Snapshots 13-128
Opening Snapshots Using a DoE File (Static ROM) 13-129
Viewer 13-130
Ansys Examples 13-132
Ansys Example 1: Static Fluent Case 13-132
Ansys Example 2: Dynamic Fluent Case 13-134
Static ROM Builder 13-136
Dynamic ROM Builder 13-137
Data Connector Component 13-138
Program Requirements for Data Connector Component 13-138
Overview of Data Connector 13-138
Adding a Data Connector Component in Twin Builder 13-140
14 - Netlist Editor 14-1
Viewing Netlists in Twin Builder 14-1

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Netlist Editor Operation 14-2
Netlist Editor Tab 14-3
Edit Menu 14-4
Using Bookmarks 14-4
Inserting a Bookmark 14-5
Deleting a Bookmark 14-5
Deleting All Bookmarks 14-5
Moving Forward to the Next Bookmark 14-5
Moving Backward to the Previous Bookmark 14-6
Searching the Netlist 14-6
Searching from the tab 14-6
Searching from the Edit Menu or by Keyboard Shortcut 14-7
Searching for and Replacing Text 14-7
Going to a Numbered Line 14-8
Exporting a Netlist or Script 14-8
Netlist & Script Editor Settings 14-9
15 - Ansys Workbench Integration Overview 15-1
Integrating Ansys Electromagnetics Suite Products with Ansys Workbench 15-2
Integration with Ansys 2024 R2 during Ansys Electromagnetics Product Installation 15-2
Integration with Ansys 2024 R2 after Ansys Electromagnetics Product Installation 15-2
Workbench Data Integration Overview 15-3
Adding New Ansys Electromagnetics Analysis Systems 15-5
Importing Ansys Electromagnetics Projects into Ansys Workbench 15-6
Editing Ansys Electromagnetics Models in Workbench 15-7
Analyzing Ansys Electromagnetics Models in Workbench 15-7
Performing Parameter Studies in Workbench 15-8
Scripting in Workbench 15-10
Ansys Electromagnetics - Ansys Multiphysics Coupling 15-10

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Multiphysics Coupling on Workbench with Ansys Thermal 15-10
Multiphysics Coupling on Workbench with Ansys Structural 15-11
Multiphysics Coupling between Ansys Electromagnetics Field Systems on Workbench 15-12
Ansys Electromagnetics CAD Integration through Workbench 15-13
CAD Integration and Geometry Sharing 15-16
CAD Integration Functionality 15-18
Bi-Directional CAD Integration 15-20
CAD Integration Model Edits 15-22
Multiple Geometry Links for CAD Integration 15-23
Healing with CAD Integration 15-24
Important Geometry Options for CAD Integration 15-25
User Defined Model (UDM) for Ansys WB Integration 15-27
UDM Compared to User Defined Primitives 15-28
Insert UDM Command on Draw Menu 15-30
UDM Properties 15-31
UDM Parameters 15-32
UDM Part Edits 15-33
Library of Models for CAD Integration 15-33
Ansys Electromagnetics to Ansys Geometry Transfer 15-35
CAD Integration Material Assignment Transfer 15-37
Geometry Transfer through Ansys DesignModeler (DM) 15-38
CAD Integration Functionality 15-39
16 - Analyzing Twin Builder Designs 16-1
Simulator Backplane 16-1
Circuit Simulator Processing 16-2
Block Diagram Simulator Processing 16-2
State Graph Simulator Processing 16-3

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

VHDL-AMS Simulation ... 16-3
Twin Builder Analyses ... 16-4
Standard Analysis Types ... 16-4
Advanced Analysis Types - Optimetrics ... 16-5
Standard Analysis Setup Options ... 16-5
Transient Analysis Setup ... 16-5
Guidelines for the Proper Choice of Time Step ... 16-6
Adding a Transient Analysis ... 16-7
AC Analysis Setup ... 16-9
Guidelines for Configuring AC Simulation Models ... 16-11
Adding an AC Analysis ... 16-12
DC Analysis Setup Options ... 16-13
Guidelines for Configuring DC Simulation Models ... 16-15
Adding a DC Analysis ... 16-15
Viewing DC Bias Values in a Schematic ... 16-16
Select Solution Options ... 16-19
Setting the Active Analysis Setup ... 16-19
Disabling/Enabling an Analysis Setup ... 16-20
Editing an Analysis Setup ... 16-20
Copying and Pasting an Analysis Setup ... 16-21
Renaming an Analysis Setup ... 16-21
Deleting an Analysis Setup ... 16-22
Solution Options ... 16-22
TR (Transient) Options ... 16-23
AC Options ... 16-27
DC Options ... 16-27
General Options ... 16-28
Damping Heuristics ... 16-28

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Data Reduction ... 16-29
Simulator Pivoting Strategy ... 16-29
Advanced Logging Info ... 16-29
Using the Simulator Performance Slider ... 16-30
SML Header Options ... 16-31
Adding Solution Options ... 16-32
Editing Solution Options ... 16-33
Copying and Pasting Solution Options ... 16-33
Renaming Solution Options ... 16-34
Deleting Solution Options ... 16-34
Importing Solution Data (.sdb format) ... 16-35
Importing a Solution Data File (non-.sdb format) ... 16-35
File Formats ... 16-37
MDX File Format ... 16-38
Data Set Parameters ... 16-39
CSV File Format ... 16-43
XLS, XLSX: Microsoft Excel Format ... 16-44
COMTRADE File Format ... 16-45
Setting the Outputs for Simulation ... 16-45
Creating Reports ... 16-46
Checking the Design Size ... 16-47
17 - Running Simulations ... 17-1
Solving a Single Setup ... 17-1
Running More Than One Simulation ... 17-2
To solve every solution setup in a design: ... 17-3
To solve every solution setup in a project: ... 17-3
Parametric Analysis of Simulation Models ... 17-3
Using Replay Analysis ... 17-4

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To manually select replay start time: 17-6
Remote Analysis 17-7
Troubleshooting 17-10
Remote Solve Node = Windows 17-10
Distributed Analysis 17-11
Configuring Distributed Analysis 17-11
Editing Distributed Machine Configurations 17-13
Selecting an Optimal Configuration for Distributed Analysis 17-17
Monitoring and Controlling the Solution Process 17-17
Twin Builder Simulation Icons 17-19
Progress Bar Menu 17-19
Monitoring Solution Progress 17-21
Modifying Parameters During a Simulation 17-21
Monitoring Queued Simulations 17-24
Large Scale DSO for Parametric Analysis 17-24
Prerequisites for Large Scale DSO 17-25
Job Management Interface for Large Scale DSO 17-26
Preparing a Project for Large Scale DSO Analysis 17-34
Large Scale DSO Command Line Syntax 17-36
Large Scale DSO Job outputs 17-38
Import Large Scale DSO Dataset Solution 17-38
Viewing and Deleting Created Dataset Solutions 17-39
Creating a Dataset Report from Imported DSO Solutions 17-40
Cloning a Report from a Dataset Solution 17-42
Large Scale DSO Results Database Organization 17-42
Large Scale DSO Job Monitoring 17-43
Large Scale DSO Deployment/Configuration 17-44
Known Issues for Large Scale DSO 17-45

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Troubleshooting for Large Scale DSO ... 17-45
Job monitoring and control ... 17-45
Job outcome ... 17-46
Large Scale DSO Theory ... 17-46
High Performance Computing (HPC) Integration ... 17-49
Scheduler Terminology ... 17-52
What a Scheduler Does ... 17-52
Installation of Ansys Electromagnetics Suite ... 17-53
Firewall Configuration ... 17-54
Installation Directory Examples ... 17-54
Ansys Electromagnetics Jobs ... 17-55
Job Submission Scripting ... 17-55
Integration with Microsoft Windows® HPC Scheduler ... 17-57
Submitting and Monitoring HPC Jobs ... 17-59
Submitting and Monitoring Jobs for Windows HPC ... 17-61
Windows® HPC Job Templates ... 17-79
Selecting Computation Resource Units (Job Unit Type) ... 17-79
Windows® HPC Job Credentials ... 17-80
Command Line Information for Ansys Electromagnetics Desktop Products ... 17-81
Integrating Ansys Electromagnetics Suite with Third Party Schedulers ... 17-83
Build Information for Scheduler Proxy Library ... 17-83
Implementation Details for Custom Scheduler Integration ... 17-84
IsProductLaunchedInYourEnvironment ... 17-84
GetTempDirectory ... 17-85
GetMachineListAvailableForDistribution ... 17-85
LaunchProcess ... 17-86
GetUseRsmForEngineLaunch ... 17-87
GetThisJobID ... 17-88

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetSchedulerDisplayName 17-89
Scheduler Proxy Interfaces 17-90
Testing Your Scheduler Integration 17-96
Testing IsProductLaunchedInYourEnvironment 17-96
Testing GetSchedulerDisplayName and GetThisJobID 17-96
Testing GetTempDirectory 17-97
Testing GetMachineListAvailableForDistribution 17-97
Testing LaunchProcess 17-97
Testing GetUseRsmForEngineLaunch 17-98
Troubleshooting Custom Scheduler Integration 17-98
None of the Proxy Functions are Called 17-98
Troubleshooting IsProductLaunchedInYourEnvironment Function 17-99
Troubleshooting GetSchedulerDisplayName 17-99
Troubleshooting GetThisJobID 17-99
Troubleshooting GetTempDirectory 17-100
Troubleshooting GetMachineListAvailableForDistribution 17-100
Troubleshooting LaunchProcess 17-100
Troubleshooting GetUseRsmForEngineLaunch 17-101
RSM Integration with Job Management UI 17-101
Re-solving a Problem 17-111
Resetting Symbol Animation 17-112
18 - Network Data Explorer 18-1
Network Data Explorer Overview 18-3
NDE Ribbon 18-4
Network Data Selection Pane 18-5
Cell and Frequency Selection Pane 18-6
Data View Pane 18-6
Loading Data Into Network Data Explorer 18-7

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Exporting Data from Network Data Explorer 18-8
Exporting SYZ Data 18-8
Exporting Macro Model 18-10
Miscellaneous Options 18-14
Advanced Options 18-14
Comparing Original S-Parameters with Exported S-Parameters 18-16
Creating a Twin Builder Component 18-16
Scripting for Network Data Explorer 18-17
Network Data Explorer Ribbon 18-17
Data Sources 18-17
Set Display Format 18-18
Display Full Port Names 18-19
Save or Reset Default Settings 18-20
Smoothing 18-20
Cell Filtering 18-22
Changing Port Properties and Reducing Matrix Size 18-23
Displaying Mixed-Mode Parameters Using Differential Pairs 18-24
Reset All Port Properties 18-26
Right-Click Context Menu Options 18-26
Display Format 18-26
Multiple Frequency Statistics 18-27
Highlight Min/Max 18-28
Select Transpose 18-28
Full Port Names 18-28
Exploring Network Data and Modifying the Display 18-28
Viewing the S, Y, or Z Matrix for a Selected Frequency 18-28
Color-Coded Matrix Plot 18-30
Changing the Color Scheme for a Matrix Color Plot 18-31

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Displaying a Graph of a Cell Across All Frequencies ... 18-32
Viewing Matrix Cell Data Across All Frequencies ... 18-33
Displaying Statistics by Frequency ... 18-33
Displaying Individual Statistics for All Frequencies ... 18-34
Creating a Statistics Plot ... 18-36
Comparing Network Data ... 18-37
Comparing Variations ... 18-37
Displaying Plot Traces from Multiple Data Sources ... 18-38
Causality Checking and Plots ... 18-38
Touchstone Calibration Wizard ... 18-45
Two-Line Method ... 18-46
Thru-Reflect-Line Method ... 18-46
Short-Open-Load-Thru Method ... 18-47
Multithreading ... 18-48
Multithreading Technical Notes ... 18-48
19 - Optimetrics ... 19-1
Parametric Overview ... 19-2
Setting Up a Parametric Analysis ... 19-3
Adding a Variable Sweep Definition ... 19-4
Specifying Variable Values for a Sweep Definition ... 19-6
Synchronizing Variable Sweep Definitions ... 19-7
Modifying a Variable Sweep Definition Manually ... 19-7
Overriding a Variable's Current Value in a Parametric Setup ... 19-8
Specifying a Solution Setup for a Parametric Setup ... 19-9
Specifying the Solution Quantity to Evaluate for Parametric Analysis ... 19-9
Setup Calculations for Optimetrics ... 19-10
Specifying a Solution Quantity's Calculation Range ... 19-12
Viewing Results for Parametric Solution Quantities ... 19-12

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding a Parametric Sweep from a File 19-13
Optimization Overview 19-14
Choosing an Optimizer 19-14
Quasi-Newton (Gradient) 19-16
Pattern Search (Search-based) 19-19
Sequential Non-linear Programming (Gradient) (SNLP) 19-22
Sequential Mixed Integer NonLinear Programming (Gradient and Discrete) 19-23
Genetic Algorithm (Random search) 19-23
MATLAB optimizer 19-24
Screening (Shifted Hammersley Sampling) Optimization 19-30
Multi-Objective Genetic Algorithm (MOGA) 19-31
Convergence Criteria in MOGA-Based Multi-Objective Optimization 19-36
Setting Up Multi-Objective Genetic Algorithm (Random Search) Optimizer 19-38
Convergence Rate % and Initial Finite Difference Delta % in NLPQ and MISQP 19-43
Mixed-Integer Sequential Quadratic Programming 19-45
Adaptive Multiple Objective Optimization 19-46
Adaptive Single Objective Optimization 19-49
Optimization Variables and the Design Space 19-53
Setting Up an Optimization Analysis 19-54
Optimization Setup for the Quasi-Newton (Gradient) Optimizer 19-56
Optimization Setup for the Pattern Search (Search-based) Optimizer 19-57
Optimization Setup for the Merit-based Sequential Quadratic Programming (Gradient) Optimizer 19-58
Optimization Setup for the Sequential Nonlinear Programming (Gradient) Optimizer 19-59
Optimization Setup for the Sequential Mixed Integer Nonlinear Programming (Gradient and Discrete) Optimizer 19-61
Optimization Setup for the Genetic Algorithm (Random Search) Optimizer 19-62
Optimization Setup for the MATLAB Optimizer 19-63

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Setting Up Screening (Search Based) Optimizer 19-65
Setting Up Multi-Objective Genetic Algorithm (Random Search) Optimizer 19-68
Setting Up Nonlinear Programming by Quadratic Lagrangian (Gradient) Optimizer 19-73
Setting Up Mixed-Integer Sequential Quadratic Programming (Gradient and Discrete) Optimizer 19-78
Setting Up Adaptive Multiple-Objective (Random Search) Optimizer 19-82
Setting Up Adaptive Single-Objective (Gradient) Optimizer 19-87
Setting the Maximum Iterations for an Optimization Analysis 19-92
Cost Function 19-93
Acceptable Cost 19-93
Cost Function Noise 19-94
Adding a Cost Function 19-94
Adding/Editing a Cost Function Calculation 19-96
Specifying a Solution Quantity for a Cost Function Goal 19-97
Setting the Calculation Range of a Cost Function Goal 19-98
Setting a Goal Value 19-98
Specifying a Single Goal Value 19-99
Specifying an Expression as a Goal Value 19-99
Specifying a Variable-Dependent Goal Value 19-99
Goal Weight 19-100
Modifying the Starting Variable Value for Optimization 19-101
Setting the Minimum and Maximum Variable Values for Optimization 19-102
Text Entry for Calc. Range or Edit Calculation Range Dialog 19-103
Overriding the Min. and Max. Variable Values for a Single Optimization Setup 19-105
Changing the Min. and Max. Variable Values for Every Optimization Setup 19-105
Step Size 19-106
Setting the Min. and Max. Step Sizes 19-107
Setting the Min and Max Focus 19-108

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Equalizing the influence of different optimization variables 19-109
To set the Min and Max Focus values 19-109
Solving a Parametric Setup Before an Optimization 19-109
Solving a Parametric Setup During an Optimization 19-110
Automatically Updating a Variable's Value after Optimization 19-110
Changing the Cost Function Norm 19-110
Explanation of L1, L2 and Max norms in Optimization 19-111
Advanced Genetic Algorithm Optimizer Options 19-113
Sensitivity Analysis Overview 19-116
Selecting a Master Output 19-117
Setting Up a Sensitivity Analysis 19-117
Setting the Maximum Iterations Per Variable 19-119
Setting Up an Output Parameter 19-119
Specifying a Solution Quantity for an Output Parameter 19-120
Setting the Calculation Range of an Output Parameter 19-121
Modifying the Starting Variable Value for Sensitivity Analysis 19-122
Setting the Min. and Max. Variable Values 19-123
Overriding the Min. and Max. Variable Values for a Single Sensitivity Setup 19-123
Changing the Min. and Max. Variable Values for Every Sensitivity Setup 19-124
Setting the Initial Displacement 19-124
Solving a Parametric Setup Before a Sensitivity Analysis 19-124
Solving a Parametric Setup during a Sensitivity Analysis 19-125
Performing Worst Case Analysis 19-125
Example of a Worst Case Analysis 19-129
Statistical Analysis Overview 19-134
Setting Up a Statistical Analysis 19-134
Statistical Analysis with Netlist Components 19-136
Setting the Maximum Iterations for a Statistical Analysis 19-136

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Specifying an Initial Seed Value for a Statistical Analysis 19-136
Specifying the Solution Quantity to Evaluate for Statistical Analysis 19-137
Setting the Solution Quantity's Calculation Range 19-138
Setting the Distribution Criteria 19-139
Overriding the Distribution Criteria for a Single Statistical Setup 19-139
Changing the Distribution Criteria for Every Statistical Setup 19-140
Statistical Cutoffs 19-141
Edit Distribution 19-142
Modifying the Starting Variable Value for Statistical Analysis 19-143
Solving a Parametric Setup during a Statistical Analysis 19-144
Using Design of Experiments 19-145
Setting Up Design of Experiments 19-147
Design of Experiments Tab 19-150
Optimal Space Filling Design (OSF) 19-152
Central Composite Design (CCD) 19-154
Box-Behnken Design (CCD) 19-155
Custom DOE Type 19-156
Latin Hypercube Sampling 19-156
Table Tab for Design of Experiments 19-157
Response Surface Tab for Design of Experiments 19-159
Variables Tab for Design of Experiments 19-160
View Analysis Result for Design of Experiments 19-162
Min-Max Search for Design of Experiments 19-166
Refinement Points Table 19-167
Performing a Manual Refinement 19-167
Response Points Table 19-168
Verification Points Table 19-169
Goodness of Fit (Predicted vs Observed Chart) 19-170

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Response Surface Results Design of Experiments Result 19-171
Using the Fast Calculation-Update Algorithm 19-174
Fast Calculation-Update Algorithm Limitations 19-176
optiSLang Integration with Electronics Desktop 19-176
Prerequisites 19-178
optiSLang User Workflow 19-178
Parametrization for optiSLang Integration 19-181
Inputs 19-181
Outputs 19-182
Parameterization with optiSLang Analysis Goals in Mind 19-182
Results and Reports for optiSLang Integration 19-183
Creating an optiSLang Setup in AEDT 19-185
Setting up Calculations 19-186
Setting Options 19-188
Finishing Setup 19-189
Example optiSLang Setups 19-190
Example 1: Solving a User-created, Pre-existing optiSLang Setup 19-190
Example 2: Increasing Performance by Avoiding Wasteful Solution of the Nominal Design 19-192
Example 3: Avoiding Redundant Computations while Exploiting AEDT Features around Postprocessing Variables 19-193
Example 4: Parametrization of an AEDT Design for optiSLang Integration 19-193
Solving an optiSLang Setup 19-193
Considerations for Using the Analyze All Command 19-194
Considerations for Distributed Analysis 19-194
optiSLang Menu Options in AEDT 19-195
Viewing optiSLang Postprocessing Results 19-196
Tuning Overview 19-198

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Tuning a Variable 19-198
Applying a Tuned State to a Design 19-199
Saving a Tuned State 19-200
Reverting to a Saved Tuned State 19-200
Resetting Variable Values after Tuning 19-200
Adding an Expression in the Output Variables Window 19-201
Excluding a Variable from an Optimetrics Analysis 19-201
Modifying the Value of a Fixed Variable 19-202
Linear Constraints 19-202
Setting a Linear Constraint 19-203
Modifying a Linear Constraint 19-204
Deleting a Linear Constraint 19-204
Running an Optimetrics Analysis 19-204
Viewing Analysis Results for Optimetrics Solutions 19-205
Viewing an Optimetrics Solution's Profile Data 19-206
Plotting Solution Quantity Results vs. a Swept Variable 19-206
Viewing Cost Results for an Optimization Analysis 19-207
Plotting Cost Results for an Optimization Analysis 19-207
Viewing Output Parameter Results for a Sensitivity Analysis 19-208
Plotting Output Parameter Results for a Sensitivity Analysis 19-208
Viewing Distribution Results for a Statistical Analysis 19-209
Plotting Distribution Results for a Statistical Analysis 19-209
Link to DesignXplorer 19-210
20 - Generating Reports and Postprocessing 20-1
The Report Dialog Box 20-1
Context Section: 20-2
Trace Tab 20-3
Families Tab: 20-4

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Families Display Tab ... 20-4
Update Report ... 20-5
Report Dialog Box Buttons ... 20-5
Plotting Spectral Domain Data ... 20-6
The Select Quantities Dialog Box ... 20-10
Creating a New Report ... 20-11
Creating a Report from a Report Data File ... 20-14
Modifying Reports ... 20-15
Showing and Hiding Active View Objects ... 20-16
Modifying the Background Properties of a Report ... 20-16
Modifying the Legend in a Report ... 20-19
Spinning a 3D Report ... 20-20
Working with the X-Axis Scrollbar (Rectangular, Stacked, Bode, and Nyquist Plots) 20-20
Selecting the Display Type ... 20-22
Creating Rectangular Plots ... 20-23
Creating Polar Plots ... 20-25
Reviewing 2D Polar Plots ... 20-27
Creating a Data Table or Numeric Display ... 20-27
Working with Data Tables ... 20-29
Creating 3D Rectangular Plots ... 20-31
Creating a 3D Rectangular Plot ... 20-32
Controlling Visual Detail in a 3D Rectangular Plot ... 20-33
Creating a Rectangular Stacked Plot ... 20-39
Creating a Rectangular Stacked Plot ... 20-41
Multiple Curves in a Stack in Cartesian Stacked Plots ... 20-42
Automatic Grouping ... 20-42
Manual Grouping ... 20-43
Manual Grouping through the Stacked Property Tab ... 20-45

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Manual Grouping Through the Context Menu 20-46
Legend Optimizations 20-47
Y-Markers 20-47
Creating a Rectangular Contour Plot 20-48
Creating a Bode Plot 20-48
Creating a Nyquist Plot 20-50
Creating a Digital Plot 20-52
Creating a Plot-On-Schematic 20-52
Probe Command 20-53
Quick Probe Command 20-54
Setting Report Setup Options 20-54
Setting Report2D Options 20-55
Report 2D Options: Curve Tab 20-57
Report2D Options: Axis Tab 20-57
Report2D Options: Grid Tab 20-57
Report2D Options: Header Tab 20-58
Report2D Options: Note Tab 20-58
Report2D Options: Legend Tab 20-58
Report2D Options: Marker tab 20-59
Report2D Options: Marker Table Tab 20-59
Report 2D Options: X-Y Markers Tab 20-60
Report2D Options: Stacked 20-61
Report2D Options: Digital Tab 20-61
Report2D Options: General Tab 20-62
Report2D Options: Table Tab 20-62
Working with Traces 20-63
Editing Trace Properties 20-65
Editing the Display Properties of Traces 20-66

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding Data Markers to Traces ... 20-69
Y Markers in stacked XY plots ... 20-71
Delta Markers in 2D Reports ... 20-79
Discarding Report Values Below a Specified Threshold ... 20-79
Adding Characteristics to a Trace ... 20-79
Adding a Recently Used Trace Characteristic ... 20-80
Adding a Trace Characteristic from Favorites ... 20-80
Adding Trace Characteristics to your Favorites ... 20-81
Adding Characteristics using the Add Trace Characteristics Dialog Box ... 20-82
Removing All Trace Characteristics ... 20-86
Defining Traces Using Range Functions ... 20-86
Copy and Paste of Report and Trace Definitions ... 20-96
Copy and Paste of Report and Trace Data ... 20-97
Removing Traces ... 20-98
Limit Lines in Cartesian Plots ... 20-98
Specifying Points to Create Limit Lines ... 20-99
Creating Limit Lines from a Selected Curve ... 20-100
Editing Limit Lines ... 20-102
Variables, Quantities and Functions ... 20-105
Sweeping a Variable in a Report ... 20-105
Selecting a Function ... 20-106
Selecting Solution Quantities to Plot ... 20-108
Plotting Imported Solution Data ... 20-108
Setting a Range Function ... 20-109
Specifying Output Variables ... 20-110
Adding a New Output Variable ... 20-110
Building an Expression Using Existing Quantities ... 20-111
Deleting Output Variables ... 20-112

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Report Data 20-112
Updating Reports (Post-Processing Data) 20-113
Deleting Reports 20-114
Opening Reports 20-114
Exporting Plot Data 20-114
Importing 2D Plot Data 20-115
Report File Formats 20-115
Creating Custom Report Templates and Defaults 20-117
Exporting Graphics Files from a Plot 20-118
Copying Reports to the Clipboard as Images 20-118
Plotting Imported Solution Data 20-119
Using Animation 20-119
Frequency Animation 20-119
Changing the Design Point 20-122
User Defined Outputs: Introduction 20-122
Named Probes and Properties in User Defined Outputs 20-123
Computation of Traces Based UDO Calculations 20-124
Dimensions Reduction by UDO Calculations 20-125
Dynamic Probes 20-125
User Defined Outputs: Python Script API 20-125
UDO Extension IMPLEMENTATION 20-126
Import Statements 20-126
DOExtension Class 20-126
IUDOPluginExtension Abstract Class 20-127
Required functions: 20-127
GetUDSName() 20-127
GetUDSDescription() 20-128
GetUDSSweepNames() 20-128

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetCategoryNames() 20-128
GetQuantityNames(string categoryName) 20-128
GetQuantityInfo(string quantityName) 20-129
GetInputUDSParams(List<UDSProbeParams> udsParams, ... 20-130
IPropertyList propList, ... 20-130
List<UDSProbeParams> userSelectionForDynamicProbes) 20-130
GetDynamicProbes(List<UDSDynamicProbes> dynamicProbes); 20-133
Compute(IUDSInputData inData 20-133
IUDSOutputData outData, ... 20-133
IPropertyList propList, ... 20-133
IProgressMonitor progressMonitor) 20-133
Optional Functions in IDO Extension Abstract Class 20-137
Validate(List<string> errorStringList, ... 20-137
Data Types Used in Python Script 20-138
IUDSInputData 20-140
GetDoubleProbeData(probeName) 20-141
GetSweepsDataForProbe(probeName, sweepName) 20-141
GetComplexProbeData(probeName) 20-141
GetSweepNamesForProbe(probeName) 20-142
GetRequiredQuantities() 20-142
GetVariableValues() 20-143
GetInterpolationOrdersData(probeName); 20-143
IUDSOutputData 20-143
SetSweepsData(sweepName, sweepData) 20-144
SetDoubleQuantityData(qtyName,qtyData) 20-144
SetComplexQuantityData(qtyName, qtyData) 20-144
Working With Properties for UDO 20-145
IPropertyList Abstract class 20-145

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

IProperty Abstract class ... 20-146
INumberProperty Abstract class ... 20-146
ITextProperty Abstract class ... 20-147
IMenuProperty Abstract class ... 20-147
Other Application Specific Classes Used in Python Scripts ... 20-148
Constants Class ... 20-148
UDSProbeParams Class ... 20-148
UDSDynamicProbes Class ... 20-149
QuantityInfo Class ... 20-150
IProgressMonitor Abstract Class ... 20-150
Using .NET Collection Classes and Interfaces ... 20-151
User Defined Outputs: Messaging Methods ... 20-153
User-Defined Outputs: Script Organization ... 20-160
Using Script Libraries ... 20-160
Using Additional .NET Assemblies ... 20-160
Toolkit ... 20-161
Battery Wizard ... 20-161
LPV Wizard ... 20-163
Motor Equivalent Circuit ... 20-166
Response Surface ROM ... 20-168
System Model Identification MIMO ... 20-170
System Model Identification SISO ... 20-175
Thermal Model Identification ... 20-179
Response/Input Generation Files from Ansys 3D Products ... 20-184
User Defined Documents (UDDs) ... 20-187
Managing Documents Listed in the Project Window under Results ... 20-193
Documents folder shortcut menu ... 20-193
Document folder Property window ... 20-193

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Viewing UDDs with an HTML Web Browser ... 20-194
UDD Script Libraries ... 20-195
User Defined Definitions: Python Script API ... 20-196
Data Types Used in Python Script ... 20-204
UDD Input interfaces ... 20-206
User Defined Document Scripting Interface ... 20-208
The UserDefinedDocument Data Format in the Script ... 20-210
Python Script to Define Document ... 20-212
Sample Script ... 20-214
Document Generator Interfaces ... 20-216
Chip Model Analyzer (CMA) ... 20-222
PinToPin Utility ... 20-222
21 - C-Models in Twin Builder ... 21-1
Introduction to the Twin Builder C Interface ... 21-1
Twin Builder C Interface Overview ... 21-1
Program Requirements ... 21-2
Installation ... 21-3
Designing a C-Model ... 21-3
Defining Model Equations ... 21-3
Transient (TR) Equations ... 21-5
DC Equations ... 21-6
AC Equations ... 21-7
Implementing C-Models ... 21-8
Microsoft Visual Studio Project Environment ... 21-9
Defining C-Models in Microsoft® Visual Studio ... 21-9
Modifying C-Models in Microsoft Visual Studio ... 21-10
Debugging C-Models in Microsoft Visual Studio ... 21-11
Programming C-Models ... 21-12

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Basic Steps 21-12
Include Twin Builder-specific Header Files 21-13
Define a Model Name 21-13
Define Inputs/Outputs Using the Prepare Function 21-14
Define the PISVC Functions 21-15
Define Model Kind and Number Using RegisterUserModel 21-18
Define Model-dependent Functions Using RegisterUMODELFct 21-19
Implementing for Transient Simulations (TR) 21-20
TR Initialize Function 21-20
TR Simulate Function 21-22
TR Validate Function 21-24
TR Close Function 21-24
Using Internal States 21-25
Implementing for Operating Point Analysis (DC) 21-26
DC Initialize Function 21-26
DC Simulate Function 21-27
DC Validate Function 21-27
DC Close Function 21-28
Implementing for Spectral Analysis (AC) 21-28
AC Initialize Function 21-29
AC Simulate Function 21-30
AC Validate function 21-30
AC Close function 21-31
C-Models with Specified Sample Time 21-31
C-Models Used as Characteristics 21-32
Using C-Models in Twin Builder 21-35
Updating C-Models 21-35
C-Models in Schematics 21-36

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

C-Models in SML Description ... 21-36
C/C++ Function Reference ... 21-36
Data Types ... 21-36
CUModDecl Object ... 21-37
CModUser Object Methods ... 21-40
CModUser object methods grouped by application area ... 21-40
Alphabetical List of CModUser object methods ... 21-42
AddAvailableAnalysisType ... 21-42
AddNode_c ... 21-43
AddNode_nc ... 21-44
AddNode_State ... 21-44
AddSubNode_nc ... 21-45
AllocateUserDataMemory ... 21-45
GetAnalysisType ... 21-46
GetCplxSVVal ... 21-46
GetDataTypeNode_nc ... 21-47
GetDefName ... 21-48
GetDCUserData ... 21-48
GetDSVVal ... 21-49
GetHierName ... 21-49
GetNode_ncParam ... 21-49
GetSVVal ... 21-50
GetUseName ... 21-50
GetUserData ... 21-51
GetValNode_nc ... 21-51
GetValNode_nc (for use with characteristics) ... 21-52
GetValNode_ncCplx ... 21-53
GetValNode_ncFile ... 21-53

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetValNode_ncStrg 21-54
GetValNode_State 21-54
GetValNode_StateCplx 21-55
GetValSubNode_nc 21-55
IsCharConn 21-56
IsParamSetFlag 21-56
IsTransientOP 21-57
SetAsThreadSafe 21-58
SetCplxGSEntry 21-58
SetCplxRSEntry 21-59
SetDataTypeNode_nc 21-59
SetInfoNode__c 21-60
SetInfoNode_nc 21-61
SetNatureTypeNode__c 21-61
SetRealGSEntry 21-62
SetRealRSEntry 21-63
SetSVVal 21-63
SetSymbolicGSEntry 21-64
SetUModCloseFct 21-65
SetUModInitFct 21-65
SetUModPrepFct 21-66
SetUModSimFct 21-66
SetUModValidFct 21-67
SetUnitNameNode 21-68
SetUserData 21-68
SetValNode_nc 21-69
SetValNode_ncFile 21-70
SetValNode_State 21-71

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SetValPtrNode_nc 21-71
SetValSubNode_nc 21-72
Callback Functions 21-73
CHAR_IN 21-73
CHAR_OUT 21-73
CHAR_OUT_DERIVE 21-73
GET_SAMPLETIME 21-74
GetInfo 21-74
getPATH 21-75
ISIM_BASE 21-75
ISIM_BASE_MAX 21-76
ISIM_BASE_MIN 21-76
ISIM_BASE_STEP 21-76
ISIM_BASE_TEMP 21-76
ISIM_BASE_XEND 21-77
ISIM_BASE_XSTART 21-77
ISIM_ECM_D_DT 21-77
ISIM_ECM_IEMAX 21-77
ISIM_ECM_ITERAT 21-78
ISIM_ECM_LDF 21-78
ISIM_ECM_NEW 21-78
ISIM_ECM_P 21-78
ISIM_ECM_RELTOL 21-79
ISIM_ECM_SOLVER 21-79
ISIM_ECM_UEMAX 21-79
OSIM_ECM_REJECT 21-79
OSIM_ECM_SYNC 21-79
OSIM_SYNC 21-80

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Report2Sim 21-80
SET_SAMPLETIME 21-81
22 - VHDL-AMS Models in Twin Builder 22-1
Understanding VHDL-AMS Models 22-1
Across and Through Quantities of Natures 22-2
Packages and Models in Libraries 22-3
Entities and Architectures of VHDL-AMS Models 22-3
Creating and Editing VHDL-AMS Models 22-4
Placing and Connecting VHDL-AMS Models 22-5
Using Transformation Models 22-5
Defining Model Properties 22-6
VHDL-AMS Language Fundamentals 22-6
Design Units 22-7
Entities and Architectures 22-8
Entity Declaration 22-8
Architecture 22-9
Packages 22-10
Package Declaration 22-10
Package Body 22-11
Package Visibility 22-12
VHDL-AMS Standard Packages and Types 22-12
STD Library 22-13
The IEEE Library 22-13
Packages for the Simulation of Digital Designs 22-13
Packages for the Simulation of Multidomain Systems 22-14
Packages for Mathematical Operations 22-15
Subprograms 22-15
Procedures 22-15

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Functions 22-17
Declarations 22-18
TYPE Declarations 22-19
SUBTYPE Declaration 22-19
NATURE Declaration 22-20
Data Object Declarations 22-20
CONSTANT Declaration 22-20
SIGNAL Declaration 22-21
VARIABLE Declaration 22-21
FILE Declaration 22-22
QUANTITY Declaration 22-22
TERMINAL Declaration 22-23
Other Declarations 22-24
ATTRIBUTE Declaration 22-24
COMPONENT Declaration 22-24
Concurrent Statements 22-25
BLOCK Statement 22-25
PROCESS Statement 22-26
Concurrent Procedure Call Statement 22-27
Concurrent ASSERT Statement 22-27
Concurrent SIGNAL Assignment Statement 22-28
Component Instantiation Statement 22-29
Concurrent BREAK Statement 22-30
Sequential Statements 22-31
WAIT Statement 22-31
ASSERT Statement 22-31
SIGNAL Assignment Statement 22-32
VARIABLE Assignment Statement 22-33

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Procedure Call Statement 22-33
IF Statement 22-34
CASE Statement 22-35
LOOP Statements 22-36
NEXT Statement 22-37
EXIT Statement 22-37
RETURN Statement 22-38
NULL Statement 22-38
BREAK Statement 22-38
Simultaneous Statements 22-39
Simple Simultaneous Statement 22-39
Simultaneous IF Statement 22-40
Simultaneous CASE Statement 22-40
Simultaneous PROCEDURAL Statement 22-42
Simultaneous NULL Statement 22-43
Identifiers, Literals, and Expressions 22-44
Identifiers 22-44
Literals 22-44
Arithmetic and Logical Expressions 22-45
Predefined Data Types 22-47
Predefined Type Declarations 22-47
Predefined Attributes 22-48
Quantity Attributes 22-48
Signal Attributes 22-49
Data Type Bounds 22-50
Enumeration Data Types 22-50
Array Indexes for an Array A 22-50
Reserved Words 22-51

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Modeling Aspects in Twin Builder ... 22-56
Processes ... 22-56
Quantities, Signals, and Variables ... 22-57
Signal Assignments with Delay ... 22-57
Data Exchange in Mixed-Signal Models ... 22-59
Signal to Quantity Assignment (Digital to Analog) ... 22-59
Quantity to Signal Assignment (Analog to DIgital) ... 22-60
Solvability ... 22-61
WORK Library ... 22-62
Alias for File Names ... 22-62
Values on Sheet ... 22-63
Vector Inputs on Sheet ... 22-64
23 - Twin Builder Modeling Language ... 23-1
Common SML Conventions ... 23-1
Comments ... 23-1
Names ... 23-2
File Names ... 23-2
Define Name and Value ... 23-3
Type Names ... 23-3
Instance Names ... 23-4
Separators ... 23-4
Continuation Sign ... 23-5
Keywords, Pre-defined Functions and Operators, Constants ... 23-5
Nodes ... 23-6
Model Attributes ... 23-6
SML Model Descriptions ... 23-7
Preprocessor Instructions ... 23-7
#INCLUDE Instruction ... 23-8

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

#SET Instruction ... 23-8
#DEFINE Instruction ... 23-9
#UNDEF Instruction ... 23-9
#IF Compile Instruction ... 23-10
#EXTSIM Instruction ... 23-10
Model Instances ... 23-11
INTERN Model Instance ... 23-12
Models with Fixed Behavior ... 23-12
Models with User-Defined Behavior ... 23-13
Initial Condition Assignments and Equations ... 23-14
Defining Actions in States ... 23-14
MODEL Instance ... 23-15
UMODEL Instance ... 23-15
COUPL Instance ... 23-16
Structural Models (Subsheets) ... 23-17
Header Information of Structural Models ... 23-17
Model Description of Structural Models ... 23-18
Usage of Structural Models ... 23-18
Configuration Controls ... 23-19
Simulator Configurations ... 23-20
Output Configurations ... 23-20
24 - Scripting ... 24-1
Running VBScripts in 64-bit Windows Installations ... 24-1
PyAEDT (Beta) ... 24-1
25 - Twin Builder Design Conventions ... 25-1
Names of Components and Variables ... 25-1
Parameter Qualifiers ... 25-2
Qualifier Lists ... 25-2

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

System Outputs ... 25-2
Component Parameters ... 25-3
Parameter Types ... 25-6
Predefined Variables ... 25-8
Pre-defined Constants ... 25-8
Equations, Expressions, and Variables ... 25-9
Operators ... 25-10
Standard Mathematical Functions ... 25-11
Unit Suffixes of Numeric Data ... 25-16
SI Units ... 25-17
Unit Handling ... 25-17
Unit Types ... 25-18
26 - Technical Notes ... 26-1
Fixing Non-Convergence in Twin Builder ... 26-1
Using Time Trigger ... 26-2
Intention ... 26-3
Handling ... 26-3
State Graph Extension ... 26-3
C/C++ Interface Extension ... 26-4
Error Messages ... 26-5
Library Extensions ... 26-5
Simplorer - ModelSim Co-simulation Interface User Guide ... 26-6
Introduction (ModelSim) ... 26-7
Installation/Pre-Setup (ModelSim) ... 26-7
Setup (ModelSim) ... 26-7
Creating Co-simulation Models (ModelSim) ... 26-8
Schematic Capture (ModelSim) ... 26-8
Netlisting (ModelSim) ... 26-10

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Co-simulation Setup (ModelSim) 26-10
Simulation (ModelSim) 26-11
Starting Simulation (ModelSim) 26-11
Controlling Simulation (ModelSim) 26-12
Ending Simulation (ModelSim) 26-12
Saving and Continuing a Simulation (ModelSim) 26-12
Displaying Results and Post Processing (ModelSim) 26-12
Limitations and Known Issues (ModelSim) 26-13
About the VHDL-AMS working Library in Twin Builder 26-14
27 - DesktopScripting with IronPython 27-1
Introduction to IronPython 27-1
IronPython Mini-cookbook 27-2
Comments 27-3
Assigning/Creating Variables 27-3
Create Lists/Arrays 27-4
27-4
Create Dictionaries/Maps 27-4
Boolean Values 27-5
Converting Numbers to Strings and Vice Versa 27-5
String formatting/concatenation 27-5
Looping over lists 27-6
Looping over a range 27-6
A note about indentation 27-7
Obtaining more Information on Python 27-8
Discovering Methods 27-9
Help on a Method 27-9
Translating Script commands from VBScript 27-10
Script Method Argument 27-10

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Primitive Types ... 27-11
Named Array ... 27-11
Named Function ... 27-11
VBscript Method Call Types ... 27-11
Converting VBScript Function calls to IronPython Syntax ... 27-12
Return Values ... 27-13
Primitive Method Arguments ... 27-13
Named Array Argument ... 27-13
Named Array Values with All Key Value Pairs ... 27-15
Named Arrays with Nested Named Arrays ... 27-16
Function Blocks ... 27-17
Scripting Using IronPython: Putting it all Together ... 27-18
IronPython Script Execution Environment ... 27-18
Script Argument for IronPython ... 27-19
Scripting Using Embedded VBScript or JavaScript ... 27-20
Scripting with IronPython ... 27-22
Standalone IronPython and Desktop IronPython ... 27-23
IronPython Samples ... 27-26
28 - Twin Builder Terminology ... 28-1
Glossary: A ... 28-1
A/D (Analog-to-Digital) ... 28-1
AC simulation ... 28-1
AC simulator ... 28-1
Action type ... 28-1
.afa file ... 28-1
Analytic frequency step response ... 28-1
Animated Symbol ... 28-1
.aws file ... 28-1

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Glossary: B 28-2
Backplane 28-2
Basic version 28-2
Basics 28-2
Behavioral model 28-2
Best representation 28-2
Bezier 28-2
Block 28-2
Block diagram 28-2
Block Diagram Module (BDM) 28-2
Bookmark 28-2
.brs file 28-3
Glossary: C 28-3
Characteristic 28-3
Characteristic values 28-3
C-Interface 28-3
C-Model Editor 28-3
Color scheme 28-3
Compiler 28-3
Computation sequence 28-3
Connection rule 28-3
Conservative node 28-3
Coordinate system 28-4
Crossing over 28-4
Cursor 28-4
Glossary: D 28-4
Data cache 28-4
Data channel 28-4

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Data filter 28-4
Data format 28-4
Data set 28-4
DC simulation 28-4
DC simulator 28-4
Differentiation 28-4
DISPLAY 28-5
Display Element 28-5
DLL 28-5
Dongle 28-5
DSDE 28-5
Glossary: E 28-5
Electric circuit 28-5
Electric Circuit Module (ECM) 28-5
Element 28-5
Entity 28-5
Euler formula 28-5
Evaluation function 28-6
Expert mode 28-6
Export filter 28-6
Extern View 28-6
Glossary: F 28-6
FFT 28-6
File output 28-6
Fitness function 28-6
Formula Module (FML) 28-6
Frequency step response analysis 28-6
Glossary: G 28-6

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Genetic Algorithm 28-6
Grid lines 28-7
Glossary: H 28-7
HMAX 28-7
HMIN 28-7
Glossary: I 28-7
ID 28-7
Information window 28-7
Initial condition 28-7
Initial state 28-7
INT 28-7
Interactivity pad 28-7
Iteration 28-7
Glossary: J 28-8
Jacobian Matrix 28-8
Glossary: K 28-8
Keyword 28-8
.krn file 28-8
Glossary: L 28-8
Language concept 28-8
Library 28-8
Local discretization error (LDF) 28-8
.log file 28-8
Glossary: M 28-8
Macro 28-8
Main skeleton (.skl) 28-9
Maximum current error (IEMAX) 28-9
Maximum number of iterations (Iteratmax) 28-9

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Maximum voltage error (VEMAX) 28-9
.mda/.mdk file 28-9
.mdx file 28-9
Model 28-9
Model tree 28-9
Module 28-9
Monitor 28-9
Monte Carlo Analysis 28-10
.mtx file 28-10
Multi simulation 28-10
Mutation 28-10
Glossary: N 28-10
Network installation 28-10
Newton-Raphson-Algorithm 28-10
Non-conservative node 28-10
Normal mode 28-10
Glossary: O 28-10
Object 28-10
Object browser 28-11
Offline 28-11
Online 28-11
Online graphic 28-11
Online graphic output 28-11
Option 28-11
Output 28-11
Glossary: P 28-11
Password 28-11
Petri net 28-11

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Pipe 28-11
Postprocessing 28-11
Preprocess 28-12
Preprocessor directive 28-12
Print colors 28-12
Project 28-12
Glossary: Q 28-12
Qualifier 28-12
Quality criterion 28-12
Queue 28-12
Glossary: R 28-12
Recombination 28-12
Roll back 28-12
Glossary: S 28-13
Sample time 28-13
Schematic 28-13
Screen colors 28-13
Section 28-13
Selection 28-13
Simulation backplane technology 28-13
Simulation model 28-13
Simulation parameter 28-13
Simulator 28-13
Simulator coupling 28-13
Simulator interface 28-14
.smd file 28-14
SML 28-14
SML Editor 28-14

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

.sml file 28-14
SML key word 28-14
Solver 28-14
.ssc file 28-14
.ssh file 28-14
State 28-14
State graph 28-14
State Graph Module (SGM) 28-15
Status line 28-15
Subsheet 28-15
Sub-simulator 28-15
Sub-skeleton 28-15
Symbol editor 28-15
Symbol level 28-15
Symbols 28-15
Synchronization 28-15
Synchronization 28-15
System quantity 28-16
System simulation 28-16
System variable 28-16
Glossary: T 28-16
Task 28-16
Template 28-16
TEND 28-16
Time Function Module (TFM) 28-16
Time limited 28-16
Time step 28-16
Toolbar 28-16

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

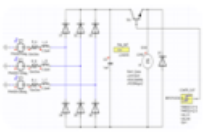
Total fitness ... 28-16
TR simulation ... 28-17
TR simulator ... 28-17
Transition ... 28-17
Transition component ... 28-17
Trapezoid Formula ... 28-17
Glossary: U ... 28-17
UDMinit ... 28-17
UDMMain ... 28-17
User defined component (UDC) ... 28-17
User defined model (UDM) ... 28-17
User management ... 28-17
Glossary: V ... 28-18
Version report ... 28-18
VHDL ... 28-18
VHDL Model Editor ... 28-18
VHDL-AMS ... 28-18
VHDL-AMS simulator ... 28-18
ViewTool ... 28-18
Glossary: W ... 28-18
Window elements ... 28-18
workflow ... 28-18
Worst Case Analysis ... 28-18
Glossary: Y ... 28-19
YMAX ... 28-19
YMIN ... 28-19
Index ... **Index-1**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

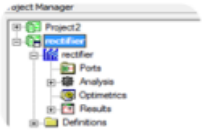
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1 - Welcome to Twin Builder™ Help

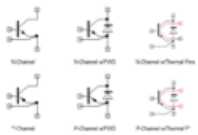
Click for help on the following topics:


Getting Started Twin Builder

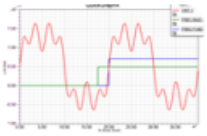
Twin Builder User Interface


Projects

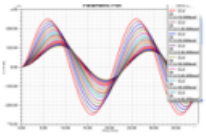
Designs


Components

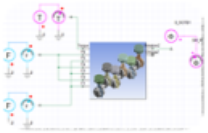
Analysis


Results

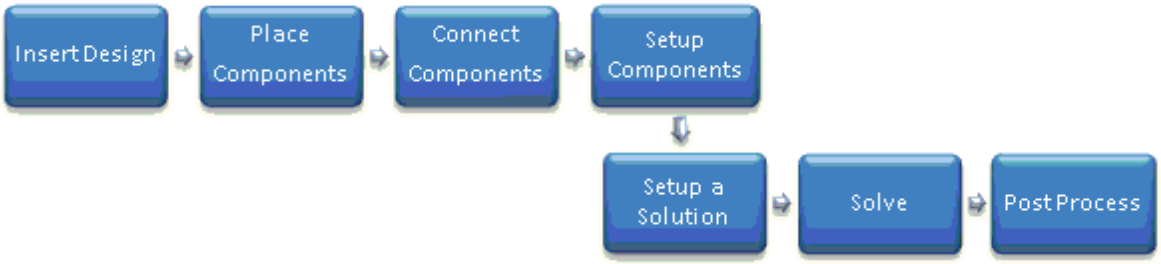
Working with Legacy Files


Optimetrics

Scripting


Coupling

Click for help on Twin Builder **Process Flow** topics:



Twin Builder User Interface Quick Links

Use these links for quick information on the following topics.

The Twin Builder Desktop	Design Area
Project Manager Window	Progress Window
Properties Window	Status Bar

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Message Manager	Running Twin Builder From a Command Line
Keyboard Shortcuts	The Component Libraries Window

The help system provides different ways to find information and navigate quickly:

- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* - Contents You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

Projects Quick Links

Use the following links for quick information on the following topics.

Creating Projects	Setting Options for Twin Builder
Twin Builder File Types	Translating Legacy Simplorer Projects and Schematics
Opening Existing Projects	Importing Simulation Models
Analyzing Designs	Removing Unused Definitions

The help system provides different ways to find information and navigate quickly:

- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* - Contents You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

Designs Quick Links

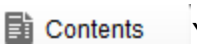
Use the following links for quick information on the following topics.

The Schematic Editor	Setting Schematic Editor Options
Creating Simulation Models	About the Schematic Editor Window
Placing Components	Operations on Components and Design Objects

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Connecting Components	Placing Ports
Sorting Components	Adding Drawing Elements
Placing Reports on Schematics	Multi-Page Schematics
Hierarchical Designs	Subcircuits

The help system provides different ways to find information and navigate quickly:

- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* - 

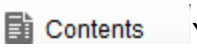
You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

Components Quick Links

Use the following links for quick information on the following topics.

Choosing Components	Placing Components
Component Libraries	Editing Components
Editing Component Parameters	Building Component Dialog Boxes
Setting Up IGBT Models	Managing Libraries
Encrypting and Encoding Components	Working with Variables

The help system provides different ways to find information and navigate quickly:

- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* - 

You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

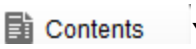
Analysis Quick Links

Use the following links for quick information on the following topics.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Overview of Analyzing Designs	Standard Analysis Types
Analysis Setup	Advanced Analysis Types - Optimetrics
Solution Options	Importing Solution Data
Setting Simulation Outputs	

The help system provides different ways to find information and navigate quickly:

- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* - 

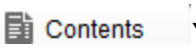
You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

Working with Legacy Files Quick Links

Use the following links for quick information on the following topics.

Translating Legacy Libraries
Simplorer File Types
Opening Existing Projects
Editing Translated Components

The help system provides different ways to find information and navigate quickly:

- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* - 

You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

Optimetrics Quick Links

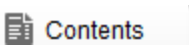
Use the following links for quick information on the following topics.

Setting up a Parametric Analysis	Setting up an Optimization

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	Analysis
Setting up a Sensitivity Analysis	Tuning a Variable
Setting up a Statistical Analysis	Setting a Range function
Setup Calculations for Optimetrics.	Adding a cost function

The help system provides different ways to find information and navigate quickly:

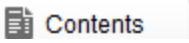
- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* -  You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

Results Quick Links

Use the following links for quick information on the following topics.

Creating Reports	Report Setup Options
Modifying Reports	Report Types
Adding Plots to Schematics	Working with Traces
Adding Data Markers to Traces	

The help system provides different ways to find information and navigate quickly:

- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* -  You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

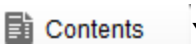
Scripting Quick Links

Use the following links for quick information on the following topics.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Recording a Script	Running a script
Stopping Script Recording	Pausing and Resuming a Script
Stopping a Script	Scripting Guide

The help system provides different ways to find information and navigate quickly:

- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* - 

You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

Twin Builder Getting Started Guides

Twin Builder documentation includes the following Getting Started Guides:

- Twin Builder Getting Started Guide

  This *Getting Started Guide* is written for Twin Builder beginners as well as experienced users using Twin Builder for the first time. This guide leads you step-by-step through creating, solving, and analyzing the results of a Three-Phase Rectifier. The guide also leads you through the process of importing a legacy Simplorer 7 Three-Phase Rectifier schematic into Simplorer Release 16.2, saving it, migrating it into the current version of Twin Builder, then solving and analyzing the translated model. Variations of the Three-Phase Rectifier are then presented to demonstrate additional capabilities of Twin Builder.

  Follow the steps in this guide, to learn how to perform the following tasks in Twin Builder:

  - Basic Twin Builder functions
  - Finding and choosing components from a library
  - Placing and arranging components
  - Connecting components on the schematic sheet
  - Modeling with electric circuit components
  - Modeling time-controlled sources
  - Modeling with state graph components
  - Modeling with block components
  - Modeling with VHDL-AMS components
  - Using display elements for displaying simulation results
- Twin Builder Modelica Tutorial
- VHDL-AMS Tutorial

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2 - Getting Started

Ansys Twin Builder™ is an integrated, multi-domain, mixed-signal simulator for complex technical systems. Whether you are a beginner or an experienced professional, Twin Builder's comprehensive suite of tools provides accurate and reliable results in very little time.

You can create related simulation models quickly, process simulations accurately and reliably with Twin Builder's unique simulator backplane technology, dynamically interact with other Ansys applications such as Maxwell® and Q3D® and with third party applications such as Simulink®, and present and arrange the results with powerful postprocessors. You can also seamlessly transfer the simulation data and results presentations to other software applications. You can use Twin Builder's powerful scripting interface to automate complex and repetitive simulation tasks.

Twin Builder is integrated into the Ansys Electronics Desktop.

Related Topics

System Requirements	Status Bar
Launching Twin Builder	Running Twin Builder from a Command Line
Introducing the Twin Builder Desktop	Top Menu Bar
Project Manager Window	Ribbons
Properties Window	Shortcut Keys
Properties Dialog Box	Undoing Commands
Message Manager	Getting Started Guides
Progress Window	Welcome to the Ansys Electronics Desktop
Design Area	Launching the Ansys Electronics Desktop
Component Libraries Window	

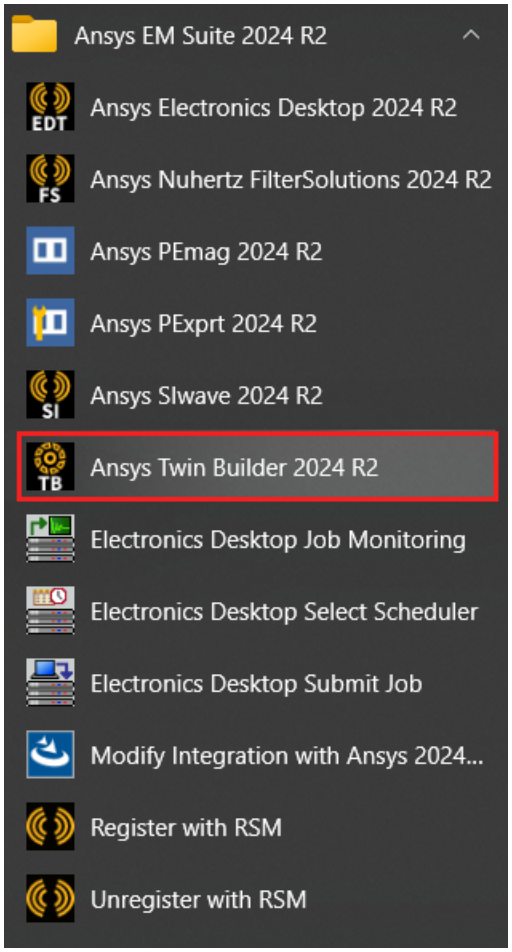
Launching Twin Builder

After you have installed Ansys Electronics Desktop, you can start Twin Builder by one of the following methods:

- Double-click the **Ansys Twin Builder** icon on your desktop.

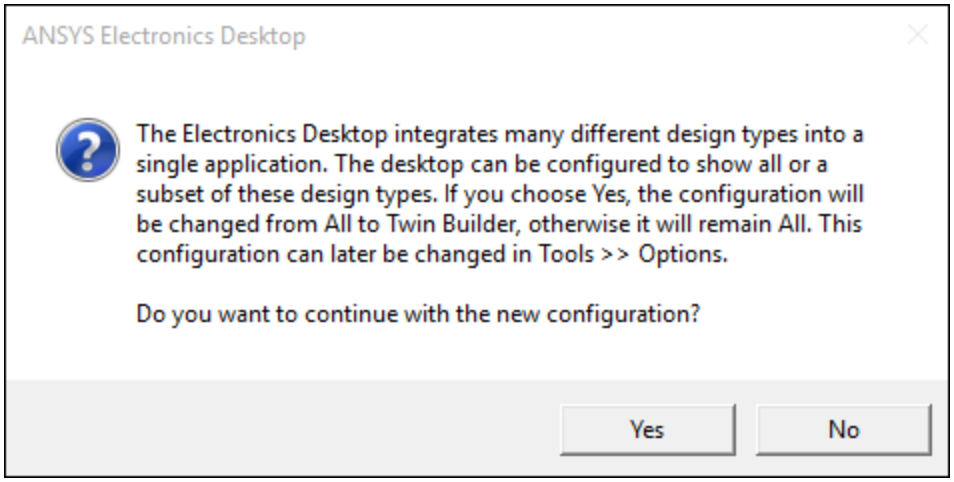
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Select **Start** > **Ansys EM Suite 2024 R2** > **Ansys Twin Builder 2024 R2** from the Windows Start menu.



If you have not changed the **Set targeted configuration** option to Twin Builder on **Tools** > **Options** > **General Options**, the following message appears. Click **Yes** to change the default configuration to Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- Click the **Ansys Electronics Desktop** icon on your desktop or on the Windows Start menu after changing the default configuration to Twin Builder.

Related Topics

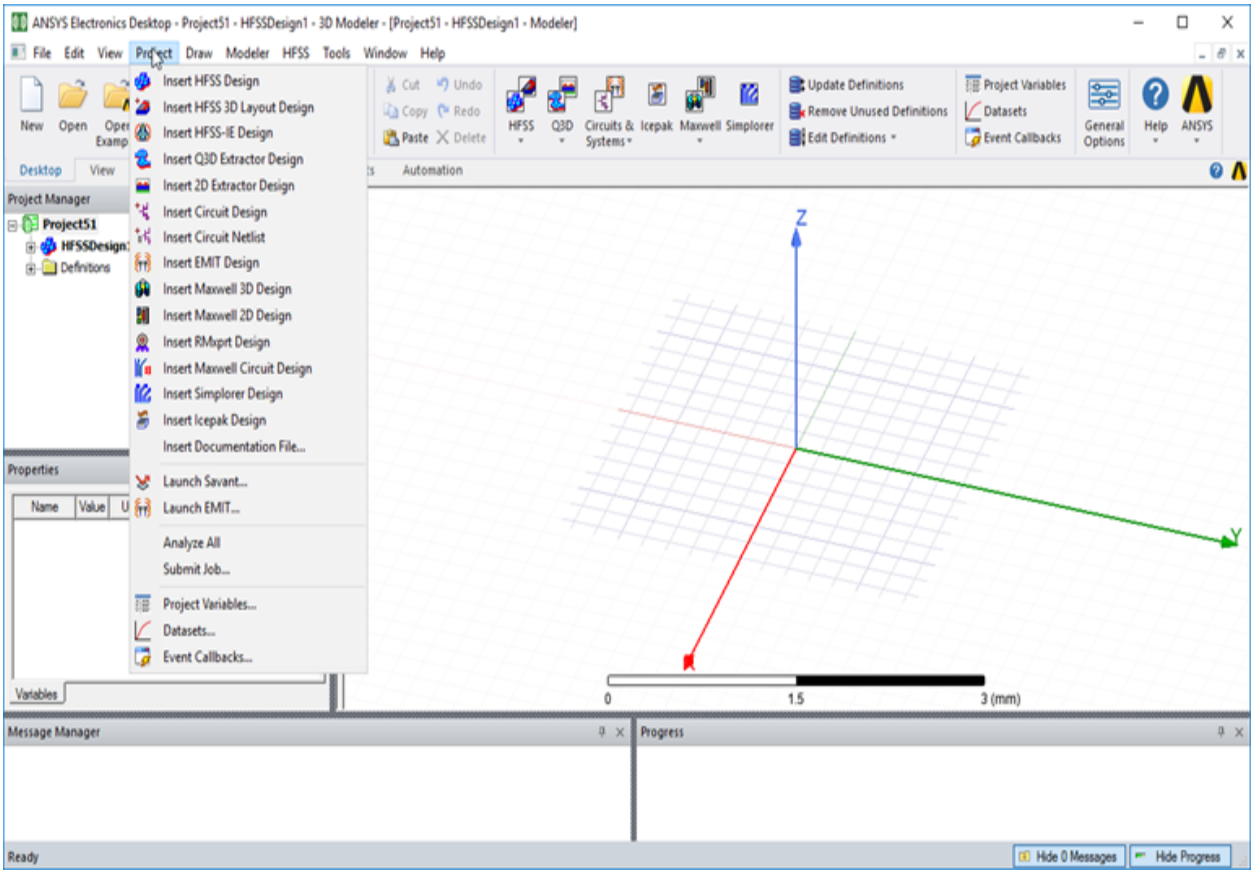
Welcome to the Ansys Electronics Desktop

Launching Ansys Electronics Desktop

Welcome to the Ansys Electronics Desktop

The Ansys Electronics Desktop, illustrated in the following figure, provides a comprehensive environment for designing and simulating various electronic components and devices. The Ansys Electronics Desktop consists of a unified user interface where both electromagnetic designs and circuits can be created. Typically, you can create or import a design, set up the simulation, validate your design, run the analysis, and post process the results.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The desktop has the following design types and features:

- HFSS - a general purpose 3D interface for the design, analysis, and simulation of electromagnetic components.
- HFSS 3D Layout - a full-wave layout-based electromagnetic simulator with a specialized interface for geometries created in layout.
- HFSS-IE - a full wave Integral Equation solver for large open problems.
- Q3D Extractor - a quasi-static 3D solver for extracting lumped RLGC parameters and Spice models.
- 2D Extractor - a 2D solver for extracting per-unit-length RLGC parameters of transmission lines.
- Circuit - a schematic-based interface to the Nexxim circuit simulator.
- Circuit Netlist - a netlist (text-based) interface to the Nexxim circuit simulator.
- EMIT – system simulation for predicting and mitigating radio frequency interference (RFI) in electronic devices.
- Maxwell 3D - uses finite element analysis (FEA) to solve three-dimensional (3D) electrostatic, magnetostatic, eddy current, and transient problems.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Maxwell 2D - uses finite element analysis (FEA) to solve two-dimensional (2D) electrostatic, magnetostatic, eddy current, and transient problems.
- RMxprt - a template-based electrical machine design tool that provides fast, analytical calculations of machine performance and 2-D and 3-D geometry creation for detailed finite element calculations in Ansys® Maxwell®.
- Maxwell Circuit - sets up external circuit designs to supply excitations to coil terminals for Maxwell 2D and 3D Eddy Current and Transient designs.
- Simplorer - an integrated, multi-domain, mixed-signal simulator for complex technical systems.
- Icepak - a general purpose 3D interface for the design, simulation, and thermal analysis of electronic components
- Mechanical – (Beta) perform Modal analyses to determine natural frequencies of vibration and Thermal analyses to determine temperatures and heat flux.

You can also launch the following tools:

- Launch Savant - a tool for simulating near-field, far-field, and coupling performance of antennas installed on platforms and within other complex and electrically large environments.
- Launch EMIT - a simulation tool for the prediction of radio frequency interference (RFI) in complex environments.

If you go to the **Project** menu, you can access all of the design types. Any combination of these design types can be inserted into a single project file. The schematics can be used to wire up the different field solver models and create a model of a high-level system. The Ansys Electronics Desktop provides an efficient way to manage complicated projects that require several different analysis tools to model all of its pieces. Designs can also be parameterized. With the help of the Optimetrics feature the best design variations can be made available to other modules when the designs are linked into a higher level simulation. This lets you study the effect of varying a design parameter on the behavior of the entire system.

Related Topics

Launching Ansys Electronics Desktop

Launching Twin Builder

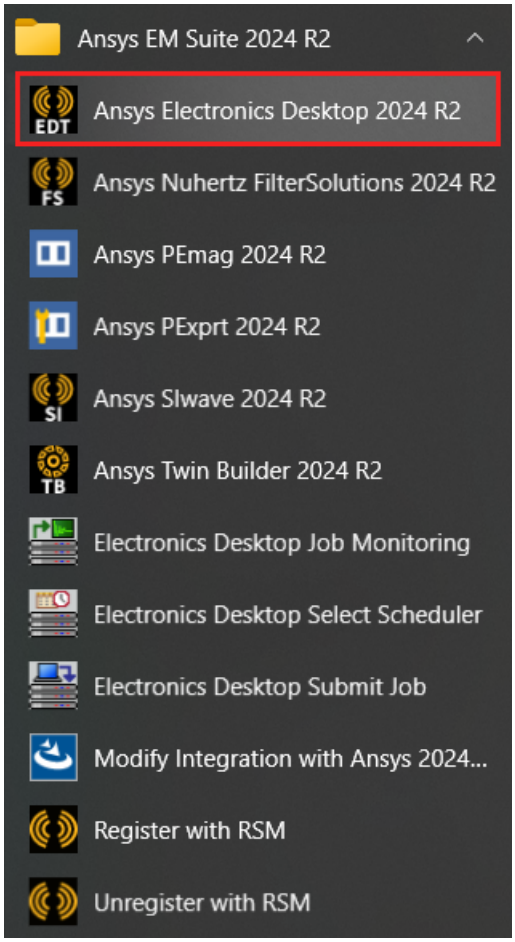
Launching Ansys Electronics Desktop

After you have installed Ansys Electronics Desktop, start the program by one of the following methods:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Double-click the **Ansys Electronics Desktop** icon on your desktop.

- Select **Start** > **Ansys EM Suite 2024 R2** > **Ansys Electronics Desktop 2024 R2** on the Windows Start menu.



Related Topics

Welcome to the Ansys Electronics Desktop

Launching Twin Builder

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Installing PyAEDT (Beta)

PyAEDT is a Python library that interacts with the AEDT API to make scripting simpler for the end user. It supports all AEDT 3D products (HFSS, Icepak, Maxwell 3D, and Q3D Extractor), 2D tools, Ansys Mechanical, EMIT, Circuit tools like Nexxim, system simulation tools like Twin Builder, and layout tools like HFSS 3D Layout and EDB. Additionally, it enables the end user to have a CPython interface with AEDT. Its class and method structures simplify operation for the end user, enabling more Pythonic code while reusing information as much as possible across the various APIs.

Documentation for PyAEDT can be found online at: https://aedt.docs.pyansys.com/version/stable/

Installing PyAEDT adds three items to the **Tools** > **Toolkit** > **PersonalLib** menu and to the **Automation** tab:

- **Console** – Launch the PyAEDT console.
- **Jupyter Notebook** – Launch Jupyter Notebook (a computational notebook) in an internet browser.
- **Run PyAEDT Script** – Launch a file browser, allowing you to select a Python script to run via PyAEDT.

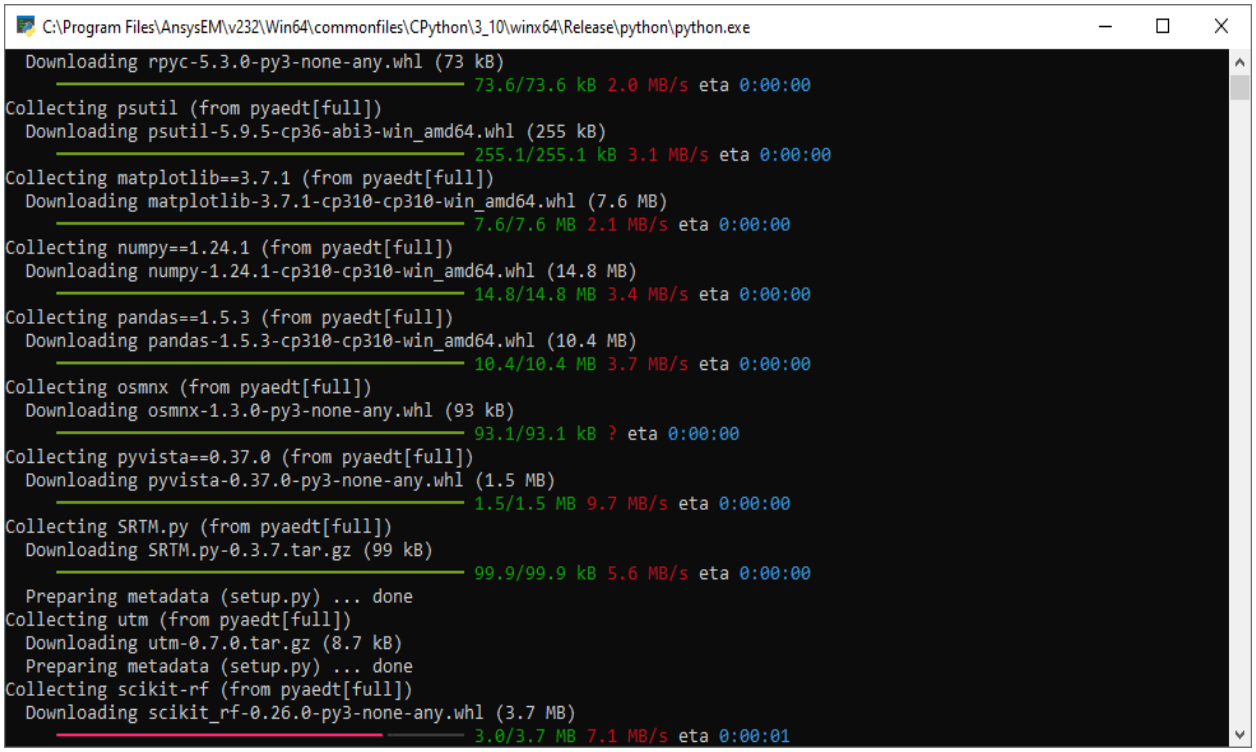
To install PyAEDT:

- From the **Automation** tab, click **Install PyAEDT**.
-

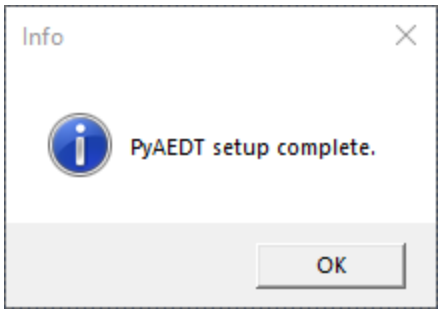
  A command window appears, and installation begins.

  > **Note:**
  >
  > It may take a moment for installation to begin. The installation takes a few minutes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

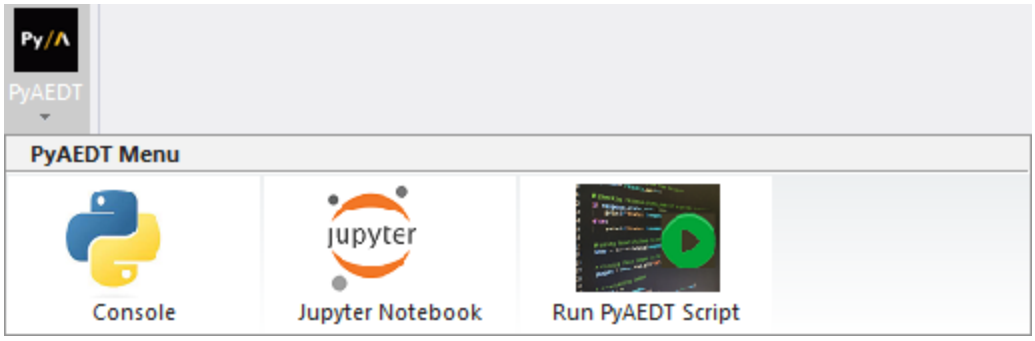


When the installation has finished, the command window closes, and an alert box appears:



The **Automation** tab updates to display a PyAEDT drop-down menu:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



System Requirements

Twin Builder supports certain versions of the Windows operating system. For supported platforms and system requirements, go to the Platform Support website and select the following document:

Ansys 2024 R2 **- Platform Support by Application / Product (PDF)**

This document covers all Ansys products. Refer to the **Electronics Applications** section.

Limitations of Linux Installations:

While the majority of the Ansys Electromagnetics Suite applications and features are supported on both Windows and Linux platforms, some, including Twin Builder, are not supported on Linux. See the following topic for details:

Windows vs. Linux Installations

Twin Builder Student Limitations

Twin Builder is supported in Ansys Electronics Desktop Student, with the following limitations:

- Component Limits: 15 Components.
- Component count includes:
  - Top level Schematic Components.
  - Components in a SubCircuit.
  - Structural SML Components in an SML Model.
- Restrictions in Compiling a Twin Model and Co-Simulation FMU.
- Restrictions in Exporting Models as Twin or FMU.

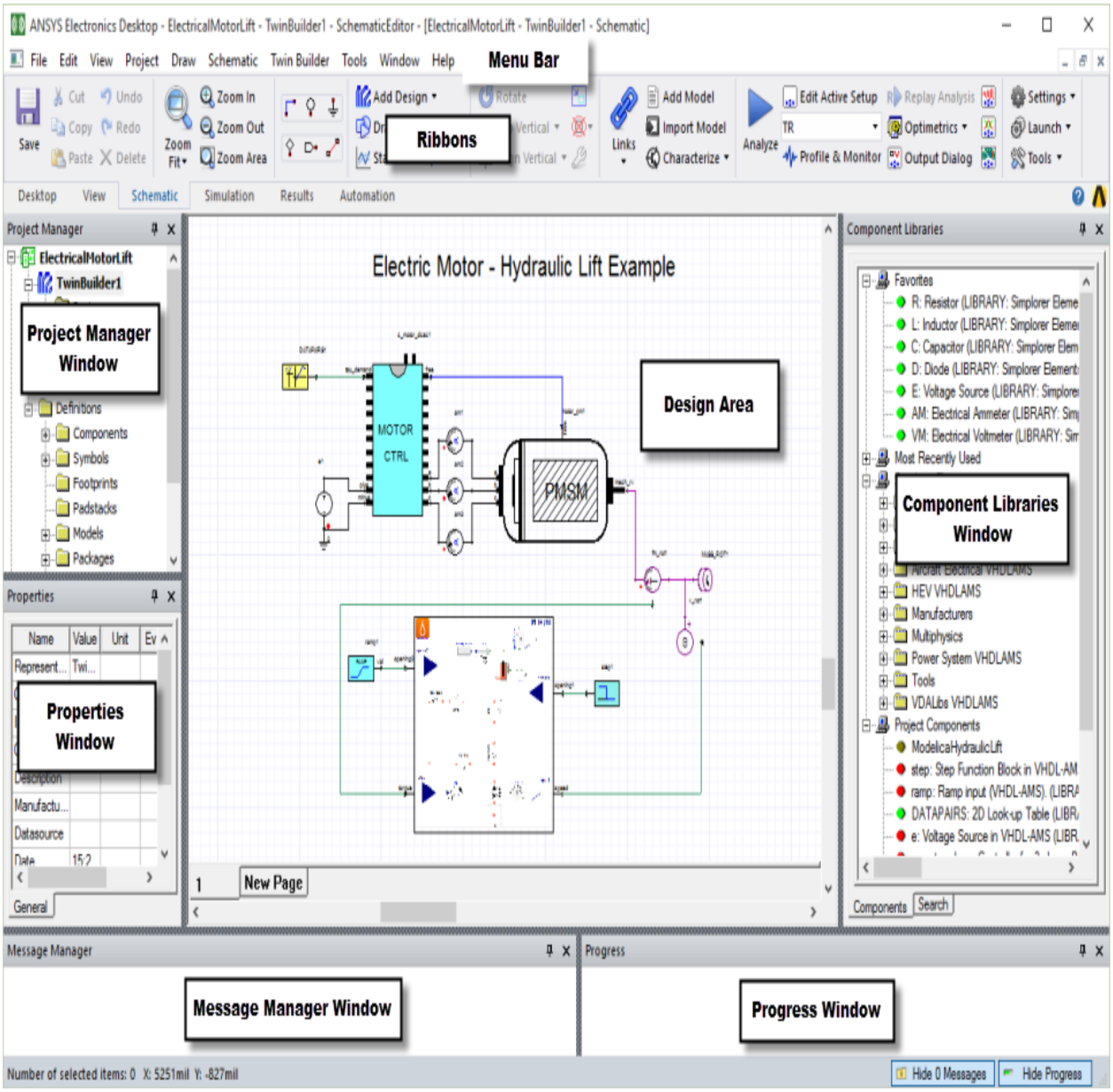
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Restrictions in saving design if components are greater than the limit.
- Circuit model generation from S-parameters is limited to a matrix size of 10.

The Twin Builder Desktop

The Twin Builder desktop consists of a menu bar, a status bar, and several windows,. The illustration below shows the windows and menus that are displayed for a typical Twin Builder design. The desktop changes, depending on the type of project, and on the window that is active in the Design Area. For example, the illustration below shows a Schematic Editor Window in the Design Area.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



This section describes the following desktop features:

Project Manager Window

Properties Window

Properties Dialog Box

Message Manager Window

Progress Window

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Design Area

Component Libraries Window

Window Layouts

Status Bar

Top Menu Bar

Ribbons

Shortcut Keys

Help

The Project Manager Window

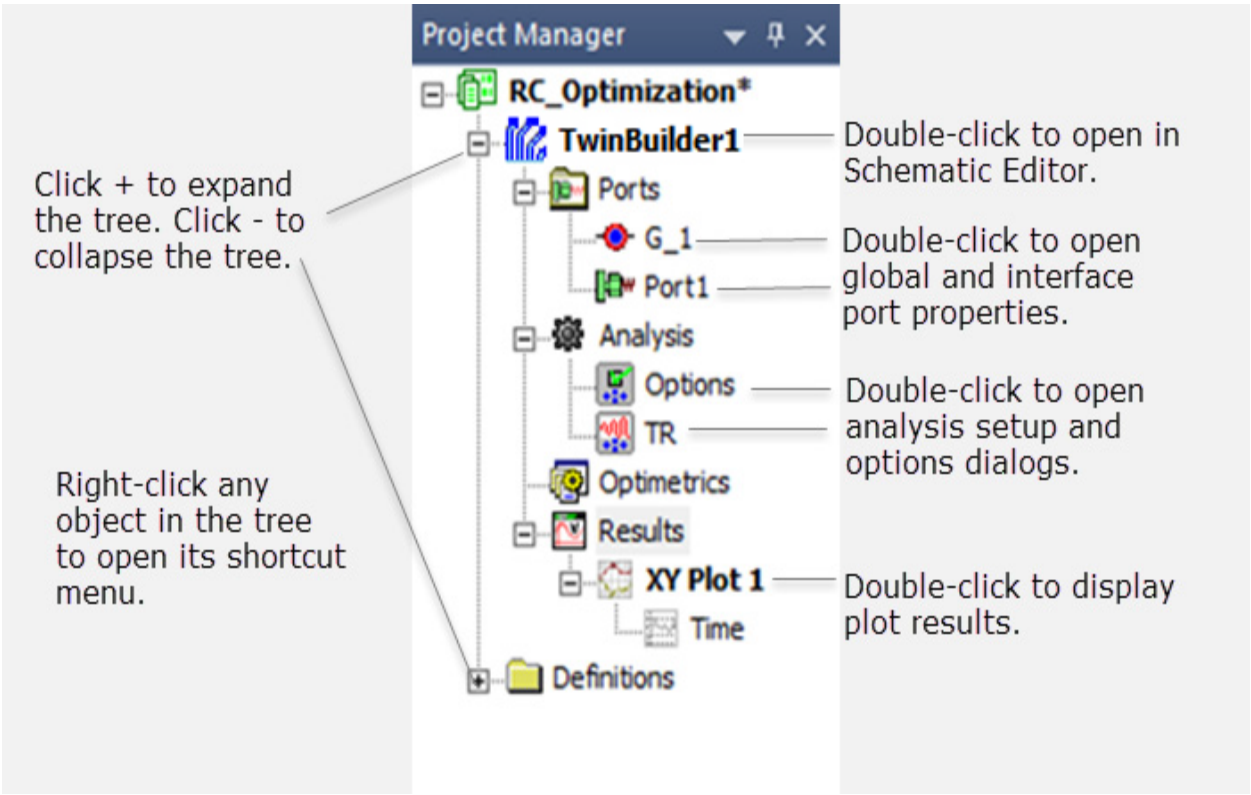
The **Project Manager** window shows the projects loaded into Twin Builder. Each project may consist of one or more designs.

To toggle display of the **Project Manager** window:

- Click **View** > **Project Manager**.
- On the ribbon, click **View** > **Docking Windows** > **Project Manager**.

The **Project Manager** window contains details about all open Twin Builder projects in a form known as the Project tree, as shown below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

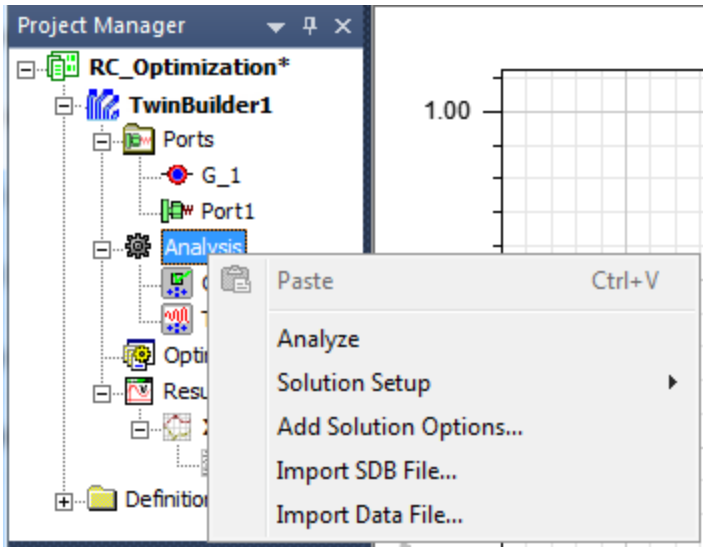


The top node listed in the Project tree is the project name. It is named **Project***n* by default, where *n* is the order in which the project was added to the current session of Twin Builder. Expand the project icon to view all the project's Twin Builder design information and material definitions. If there are multiple designs in a project, each design's ports, analysis solution setup, and results are displayed as entries in a separate subtree.

Depending on the options set in the **User Interface** panel under **General** in the Options dialog box, the selected Project tree may be rendered in **bold** text; or a small "window" icon may display next to the selected Project tree. The icon will be gray if the editor or plot window associated with the selected element is closed, or not in focus. Click the gray icon to open the window and bring it into focus (on top).

Right-click a folder or item to display a shortcut menu. For example, right-click **Analysis** to display this shortcut menu:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



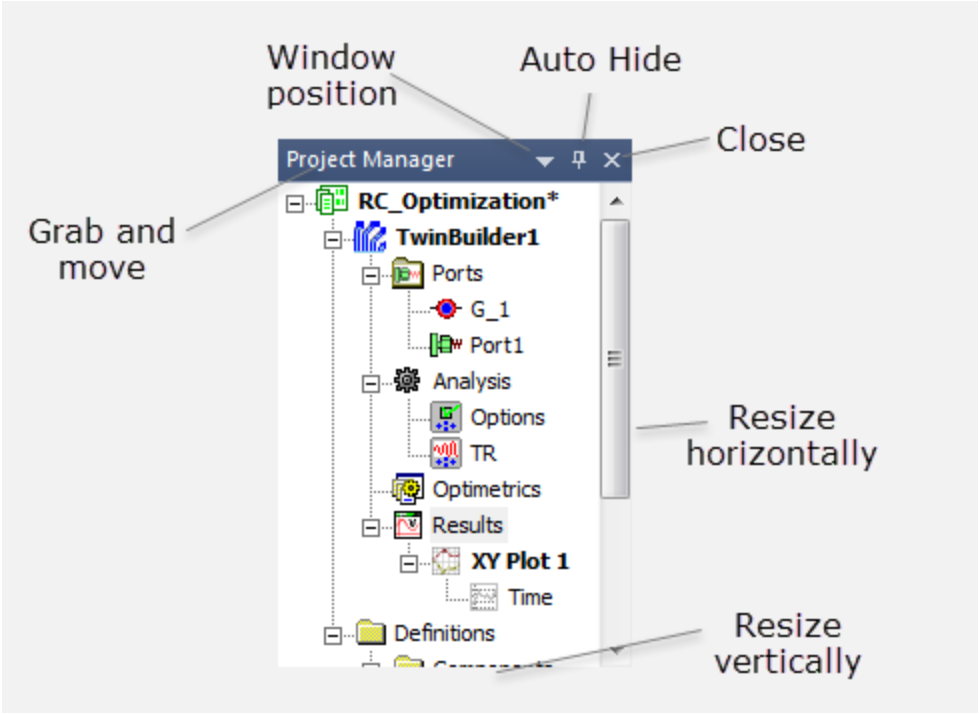
Virtually all project editing and management can be done in the **Project Manager** pane; simply right-click the items in the pane. Many of these functions are also available from the main menu bar in the **Twin Builder** menu.

You can move and resize the **Project Manager** window as needed. You can also dock it to any edge of the Twin Builder desktop.

Hints	• Grab and drag the window by its title bar to move it. • Click the push pin in the title bar to toggle between hiding and auto hiding the window. • Resize the window by dragging its edges. • Click the triangle in the title bar to select to float (undock), dock, auto hide, and hide the window. • Click the "**X**" in the title bar to close the window.

There are controls on the window when it is docked, as shown below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Related Topics

Viewing Twin Builder Design Details

Setting the Project Tree to Expand Automatically

You can set the Project tree to expand when an item is added to a project.

1. Click **Tools** > **Options** > **General Options**. The **Options** dialog box appears.
2. Click **General** > **User Interface** > **Expand Project Tree on Insert**.

Viewing Twin Builder Design Details

Once you insert an Twin Builder design into a project, it is listed as the second node in the Project tree. It is named **TwinBuilder***n* by default, where *n* is the order in which the design was added to the project. Expand the design icon in the Project tree to view all of the specific data about the model, including its solution and post-processing information.

The **TwinBuilder***n* node contains the following project details:

Ports	Displays the interface and global ports added to a Twin Builder design.
Analysis	Displays the solution setups and solution options for a Twin Builder design. A

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	solution setup specifies how Twin Builder will compute the solution. The solution options provides settings such as the integration formula, iterations, and maximum and minimum errors.
Optimetrics	Displays any optimetrics setups added to a Twin Builder design.
Results	Displays any post-processing reports generated.

> **Note:**
>
> To edit a project's design details, right-click a design setup icon in the Project tree and select **Design Properties**.
>
> A dialog box appears with that setup's parameters, which you can then edit.

The Definitions Folder

The **Definitions** folder under **TwinBuilder***n* contains a listing of the components, symbols, models, packages and scripts currently in use by designs in the project.

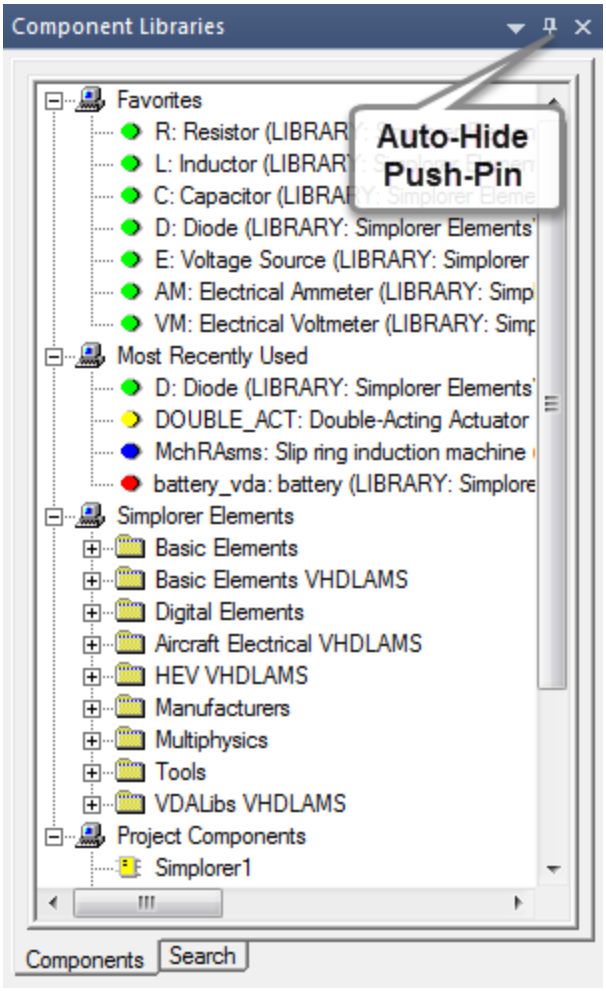
- Right-click an item to display a menu of editing commands for that object type. This menu for components includes **Load Example**. Select **Load Example** to load the example project for that component.
- Double-click an item to open an editor (for example, **Symbols**) or editing dialog box (for example, **Components**) used to set the properties of that item.

The Component Libraries Window

Use the **Component Libraries** window to search for and select schematic components that are available in the installed libraries. The **Component Libraries** window contains a **Components** tab and a **Search** tab.

A typical **Component Libraries** window is shown below. Your actual window contents will vary with the libraries installed and with use.

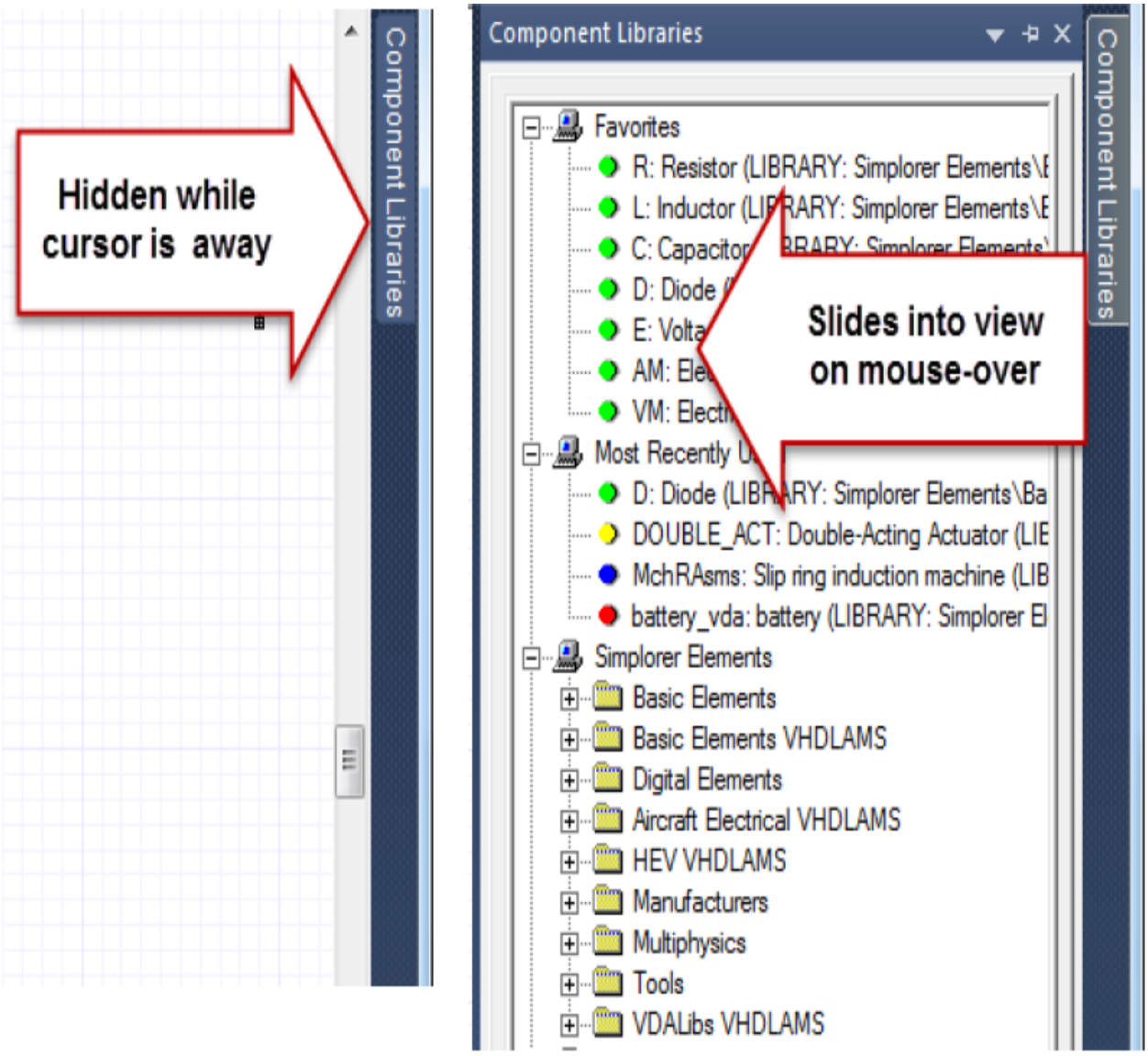
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Auto-Hide Feature

Click the auto-hide push pin in the upper right corner of the **Component Libraries** window to toggle auto-hide. When auto-hide is enabled, the **Component Libraries** window slides out of sight and a **Component Libraries** label appears on the edge of the main window. Place your mouse pointer over the label to display the window.

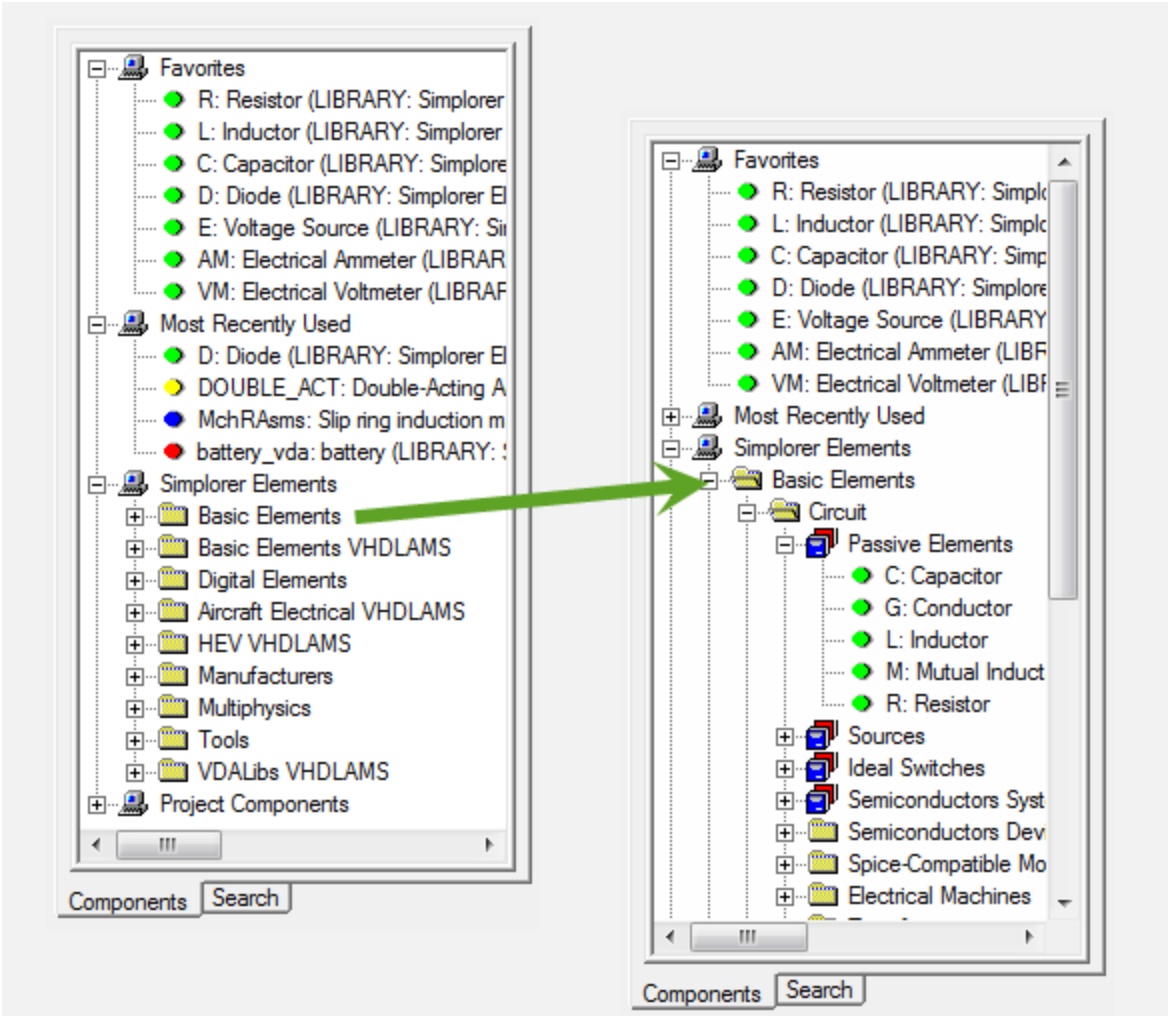
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The Components Tab

Use the **Components** tab in the **Component Libraries** window to browse and select schematic components from the available elements libraries. A typical **Components** tab is shown below. Your actual window contents will vary with the libraries installed. The installed libraries are listed alphabetically under **Simplorer Elements** in hierarchical folders grouped by component function. For example, the **Basic Elements** library expands into several categories and sub-categories as shown.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Individual elements are designated by colored symbols next to the element abbreviation and name.

Right-click a component to:

- Add the component to the Favorites list.
- Place the component on a schematic.
- Edit the component.
- View component help.
- Load the example project (if one exists) for the selected component.

Simplorer Element Library Symbols

The symbols used to identify element types in the component libraries are:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Internal Twin Builder components. Most of the components in the Basic library are internal ones. The models are calculated within the internal simulator.

- Standard and user-defined C-Models.

- Standard and user-defined VHDL-AMs models.

- Standard and user-defined SML models.

- User-defined SPICE models.

Because the number of libraries and elements in the libraries can be large, the **Component** tab includes **Favorites** and **Most Recently Used** lists. These lists provide quick access to components. Elements used in the current project are listed under **Project Components**.

> **Note:**
>
> For security reasons, encrypted components are not saved in the **Most Recently Used** or **Favorites** lists.

Using the Favorites and Most Recently Used Lists

By default the **Favorites** list contains these elements:

- R (Resistor)
- L (Inductor)
- C (Capacitor)
- D (Diode)
- E (Voltage Source)
- AM (Ammeter)
- VM (Voltmeter)

To add a library element to the **Favorites** list, right-click an element and select **Add to Favorites**. When a component is in the **Favorites** list, you can select, place, edit, view the component help, and load its example project (if one exists) from there.

To remove an element from the **Favorites** list, right-click the element and select **Remove from Favorites**.

The **Most Recently Used** list contains the most recently used elements. These elements can be selected and placed from this list. The default number of elements is 10, but you can change this number in the **Options** dialog box. Click **Tools** > **Options** > **General Options** to display the **Options** dialog box. Select **General** > **Component Libraries Options**, then enter a value in the **Component Tree Options** area.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Help for Components

To launch help for a component, right-click the component and select **View Component Help** from the shortcut menu.

Loading Component Example Projects

Example projects for the supplied components are installed in the *<product_installation>***Examples** directory. To load an example project for a component, right-click the component and select **Load Example**.

Placing Components on a Schematic

To place a component, do one of the following:

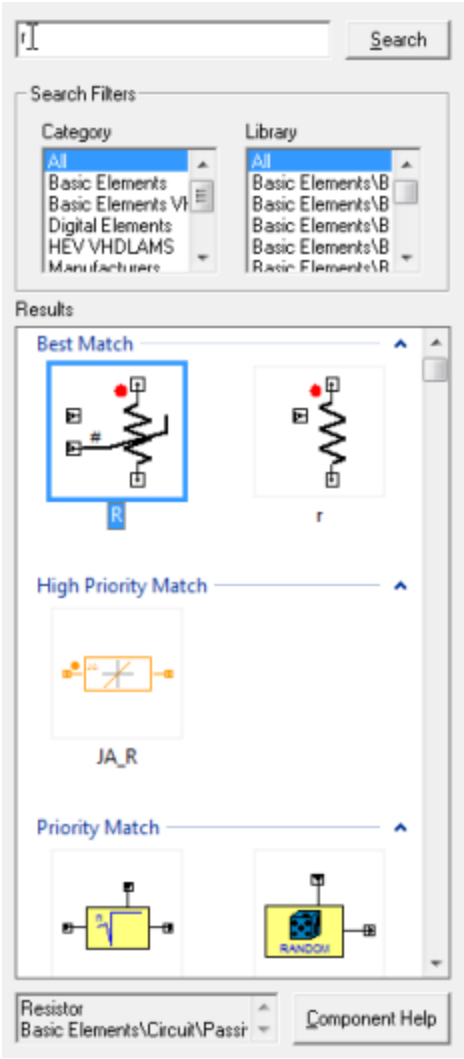
- Double-click the component.
- Drag a component into the schematic editor.
- Right-click a component and select **Place Component**.

See ***Selecting and Placing Components*** for details.

The Search Tab

Use the **Search** tab to enter full or partial words (case-insensitive) present in the names or descriptions of components in the search field. The search is done dynamically as you type. You can select the specific categories and libraries you want to include in the search. Use the Ctrl key to select multiple categories or libraries.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Results appear as symbols arranged in groups. Components which match the search string exactly appear in the **Best Match** group.

Select a component to see its description and library path in the text box at the bottom of the tab. Drag the desired component from the **Results** panel onto the schematic editor, or double-click it to attach it to the mouse. Click **Component Help** to view the help for the selected component.

Properties Window

The **Properties** window displays the attributes of an item selected in the Project tree (such as a design, report, or analysis setup) or in the schematic editor. The **Properties** window lets you edit the selected item's properties. The specific properties, and your ability to edit them in the **Properties** window, will vary depending on the type of item selected. The tabs available in the **Properties** window vary depending on the selection.

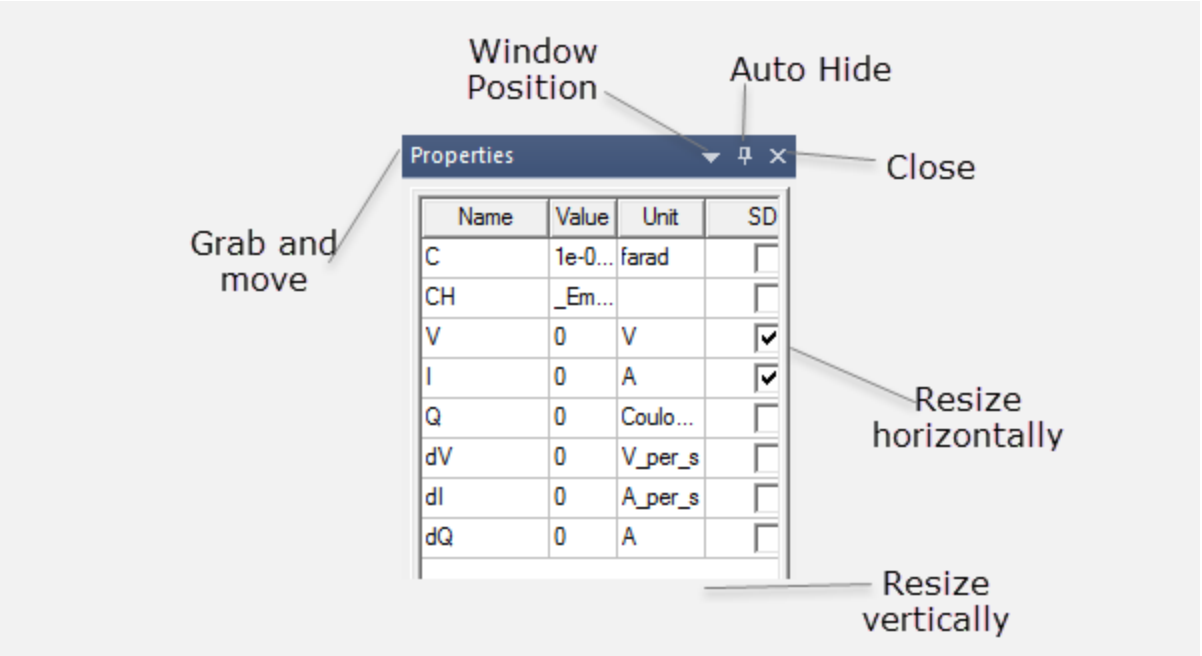
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For example, the **Properties** window for components can have up to five tabs:

- The Param Values Tab
- The General Tab
- The Symbol Tab
- The Quantities Tab
- The Signals Tab

You can move and size the **Properties** window as needed. You can also dock it to any edge of the Twin Builder desktop.

Hints	• Drag the window by its title bar to move it. • Click the push pin in the title bar to toggle between hiding and auto hiding the window. • Resize the window by dragging its edges. • Select to float (undock), dock, auto hide, and hide. Click the triangle in the title bar. • Click the "**X**" in the title bar to close the window.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> The **Show advanced property data** check box on the **Schematic Editor Options: General** tab controls the display of less often used tabs for components, ports and nets.

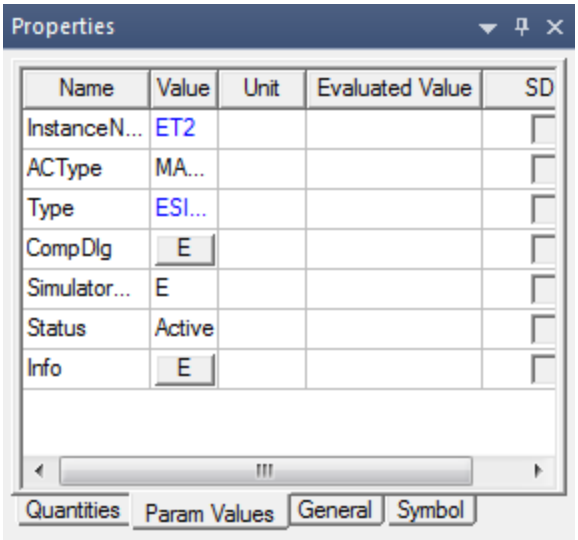
Related Topics

Showing and Hiding the Properties Window

Properties Dialog Box

Param Values Tab (Properties Window)

The **Param Values** tab lists the component's simulation parameters such as the **InstanceName**, **Type**, and **Status**.



- Click the button in the **Value** column for **CompDlg** to open the component special dialog box for entering parameter values.
- Click the button in the **Value** column for **Info** to launch the help topic for the component.
- To set or change the value of a component parameter, click the **Value** field and enter the new value. If the value requires a multiplier unit (such as **kOhm** for "×1000") click the **Unit** field to select the multiplier unit. The **Evaluated Value** shows the resulting number.
- Specify a parameter value using an expression that evaluates to a constant. The expression is retained in the **Value** field, while the **Evaluated Value** field shows the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

constant resulting from evaluating the expression. You can then identify and modify the expression in the future.

- Component parameters with composite properties also have buttons in the **Value** column. A composite property is an array or record property (parameter, quantity, or signal), generally based on a model property, that represents a combination of elements. Click the button to view details and edit these properties in the **Array/Record Element Values** dialog box.

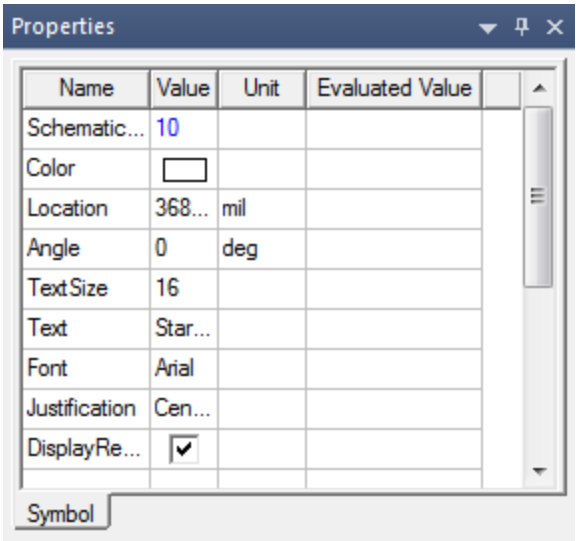
Name	Value	Unit	Evaluated Value	SDB
genrec	{45uf,6e45}			
genrecar	{{0,4},{56,9}}			
genarec	{{3,17},{0,-23}}			
genrecarec	{{{1,2},{3,4}},{{9,8...			
n	2			
InstanceN...	U1			
Simulator...	record_generic_ch...			
Status	Active			

General Tab (Properties Window)

The **General** tab lists the selection's name, symbol name, reference designator, and other data. The information on this tab is read-only.

Symbol Tab (Properties Window)

The **Symbol** tab provides information on a number of modifiable attributes of component symbols, displayed component properties, and primitive drawing elements displayed in the schematic.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The contents of the **Symbol** tab vary depending upon the number and type of components, properties, and drawing elements selected in the schematic.

Symbol tab fields can be edited using the following guidelines:

Component Symbol

- **Component Location** – Click a cell in the **Value** column and enter a new set of X,Y coordinates for the symbol. Click a cell in the **Unit** column to assign a unit of measure to the coordinates. Press **Return** to move the component to the new location.
- **Component Angle** – Click a cell in the **Value** column and select an angle from the drop-down menu. The available choices are 0, 90, 180, and 270 degrees. The symbol rotates counterclockwise as soon as you select an angle.
- **Component Mirror** – Select a check box in the **Value** column to flip the component left-to-right. The mirror operation is performed as soon as you select the box. Clear the check box to return the symbol to its original orientation.
- **Use Symbol Color** – Select a check box in the **Value** column to use the default symbol color for the component. This hides the **Component Color** field and overrides the color set in it.
- **Component Color** — Click the colored bar in the **Value** field to open a palette from which to select a new color for the symbol. The new color is applied when the symbol is unselected.

Displayed Component Properties (Text)

- **PropDisplay Location** — Click a cell in the **Value** column to enter a new set of X,Y coordinates for the property. Click a cell in the **Unit** column to assign a unit of measure to the coordinates. Press **Return** to move the displayed property to the new location.
- **PropDisplay Angle** — Click a cell in the **Value** column and enter an angle. The property rotates counterclockwise when you press Enter or select another element.
- **PropDisplay Font** — Click a cell in the **Value** column and select a font from the drop-down menu. The displayed property font changes as soon as you select the new font.
- **PropDisplay Font** — Click a cell in the **Value** column and enter a font size. Font sizes are in points.
- **PropDisplay Justification** — Click a cell in the **Value** column and select a justification setting from the drop-down menu. The displayed property justification (position relative to **PropDisplay Location**) changes as soon as you select the new setting.

Primitive Drawing Elements

- Select a primitive drawing element in the schematic editor to view and edit its properties. Primitive drawing elements are arcs, circles, lines, rectangles, polygons, text boxes, and images. Editable properties vary with the kind of element. See ***Adding Primitive Drawing Elements*** for additional information on specific property types for each element.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Quantities Tab (Properties Window)

The **Quantities** tab lists the simulation parameters of the selected component.

- To set or change the value of a component parameter, click a cell in the **Value** column and enter the new value.
- If the value requires a multiplier unit (such as **meg** for "×1000000"), click a cell in the **Unit** column to select the multiplier unit.
- To save the parameter values in the project database, select the **SDB** check box.
- Parameters with composite properties have buttons in the **Value** column. A composite property is an array or record property (parameter, quantity, or signal), generally based on a model property, that represents a combination of elements. Click the button to view details and edit these properties in the **Array/Record Element Values** dialog box.

Signals Tab (Properties Window)

The **Signals** tab lists the **Names** and **Values** of signals for digital elements.

Signals with composite properties have buttons in the **Value** column. A composite property is an array or record property (parameter, quantity, or signal), generally based on a model property, that represents a combination of elements. Click the button to view details and edit these properties in the **Array/Record Element Values** dialog box.

To save the parameter values in the project database, select the **SDB** check box.

Showing and Hiding the Properties Window

To toggle display of the **Properties** window:

- Select the **View** menu and click **Property Window**.
- Right-click the status bar at the bottom of the desktop and click **Properties** from the shortcut menu.
- In the docked **Properties** window, click the "x" at the upper right.

> **Note:**
>
> The docked **Properties** window does not contain the **Property Displays** tab. To edit property displays for an element, you must open the **Properties** window for that element from the schematic editor.

Launching Help from the Properties Window

Follow this procedure to launch the help for a component from the **Properties** window:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Click the **Param Values** tab.
2. Scroll down to the **Info** parameter entry and click the button in its **Value** cell.

The Properties Dialog Box

The **Properties** dialog boxes provide a way to show and — where appropriate — add, edit, and remove properties or parameters of:

- Components on a schematic
- Project variables
- Design properties and (local) variables
- Component definitions

Properties dialog boxes can have one or more tabs:

- Quantities
- Signals
- Parameter Values
- General
- Symbol
- Property Displays

On-sheet Component Properties Dialog Box

The on-sheet component **Properties** dialog box shows the properties or parameters of selected objects, and allows editing the values of these properties. It extends the **Properties** window functions with additional editing commands, and with settings for tuning, optimization, sensitivity, and statistical analysis that are not available through the **Properties** window.

To open the **Properties** dialog box for an on-sheet component, right-click the component in the schematic editor and select **Properties**.

On-sheet component properties dialog boxes include one or more of the following tabs, depending on the component and whether the **Show advanced property data** Schematic editor option is selected:

- Parameter Values Tab
- General Tab
- Symbol Tab
- Quantities Tab
- Signals Tab
- Property Displays Tab

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Design Properties Dialog Box

To open the **Properties** dialog box for a design, right-click a design in the Project tree and click **Design Properties**, or click **Twin Builder** > **Design Properties**.

> **Note:**
>
> Editing the design properties of a subcircuit will clear the Undo/Redo history because changes to subcircuit design properties can affect connectivity in the parent schematic.
>
> Editing design properties in a top-level schematic will not clear Undo/Redo.

Design Properties dialog boxes include the following tabs:

- Parameter Defaults Tab
- Local Variables Tab
- General Tab
- Quantities Tab
- Signals Tab

Parameter Defaults Tab (Design Properties Dialog Box)

Use this tab to add, edit, or remove properties, and set the default values for design parameters.

Value Option

Select this radio button to display, and set the default values for these component parameters used during general analysis:

- **Name** –This column displays the names of component parameters. Names can be edited for case only.
- **Value** – This column lets you change the initial values of parameters.
- **Unit** – This column lets you set the unit of measure for the parameter value.
- **Evaluated Value** – This read-only column displays the evaluated value of the parameter.
- **Description** – This column lets you enter a description for the parameter.
- **Netlist Unit** – This column lets you set the unit of measure used when the parameter is netlisted.
- **Callback** – Click **Callback** to associate a callback script with the parameter.
- **Read-only** – This check box controls whether the parameter settings can be changed.
- **Show Hidden** – This check box controls whether the parameter is hidden by default.
- **Property Type** – Displays the type (for example, Real) for generic properties.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Show Pin** – This check box lets you show a pin for the property on the component's symbol.
- **Sweep** – This check box enables the value for sweep.
- **Default SDB** – This check box enables the value to be stored in the project SDB.

Statistics Option

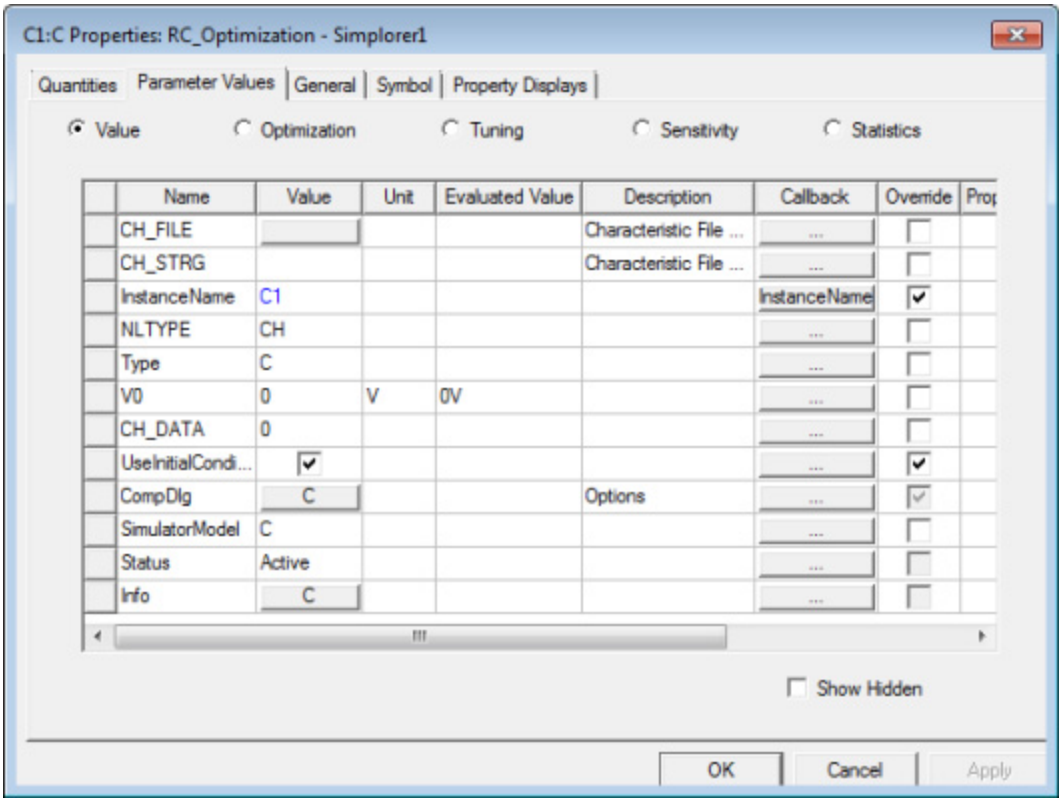
Select this radio button to display and set the values for these design parameters applicable to statistical analysis.

- **Name** – This column displays the names of component parameters.
- **Include** – This check box controls whether the associated component parameter will be varied during statistical analysis. Select **Include** for each parameter you want Twin Builder to vary during statistical analysis.
- **Distribution** – This setting displays and controls whether the parameter value distribution is Uniform, Gaussian, lognormal, or user-defined. To change the **Distribution** setting, click the cell to display and choose the options.
- **Distribution Criteria** – Click the button in this column to open the **Edit Distribution** dialog box in which you can set distribution criteria. The **Distribution Type** drop-down menu displays whether the distribution is Uniform or Gaussian, and you can specify values for Cutoff Probability, Mean, and Tolerance. Units for Mean and Tolerance can be set using their adjacent drop-down menus.

Parameter Values Tab (Properties Dialog Box)

The **Parameter Values** tab lists the simulation parameters of selected components.

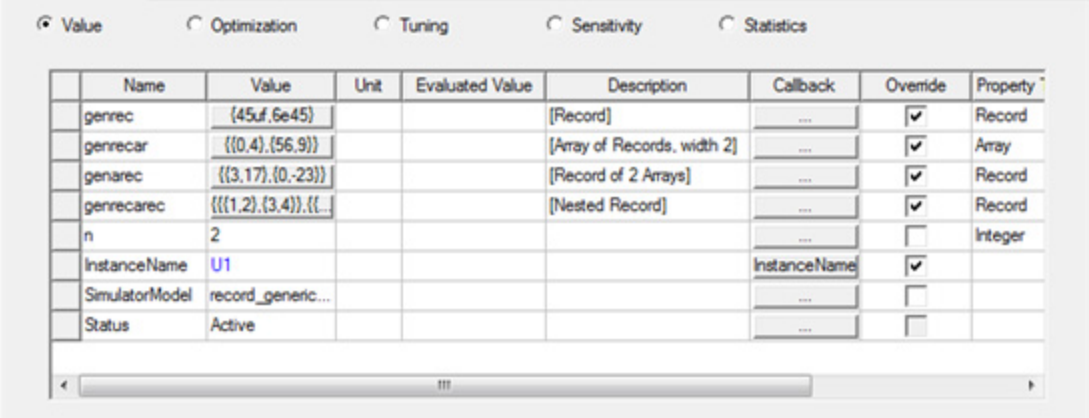
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- Click a cell in the **Value** or **Unit** columns to change the value of a component parameter. If the value requires a multiplier unit (such as **kOhm** for "×1000") click a cell in the **Unit** column to select the multiplier unit. The **Evaluated Value** column shows the resulting number.
- Click the button in the **Value** field for **CompDlg** to open the component special dialog box for entering parameter values.
- Click the button in the **Value** field for **Info** to launch the help topic for the component.
- Component parameters that have composite properties have buttons in the **Value** field. A composite property is an array or record property (parameter, quantity, or signal), generally based on a model property, that represents a combination of elements. An array is a group of the same type of element, such as an array of real quantities, or of std_logic signals. The elements in an array may be composites themselves, but all are the same type. A record is a group of the same, or different, types of element. A record may contain a std_logic signal and a real array. These concepts are generally based on the VHDL-AMS array and record types, and these types of properties currently can only be created from models. Click the button to view details and edit these properties in the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Array/Record Element Values dialog box.



- Specify a parameter value using an expression that evaluates to a constant (shown above). The expression appears in the **Value** field, while the **Evaluated Value** field shows the constant resulting from evaluating the expression. You can then identify and modify the expression in the future. Use the **Property Displays** tab to display the expression, the evaluated value, or both.

> **Note:**
>
> Select the **Show Hidden** check box to view hidden properties of the component. Hidden properties contain system-defined values and rules for interpreting predefined component parameters. Modifying hidden properties requires specialized knowledge of the component, and is not needed for normal operation.

The box in the **Override** column is selected when any of the parameter's default values change. **Property Type** shows the type (for example, Real) for generic properties. **Show Pin** displays a pin for the property on the component's symbol. **Sweep** enables the value for sweep. **SDB** enables the value to be stored in the project SDB.

When you open the **Properties** dialog box for a design (**Design Properties**), the **Parameter Values** tab is initially empty. Click **Add** to add properties to the design. See ***Defining a Project Variables*** for details.

> **Note:**
>
> The **Add** button does not appear on the **Properties** dialog box for a component.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

General Tab (Properties Dialog Box)

The **General** tab lists the selection's name, symbol name, reference designator, and so on. The information on this tab is read-only.

Symbol Tab (Properties Dialog Box)

The **Symbol** tab provides information on the location of the component symbol in the schematic. The information is identical to that on the Symbol tab of the **Properties** window.

When the **Properties** dialog box is opened for a design (**Design Properties**), both the General and Symbol tabs are initially empty.

Quantities Tab (Properties Dialog Box)

The **Quantities** tab lists the simulation parameters of the selected component. Quantities that have composite properties have buttons in the **Value** column. A composite property is an array or record property (parameter, quantity, or signal), generally based on a model property, that represents a combination of elements. Click the button to view details and edit these properties in the **Array/Record Element Values** dialog box.

The tab information is similar to that on the **Quantities** tab of the **Properties** window with these additions:

- **Description** – Displays a description of the parameter.
- **Callback** – Click **Callback** to associate a callback script with the parameter.
- **Override** – Select this check box to indicate that one or more of the parameter's defaults have changed.
- **Direction** – Shows the parameter direction (In, Out, InOut, or Don't Care).
- **Show Pin** – Select this check box to control whether the parameter pin appears on a schematic.
- **Sweep** – Select this check box to control whether the parameter may be swept.

Signals Tab (Properties Dialog Box)

The **Signals** tab lists the names and values of signals for digital elements.

- **Name** – Displays the names of component parameters.
- **Value** – Lets you change the initial values of parameters.

  Signals that have composite properties have buttons in the **Value** field. A composite property is an array or record property (parameter, quantity, or signal), generally based on a model property, that represents a combination of elements. Click the button to view details and edit these properties in the **Array/Record Element Values** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

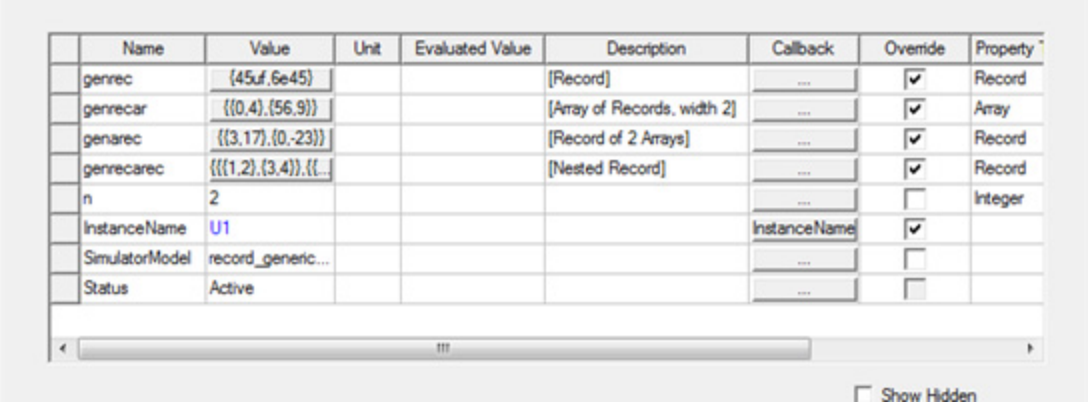
- **Unit** – Lets you set the unit of measure for the parameter value.
- **Description** – Displays a description of the parameter.
- **Callback** – Lets you associate a callback script with the parameter.
- **Override** – When selected, indicates that one or more of the parameter's defaults have changed.
- **Signal Type** – Shows the type of signal (for example, real or bit) set for the parameter.
- **Direction** – Shows the parameter direction (In, Out, InOut, or Don't Care).
- **Show Pin** – Select this check box to control whether the parameter pin appears on a schematic.
- **Sweep** – Select this check box to control whether the parameter may be swept.
- **SDB** – Select this check box to control whether the parameter is saved in the SDB.

Array/Record Element Values Dialog Box

A component properties dialog box may include array or record properties, generically called *composite properties*, for parameters, quantities, or signals.

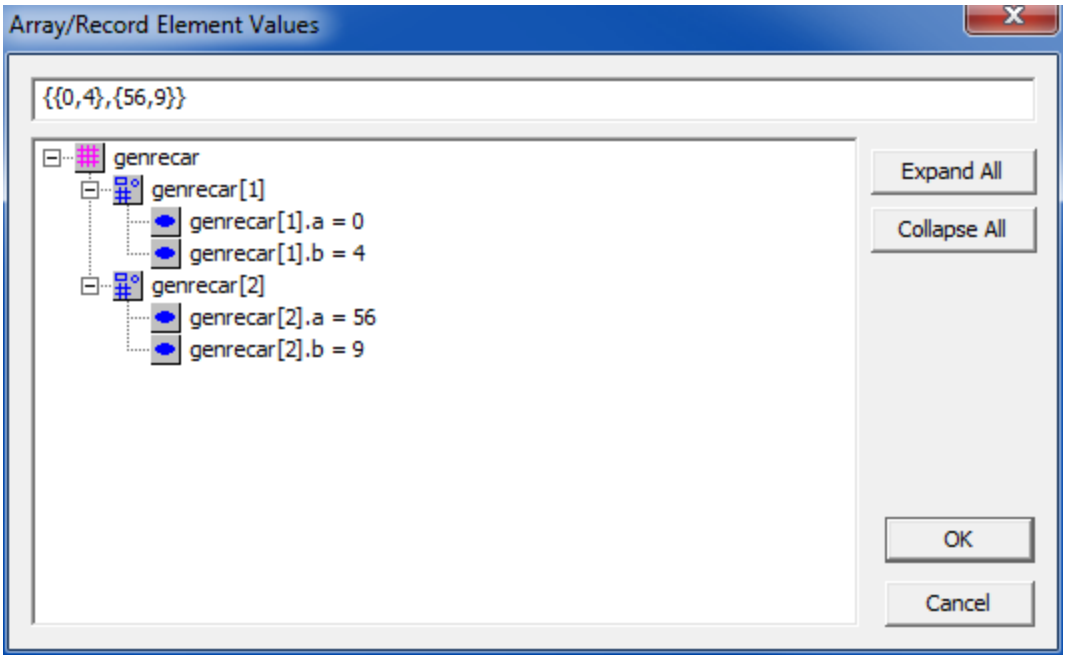
- A composite property is an array or record property (parameter, quantity or signal), generally based on a model property, that represents a combination of elements.
- An array is a group of the same type of element, such as an array of real quantities, or of std_logic signals. The elements in an array may be composites themselves, but all are the same type.
- A record is a group of the same or different types of element. A record may contain a std_ logic signal and a real array.
- These concepts are generally based on the VHDL-AMS array and record types, and these types of properties can only be created from models at this time.

The following image contains four composite properties.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

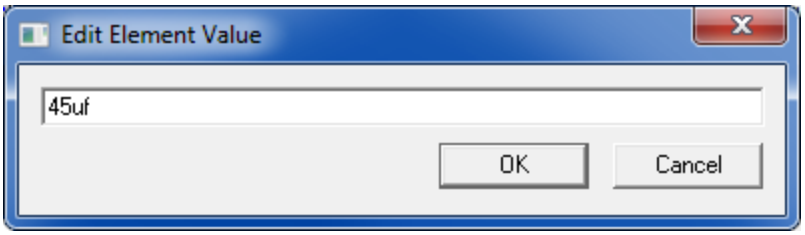
Note that the **Value** field for each composite property contains a button labeled with the values of the elements in the respective array or record. Click a **Value** button to open an **Array/Record Element Values** dialog box, in which you can assign element values for array and record properties. For example:



The edit field at the top shows the current composite value, represented as a comma-separated set of numbers, variables, functions, or expressions enclosed in curly braces. Nested arrays or records (an array of records where the record has an array and various other values) are also enclosed in curly braces. Such a value could look like {1,3,{{22,55},44},7e12}. Modify the current value in the edit field to set the array or record value. The updated values will be shown in the tree when focus is changed.

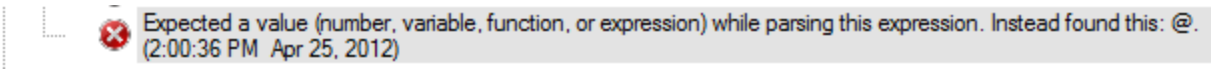
Click **Expand All** and **Collapse All** to expand and collapse the entire tree at one time.

Click a leaf element to open the **Edit Element Value** dialog box:



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Enter a value then click **OK**. The value is validated and updated in both the tree and composite view. Entries that fail the validation check produce messages similar to the following in the **Message Manager** pane:



Related Topics

Parameter Values Tab (Properties Dialog Box)

Quantities Tab (Properties Dialog Box)

Signals Tab (Properties Dialog Box)

Param Values Tab (Properties Window)

Quantities Tab (Properties Window)

Signals Tab (Properties Window)

Property Displays Tab (Properties Dialog Box)

The **Property Displays** tab controls the display of component properties on the schematic. Displayed properties appear as text labels on the schematic.

> **Note:**
>
> The **Property Displays** tab controls only the display of labels associated with component parameters. It does not enable or define those parameters as outputs for simulation. For more information on selecting output parameters, see *Setting the Outputs for Simulation*.

To add a property to those displayed on the schematic:

1. Click **Add**. A new row appears.
2. Click the **Name** field and select the parameter you want to add from the drop-down menu.
3. Click the **Visibility** field to select the type of information you want to display:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **None** — No label displayed.
- **Name** — Only the parameter name is displayed (for example, **R**).

> **Note:**
>
> The output property displays of types **Value**, **Both**, **Evaluated Value** and **Evaluated Both** support animation during simulation to display the current value. However, this is not supported for input properties as their value is specified by the users.

- **Value** — Displays the component value. This can be a single value (for example, 10000) or an expression (for example, 10000*5).
- **Both** — Displays the parameter name and its value, for example, R = 10000 or R=10000*5.
- **Evaluated Value** – Displays the evaluated value of an expression that has been used for the value of a parameter, for example, 50000 for the expression 10000*5.
- **Evaluated Both** – Displays the parameter name and the evaluated value, for example, R = 50000 for the expression 10000*5.

4. Click the **Location** field for a parameter to specify the location for the displayed parameter and/or value. The locations are **Left**, **Top**, **Right**, **Bottom**, **Center**. When you have set the location with the cursor in the schematic, the **Location** field has the entry **Custom**.

To remove a property from display on the schematic:

- Select the property you want to remove from display in the **Property Display** list and click **Remove**.
- Click the **Visibility** field of the property you want to remove from display and select **None**.

Add/Edit Property Dialog Boxes

In the **Add Property** dialog box, you can enter a property **Name**, enter an initial **Value**, and choose a property type with the radio buttons (text input, menu, check box, and so on). Depending on the property type, you can also choose a **Unit Type** (such as angle, frequency, and capacitance) and the default **Units** (deg, GHz, pF, and so on) for the property from drop-down lists.

An informational panel provides guidance based on the chosen value type for entering data in the various fields.

> **Warning:**
>
> Do not use the names of reserved system parameters for property names.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The **Edit Property** dialog box is similar to the **Add Property** dialog box. Editable fields are the same as those described above.

The Message Manager

The **Message Manager** window displays messages associated with a project's development, such as error messages about the design's setup or informational messages about the progress of an analysis.

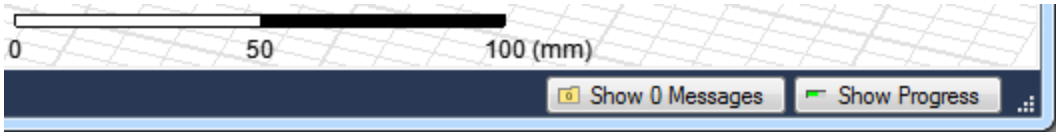
Messages are organized by project, then by circuit design. Because a design can contain multiple top-level and subcircuits, and multiple analyses can be set up for each, this organization helps you determine where errors have occurred.

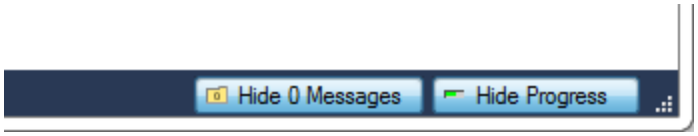
The **Message Manager** window can be moved and sized as needed. It can also be docked to any edge of the Twin Builder desktop. Message text wraps within the window. A vertical scrollbar displays as needed allowing you to move through the list message-by-message.

Note	If a message is too large to fit in the window, you cannot click the scrollbar to page-up or page-down. To view such messages in their entirety, you can resize the window or right-click the message and select **Details** to show message details.
Hints	• Drag the window by its title bar to undock, move, and dock it. • Resize the window by dragging its edges. • Click the pin icon in the title bar to auto-hide the window. • Click **X** to close the window. • Right-click a message and select **Details** to show message details.

To display or hide the **Message Manager**:

- Click **View** > **Message Manager** or
- Click **Show Messages** or **Hide Messages** on the status bar.





Messages in the **Message Manager** window are organized with global messages first, then by project, then by circuit. Because a design can contain multiple circuits and subcircuits,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

sometimes with multiple analyses for each, this organization helps you to quickly determine where errors have occurred. Messages are stamped with the date and time (hh:mm:ss) they were generated. The following icons appear next to a message to indicate information, warnings, errors, or actions:

	An informative message.
	A warning message that may require your attention.
	An error message that may require your attention.
	An action associated with the message. Click the message to invoke the action (the cursor will change to a hand icon when it is placed over the action message).

Related Topics

- Hiding the Message Manager Window Until Messages Appear
- Clearing Messages
- Showing Message Details

Hiding the Message Manager Window Until Messages Appear

If you prefer to hide the **Message Manager** pane until a message is generated:

1. Turn off the **Message Manager** pane by clearing the **Message Manager** check box on the **View** menu.
2. Select **Tools** > **Options** > **General Options** The **Options** dialog box appears.
3. On the **General** > **User Interface** tab, select **Show Message Window on new messages**.

The **Message Manager** pane will re-open when Twin Builder reports any errors, warnings, or successful completion of any simulations.

Clearing Messages

The **Message Manager** window is cleared at the start of each analysis. To manually clear messages, right-click the message tree and select **Clear Messages** for <*ProjectName*>.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

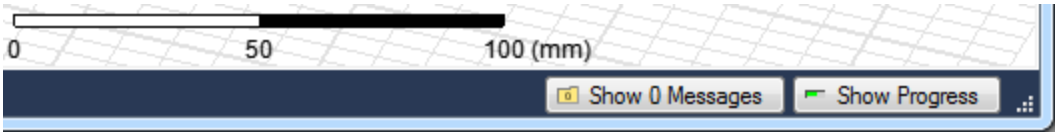
Showing Message Details

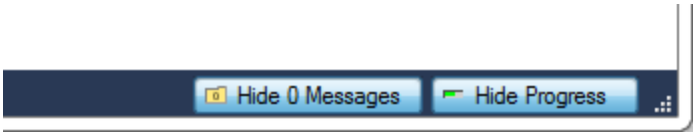
To display detailed information about individual messages in the **Message Manager** window, right-click a message and select **Details** from the context menu. The **Message Details** dialog box displays the **Project** and **Design** to which the message applies, and a detailed **Description** of the message.

The Progress Window

To display or hide the **Progress** window:

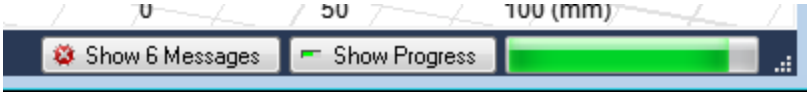
- Click **Show Progress** or **Hide Progress** on the status bar:





- Click **View** > **Progress Window**.
- Right-click the history tree, then click **Progress** from the shortcut menu.

When more than one progress bar is active, the top progress bar is represented on the status bar with a progress indicator.



To configure the **Progress** window to appear only when a simulation is running:

1. Click **View** > **Progress** to turn off the **Progress** window display.
2. Click **Tools** > **Options** > **General Options**. The **Options** dialog box appears.
3. On the **General** > **User Interface** tab, select **Show Progress Window when displaying progress bars**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Click **OK**.

> **Note:**
>
> The **Progress** window is also a dockable window, so you can position it where you like.

The Design Area

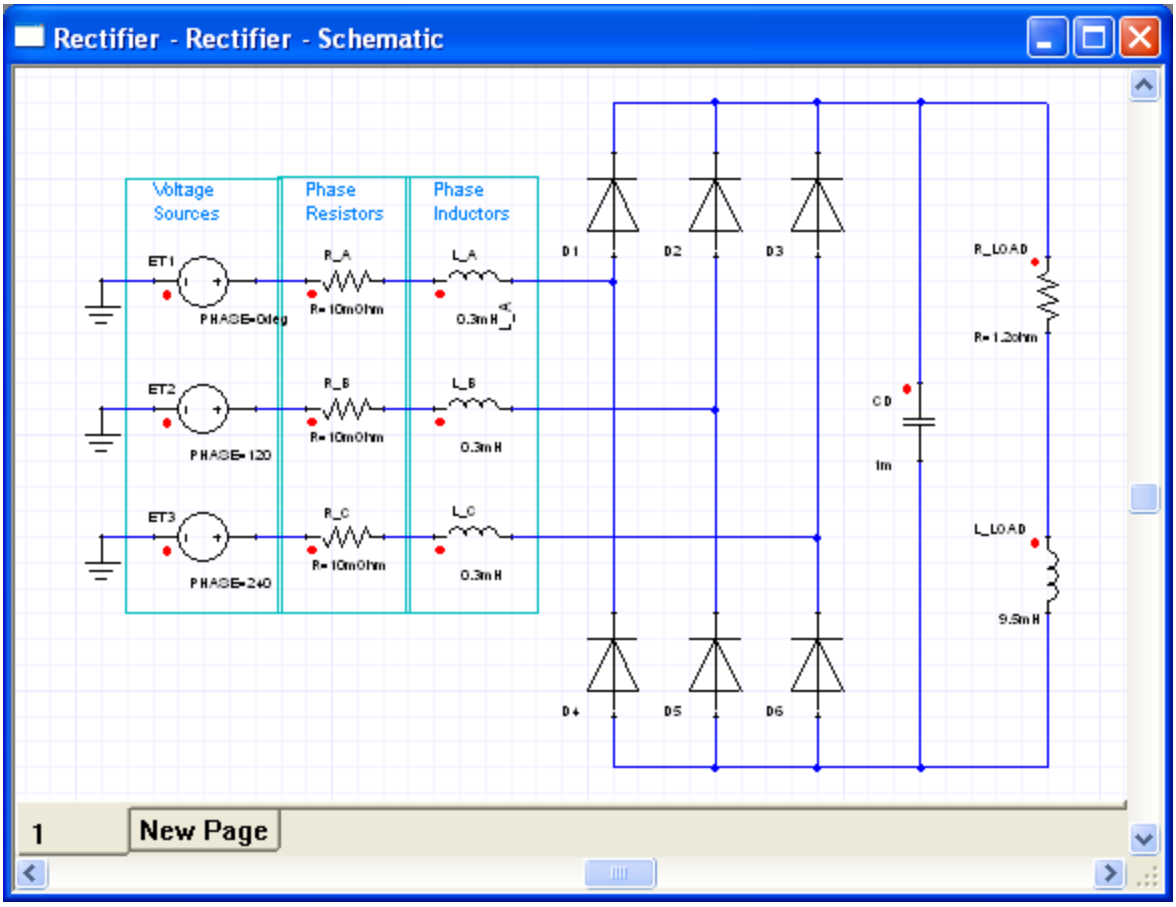
The Design Area can display one or more editor windows and report windows. This section briefly introduces the various Design Area windows.

- Schematic Editor Window
- Symbol Editor Window
- Netlist Editor Window
- Model Editor Window
- Script Editor Window
- Report Window

Schematic Editor Window

The Schematic Editor window lets you place components and wire them together.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Click and drag components to move them, or copy and paste components and their wires within the schematic editor. You can also copy and paste to other schematics.

As you place the cursor near a pin of a component, it changes from an arrow to an **X**, indicating that the schematic editor is in the wiring mode. In wiring mode, left-click to start drawing a wire. Left-click again to end the wire.

Click the toolbar icons of commonly used items such as ports, grounds, and page connectors to place them in the schematic. You can also select and place them using the **Draw** menu.

View controls to zoom in, zoom out, and fit the drawing to the editor window are available on the **View** menu, and on the shortcut menu that opens when you right-click a schematic.

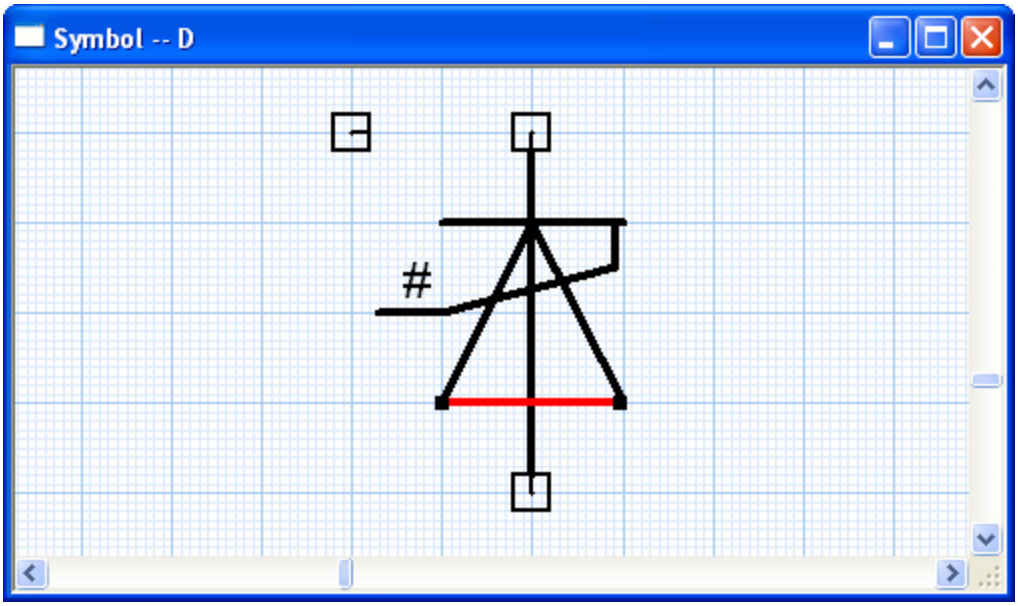
Related Topics

Schematic Editor

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Symbol Editor Window

The Project tree **Definitions** includes a **Symbols** folder that lists the symbols for components used on the current schematic. Double-click the desired item, or right-click and select **Edit Symbol** to open the symbol for editing.



The symbol editor lets you:

- Draw graphical primitives such as rectangles, circles, and arcs, with the Draw menu.
- Add pins for electrical connections using the Pin option on the Draw menu, and possibly modifying the properties of these pins with the Pin List dialog box.
- Add text labels, using the Text option on the Draw menu.
- Add property displays, using the Property Display Setup option on the Symbol menu.
- Update the current project with the new or revised symbol definition using the Update Project option on the Symbol menu.

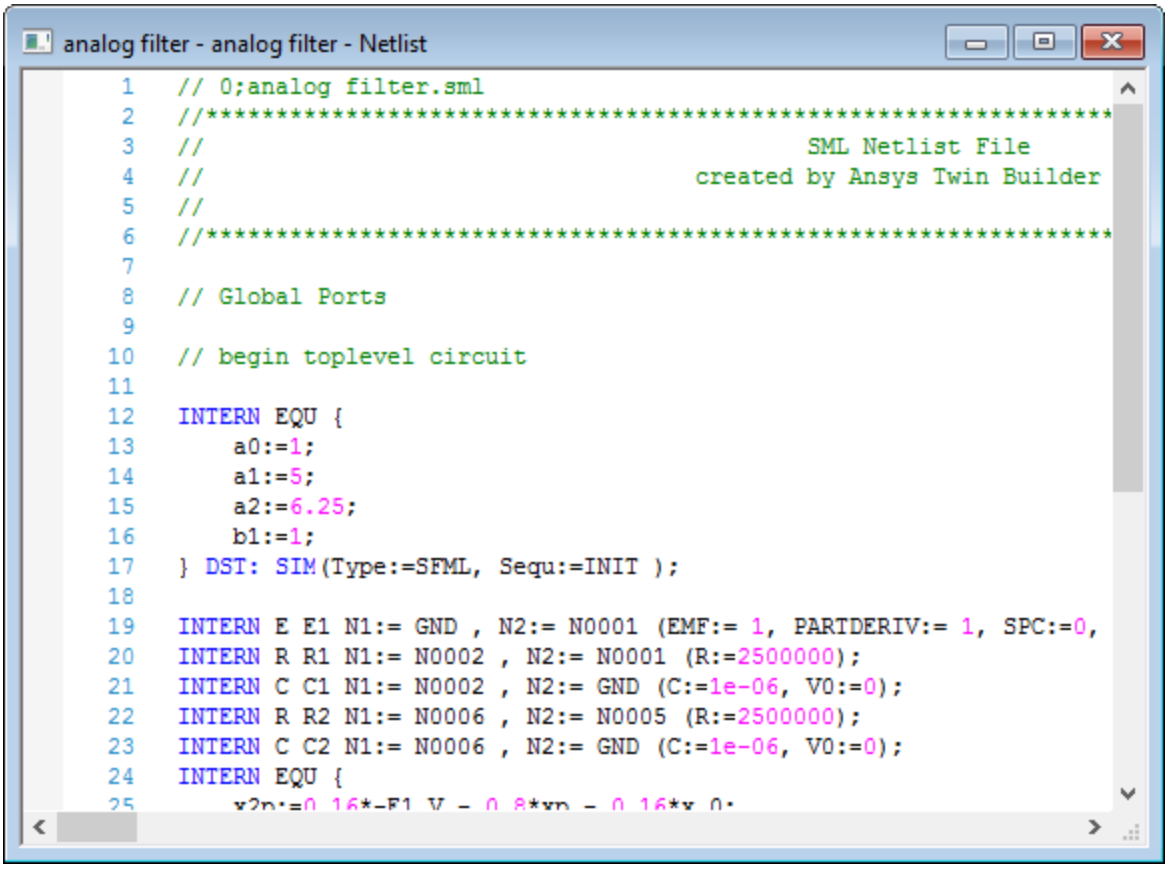
Related Topics

- Symbol Editor

Netlist Editor Window

The Netlist Editor lets you view the netlist generated from the schematic. Click **Twin Builder** > **Browse Netlist** to open this window.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



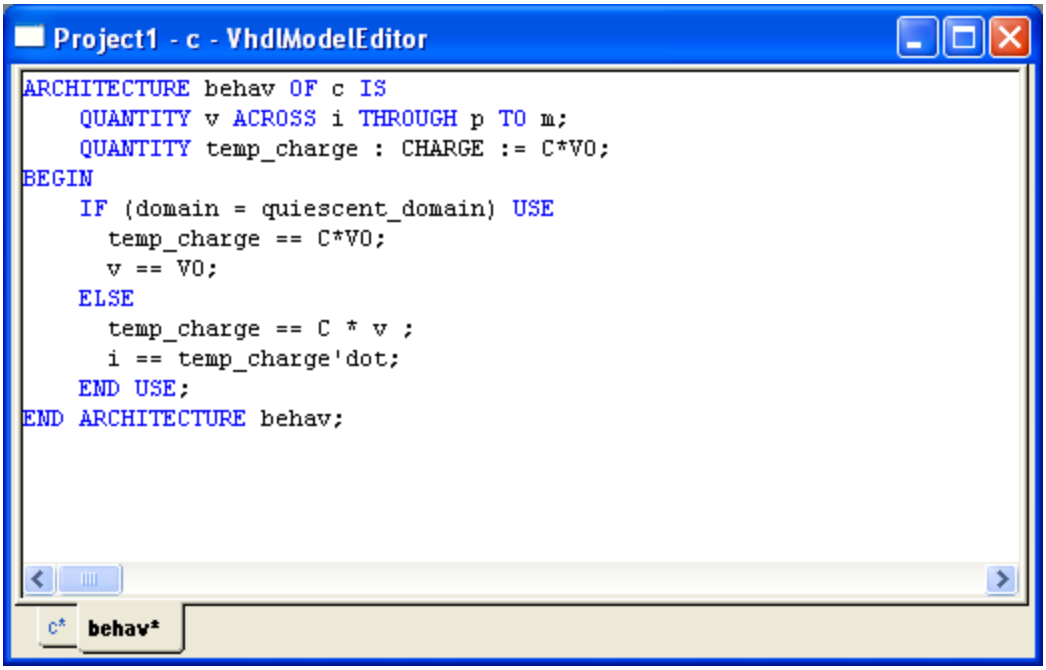
Related Topics

Netlist Editor

Model Editor Window

Edit Twin Builder elements modeled in SML, VHDL, C, and Spice in the model editor window. A typical VHDL model editor window is shown below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Related Topics

C-Model Editor

VHDL Model Editor

SML Model Editor

Spice Model Editor

Script Editor Window

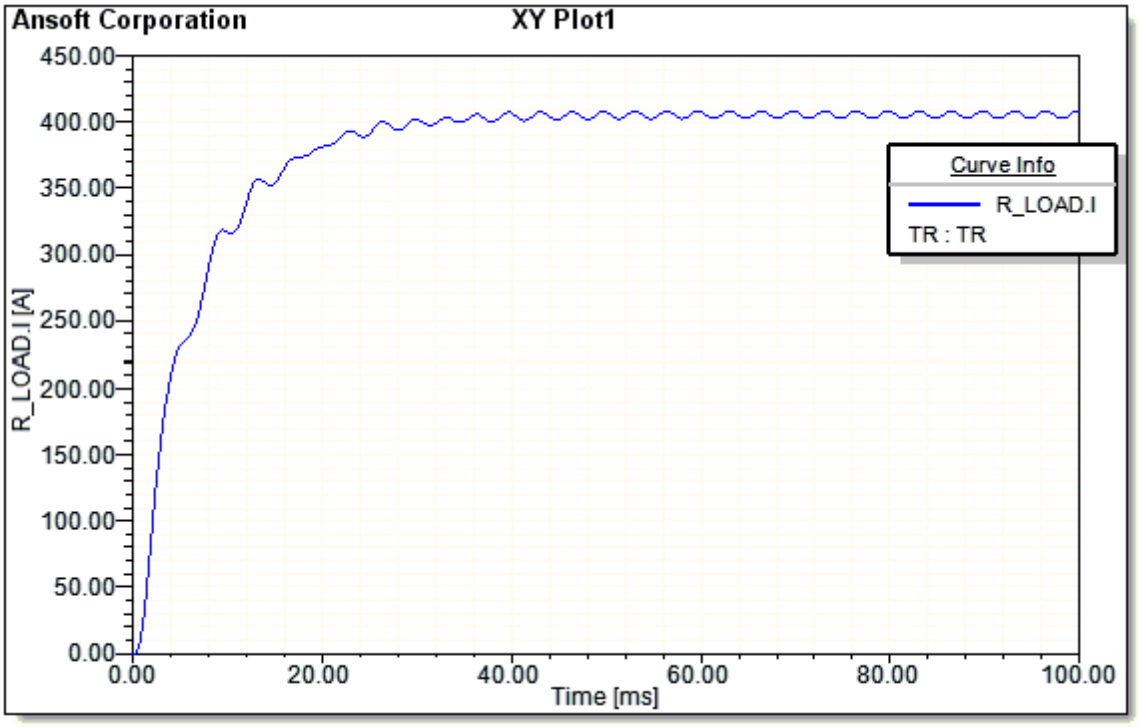
The **Script Editor** window lets you edit Java and Visual Basic scripts.

For details on working with scripts, see Scripting.

Report Window

When a design has been successfully simulated, you can generate a report of the results in a wide variety of forms, including XY graphs, polar graphs, 3D graphs, Smith charts, and data tables. Various attributes of each can be customized to your liking. Here is an example of a 2D report:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



For more information, see Generating Reports and Postprocessing.

Window Layouts

Create and save specific window layouts for the desktop and apply them when needed. For example, you can create a layout that shows a large component window. There are options to apply a saved layout, restore the default window layout, and remove saved layouts.

To save a window layout:

1. Click **View** > **Docking Window Layouts** > **Save Current Layout**. The **Layout Name** dialog box appears.
2. Enter the layout name and click **OK**. The layout is saved and added to the list.

To apply a saved layout:

1. Click **View** > **Docking Window Layouts** and choose a layout from the list.

To restore the default window layout:

1. Click **View** > **Docking Window Layouts** > **Default**.

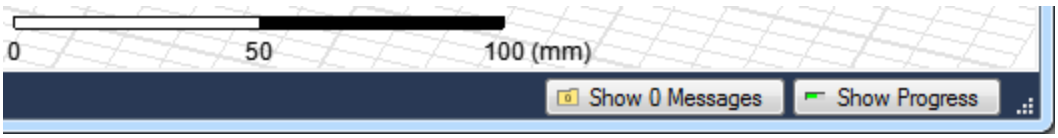
To remove window layouts:

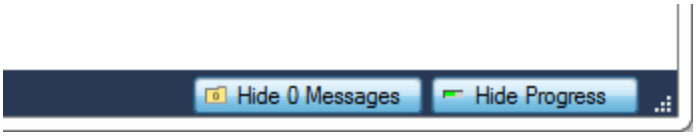
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Click **View** > **Docking Window Layouts** > **Remove Saved Layouts**. The **Modify Layouts** dialog box appears.
2. Select one or more layouts and click **Delete**, or click **Delete All**.

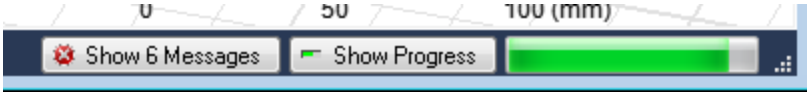
The Status Bar

The status bar is located at the bottom left of the Twin Builder and SheetScan application windows. It displays information about the command currently being executed; it also contains buttons to show or hide the **Message Manager** and **Progress** windows.

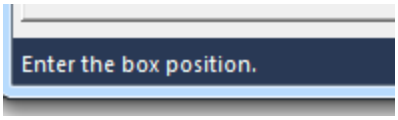




When more than one progress bar is active, the top progress bar is represented on the status bar with a progress indicator.



It also displays information about the command currently being performed. Directions for inputs appear on the left:



Desktop Menus

The following topics describe the menu bar options and other useful features of the Twin Builder desktop.

Top Menu Bar

Shortcut Keys

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Working with the Menu Bar

The menu bar contains drop-down menus for controlling Twin Builder and the various editors and viewers.

From left to right, Twin Builder contains the following menus:

File	Manage Twin Builder project files and printing options.
Edit	Select, cut, copy, paste, and modify the contents of the active editor window; and undo and redo actions.
View	Display or hide desktop components and model objects, modify schematic editor visual settings, and modify the model view.
Project	Add a Twin Builder design to the active project, view, define datasets, and define project variables.
Draw	Draw primitive elements, ports, 2D reports, and wires, and to rotate and flip schematic and symbol elements.
Schematic	Set up the grid display, page display properties, perform ERC check, set rules for naming wires, edit the symbol for a circuit, push up and pop down through levels of a hierarchical schematic, and layout probes and stacking. *This menu does not appear unless you have an active schematic displayed.*
Twin Builder	Set up and manage all the parameters for the active design. Most of these project properties also appear in the Project tree. You can also start an analysis, set up an optimetrics analysis, and create reports. *This menu does not appear unless you have an active schematic displayed.*
Symbol	Manage the parameters for the symbols associated with components and to update the current project with changes made in the current symbol. *This menu does not appear unless you have an active symbol editor window displayed.*
Netlist	Export the netlist for a design. *This menu does not appear unless you have an active netlist editor window displayed.*
Script	Manage Java and Visual Basic scripts. *This menu does not appear unless you have an active script editor window displayed.*
VHDL, **C**, and **SML Model Editor**	Manage the parameters for component models defined using either the VHDL, C, or SML modeling languages, and to update the current project with changes made in the current model. *These menus do not appear unless you have an active model editor window displayed.*

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Report2D and **Report3D**	Manage the parameters for modifying and updating the various report types available in Twin Builder. *These menus do not appear unless you have selected a report from the Results folder in the Project Manager.*
Tools	Modify the active project's material library, arrange the material libraries, run and record scripts, update project definitions from libraries, customize the desktop's toolbars and keyboard shortcuts, and modify many of the software's default settings.
Window	Rearrange the schematic windows and toolbar icons.
Help	Access the help system and view the current Twin Builder version information.

External User Tools

Follow this procedure to add an external user tools menu to Twin Builder:

1. Select **Tools** > **External Tools**. The **Customize User Tools Menu** dialog box appears.

   > **Note:**
   >
   > - If **User Tools** menu items have been defined, those items appear in the **Menu Contents** area.
   > - Click **Move Up**, **Move Down**, **Remove**, and **Add** to navigate menu items.

2. Click **Add** to enable these fields:
   - **Menu Text** – Enter the name of the tool you want to appear in the **Tools** menu.
   - **Command** – This field displays the external executable. Click to navigate to the location of the executable file you want to run when the command is selected.
   - **Arguments** – This field accepts command arguments from the **>** menu selections for **File Path**, **File Directory**, **File Name**, **File Extension**, **Project Directory**, or **Temp Directory**.
   - **Initial Directory** – This field specifies the initial directory for the command to operate.

     Click to navigate folders on your system, or across the network.
3. Click **OK** to add the newly defined user tools to the **Tools** menu, or **Cancel** to close the dialog box without changes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> User tools appear at the bottom of the **Tools** menu.

Shortcut Keys

Many commands in Twin Builder may be accessed through keyboard shortcuts. Use these keystroke combinations to bypass the menu system and directly execute commands. They are designated and chosen as follows:

Modifier + **Key**	• Hold down the modifier (such as **Shift** or **Ctrl**) and press the key. • Type all shortcut keys in lower case.
Modifier + **MMB** + **drag**	• Hold down the modifier (such as **Shift** or **Ctrl**), press and hold the **middle** mouse button, and **drag** the mouse.

Twin Builder Desktop shortcut keys

Ctrl + **n**	New
Ctrl + **o**	Open
Ctrl + **p**	Print
Ctrl + **s**	Save
Ctrl + **0**	Cascade windows
Ctrl + **1**	Tile windows horizontally
Ctrl + **2**	Tile windows vertically
Del	Delete
F1	Open help (context-sensitive)
F2	Rename
F12	Analyze - Run simulation using the active analysis setup

Symbol Editor shortcut keys

Ctrl + **a**	Select all elements
Ctrl + **c**	Copy selected element

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ctrl + **v**	Paste
Ctrl + **x**	Cut selected element
Ctrl + **y**	Redo Edit/Delete
Ctrl + **z**	Undo Edit/Delete
Del	Delete
Ctrl + **t**	Insert text
Ctrl + **b**	Insert pin
Ctrl + **r**	Rotate selected elements
Ctrl + **MMB** + **drag**	Pan view
Shift + **MMB** + **drag**	Zoom view

Report 3D Viewer shortcut keys

Ctrl + **d**	Fit drawing
MMB + **drag**	Rotate view
Ctrl + **MMB** + **drag**	Pan view
Shift + **MMB** + **drag**	Zoom view

Report 2D Viewer shortcut keys

Ctrl + **a**	Select all traces
Ctrl + **c**	Copy selected trace
Ctrl + **v**	Paste
Ctrl + **x**	Cut selected trace
Ctrl + **y**	Redo report action
Ctrl + **z**	Undo report action
Del	Delete
Ctrl + **MMB** + **drag**	Pan
Shift + **MMB** + **drag**	Zoom

Netlist Editor, VHDL-AMS Package Editor, VHDL-AMS Editor, SML Editor, and C-Model Editor shortcut keys

Ctrl + **a**	Select all traces

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ctrl + **b**	Delete all bookmarks
Ctrl + **c**	Copy
Ctrl + **f**	Find
Ctrl + **g**	Go to line number
Ctrl + **r**	Replace
Ctrl + **v**	Paste
Ctrl + **x**	Cut
Ctrl + **y**	Redo
Ctrl + **z**	Undo
Del	Delete
F2	Next bookmark
Ctrl + **F2**	Toggle bookmark
Shift + **F2**	Previous bookmark

Note:

For view navigation shortcuts, specifically those involving the middle mouse button (MMB):

- The modifier keys listed on this page are the default view navigation assignments. Optionally, you can choose to use the legacy shortcuts that were appliable to version 2023 R2 and earlier. See Choosing the View Navigation Options for more information.

Related Topics

Schematic Editor Shortcuts

Choosing the View Navigation Options

New mouse button and modifier key assignments (hotkeys) have been added for navigating the model view (rotate, pan, and zoom functions). Simultaneously, legacy hotkeys for these functions continue to be supported by default. Optionally, you can choose to disable the legacy view navigation shortcuts that were applicable to version 2023 R2 and earlier, as follows:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Access the *Options* dialog box using one of the following two methods:
   - From the menu bar, click **Tools** > **Options** > **General Options**.
   - From the **Desktop** ribbon tab, click **General Options**.
2. In the tree at the left side of the dialog box, select **General** > **User Interface**. Then:
   - To use the current (default) view navigation behavior, ensure that **Enable Legacy View Navigation** is cleared.
   - To use the legacy view navigation behavior, select **Enable Legacy View Navigation**.
3. Click **OK**.

   The navigation option change becomes effective immediately (no program restart required).

Current vs. Legacy View Navigation

The following table summarizes changes in mouse button and hotkey assignments between the current scheme (version 2024 R1 and newer) and the legacy scheme. "*Legacy*" assignments are applicable to version 2023 R2 or earlier and when the *Enable Legacy View Navigation* option is selected in newer versions):

> **Note:**
>
> The *"Current"* assignments are available whether or not the *Enable Legacy View Navigation* option is selected:

Function	Key / Mouse Button Assignment	
	Current	**Legacy**
Pan (all contexts)	Ctrl + MMB + drag or Ctrl + *arrow keys*	Shift + LMB + drag or Ctrl + *arrow keys*
Rotate (3D contexts)	MMB + drag or Alt + *arrow keys*	MMB + drag or Alt + LMB + drag or Alt + *arrow keys*

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Function	Key / Mouse Button Assignment	
	Current	Legacy
Zoom (all contexts)	Wheel or Shift + MMB + drag or Ctrl + "+", Ctrl + "-"	Wheel or Alt + Shift + LMB + drag or Ctrl + "+", Ctrl + "-"
• "**LMB**" = Left Mouse Button • "**MMB**" = Middle Mouse Button • "**Wheel**" refers to rotation of the mouse wheel. • In some 2D contexts, where rotation is not applicable, you can use **MMB + drag** to pan the view for either scheme. • For **Alt + *arrow keys***, rotation is about a vertical or horizontal axis, regardless of the model view orientation. • For zooming, if using the "=\+" key, instead of the "+" key on the numeric keypad, do **not** press Shift.		

Editing Keyboard Shortcuts

Use the **Keyboard Shortcuts** dialog box to create and edit shortcut keys. You can **Save** and **Load** customized keyboard shortcut sets, and **Restore** all keyboard shortcuts to their default settings.

To edit keyboard shortcuts:

1. Select **Tools** > **Keyboard Shortcuts**. The **Keyboard Shortcuts** dialog box appears.
2. Select a command from the **Categories** and **Commands** lists. The **Shortcuts for selected command** drop-down list shows the keyboard shortcuts assigned to the selected command.
   a. To delete a keyboard shortcut, select it in the drop-down list and click **Remove**.
   b. To add a keyboard shortcut, place the cursor in the **Press new shortcut key** box and type the new key or key combination (for example, **Shift**+**Esc**, **F7**, **Ctrl**+**Shift**+**K**).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> If the new shortcut key is already used, **Show currently used by** lists which command currently uses it.

c. Click **Assign** to create the new shortcut.

3. Click **Save** to save the modified set of keyboard shortcuts in an Ansoft Keyboard Shortcut (.**aks**) file.

Undo and Redo Commands

Select **Edit** > **Undo** to undo the last command or operation you performed. **Redo** re-executes the last operation that was undone.

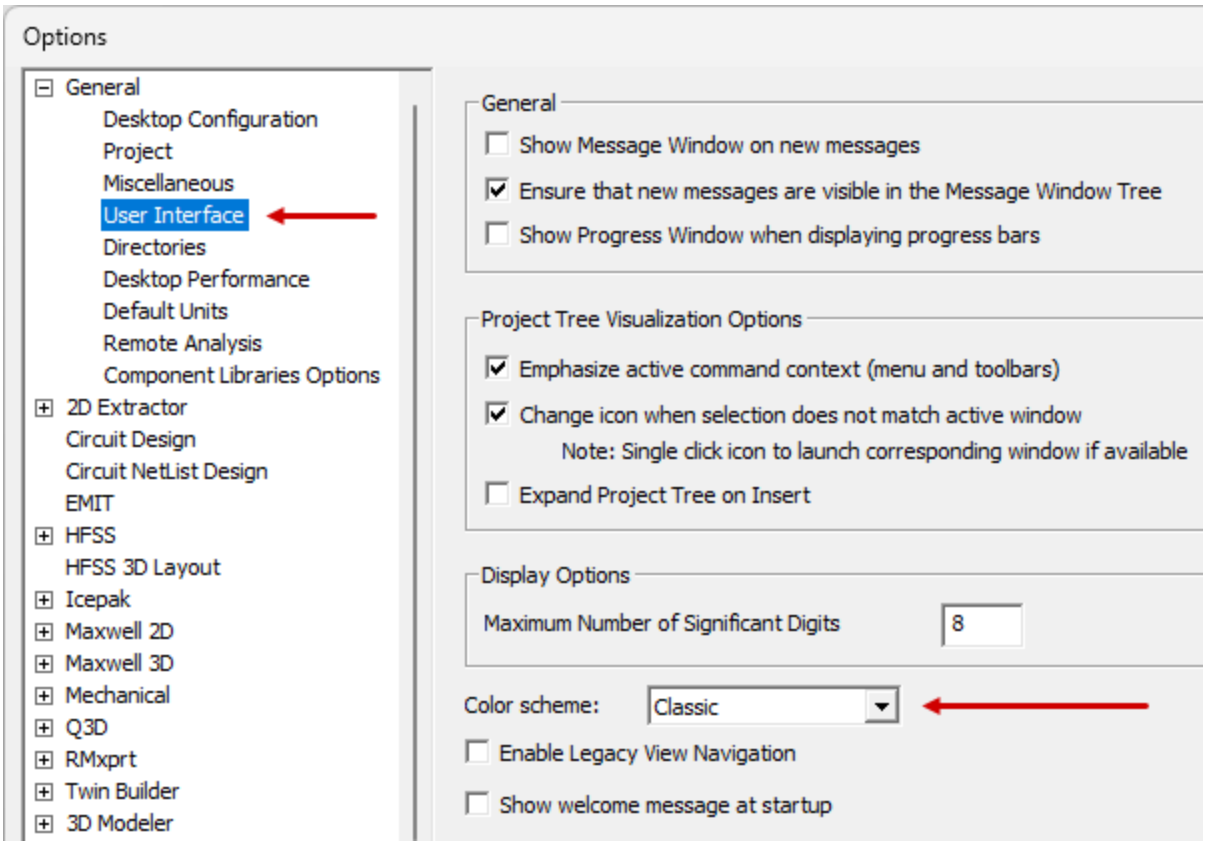
> **Note:**
>
> - Undo/redo is available for all editors and reports.
> - You cannot undo an analysis that has been performed.
> - When you save a project, the undo/redo history is cleared for the project and its designs.

Choosing a Color Scheme [Beta]

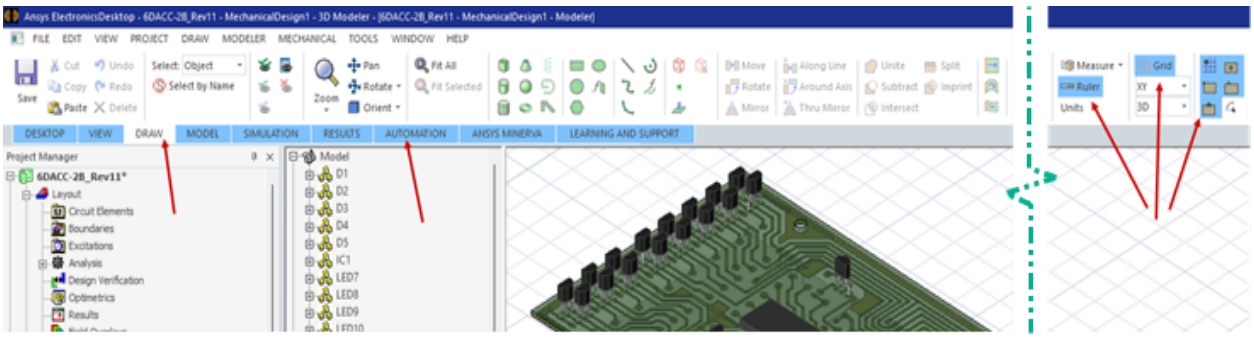
In addition to the classic Twin Builder color scheme, two additional color schemes are now available as beta features. Choose the desired color scheme as follows:

1. Access the *Options* dialog box using one of the following two methods:
   - On the **Desktop** ribbon tab, click **General Options**.
   - From the menu bar, click **Tools** > **Options** > **General Options**.
2. In the tree at the left side of the dialog box, expand **Desktop Configuration** and select **User Interface**:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

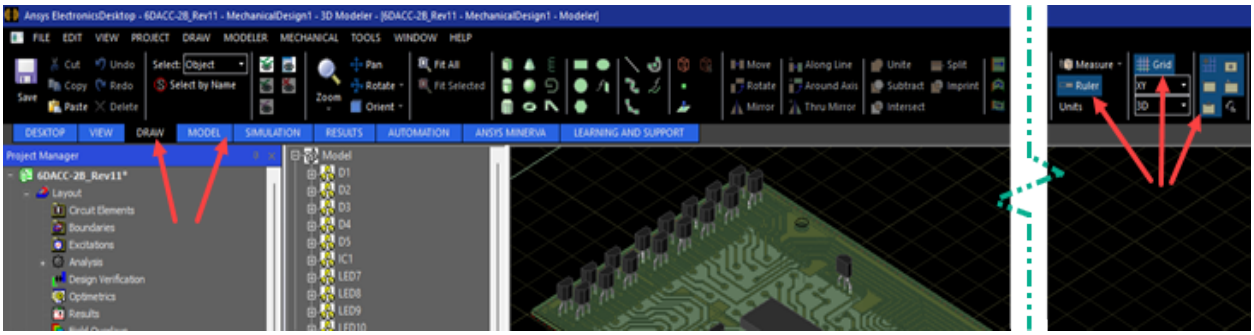


3. From the **Color scheme** drop-down menu, choose one of the following three options:
   - **Light (beta):** A light gray and white scheme with light blue highlighting for selected ribbon icons and for inactive ribbon tabs. The active ribbon has a white background (including the tab):

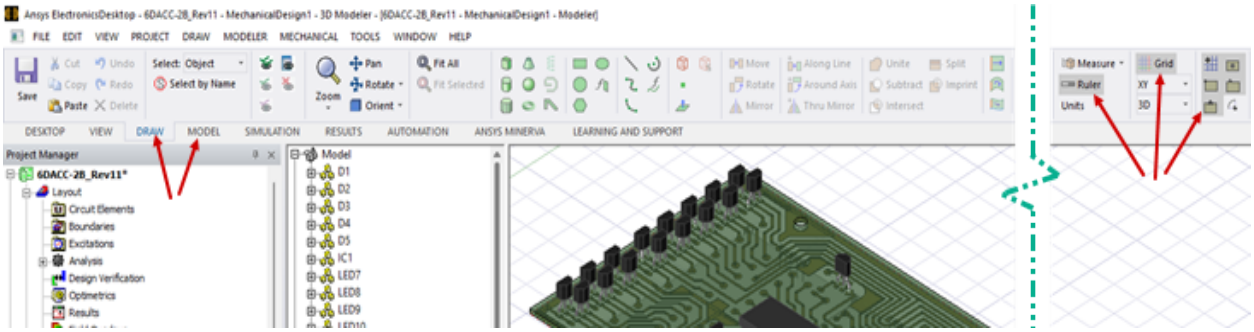


   - **Dark (beta):** A dark gray and black scheme with medium blue highlighting for selected ribbon icons and for inactive ribbon tabs. The active ribbon has a black background (including the tab):

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- **Classic:** A light gray and white scheme with medium gray highlighting for selected ribbon icons. The ribbon background is light gray. The name of the active ribbon tab has a blue font. A dark gray font is used for the inactive ribbon tab names:



4. Click **OK** to close the *Options* dialog box.

Working with Ribbons

The ribbon is a rectangular area across the top of the application containing features appropriate for the inserted project. It contains various tabs, each one representing a subset of program functionality. The tabs contain icons for related commands that are organized, grouped, and labeled. You can mouse over the icons to see tooltips.

Sizing the application affects the icons displayed for each selected tab, with priority given to the most used features.

Desktop tab

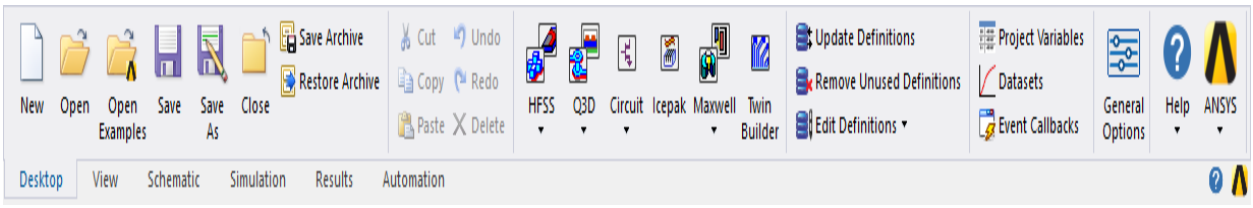
Twin Builder:

This shows the Desktop ribbons when Twin Builder is opened directly from the Start menu or from the Twin Builder shortcut on your desktop. This tab includes functions regarding projects, examples, archives, definitions, project variables, datasets, general options, and help.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ansys Electronics Desktop:

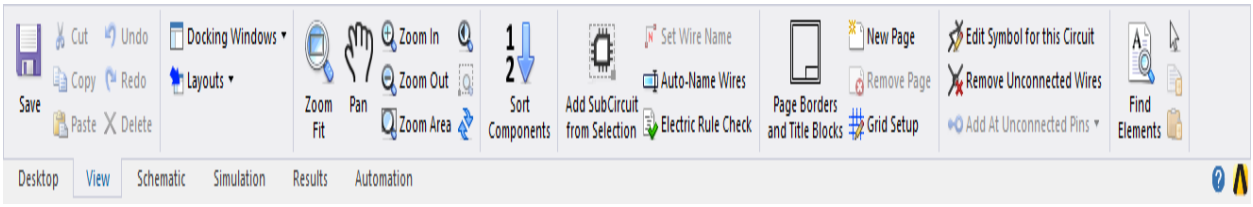
This shows the Desktop ribbons when a Twin Builder design has been inserted into the Ansys Electronics Desktop.



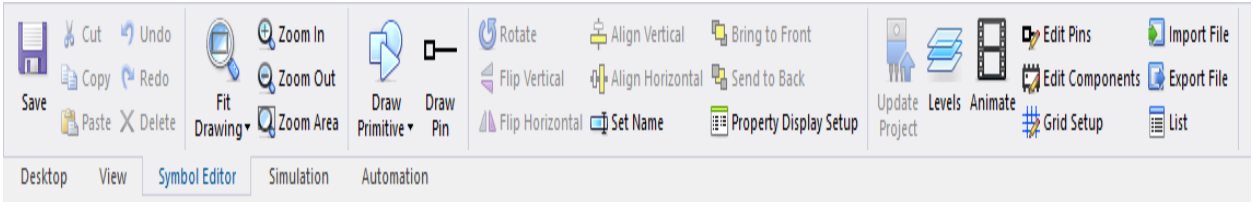
The following ribbons are displayed when Twin Builder is opened directly from the Start menu or from the Twin Builder shortcut on your desktop.

View tab

This tab contains options for the display, such as docking windows, layouts, page borders, title blocks, sort components, add subcircuit from a selection, grids, edit symbol for this circuit, and elements.



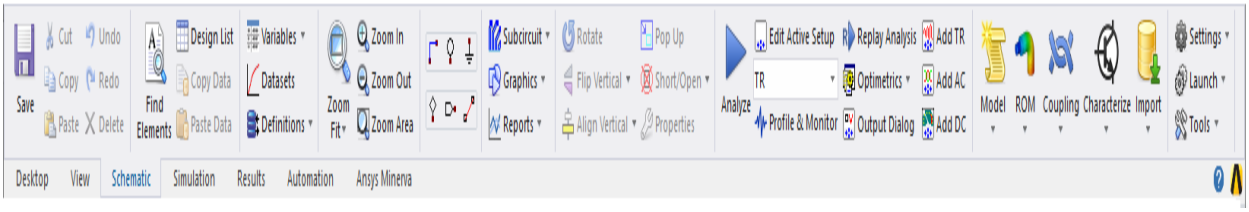
Click **Edit Symbol for this Circuit** to access the **Symbol Editor** tab.



Schematic tab

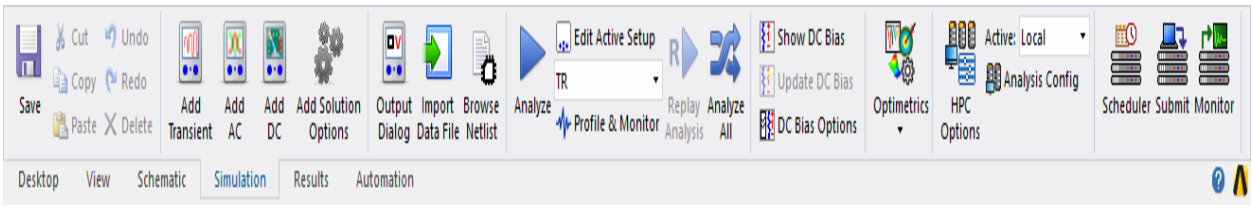
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

This tab includes options for zoom, adding designs, adding graphics and text, running reports, adding models, links for adding components, characterizing semiconductors and power modules, adding TR, AC, and DC analysis setups, and tools.



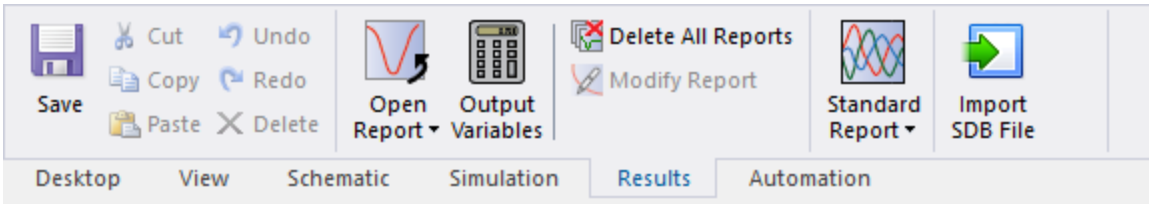
Simulation tab

This tab includes options for add transient, AC, DC and solution options, analyze and analyze all, optimetrics, HPC options, scheduler, and job submission.



Results tab

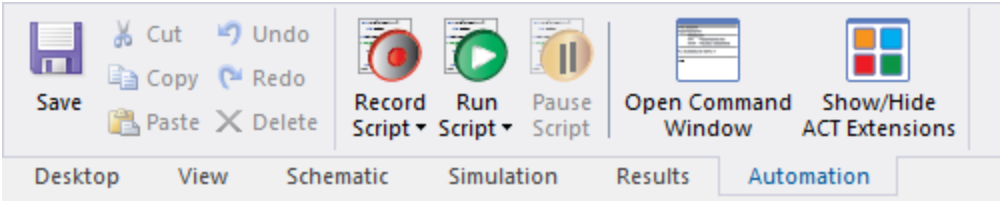
This tab includes options for reports, including open report, output variables, standard report, and import SDB file.



Automation tab

This tab includes controls for recording and running scripts, as well as access to the command window and display settings for ACT extensions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Running Twin Builder from a Command Line

Twin Builder includes line arguments you can include when launching from a command line or terminal prompt. All command-line arguments are case-insensitive. The commands associated with batch options can be used with a Job Management Interface for submitting jobs to Ansys or RSM and other supported schedulers. See:

RSM Integration with Job Management UI

Integration with Microsoft Windows® HPC Scheduler

Command-line syntax

ansysedt <*options***> <***run command***> <***project name/script name***>**

Run Commands

The following run commands are available in Twin Builder. Of the commands (**BatchSave**, **BatchSolve**, **BatchExtract**, **RunScript**, **RunScriptAndExit**), one or none must be used as arguments after **ansysedt**. Links to the valid options for each run command are listed and/or linked to descriptions.

-BatchSave <*project file name*>

Saves a named project to the current version. You can run this command with the **-Iconic** option, the -**Logfile** option, and the **-ng** option (no graphics).

-**BatchSolve** <*project file name*>

Solves all adaptive setups, sweeps, as well as Optimetrics setups found in the project file. If parallel solve is possible, you can use the **-Distribute** option in conjunction with -**BatchSolve**. You can run this command with the **-Iconic** option, the -**Logfile** option, the - **ng** option (no graphics), and the -**WaitForLicense** option.

Additional parameters for batch solves include the following. It is good practice to put quotes around the path to the executable, and the full path to the project. This ensures that spaces in the path or project will not be an issue. The same is true of the design name, if there are spaces. The quotes must enclose the entire argument including the Nominal or Optimetrics part.

[*designName*] - batch solve all setups for design with the name given under the project.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

[*designName*]:**Nominal** - batch solve all nominal setups for design with the name given under the project.

[*designName*]:**Optimetrics** - batch solve all Optimetrics setups for design with the name given under the project.

[*designName*]:**Nominal**:[*setupname*] - batch solve the specified nominal setup for design with the name given under the project. The [*setupname*] variable is case insensitive.

[*designName*]:**Optimetrics**:[*setupname*] - batch solve the specified Optimetrics setup for design with the name given under the project.The *setupname*variable is case insensitive.

-Local | -Remote | -Distributed

Perform the -Batchsolve on a local machine, a remote machine, or as a distributed solve using a specified machine list (see below). These command line options are mutually exclusive. That is, only one of these options should be specified. The settings persist only for the current session.

If you specify **-Local**, a machine list is not needed.

For **-Remote**, you should provide a machine list with a single hostname.

For **-Distributed**, you should provide a machine list or file path.

-Distributed takes optional arguments which modify the job distribution parameters. When the optional parameters are not present, the behavior is single level distributed solves with no change in order of precedence among possible distribution types. The optional parameters are:

includetypes= <default> | <distribution type 1, distribution type 2, ...>

If the distribution types are specified, only the listed distribution types are enabled. If default is specified, a default set of enabled distribution types will be used.

excludetypes= <default> | <distribution type 1, distribution type 2, ...>

If the distributed types are specified, all distribution types except those listed will be enabled. If default is specified, a default set of enabled distribution types will be used.

maxlevels= 1 | 2

This is the maximum level of job distribution. Single- and double-level distribution are supported.

numlevel1= number of level 1 tasks

When two level distribution is enabled, **numlevel1** specifies the number of level 1 tasks.

When using **-Distributed**:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- If neither **includetypes=** or **excludetypes=** are specified, default job distribution types will be used.
- If **maxlevels** is not specified, multilevel distribution will be disabled.

The **Job Distribution** tab in the **Analysis Configuration** dialog box displays the valid values for job distribution types.

-**MachineList file="<***machine list file path***>"**

In this format, the DSO machines are listed in a file. The machine names are listed in the text file, one hostname per line. The pathname of the file is file_path_name. The machines may be specified by IP address or by hostname, provided that the hostnames can be resolved on the local host. The number of distributed COM engines run on each host is equal to the number of times that the hostname appears in the list. That is, if host1 appears in the list once, and host2 appears in the list twice, then one COM engine will run on host1 and two COM engines will run on host2.

file=<*machine list file path***>** will also accept machine specifiers in the specified file using the format:

```
<machine name>:<total number of tasks>:<total number of cores>.
```

For example:

```
"Orion:4:8, Aries:3:12, Pluto:6:12"
```

With this form, duplicate machine names are not allowed, and the number of cores must be greater than the number of tasks.

You can use either form of the MachineList option to indicate the machine(s) on which to run a distributed batchsolve. The settings persist only for the current session.

When you use a file to define the machines available for a distributed solve you should list the machine addresses or names on separate lines:

```
192.168.1.1
192.168.1.2
(etc)
```

-**MachineList num = <***numb distributed engines***>**

This format is used when a scheduler (such as LSF, PBS, SGE or Windows HPC) manages the jobs sent to a cluster of hosts. In a scheduler environment, you can specify the number of distributed engines to use for distributed processing. In this case, do not specify the machine names after the flag because the names are provided by the scheduler. For example, in the Windows HPC environment, you can write the number of distributed engines as follows.

```
-machinelist num=4
```

The COM engines will be distributed across the hosts allocated to the job by the scheduler.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

-batchoptions "<*option*1>' '<*option*2>'..."

All options specified through **Tools** > **Options** dialog boxes go to the user-level registry. You can override such registry entries via the **-batchoptions** command line. These overrides apply only to the current Desktop session. This feature is available for all desktop products. The registry setting overrides may be specified on the command line, or may be in a file with the file path name specified on the command line. The **-batchoptions** command line option is only valid for batch jobs; it is ignored if neither **-BatchSolve** nor **-BatchSave** command line options are specified.

-**BatchExtract <***batchExtractScriptFile***> <***projectFile***>**

This command allows the following commands to be executed non-graphically via script and without checking out any GUI licenses: Update Reports, ExportToFile. A project file *must* be specified when the command line option **BatchExtract** is used. This means that commands in the <*batch extract script file*> will only be executed in the specified project. The "open/close" project commands are not supported in **BatchExtract** mode.

> **Note:**
>
> - **-ng** *must* be used with BatchExtract or it will fail with an error.
> - Including unsupported commands in the *batchExtractScriptFile* will terminate the script execution.

Examples:

- -**ng -BatchExtract <***batchExtractScriptFile***> <***projectFile***>**
- **-ng -BatchSolve ..... -BatchExtract <***batchExtractScriptFile***> <***projectFile***>**

The commands in *batchExtractScriptFile* will be executed after **BatchSolve** is done and before the project is saved.

Note that **BatchSolve** will continue to require solve licenses.

Example Script For Report Export:

```
ansysedt -ng -batchextract exportToFile.py "C:\Program
Files\AnsysEM\v242\Win64\Examples\Twin
Builder\Applications\Simple\Basic\mixer.aedt"
```

where exportToFile.py contains:

```
oDesktop.RestoreWindow()
oProject = oDesktop.SetActiveProject("mixer")
oDesign = oProject.SetActiveDesign("mixer")
oModule = oDesign.GetModule("ReportSetup")
oModule.UpdateReports(["XY Plot 1"])
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
oModule.ExportToFile("XY Plot 1", "exportToFilePy.csv")
```

-Monitor

You can monitor progress and messages on standard output, during non-graphical analysis. Progress, warning and info messages are logged to the standard output stream. Error and fatal messages are logged to the standard error stream. Schedulers intercept these streams and provide commands for display of this output - see individual scheduler documentation for specifics.

Examples:

```
C:\Program Files\AnsysEM\v242\Win64\ansysedt.exe -distributed
-machinelist list="192.168.1.1,192.168.1.2"
-batchsolve design_transient:Optimetrics "C:\distrib_project.adsn"


C:\Program Files\AnsysEM\v242\Win64\ansysedt.exe -batchsolve
TwinBuilderDesign1:Nominal "C:\Project1.aedt"
"C:\Program Files\AnsysEM\v242\Win64\ansysedt.exe"
-Iconic -Queue
-LogFile "H:\TwinBuilder\_TwinBuilderQueue\fence-v2__Array with
Fence4.log"
-BatchSolve "Array with Fence4:Nominal" "H:\TwinBuilder\fence-
v2.aedt"
```

-**RunScript** <*script file name*>

Run the specified script. Use the **-ScriptArgs** option to add one or more arguments to this command You can also use the **-Iconic** option.

-**RunScriptAndExit** <*script file name*>

Run the specified script and exit. Use the **-ScriptArgs** option to add one or more arguments to this command. You can also use the **-Iconic** option, the -**Logfile** option, and the -**WaitForLicense** option.

<none>

If you do not specify a run command with **hfss** on the command line, you can still specify the -**Help** and **-Iconic** option.

<project file>

Open the specified project on start up. Open a window listing the -**batchoptions** help. For instance:

E:\Program Files\AnsysEM\v242\Win64\ansysedt -batchoptionhelp

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Options

The following options can be associated with one or more of the run commands.

-batchoptionhelp

Open a window that displays the different command-line options. This is only used when none of the four run commands are used.

-Distribute

Distribute a batch solve to multiple machines. This option must be combined with the **-BatchSolve** run command and must be specified before it in the command line. See Distributed Analysis for more information on distributed analysis.

Example:

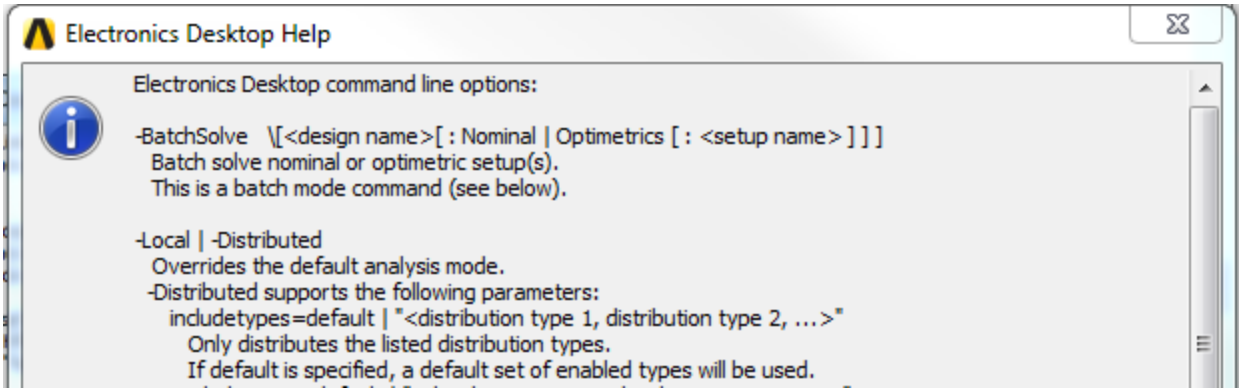
```
C:\TwinBuilder\ansysedt.exe -distribute -batchsolve
TwinBuilderDesign1:Optimetrics:ParametricSetup1 "C:\Project1.aedt"
```

-Help

Open a window that displays the different command-line options. This is only used when none of the four run commands are used.

For example:

```
E:\Program Files\AnsysEM\v242\Win64\ansysedt -help
```



-Iconic

Run Twin Builder with the window iconified (minimized). This can be used with all or none of the run commands.

-LogFile <*log file name*>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Specify a log file (use in conjunction with **-BatchSave** or **-BatchSolve** or **-RunScriptAndExit** run commands). If no log file is specified, <project_name>.log file will be written to the <project_name>.batchinfo directory with the current specification.

-ng

Run Twin Builder in non-graphical mode. (Use in conjunction with **-BatchSave** or **-BatchSolve**run commands. *Must* be used with **-BatchExtract** command.)

-WaitForLicense

Wait for licenses (use in conjunction with **-BatchSolve** or **-RunScriptAndExit**).

-ScriptArgs <*scriptArguments*>

Add arguments to the specified script in conjunction with **-RunScript** and **-RunScriptAndExit**.

ScriptArgs looks at the single argument after it and uses those as script arguments. You can pass multiple arguments to scriptargs by surrounding the script arguments in quotes. For example:

```
ansysedt.exe -scriptargs "TwinBuilderDesign1 Setup1" -
RunScriptAndExit C:\temp\test.py
```

In Python, the command line parameter following **-scriptargs** is passed without modification as a single string in the ScriptArgument python variable.

In VBscript, the command line parameter following **-scriptargs** is split into multiple strings and converted to a VBscript collection which is accessible via the **AnsoftScript.Arguments** collection. To access these arguments, for example:

```
msgbox AnsoftScript.Arguments(0) // Returns TwinBuilderDesign1
msgbox AnsoftScript.Arguments(1) // Returns Setup1
```

In either case, **Design1** is taken into Twin Builder as the first argument, and **Setup1** as the second argument. If you failed to use quotation marks, **Design1** would be taken as the first argument and **Setup1** would not be understood by Twin Builder.

Related Topics

For -batchoptions Use: Project Directory and Lib Paths

For -batchoptions Use: TempDirectory

For -batchoptions Use: Various Desktop Settings

Batch Options Command Line Examples

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Examples and Further Explanations of -batchoptions Use

This section provides examples and further explanations of **-batchoptions**.

- Example with registry settings specified on the command line
- Example with registry settings specified in a file
- When to use the -batchoptions Desktop command line option

The following examples use Ansys Electronics Desktop- and Twin Builder-specific settings. This feature is available for all Ansys Electronics Desktop products.

- The registry path separator is the slash (/) character.
- Each complete registry key (that is, a registry path and option name) is enclosed in single quotes.
- Registry string values are enclosed in single quotes.
- After the -batchoptions switch, the set of registry keys and values that follows it must be enclosed in double quotes. However, if a batchoptions file is referenced (instead of listing the options directly on the command line), the double quotes are not used around the file name.
- Backslashes in registry key string values must be escaped with another backslash (\\ ), since a backslash by itself is an escape code within strings.

Example with registry settings specified on the command line

```
ansysedt.exe -batchsolve -batchoptions
"'Desktop/Settings/ProjectOptions/NumberOfProcessors'=4
'Desktop/ProjectDirectory'='C:\\projects\\test'" projectname.aedt
```

This command line overrides registry values of the NumberOfProcessors (Desktop/Settings/ProjectOptions key) and ProjectDirectory (Desktop key) options.

> **Note:**
>
> - Multiple registry settings may appear in a single **-batchoptions** value, separated by whitespace.
> - The **-batchoptions** value must be enclosed in double quotes if it contains any whitespace

Example with registry settings specified in a file

```
ansysedt.exe -batchsolve -batchoptions <file name>
projectname.aedt
```

where the referenced file, <*file name*>, contains:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
$begin 'Config'
  'Desktop/Settings/ProjectOptions/NumberOfProcessors'=4
  'Desktop/Settings/ProjectOptions/Twin
  Builder/UpdateReportsOnSolve'='Never'
  'Desktop/ProjectDirectory'='C:/projects/test'
$end 'Config'
```

This command overrides the registry values of the NumberOfProcessors (Desktop/Settings/ProjectOptions key), UpdateReportsOnSolve (Desktop/Settings/ProjectOptions/Twin Builder key), and ProjectDirectory (Desktop key) options. These overrides apply only to the current Electronics Desktop session.

> **Note:**
>
> - The -batchoptions ***<file name>*** value must be enclosed in double quotes if it contains whitespace
> - The **$begin 'Config'** and **$end 'Config'** lines are required

For additional options you can override from the command line with **-batchoptions**, see:

- For -batchoptions Use: Project Directory and Lib Paths
- For -batchoptions Use: TempDirectory
- For -batchoptions Use: Various Desktop Settings

When to Use the -batchoptions Desktop Command Line Option

You can set analysis parameters for batch mode jobs using the graphical user interface (GUI). For example, you can set all options for a batch job using the **Add Batch Option** dialog box, which is accessed through the **Submit Job To** dialog box (click **Submit** on the **Simulation** ribbon tab). These values override those saved in the options registry but do not alter the registry.

For graphical analyses that do **not** use batch mode, you specify the analysis parameters using the GUI (**Tools** > **Options** > **General Options** or **HPC and Analysis Options**). These settings are written to the registry when you exit Twin Builder. The settings are read from the registry when the application is started. Therefore, when you start Twin Builder, all settings retain the values from the previous session of the same user on the same machine. If there was not a previous session of the same user on the same machine, then the values are obtained from other registry configuration files or from a default value.

When running a batch analysis, any setting that is not specified using the **-batchoptions** command line option is taken from the registry. This value is typically the setting from the last session of the same user on the same machine. However, you can use the **-batchoptions** command line option to override the parameter with values specified on the command line or in a

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

batchoptions file. The values specified using the **-batchoptions** command line option only apply to the batch job, and do not affect the parameter values stored in the registry.

If important **-batchoptions** values are not specified when running a batch job, the paramaters could be affected by an interactive session running on the same host by the same user. Parameter changes can occur if you set an option in the GUI and exit the program, or if another process that accesses the registry exits. To be sure of the desired batch job outcome, avoid changing options in concurrent interactive sessions or include the desired -batchoptions in the command line.

For -batchoptions Use: Project Directory and Lib Paths

The **PersonalLib**, **syslib** and **userlib** settings are different from other settings. If the final directory name is different from what is expected, then **PersonalLib**, **syslib**, or **userlib** is appended as a final directory. In addition, these settings may come from a different registry value if the registry values shown above are not set.

Registry Key	Default Value	Units or Values	Description
Desktop/ProjectDirectory	Ansoft subdirectory of user's **HOME** directory or **Documents** directory.	Directory path	Directory where new projects are created
Desktop/PersonalLib	**PersonalLib** subdirectory of user's **HOME** directory or **Documents** directory.	Directory path	Directory **PersonalLib** is appended if final directory is not **PersonalLib**
Desktop/syslib	**syslib** subdirectory of installation directory.	Directory path	Directory **syslib** is appended if final directory is not **syslib**
Desktop/userlib	**userlib** subdirectory of installation directory.	Directory path	Directory **userlib** is appended if final directory is not **userlib**

Related Topics

For -batchoptions Use: TempDirectory

For -batchoptions Use: Various Desktop Settings

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Running Twin Builder from a Command Line

For -batchoptions Use: TempDirectory

Registry Key	Default Value	Units or Values	Description
TempDirectory	Set by installer	-	Directory for temporary files.

Related Topics

For -batchoptions Use: Project Directory and Lib Paths

For -batchoptions Use: Various Desktop Settings

Running Twin Builder from a Command Line

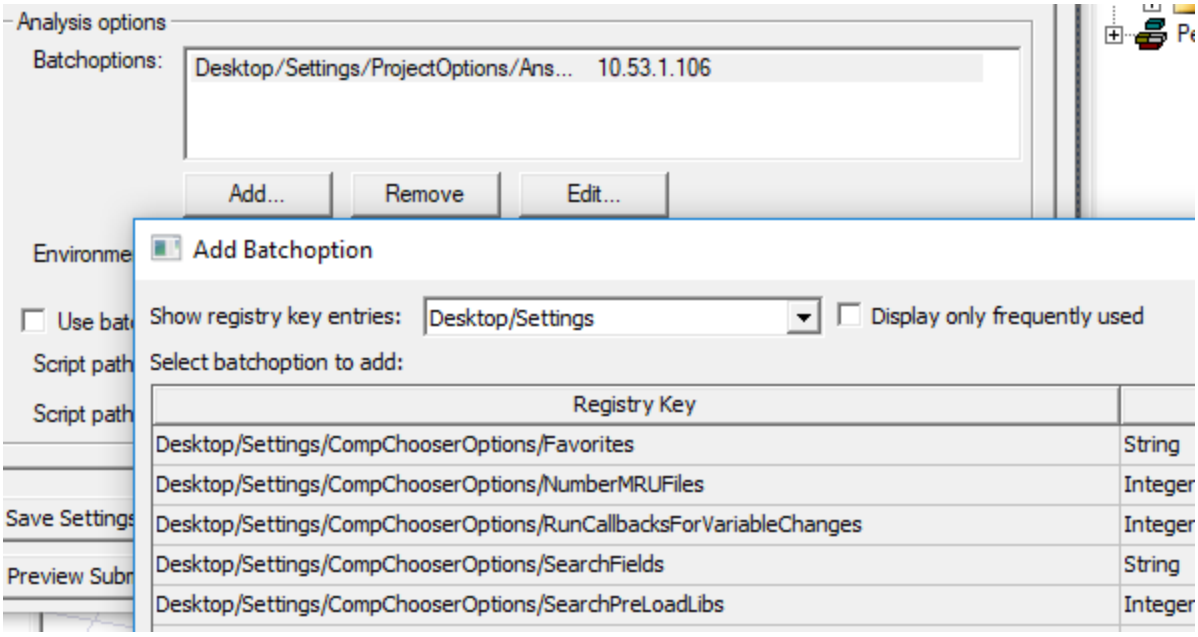
For -batchoptions Use: Various Desktop Settings

Note that most of these options only affect the GUI. To view these options from the *Submit Job To* window:

1. In the **Analysis options** area of the **Submit Job To** window, click **Add...**.
2. In the *Add Batchoption* window that appears, select **Desktop/Settings** from the **Show**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

registry key entries drop-down menu.



Registry Key	Default Value	Units or Values	Description
Desktop/Settings/ProjectOptions/AnimationMemory	200	Megabytes (MB)	Stop animations when available memory falls below this value.
Desktop/Settings/ProjectOptions/AnsoftCOMPreferredIPAddress (see note below table)	"" (empty string)	IP address (as a string)	IP address used to connect from COM engines to ansysedt.exe.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Registry Key	Default Value	Units or Values	Description
Desktop/Settings/ProjectOptions/AnsoftCOMPreferredSubnetAddress (see note below table)	"" (empty string)	Subnet address	Subnet used to connect COM engines to ansysedt.exe – allowed formats are: • IPv4 network prefix in CIDR notation Example: 123.123.123.0/24 • IPv4 network prefix with /subnet mask appended Example: 123.123.123.0/255.255.255.0
Desktop/Settings/ProjectOptions/AutoSaveInterval	10	edits	Number of edits to allow between autosaves.
Desktop/Settings/ProjectOptions/AutoShowMessageWindow	1 (true)	0 (false) or 1 (true)	Show message window on new messages.
Desktop/Settings/ProjectOptions/AutoShowProgressWindow	0 (false)	0 (false) or 1 (true)	Show progress window when starting a simulation.
Desktop/Settings/ProjectOptions/DiskLimitForAbort	0	Megabytes (MB)	A warning displays when available disk space falls below this value.
Desktop/Settings/ProjectOptions/DoAutoSave	1 (true)	0 (false) or 1 (true)	Enable or disable autosaves.
Desktop/Settings/ProjectOptions/DrawStateIconsInProjectTree3	1 (true)	0 (false) or 1 (true)	Change icon when selection does not match active window.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Registry Key	Default Value	Units or Values	Description
Desktop/Settings/ProjectOptions/ExpandMessageTreeOnInsert	1 (true)	0 (false) or 1 (true)	Ensure that new messages are visible in the message window tree.
Desktop/Settings/ProjectOptions/ExpandOnInsert	0 (false)	0 (false) or 1 (true)	Expand Project tree on insert.
Desktop/Settings/ProjectOptions/HighlightActive ContextInProjectTree2	1 (true)	0 (false) or 1 (true)	Emphasize active command context (menu and toolbars).
Desktop/Settings/ProjectOptions/SavePreviewImagesInProject File	1 (true)	0 (false) or 1 (true)	Save preview images in project file.
Desktop/Settings/ProjectOptions/UpdateReportOnFileOpen	0 (false)	0 (false) or 1 (true)	Update reports on file open.

Note:

The preferred IP address and preferred subnet address settings are mutually exclusive. If both are specified to be non-empty strings, then the preferred IP address takes precedence, and the preferred subnet address is ignored. This feature is typically used for cluster environments using batch solves. The setting can be made via **batchoptions** but can also be done via UpdateRegistry.

Related Topics

For -batchoptions Use: Project Directory and Lib Paths

For -batchoptions Use: TempDirectory

Running Twin Builder from a Command Line

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Obtaining Information about the Software and Release

To obtain information about the software and release:

1. Click **Help** > **About Ansys Electronics Desktop**. The **About Ansys® Electromagnetics Suite** [*release number*] dialog box appears, listing information about the product.
2. Click the **Installed Components** tab to view a list of software installed.
3. Click the **Client License Settings** tab to view information about:
   - Provider Name
   - ANSYSLI Version
   - FlexNet Publisher Servers
   - Admin Directory
   - Customer Number
   - FLEXlm Version
   - Redirect Info
4. Click **Export** to save the software information as a text file.
5. Click **OK** to close the **About Ansys® Electromagnetics Suite** [*release number*] dialog box.

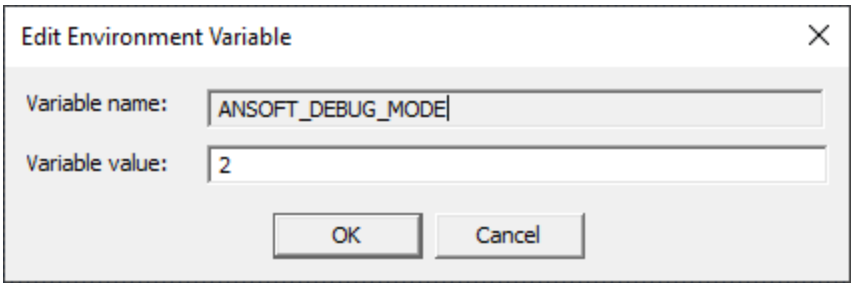
Debug Logging

Electronics Desktop provides error logging capabilities.

To begin logging errors:

1. Click **Tools** > **Debug Logging**.

   The **Edit Environment Variable** window appears, showing the ANSOFT_DEBUG_ MODE variable.



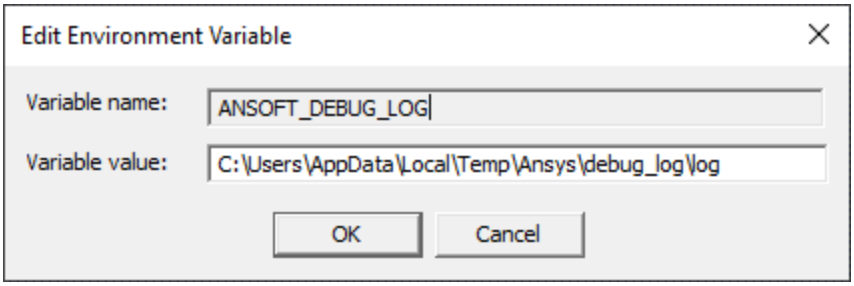
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Enter a **Variable value**.

   The default value is 2, and should not be changed unless directed by technical support.

3. Click **OK**.

   The **Edit Environment Variable** window updates to display the ANSOFT_DEBUG_LOG variable.



4. Enter a folder path to where log files (*.**log**) are stored.
5. Click **OK**.

   Errors are logged to the specified folder until you disable logging.

To stop logging errors:

- Select **Tools** > **Stop Debug Logging**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3 - Help Menu

The **Help** menu displays different selections depending upon the type of design inserted to the active project (such as Twin Builder, HFSS). The following describes the basic selections displayed in the **Help** menu.

> **Note:**
>
> Ansys Electronics Desktop Student includes access to PDF documentation only.

- **Twin Builder Help** opens to the product help within the Electronics help system. You can also access PDF versions from within the help system.
- **Twin Builder Scripting Help** opens to the product's scripting help.
- **Twin Builder Getting Started Guides** opens to a list of links to the product's Getting Started Guides. These Getting Started Guides walk you through projects that demonstrate features of the product solvers.
- **Twin Builder PDFs** provides access to PDFs for Twin Builder, including the main help, scripting guide, and Getting Started Guides.
- **Twin Builder Components** is enabled when you insert a design to the active project, and opens the Twin Builder Components help to a section that pertains to the component type.
- **Ansys Customer Support** opens a browser page to the Ansys Customer Portal. At the website you can learn more about Ansys products and services and log on to contact Ansys technical support staff.
- **What's New in this Release** opens a PDF that describes *What's New in Ansys Electronics Desktop* for the release.
- **Ansys Product Improvement Program** opens a dialog box that describes the Product Improvement Program option.

  > **Note:**
  >
  > You can disable the Ansys Product Improvement Program so you are not prompted to enable the Program when you first start Electronics Desktop. To do this, after installing the software, run this command as a user with permissions to modify the installed fileset:
  >
  > **UpdateRegistry.exe -set -ProductName ElectronicsDesktop2024.2 -RegistryKey Desktop/Settings/ProjectOptions/ProductImprovementOptStatus -RegistryLevel install -RegistryValue 1**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **About Ansys Electronics Desktop** opens a dialog box that displays the Ansys® Electromagnetics Suite release number and contains tabs that show information about the **Installed Components** and **Client License Settings**.

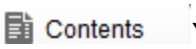
F1 Context-Sensitive Help

To access **F1** help from the Ansys Electronics Desktop user interface, do one of the following:

- To open a help topic about a menu option, hover the cursor over the item and press **F1**.
- To open a help topic about a dialog box, open the dialog box and press **F1**.

Finding Information in the Help

The help system provides different ways to find information and navigate quickly:

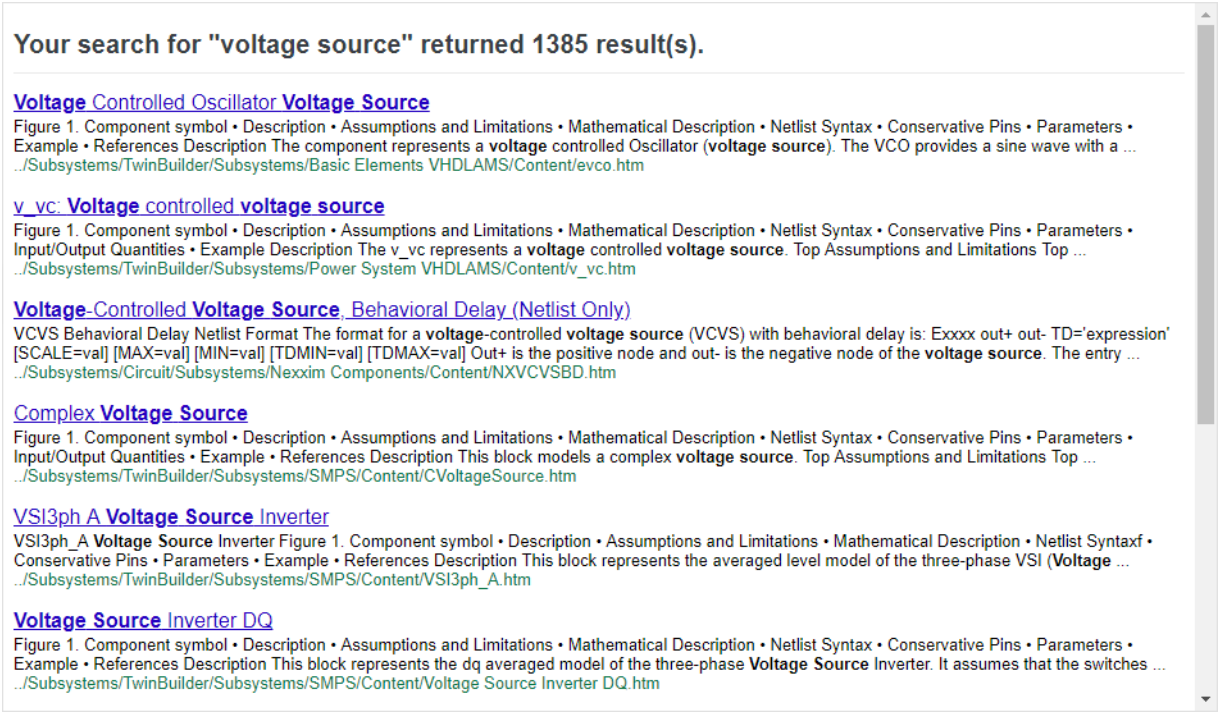
- Press **F1** on any open dialog box to open its help.
- *A hierarchical table of contents* - 

You can browse through the table of contents, expand entries, and close entries. Click on an entry to see it in the content area.
- *A full text search* - To locate every occurrence of a word or phrase that may be contained in the help, use the search function.

Using the Search Function in the Help

When you enter words or strings to search for in the help, the search engine, by default, lists all topics in which any of the words occur. For example, if you enter “voltage source” without the quotation marks, the results show all topics that contain “voltage” or “source.”

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

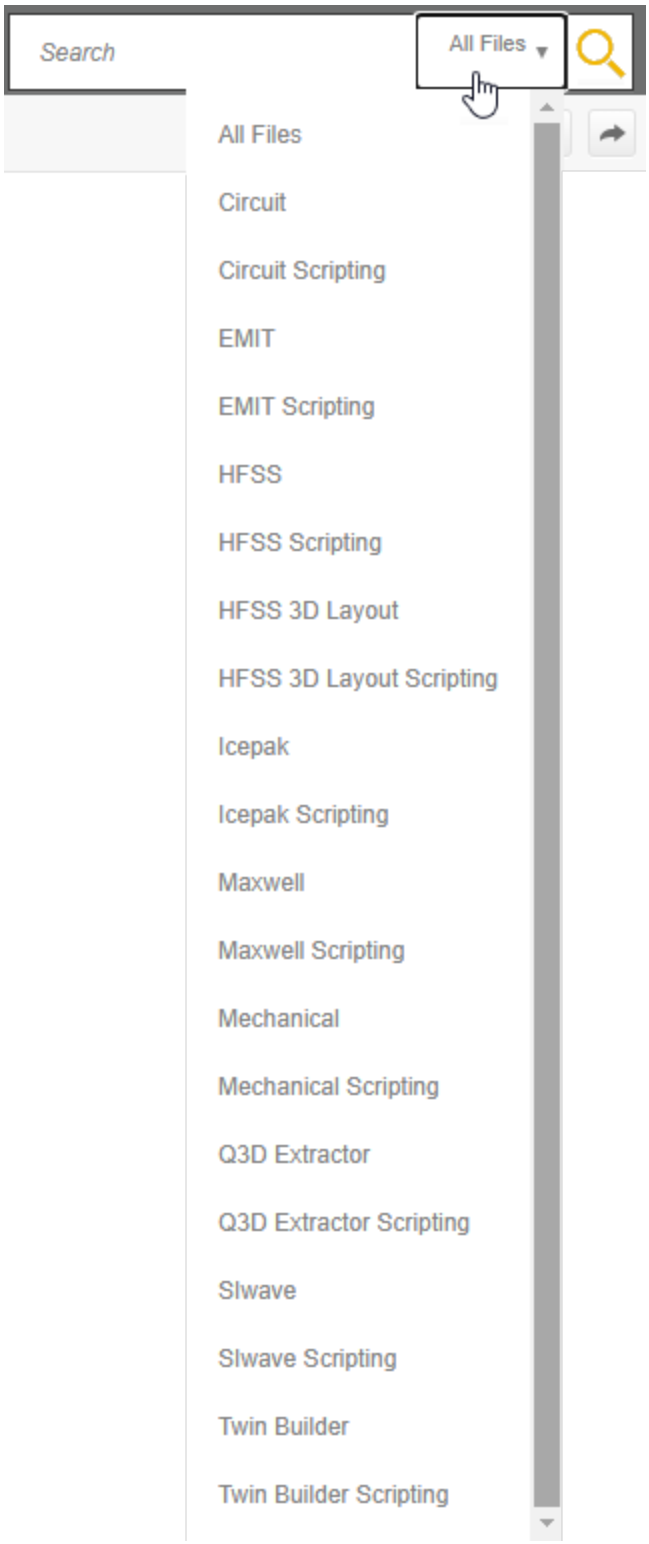


This method probably provides more hits than you want. The Search function in the help provides several methods for making searches more specific.

Performing a Basic Search

1. Type the words or string in the search box.
   - If you are searching within the full Electronics help system, you see a search box that includes a drop-down for specifying a product, specifying a product's scripting guide, or searching across all products. When you change the filter, the results dynamically reflect the selected filter.

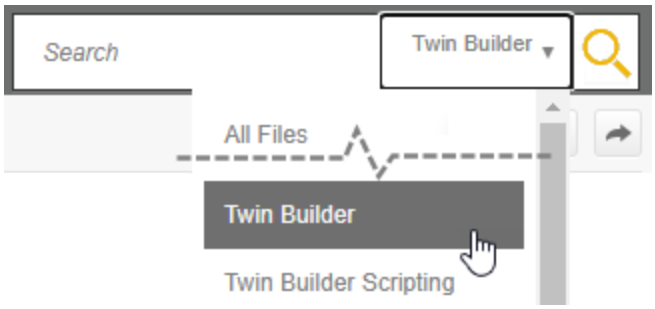
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- If you used F1 from Twin Builder, you see a search box that permits searches for all files in Twin Builder (for example, *All Files*), all files except the Twin Builder scripting guide

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

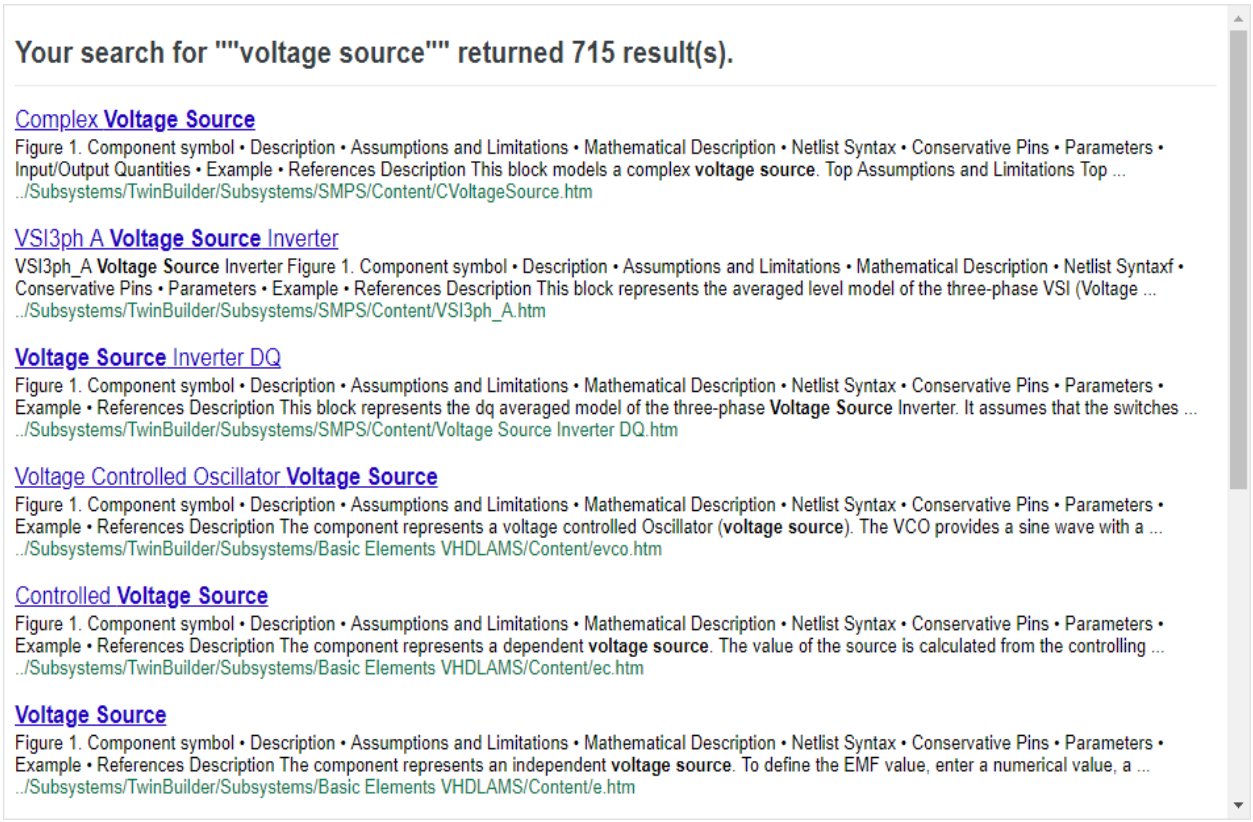
(for example, *Twin Builder*), or just files in the *Twin Builder* scripting guide (for example, *Twin Builder Scripting*).



2. Click the topic you want in the list results.
   - To view a different topic, press your browser's back button to return to the list results.
   - To turn off highlighting on the page you are viewing, click the Remove Highlights icon .

Searching with Quotation Marks

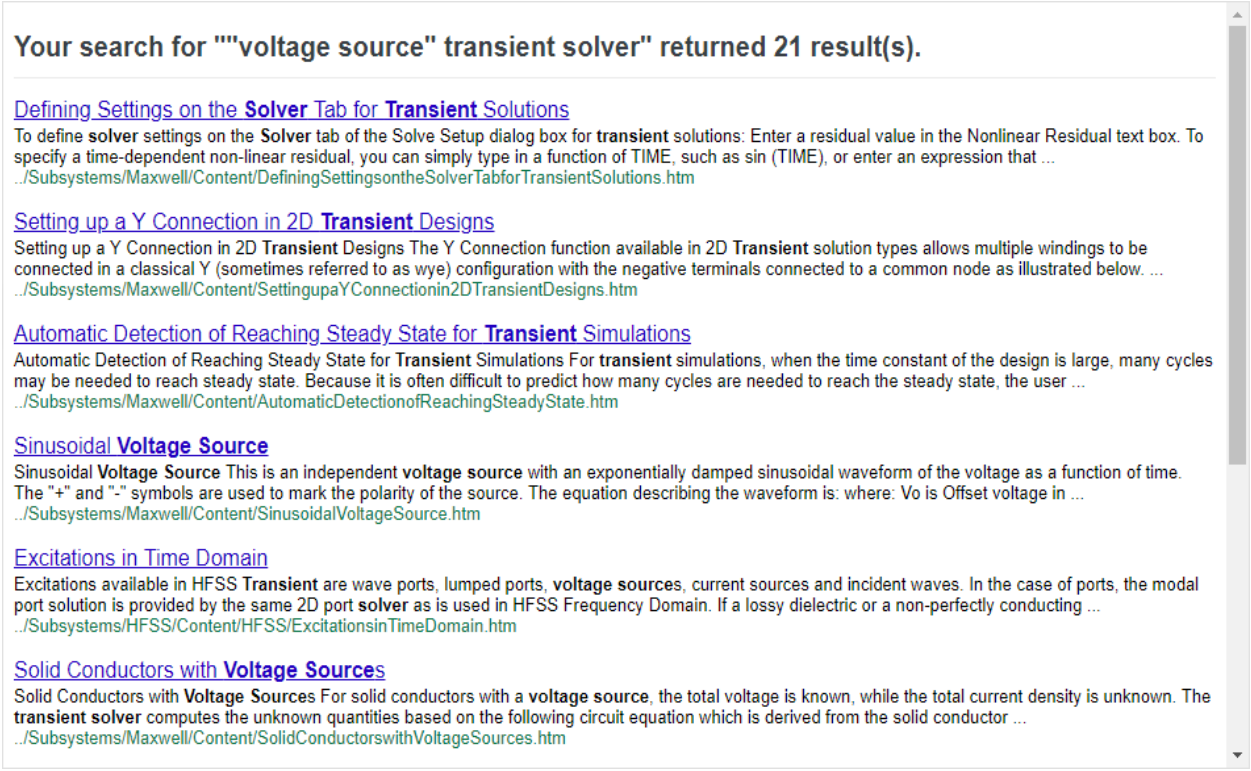
If you enter “voltage source” with the quotation marks, the results show all topics that include the phrase.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If you want to limit the results more, you can enter additional words, such as:

“voltage source” transient solver



Note:

- Searches are not case sensitive, so you can type your search in uppercase or lowercase characters.
- You may search for any combination of letters (a-z) and numbers (0-9).
- Punctuation marks (period, colon, semicolon, comma, hyphen) are ignored during a search.
- When searching for a file name with an extension, group the entire string in quotation marks (for example, "file name.ext").

Using Boolean Operators

You can also use Boolean operators to affect the number of topics listed.

Operator (s)	Usage	Example(s)
AND	Lists all topics that contain all of the terms.	Net AND Selection

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Operator (s)	Usage	Example(s)
+ &		Net + Selection Net & Selection
OR |	Lists all topics that contain any of the terms.	Net OR Selection Net | Selection
NEAR	Lists all topics that contain the terms near the other terms.	Net NEAR Selection
NOT ! ^	Lists all topics that contain the first term but not the second.	Net NOT Selection Net ! Selection Net ^ Selection

Use parentheses to group terms and operators. For example:

- solver AND (Circuit) NOT HFSS NEAR dynamic
- solver AND (HFSS OR Circuit) NOT (Q3D OR 2d) NEAR dynamic
- “dynamic link” ! (HFSS | circuit)

> **Important:**
>
> Because the characters +, &, |, !, and ^ are used as operators, you cannot search for them in the help. Doing so will result in an error.

Getting Help from Ansys Technical Support

For information about Ansys Technical Support, go to the Ansys corporate support Web site (http://www.ansys.com/Support), or contact your Ansys account manager.

Email can work well for technical support. All Twin Builder software files are ASCII text and can be sent conveniently by email. When reporting difficulties, it is extremely helpful to include very specific information about what steps were taken or what stages the simulation reached. This allows more rapid and effective debugging.

Provide the following information when you contact us:

- Your name and affiliation.
- Serial number, version number, and the name of Ansys Electromagnetics software.
- The type of hardware and operating system you are using.
- A description of what happened and what you were doing when the problem occurred.
- The exact wording of any messages that appeared on the screen.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Help Menu

The Ansys Product Improvement Program

This product is covered by the Ansys Product Improvement Program, which lets ANSYS, Inc., collect and analyze anonymous usage data reported by our software without affecting your work or product performance. Analyzing product usage data helps us to understand customer usage trends and patterns, interests, and quality or performance issues. The data enable us to develop or enhance product features that better address your needs.

How to Participate

The program is voluntary. To participate, select **Yes** when the Ansys Product Improvement Program dialog box appears. Only then will collection of data for this product begin.

How the Program Works

After you agree to participate, the product collects anonymous usage data during each session. When you end the session, the collected data is sent to a secure server accessible only to authorized Ansys employees. After Ansys receives the data, various statistical measures such as distributions, counts, means, medians, modes, etc., are used to understand and analyze the data.

Data We Collect

For all products that offer the Ansys Product Improvement Program, we only collect anonymous data such as session statistics, hardware information, types of loading, solution types, solution statistics, and similar data. The specific data collected varies from product to product.

For Ansys Electronics, we collect the following information:

- Application
  - Build information
  - System information
    - Country
    - Country code
    - CPU architecture
    - CPU brand
    - CPU identifier
    - Graphics card
    - Operating system

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

    - Operating system version
    - Processor count
    - Time zone
    - Total RAM value
- Session
    - Workbench session
    - Total CPU time
    - Execution mode
    - Start method
    - Number of processes
    - Number of compute nodes (HPC)
    - Session begin
    - Session end
- Mesh
    - Number of nodes
    - Number of elements
    - Number of zones
    - Number of faces

Data We Do Not Collect

The Product Improvement Program does not collect any information that can identify you personally, your company, or your intellectual property. This includes but is not limited to names, addresses, file names, part names, geometry- or design-specific inputs, material property values, etc. We make no record of where we collect data from.

Opting Out of the Program

You may stop your participation in the program any time you wish. To do so, select **Help** > **Ansys Product Improvement Program**. A dialog box appears and asks if you want to continue participating in the program. Select **No**, then click **OK**. Data will no longer be collected or sent.

The ANSYS, Inc., Privacy Policy

All Ansys products are covered by the ANSYS, Inc., Privacy Policy, which you can read here.

Frequently Asked Questions

1. *Am I required to participate in this program?*

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

No, your participation is voluntary. We encourage you to participate, however, as it helps us create products that will better meet your future needs.

2. *Am I automatically enrolled in this program?*

No. You are not enrolled unless you explicitly agree to participate.

3. *Does participating in this program put my intellectual property at risk of being collected or discovered by Ansys?*

No. We do not collect any project-, company-, or model-specific information.

4. *Can I stop participating even after I agree to participate?*

Yes, you can stop participating at any time. To do so, select **Help** > **Ansys Product Improvement Program**. A dialog box appears and asks if you want to continue participating in the program. Select **No**, then click **OK**. Data will no longer be collected or sent.

5. *Will participation in the program slow the performance of the product?*

No, the data collection does not affect the product performance in any significant way. The amount of data collected is very small.

6. *How frequently is data collected and sent to Ansys servers?*

The data is collected during each use session of the product. The collected data is sent to a secure server once per session, when you exit the product.

7. *Is this program available in all Ansys products?*

Not at this time, although we are adding it to more of our products at each release. The program is available in a product only if this Ansys Product Improvement Program description appears in the product documentation, as it does here for this product.

8. *If I enroll in the program for this product, am I automatically enrolled in the program for the other Ansys products I use on the same machine?*

Yes. Your enrollment choice applies to all Ansys products you use on the same machine. Similarly, if you end your enrollment in the program for one product, you end your enrollment for all Ansys products on that machine.

9. *How is enrollment in the Product Improvement Program determined if I use Ansys products in a cluster?*

In a cluster configuration, the Product Improvement Program enrollment is determined by the host machine setting.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Conventions Used in the Help

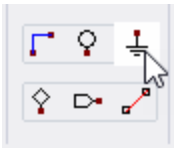
Please take a moment to review how instructions and other useful information are presented in this documentation.

- Procedures are presented as numbered lists. A single bullet indicates that the procedure has only one step.
- Bold type is used for the following:
  - Keyboard entries that should be typed in their entirety exactly as shown. For example, “**copy file1**” means you must type the word **copy**, type a space, then type **file1**.
  - On-screen prompts and messages, names of options and text boxes, and menu commands. Menu commands are often separated by greater than signs (>). For example, “select**HFSS** > **Excitations** > **Assign** > **Wave Port**.”
  - Labeled keys on the computer keyboard. For example, “Press **Enter**” means to press the key labeled **Enter**.
- Italic type is used for the following:
  - Emphasis.
  - The titles of publications.
  - Keyboard entries when a name or a variable must be typed in place of the words in italics. For example, “**copy** *file name*” means you must type the word **copy**, then type a space, then type the name of the file.
- The plus sign (+) is used between keyboard keys to indicate that you should press the keys at the same time. For example, “Press Shift+F1” means to press the **Shift** key and, while holding it down, press the **F1** key also. You should always depress the modifier key or keys first (for example, Shift, Ctrl, Alt, or Ctrl+Shift), continue to hold it/them down, then press the last key in the instruction.

Accessing Commands: *Ribbons, menu bars,* and *shortcut menus* are three methods that can be used to see what commands are available in the application.

- The *Ribbon* occupies the rectangular area at the top of the application window and contains multiple tabs. Each tab has relevant commands that are organized, grouped, and labeled. An example of a typical user interaction is as follows:

  "Click **Schematic** > **Ground**"



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

This instruction means that you should click the **Ground** command on the **Schematic** ribbon tab. An image of the command icon, or a partial view of the ribbon, is often included with the instruction.

- The *menu bar* (located above the ribbon) is a group of the main commands of an application arranged by category such File, Edit, View, Project, etc. An example of a typical user interaction is as follows:

  "On the **File** menu, click the **Open Examples** command" means you can click the **File** menu and click **Open Examples** to launch the dialog box.

- Another alternative is to use the *shortcut menu* that appears when you right-click an object. An example of a typical user interaction is as follows:

  “Right-click and select **Assign Excitation** > **Wave Port**” means when you right-click an object, you can execute the excitation commands from the shortcut menu (and the corresponding sub-menus).

Getting Help: Ansys Technical Support

For information about Ansys Technical Support, go to the Ansys corporate Support website, http://www.ansys.com/Support. You can also contact your Ansys account manager in order to obtain this information.

All Ansys software files are ASCII text and can be sent conveniently by e-mail. When reporting difficulties, it is extremely helpful to include very specific information about what steps were taken or what stages the simulation reached, including software files as applicable. This allows more rapid and effective debugging.

Help Menu

To access help from the Help menu, click **Help** and select from the menu:

- **[product name] Help** - opens the contents of the help. This help includes the help for the product and its *Getting Started Guides*.
- **[product name] Scripting Help** - opens the contents of the *Scripting Guide*.
- **[product name] Getting Started Guides** - opens a topic that contains links to Getting Started Guides in the help system.

Context-Sensitive Help

To access help from the user interface, press **F1**. The help specific to the active product (design type) opens.

You can press **F1** while the cursor is pointing at a menu command or while a particular dialog box or dialog box tab is open. In this case, the help page associated with the command or open dialog box is displayed automatically.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

What's New in this Release

Click **Help** > **What's New in this Release** to open a PDF file describing the important features.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4 - Working with Twin Builder Projects

A Twin Builder project is a folder that includes one or more Twin Builder *designs* and their associated settings, including report definitions, in a file with an **.aedt** extension. To the fullest extent possible, Twin Builder projects are portable in that they include, rather than merely reference, the library elements such as graphical symbols, components, scripts, and models they contain. Each design ultimately includes report and post-processing information.

A new project called **Project***n* opens when the software starts. A design named **Twin Builder***n* opens with the new project. You can also select **File** > **New** to open a new project.

> **Note:**
>
> Use the **File** menu commands to manage projects. If you move or change the names of files without using these commands, the software may not be able to find information necessary to solve the model.

Related Topics

Setting Options

Model Editor Settings

Twin Builder Files

Setting up a Twin Builder Design

Creating Projects

Opening Existing Projects

Translating Legacy Simplorer Projects and Schematics

Closing Projects

Saving Projects

Archiving Projects

Restoring Archives

Renaming Projects

Deleting Projects

Removing Unused Definitions

Datasets

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Undoing Commands

Redoing Commands

Inserting a Documentation File

Importing Simulation Models

Printing

Saving Project Notes

Event Callbacks

Analyzing Designs

Setting Options for Twin Builder

Click **Tools** > **Options** > **General Options** to set options for Twin Builder as well as the Ansys Electronics Desktop. The following linked options are pertinent to Twin Builder:

- General options, such as project options, units settings, and remote analysis options.
- 2D Extractor
- Circuit Design
- Circuit Netlist Design
- HFSS
- HFSS 3D Layout
- HFSS-IE
- Icepak
- Maxwell 2D
- Maxwell 3D
- Q3D
- RMxprt
- Twin Builder, such as options for Spice, VHDL, and Modelica compilers.
- 3D Modeler options, such as cloning options, display colors and render settings, snap modes and mouse sensitivity.
- Layout Editor
- Machines
- Model Editor, such as editor settings for C-Model, SML, Spice, VHDL, VHDL Package, and Modelica.
- Netlist & Script Editor, such as bookmark color and fonts.
- Schematic Editor, such as symbol graphics, connectivity, and object placement.
- Optimetrics

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Reporter options, including Report Setup, including advanced mode editing, the number of significant digits to display, and drag and drop behavior, and Report2D options, such as fonts, labels, line styles, and colors.

Click **Tools** > **HPC and Analysis Options**, to specify design-specific options.

Additionally, **Tools** > **Options** > **Export Options Files** writes XML files containing options settings at all levels to the specified directory. This feature makes it easier for different users to use Ansys Electromagnetics tools installed on shared directories or network drives. The Example Uses for Export Options Features section outlines some use cases enabled by this feature.

Related Topics

Setting Options via Configuration Files

Example Uses for Export Options Features

User Options and the Update Registry Tool

Batchoptions Command Line Examples

Setting General Options

To set general options in Twin Builder:

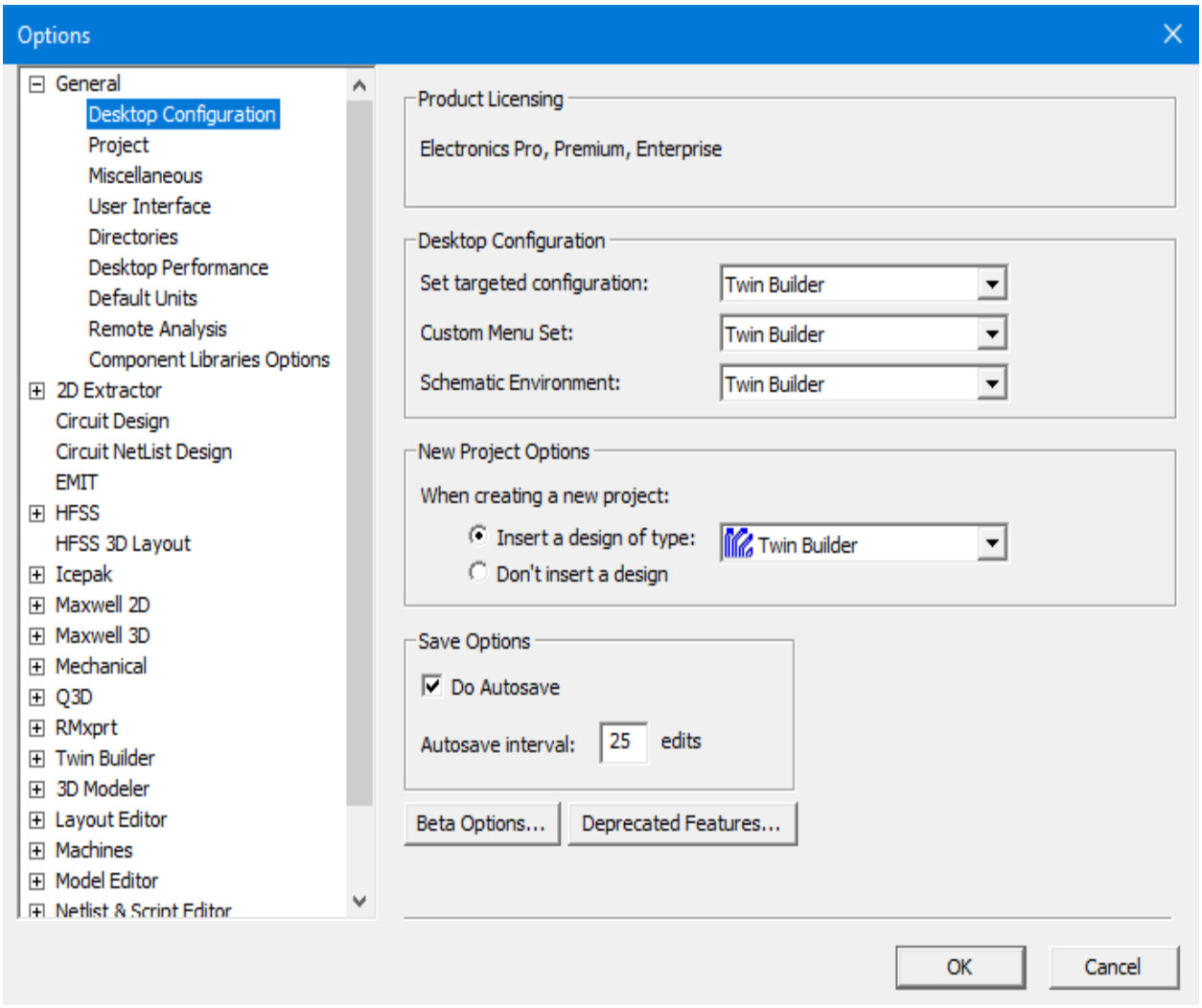
1. Click **Tools** > **Options** > **General Options**. The **Options** dialog box appears.
2. Click **General**. The following options are available:
   - Desktop Configuration
   - Project
   - Miscellaneous
   - User Interface
   - Directories
   - Desktop Performance
   - Default Units
   - Remote Analysis
   - Component Libraries Options
2. Click the desired options, and edit the settings as needed.
3. Click **OK**.

General Options: Desktop Configuration

These options are set in the **Desktop Configuration** panel under **General** in the **Options** dialog box. Options affect the solver icons displayed in the **Project** menu for inserting designs. Some

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

selections also affect the default appearance of the **Import**, **Automation**, and **Definition** toolbar menus. You can select the default project type, and whether a project is inserted. This lets you customize aspects of the Electronics Desktop configuration to focus on your work priorities.



Product Licensing

Ansys Electronics Desktop now uses Electronics Pro, Premium, Enterprise (PPE) product licensing. Legacy product licensing and DSO are no longer supported.

With PPE, HPC licensing is used to enable all cores, GPUs, and distributed tasks.

Distributing Optimetrics Variations

Twin Builder distributes Optimetrics variations with Ansys HPC licenses.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder also requires the on-demand model licenses for Modelica and Dynamic ROM model simulation.

Desktop Configuration

This section contains the following: **Set targeted configuration**, **Custom Menu Set**, and **Schematic Environment**.

- **Set targeted configuration** Each selection here also affects the default selections for the **Custom Menu Set** and **Schematic Environment** options described below, as well as the default design type for a new project. Select the desired targeted configuration from the **Set targeted configuration** dropdown menu.
    - **All** – Set the **Custom Menu Set** to **Default**, the **Schematic Environment** to **Circuit** and the default design type for a new project to **HFSS**.
    - **EM** – Set the **Custom Menu Set** to **EM**, the **Schematic Environment** to nothing, and the default design type for a new project to **Maxwell 3D**.
    - **RF** – Set the **Custom Menu Set** to **RF**, the **Schematic Environment** to **Circuit** and the default design type for a new project to **HFSS**.
    - **SI** – Set the **Custom Menu Set** to **SI**, the **Schematic Environment** to **Circuit** and the default design type for a new project to **HFSS 3D Layout Design**.
    - **Twin Builder** – Set the **Custom Menu Set** to **Twin Builder**, the **Schematic Environment** to **Twin Builder** and the default design type for a new project to **Twin Builder**.
- **Custom Menu Set** shows these options:
    - **Default** shows every solver in the **Project** menu.
    - **EM** shows the Electromagnetics solvers in the **Project** menu.
    - **RF** shows the Radio Frequency solvers, while RF.0 shows all solvers.
    - **SI** shows the Signal Integrity solvers.
    - **SI1** and **SI1.0** show HFSS 3D Layout, Circuit, and Circuit Netlist, as well as adding the **Import**, **Automation**, and **Definitions** toolbar menus.
    - **SI2** and **SI2.0** show all solvers but do not include the **Import**, **Automation**, and **Definitions** toolbar menus.
    - **Twin Builder** shows only the Twin Builder solver on the **Project** menu.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:** The default menu set selected depends on the selected targeted configuration.

- **Schematic Environment** shows these options: **Circuit**, **Twin Builder**, **Maxwell**.

New Project Options

The settings in this section specify whether a design of a particular type is inserted when you create a new project. If you define Twin Builder in **Set targeted configuration**, the default for **Insert a design of type** is Twin Builder. You can select other options from the drop-down list, or select the **Don't insert a design** option. The default is based on the selected targeted configuration.

Save Options

The setting in this section enable autosave and the interval in edits for autosaving. To enable autosave, select the **Do Autosave** check box. Set the number of editing actions after which a backup file is automatically saved to your hard drive in the **Autosave interval** text box. The default is **10**. The backup file is given the same name as your project and has an **aedt.auto** file extension.

Beta Options

Click **Beta Options** to open a list of in-process features. Select an option to enable it in your current session.

Deprecated Features

Click **Deprecated Features** to open a list of legacy features. Select an option to enable it in your current session.

General Options: Project

Select **General** > **Project** in the **Options** dialog box to set these options.

Identify design types to be opened in read-only mode by selecting the check boxes next to the design types.

- 2D Extractor
- Circuit Design
- Circuit Netlist
- EMIT
- HFSS
- HFSS 3D Layout Design
- HFSS-IE
- Icepak

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Maxwell 2D
- Maxwell 3D
- Q3D Extractor
- RMxprt
- Twin Builder

General Options: Miscellaneous

These options are set in the **Miscellaneous** panel under **General** in the **Options** dialog box.

Ansys Workbench Integration

The Ansys Workbench Application **Path** lets you specify a path to an Ansys Workbench installation, if you have one. This path can be used by the Optimetrics feature for connecting to the DesignXplorer.

MATLAB Optimization

If you have an installation of MATLAB installed you can use it as an Optimizer. This MATLAB path setting must to point to the version of MATLAB to be used for performing the optimization. .

> **Note:**
>
> The platform (64-bit of the specified version of MATLAB) must match the platform of this application.

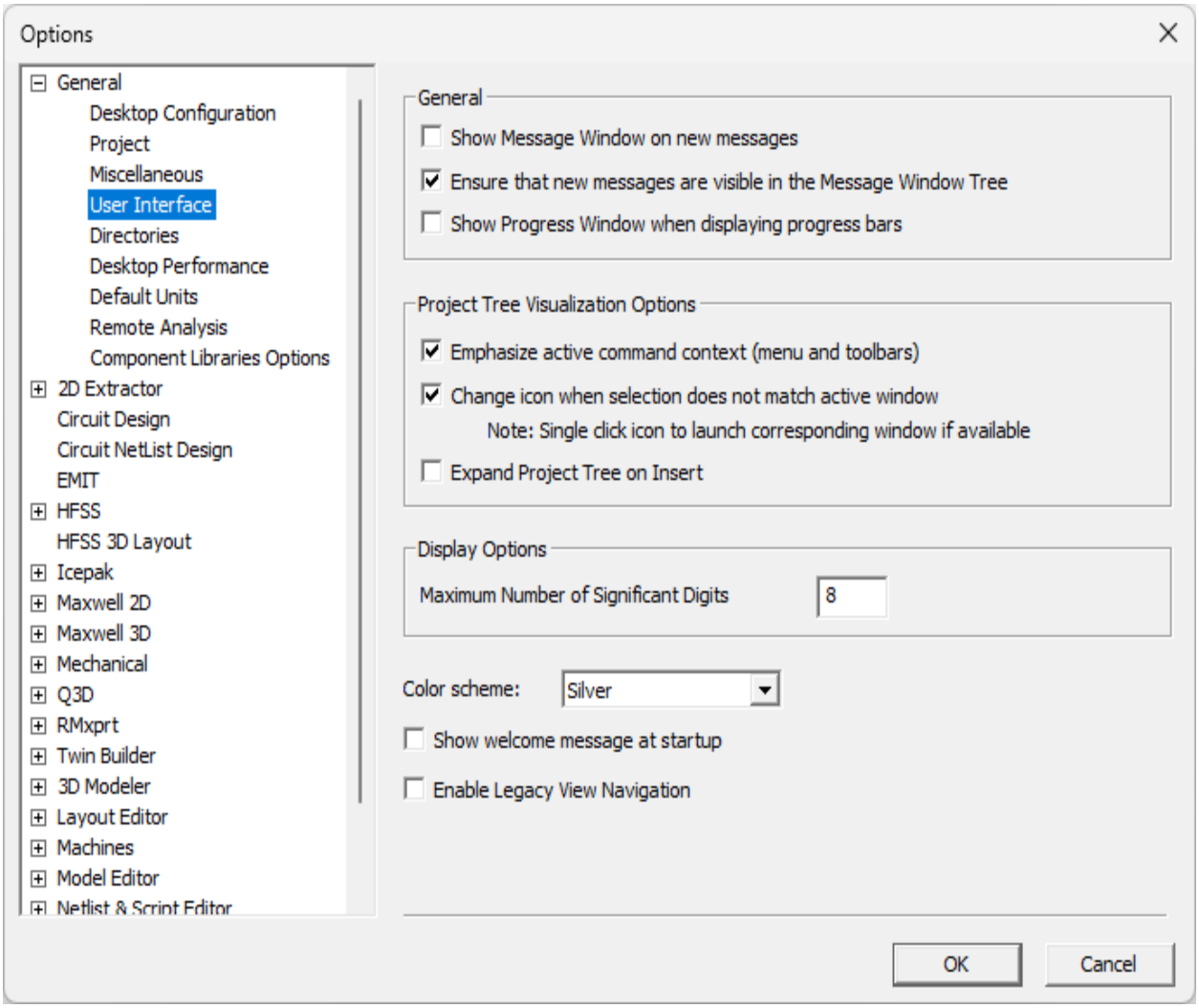
Save Preview Images

Click **Save preview images in project file** if you want the images to be saved in your projects.

General Options: User Interface

Under General Options, **User Interface** options allow you to change how Ansys Electronics Desktop displays messages, command text, the project tree, welcome messages, and more.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



In the **General** area, options include:

- **Show Message Window on new messages** – When selected, the message window automatically opens if a message arrives.
- **Ensure that new messages are visible in the Message Window Tree** – When selected, the message window expands as needed to display messages.
- **Show Progress Window when displaying progress bars** – When selected, the progress window automatically opens while simulations are in progress.

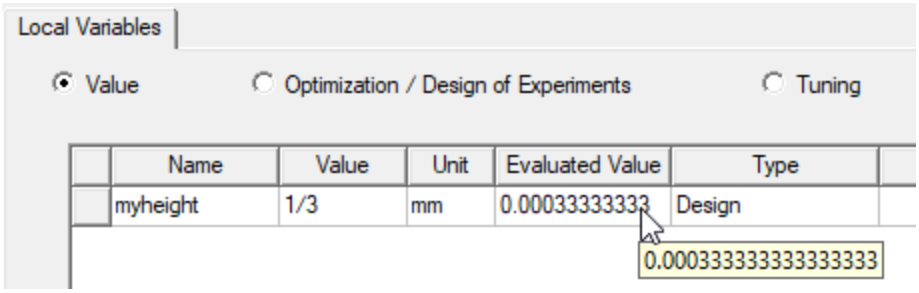
In the **Project Tree Visualization Options** area, options include:

- **Emphasize active command context** – when selected, active elements in the Project Tree display in bold text.

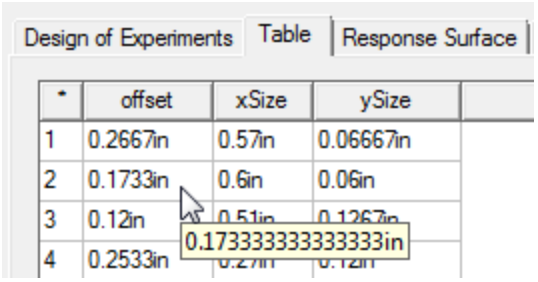
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Change icon when selection does not match active window** – when selected, a small, window-shaped overlay icon displays in the corner of the selected Project Tree element. This icon changes when the data in the active window is unrelated to the selected project item (data affecting the same model is considered to be related). Clicking the icon opens the window and brings it into focus.
- **Expand Project Tree on Insert** – when selected, the Project Tree automatically expands when you insert a new design.

In the **Display Options** area, specify the **Maximum Number of Significant Digits** to display. The default is 8 and the maximum is 20. This affects the digits displayed in the **Solutions** dialog box, evaluated variable values, Animation dialog boxes, Optimetrics, Reports, and so forth. You can still see the full precision values in tooltips by holding a cursor over the displayed value.

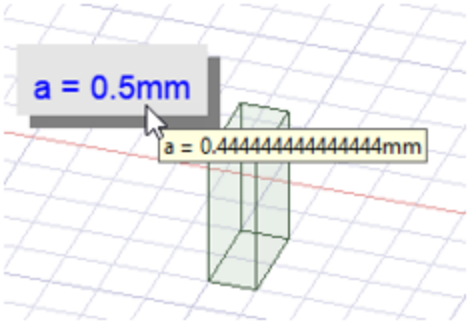


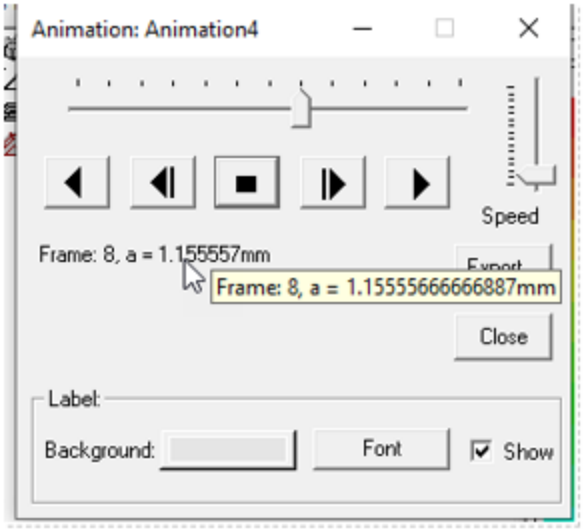
In the case of variable values, if you have assigned more significant digits, you will see these when editing the variable value. In the case of table displays of values, the tooltip display shows all available digits when the mouse pointer is over a result:



The tooltip functions to show internal digits in throughout the Ansys Electronics Desktop interface.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.





Select a **Color Scheme**. The choices are **Light (Beta)**, **Dark (Beta)**, or **Classic**: See Choosing a Color Scheme for more information and samples of each scheme.

Select **Show welcome message at startup** if you want to see a welcome message when Electronics Desktop opens.

Enable legacy view navigation enables you to choose between two schemes for view navigation keyboard shortcuts and mouse button assignments, as follows:

- **Cleared:** Use only the view navigation mouse-button and hotkey assignments introduced in [[[Undefined variable Primary.AnsysElectronicsDesktop]]] version 2024 R1 and applicable to subsequent versions. Legacy (2023 R2 and earlier) mouse-button and hotkey assignments will *not* be recognized when this option is cleared.
- **Selected:** Both the legacy (2023 R2 and earlier) and current (2024 R1 and newer) mouse-button and hotkey assignments are supported, and either can be used for view navigation.

See Choosing the View Navigation Options for more information, including a detailed comparison of the two schemes.

General Options: Directories

These options are set in the **Directories** panel under **General** in the **Options** dialog box. The **Directories** panel contains the following sections:

- The **Project** directory is where your Twin Builder projects and personal libraries (**PersonalLib**) reside, and where Twin Builder first looks to open files. Enter a directory path in the **Project** directory text box, or click ... to find and select the desired directory.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- The **Temp** directory is used by Twin Builder for temporary storage of files during various operations. Enter a directory path in the **Temp** directory text box, or click to find and select the desired directory.
- The **SysLib**, **UserLib**, and **PersonalLib** directories are where Twin Builder system (**syslib**), user (**userlib**), and personal (**PersonalLib**) libraries reside. Enter a directory path in the appropriate directory text box, or click to find and select the desired directory.
- To reset the **SysLib**, **UserLib**, and **PersonalLib** directories to their default directories, click **Reset Library Directories**. By default, the **SysLib** and **UserLib** directories are located in the directory where Twin Builder is installed. The **PersonalLib** default directory resides in the **%UserProfile%\Documents\Ansoft\PersonalLib** directory.

General Options: Desktop Performance Options

These options are set in the **Desktop Performance** panel under **General** in the **Options** dialog box.

Report Update Options for Design Type

- **Design Type** – select the design type for which you are setting report update options.
- **Update reports on file open**– specifies that reports are updated whenever you open a file with solution data. Enabled by default for Twin Builder.
- **Dynamically update reports during edits** – controls the version of solution data to be reported for the specified **Design Type**.

> **Note:**
>
> Twin Builder solution data is versioned. For example, if you analyze and produce solution data for a particular state of a design — then edit the design and re-simulate — solution data is now available for *both* states of the design.

To illustrate the effect of the **Dynamically update postprocessing data during edits** option, consider the following scenario:

- Simulate a design and create a report.

  The report displays the current solution data (*State 1*).

- Make changes to the design.

  The report you created is marked ‘Invalid’ (a large “X” appears in the upper-right corner of the report window and on the report icon in the **Results** section of the **Project** tree).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Re-run the simulation.

The report is updated with new solution data (*State 2*), and the report is marked as valid (the “X” disappears).

- If you click **Undo**:

Twin Builder still has the previous solution data (*State 1*). If you disable **Dynamically update postprocessing data during edits**, the *State 2* report data becomes invalid again. If **Dynamically update postprocessing data during edits** is enabled, Twin Builder reloads the now valid *State 1* solution data into the report and marks it valid.

- If you then click **Redo**:

The result again depends on the setting of the **Dynamically update postprocessing data during edits** option. If the option is disabled, the report becomes valid again (the *State 1* data was not reloaded by the **Undo** operation, so the report is still displaying *State 2*). If the option is enabled, the *State 2* data is reloaded dynamically and report is also valid.

Animation

Computing animated plots requires significant memory which depends upon the complexity of the plot type. The **Animation** setting is used to prevent problems related to low memory should an animation require large memory allocation.

Use the entry field to specify the amount of memory in megabytes to preserve when computing animation frames. Animation frame computation stops when the memory limit is reached.

Desktop Pre/Post Processing

Define the number of processors to be used. This setting applies only to pre/post processing algorithms executed on the desktop. Pre/post processing algorithms executed on the engine use the same number of processors as specified for simulation.

Disk Space Warning

You can also have Twin Builder **Warn when available disk space is less than** the specified number of gigabytes.

General Options: Default Units

The **Default Units** panel under **General** in the **Options** dialog box lets you set the default unit values for the following metrics: **Length**, **Frequency**, **Resistance**, **Angle**, **Power**, **Inductance**, **Time**, **Voltage**, **Capacitance**, **Temperature**, **Current**, **Force**, **Torque**, **Speed**, **Angular Speed**, **Magnetic Induction**, **Weight**, **Magnetic Field Strength**, **Pressure,** and **Conductance**. Select the desired units using the drop-down lists below the corresponding metrics.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> For more information about unit values used in Twin Builder and other Ansys Electromagnetics products, see Unit Types.

These default unit values are applied to any quantity not explicitly assigned a unit specification when entered into a Twin Builder editor.

General Options: Remote Analysis

Under General Options, use the **Remote Analysis** options to launch all analyses as a service or specified user rather than as the current user.

In the **RSM Service Options** area, options include:

- **Ansoft Service Port** – Click **Change** to update the port number. Ansys Electromagnetics RSM Service should be running on this port for all distributed machines.
- **Send Analysis Request As** – Select either **Service User** or **Specified User**. Select **Specified User** to enable the **User Name**, **Password**, and **Domain/Workgroup** fields.

  > **Note:**
  >
  > If any of the remote machines is Unix-based, you must specify the current user.

- **Disable Access By Remote Machines** – If desired, select to disable access for remote machines.

When multiple IP addresses are available, use the **Desktop-Engine Connection** area to specify the preferred IP address for communication:

- **Use Default** – your system's default IP address.
- **Specified Address** – an IP address you specify.
- **Specified Subnet** – a subnet you specify. Subnet may be network prefix and prefix length (123.12.123.0/22), network prefix and subnet mask (123.12.123.0/255.255.252.0), or network prefix only (123.23.123.0).
- **Use Loopback Address for Local Solves** – modifies how Electronics Desktop connects to the simulation engine when solving on a local machine. Selecting this option can make the connection more reliable and avoid interruptions associated with connecting and disconnecting with VPN.

Changing the Listening Port used by Ansys RSM Service

To change the listening port used by the RSM Service, you must change the ansoftrsmservice.cfg file as follows:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Specify the ListenPort within a 'CommDetails' block, which must be within a 'Default:CommDetails' block, which must be within the top level block of the file (the 'AnsoftCOMDaemon' block). The following example changes the listen port from 32958 to 32957, with these blocks at the beginning of the file:

```
$begin 'AnsoftCOMDaemon'

$begin 'Default:CommDetails'

$begin 'CommDetails'

ListenPort='32957'

$end 'CommDetails'

$end 'Default:CommDetails'

. . . .

$end 'AnsoftCOMDaemon'
```

For the second level block, ensure that there is a single colon character and no spaces or tabs separating the two parts of the block name 'Default:CommDetails'. The third level block, with name 'CommDetails' is also required. Use caution when editing this file by hand, because any typos in the block or value names may cause the data to be ignored.

Related Topics

Remote Analysis

General Options: Component Libraries Options

These options are set in **Component Libraries Options** under **General** in the **Options** dialog box.

You can **Use extractor mode by default for separate dynamic link desktops** by selecting the check box.

You can **Run affected property callbacks when variable changes value** by selecting the check box.

The **Component Tree Options** panel contains these controls:

- **Show Favorites** specifies that a list of favorites, in the form of an expandable tree, displays in the Component Libraries's Components tab window.
- **Show Most Recently Used** specifies that a list of the components you have most recently used displays in the project window.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Most Recently Used list contains** specifies the number of recently used components to display.

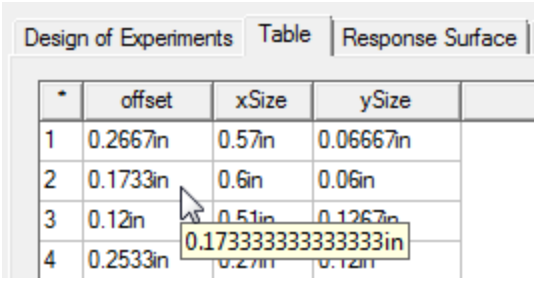
The **Search Options** area contains the following controls:

- Use **Set** to determine whether searches are performed on **All components**, **Current list only**, or **Append to current list**.
- **Load Libraries at Initialization** – When selected, Twin Builder pre-loads all system and configured libraries into the Component Libraries search database when the Component Libraries are initialized at Twin Builder startup. Clear this option to reduce the startup time of Twin Builder, in which case the libraries are loaded at first search.

Options: Optimetrics Options

To set Optimetrics options, select **Tools** > **Options** > **General Options** to open the **Options** dialog box, then select **Optimetrics**. The **Optimetrics** options involve the maximum number of significant digits to be displayed when showing analysis results:

With variable values, if you have assigned more significant digits, you will see these when editing the variable value. With table displays of values, the tooltip display shows all available digits when the mouse pointer is over a result.



Related Topics

Optimetrics

Setting Twin Builder Options

To set Twin Builder options for new designs:

1. Click **Tools** > **Options** > **General Options** to open the **Options** dialog box, then select **Twin Builder** in the options tree. The **Twin Builder** options are:
   - General
   - Port Options
   - C-Model Options

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Spice Compiler
- VHDL Compiler
- Modelica Compiler
- Python Model Options

2. Make changes to the desired options.
3. When finished, click **OK** to accept the changes and close the dialog box.

Twin Builder Options: General Options

Dynamic Plot Update

- **Update reports during simulation** – Enables dynamic update of plots during simulations.

Synchronizing Libraries

- **Synchronize all libraries after export** – All libraries are synchronized after model export. This setting is enabled by default.
- The default for **Array expansion limit in output** is 256.

> **Note:**
>
> The locations of the Personal Library (**PersonalLib**) and User Library (**UserLib**) directories are determined by the **Project Directory** and **Library Directory** settings in **General Options** > **Directories**.

Legacy DSO Usage

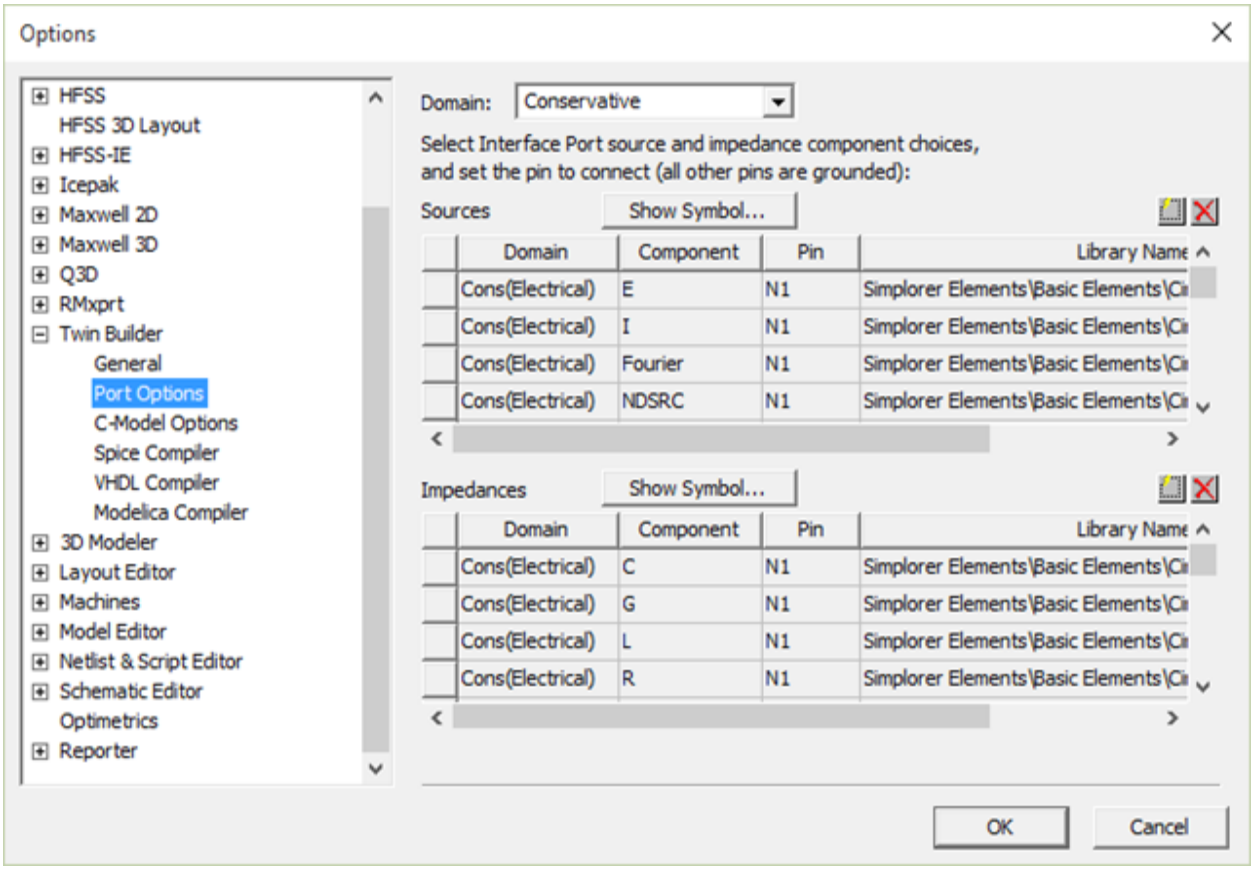
- **Use Legacy DSO License for Parametric Analysis** – You can use legacy DSO licenses for parametric analyses if all analyzed designs are Twin Builder designs and do not contain coupling components. Otherwise, parametric analyses use HPC licenses.

Twin Builder Options: Port Options

The **Port Options** section lets you set up the lists of source and impedance components used for interface ports. The **Domain** determines whether the **Sources** and **Impedances** controls are enabled. For **Conservative**, both sources and impedances controls are enabled. For **Quantity** and **Parametric** domain, only the sources controls are enabled.

> **Note:**
>
> Twin Builder: **Port Options** is available when a Twin Builder design is active.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Follow this procedure to add a component:

1. Click and select a component from a library in the component library Select Definition dialog box.
2. Select the **Pin** to be connected to the port net from the drop-down list. Click **Show Symbol** to allow the symbol, including its pin names, to be seen and resized to make the pin choice easier.

To remove a component from the list, select it and click . You can also remove multiple components from the list: shift-click to select a contiguous block of components, or Ctrl-click to select multiple components individually.

Available port source and impedance components default to the sets listed below:

- Conservative Domain Port Source Components
- Conservative Domain Port Impedance Components
- Quantity Domain Port Source Components

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Conservative Domain Port Source Components

For conservative ports, the default source components available depend on the port's nature as listed below.

Electrical Nature default port source components
• Simplorer Elements\Basic Elements\Circuit\Sources:**E** (Voltage Source)
• Simplorer Elements\Basic Elements\Circuit\Sources:**I** (Current Source)
• Simplorer Elements\Basic Elements\Circuit\Sources:**Fourier**
• Simplorer Elements\Basic Elements\Circuit\Sources:**PSRC**
• Simplorer Elements\Basic Elements VHDLAMS\Circuit\Sources:**e** (VHDL Voltage Source)
• Simplorer Elements\Basic Elements VHDLAMS\Circuit\Sources:**i** (VHDL Current Source)

Thermal Nature default port source components
• Simplorer Elements\Basic Elements\Physical Domains\Thermal:**T** (Temperature source)
• Simplorer Elements\Basic Elements\Physical Domains\Thermal:**H** (Heat flow source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Thermal:**t** (VHDL Temperature source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Thermal:**h** (VHDL Heat flow source)

Fluidic Nature default port source components
• Simplorer Elements\Basic Elements\Physical Domains\Fluidic:**P** (Pressure source)
• Simplorer Elements\Basic Elements\Physical Domains\Fluidic:**Q** (Flow source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Fluidic:**p** (VHDL Pressure source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Fluidic:**q** (VHDL Flow source)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Magnetic Nature default port source components
• Simplorer Elements\Basic Elements\Physical Domains\Magnetic:**FLUX** (Flux source)
• Simplorer Elements\Basic Elements\Physical Domains\Magnetic:**MMF** (Magnetomotive force source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Magnetic:**fluxsrc** (VHDL Flux source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Magnetic:**mmfsrc** (VHDL Magnetomotive force source)

Mechanical (Displacement-Force representation) Nature default port source components
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**F_ROTB** (Torque source)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**S_ROTB** (Angle source)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**V_ROTB** (Angular velocity source)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-Force-Representation\Translational:**F_TRB** (Force source)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-Force-Representation\Translational:**S_TRB** (Position source)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-Force-Representation\Translational:**V_TRB** (Velocity source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**f_rotb** (Torque source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**s_rotb** (Angle source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**v_rotb** (Angular velocity source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Mechanical (Displacement-Force representation) Nature default port source components
Domains\Mechanical\Displacement-Force-Representation\Translational:**f_trb** (Force source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Displacement-Force-Representation\Translational:**s_trb** (Position source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Displacement-Force-Representation\Translational:**v_trb** (Velocity source)

Mechanical (Velocity-Force representation) Nature default port source components
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational_V:**F_ROT** (Torque source)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational_V:**V_ROT** (Angular velocity source)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Translational_V:**F_TR** (Force source)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Translational_V:**V_TR** (Velocity source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational V:**f_rot** (Torque source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational V:**v_rot** (Angular velocity source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Translational V:**f_tr** (Force source)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Translational V:**v_tr** (Velocity source)

Conservative Domain Port Impedance Components

For conservative ports, the default impedance components available depend on the port's nature as listed below.

Electrical Nature default port impedance components
• Simplorer Elements\Basic Elements\Circuit\Passive Elements:**C**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Electrical Nature default port impedance components
(Capacitor)
• Simplorer Elements\Basic Elements\Circuit\Passive Elements:**G** (Conductor)
• Simplorer Elements\Basic Elements\Circuit\Passive Elements:**L** (Inductor)
• Simplorer Elements\Basic Elements\Circuit\Passive Elements:**R** (Resistor)
• Simplorer Elements\Basic Elements VHDLAMS\Circuit\Passive Elements:**c** (VHDL Capacitor)
• Simplorer Elements\Basic Elements VHDLAMS\Circuit\Passive Elements:**g** (VHDL Conductor)
• Simplorer Elements\Basic Elements VHDLAMS\Circuit\Passive Elements:**l** (VHDL Inductor)
• Simplorer Elements\Basic Elements VHDLAMS\Circuit\Passive Elements:**r** (VHDL Resistor)

Thermal Nature default port impedance components
• Simplorer Elements\Basic Elements\Physical Domains\Thermal:**CTH** (Thermal capacitance)
• Simplorer Elements\Basic Elements\Physical Domains\Thermal:**RTH** (Thermal resistance)
• Simplorer Elements\Basic Elements\Physical Domains VHDLAMS\Thermal:**cth** (VHDL Thermal capacitance)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Thermal:**rth** (VHDL Thermal resistance)

Fluidic Nature default port impedance components
• Simplorer Elements\Basic Elements\Physical Domains\Fluidic:**CHYD** (Hydraulic capacitance)
• Simplorer Elements\Basic Elements\Physical Domains\Fluidic:**LHYD**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Fluidic Nature default port impedance components
(Hydraulic inductance)
• Simplorer Elements\Basic Elements\Physical Domains\Fluidic:**RHYD** (Linear orifice)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Fluidic:**chyd** (VHDL Hydraulic capacitance)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Fluidic:**lhyd** (VHDL Hydraulic inductance)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Fluidic:**rhyd** (VHDL Linear orifice)

Magnetic Nature default port impedance components
• Simplorer Elements\Basic Elements\Physical Domains\Magnetic:**RMAG** (Magnetoresistor)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Magnetic:**rmag** (VHDL Magnetoreluctance)

Mechanical (Displacement-Force representation) Nature default port impedance components
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**DAMP_ROTB** (Damper)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**MASS_ROTB** (Mass)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**damp_rotb** (VHDL Damper)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Displacement-Force-Representation\Rotational:**mass_rotb** (VHDL Mass)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-Force-Representation\Translational:**DAMP_TRB** (Damper)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Displacement-

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Mechanical (Displacement-Force representation) Nature default port impedance components
Force-Representation\Translational:**MASS_TRB** (Mass)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Displacement-Force-Representation\Translational:**damp_trb** (VHDL Damper)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Displacement-Force-Representation\Translational:**mass_trb** (VHDL Mass)

Mechanical (Velocity-Force representation) Nature default port impedance components
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational_V:**DAMP_ROT** (Damper)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational_V:**MASS_ROT** (Mass)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational V:**damp_rot** (VHDL Damper)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational V:**mass_rot** (VHDL Mass)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Translational_V:**DAMP_TR** (Damper)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Translational_V:**MASS_TR** (Mass)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Translational V:**damp_tr** (VHDL Damper)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Translational V:**mass_tr** (VHDL Mass)

Mechanical (Velocity-Force representation) Nature default port impedance components
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational_V:**DAMP_ROT** (Damper)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational_V:**MASS_ROT** (Mass)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Mechanical (Velocity-Force representation) Nature default port impedance components
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational V:**damp_rot** (VHDL Damper)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Rotational V:**mass_rot** (VHDL Mass)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Translational_V:**DAMP_TR** (Damper)
• Simplorer Elements\Basic Elements\Physical Domains\Mechanical\Velocity-Force-Representation\Translational_V:**MASS_TR** (Mass)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Translational V:**damp_tr** (VHDL Damper)
• Simplorer Elements\Basic Elements VHDLAMS\Physical Domains\Mechanical\Velocity-Force-Representation\Translational V:**mass_tr** (VHDL Mass)

Quantity Domain Port Source Components

Description	**Library Path**
Const	Simplorer Elements\Basic Elements\Blocks\Sources:**CONST**
Step	Simplorer Elements\Basic Elements\Blocks\Sources:**STEP**
Random	Simplorer Elements\Basic Elements\Blocks\Sources:**RANDOM**
(VHDL) Const	Simplorer Elements\Basic Elements VHDLAMS\Blocks\Sources:**const**
(VHDL) Step	Simplorer Elements\Basic Elements VHDLAMS\Blocks\Sources:**step**
(VHDL) Random	Simplorer Elements\Basic Elements VHDLAMS\Blocks\Sources: **random**
Exponential function	Simplorer Elements\Basic Elements\Tools\Characteristics:**EXP**
Hyperbolic function	Simplorer Elements\Basic Elements\Tools\Characteristics:**HYP**
Second order polynomial function	Simplorer Elements\Basic Elements\Tools\Characteristics:**PO2**
2D lookup table	Simplorer Elements\Basic Elements\Tools\Characteristics:**XY**
Arctangent function	Simplorer Elements\Basic Elements\Tools\Time Functions:**ARCTAN**
2D lookup table with interpolation	Simplorer Elements\Basic Elements\Tools\Time Functions:**DATAPAIRS**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Description	Library Path
Needle function	Simplorer Elements\Basic Elements\Tools\Time Functions:**NEEDLE**
Pulse function	Simplorer Elements\Basic Elements\Tools\Time Functions:**PULSE**
Pulse width modulation function	Simplorer Elements\Basic Elements\Tools\Time Functions:**PWM**
Sawtooth function	Simplorer Elements\Basic Elements\Tools\Time Functions:**SAWTOOTH**
Sinusoidal function	Simplorer Elements\Basic Elements\Tools\Time Functions:**SINE**
Trapezoidal function	Simplorer Elements\Basic Elements\Tools\Time Functions:**TRAPEZ**
Triangular function	Simplorer Elements\Basic Elements\Tools\Time Functions:**TRIANG**
(VHDL) Arctangent function	Simplorer Elements\Basic Elements VHDLAMS\Tools\Time Functions:**arctan**
(VHDL) Needle function	Simplorer Elements\Basic Elements VHDLAMS\Tools\Time Functions:**needle**
(VHDL) Pulse function	Simplorer Elements\Basic Elements VHDLAMS\Tools\Time Functions:**pulse**
(VHDL) Sawtooth falling function	Simplorer Elements\Basic Elements VHDLAMS\Tools\Time Functions:**sawtoothl**
(VHDL) Sawtooth rising function	Simplorer Elements\Basic Elements VHDLAMS\Tools\Time Functions:**sawtoothr**
(VHDL) Sinusoidal function	Simplorer Elements\Basic Elements VHDLAMS\Tools\Time Functions:**sine**
(VHDL) Trapezoidal function	Simplorer Elements\Basic Elements VHDLAMS\Tools\Time Functions:**trapez**
(VHDL) Triangular function	Simplorer Elements\Basic Elements\Tools\Time Functions:**triang**

Related Topics

Setting Interface Port Properties

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder Options: C-Model Options

Use these options to define settings for the compilation and import of C-Models.

- **C++ Compiler** – Set the C++ compiler used to compile C-Models. By default, the latest version of the installed Visual Studio C++ compilers is used for compilation. Currently VS C++ 2015, VS C++ 2017, VS C++ 2019, and VS C++ 2022 are supported. Professional and Community versions are supported. For more details, see ***Program Requirements*** in the ***Introduction to the Twin Builder C Interface*** section.
- **Model DLL Location** – Configure the default handling of binaries during model import.
  - **Keep DLL on Disc** – Determine whether the binaries are stored on disc as .**dll** files or copied into the model definition in the project. Clear this check box to store the binaries with the model; they will be extracted to a temp location for simulation.
  - **Warn about Override/Replace of Existing DLL** – Determine whether a warning message appears when the generated C-Model DLL replaces an existing DLL at the same location, or if it overrides a C-Model DLL with the same name that is located in a lower-priority location. For example, a C-Model DLL in **PersonalLib\bin** overrides a C-Model DLL with the same name in **UserLib\bin** or **SysLib\bin**.
  - **Copy DLL to Bin Directory of:** – Select where to copy the binaries during model import. Binaries saved to disc as .**dll** files must be stored in **<PersonalLib>\bin** or **<UserLib>\bin**.
- **Default Model Source** – Configure where to store the C-Model sources.
  - **None** – The sources for the C-Model are not stored in a separate location or kept on disc after the model editor is closed. The compiled C-Model will have no knowledge about the model sources and cannot reopen them at a later point. This setting is useful if the C-Model will be given away and there is no need or desire to reopen and edit the model again.
  - **Path to Project** – The generated C-Model component stores the location of the source files. The project for the C-Model can be reopened and edited in the C-Model editor later if the sources are still located at the same place.
  - **Zip Model Source** – The generated C-Model component stores the model sources. The project for the C-Model can be reopened and edited in the C-Model editor later, and the model sources get restored to a temporary location. This setting is useful if anyone can edit the model.
- **Remote Solve** – Configure how the system copies C-Models to the remote site when using a remote solver.
  - **Copy Model DLLs to Remote Location manually** – When you select this check box, the remote solver expects all required C-Model DLLs which are not stored with the C-Model (see **Model DLL Location** above) to be accessible from one of the library locations on the remote machine (**PersonalLib\Bin** or **UserLib\Bin**). Clear this check box to transfer the C-Model DLLs to a temporary location on the remote machine before the start of each simulation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> The auto-copy function does not work for sub-DLLs that are loaded (dynamically loaded or statically linked) by the top C-Model DLL.

Twin Builder Options: Spice Compiler Options Tab

- **Default Spice Type** – Sets the default Spice compiler type either to Berkeley Spice or PSPICE.
- **Import Spice Text as** – Determines whether Spice text will be imported as SPICE or SML.
- **Create Testbench from Main Circuit** – When enabled, the compiler creates a Spice Testbench from the main circuit.

Related Topics

Setting Twin Builder Options

General Options

Twin Builder Options: VHDLCompiler Options Tab

These options let you define settings for the compilation VHDL-AMS models.

- **C++ Compiler** – Lets you set the C++ compiler that is used to compile VHDL-AMS models. By default, the Gcc compiler included in the installation (<*Installpath*>**\Common\MinGW**) is used for compilation. It is possible to use a Visual Studio C++ compiler. Currently VS C++2015, VS C++2017, VS C++2019, and VS C++2022 are supported. For more details, see ***Program Requirements*** in the ***Introduction to the Twin Builder C Interface*** section.
- **Use compiler optimization** – This is the default optimization setting during the C++ compilation when creating a new VHDL-AMS model. This setting can later be changed on a per-model basis in the VHDL model editor settings.
- **Message Filter** – These options let you toggle the display of warning and trace messages during the compilation of VHDL models and packages.
  - **Trace messages** – Informational messages, such as the status of the compile process, and loaded sub-models.
  - **Warning messages** – Low impact errors that do not cause the compiler to stop. The errors might cause the model to not function properly.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Error messages (other than warning messages) always display.

Twin Builder Options: Modelica Compiler Options

Use these options to define settings for the compilation of Modelica models.

- **Memory Usage Limit** – Limit the use of memory by the Modelica compiler. Increase this number when model compilation fails due to memory size.
- **Modelica C-compiler** – Set the C++ compiler used to compile Modelica models.
  - **GCC (internal)** – Default. Use the Gcc compiler from <*Installpath*>**\Common\MinGW-5.1.0**.
  - **GCC (user defined)** – Use the Gcc compiler from the MinGW location specified in the text control. To change it, click ... .
  - **Visual Studio** – Use the C compiler from an installed Visual Studio. Currently VS 2015, VS 2017, VS 2019, and VS 2022 are supported. For more details, see ***Program Requirements*** in the ***Introduction to the Twin Builder C Interface*** section.

    > **Note:**
    >
    > If the Modelica model incorporates external compiled C-code (external DLLs), compile the Modelica model with a Visual Studio C++ compiler.
- **JRE Location** – Define the location where the Modelica compiler looks for the Java Runtime Environment (JRE).
  - **Use Internal JRE** – Use the JRE included in the installation.
  - **Use JRE from JAVA_HOME** – Determine the location of the JRE by the JAVA_HOME environment variable.
  - **User-defined location** – Access the JRE from the location specified in the text control. To change it, click ... .

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> - Twin Builder requires 64-bit Java when using a JRE other than the one included in the installation.
> - Changes to the location of the JRE used for the Modelica compiler take effect after restarting Twin Builder.

- **Advanced Options** – Define non-standard settings for the Modelica compiler.
  - **Advanced Command-Line Options** – Settings for non-standard features of the Modelica model compilation. Those settings can be obtained from Modelica support on a case-by-case basis.
  - **Advanced Settings** – Other advanced settings.
    - **Keep Temp Files** – Keep temporary Modelica files that are generated during compilation in a separate folder. Use this setting to collect all files necessary for the investigation of a support ticket. The default is **No**.
    - **Allow Lib Version Mismatch** – If the library version of an instantiated model is lower than the required version specified in an annotation, the Modelica compiler issues a warning instead of an error. The default is **Yes**.
- **Messages & Diagnostics** – Define the setting location for additional information during model compilation.
  - **Message Level** – Define the level of messages displayed in the **Message Manager** pane during the compilation of Modelica models.
    - **Errors** – Only errors appear in the **Message Manager** pane.
    - **Warnings** – Errors and warnings appear in the **Message Manager** pane.
    - **All** – All messages appear in the **Message Manager** pane.
  - **Log Level** – Choose the level of detail that appears in the Modelica compiler log file. The default is **Errors**.
  - **Diagnostics** – Generate additional diagnostics information about the Modelica model. The default is **OFF**.
  - **Save To** – Define the location where the Modelica compiler stores additional information during model compilation, and the diagnostics. The default location is the temp folder defined in the **General** > **Directories** page. The diagnostics and compiler log information display in a browser when you double-click the messages in the **Message Manager** pane.

Twin Builder Options: Python Model Options

Use these options to define settings for the incorporation of Python models in the simulation process.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Python Location (x64)** – Browse and select the Python installation folder. The installation must be 64-bit and must include **/libs/** and **/include/** folders.

  > **Note:** Make sure to add the Python installation path to your system or user environment variable (PATH).

- **C++ Compiler** – Choose from a list of Microsoft® Visual Studio® C++ compilers installed on your machine that are supported by Twin Builder. Visual Studio C++ Community or Professional version are required. For more details, see ***Program Requirements*** in the ***Introduction to the Twin Builder C Interface*** section.

Model Editor Settings

You can set text options for the following Twin Builder model editors:

- C-Model
- SML
- Spice
- VHDL
- VHDL Package
- Modelica

1. Select **Tools** > **Options** > **General Options**, then expand **Model Editor**.
2. Select a model editor, then click its text editor to make changes.
   - Select **Display Line Numbers** to display line numbers in the editing window.
   - Click **Bookmark** to open a **Color** selection dialog box in which you can select a color for bookmarks.
   - The font panel displays the current font **Name** selection and font **Size**. Some non-editable sample text displays the currently selected font and size. To change the font, select the desired font name from the **Name** drop-down list. To change the font point size, edit the Size text field.
3. Make the desired selections, and click **OK**.

For information about the settings for Modelica, see Modelica Model Editor Options.

Setting HPC and Analysis Options

All analysis parameters are accessed with a single dialog box. The machine list and options settings are integrated into analysis configurations. The default configuration is for solving on a single, local machine. You can create many analysis configurations for remote and distributed solutions, and switch between them depending on the job being solved. Multiprocessing is integrated into the machine lists.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

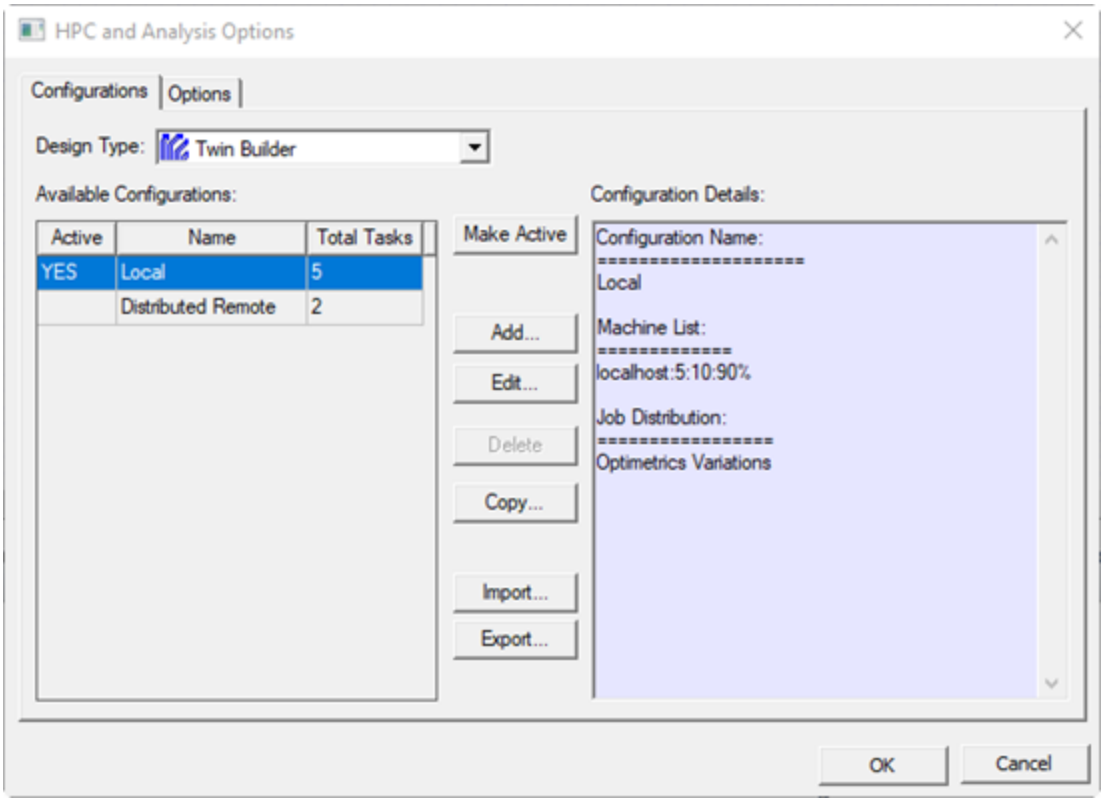
To set the HPC and analysis options, select **Tools** > **Options** > **HPC and Analysis Options**. The **HPC and Analysis Options** dialog box appears, displaying two tabs: **Configurations** and **Options**.

Distributing Optimetrics Variations

Twin Builder does not support any distribution for multiple cores, tasks, or GPUs. However, to support HPC analysis in Co-simulations some settings like cores and processor usage might be required (see ***Editing Distributed Machine Configurations***).

In general, Twin Builder distributes Optimetrics variations using Ansys HPC licenses. In specific cases (see ***Using Legacy DSO Licenses for Parametric Analysis in Twin Builder***) it is still possible to distribute variations using the legacy distributed solve option (DSO) license. Future releases, however, will require HPC licenses for distributed solves.

Distributed analysis still requires certain on-demand model licenses (for example, for Modelica, ROM, and Twin models) during simulation if those models are used in the design.



Configurations Tab

In the **Configurations** tab, you can select the design type and associated configurations. Select a design type to display a list of the available configurations for that type. Select a configuration

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

to display the details of that configuration. A name can describe the use for which a configuration has been defined. The **Total Tasks** column shows the number of tasks that the analysis configuration can execute.

Design Type

Define configurations for the Twin Builder design type. The Active configuration is used when solving an analysis for the design type.

Available Configurations List

Click the desired configuration in the **Available Configurations** list for each design type, then click **Make Active**. The active configuration is indicated with a **YES** in the **Active** column.

- **Add...** – Create a new analysis configuration.
- **Edit...** – Edit the currently selected analysis configuration.
- **Delete** – Delete the currently selected analysis configurations.

> **Note:**
>
> You cannot delete the local configuration.

- **Copy...** – Create a new analysis configuration, and launches the **Analysis Configuration** dialog box to edit it.
- **Import...** – Import an **.acf** file to create an analysis configuration.

> **Note:**
>
> Importing analysis configurations always adds the imported analysis configurations to the current design type. If there is a name conflict between an imported analysis configuration and an existing analysis configuration, the imported configuration is renamed, and you are notified.

- **Export...** – Export the selected analysis configurations to an **.acf** file. You can then import the configurations into a different design type or import them on a different machine.

Options Tab

For HPC License, select **Workgroup** or **Pack**.

HPC licensing enables the use of cores and GPUs to accelerate simulations. In general, each core requires one unit of HPC, while each GPU requires eight units. The selected HPC license type determines which license is used, and how units of HPC are converted to license counts.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Workgroup** (formerly "pool") – One HPC workgroup license enables one unit of HPC.
- **Pack** – One HPC pack license enables eight units of HPC. Additional packs multiply by four, enabling 32, 128, 512,... , in the context of a single simulation.

Electronics Desktop products include four units of HPC for each licensed simulation. This means that up to four units can be used without requiring HPC licenses; license counting begins with the fifth unit. For example, a simulation that uses 36 cores requires 32 HPC units after subtracting the four included cores. This simulation will check out 32 HPC workgroup licenses, or two HPC pack licenses.

HPC licenses enable all parallel and distributed simulations, including distributed variations. Distributed variations require a single set of solver licenses, plus HPC to enable the variations.

For HPC Workgroup, distributing *N* variations requires 8*(*N*–1) workgroup licenses and, together with the solver licenses, enables up to four HPC units per variation. Each additional set of *N* workgroup licenses will enable one additional HPC unit per variation. For HPC Pack, distributing *N* variations requires *N*–1 pack licenses and, together with the solver licenses, enables up to four HPC units per variation. Each additional set of *N* pack licenses will enable 8, 32, 128,... additional HPC units per variation.

Ansys licensing supports distributed simulations when Ansys Electronics Desktop is called from other Ansys tools, such as optiSLang and Workbench. In such cases, distributed design points (variations) generally use HPC counts as described above.

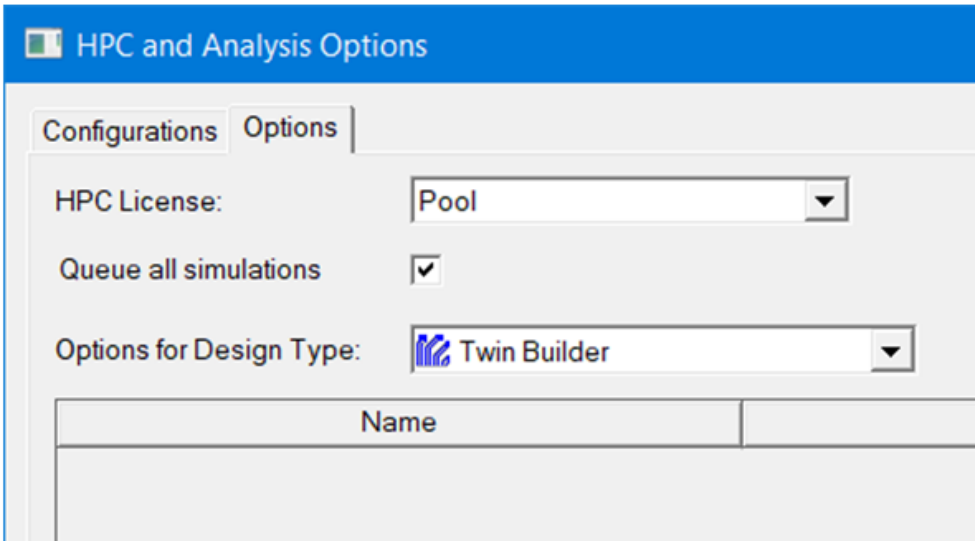
> **Note:**
>
> Licensing for some calling products may include some distributed design points, in which case the total required HPC will be reduced.

The **Options** tab in the **HPC and Analysis Options** dialog box contains design type specific options. These options are not part of an analysis configuration; instead, they are always in effect for the given design type when the following is true:

- A design of the matching design type is being solved.
- You have not specified corresponding overriding batch options on the command line.

You can enable queuing in the **Options** tab. If the **Queue all simulations** check box is selected, the Desktop queues any active simulations for design types that have **Save before solving** turned off in **General Options**, then processes them in order. You can view and change the queue by using Show Queued Simulations.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



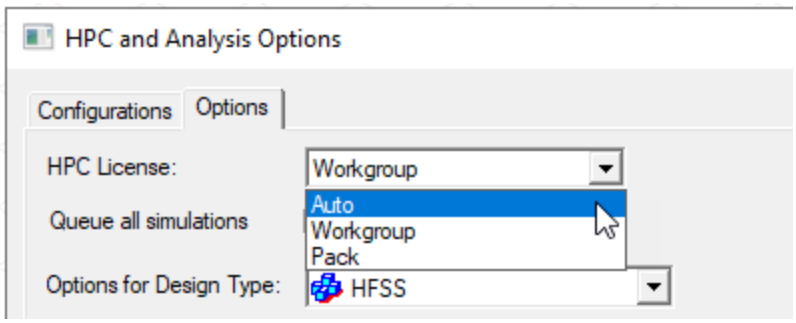
To edit configurations, see ***Editing Distributed Machine Configurations***.

Related Topics

High Performance Computing (HPC) Integration

Licensing Settings Tool

HPC licensing includes a choice called **Auto**, along with **Workgroup**, and **Pack**. The **Auto** choice is available in the HPC License combo box found on the **Options** tab of the **HPC and Analysis Options** window. If you make changes in the License Settings Tool, we recommend that you restart Electronics Desktop.



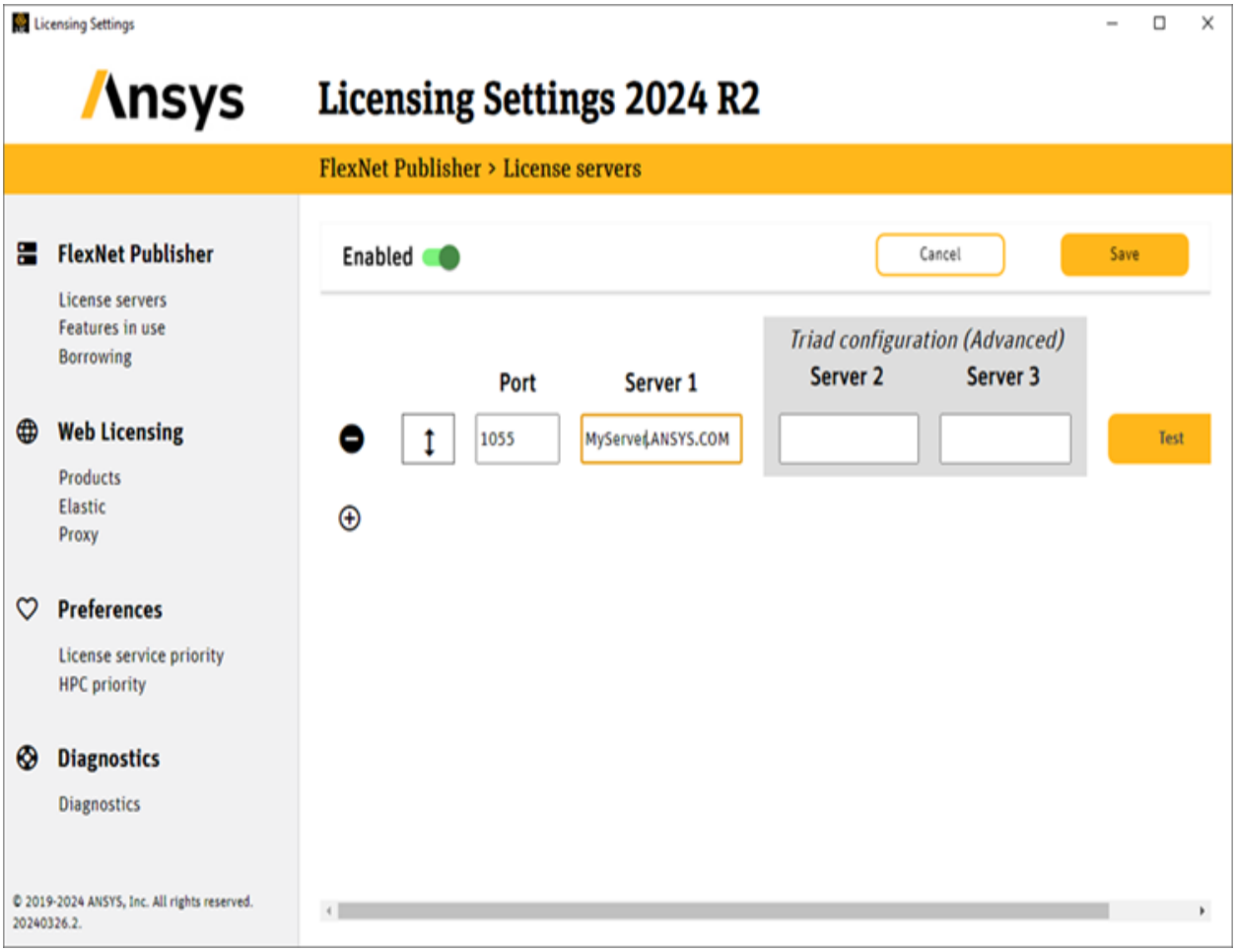
Auto means delegate the choice of Workgroup or Pack licensing to the Ansys Licensing Settings tool, which is a separate application installed along with Electronics Desktop. The HPC settings specified by the Licensing Settings tool are shared by other Ansys products. This provides you a single place where you can set the HPC preferences.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:** Important: HPC licensing preferences in the Licensing Settings tool are used only when **Auto** is selected in Electronics Desktop.

Launching the Licensing Settings Tool from the Main Menu

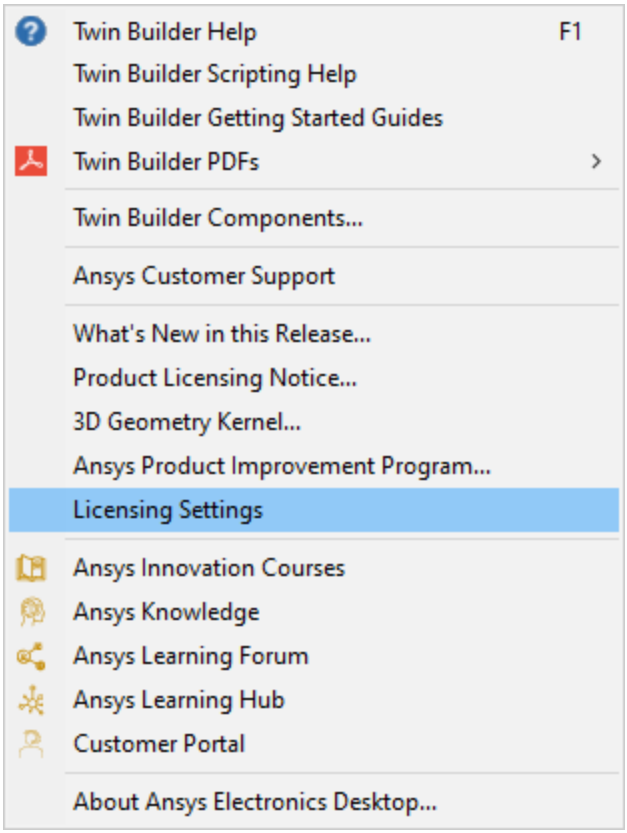
Click **Tools** > **Licensing Settings** to launch the Licensing Settings tool.



Licensing Settings Help

You can obtain help with the Licensing Settings tool from the **Licensing Settings** menu item in the **Help** menu.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



License Fallbacks

When using Auto HPC licensing, in some cases if there aren't enough available licenses of the highest priority HPC license type, the lower priority HPC license type will be used. For instance, assume the user has set the priority order to "Ansys HPC, Ansys HPC Pack" in the Licensing Settings tool. If the user requests a nominal solve using 14 cores (10 Ansys HPC), but 10 Ansys HPC licenses aren't available, 2 Ansys HPC Pack licenses will be used if available.

Exporting Options Files

Options files at all levels that may affect a product running as a specific user on a specific host may easily be exported. Select **Tools** > **Options** > **Export Options Files** and browse to a location to store your files. All of the configuration files for the current user and current host are copied to the specified directory. Config files for the install, install_machine, user, and user_machine levels will be copied, if they exist. One additional file, **admin.XML**, is also copied to the destination directory. This file does not contain user configurable options.

Related Topics

Setting Options for Twin Builder

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Setting Options via Configuration Files

Example Uses for Export Options Features

User Options and the Update Registry Tool

Batchoptions Command Line Examples

Setting Options via Configuration Files

> **Note:**
>
> Because Twin Builder can interact with other Ansys Electromagnetics products, the following sections on setting options via configuration files include examples and information applicable to various other Ansys Electromagnetics products.

In addition to Setting Options for Twin Builder, that is, from the user interface (UI), you can also set options in several configuration files. Option settings in configuration files may apply to all users or only to a specific user, and they may apply to all hosts or only to specific hosts. There are four levels of settings, listed below from most specific (highest precedence) to most general (lowest precedence):

- Host-dependent user options – apply to the specified user on the specified host only
- Host-independent user options – apply to the specified user on all hosts
- Host-dependent default options – apply to all users on the specified host
- Installation default – default for all users on all hosts

In the list above, settings at any level override settings at lower levels . If there is no setting in any file, the application default value is used. See UpdateRegistry for instructions on selecting these levels.

> **Important:**
>
> Options set from the Desktop UI override and update user settings in configuration files. Otherwise, the existing configuration file settings are used.

Topic Navigation:

Behavior Examples

Rules for Modifying Option Settings

Configuration File Locations

Products with Multiple Desktop Applications

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Table of Directories and Files

Behavior Examples

Consider running an application as user **jsmith** on host **host123**. Also, assume the following conditions apply:

- There is no *host-dependent user setting* for the **Expand Project Tree on Insert** option in the *host-dependent user options* config file for user **jsmith** on host **host123**.
- But, the option exists in the *host-independent user options* config file for user **jsmith**.

In this situation, the latter setting is used (if it is not overridden using the Desktop UI). Any settings in the *host-dependent default options* config file or the *installation default* config file are ignored.

As another example, consider running an application as user **jdoe** on host **host123**. Also, assume that the following conditions apply:

- There is no setting for the **Expand Project Tree on Insert** option in the *host-dependent user options* config file for **jdoe** on **host123**.
- There is no setting in the *host-independent user options* config file for user **jdoe**.
- And, there is no setting in the *host-dependent default options* config file for host **host123**.

In this situation, the value from the *installation default* config file is used, if present.

Rules for Modifying Option Settings

Option settings displayed in the Ansys Electronics Desktop UI follow the above rules. That is, if there is a setting in any of the option config files, then the setting from the highest priority config file is displayed in the Desktop UI. If there is no setting in any of the option config files, then the global default value is used.

You can modify settings using the various **Options** dialog boxes accessed via the **Tools** > **Options** menu. Click **OK** to save and immediately implement your changes, and close the dialog box; click **Cancel** to close the dialog box without saving your changes. Changes are written to the host-dependent user options config file when you exit the Electronics Desktop application. The changed values written to this file are then used the next time that the application is launched by the same user on the same host. The Desktop UI option settings are not written to any of the other option config files.

Configuration File Locations

The configuration files for host-dependent and installation **default** options reside at: <*installation_directory*>\<*version*>\<*platform*>\config. The configuration files for host-dependent and host-independent **user** options reside in a subfolder of the user's Documents folder (for Windows) or the user's HOME folder (for Linux). See the tables below for specific Windows and Linux file names and paths.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Products with Multiple Desktop Versions

For products that have multiple Desktop versions, each version has a separate user-specific **config** folder with a different value for <*ApplicationName&Version*> (the parent folder name). Simarly, each version has a separate **config** folder for the *host-dependent default* and *installation default* config files. This folder is under the path, <*InstallationDirectory*>\<*version*>\<*platform*>. See the tables below for specific Windows and Linux file names and paths.

Table of Directories and Files

The following table shows the directories and files, where the **Level Name** is the name used to describe an options config file when using the UpdateRegistry tool:

Config File	Level Name	File Name	Windows Directory Path	Linux Directory Path
host-dependent user options	user_machine	<*hostname*>_user.XML	%UserProfile%\Documents\Ansoft \<*ApplicationName&Version*>\config	$HOME/Ansoft /<*ApplicationName&Version*>/config
host-independent user options	user	user.XML		
host-dependent default options	install_machine	<*hostname*>.XML	<*InstallationDirectory*>\v<*version*> \Win64\config	<*InstallationDirectory*>/v<*version*> /Linux64/config
installation default	install	default.XML		

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

- **<*hostname*>** is the name of the computer on which the Electronics Desktop software is installed
- **%UserProfile%** is a Windows variable that represents the currently active user's profile (for example, C:\Users\JohnDoe)
- **<*ApplicationName&Version*>** is the product name (without spaces) followed by the four-digit year of the version, a decimal point, and the minor release number (such as ElectronicsDesktop2024.2)
- **$HOME** is the user's home directory on Linux
- **<*InstallationDirectory*>** is the root folder where the Electronics Desktop software is installed (typically, C:\Program Files\AnsysEM, on Windows, or /opt/AnsysEM, on Linux)
- **<*Version*>** is the last two digits of the product version's year followed by the minor release number, without a decimal point (such as 232)

The following table shows an example of specific file names and directory names for a typical Ansys Electronics Desktop installation on Microsoft Windows and on Linux. These are the files that apply to software version **2023 R2**, user "**jsmith**," and hostname "**host123**":

Config File	Level Name	File Name	Windows Directory Path	Linux Directory Path
host dependent user options	user_machine	host123_user.XML	C:\Users\jsmith\Documents\Ansoft\ElectronicsDesktop2024.2\config	/home/jsmith/Ansoft/ElectronicsDesktop2024.2/config
host independent user options	user	user.XML		
host dependent default options	install_machine	host123.XML	C:\Program Files\AnsysEM\v242\Win64\config	/opt/AnsysEM/v242/Linux64/config
installation default	install	default.XML		

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

As with the temporary file location configuration files, the settings in these options files have precedence in the following sequence: user_machine (highest precedence), user, install_machine, install (lowest precedence). The first time you start and exit the application, the file at the "user_machine" level is created (<*hostname*>_user.XML). The other files are only created if you use the UpdateRegistry tool to specify an option at the "user," "install_machine," or "install" level. If the temporary directory is set to an empty string in a configuration file, then that setting is ignored.

Related Topics

Setting or Removing Option Values in Configuration Files: UpdateRegistry Command

Example Uses for Export Options Features

User Options and the Update Registry Tool

-Batchoptions Command Line Examples

Setting or Removing Option Values in Configuration Files: UpdateRegistry Command

A command line tool, **UpdateRegistry**, is available to modify option settings in the options config files. You can use this command to add, change, or remove settings from any of the option config files. This tool is included in the installation directory of each product.

This feature is intended to make it easier for different users to use Ansys Electromagnetics tools installed on shared directories or network drives.

UpdateRegistry has multiple command line formats as shown below. The -Set format is used to set or change an option value. The -Delete format is used to delete an option setting. The following command line options are mutually exclusive: -Set, -Get, -Delete, -GetKeys, and -FromFile.

UpdateRegistry -Get Command

This is used to view an option value in an option config file. If the setting exists in the specified config file or files, then the value, the value type and the config file where the value was found will be reported. If no value is found, then that will also be reported.

Usage:

UpdateRegistry -Get -ProductName <name> -RegistryKey <keyPath> [-RegistryLevel <level>]

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<name>

Required. The application or product name and version, as described above. For example, **ElectronicsDesktop2018.1**. If the name contains spaces, it must be enclosed within quotation marks.

<keyPath>

Required. The pathname of the option setting. For example:

Desktop/Settings/ProjectOptions/AnimationMemory.

<level>

Optional. A string denoting which config file to search. One of: **install**, **install_machine**, **user**, and **user_machine**. If the level is not specified, then all config files are searched in order of precedence.

UpdateRegistry -GetKeys Command

Use this command to view the allowed key names for all of the option settings, or to view a subset of the key names that match a string. For each key displayed, the current value, if any, is also reported. If a key has a value in multiple config files, then only the highest precedence value is reported.

Usage:

UpdateRegistry -GetKeys [<pattern>] -ProductName <name> [-Case]

<pattern>

Optional. If no pattern is specified, then all allowed key names are reported. If a pattern is specified, then only keys that match the pattern are shown. For example: **Settings/Project**. If the name contains spaces, then it must be quoted. By default, the pattern match is case insensitive. If the -Case command line option is specified, then the pattern match is case sensitive.

<name>

Required. The application or product name and version, as described above. For example, **ElectronicsDesktop2018.1**. If the name contains spaces, it must be enclosed within quotation marks.

UpdateRegistry -Set Command

This command is used to add or modify an option setting in an option config file. If the option config file does not exist, it will be created. If the setting does not exist in the specified config file, it will be added. If the setting already exists in the specified config file, then the value will be changed to the specified value.

Usage:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

UpdateRegistry -Set -ProductName <name>

-RegistryKey <keyPath>

-RegistryValue <value> [-RegistryLevel <level>

<name>

Required. The application or product name and version, as described above. For example, **ElectronicsDesktop2018.1**. If the name contains spaces, it must be enclosed within quotation marks.

<keyPath>

Required. The pathname of the option setting. Example:

`Desktop/Settings/ProjectOptions/AnimationMemory.`

<value>

Required. The new value of the option, typically a string or a number. If the value contains spaces, it must be quoted.

<level>

Optional. A string denoting which config file to modify. One of: **install**, **install_machine, user** , and **user_machine**. If the level is not specified, then the **user_machine** (host-dependent user options) file is modified.

UpdateRegistry -Delete Command

This command is used to remove an option setting from an option config file. If the setting does not exist in the specified config file, the file will not be changed. If the setting exists in the specified config file, then it will be removed. A setting may need to be removed from an option config file to allow the setting from a lower priority file to be used by the application.

Usage:

```
UpdateRegistry -Delete -ProductName <name>
-RegistryKey <keyPath>
[ -RegistryLevel <level> ]
```

<name>

Required. The application or product name and version, as described above. For example, **ElectronicsDesktop2018.1.** If the name contains spaces, it must be enclosed within quotation marks.

<keyPath>

Required. The pathname of the option setting. Example:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
Desktop/Settings/ProjectOptions/AnimationMemory.
```

<level>

Optional. A string denoting which option config file to modify. One of: **install**, **install_machine**, **user** , and **user_machine**. If the level is not specified, then the **user_machine** (host-dependent user options) file is modified.

UpdateRegistry -FromFile Command

You can use this form of the UpdateRegistry command to set multiple key-value pairs from a file with a single UpdateRegistry command. You specify the -FromFile command line option. This option must be followed by a file name. The file may contain multiple entries, where each entry contains a registry key and a registry value. The key-value pairs are added to the registry level specified by the -RegistryLevel command line option; if no -RegistryLevel is specified, then the default registry level (user_machine) is used.

UpdateRegistry File Format

The file format is similar to the -batchoptions file format. An example **UpdateRegistry** file is shown below:

```
$begin 'AddEntries'
  'TempDirectory'='C:/temp/AnsysEM'
  'Desktop/Settings/ProjectOptions/HPCLicenseType'='Pool'
  'Hfss/UseLegacyMultiprocessingLicense'=1
$end 'AddEntries'
```

Additional notes on the file format:

- The file may contain an arbitrary number of entries, one per line.
- Leading whitespace on each line is ignored. Spaces or tabs may be used to make the file more readable.

Registry key path name:

- The registry key pathname appears before the **=** sign on each line.
- Each registry key pathname must be enclosed in single quotes.

Registry value:

- The registry value appears after the **=** sign on each line.
- Integral registry values must not be enclosed in quotes.
- All other registry values are treated as strings, and must be enclosed in single quotes.
- The forward slash "/" and back slash "\" are directory separators on Windows.
- The back slash "\" is used as an escape character in the value string. That is, this character removes the special meaning of the following character.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- The single quote character normally ends the value string. The back slash may be used to remove this special meaning, and include a single quote in the string.
- To use a back slash as a directory separator on Windows, it must be escaped. That is, a double back slash "\\" is used to denote a single directory separator.

Alternative UpdateRegistry File Format

- Analysis Configuration File format, which is exported from the **HPC and Analysis Options** dialog box.

Setting Options via Configuration Files

Example Uses for Export Options Features

User Options and the Update Registry Tool

Batchoptions Command Line Examples

Example Uses for Export Options Features

The **Tools** > **Options** > **Export Options** feature is intended to make it easier for different users to use Ansys Electromagnetics tools installed on shared directories or network drives. This section outlines some use cases enabled by this feature.

> **Note:**
>
> Functionality featured in the examples in this topic apply to multiple design types.

Options that Apply to All Users

In many cases, an Ansys Electromagnetics tool installation is administered and maintained by a single user or group and used by a number of other users or groups. The permissions of the Ansys Electromagnetics tool installation may be set so that the administrator may add, delete or modify files, but other users may only read or execute these files. The administrator may set the recommended option settings in the installation default config file and/or the host dependent default options config file. These config files reside within the installation directory hierarchy, and should generally have the same permissions as other Ansys Electromagnetics tool installation files; that is, administrators can control these settings, but other users cannot add, remove, or change them.

You can override any of these settings. This may be done using the Desktop UI, which affects the host-dependent user options config file. It may also be done using the host-independent user options config file. If user has overridden an option setting in either of the user files, the user may revert back to the option settings provided by the administrator by removing the setting of the same option in the host-dependent user option config file and/or the host-independent user option config file.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For global defaults, the administrator may set a value in the installation default config file. These settings will to apply to all users on all hosts.

In some cases, there are significant differences between the capabilities of different hosts. The host-dependent default config file may be used to specify different default values on some hosts. Any setting in a host dependent default config file would affect all users running on the specified host. The installation default value is used if there is no value specified for the setting in the host-dependent default config file for the current host. Note that the host-dependent default config file is named <*hostname*>.xml, where *hostname* is the name of the host computer.

Example: Searching for a Registry Key Pathname

Both administrators and ordinary users may occasionally use the **UpdateRegistry** command line tool to add, change or delete settings. To use this tool, the registry key pathname must be known by the user. The -GetKeys option may be used to quickly search for a key pathname if some information is known about it. For example, if the administrator knows that there is a setting related to issuing warning messages when available disk space is low, but does not know the exact key name, the following command may list some of the keys related to disk space:

```
UpdateRegistry -GetKeys disk -ProductName ElectronicsDesktop2024.2
```

This will display a list of all keys that match the string "disk" case insensitively.

Typical output may look like this:

```
Registry keys matching pattern <disk> case insensitively:
Desktop/Settings/ProjectOptions/DiskLimitForAbort: value is <0> at
level <user_machine>
```

Example: Setting an Installation Default Value

The normal default for the Options/General/Desktop Performance/Warn when available disk space is less than setting is 0 MB. If the administrator is concerned that running out of disk space might be a common problem, the administrator could set the installation default for the warn setting setting to 1000 MB, for example. This limit would then apply to all users running on all hosts. The administrator could use the following command to change this setting for Ansys Electronics Desktop:

```
UpdateRegistry -Set -ProductName ANSYSElectronicsDesktop2024.2
-RegistryKey Desktop/Settings/ProjectOptions/DiskLimitForAbort
-RegistryValue 1000
 -RegistryLevel install
```

Example: Setting a Host-Dependent Default Value

For this example, assume that all hosts have two cores, except for three hosts: bighost1, bighost2, and bighost3, which have eight cores each. The administrator has set the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Desktop/Settings/ProjectOptions/NumberOfProcessors option value to 2 in the installation default config file. The administrator may set the Desktop/Settings/ProjectOptions/NumberOfProcessors option value to 8 in the host-dependent default config files for the three hosts having 8 cores: bighost1, bighost2 and bighost3. The administrator may log in to host bighost1 and run the following command to change this setting for the host-dependent default options config file for host bighost1:

```
UpdateRegistry -set -ProductName ElectronicsDesktop2024.2
-RegistryKey Desktop/Settings/ProjectOptions/NumberOfProcessors
-RegistryValue 8
-RegistryLevel install_machine
```

To make this change for the other two hosts, the administrator would log in to bighost2 and bighost3, and run the same command on each of those hosts.

Example: Reverting from a User-Defined Option Value to the Administrator Default

Consider the case in which Electronics Desktop was installed and the administrator initially did not set a value for the Desktop/Settings/ProjectOptions/DiskLimitForAbort setting in the default installation config file. User jsmith (who always uses host jshost) wanted to be warned before disk space dropped to zero, so he set the Desktop/Settings/ProjectOptions/DiskLimitForAbort to 100 MB using the UI. This setting is recorded in the host dependent user options config file for host jshost and user jsmith. Now the administrator learns that many users are running into disk space issues, so that administrator sets the installation default value for the setting Desktop/Settings/ProjectOptions/DiskLimitForAbort to 1000 MB, as in the above example.

When user jsmith runs Electronics Desktop on host jshost, the disk limit is 100 MB, not 1000 MB, because the host-dependent user options config file overrides all of the other config files. User jsmith may revert to the administrator provided default by removing this setting from the host dependent user options config file for host jshost and user jsmith. The following command may be run by user jsmith on host jshost to remove this setting:

```
UpdateRegistry -Delete -ProductName ElectronicsDesktop2024.2
-RegistryKey Desktop/Settings/ProjectOptions/DiskLimitForAbort
-RegistryLevel user_machine
```

If user jsmith had added a value for this setting to the host-independent user options config file, then user jsmith would also run the following command to remove this setting from the host-independent user options config file:

```
UpdateRegistry -Delete -ProductName ElectronicsDesktop2024.2
-RegistryKey Desktop/Settings/ProjectOptions/DiskLimitForAbort
-RegistryLevel user
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

User Options and the Update Registry Tool

When you change an options value using the Desktop UI, the new value is stored in the host-dependent user options config file. You can also use the UpdateRegistry tool to add or modify settings in the host-dependent user options config file. You cannot use the Desktop UI to remove settings from the host-dependent user options config file, however. You must use the UpdateRegistry tool to remove settings from the host-dependent user options config file.

If you have not explicitly created a host-dependent or host-independent user options config file, then when you first run an Ansys tool on a host, all settings will come from the host-dependent default options configuration file or the installation default options configuration file. Any settings for another host in a host-dependent user options configuration file will not be carried over to the new host. This may be inconvenient if you have preferred option settings that differ from the settings that apply to all users, especially if you run the Ansys tool on a number of different hosts. In this case, you may set these option values in your host-independent user options config file. Then, these option values will be used on all new hosts, overriding any values set by the administrator to apply to all users. Any changes made in the UI will only affect your host-dependent user options cfile for the current host.

Getting a Value from a Specific Configuration File

Getting a Value Using Precedence Rules

Example of Removing a Host Dependent User Option Setting

Example Adding a Host Independent User Option Setting

Setting the Temporary Directory

Temporary Directory Configuration File Format

Setting the Temporary Directory Using the GUI

Setting or Removing Temporary Directory Values in Configuration Files: UpdateRegistry Command

Getting a Value from a Specific Configuration File

In the previous example, user **jsmith** may decide to check the **Desktop/Settings/ProjectOptions/DiskLimitForAbort** setting in the host-independent user configuration file before making any changes. Use this command to quickly view this setting for HFSS 14.0 before making the change:

```
UpdateRegistry -Get -ProductName HFSS15.0 -RegistryKey
Desktop/Settings/ProjectOptions/DiskLimitForAbort -RegistryLevel user
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Getting a Value Using Precedence Rules

In many cases, you may be more interested in the value of a setting that will be applicable when running the product than in the setting in a single configuration file. If you use the **-Get** option with no **-RegistryLevel** specified, then the value reported is the value found in the highest precedence configuration file. If user **jsmith** is interested in the highest precedence value for the **Desktop/Settings/ProjectOptions/DiskLimitForAbort** setting, use this command to report this information:

```
UpdateRegistry -Get -ProductName ElectronicsDesktop2024.2 -
RegistryKey Desktop/Settings/ProjectOptions/DiskLimitForAbort
```

Example of Removing a Host Dependent User Option Setting

For this example, user **jsmith** always uses host **jshost** to run Twin Builder. At some point, **jsmith** set the Autosave interval in the **General Options** dialog box, **Project Options** tab to 1000 edits, and this value was written to the **jsmith** host-dependent user options config file for host **jshost**. Now, **jsmith** wants to remove this setting and return to the default value of 10. **jsmith** may run this command on host **jshost** to remove the **Desktop/Settings/ProjectOptions/AutoSaveInterval** option value from this configuration file:

```
UpdateRegistry -Delete -ProductName Simplorer2024.1
-RegistryKey Desktop/Settings/ProjectOptions/AutoSaveInterval
-RegistryLevel user_machine
```

Related Topics

User Options and the Update Registry Tool

Example Adding a Host Independent User Option Setting

Setting the Temporary Directory

Temporary Directory Configuration File Format

Setting the Temporary Directory Using the GUI

Setting or Removing Temporary Directory Values in Configuration Files: UpdateRegistry Command

Example Adding a Host-Independent User Option Setting

Consider the case in which there is no value set for the **Desktop/Settings/ProjectOptions/DiskLimitForAbort** setting for all Twin Builder users. The default is then 0 MB. User **jsmith** uses a variety of hosts and wants to be warned whenever disk

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

space drops to 250 MB on any host. User **jsmith** may use the following command to set the **Desktop/Settings/ProjectOptions/DiskLimitForAbort** option value to 250 MB for all hosts:

```
UpdateRegistry -Set -ProductName Simplorer2024.1
-RegistryKey Desktop/Settings/ProjectOptions/DiskLimitForAbort
-RegistryValue 250 -RegistryLevel user
```

Related Topics

User Options and the Update Registry Tool

Setting the Temporary Directory

Temporary Directory Configuration File Format

Setting the Temporary Directory Using the GUI

Setting or Removing Temporary Directory Values in Configuration Files: UpdateRegistry Command

Setting the Temporary Directory

The temporary directory may be viewed or set using the Electronics Desktop user interface (UI), or from the command line.

To set the directory via the UI, select **Tools** > **Options** > **General Options**. In the tree at the left side of the dialog box, expand the **General** branch and select **Directories**. Then, use the **Temp** field and **Override** check box to enter a desired directory path. Values set in this manner are written to the user_machine level configuration file for the temporary directory. If the **Override** check box is cleared, click **OK** to change the user_machine level setting for the temporary directory to an empty string. This enables settings from the next highest precedence configuration file. The file that provides the currently active temporary directory setting is shown under the **Temp** edit box.

To set the temporary directory from the command line, using the **-batchoptions** command line option. See ***Running Twin Builder from a Command Line*** and ***-Batchoptions Command Line Examples***.

The temporary directory may be configured with an installation default value, as well as a host-dependent default value, a host-independent user-specified value and a host-dependent user-specified value. Temporary directory settings are stored in different files from the other option settings. These files are located in the same directories as the configuration files for the other option settings. The following table shows the directories and files used to store temporary directory settings.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Config File	Level Name	File Name	Windows Directory Path	Linux Directory Path
Host-dependent, user-specific temporary directory	user_machine	<*hostname*>.cfg	%UserProfile%\Documents\Ansoft \<*ApplicationName&Version*>\config	$HOME/Ansoft /<*ApplicationName&Version*>/config
Host-independent, user-specific temporary directory	user	default.cfg		
Host-dependent, default temporary directory	install_machine	<*hostname*>.cfg	<*InstallationDirectory*>\v<*version*> \Win64\config	<*InstallationDirectory*>/v<*version*> /Linux64/config
Installation default temporary directory	install	default.cfg		

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

- **<*hostname*>** is the name of the computer on which the Electronics Desktop software is installed
- **$HOME** is the user's home directory on Linux
- **<*ApplicationName&Version*>** is the product name (without spaces) followed by the four-digit year of the version, a decimal point, and the minor release number (such as ElectronicsDesktop2024.2)
- **%UserProfile%** is a Windows variable that represents the currently active user's profile (for example, C:\Users\JohnDoe)
- **<*InstallationDirectory*>** is the root folder where the Electronics Desktop software is installed (typically, C:\Program Files\AnsysEM, on Windows, or /opt/AnsysEM, on Linux)
- **<*Version*>** is the last two digits of the product version's year followed by the minor release number, without a decimal point (such as 221)

As with other options, the settings in these files have precedence in the following sequence: user_machine (highest precedence), user, install_machine, install (lowest precedence). The installer creates the file at the "install" level (default.cfg). The first time you start and exit the application, the file at the "user_machine" level is created (<*hostname*>.cfg). The other files are only created if you use the UpdateRegistry tool to specify an option at the "user" or "install_machine" level. If the temporary directory is set to an empty string in a configuration file, then that setting is ignored.

Related Topics

User Options and the Update Registry Tool

Example Adding a Host Independent User Option Setting

Temporary Directory Configuration File Format

Setting the Temporary Directory Using the GUI

Setting or Removing Temporary Directory Values in Configuration Files: UpdateRegistry Command

Temporary Directory Configuration File Format

This section describes the format of the temporary directory configuration files. The format is the same for files at all four levels: **user_machine**, **user**, **install_machine**, and **install**. These files are text files; you can use any text editor to modify or create temporary directory configuration files.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

An example temporary directory configuration file is shown below:

```
$begin 'Config'
tempdirectory='C:/TEMP/AnsysEM'
$end 'Config'
```

The temporary directory specified by this configuration file is **C:/TEMP/AnsysEM**.

Additional notes:

- The string containing the path name of the temporary directory must be enclosed in single quotes.
- The forward slash **/** and back slash **** are directory separators on Windows.
- The back slash **** is an escape character in the **tempdirectory** string. That is, this character removes the special meaning of the following character.
- The single quote character normally ends the **tempdirectory** string. The back slash may be used to remove this special meaning, and include a single quote in the string.
- To use a back slash as a directory separator on Windows, it must be escaped. That is, a double back slash **** denotes a single directory separator.
- On Windows, a UNC path normally begins with two back slash characters. In a **tempdirectory** string, each of these back slash characters must be doubled, so four consecutive back slashes **** are used in the config file.

UNC Example

Config file:

```
$begin 'Config'
tempdirectory='\\\\hostxyz\\TEMP\\abc'
$end 'Config'
```

Here **hostxyz** is a host with a sharename **TEMP** having subdirectory **abc** used as the temporary directory. This shows that four back slashes are required for UNC names and that back slashes used as directory separators must be doubled.

Single Quote Example

Config file:

```
$begin 'Config'
tempdirectory='C:/TEMP/ab\'cd'
$end 'Config'
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Temporary directory is **C:/TEMP/ab'cd**. This shows how to include a single quote in a temp directory path name. It also shows that forward slashes may be used as directory separators on Windows.

Related Topics

User Options and the Update Registry Tool

Example Adding a Host Independent User Option Setting

Example for Setting the Temporary Directory

Setting the Temporary Directory Using the GUI

Setting the Temporary Directory From the Command Line

Setting the Temporary Directory using the GUI

The temporary directory may be viewed or set using the Desktop GUI. Select **Tools** > **Options** > **General Options** > **General** > **Directories**. Select the **Override** check box to enter a directory path name or to click to bring up a directory file browser dialog box, from which you can select a temp directory. Values set in this manner are written to the **user_machine** level configuration file for the temporary directory. If the **Override** check box is cleared, the **user_machine** level setting for the temporary directory is changed to an empty string when you click **OK**. This enables settings from the next highest precedence config file. The config file which provides the currently active temporary directory setting is shown under the **Temp** field if the **Override** check box is cleared.

Setting the Temporary Directory From the Command Line

The temporary directory may be set from the command line using the **-batchoptions** command line option. See ***Running Twin Builder from a Command Line***. ***Batchoptions Command Line Examples*** includes examples that show how to set the temporary directory from the command line.

Related Topics

User Options and the Update Registry Tool

Example Adding a Host Independent User Option Setting

Example for Setting the Temporary Directory

Temporary Directory Configuration File Format

Batchoptions Command Line Examples

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Running Twin Builder from a Command Line

Setting or Removing Temporary Directory Values in Configuration Files: UpdateRegistry Command

Use the **UpdateRegistry** command line tool to view, add, change, or remove the temporary directory setting from any of the temporary directory config files. The registry key for viewing or modifying the temporary directory is **TempDirectory**. The **-Get**, **-Set**, and **-Delete** options are valid for viewing, adding, changing, and deleting a temporary directory setting. The **-GetKeys** option does not list the temporary directory key.

Related Topics

User Options and the Update Registry Tool

Batchoptions Command Line Examples

Use the **-batchoptions** entries command line argument to specify one or more batchoptions settings on the command line. To specify multiple entries using a single **-batchoptions** argument, the entries should be enclosed in double quotes. Alternatively, the batchoptions may be specified in a file using the **-batchoptions** <*file name*> command line argument format. In this case, the file name is an absolute or relative pathname of the file containing the batchoptions, as described above. The two approaches may not be combined: either all batchoptions must be in a file or all batchoptions must be specified explicitly on the command line.

Batchoptions File Format

Example -BatchOptions with -Remote

Example -Batchsolve for Local

Related Topics

Running Twin Builder From a Command Line

Batchoptions File Format

> **Note:**
>
> Functionality featured in the example in this section applies to multiple design types.

An example batchoptions file is shown below:

```
$begin 'Config'
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    'Desktop/ProjectDirectory'='C:/test/projects'
    'Desktop/Settings/ProjectOptions/NumberofProcessors'=2
$end 'Config'
```

Additional notes on the file format:

- The file may contain an arbitrary number of batchoption entries, one per line.
- Leading whitespace on each line is ignored. Spaces or tabs may be used to make the file more readable.
- The **Registry Key** appears before the equals sign (=) on each line and must be enclosed in single quotes ('). The registry key includes the key path and the name of the registry value.
- The registry value (or option value) appears after the equals sign on each line.
- Registry keys are case-insensitive.
- There are two supported types of registry values—string and integer:
  - Each **string** value must be enclosed in single quotes (').
  - Do **not** enclose **interger** values in quotes.
- For file paths within string values, the forward slash (/) may be used as a directory separator on both Windows and Linux systems.
- Alternatively, on Windows only, the customary backslash (\ ) may be used as a directory separator. However, in string values, the backslash is used as an escape code for indicating special characters that cannot be typed directly (such as \n for a new line). Therfore, if you use the backslash as a directory path separator, each instance must be doubled (\\ ). An example is: '\\\\host3\\temp\\Ansoft'. In this case, each double backslash is interpreted as a single backslash (\\host3\temp\Ansoft).
- The single quote character (') normally ends a string value. If you need to include a single quote (or apostrophe) within a string, use the backslash-apostrophe (**\'**) escape sequence. For example, the string '%UserProfile%/Documents/Ansoft/John\'s_Files', is interpreted as: %UserProfile%/Documents/Ansoft/John's_Files.

Example -BatchOptions with -Remote

In this example, we run a batch Twin Builder analysis of project file **project1.aedt** which contains a Twin Builder design. We want all temporary files and directories created in directory **C:\temp\batchsolve** instead of using the installation default for the temporary directory. We decide to run the analysis on a remote host, at IP address 12.34.56.78. Because of limited memory on the remote host, we decide to run the analysis using only a single COM engine. Because the remote host has four cores, we decide to use four threads for multiprocessing, for both distributed and non-distributed parts of the analysis. We can use the **-Remote** option to specify that there will be a single remote COM engine.

Here is a sample command line for this analysis, where the project file **\\somehost\projects\project1.aedt** is located in a shared directory specified using a UNC path:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
ansysedt.exe -BatchSolve -Remote -Machinelist list=12.34.56.78
-batchoptions "TempDirectory='C:/temp/batchsolve'
\\somehost\projects\project1.aedt
```

An alternative is to use the **-Distributed** command line option. Because the **-Machinelist** list contains only one host, there is a single remote COM engine in this case, also.

```
ansysedt.exe -BatchSolve -Distributed -Machinelist list=12.34.56.78 -
batchoptions "TempDirectory='C:\\temp\\batchsolve'
\\somehost\projects\project1.aedt
```

The above command lines show that **/** is a directory separator in Windows. The **** is also a directory separator in Windows, but it must be doubled to **** because **** is an escape character.

Related Topics

Batchoptions Command Line Examples

Example -Batchsolve for Local

Example -Batchsolve for Local

In this example, we run a batch analysis of project file **testproject.aedt** on the local host. We want all temporary files and directories created in the **C:\temp\batchsolve** directory instead of using the installation default for the temporary directory. Because the local host has four cores, we decide to use four threads for multiprocessing, for both distributed and non-distributed parts of the analysis.

Here is a sample command line for this analysis, where the project file **\\host123\projects\testproject.aedt** is located in a shared directory specified using a UNC path:

```
ansysedt.exe -BatchSolve -Local -batchoptions
"TempDirectory='C:/temp/batchsolve'
\\host123\projects\testproject.aedt
```

Note that batchoptions that contain embedded spaces must be in single quotes.

Related Topics

Batchoptions Command Line Examples

Example -BatchOptions with -Remote

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Batchoptions and Analysis Configurations in the Registry

Use analysis configurations to specify machines and options for local, remote, and distributed analysis, including capabilities enabled by HPC licenses.

How Analysis Configurations are Stored in the Registry

A configuration contains information beyond the machine or machines to use for a solution. Examples are the number of processors to allocate to the analysis for each machine in the list, memory limits, directory locations for personal libraries and temporary files, and many other preferences.

Note:

- Options are arranged as keys and values (in a structure similar to the *Windows Registry*). However, these options are not a part of the *Windows Registry* but are separately stored and maintained by Ansys Electronics Desktop. For more information concerning the configuration files comprising the options registry, see the following topics:

  Setting Options Via Configuration Files
  Setting the Temporary Directory

- For access to options and functionality beyond what is directly accessible via the user interface or batch options, refer to the documentation of the **UpdateRegistry** tool. This tool is discussed in the following help topic and in the topics that follow it in the same branch of the product help:

  Setting or Removing Option Values in Configuration Files: UpdateRegistry Command

General settings are associated with the Desktop application or 3D Editor. **HPCLicenseType** and **tempdirectory** values are at the root level of the registry. Other options are specific to a particular product or design type. For example, certain mesh, boundary, and memory limit settings as well as many other preferences are product-specific and are therefore associated only with the applicable design types. Such options appear with a consistent value name but in a different registry path for each applicable design type.

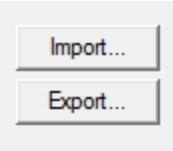
Copying a Configuration from one Design Type or Product to Another

To copy a configuration from one design type (or product) to another:

1. Click **Tools** > **Options** > **HPC and Analysis Options**. The **HPC and Analysis Options** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. On the **Configurations** tab, click **Export…** to export the configuration to a file.



3. Switch to the destination design type (or product) and click **Import…** to import the configuration data.

   The system ignores any data that is not applicable to the destination design type, and assigns default values to any settings present in the destination design type that were not present in the source configuration. You may then edit the copy.

Using HPC and Analysis Options for Configurations

Due to the complexity of the registry values for the configurations we do not recommend directly editing these values using the *UpdateRegistry* tool. Instead, use the **HPC and Analysis Options** dialog box to edit or create a configuration. (See ***Setting HPC and Analysis Options***.) Configurations created or edited using the GUI are stored in the **user_machine** level of the registry. A configuration may be created for one of the other registry levels using the following steps.

1. Create the configuration using the Analysis Configurations GUI and export the configuration to a file.
2. Delete the configuration using the GUI so that it will not be present in the **user_machine** level. Then, exit the GUI.
3. Use the **UpdateRegistry** tool to import the data into the desired registry level using the **-FromFile** option to specify the file exported via the GUI, and using the **-RegistryLevel** option to specify the registry level where the configuration is to be stored. For example, an administrator may use this approach to create a configuration at the **install** level that may be used by any user on any machine.

Batch Options

There is a large number of both general and product-specific options supported by the software. These options have evolved over time. Therefore, older batchoptions files may no longer be valid, and the options listed in the user interface of the current software may differ from earlier versions you have used. Additionally, there are options beyond those listed within the user interface (that is, the UI provides a subset of the available options). You can generate a more complete list of options by running the UpdateRegistry tool with the -GetKeys switch and piping the output to a text file, as detailed in the following Windows procedure:

1. Navigate to the following folder:

   <*installation_directory***>\v<***version***>\Win64**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Type and enter the following command:

   UpdateRegistry -GetKeys -ProductName ElectronicsDesktop20*xx.y* **>** <*text_file_path*>**\Batchoptions.txt**

   Substitute the last two digits of the installed product version year and the minor release number for *xx.y* (such as **22.1**). Also, substitute the desired *text_file_path* (such as **%UserProfile%\Desktop**).

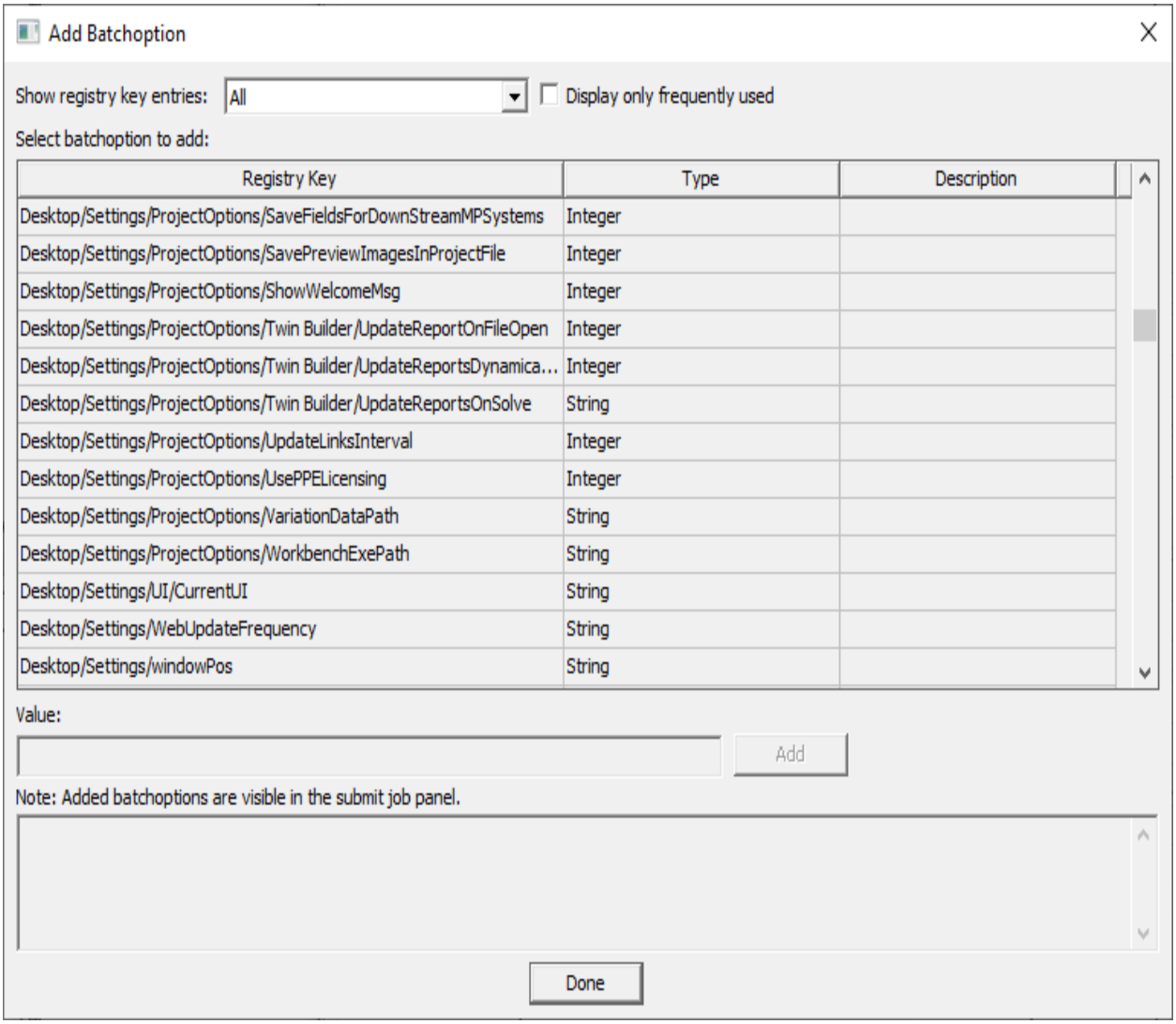
   The resulting **Batchoptions.txt** file will contain a nearly complete list of available options.

This procedure also works from the <*installation_directory*>/v<*version*>/Linux64 folder on the Linux platform.

> **Note:**
>
> For special product-specific options that are neither available from the GUI nor listed by the UpdateRegistry tool, see Special Batch Options. Even though the -GetKeys switch does not list these special options, you can still use the UpdateRegistry tool to set them.

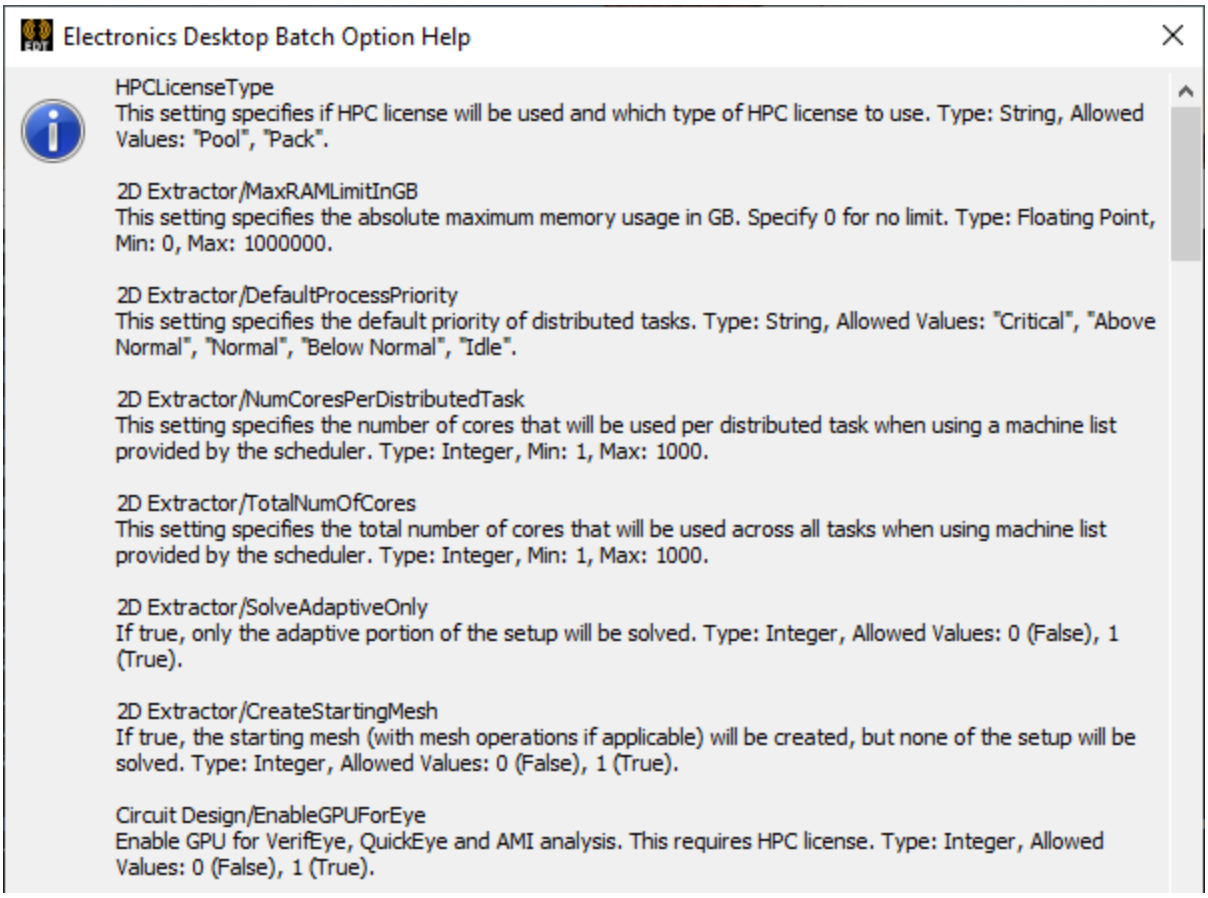
When you submit jobs to a remote computer or cluster, you can specify batch options using the job submission GUI. When using the GUI, you select the batch options from a list and therefore avoid typographical errors. For the most commonly used batch options, there is detailed information about the allowed values. Click **Submit** on the **Simulation** ribbon tab to access the *Submit Job To* dialog box. Then, click **Add** to access the **Add Batchoption** dialog box pictured below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



To assist users who need to specify batch options and are unable to use the job submission GUI, a new help option has been added. If the Electronics Desktop application is launched with the **-batchoptionhelp** command line argument, a message box is displayed which lists and describes the most common design type-specific batch options:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Setting Analysis Configurations Using the User Interface

The Desktop User Interface may be used to select the analysis configuration to be used for each design type. Each analysis configuration is identified by a unique name. View these settings with the **HPC and Analysis Options** dialog box.

See Setting HPC and Analysis Options.

Twin Builder File Types

Some common Twin Builder file and folder types are listed below:

project_name.**aedt**	Twin Builder project file. When you create a Twin Builder project, it is given an **.aedt** file extension and stored in the directory you specify. Files and folders related to that project are also stored in that directory.
project_name.	Twin Builder folder containing results data for a project.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

aedtresults	
design_name	Twin Builder folder containing design and results data for a design. These folders reside in the *project_name*.**aedtresults** folder.
design_name.**asol**	The **.asol** file contains the database of all solved variations as well as where the resulting data is stored in the corresponding *design_name* folder. These files are stored in the *project_name*.**aedtresults** folder.
.ssh	Legacy Simplorer schematic file. Refer to Translating Legacy Simplorer Projects and Schematics for details.
.ssc	Legacy Simplorer project file. Refer to Translating Legacy Simplorer Projects and Schematics for details.
.smd	Legacy Simplorer library file.
.anf	Ansoft neutral file. Refer to *Importing ANF Design Data* for details.

> **Warning:**
>
> Observe that the operating system limits file names to a maximum of 255 characters — including path name and drive designator.

Setting Up a Twin Builder Design

To set up a Twin Builder design, follow this procedure.

1. Insert a Twin Builder design into a project.

   After you insert a design, you do not need to perform the remaining steps sequentially. However, steps 2 through 8 must be completed before a solution (analysis) can be generated and the results displayed in reports.

2. Select and place components on schematic.
3. Connect components.
4. Define component parameters and properties.
5. Define the simulation outputs.
6. Set up a solution.
7. Set solution options.
8. Create reports.
9. Run the analysis.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Creating Projects

To create a new project:

1. Click **File** > **New**. A new project appears in the Project tree. It is named **Project***n* by default, where *n* is the order in which the project was added to the current session.

> **Note:**
>
> - You can specify a name for the project when you save it.
> - The operating system limits file names to a maximum of 255 characters (including path name and drive designator).

   Project definitions, such as components and models used in the project designs, are stored in a **Definitions** folder under the project name in the Project tree.

2. By default, Twin Builder inserts a design named **TwinBuilder1** under the project.

   To insert a design manually, do one of the following:

   - Click **Project** > **Insert Twin Builder Design**.
   - In the Project tree, right-click the icon for the current project, then click **Insert** > **Insert Twin Builder Design**.

   The **Schematic Editor** dialog box appears to the right of the **Project Manager**. You can now create the schematic.

> **Note:**
>
> Click the plus sign to the left of the design icon in the Project tree to expand the Project tree and view specific data about the design.

Related Topics

The Schematic Editor

Setting the Project Tree to Expand Automatically

Opening Existing Projects

Follow this procedure to open a previously saved project.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **File** > **Open**.
2. Use the file browser to find the desired Twin Builder **.aedt** project file.

   > **Note:**
   >
   > By default, only **.aedt** files are displayed in the file browser.

3. Select the file you want to open.

   The file browser includes a preview area containing an **Image** tab that displays a representation of the design selected in the drop-down **Designs** menu. You can preview any notes saved for the selected design in the **Notes** tab. The file **Type**, and whether the selected Design has been **Solved** also display.

4. Click **OK**.

   The project information appears in the Project tree.

   > **Note:**
   >
   > If you open another project without editing the first project, Twin Builder removes that project.

   If newer definitions are available for any of the components in the project, informational messages recommending that you update definitions display in the **Message Manager** pane.

   To open saved project:

   - Drag a Twin Builder project file icon to the Twin Builder icon.
   - Drag a Twin Builder project file icon to the Twin Builder desktop.
   - Double-click a Twin Builder project file icon.

Related Topics

Opening Recent Projects

Opening Recent Projects

To open a recently saved project in Twin Builder:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Click **File**, then select from the list at the bottom of the menu.

> **Note:**
>
> If you open another project without editing the first project, Twin Builder removes that project.

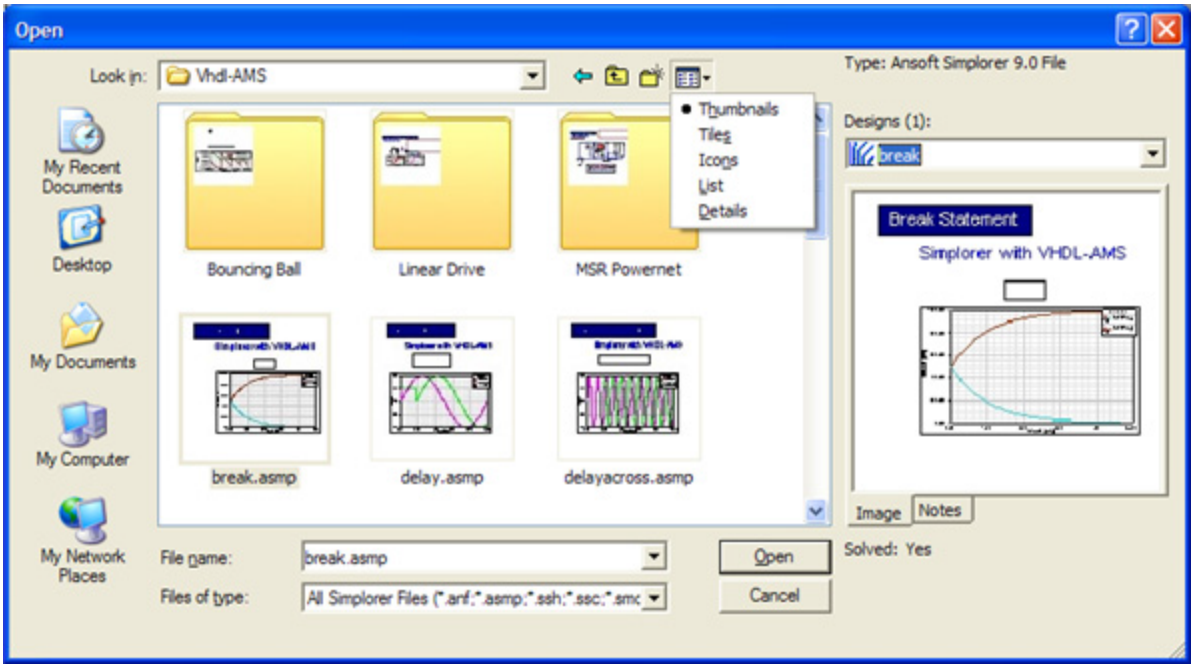
Opening Example Projects

Select **File** > **Open Examples** to access and open example projects included with the Twin Builder product installation. This displays a file browser opened to the **Examples** folder in the product installation folder.

Use the file browser to find the desired Twin Builder **.aedt** project file. The file browser includes a preview area containing an **Image** tab and a **Notes** tab.

- The **Image** tab displays a representation of the design selected in the dropdown **Designs** menu.
- The **Notes** tab previews any notes saved for the selected design.

The file **Type**, and whether the selected design has been solved are also displayed.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Translating Legacy Simplorer Projects and Schematics

Twin Builder does not support Simplorer 7.0 schematics. If you want to port a design from Simplorer 7.0, you must use Simplorer Release 16.2 or earlier to translate it, then bring it into the current release.

When Component Terminals Do Not Match the Model Terminals

In some instances, legacy schematics may contain models (such as transistors with substrate pins) that have a different number of terminals than the associated component or symbol. Several components in the legacy **Compatibility 4x7** library fall into this category. During translation, such components display an alert in the **Message Manager** pane similar to the following:

```
Component translation: Terminals (BULK) of component NJFET5 don't
match model - please check , (Simplorer 7:
328:COMPATIBILITY4X7:NJFET5). (9:15:59 AM Nov 07, 2007)
```

In the example above, the terminal labelled **BULK** for the **NJFET5** component does not match the referenced model in the **COMPATIBILITY4X7** library. Before you can successfully simulate, you must edit the component definition to correct the terminal mismatch. Follow this proecdure:

1. Translate legacy libraries used by components in the legacy schematic. In the example message above, the **COMPATIBILITY4X7** library needs to be translated.
2. In the **Project Manager Definitions** > **Components** folder, right-click the component you want to edit (**NJFET5** in the example), and select **Edit Component** from the shortcut menu. The **Edit Component** dialog box appears.
3. On the **General** tab, click **Properties** to open the component **Properties** dialog box.
4. On the **Parameter Defaults** tab, click the button in the **Value** column for the **SimplorerNetlist** property to open the **Edit Netlist String** dialog box.
5. Copy the entire netlist string, then close the **Edit Netlist String** and the **Properties** dialog boxes.
6. On the **Simulation Models** tab of the **Edit Component** dialog box, remove the existing model from the list.
7. Click **Add Netlist Line** and paste the netlist string in the **Netlist Line** dialog box.
8. In the netlist string, locate the string: **BULK:= %3** and change **%3** to **GND** and click **OK** to close the dialog box.
9. Click **OK** on the **Edit Component** dialog box to accept the changes.
10. Repeat the above procedure as needed for any similar occurrences of terminal mismatch.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Importing ANF Design Data

Ansoft Neutral File (ANF) formatted files typically are generated by third party tools or by the Ansys “AnsoftLinks” program that translates third-party designs. ANF is a public, neutral file format that allows third party tools to exchange design data with Ansys products. An ANF file can contain schematic data, and component data. A given ANF file may contain just schematic data with no component data, or may contain just component data with no schematic data.

> **Note:**
>
> If material properties are not provided in the ANF file, the required material properties are taken from the Twin Builder material library database. These properties may be different from those in the tool used to generate the ANF file.

When geometry is imported from ANF, ports are added, and a subdirectory for the ANF-based project is configured in the Project directory. When ANF-based projects are saved, the **Save As** dialog box always opens, regardless of where the ANF file originated.

1. To open an ANF-based project, click **File** > **Open** on the top menu bar. The **Open** dialog box appears.
2. Browse to the directory containing the file with the project you want to open. Use the **Files of type** field to display the files with the ANF format.
3. Click **Open**. Twin Builder evaluates the information in the ANF file and opens a window for you to select the Twin Builder product for importing the ANF design:
4. Select **Twin Builder** and click **OK**. The **ANF Conversions** dialog box appears.
5. The **Current ANF (Components) Map** panel lists the components from the ANF file and any associated models. You can enter the names of a model manually in the **Model Name** field. The ModelName property for the component in Twin Builder will be set to the entry in the **Model Name** field in the mapping, and the Netlist property will begin with that name instead of the component name.
6. **Update External Component Map** saves any mappings you enter manually to a text file. Click **Save Map File** to open a File **Open** window. Use the window to browse to the directory where the map file is to reside, then enter the name of the file. The map file is saved with a **.namemap** extension. The same mapping file should be used for both component and property mappings. After the mapping file has been created, **Update External Component Map** saves any mappings you enter manually.
7. The **Current ANF (Property) Map** panel lists the mapping of properties in the ANF file to the properties used when the file is imported. Enter the mapped name of a property manually in the **Model Name** field. The name for the property in Twin Builder will be set to the entry in the netlist.
8. **Update External Property Map** saves any mappings you enter manually to a text file. Click **Save Map File** to open a File **Open** window. Use the window to browse to the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

directory where the map file is to reside, then enter the name of the file. The map file is saved with a **.mapping** extension. After the mapping file has been created, **Update External Component Map** saves any mappings you enter manually.

9. **Load Map File** opens a File **Open** window. Use the window to browse to the directory where the map file resides, then select or enter the name of the file. The map file can contain component and/or property mappings, and must have a **.mapping** extension. The same mapping file should be used for both component and property mappings.
10. The **External Component Map** panel lists the components from an external mapping file. The display changes each time the external file is updated.
11. The **External Property Map** panel lists the properties from an external mapping file. The display changes each time the external file is updated.
12. The **Distributed Components** panel lets you specify that imported interconnects will be converted to distributed components if they fall within a minimum coupling distance (specify distance), and to select the nets to be converted.
13. Click **Close**. The Schematic Editor shows the design, and the Projects window shows the imported project.

Closing Projects

To close the current Twin Builder project without exiting Twin Builder, do one of the following:

- Click **File** > **Close**.
- Right-click the project icon in the **Project Manager** window and select **Close** from the shortcut menu.

> **Note:**
>
> If there are unsaved changes to the project—signified by the presence of an asterisk at the end of the project name in the **Project Manager** window—the system prompts you to save the changes before closing the project. See ***Saving Projects*** for details.

Saving Projects

Click **File** > **Save As** to:

- Save a new project.
- Save the active project with a different name or in a different location.
- Save the active project in another file format for use in another program.

Click **File** > **Save** to save the active project.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

- If the active Twin Builder project is part of an Ansys Workbench Project, **Save** and **Save As** will not appear in the **File** menu.
- When attempting to save a project, if there are unsaved changes in an open editor other than the schematic editor (the symbol editor, for example), a dialog appears asking if you want to save changes to the work in the open editor.

Click **File** > **Save as Technology File** to save the selected design in ***.asty** format.

A prompt appears when you attempt to save the previous file. If you agree to the prompt, the file is upgraded to the Twin Builder version in which you are running the software. In this case the file may no longer be compatible with previous versions of Twin Builder. If you do not agree to the prompt, the file is not saved, so the file retains the previous compatibility.

Click **File** > **Save Workbench Project** to save the active Twin Builder project to its associated Ansys Workbench Project location. Refer to the Ansys Workbench documentation for information on working with Workbench Projects.

Related Topics

Saving a New Project

Saving a Copy of a Project

Deleting Projects

Saving a New Project

1. Click **File** > **Save As**.
2. Use the file browser to find the directory where you want to save the file.
3. Type the name of the file in the **File name** box.
4. Use the correct file extension for the file type.
5. Click **OK**.

   Twin Builder saves the project to the location you specified.

Warning:

- Be sure to save projects periodically. Saving frequently helps prevent the loss of your work if a problem occurs.
- Although Twin Builder has an autosave feature, it may not save frequently enough for your needs.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Saving a Copy of a Project

Saving a Copy of a Project

To save an existing, active project with a new name, a different file extension, or to a new location:

1. Click **File** > **Save As**.
2. Use the file browser to find the directory where you want to save the file.
3. Type the name of the file in the **File name** box.
4. Select the desired file extension for the file type.
5. If the window has a **Switch to saved** field, do one of the following:
   - Leave the field selected to display the new file name and close the current file.
   - Cancel the **Switch to saved** selection to save the file under the new name without changing which file is displayed.
6. Click **OK**.

   Twin Builder saves the project with the new name or file extension to the location you specified.

Related Topics

Saving a New Project

Saving Project Data Automatically

Twin Builder stores recent actions you performed on the active project in an autosave file. Should a workstation crash or other unexpected problem occur, the autosave file can be used to recover project data up to the most recent autosave. The autosave file is stored in the same directory as the project file and is named *Project_Name*.**aedt.auto** by default.

Twin Builder saves all data for the project to the autosave file, *except* solution data. By default, Twin Builder saves project data after every 10 edits. An edit is any action you perform which changes data in the project or the design – including actions associated with project management, model creation, and solution analysis.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Autosave *always* increments forward; therefore, even when you undo a command, Twin Builder counts it as an edit.

Once the specified number of edits is carried out, a model-only save will occur. This means that Twin Builder does not save solutions data or clear any undo/redo history.

When Twin Builder autosaves, an **.auto** extension is appended to the original project file name. For example, **Project1.aedt** will be saved as **Project1.aedt.auto**.

> **Warning:**
>
> When you close a project, Twin Builder deletes the autosave file.

To modify the autosave settings, refer to the Project Options section.

Related Topics

Recovering Project Data in an Auto-Save File

Recovering Project Data in an Autosave File

With autosave activated, after a problem occurs, Twin Builder checks the lengths of the original file and the autosave recovery file.

- If both the original and the autosave file are of nonzero length, the **Crash Recovery** dialog box appears. You can re-open the original project file (*Projectn***.aedt**) to recover the solution data, or open the autosave file.
- If the original file has nonzero length and the autosave file has zero length, Twin Builder ignores the autosave file and attempts to open the original file.
- If the original file has zero length and the autosave file has nonzero length, Twin Builder displays a message and attempts to open the autosave file.
- If both the original file and the autosave file have nonzero length, Twin Builder displays an error message.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Warning:**
>
> When you recover a project's autosave file you *cannot* recover any solutions data.
>
> Recovering an autosave file means you will lose any solutions data that existed in the original project file.

To recover project data in an autosave file:

1. If your workstation or Twin Builder has unexpectedly crashed, launch Twin Builder from your desktop.
2. Select **File** > **Open**, and select the original *Project_Name***.aedt** project file for which you want to recover its autosave file.

   The **Crash Recovery** dialog box appears, giving you the option to open the original project file or the autosave file.
3. Select **Open project using AUTOSAVE file** to recover project data in the autosave file, then click **OK**. Twin Builder replaces the original project file with the data in the autosave file.

   Twin Builder immediately overwrites the original project file data with the autosave file data, removing the results directory (solutions data) from the original project file as it overwrites to the autosave file.

> **Warning:**
>
> - If you choose to recover the autosave file, you cannot recover the original project file that has been overwritten.
> - Recovering data in an autosave file is not reversible.

Related Topics

Saving Project Data Automatically

Archiving Projects

Select **File** > **Archive...** to bundle a project—and any other files related to the project that you want to include—in an **.aedtz** file or **.zip** format archive. You can include notes about the contents of the archive and specify whether to include results and solution files, or related external files. Use **Archive** to attempt to detect the necessary files for linked projects and include

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

them in the archive. For example, if a project linked to the main project also has linked or associated files, you can add them.

Archive File Types

Internally, project archive files are .**zip** files, and are compatible with any program that can read .**zip** files (such as WinZip or 7Zip). The system adds a **z** to the project file extension (for example,.**aedtz**) to ensure that archive files have a unique extension. Note that .**zip** files are also included as a possible filter in file selection dialogs.

File Relocation Considerations

In a project to be archived, you can store external files anywhere on your system. One of the goals is for the restored project to be relatively self contained, and NOT to allow the restoration of an archived project to haphazardly write files anywhere on your system.

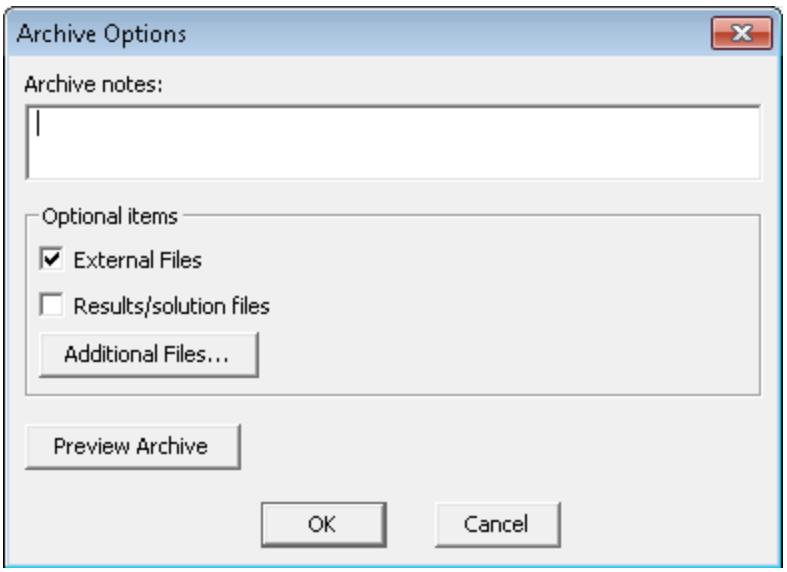
To achieve this, it is sometimes necessary to change the location of files in the archived project such that the external files are now located in the project directory. At archive time, any external files not located in the project directory are relocated to the **restored_files** subdirectory of the project directory in the archived project. Any external files located in the user library or system library are relocated to the personal library directory. Note that the project file written into the archive will refer to the files at the new locations, and the original project file will not be altered.

Archive Preview

Archive includes a preview feature that lets you review the contents of a planned archive.

To archive the current project:

1. Select **File** > **Archive...**. The **Archive Options** dialog box appears.



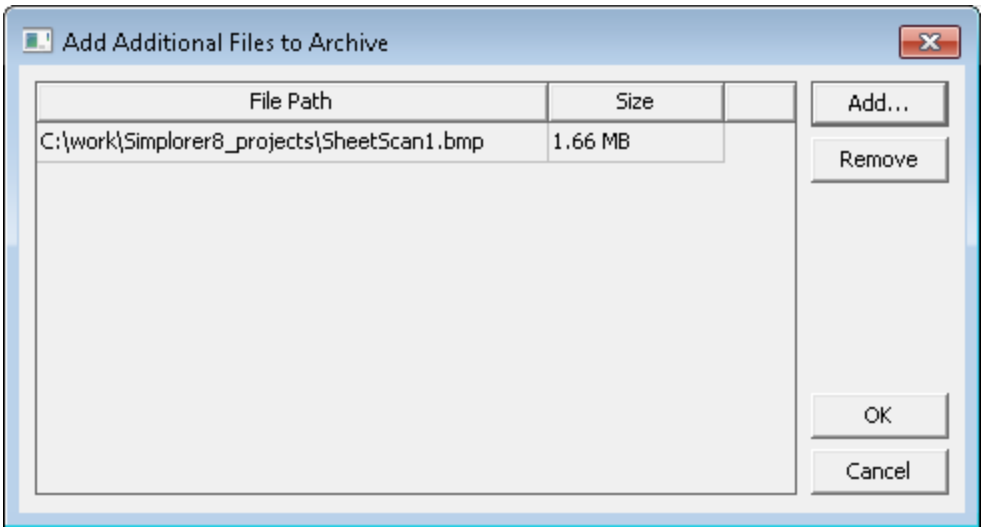
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Archive notes: You can specify notes that will be visible when previewing the archive. These notes can be viewed from the preview dialog without actually restoring the archive.

External Files: Select this check box to include all external files in the archive. The **External Files** check box refers to any existing files associated with the project, such as linked files, or files added through the **Project** > **Insert Documentation File** command or **Project** > **Data Set** command.

Results/solution files: Select this check box to include the entire results directory in the archive. This may greatly increase the size of the archive file.

Click **Additional Files...** to open the **Add Additional Files to Archive** dialog box.

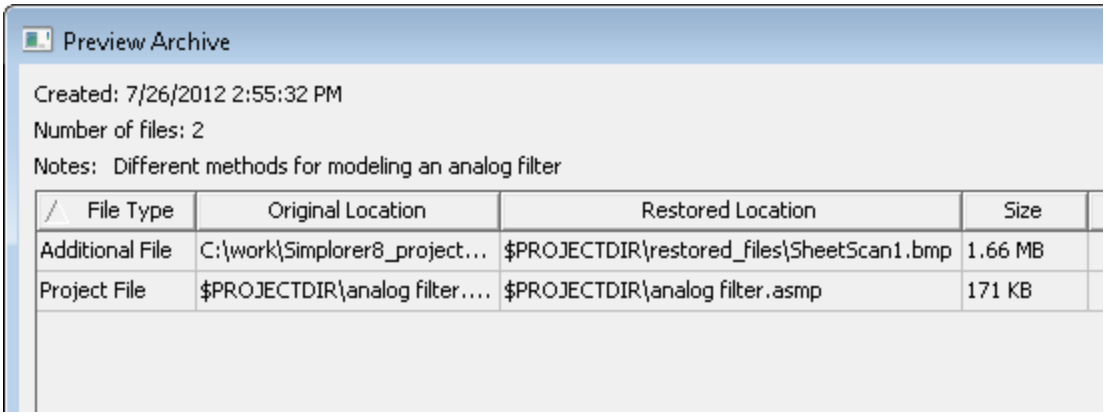


Click **Add...** to locate any additional files (such as documentation files or datasets) you want to include in the archive. You can also select then **Remove** any files listed, and **OK** or **Cancel** any proposed changes.

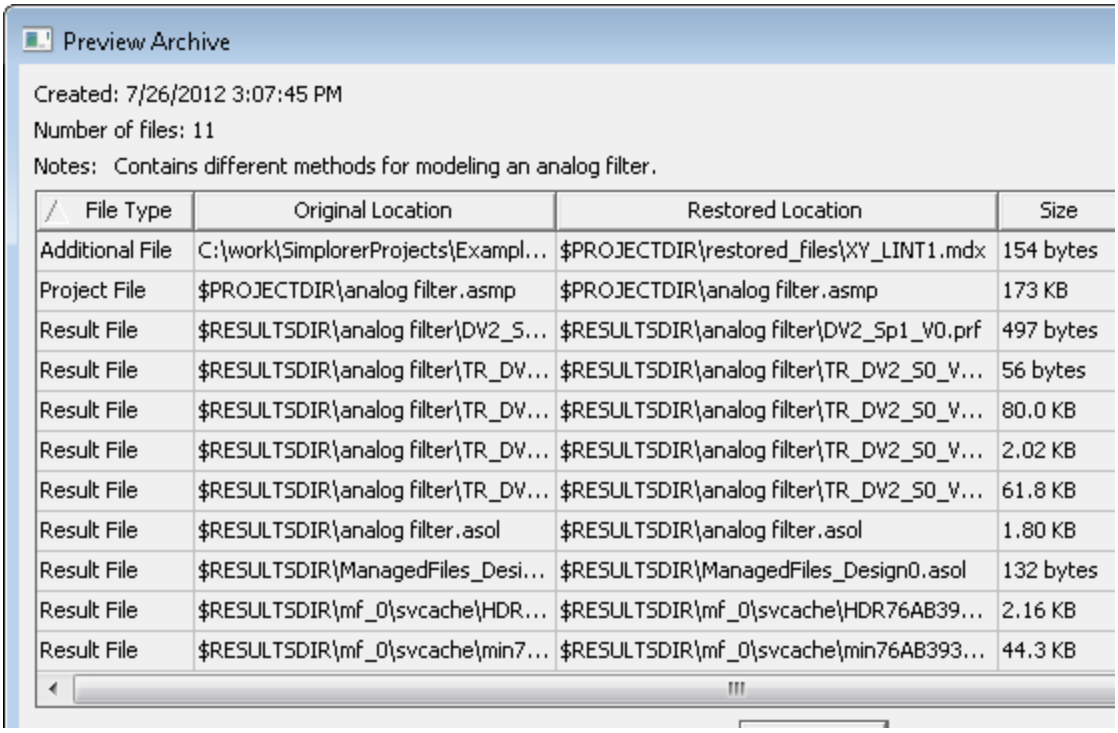
2. Select any optional items, and make any desired archive notes in the text field.
3. When you have made your selections for optional items, select **Preview Archive** to look at the archive contexts, and the locations where restoring from the archive would place

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

them.



Select **Results/solution** if you want to archive those files. Below is a preview showing some included results files.



Previewing an archive before creating the archive can be helpful in order to see exactly what files will be included in an archive, as well as how those files are being relocated. Another purpose of previewing an archive is to view warnings and consider if any additional files need to be added to the archive.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The preview dialog box also displays the archive notes, creation date, and the number of included files.

4. When you are ready to create the archive, close the preview, and specify the format you want to use, **.aedtz** or **.zip**, and specify the archive location and name. Click **OK** to create the archive.

Related Topics

Restoring Archives

Restoring Archives

Follow this procedure to restore an existing archive created with **File** > **Archive**:

1. Select **File** > **Restore Archive**.

   This displays an **Archive to Restore** browser window that lets you navigate your file system for archive files of type **.aedtz** or **.zip**.

2. Select an archive file and click **View Archive** to preview the contents.

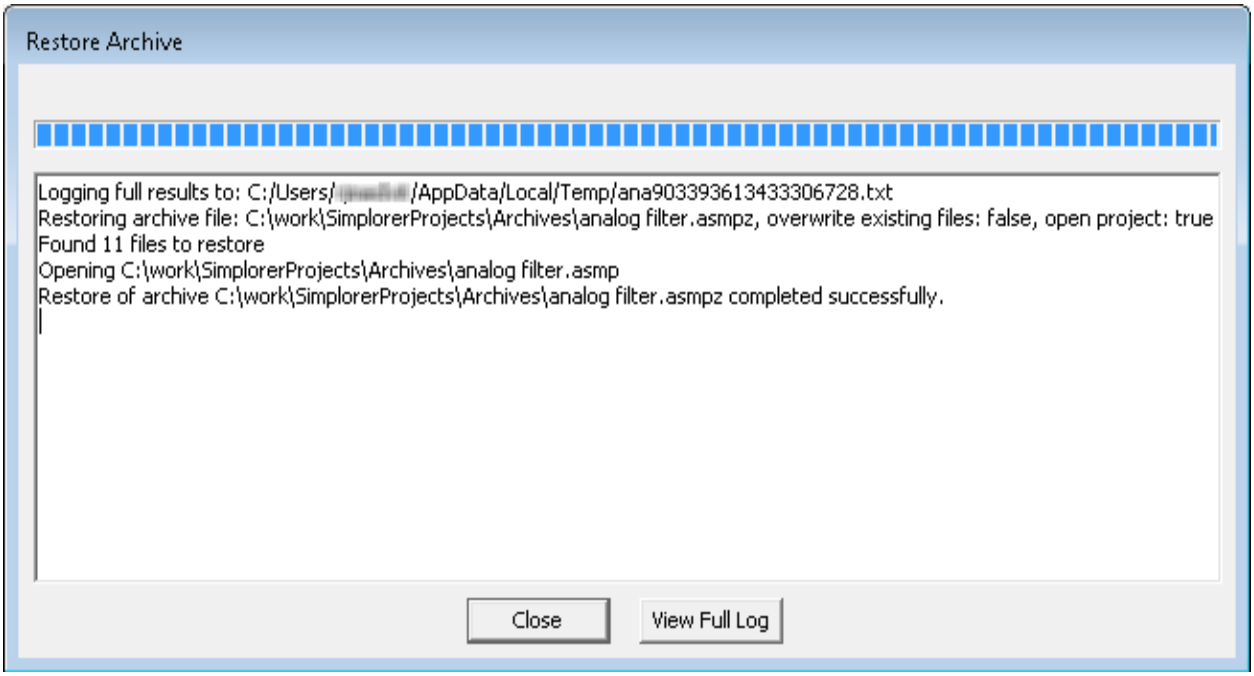
   The preview dialog box shows the same warnings that were generated at archive time. These warnings can identify additional steps needed to update any files which were manually added to the archive.

3. Click **Open** to display a **Project File Restore Location** browser and navigate to the place where you want to restore the file.

   You can edit the file name, and check options to **Overwrite existing files** and to **Open project after restoring**.

4. Click **Save** to restore the archived file. A dialog box displays progress and results of the restoration process.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



A full log file is also generated which contains detailed information about the restore process. The first line in the text window displays the location of the full log file. After the restore has been completed, click **View Full Log** to display a detailed log of the restoration.

Related Topics

Archiving Projects

Renaming a Project

To rename an existing, active project:

1. Right-click a project in the Project tree and select **Rename**.
2. Type the new project name and press **Enter**.

   The new project name appears in the directory and the project remains in the original location.

Deleting Projects

To delete a project:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Right-click a project in the Project tree and select **Delete**. You can also click **Edit** > **Delete**.
2. A dialog box displays asking you to confirm the project . Click **OK** to delete the files or **Cancel** to retain it.

Removing Unused Definitions from a Project

When you use components, symbols, models, packages, materials, and scripts in a project design their definitions are added to the project **.aedt** file, and are visible in the Project tree **Definitions** folder. These definitions remain in the project **.aedt** file across sessions even if you delete every instance of their usage from the project's designs.

Because the project version of a given definition takes precedence over all other instances of that definition—even a revalued definition in a newly exported **PersonalLib** library file—unused definitions still present in a project can lead to unintended results if you do not delete them.

> **Note:**
>
> Typically, a component definition has a symbol definition associated with it. Some components, such as VHDL-AMS components, also can have model and package definitions associated with them. Because of these dependencies, you cannot remove a symbol, model, or package definition until you have first removed the component definition that uses them. Thus you may need to perform the following procedure several times to completely remove a component definition and its associated definitions.

1. If you have removed every usage instance of a definition from a project, you can remove its **Definitions** entry by selecting **Tools** > **Project Tools** > **Remove Unused Definitions**. The **Unused Definitions** dialog box appears.
2. Select definitions to delete. Select individual definitions using the check boxes next to each, or click **Select All** to choose all definitions in the list.
3. When finished selecting definitions to delete, do one of the following:
   - Click **Apply** to remove the selected items while remaining in the dialog box. You can continue removing definitions.
   - Click **OK** to remove the selected items and exit the dialog box.

Datasets

Datasets are collections of plotted data points that can be extrapolated into an equation based on the piecewise linear makeup of the plot. Each plot consists of straight line segments whose vertices represent their end points. A curve is fitted to the segments of the plot and an expression is derived from the curve that best fits the segmented plot. The created expression

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

can then be used in piecewise linear intrinsic functions or as characteristic data for some component parameters.

Related Topics

Datasets Dialog Box

Adding Datasets

Importing Datasets

Editing Datasets

Cloning Datasets

Exporting Datasets

Removing Datasets

Using SheetScan

Datasets Dialog Box

The **Datasets** dialog box lists all datasets currently defined for the project. A preview window displays a plot of the currently selected dataset. Click **Project** > **Datasets** to open the dialog box.

While working in the **Datasets** dialog box, you can:

Add – Opens the **Add Dataset** dialog box in which you can define a dataset by entering data coordinates directly, or by importing data from a file.

Edit – Opens the selected dataset in the **Edit Dataset** dialog box for editing.

Remove – Removes the selected dataset from the project.

Clone – Copies the selected dataset to an editing window for modification. The original dataset remains intact.

Import – Opens the **Import Dataset** dialog box in which you can locate and import characteristics data from several file types.

Export – Opens the **Export Dataset** dialog box in which you can export the selected dataset to a tab-delimited file.

SheetScan – Opens the SheetScan tool in which you can extract characteristics data from graphics such as data sheets.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Adding Datasets

Importing Datasets

Editing Datasets

Cloning Datasets

Exporting Datasets

Removing Datasets

Using SheetScan

Adding Datasets

You can add a dataset to a project in any of three ways:

- Add data manually.
- Import data from an external file.
- Extract data from a graphic using the SheetScan tool.

Related Topics

Importing Datasets

Editing Datasets

Cloning Datasets

Adding Datasets Manually

1. Click **Project** > **Datasets**.The **Datasets** dialog box appears.
2. Click **Add**. The **Add Dataset** dialog box appears.
3. A default name for the dataset appears in the **Name** text box. Rename the dataset if desired.
4. To enter data manually, type the x- and y-coordinates for the first data point in the first row of the **Coordinates** area. Type the x- and y-coordinates for the remaining data points in the dataset using the same method.
    - After you type a point's coordinates and move to the next row, the point is added to the preview plot, adjusting the plot view with each newly entered point.
    - Use the buttons beneath the coordinates table to **Add Row Above** and **Add Row Below** the currently selected row, to **Delete Rows**, and to **Append Rows** to the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

end of the table.

> **Note:**
>
> - The x-coordinate values for successive data points must increase within ten significant digits.
> - You can also click **Import Dataset** to import data coordinates. Doing so will overwrite any existing values in the **Coordinates** area.

5. Click **Swap X-Y Data** to exchange all x- and y-coordinate values.
6. Click **Export Dataset** to open a file browser in which you can export the dataset to a tab-delimited file.
7. When you are finished entering the data point coordinates, click **OK**.

   The dataset plot is extrapolated into an expression that you can use in parametric analyses or assign to a property value.

Related Topics

Importing Datasets

Editing Datasets

Cloning Datasets

Exporting Datasets

Dataset Preview Plot Properties

1. Double-click an element to open its properties dialog box. You can change the properties of various elements of the preview plot such as the major and minor grid colors, title font, and trace line style.
2. Depending on the type of element selected, use the tabs to set the **Color**, **Font**, **Line Style**, **Scaling**, **Title**, and **Legend** properties.
3. Right-click inside the plot window to open a menu containing commands to add and delete data markers and labels, change trace type, and print the plot.
4. Click **Save as Defaults** to save your changes as default values.
5. Click **OK** to apply the changes only while the current dataset plot window is open. Click **Save as Defaults** to keep the changes.

Importing Datasets

Follow this procedure to import data for a dataset from a file:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Project** > **Datasets**. The **Datasets** dialog box appears.
2. Click **Import**.
3. In the file browser window that appears, choose data from the following file types:

.mdx, .mda	Twin Builder Characteristic format
.xls, .xlsx	Microsoft Excel
.txt	text file
.csv	Comma-separated value
.out	Maxwell SPICE (read-only - reads data inside the KW_DATA section)
.cfg	Comtrade (IEEE Std C37-111-1999)
.dat	TEK Oscilloscope
.tab	tab delimited data files

   a. If you select a file type other than **.xls**, **.xlsx**, **.txt**, **.dat**, **.out**, or **.csv** the data is imported immediately into the **Add Dataset** dialog box.

   > **Note:**
   >
   > Importing an **.xls** or **.xlsx** file requires Microsoft Excel to be available.

   b. Select an **.xls** or **.xlsx** file containing multiple sheets to open a **Table Properties** dialog box where you can choose the desired sheet from a drop-down list. Otherwise, selecting an **.xls** or **.xlsx** file imports the data immediately into the **Add Dataset** dialog box.

   > **Note:**
   >
   > - The x-coordinate values for successive data points must increase within ten significant digits. Non-numeric entries are assigned a value of zero.
   > - The first row of data is assumed to contain column headings and is ignored.

   c. Selecting a **.txt**, **.dat**, **.out**, or **.csv** file opens an **Import** dialog box in which you can specify settings for reading the data in the file for import. Choose the **Separator** and **Decimal Symbol**, as well as the line at which to begin the import. Select the channels to import with the check boxes in the **Select Channels** combo box. The dialog box shows both the original text and the text as it would appear when imported based on

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the current import settings. For more information, see ***Import Data from Generic Text Files***.

4. When satisfied with the import settings, click **OK** to import the data.

   Imported datasets take the name of the imported file. If a dataset of the same name already exists, a number will be appended to the end of the new dataset (for example, a duplicate 'dataset' becomes 'dataset1').

5. After importing the data, you can modify it manually (see ***Editing Datasets***).

> **Note:**
>
> You can also import a dataset in the **Add Dataset** and **Edit Dataset** windows.

Related Topics

Adding Datasets

Editing Datasets

Cloning Datasets

Exporting Datasets

Removing Datasets

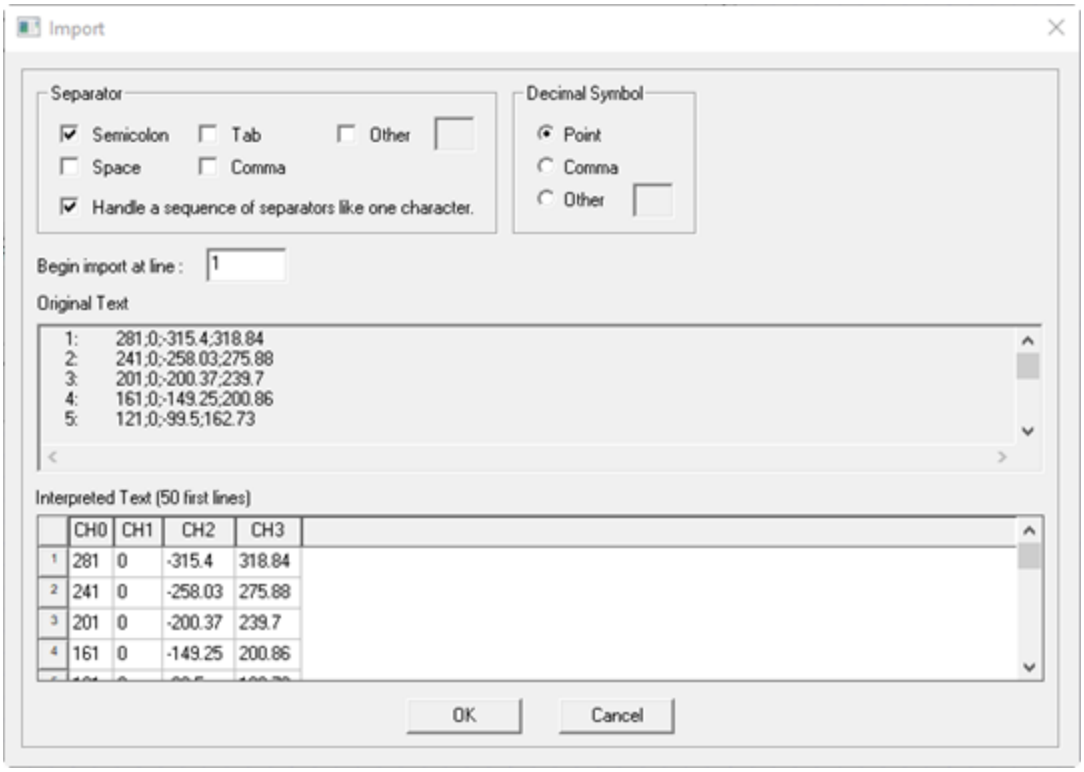
Using SheetScan

File Formats

Import Data from Generic Text Files

In this import data dialog box, you can specify settings for reading table data from a generic text for import (.**txt**, .**dat**, .**out**, or .**csv**). You can choose the Separator and Decimal Symbol, as well as the line at which to begin the import. When importing simulation data, you can select the channels to import with the check boxes in the **Select Channels** combo box. The dialog box shows both the original text and the text as it would appear when imported based on the current import settings.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- **Separator** – Choose the separation character between the data items. Depending on the file format and some initial scanning of the data, some separator will be pre-selected. After changing the setting for the separator, the data is re-scanned, and the preview updated.
- **Decimal Symbol** – Choose the decimal character of the data items. The decimal character is chosen by default depending on the computer language, but it can be changed. After changing the setting for the decimal symbol, the data is re-scanned, and the preview updated.
- **Begin Import at Line** – Set the line number at which the data is imported into the dataset. If the file contains one or more header lines, which may contain comments or additional information, those lines can be skipped. If the first line of a data file contains a header line with the names and possibly units of the data columns, the import tries to scan and use this data. If any specified data unit is not a base SI unit (for example, mV or kOhm) the data values will be converted into the base SI unit.
- **Original Text** – Show the original text of the first 50 lines of the data file.
- **Interpreted Text** – Shows a preview of the data for the first 50 lines as it would be imported.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Editing Datasets

1. Select **Project** > **Datasets**. The **Datasets** dialog box appears.
2. Click the dataset name you want to modify and click **Edit**. The **Edit Dataset** dialog box appears.
3. Optionally, type a name other than the current name for the dataset in the **Name** text box.
4. Edit values for existing data points, adding new datapoints as desired.

   Use the buttons beneath the coordinates table to add rows above or below the currently selected row, to delete rows, and to append rows to the end of the table.

   Optionally, click **Swap X-Y Data** to exchange all x- and y-coordinate values.

   > **Note:**
   >
   > The x-coordinate values for successive data points must increase within ten significant digits.

   The plot is adjusted to reflect the revised data points.
5. When you are finished editing, click **OK** to save the changes and return to the **Datasets** dialog box.
6. When finished editing datasets, click **Done**.

Related Topics

Adding Datasets

Importing Datasets

Cloning Datasets

Exporting Datasets

Removing Datasets

Using SheetScan

Cloning Datasets

Cloning a dataset generates a copy of an existing dataset. The clone can then be modified as needed.

1. Click **Project** > **Datasets**. The **Datasets** dialog box appears.
2. Click the dataset name you want to clone and click **Clone**. The **Clone Dataset** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Modify the dataset as needed.

Related Topics

Adding Datasets

Importing Datasets

Editing Datasets

Exporting Datasets

Removing Datasets

Using SheetScan

Exporting Datasets

1. Select **Project** > **Datasets**. The **Datasets** dialog box appears.
2. Click **Export**. The **Export Dataset** dialog box appears.
3. Browse to the location you want to store the exported dataset.
4. Name the file (exported datasets are tab-delimited and are given a .**tab** extension), and click **Save** to complete the export operation.

> **Note:**
>
> You can also export a dataset in the **Add Dataset** and **Edit Dataset** dialog boxes.

Related Topics

Adding Datasets

Importing Datasets

Editing Datasets

Cloning Datasets

Removing Datasets

Using SheetScan

Removing Datasets

1. Click **Project** > **Datasets**. The **Datasets** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Click the dataset name you want to remove and click **Remove**.
3. When finished removing datasets, click **Done**.

Related Topics

Adding Datasets

Importing Datasets

Editing Datasets

Cloning Datasets

Exporting Datasets

Using SheetScan

Using SheetScan

SheetScan lets you extract characteristics data from graphics such as data sheets which have been scanned and saved in any of the following formats: .**bmp**, .**dib**, .**jpg**, .**gif**, .**tif**, .**tga**, or .**pcx**.

To use SheetScan, Click **Project** > **Datasets**. The **Datasets** dialog box appears. Click **SheetScan** to open the **SheetScan** dialog box.

In addition to importing graphic files directly, SheetScan can search the Internet for datasheet information and transfer a snapshot of the web page to the SheetScan editor where you can map axes on the image as an overlay. You can manually add datapoints to approximate the characteristic curves on the datasheet. The sampled data can be converted to Twin Builder format, and the extracted data exported to a Twin Builder dataset or saved to a file.

The process for creating a dataset using SheetScan involves four basic operations:

- Load a datasheet into SheetScan.
- Define a coordinate system for the imported datasheet picture.
- Define a characteristic curve using the datasheet picture as reference.
- Export the characteristic curve data to a file or to a dataset.

Related Topics

SheetScan Toolbars

SheetScan Settings

The Curve Values Window

Loading a Datasheet into SheetScan

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Deleting a Datasheet Picture

Defining a SheetScan Coordinate System

Defining a Characteristic Curve in SheetScan

SheetScan Toolbars

Three toolbars are available in SheetScan. They provide convenient access to commands that are also found in the SheetScan main menu. Select **View** > **Toolbar** to toggle toolbar display.

- The **Standard** toolbar includes access to basic Windows functions such as file **Open** and **Save**, **Cut**, **Copy**, **Paste**, **Print**, and **Help**.

- The **Curve** toolbar contains tools for working with curve values. Use the drop-down menu to select the curve on which to work. Use the other tools to change curve settings, change the curve's coordinate system, and to select, append, delete, and insert points on the active curve.

- The **Zoom** toolbar provides tools for scaling the current view, zooming in and out, resetting the zoom to 100 percent, and toggling the display of the curve's grid on and off.

SheetScan Settings

Select **Options** > **Settings** to configure default settings. The **Settings** dialog box contains three tabs:

- **Document** – Set the width and height of the sheet created when a picture imported into the SheetScan editor. You can enter the dimensions manually, or let SheetScan adapt the dimensions to the picture being loaded.
- **Axis** – Set the default **Name**, **Unit** of measure, **Scaling** factor, and **Offset** value for the X- and Y-axes. Select **Monotonicity in X** to prevent you from adding consecutive data points whose X-values are not increasing.
- **Representation** – Choose whether to connect points on the characteristic curve and to choose the color of the connecting line. You can also display markers for the point chosen when defining a curve, to set the color of displayed markers, and to set the color of markers when they are selected.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You can override the default settings for individual curves. See Defining a Characteristic Curve in SheetScan.

Related Topics

Loading a Datasheet Picture into SheetScan

Defining a SheetScan Coordinate System

Defining a Characteristic Curve in SheetScan

The Curve Values Window

Click **View** > **Curve Values** to toggle the display of a dockable **Curve Values** pane that displays the data points you place when creating a characteristic curve. Data for each curve on a sheet is displayed on its own tab. You can manually change the X and Y values in the table to fine-tune the characteristic curve.

Related Topics

Loading a Datasheet Picture into SheetScan

Defining a SheetScan Coordinate System

Defining a Characteristic Curve in SheetScan

Loading a Datasheet Picture into SheetScan

By default, SheetScan opens a new, blank datasheet editing window. There are two ways to load a datasheet picture into the editor.

Loading a Datasheet Picture Directly

1. Browse directly to the datasheet picture file. Click **Picture** > **Load picture** to open a file browser window.
2. When you have located the desired file, click **OK** to load the image into the SheetScan editor. Supported file types include: .**bmp**, .**dib**, .**jpg**, .**gif**, .**tif**, .**tga**, .**pcx**, .**htm**, and .**html**.

Loading a Datasheet Picture Using the HtmlViewer

1. Click **Picture** > **Internet** to open the **Sheetscan HtmlViewer**.
2. Browse the Internet for the desired datasheet.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Resize the **HtmlViewer** window and adjust its scrollbars until the desired portion of the datasheet is in view
4. Click to copy the visible contents of the **HtmlViewer** window into the SheetScan editor window.

> **Note:**
>
> To hide the datasheet picture, click **View** > **Picture**.

Related Topics

Loading a Datasheet Picture into SheetScan

Defining a SheetScan Coordinate System

Defining a Characteristic Curve in SheetScan

Deleting a Datasheet Picture

To delete a Datasheet picture, click **Picture** > **Delete picture**.

> **Warning:**
>
> You cannot undo this action. If you delete a picture from the SheetScan editor, you must reload it from the source file or Internet web page.

Defining a SheetScan Coordinate System

Follow this procedure to define the coordinate system:

1. Select **Coordinate System** > **New** to open the **Coordinate System** dialog box.
2. Click **Point1**. The **Coordinate System** dialog box disappears temporarily and the cursor changes to a crosshairs.
3. Click inside the datasheet graph. The **Coordinate System** dialog box reappears displaying the X- and Y-Coordinate values for the chosen point.
4. Enter the X- and Y-values for this point. Typically, these values will correspond to the values taken from the axis scale values on the datasheet.
5. Select the desired scaling (linear, logarithmic, or decibel) for both the X- and Y-axes.
6. Repeat steps 2 through 4 for the **Point2** and **Point3** buttons.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7. Click **OK**. The grid appears in the graphic.

> **Note:**
>
> - Select **Coordinate System** > **Properties** to edit the grid after placement. You can also click on the **Curve** toolbar, or right-click the SheetScan editing window and select **Coordinate system** from the shortcut menu.
> - Click **View** > **Grid** to toggle display of the grid, or click on the **Curve** toolbar.

Defining a Characteristic Curve in SheetScan

Once you have loaded a datasheet picture in the editor and have defined a coordinate system, follow this procedure to define one or more characteristic curves:

1. Click **Curve** > **New**. The **Curve Settings** dialog box opens.
2. Define the properties of the curve. See SheetScan Settings for a detailed explanation of the settings you can make on the **Axis** and **Representation** tabs.
3. When finished defining curve properties, click **OK**. The cursor changes to cross hairs.
4. Click the points of the characteristic which you want to capture for the dataset. The points are connected.
5. Repeat steps 1 through 4 for each additional characteristic curve you want to define.

Selecting a SheetScan Characteristic Curve

1. To select a SheetScan curve for editing, do one of the following:
   a. Click **Curve** > **Select** on the SheetScan menu bar.
   b. Right-click in the editor window and select **Select Curve**. The **Select Curve** dialog box opens.
2. Click the desired curve name to highlight it and click **OK** to select the curve.

> **Note:**
>
> If the **Curve Values** dialog box is open, you can also click the tab of the desired curve to select it.

Changing Characteristic Curve Settings

Follow this procedure to change curve settings for a characteristic curve:

1. Select the curve whose settings you want to change.
2. Click **Curve** > **Change Settings**. The **Curve Settings** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Change the properties of the curve as desired. See SheetScan Settings for a detailed explanation of the settings you can make on the **Axis** and **Representation** tabs.
4. When finished changing curve properties, click **OK**.
5. Repeat steps 1 through 4 for each additional characteristic curve you want to change.

Editing a SheetScan Characteristic Curve

The following SheetScan curve editing functions are available on the **Curve** menu, the Curve toolbar, or the editor window context menu:

- Select points – click a point to select it. Ctrl+click selects multiple points.
- Append points – click to add data points to the end of a curve.
- Delete points – click a data point to remove it from the curve.
- Insert points – click to insert new data points between existing data points.

Deleting a SheetScan Characteristic Curve

To delete a characteristic curve and all of its associated data points:

1. Select the desired curve.
2. Click **Curve** > **Delete**.

> **Warning:**
>
> You cannot undo this action. If you delete a curve and its data points from the SheetScan editor, you must reconstruct it manually.

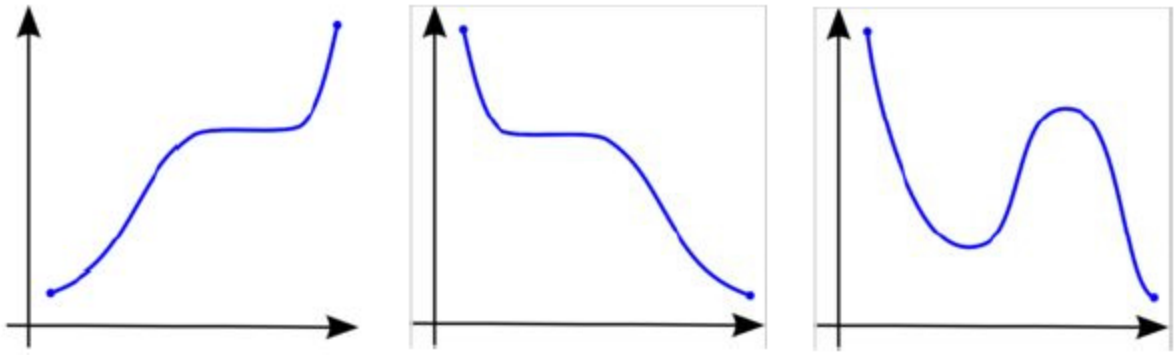
Checking for Monotonicity in X

Twin Builder requires that characteristics be monotonically increasing along the X-axis. In other words, successive data points must have increasing X-values, while Y-values may both increase and decrease. You can check for monotonicity in X as follows:

1. Select **Curve** > **Check Monotonicity**.

   If the characteristic curve is monotonically increasing in X-value, the check completes without notice. Typical examples of curves that meet monotonicity criteria appear below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. If the characteristic curve is not monotonically increasing in X-value, a dialog box displays informing you that errors were found. Click **Yes** to have SheetScan correct the errors.

Importing Characteristic Data into SheetScan

SheetScan supports data import from the following file types: Twin Builder Characteristic (*.**mdx**, *.**mda**), Microsoft Excel (*.**xls**, *.**xlsx**), text (*.**txt**), comma separated value (*.**csv**), Spice (*.**out**), Comtrade (*.**cfg**), and TEK Oscilloscope (*.**dat**).

1. Click **File** > **Import** to import characteristic curve data into SheetScan.
2. In the file **Open** dialog box, select the desired data file and click **OK**. The **Curve Settings** dialog box appears.
3. Change **Curve Settings** as needed and click **OK** to complete the data import.

   The new characteristic curve is added to the current SheetScan sheet.

Exporting SheetScan Data

You can export SheetScan data directly to a dataset, or to any of the following data file formats: Twin Builder Characteristic (*.**mdx**), comma separated value (*.**csv**), or Comtrade (*.**cfg**).

1. Select **File** > **Export** to export the curve.
2. In the **Save** dialog box, choose **Current Curve** (default) to export current curve data, or **Curves** if you want to choose the curves whose data you want to export. Choosing **Curves** reveals a list box showing all of the curves available for export. Select the **Export** check box for the desired curves.
3. Choose **Equidistant** if you want to set the **Start** and **StopX-Channel** values and a **Sample Rate** or **Number of samples** for the exported datasets,
4. For **Curves**, choose **Multidimensional Table** if you want to enter a **Y-Value** for each curve. Data output using the **Multidimensional Table** option can only be exported to a file.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Choose **Dataset** to export curve data directly to the project's dataset file. Choose **File** to export curve data to any of the file formats listed above.

Related Topics

Adding Datasets

Importing Datasets

Editing Datasets

Cloning Datasets

Exporting Datasets

Removing Datasets

Undoing Commands

Select **Edit** > **Undo** to cancel the last action you performed on the active project or design. This is useful for undoing unintended commands related to project management, model creation, and post-processing.

> **Note:**
>
> You cannot undo an analysis that you've performed on a model using **Twin Builder** > **Analyze**.

1. In the **Project Manager** window, do one of the following:
   - To undo the last action you performed on the *active project*, such as inserting a design or adding project variables, click the project icon.
   - To undo the last action you performed on the *active design*, such as drawing an object or deleting a report, click the design icon.
2. Click **Edit** > **Undo**, or click **Undo** on the ribbon.

   The most recent action is now undone.

> **Note:**
>
> When you save a project, Twin Builder always clears the entire undo/redo history for the project and its designs.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Redoing Commands

Redoing Commands

Click **Edit** > **Redo** to reapply the last action that you canceled. You can redo a canceled action related to project management, model creation, and post-processing.

1. In the **Project Manager** window, do one of the following:
   - To redo the last action you canceled on the *active project*, such as inserting a design or adding project variables, click the project icon.
   - To redo the last action you canceled on the *active design*, such as drawing an object or deleting a field overlay plot, click the design icon.
2. Click **Edit** > **Redo**, or click **Redo** on the ribbon.

   The most recent canceled action is now reapplied.

> **Note:**
>
> When you save a project, Twin Builder always clears the entire undo/redo history for the project and its designs.

Related Topics

Undoing Commands

Inserting a Documentation File

You may want to add a documentation file to the Project tree so that you can readily locate it for reference. For example, you may have documentation that describes the details of your project; or perhaps a presentation explaining the project.

Follow this procedure to include documentation files with the project:

1. Click **Project** > **Insert Documentation File**.

   This opens a file browser dialog that lets you navigate your file system.
2. Select the desired file and click **OK**.

   The documentation file is added to the Project tree.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You can add file types other than documents—including executable files—to the Project tree using this command.

Importing and Updating Twin Builder Models and Components

Twin Builder lets you **Import Twin Builder Models** into a project. Similarly, in the model editors you can **Compile & Update Project** models and associated components that are already present.

Related Topics

Importing Twin Builder Models

Using the Compile & Update Project Command

Importing Twin Builder Models

Twin Builder lets you import multiple models and symbols, as well as create or update multiple components using the imported models and symbols. The following simulation model file types can be imported:

- VHDL Models (**VHD**)
- VHDL Encrypted Models (**VEM**)
- SPICE Subcircuits (**CIR**, **LIB**, **SPC**, **SPI**, **MOD**)
- SML Models (**SML**)
- C-Model Dlls (**DLL**)
- Modelica Models (**MO**, **MOC**)
- External Compiler Model Dlls (**DLL**)

Additionally, SVG Symbols (**SVG**) can be imported.

Follow this procedure to import simulation models into a Twin Builder project:

1. Select **Tools** > **Project Tools** > **Import Twin Builder Models**. A file selection dialog box appears in which you can browse to the file containing the models to import.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. When you have located the desired model file, click **OK**.

> **Note:**
>
> The location of imported CModel **DLL** files is set in the Twin Builder Options panel.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

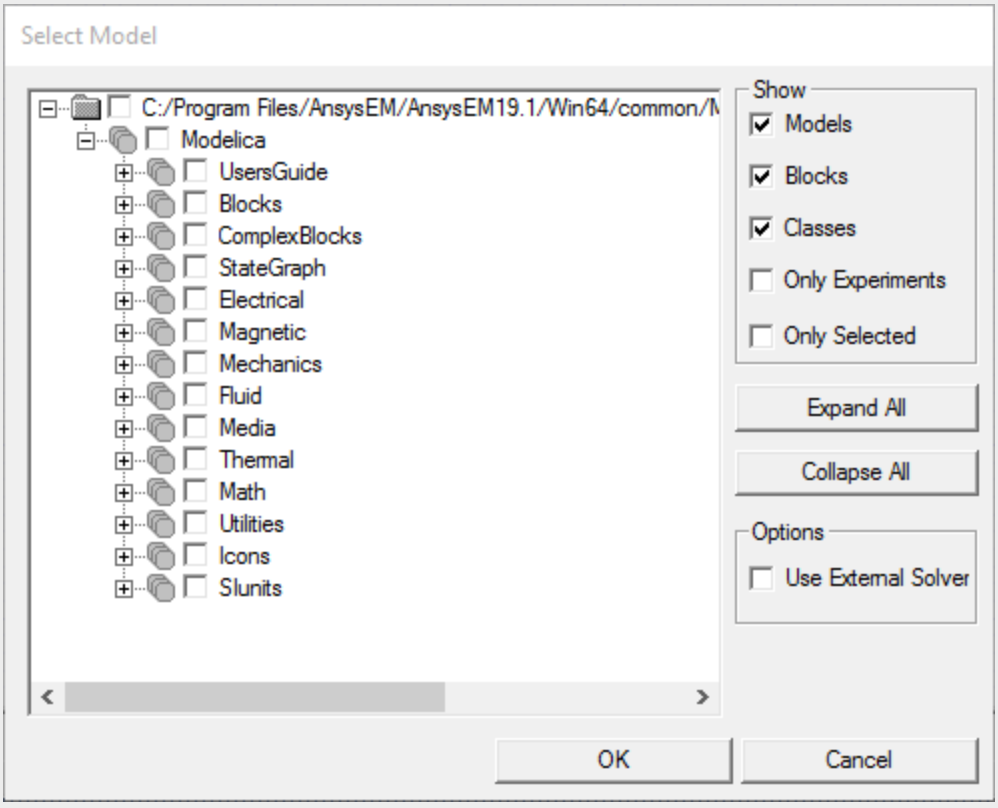
Note:

For a Modelica import, a **Select Model** dialog box appears showing the models, blocks, and classes contained in the selected model or package file. It shows the complete package hierarchy if the selected file is from the library. Review the package hierarchy and choose the desired model, block, or class.

Select **Use External Solver** to enable simulation of the selected model using an external solver. The component of selected model will include the following properties that are used to control the usage of the external solver:

- **_cs_solver** – Solver used in Co-Simulation. 0 - CVode, 1 – fixed step-size explicit Euler
- **_cs_rel_tol** – relative tolerance when CVode is used
- **_cs_step_size** – step size for Explicit Euler

When finished, click **OK**.

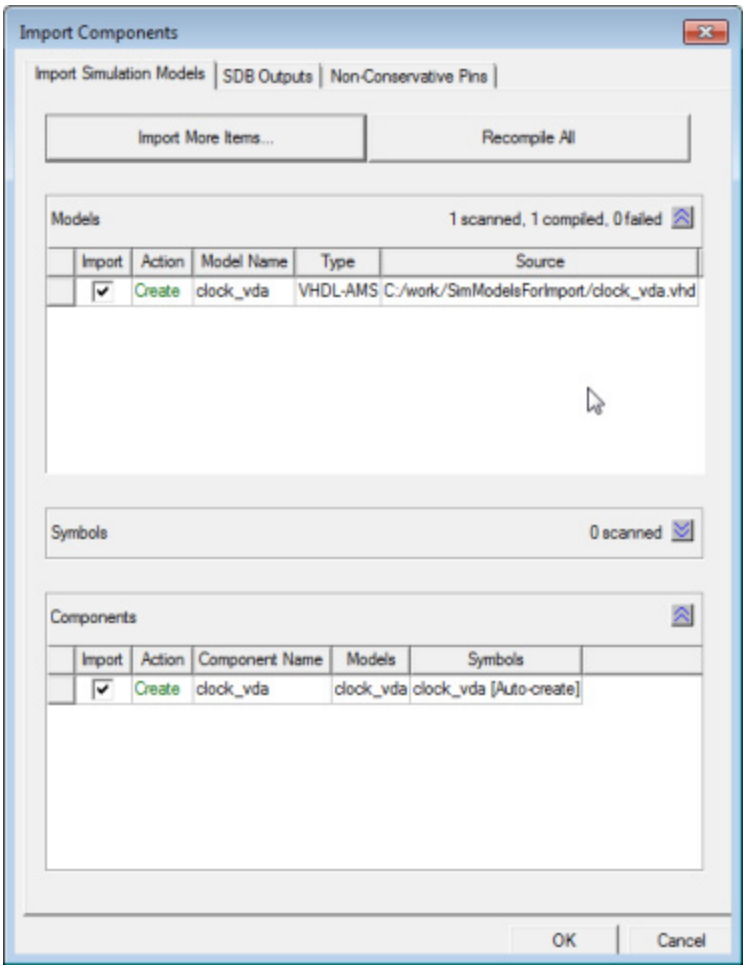


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> After model compilation, select the model interfaces you want to expose in the Twin Builder component. See ***Selecting Component Interface*** for detailed information.

A progress dialog box shows compilation progress of each model. Click **Stop** to end the compilation progress, ignoring the remaining models. Models compiled before you clicked **Stop** are imported.

The **Import Components** dialog box appears.



- The **Import Simulation Models** tab **Models** area lists the models that can be imported, provides an **Import** check box, lists the model **Type** (SML, VHDL, and so on), and the Source path. If a model already exists, the **Action** field indicates that the model will be updated; while for new models, it indicates that a model will be created.

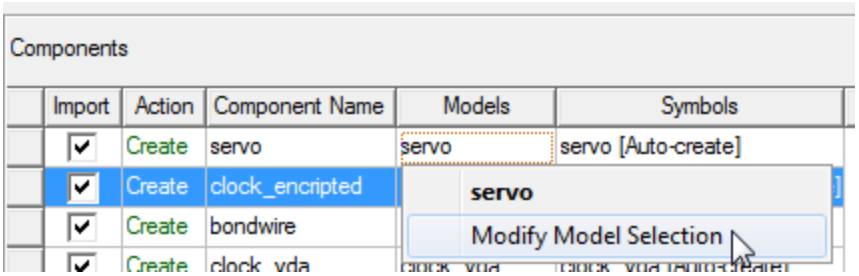
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If a model fails to compile, the dialog box shows failed status for that model, and a new failed simulation model tab is added to the dialog box listing the error messages for each failed model.

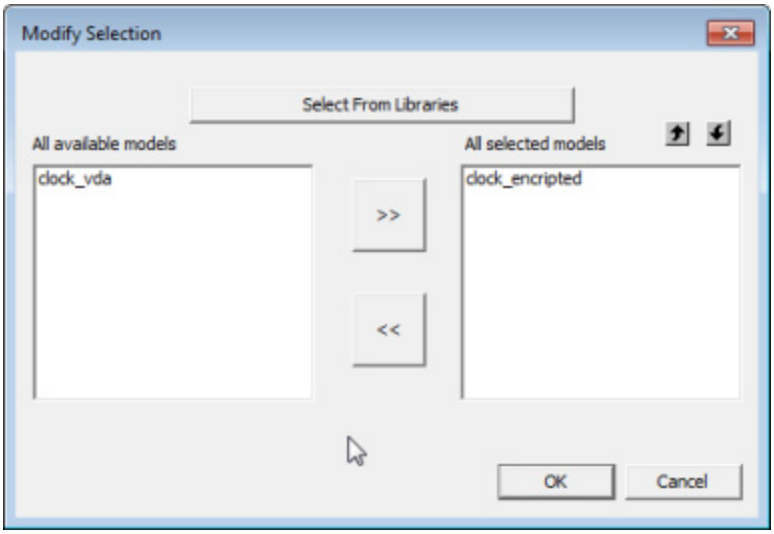
- Similarly, the **Components** area lists the components available for **Import** by name. If a component already exists, the **Action** field indicates that the component will be updated; for new components, it indicates that a component will be created. It also provides a button for choosing a **Symbol** for the component from the symbol library; and a button for choosing a **Model** for the component. The buttons display the names of the model and symbol currently associated with the component.

  To add models to a component in the panel:

  a. Find the component to which you want to add more models, then click **Models** > **Modify Model Selection**.



  b. The **Modify Selection** dialog box displays, listing all available imported models with compatible interfaces in the left box. Use the arrows to move selected models to the right list-box. To select models from a library, click **Select From Library**.



  c. Click **OK** to add all selected models to the component.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- An **SVG** image file associated with a model name, and residing in the same folder as the model will be imported and attached to the component. For example, if you upload **C:\Lib_Model\Upload\simmodel1.vhd** (containing a model named "**simmodel**"), and if there is an **SVG** file associated with the model name in any of the following scenarios:
- **C:\Lib_Model\Upload\simmodel.svg** (Same folder as the model file)
- **C:\Lib_Model\Symbol\Simmodel.svg** (Same folder level as the model file's folder)
- **C:\Lib_Model\simmodel.svg** (One level above the model file's folder) - the **svg** file will be attached to the component.
- The **Symbols** panel is minimized by default. Click to expand the panel and view its contents.

To change the symbol selection for a component:

a. Click the **Symbols** field of the component whose symbol you want to modify to open a drop-down list and select **Modify Symbol Selection**.
b. The **Modify Selection** dialog box displays all available **.svg** symbols in the left box. Use the arrows to move the selected symbol to the right box. To select symbols from a library, click **Select From Library**.

> **Note:**
>
> Components can have only one symbol.

c. Click **OK** to add the selected symbol.
   - Click **Import More Items** to add more models from other folders.
   - A button lets you **Recompile All** the models in the **Models** area. You can also select models in the list and **Recompile Selected** models.

3. Ensure that the **Import** box next to each model and component you want to create is checked.

> **Note:**
>
> If you choose not to create a component, the source files will be compiled – no component is created.

4. On the **SDB** tab, select the quantities and signals you want to set as default outputs for each created component. By default all signals are selected.
5. On the **Non-Conservative Pins** tab, select the non-conservative quantities and signals for which you want pins included on the new component symbol. All terminal pins are added by default; and all signal pins are selected by default.
6. Click **OK** to create a component for each imported model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The new components and models are added to the **Components** and **Models** folders within the current project's **Definitions** folder in the **Project Manager**.

New models are also added to the list of **Project Components** on the **Component Libraries** Components tab where they can be placed onto a schematic for simulation.

> **Note:**
>
> For Modelica models, extra parameters are added for initializing input values. See ***Compiling and Updating a Project to Add a Modelica Model and Component*** for more details.

Handling of Incompatible Definition Names during Simulation Model Import

The syntax rules for definition or instance names in other simulation tools may be different from those in Twin Builder, which may make them incompatible with Twin Builder. For that reason Twin Builder applies some algorithms to make the names compatible with the Twin Builder syntax rules. The most common case for this to happen is during the import of Spice/PSPICE models.

- If the name starts with a number, an underscore (_) is prepended
- Any character inside a name that is any of [a-z][A-Z][0-9][_] is replaced by its ASCII representation (for example, "&" is replaced by "_ASC38_")
- Some special cases commonly used in Spice models:
  - Numbers used as net names: **N_** is prepended.
  - The minus sign (-) is replaced by **_M_**.
  - The plus sign (+) is replaced by **_P_**.

Related Topic

Using the Compile & Update Project Command in a Model Editor

Using the Compile & Update Project Command

The **Compile & Update Project** menu item is common to the C-Model, VHDL-AMS, SPICE, and SML model editors. For all model types Update Project enables:

- Project definitions to be updated.
- New or revised model definitions to be imported (**Import** command).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Creation of components for the models and their addition to the current project.

> **Note:**
>
> The VHDL Package Editor also has an **Update Project** command, which enables creation of the package and its addition to the current project.

To update project definitions from the model editor:

1. Select *model_type***Model Editor** > **Compile & Update Project**. The **Update Components**dialog box appears.

   The **Models** pane lists the name of the model. The **Action** field indicates **Create** or **Update**, and a check box to enable creating or importing the model.

2. The **Components** pane lists all components in the current project that use the model being updated. This list includes any component that has its origin in another library name. The **Symbol** fields are populated with the names of the symbols used by these components. To select **SDB Outputs** or **Non-Conservative Pins**, it is necessary to check **Import Component**.

   > **Note:**
   >
   > If you are updating the model text for the first time (creating a new component), the components appear with the same name as the model name, and the symbol will be created.

   Select the **Import** check box next to each model for which you want to create a component.

   > **Note:**
   >
   > If you do not create a component, the model source files will be compiled – no component is created.

   On the **SDB** tab, select the quantities and signals you want to set as default outputs for each created component.

3. On the **Non-Conservative Pins** tab, select the non-conservative quantities and signals for each component for which you want pins included on the component symbol.
4. Optionally, click the component's **Symbols** button and choose **Modify Symbol Selection**. The symbol **Modify Selection** dialog box appears. Select an **SVG** graphic symbol for the new component from the All available symbols list. You can also click **Select From Libraries** to choose a symbol from another library.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You can modify the symbol after import using the Symbol Editor.

5. Click **OK** to create or update the components and associated symbols for the imported model.

> **Note:**
>
> - In general, if you are updating existing components, the component symbols are updated to reflect the addition or removal of pins based on your selections on the **Non-Conservative Pins** tab.
> - If a symbol for a component you are updating is being used by some other component that uses a different model, a new clone of the current symbol is created and updated for the component.

New components and models are added to the **Components** and **Models** folders within the current project's **Definitions** folder in the **Project Manager**.

New models are also added to the **Components** tab in the **Component Libraries** dialog box where they can be placed onto a schematic.

Related Topics

Importing Simulation Models

Saving Uncompiled Model Changes

The Twin Builder model editors for VHDL-AMS, SML, Spice, and Modelica can save the model text to the model without having it compiled. These changes are saved with the model independently from the last compiled model state so you can save a working state of the model text and work on it later. It is not possible to save multiple states of a model text or switch between them. After the model text is compiled successfully, the saved working state text is removed from the model.

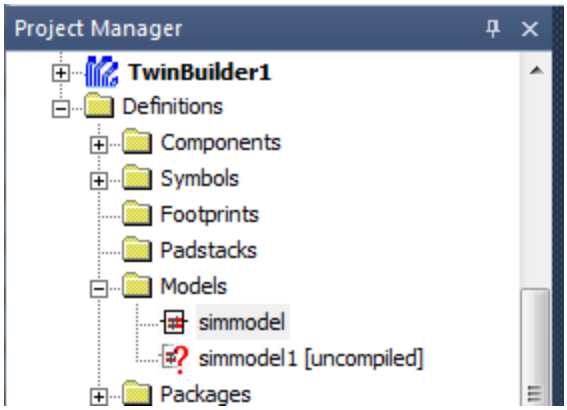
> **Note:**
>
> - You cannot export an uncompiled version of a model test to a model library.
> - If you export a model definition that has been compiled but contains a newer uncompiled version of the model text, the uncompiled version of the model text will be removed from the exported model definition.

To save the uncompiled text of a model:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Click *modeltype***Model Editor** > **Save Uncompiled Changes**.

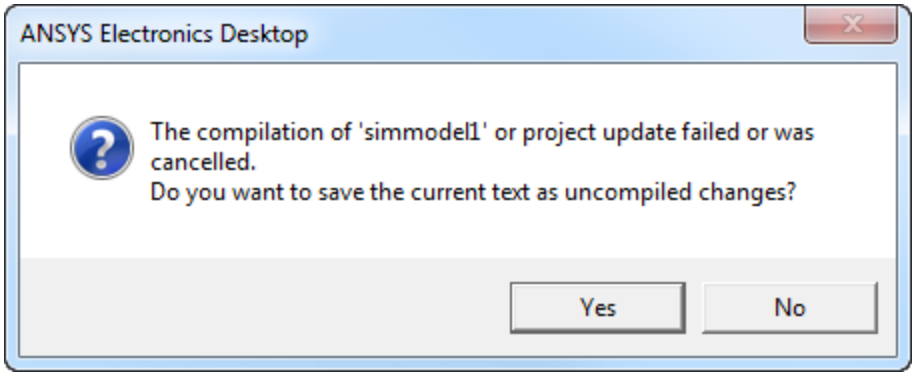
  The model definition is displayed in the **Project Manager** pane:



  The icon next to it indicates its state:

simmodel [uncompiled]	Model text that has never been compiled.
simmodel	Model text that has been compiled successfully and has no uncompiled version stored.
simmodel	Model text that has been compiled successfully and has an uncompiled version stored with it.

- When closing the model editor, Twin Builder tries to compile any uncompiled changes of the model text. If this compilation fails or is canceled, you can save those uncompiled changes. Click **Yes** on the following prompt.



To reopen the uncompiled text of a model:

1. Select **Edit Model** in the context menu of the component instance on the schematic or the model definition in the Project Manager.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. On opening, the model editor asks whether to load the uncompiled text or the text of the last compiled state of the model.

> **Note:**
>
> If you don't load the uncompiled model text when opening the model editor, the uncompiled version of the model text is removed from the model.

Related Topics

Using the VHDL-AMS Model Editor

Using the SML Model Editor

Using the Spice Model Editor

Using the Modelica Model Editor

Exporting a VHDL-AMS Model Description

The standardization of the VHDL-AMS language ensures that models developed in VHDL-AMS are exchangeable among simulators from different vendors. The models you create using VHDL-AMS in Twin Builder can be exported to ASCII files for simulation in other VHDL-AMS tools. Twin Builder lets you export one or more VHDL-AMS models from your design, and also to export entire schematic sheets of VHDL-AMS models. The export of models from the library is discussed in "Adding VHDL-AMS Models to a Library". Exporting sheets to VHDL-AMS netlists is demonstrated in this case study example.

Schematics can be built with VHDL-AMS models from the model libraries. These schematics can be exported to "pure" VHDL-AMS netlists that can be used in other simulation tools. Sheets that contain not only VHDL-AMS models but also non-VHDL-AMS models (internal/SML, and C) can be exported to a VHDL-AMS netlist. In this case, the non-VHDL-AMS models are exported as foreign models, compliant with the IEEE 1076.1 standard.

> **Note:**
>
> Netlists containing non-VHDL-AMS models can only be simulated in Twin Builder and in most cases cannot be used with other VHDL-AMS simulation tools. To use these netlists in another simulator, the non-VHDL-AMS models must be replaced manually in the code by equivalent foreign models that the target simulator can use.

To export a design to a VHDL-AMS model:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Tools** > **Design Tools** > **Export to VHDL-AMS**. The **Export VHDL-AMS Model Description** dialog box appears.
2. Enter the name of the VHDL-AMS file that will contain the exported code of the top-level entity. Click to browse to a new location if desired. The exported file has the .**vhd** extension.
3. A suggested **Name of Top Level Entity** displays. The name in this field specifies the top-level entity name in the exported VHDL-AMS netlist. The top-level entity instantiates the components on the sheet and provides the interconnection information between the components. The suggested name can be modified, but the name must conform to VHDL-AMS identifier syntax requirements (see ***VHDL-AMS Language Fundamentals*** for more details). If the name does not conform to the VHDL-AMS identifier syntax requirements, it will be highlighted for correction.

   In addition to the top-level entity, you can export the VHDL-AMS code of the individual library components that are used in the sheet.

4. To export individual components, choose one of the following options from the **Component** panel:
   - **Export all components** – Exports the source code of all VHDL-AMS models on the schematic.
   - **Export only user-defined components** – exports the source code of only those models that are not available in the pre-installed Twin Builder libraries.

   When you choose an option, additional options in the **Export Components to** panel appear. These options let you specify the target location of the source code belonging to the individual components on the sheet.

   - **Same file as top level entity** – generates one VHDL-AMS file that contains the source code of all the models as well as the description of the top-level entity.
   - **Single file separate from top level entity** – generates the source code of the models in the *file_name*_**Components.vhd** file, where *file_name* refers to the name of the file provided earlier.
   - **Individual files separate from top level entity** – creates several VHDL-AMS files separate from the file containing the top-level netlist. Each of these files contains the VHDL-AMS description of a single model used in the sheet. The file's name is derived from the name of the entity that the file contains.

   A sheet can contain models that are not developed in VHDL-AMS, but are instead developed as internal/SML/C-Models. These models can also be exported to a netlist as foreign models with a VHDL-AMS wrapper.

> **Note:**
>
> Foreign models generated by Twin Builder can only be simulated in Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. To allow VHDL-AMS wrappers to be created around non-VHDL-AMS models, select the **Export SML Components** check box.

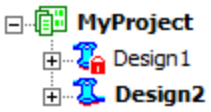
   > **Note:**
   >
   > - If the sheet contains non-VHDL-AMS models and this check box is cleared, the export operation will not be completed.
   > - For pure VHDL-AMS compatibility, do not select this option.

6. Click **OK** to complete the export.

Setting Read Only Designs

Designs can be set as either Read Only or Full Access. Full Access is the default status; you can modify the design in this mode. In Read Only mode, you may only run the design or link it to another design. Setting a design as Read Only protects it from accidental modification.

Read Only designs are marked with a red lock in the **Project Manager** pane:



To change a single design's designation:

1. Right-click the design in the **Project Manager** pane.
2. Select either **Convert to Read Only** or **Convert to Full Access**.

To change the designation of every design in a project:

1. Right-click the project in the **Project Manager** pane.
2. Select **Convert All Designs to Read Only** or **Convert All Designs to Full Access**.

> **Note:**
>
> Read Only settings are not saved when you save the project file.

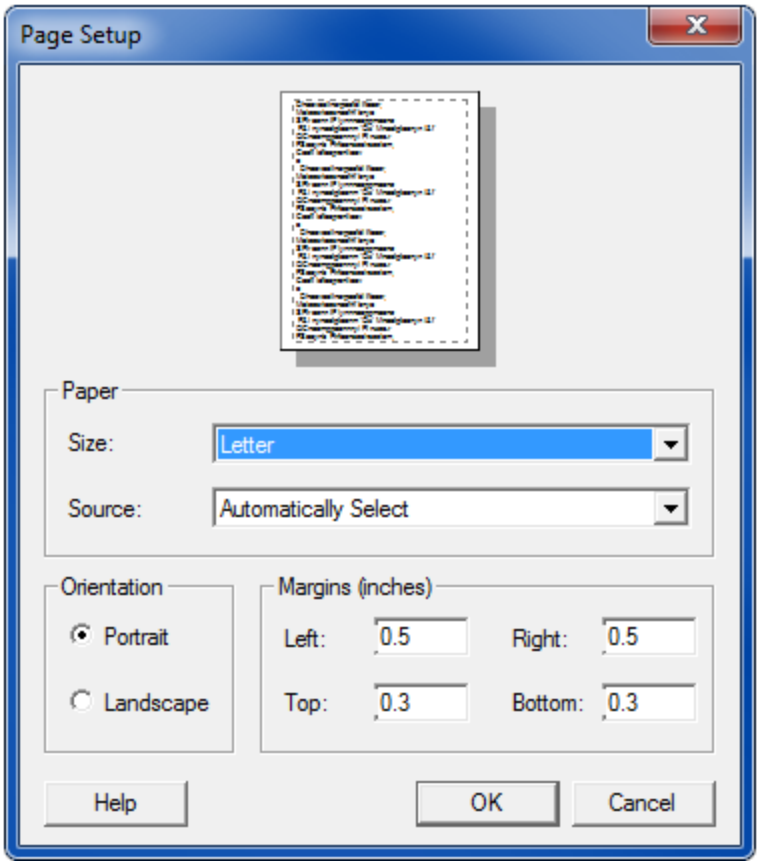
Printing

Use the printing functionality to print the contents of the active window in the design area.

- Select **File** > **Page Setup** to set up formatting to print the active window in the Design Area. The **Page Setup** dialog box includes formatting options such as paper size, orientation, print margins, borders, and labels that are specific to the editor (Schematic,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Netlist, and so on) in the active window. The following example shows the **Page Setup** dialog box for all Twin Builder text editors (Netlist, SML, SPICE, VHDL-AMS, Script, Package, C-Model, Modelica).



See *Printing a Schematic* for additional information on Page Setup for the Schematic Editor.

- When using Sheetscan, click **File** > **Printer Setup** to open a standard Windows **Print Setup** dialog box set up the printer to print the active SheetScan document.
- Click **File** > **Print Preview** to display a preview of the print job.
- Click **File** > **Print** to print the active window in the Design Area. The dialog box may include formatting options such as borders and labels that are specific to the editor (Schematic, Symbol, Netlist, and so on) in the active window.

See *Printing a Schematic* for additional information on printing from the Schematic Editor.

Adding and Saving Project Notes

You can save notes about a project, such as its creation date and a description of the device being modeled. This is useful for keeping a running log about the project.

To add notes to a project:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Twin Builder** > **Edit Notes**. The **Design Notes** dialog box appears.
2. Click the window and type your notes.
3. Click **OK** to save the notes with the current project.

   A **Notes** icon appears in the Project tree.

To edit existing project notes:

- Double-click the **Notes** icon in the Project tree.

  The **Design Notes** dialog box appears, in which you can edit the project's notes.

Event Callbacks

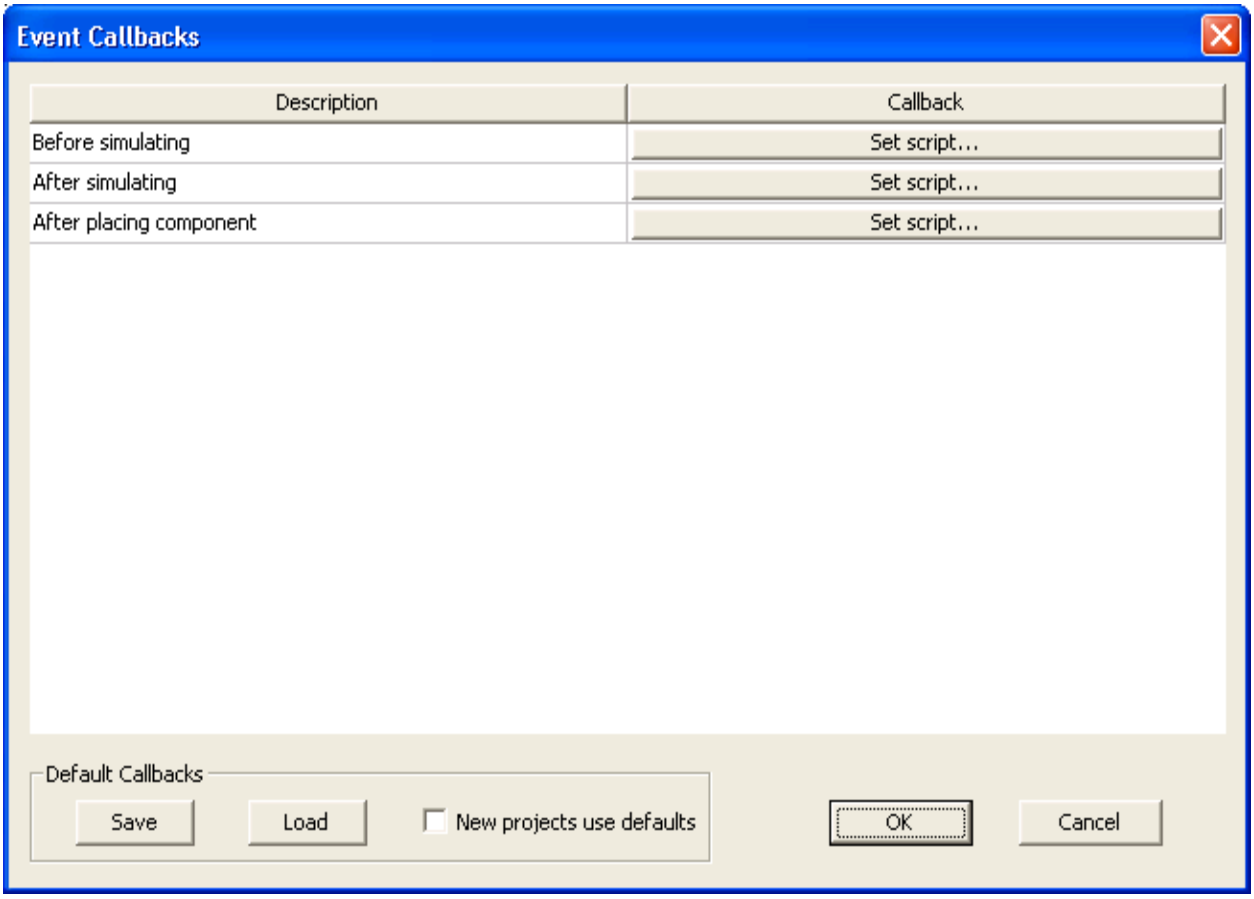
Operations you perform in Twin Builder are called "events" and can include operations such as placing a component or running a simulation. Use **Event Callbacks** to define custom JavaScript and VBScript routines that will run after a triggering event in Twin Builder is detected. After configuring callback scripts, you can save the script settings to the registry and:

- Load callback script settings into any project.
- Set an option to load settings into all new projects that you create.

To define an event-callback script, or to access a previously-defined callback script:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

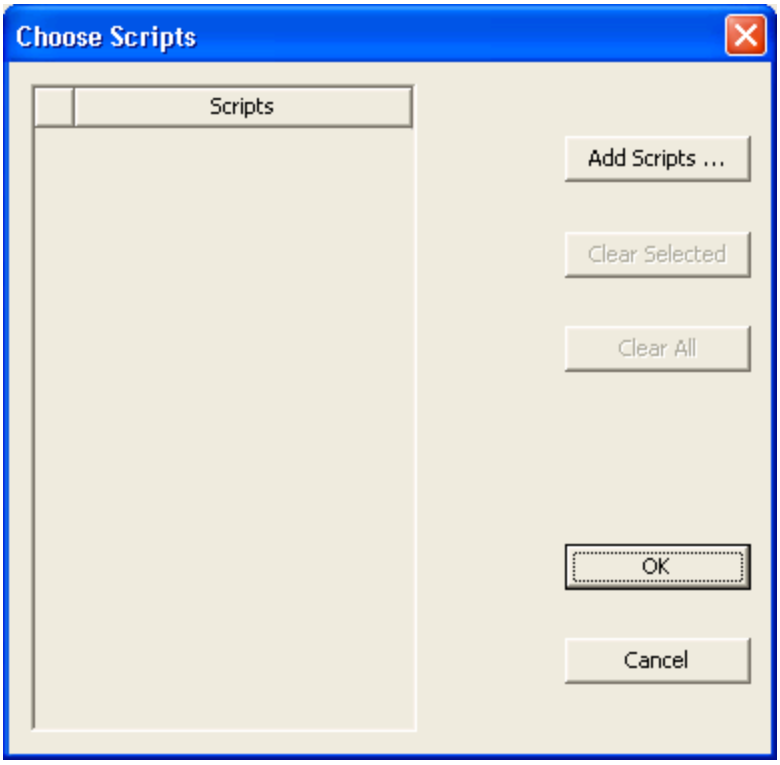
1. Select **Project** > **Event Callbacks**. The **Event Callbacks** dialog box appears.



   The **Description** column lists the type of event that will trigger the invocation of the corresponding scripts listed in the **Callback** column. When you define an **Event Callback** script, you specify one or more scripts that will be run after a particular Twin Builder event is detected.

2. If no script name appears in the **Callback** column, click the appropriate **Set script** button to define a new script. The **Choose Scripts** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

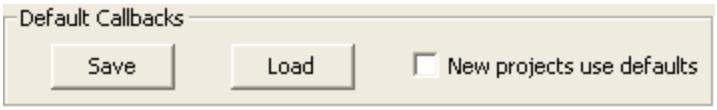


Use the **Choose Scripts** dialog box options to control the following:

- **Add Scripts** opens the **Select Definition** dialog box in which you can select scripts you want to add to the list of available callback scripts.
- **Clear Selected** clears the selected scripts from the control grid at left.
- **Clear All** clears all scripts from the control grid at left.
- **OK** closes the dialog box and implements your changes. The names of the callback scripts in the list become the button name in the **Callback** column in the **Callback Events** dialog box.
- **Cancel** closes the dialog box and cancels your changes.
- When the **Scripts** column is populated, you can reorder the listed scripts by dragging them up and down in the grid. You may also select multiple scripts in the grid and **Add Scripts** or **Clear Selected**.

3. To modify the list of scripts to be applied to a Callback Event, click the desired **Callback** button to open the **Choose Scripts** dialog box in which you can make the desired changes.
4. After defining callback scripts, the **Default Callbacks** section at the bottom left of the **Event Callbacks** dialog box lets you control the following:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- **Save** saves all of the current **Event Callback** script settings.
- **Load** loads the saved settings. Note that any current settings will be lost. As each script name is loaded, a check is done to see if that script is accessible. If it is not accessible, the script setting for that event is cleared and a warning message is displayed in the **Message Manager** pane.
- **New projects use defaults** ensures that each time you select **File** > **New**, and each time you launch Twin Builder and a new project is created, the default **Event Callback** script settings are loaded and applied to the new project. Each script name setting is used to load the corresponding script into the project. If a particular script is not found, a warning message appears in the **Message Manager** pane. If this occurs, ensure that the library containing the script is configured before creating a new project.

5. When you have finished defining one or more callback scripts, click **OK** to close the window and implement your changes, or click **Cancel** to cancel your changes.
6. The callback scripts you have defined are invoked whenever Twin Builder detects the occurrence of the associated events. For more information, see ***Event Callback Scripting*** in the Twin Builder Scripting help.

Using Legacy DSO Licenses for Parametric Analysis

In general, Twin Builder distributes Optimetrics variations using Ansys HPC licenses. In specific cases it is still possible to distribute variations using the legacy distributed solve option (DSO) license. Future releases, however, will require HPC licenses for distributed solves.

To use legacy DSO licenses for a parametric analysis in Twin Builder:

- All designs that are analyzed (by launching Analyze All on the project level) must be Twin Builder designs.
- Any design that is analyzed (by launching Analyze All on the design level or **Analyze** for a specific simulation or Optimetrics setup) must not use any coupling components such as Maxwell, HFSS, PExprt, and so on.
- At least one of the simulation tasks must be an Optimetrics setup.

The setting to turn on the use legacy DSO licenses for a parametric analysis in Twin Builder, can be found in the Twin Builder General Options (**Tools** > **Options** > **General Options** > **Twin Builder**). See ***Twin Builder Options: General Options*** for more information.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Working with PExprt Designs

Click **Schematic** > **Launch** > **PExprt** on the ribbon, or click **Twin Builder** > **Launch PExprt** to launch the PExprt desktop.

You can **Open an existing design** or **Create a new design**.

> **Note:**
>
> Refer to the help included with PExprt for detailed information on how to use that application to work with PExprt designs.

Related Topics

Opening an Existing Design using PExprt

Creating a New Design using PExprt

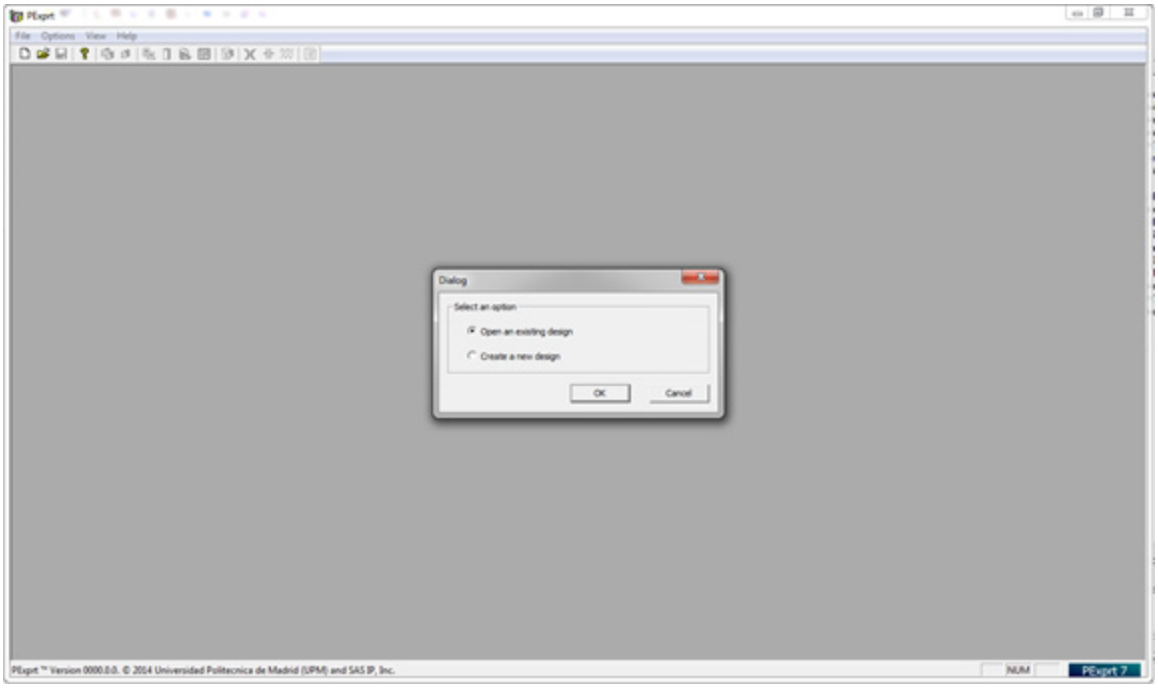
Opening an Existing Design Using PExprt

To open an existing PExprt design using PExprt:

1. Click **Schematic** > **Launch** > **PExprt** on the ribbon, or click **Twin Builder** > **Launch PExprt** to launch the PExprt desktop application.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. In PExprt, click **Open an existing design**.



3. Select the design you want to work on.
4. When you are finished with the design, click **Save**.
5. Click **File** > **Exit**.
6. Click **Yes** when prompted to exit.

Related Topics

Working with PExprt Designs

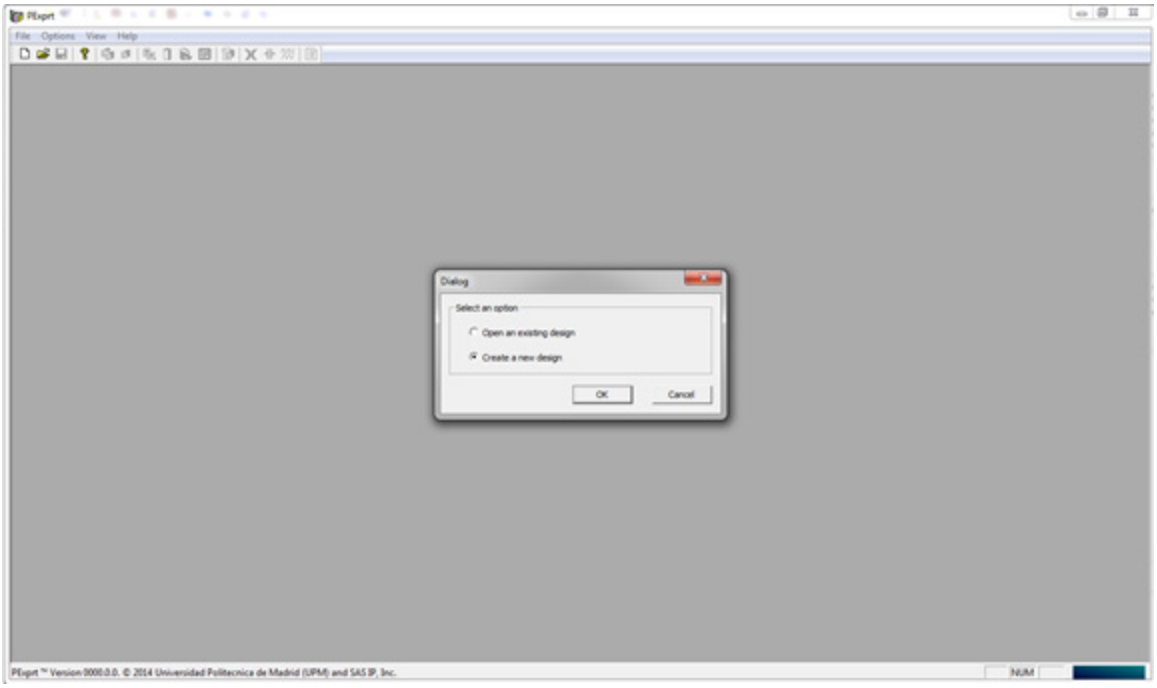
Creating a New Design using PExprt

To create a new PExprt design using PExprt:

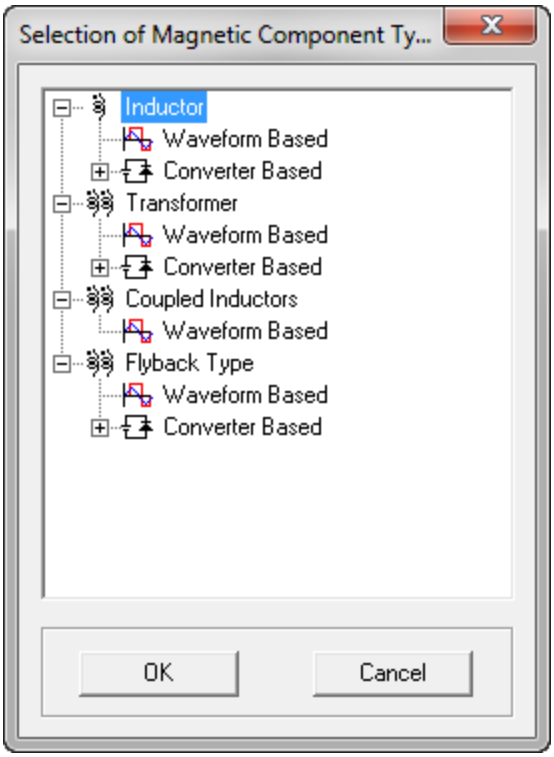
1. Click **Schematic** > **Launch** > **PExprt** on the ribbon, or click **Twin Builder** > **Launch PExprt** to launch the PExprt desktop application.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. In PExprt, choose **Create a new design**, and click **OK**.



The **Selection of Magnetic Component Types** dialog box appears.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Select a magnetic component type, and click **OK**.

See the PExprt help for information about using the PExprt software.

4. When you are finished with the design, click **Save**.
5. Click **File** > **Exit**.
6. Click **Yes** when prompted to exit.

Related Topics

Working with PExprt Designs

Using the Material Editor

A material is a set of information that defines the physical properties of a substance used as the physical foundation of a component model such as a diode. Material definitions are stored in library files with an **.amat** file extension.

Editing an Existing Material

Creating a New Material

The View/Edit Material Dialog Box

Specifying Thermal Modifiers

Editing an Existing Material

1. To edit an existing material, display its properties in the **View/Edit Material** dialog box by doing one of the following:
   a. Open the **Edit Libraries** dialog box, select the **Materials** tab, search for and select the material you want to edit, then:
      - Double-click its name.
      - Click **View/Edit Materials**.
   b. In the **Definitions**/**Materials** subfolder in the **Project** window tree for the project that contains the material you want to edit, locate the icon for the material you want to:
      - Double-click it.
      - Right-click it and select **View/EditMaterials**.
2. Edit the material properties in the **View/Edit Material** dialog box.
3. When finished editing materials, click **OK** to close the **Edit Libraries** dialog box.

Creating a New Material

1. To create a new material, do either of the following:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

a. Open the Edit Libraries dialog box, select the **Materials** tab, then click **Add Material**.

   The View/Edit Material dialog box opens.

b. Open the Edit Libraries dialog box, select the **Materials** tab, locate and select an existing material definition on which you would like to base your new definition, and click **Clone Material(s)**.

   The selected definition is copied under a new name, and the View/Edit Material dialog box opens.

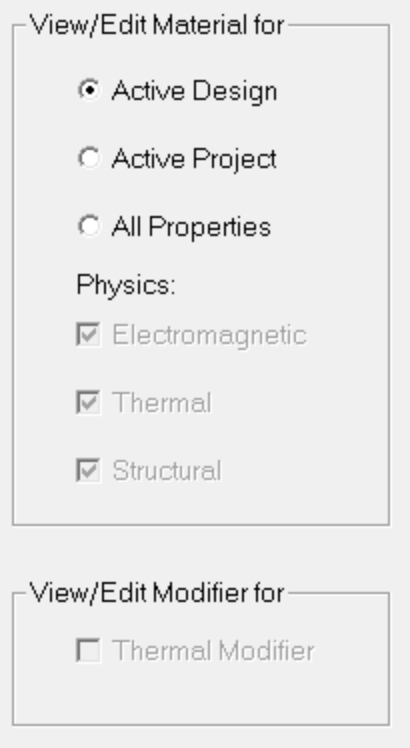
2. Set the values for the new material properties in the View/Edit Material Dialog Box.
3. When finished, click **OK** to close the Edit Libraries dialog box.

The View/Edit Material Dialog Box

The **View/Edit Material** dialog box displays and sets values for the properties of a material.

- The **View/Edit Material for** area lets you select whether to edit material properties for the active design, the active project, or all properties. The selection controls the set of properties displayed in the material properties list.
  - Select **All Properties** to change the physics options: electromagnetic, thermal, or structural. The selection chosen controls the set of properties displayed in the material properties list. In addition, **All Properties** enables the **Calculate Properties for** drop-down list for defining properties for **Permanent Magnetic** and **Permanent Electric Polarization**. See ***Calculating the Properties for a Permanent Magnet***.
  - Select **Active Project** or **All Properties** to enable the **Thermal Modifier** check box. For more information, see ***Specifying Thermal Modifiers***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- To specify or change the material name, click the **Material Name** box and enter a name.
- Click **Reset** to restore any property values changed during an editing session to the values present at the beginning of the session.
- Click **Cancel** to close this dialog box without committing your changes.
- Click **OK** to commit changes made in this dialog box and close it.

> **Note:**
>
> If you would like the changes you have made in a material to be available for use in other projects, you must export the material to a library as described in ***Editing Materials Libraries***.

Properties of the Material List

The material properties list orders properties by name, and shows the type, value, unit, and optionally the thermal modifier used for each property.

- **Name** – Displays the property name. Property names are read-only.
- **Type** – Displays and sets the property type: **Simple** or **Anisotropic**. To change a property type setting, click the **Type** cell and select the desired type. Some property types are not editable.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Value** – Displays and sets the property value. To change a property value, click one and type a new value or parameter name.
- **Units** – Displays and—where applicable—sets the unit that applies to the **Value** entry. For example, you can set the unit of magnetic saturation to **Gauss**, **uGauss**, **Tesla**, or **uTesla**. To change a unit, click it and select the desired unit.
- **Thermal Modifier** – Optional field enabled when you select the **Thermal Modifier** check box. Displays and sets a thermal modifier for the associated material property – in effect making the property temperature-dependent. To set or change a thermal modifier, click the value you want to change and choose **None** or **Edit**.

Click **Edit** to open the **Specify Thermal Quadratic Parameters** dialog box in which you can modify the parameters that determine the thermal modifier value.

Specifying Thermal Quadratic Parameters

This dialog box lets you set parameters (coefficients) that determine the value of a material property's thermal modifier used to make the property temperature-dependent. Two tabs provide for specifying both basic and advanced coefficients.

- **Basic Coefficient Set** – Set a reference temperature and unit of measure, and the values of the coefficients of the first- and second-order terms of the quadratic formula that controls the thermal modifier value.
- **Advanced Coefficient Set** – Set the upper and lower temperature limits within which the quadratic formula is valid. You can also manually override the automatic calculation of thermal modifier values to be used for temperatures outside of the upper and lower temperature limits.

Related Topics

Specifying Thermal Modifiers

Material Properties

The following are the material properties you can display and modify. Displayed properties depend on the selection made in the **View/Edit Material for** area.

> **Note:**
>
> The values of material properties may be parameterized.

- **Material Name** – Sets the material name. To specify or change the name, click the box and type the desired name.
- **Relative Permittivity** – Sets the material relative permittivity. To specify or change the value, click in the box and type the desired value.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Bulk Conductivity** – Sets the material bulk conductivity. To specify or change the value, click in the box and type the desired value.
- **Dielectric Loss Tangent** – Sets the material dielectric loss tangent. To specify or change the value, click in the box and type the desired value.
- **Magnetic Loss Tangen**t – Sets the material magnetic loss tangent. To specify or change the value, click in the box and type the desired value.
- **Electric Coercivity** – Sets the material electric coercivity. As this is a vector quantity, settings are provided for the vector magnitude, and for the X, Y, and Z components of the vector. To specify or change a value, click the box and type the desired value.
- **Magnetic Coercivity** – Sets the material magnetic coercivity. As this is a vector quantity, settings are provided for the vector magnitude and units, and for the X, Y, and Z components of the vector. To specify or change a value, click in the box and type the desired value. To change the units, click in the box and select the desired unit from the list.

- **Thermal Conductivity** – Sets the material thermal conductivity. To specify or change the value, click in the box and type the desired value.
- **Magnetic Saturation** – Sets the material magnetic saturation. To specify or change the value, click in the box and type the desired value. To change the units, click in the box and select the desired unit from the list.
- **Lande G Factor** – Sets the material Lande G factor. To specify or change the value, click in the box and type the desired value.
- **Delta H** – Sets the material delta H and associated measured frequency. To specify or change a value, click in the box and type the desired value. To change units, click in the box and select the desired unit from the list.
- **Core Loss Type** – Sets the core loss type. To specify or change the value, click in the box and select the desired value from the list.
- **Mass Density** – Sets the mass density of the material. To specify or change a value, click in the box and enter the desired value.
- **Composition** – Sets the composition of the material. To specify or change the value, click in the box and select the desired value from the list.
- **Specific Heat** – Sets the material specific heat. To specify or change a value, click in the box and type the desired value.
- **Young's Modulus** – Sets the Young's modulus value for the material. To specify or change a value, click in the box and type the desired value.
- **Poisson's Ratio** – Sets the value of Poisson's ratio for the material. To specify or change a value, click in the box and type the desired value.
- **Thermal Expansion Coefficient** – Sets the material thermal expansion coefficient. To specify or change a value, click in the box and type the desired value.

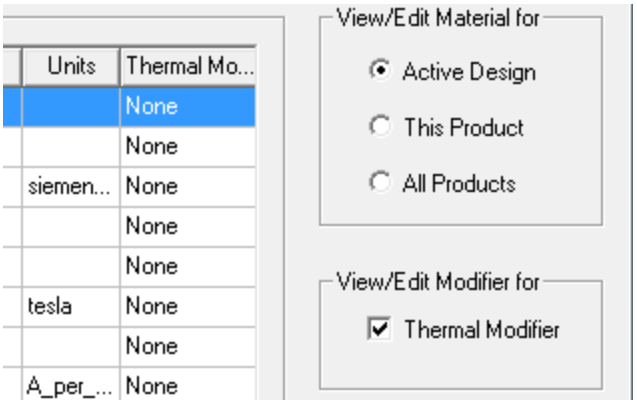
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Validate Now – Click **Validate Now** to validate the current material property values. If validation succeeds, a green check mark appears below **Validate Now**. If validation fails, a red **X** appears instead, and an error message informs you of which parameter values are invalid, and why.

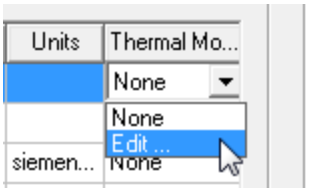
Specifying Thermal Modifiers

To specify thermal modifiers for a material:

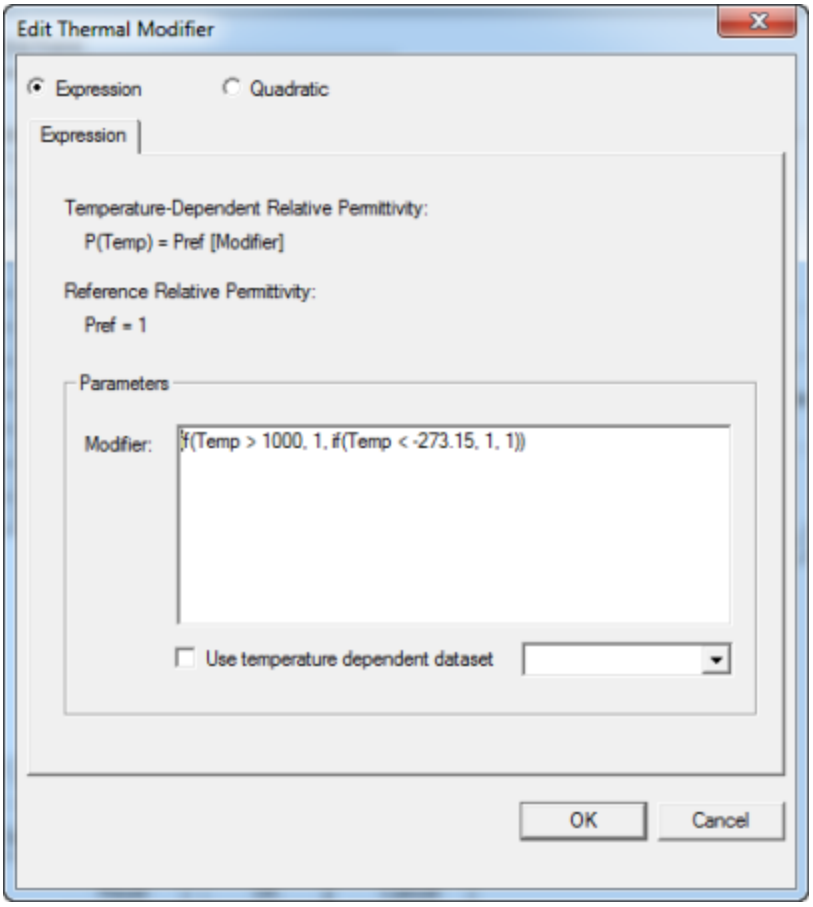
1. In the **View / Edit Material** dialog box, select the **Thermal Modifier** check box. The **Thermal Modifier** column appears in the **Properties of the Material** area.



2. Select **Edit...** from the drop-down list. The **Edit Thermal Modifier** dialog box appears.

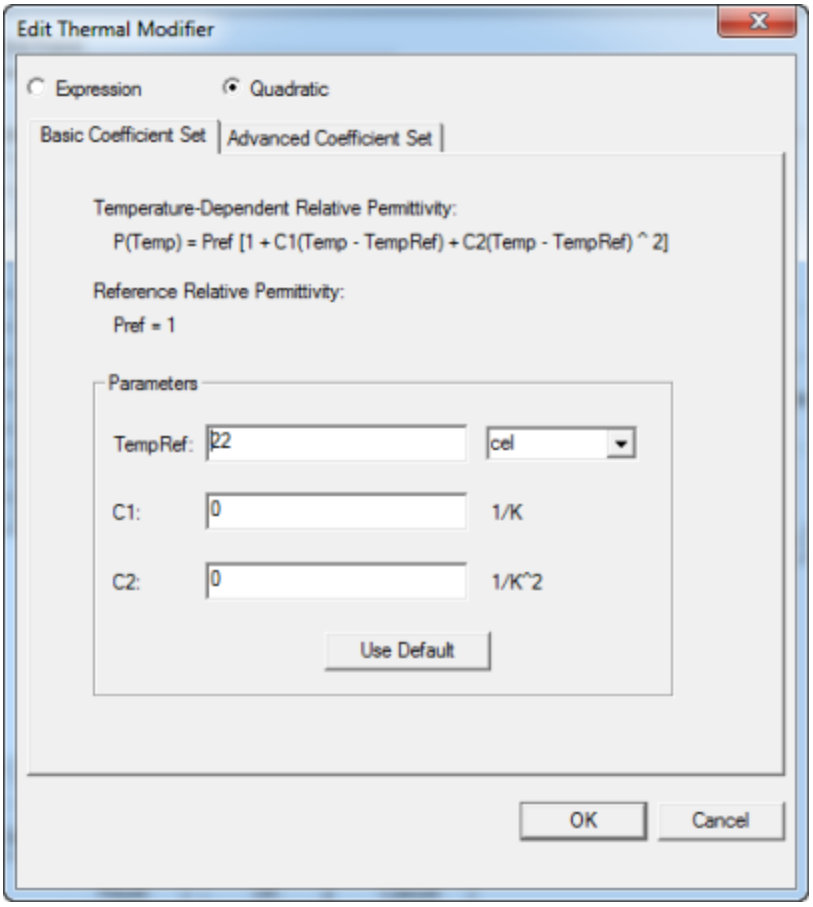


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

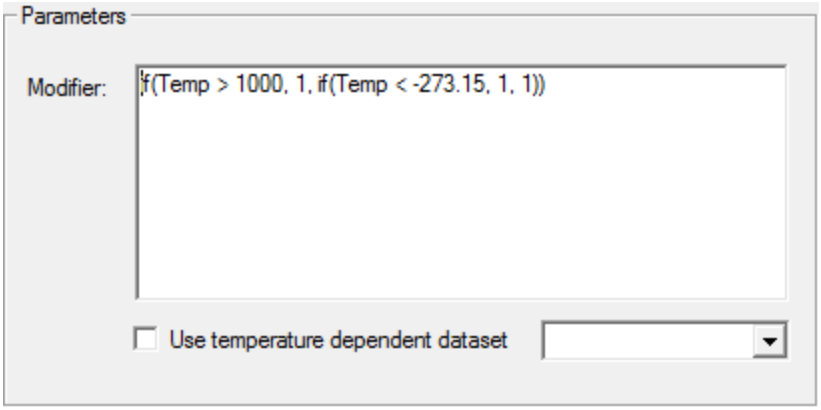


3. Select the **Expression** radio button to display the **Parameters Modifier** text field (as shown above) or the **Quadratic** radio button to display the tabs for **Basic Coefficient Set** and **Advanced Coefficient Set** (as shown below).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



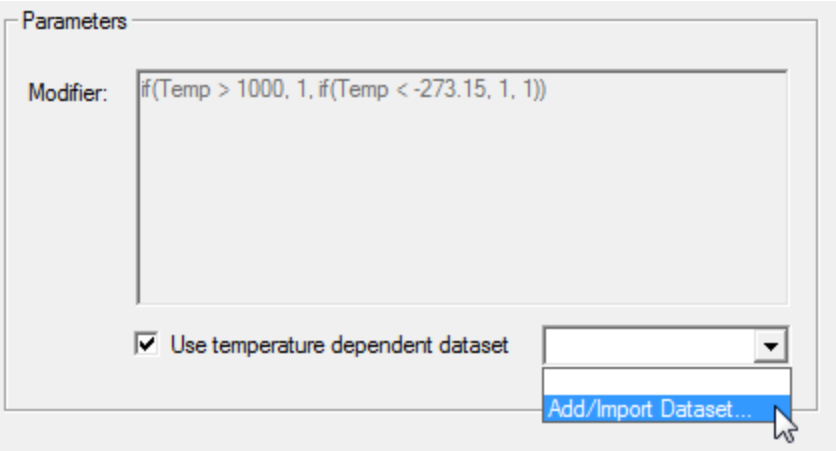
- With **Expression** selected, you can write an equation for a thermal modifier in the Parameters **Modifier** text field.



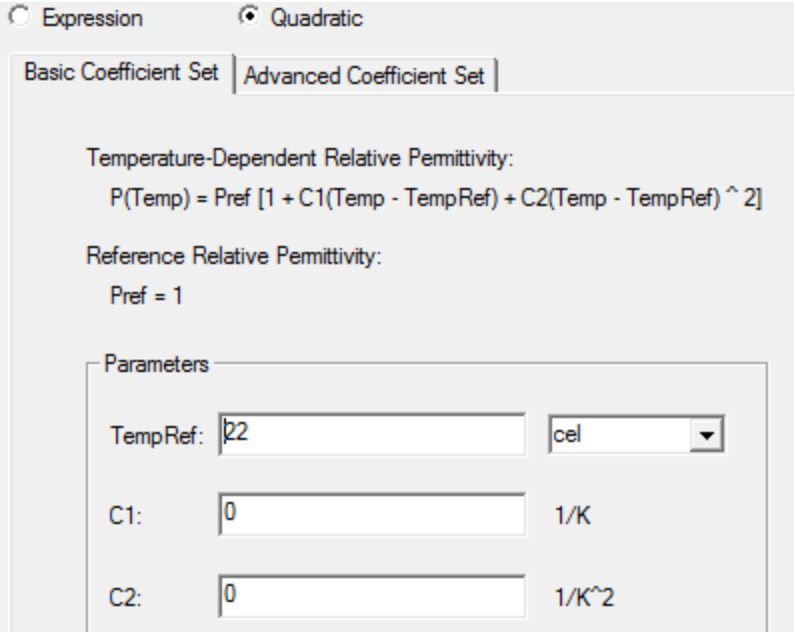
The expression whether to use Celsius or Kelvin is totally problem dependent. If a material thermal coefficient is defined as alpha/c_deg, then it is Celsius. On the other hand, if a material thermal coefficient is alpha/k_deg, then it is Kelvin.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Select the **Use temperature dependent dataset** check box to disable the **Modifier** text field. Select **Add/Import Dataset** from the drop-down list to define the thermal modifier as a dataset.



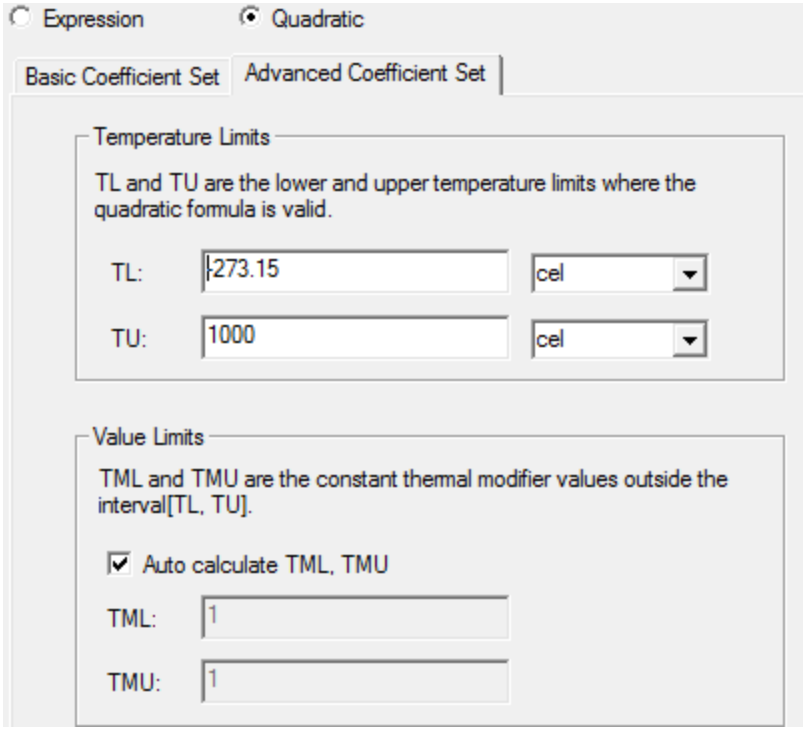
- Select the **Quadratic** radio button and click the **Basic Coefficient** tab. You can edit fields for the TempRef and units, and fields for C1 and C2 with this equation:
- P(Temp) = Pref[1+ C1(Temp - TempRef) + C2(Temp - TempRef)^2]
  where **TempRef** is 22 cel by default and **Pref** is the reference relative permittivity.



> **Note:**
>
> The coefficients, C1 and C2, should be negative to yield physical results.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Select the **Quadratic** radio button and click the **Advanced Coefficient Set** tab. You can edit fields for lower and upper temperature limits (TL and TU respectively) and select their units from the drop-down list.



  You can edit the constant value limit for the thermal modifier values outside the limits. By default, these are calculated. Clear the **Auto Calculate TML, TMU** check box to specify new values for thermal modifier lower (TML) and thermal modifier upper (TMU).

4. Click **OK** to accept the edits and return to the View/ Edit materials dialog box.

Related Topics

Adding Datasets

View/ Edit materials dialog

Specifying Thermal Quadratic Parameters

Calculating the Properties for a Permanent Magnet

From the **View/Edit Materials** dialog box:

1. Set **View/Edit Material for** to **Active Project** or **All Properties** to enable the **Calculate Properties for** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Select **Permanent Magnet** from the drop-down list to display the **Properties for Permanent Magnet** dialog box. This dialog box contains the following fields.

Mu	Provide a value for relative permeability.
Hc	Coercive field force H_c in the units specified. Provide a value and select units from the drop-down list.
Br/Mp	Select this to enable the **Br** and **Mp** radio buttons.
Br	Residual flux density B_r, in Tesla. If enabled, provide a value and select units from the drop-down list.
Mp	Permanent Magnetization M_p, in A/m. If enabled, provide a value and select units from the drop-down list.

3. Click **OK** to close the dialog box and return to the **View/Edit Material** dialog box.

The values for **Relative Permeability** and **Magnitude** under **Magnetic Coercivity** become the new default values.

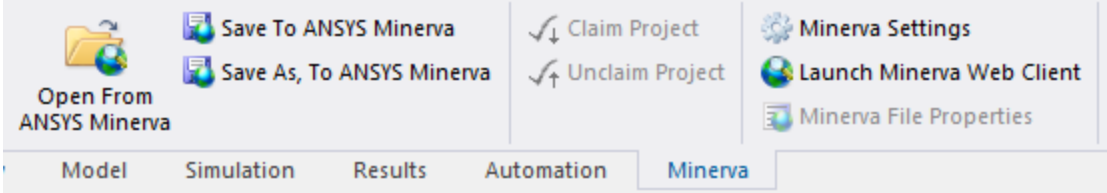
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5 - Minerva Remote Storage Environment

Ansys Minerva is a remote storage environment where you can store Ansys project archive files. You can collaborate with other users by downloading a project, making changes, then uploading the project. Minerva has its own HTML web interface which has robust functionality.

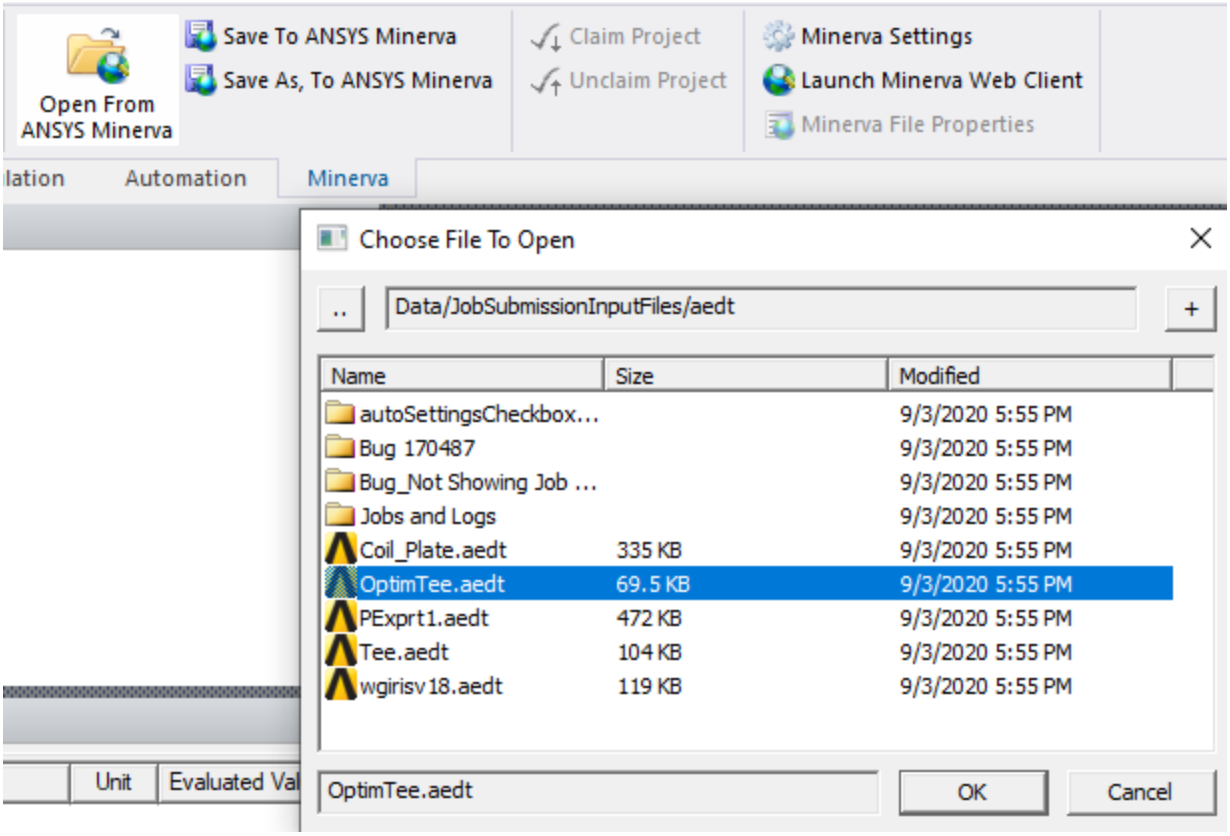
When you first log in via the desktop, or to the Web Client, you must provide a Minerva URL, a database name, a user name, and a password.

Minerva includes a simple client integrated into the Electronics Desktop as well as a means to launch the Minerva Web client. The simple client has the following functionality:



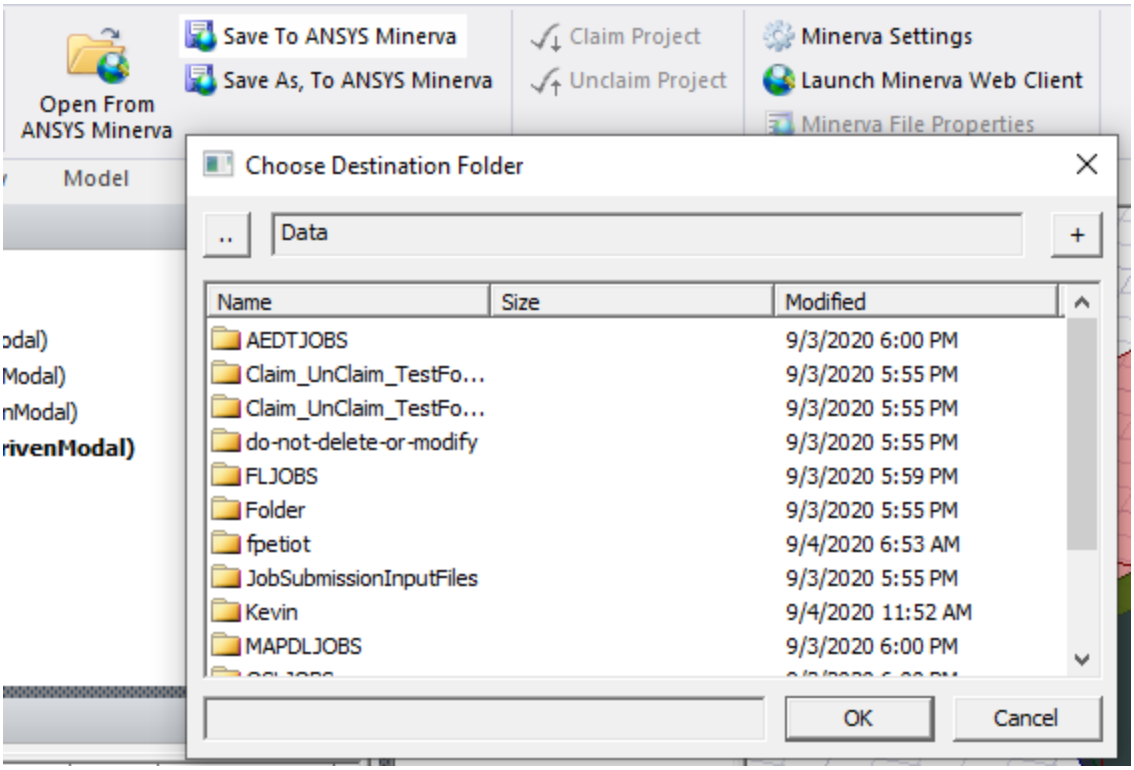
Open From Ansys Minerva: opens the Choose File to Open browser that lets you select and download an archive file, extract it, and open it. Click **[..]** to move up one level in the file hierarchy. Click **[+]** to create a new folder at the current level.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



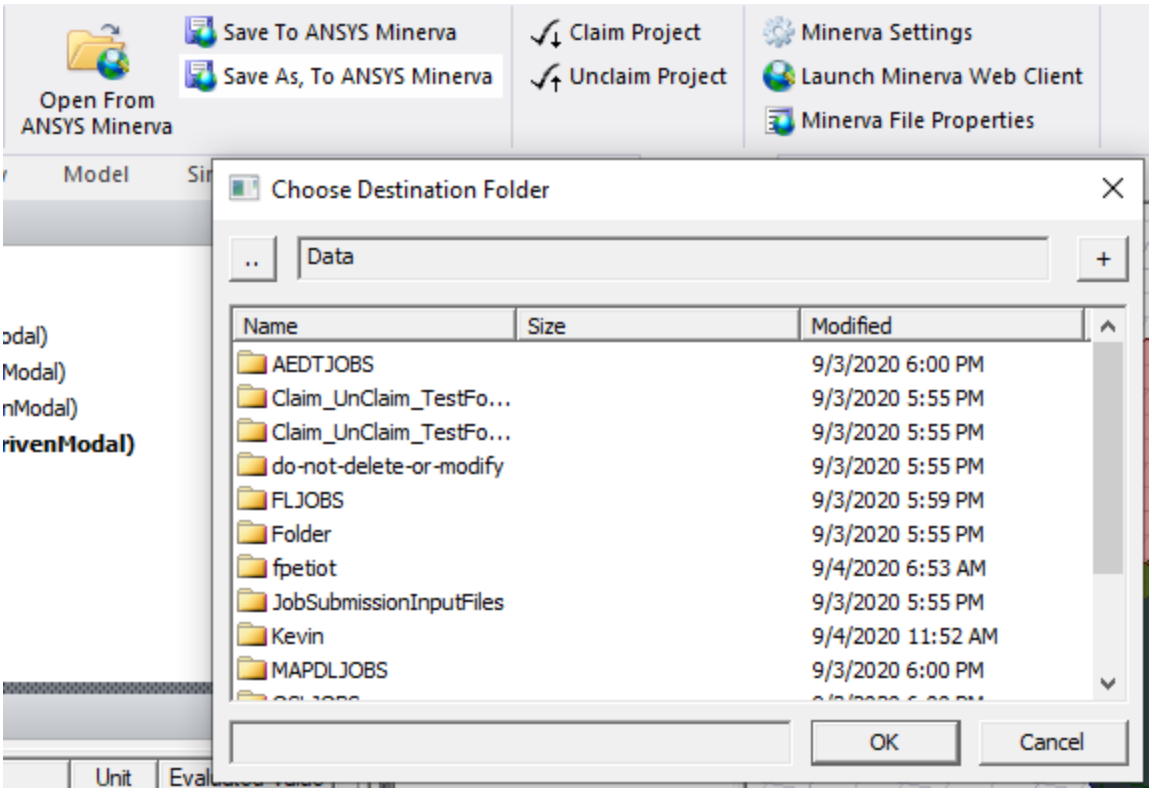
Save To ANSYS, Minerva: archives the current project, and saves it to a remote Minerva system.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Save As, To Ansys Minerva: prompt for remote file, archive project, save to remote Minerva system.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

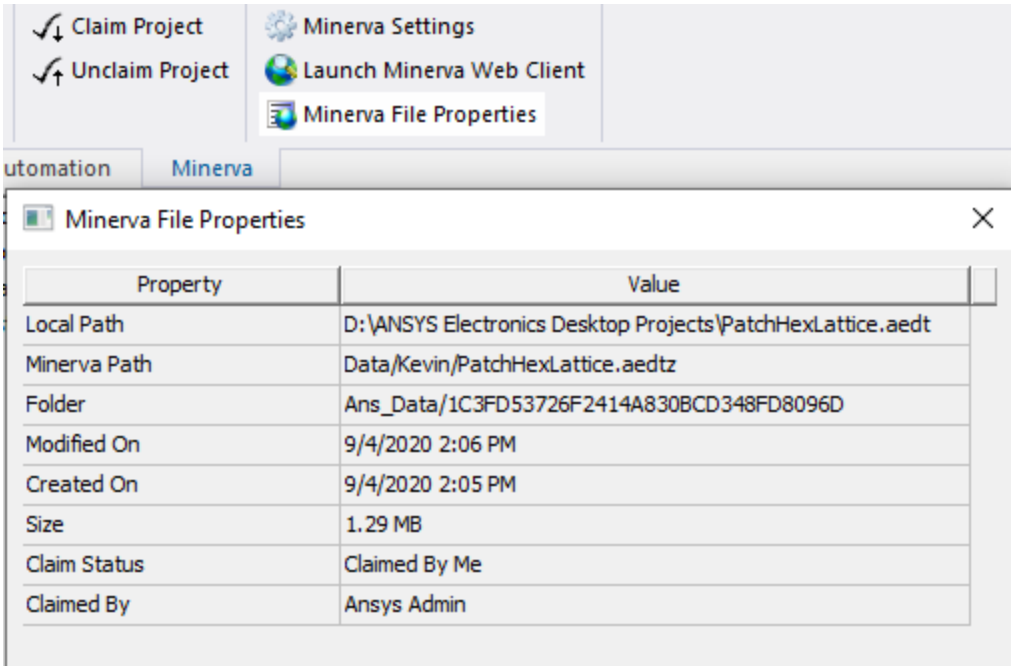


Claim Project: claims file so it cannot be modified by other users.

Unclaim Project: releases claimed file.

Minerva File properties: opens a dialog box letting you view remote file properties.

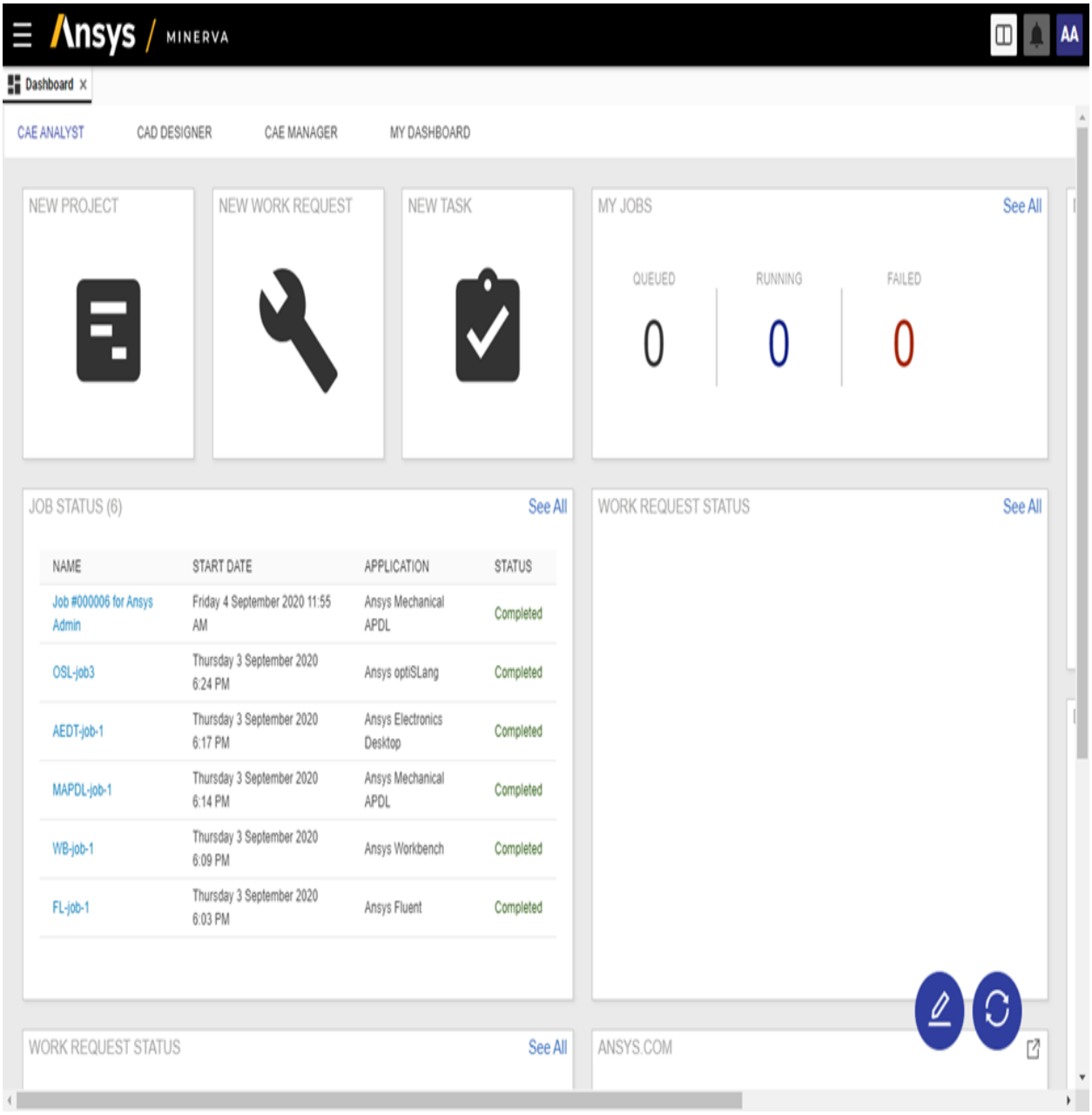
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Launch Minerva Web Client:

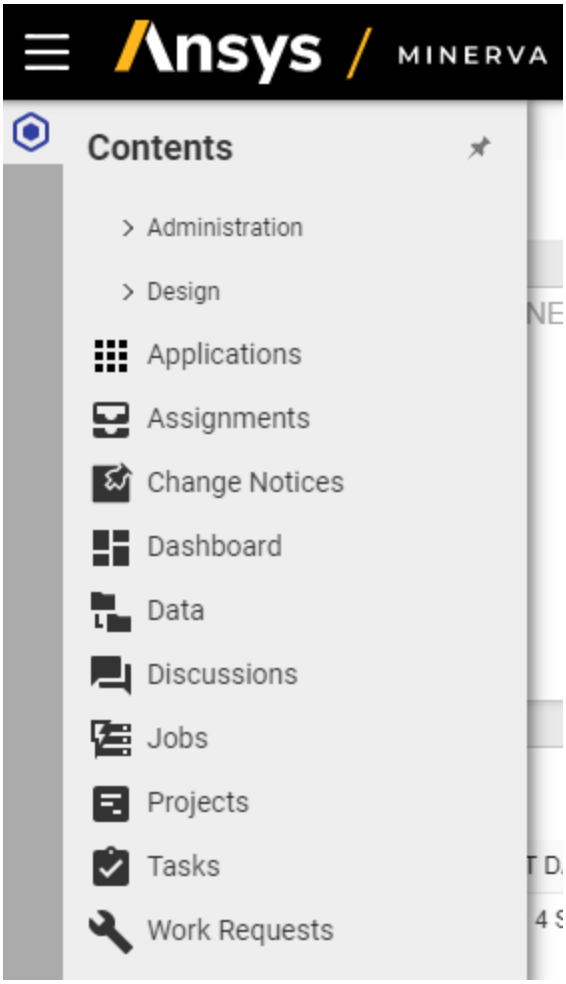
This opens a dialog calling for you to specify provide login information. It attempts to open a web browser window, so your default browser must be compliant.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



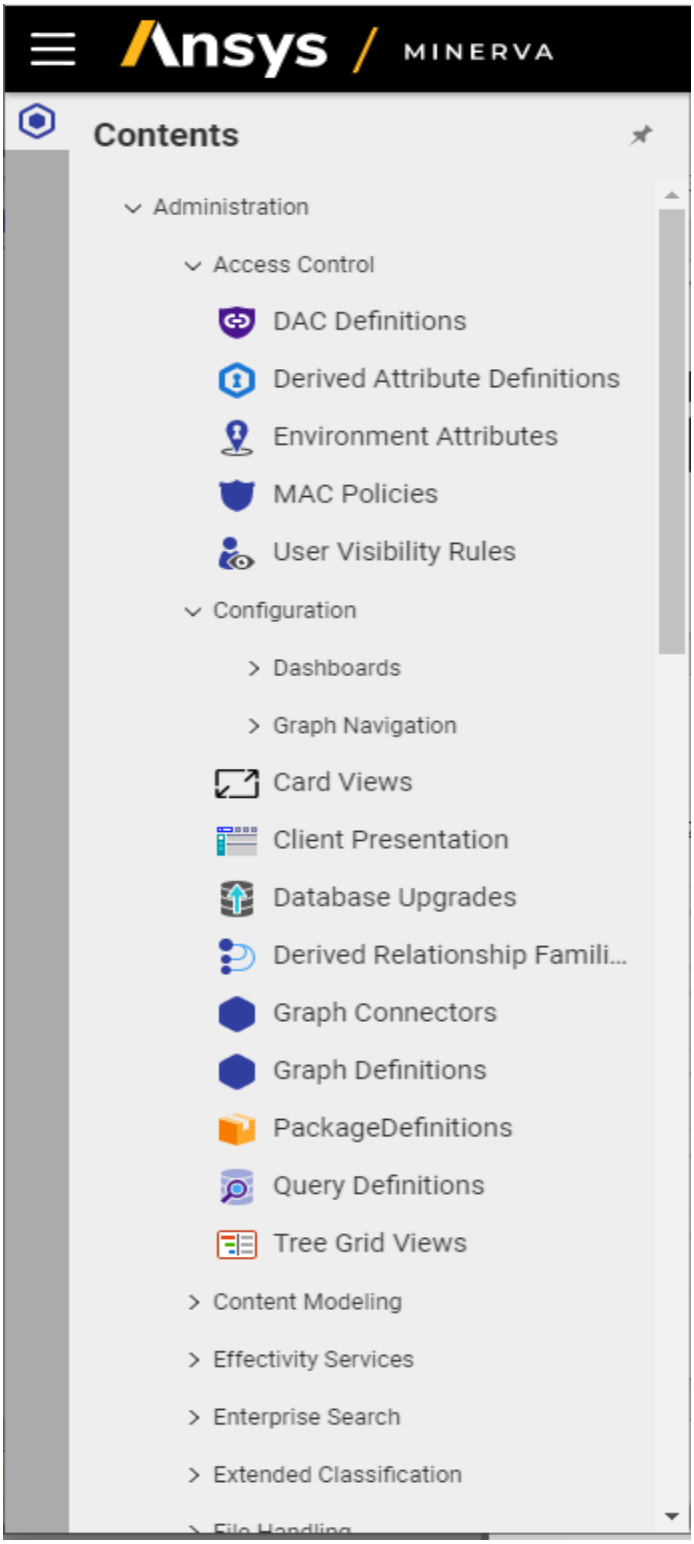
Click the triple bar icon left of the Ansys logo to access the more robust Minerva Web Client functionality.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The angle icons for Administration open up further functionality.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6 - Working with Libraries

This section describes how to configure and manage Twin Builder external library resources. Use these topics to create and edit elements within the libraries, and to set up circuits within Twin Builder projects to access particular libraries.

Introduction to Components and Component Libraries

Managing Library Contents

Using the Component Editor

Using the Symbol Editor

Using the Script Editor

Using the Password Manager

Introduction: Components and Component Libraries

These topics cover creating, editing, and managing Twin Builder component definitions and component libraries.

Component Definition Overview

Library Concepts

Translating Legacy Libraries

Ansys Customer Portal Extension Library

Component Definition Overview

In Twin Builder, a component is a collection of information that defines a signal-processing function or source, or a data container or data channel, for inclusion in a schematic design. A component may represent:

- A calculated model, such as a resistor, source, transmission line, or transistor, for which a mathematical model exists in a Twin Builder simulation engine.
- An interpretive user-defined model (IUDM).
- A data container, such as a device model data component or an *N*-port black box component that contains network information in spreadsheet form.
- A data input channel that introduces external data into an analysis, such as an *N*-port black box component that refers to network data in an external file, or a SUBCKT component that refers to a subcircuit netlist fragment in an external file.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

A component definition also references separately-stored data for its schematic symbol. These component dependencies are stored as library objects in their own additional files.

See *Using the Component Editor* for information on how to create and modify library components.

Parameter Data and Dependent Data

The information stored in any component includes parameter data and dependent data. The parameter data describes the internal electrical characteristics of the component, and is stored with the component definition in the component library **.aclb** file. The dependent data describes associated physical characteristics, like the schematic symbol and is contained in an external library file referenced by the component definition.

Materials

Material information defines the physical properties of a substance — the physical foundation of a component model. Material definitions are stored in library files with the extension **.amat**. See *Using the Material Editor* for information on how to create and modify materials.

Scripts

The **Script Editor** supports the viewing, modification, saving, and exporting of script file information for footprint elements and graphical primitives. After modifying a script, the information can be saved, used to render a footprint, or exported. See *Using the Script Editor* for information on how to create and modify scripts.

Components

The **Component Editor** supports the viewing, modification, saving, and exporting of information for component elements and graphical primitives. After modifying a component, the information can be saved, used to render a component, or exported. See *Using the Component Editor* for information on how to create and modify components.

Models

Twin Builder editors support the viewing, modification, saving, and exporting of information for C-Models, SML models, SPICE models, and VHDL-AMS models. See *Using the C-Model Editor*, *Using the VHDL Model Editor*, *Using the SPICE Model Editor*, or *Using the SML Model Editor* for information on how to create and modify these models.

Packages

Packages are collections of re-usable declarations and definitions such as types, constants, functions, procedures, and natures. Standardized packages from IEEE (such as **math_real** and **textio**) and proposed packages from IEEE (such as **electrical_systems** and **thermal_systems**) are available in the Ieee (**Ieee.apkg**) and Std (**Std.apkg**) libraries. See *Using the Package Editor* for information on how to create and modify packages.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Symbols

A symbol is graphical representation of a component in a Twin Builder schematic. Symbol definitions are stored in library files with the extension **.aslb**. See *Using the Symbol Editor* for information on how to create and modify symbols.

Twin Builder Netlist

A Twin Builder netlist is an SML textual representation of a design that describes the design's model instances, connections, ports, and subdesigns, as well as instructions for data recording and analysis type information. Each component instance in a schematic provides an entry for the netlist based on the model or netlist line (using SML syntax and netlist string property expansion rules) chosen for that particular instance. The Twin Builder UI creates a netlist from the schematic design, prepares a binary version of the netlist (SMT), then provides this binary file to the simulator to perform the simulation.

Library Concepts

In Twin Builder, library component definitions are intended to be accessed once per component, when you place the first instance of a component onto a schematic. Once a component is placed, the definition for the component transfers from the library to the project file.

Editing a component definition and updating instances is then controlled from the *project definition*, which is listed in the **Components** folder in the Project tree.

In other words, you do not edit component definitions in the library, but in the project. If you want your edits to be reflected in the library definition, for use in another design, you must export the edited components to the library.

Related Topics

Library Directories

Specifying the Location of Library Files

Managing Library Files

Component and Library Search Precedence

Updating Project Definitions from Library Definitions

Model and Library Synchronization

Removing Definitions from a Project

The VHDL-AMS "work" Library in Twin Builder

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Library Directories

The stock library files that ship with Twin Builder are stored under the **syslib** directory. These libraries are intended to be read-only and cannot be modified.

In addition to the system libraries, Twin Builder recognizes two user-configurable library structures: the **User Library** and the **Personal Library**. These are used to add foundry support, user-defined models, and any custom or proprietary sets of components or simulation models. Customarily, **userlib** is a network repository for proprietary or corporate definitions available to all seats in an enterprise. **PersonalLib** contains project- and circuit-specific libraries as needed by individual designs.

A root library directory is set up at installation. If none is specified, the default is the root Twin Builder directory.

The VHDL-AMS *work* Library in Twin Builder

The working library for a VHDL-AMS model in Twin Builder is always the library that the model came from. Consequently:

- The **work** library is identical to the **project** library for all models that were imported from a text file or created manually in the VHDL-AMS model editor.
- The **work** library is *xxx* for all models that were loaded from library *xxx*.

  For example, if a model **x1** that was loaded from library **L1** loads two models: **work.x2** and **L2.x3**; and model **x3** from library **L2** also loads a model **work.x2**, then **x2** referred in **L1.x1** would be loaded from library **L1**, but **x2** referred in **L2.x3** would be loaded from library **L2**.
- If a model refers to other models using **work** and this set of models will be exported to different libraries, then **work** should be changed to specific library names to ensure that the referenced models are found.
- All models referred to using **work** must be exported to the same library.

Related Topics

About the VHDL-AMS *working* Library in Twin Builder

VHDL-AMS Model Editor

VHDL-AMS Models in Twin Builder

Specifying the Location of Library Files

You can relocate library directories to a new valid library location. To specify library directories:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Click **Tools** > **Options** > **General Options**. The **Options** dialog box appears.
2. Click **General** > **Directories**. Type your folder locations in the **Temp**, **SysLib**, **UserLib**, and **PersonalLib** fields, or use the button to locate them.
3. Click **OK**.

To return the library directory specification to its default value, click **Reset Library Directories**.

Note that the personal library (**PersonalLib**) folder is located at *ProjectDirectory***PersonalLib**, where *ProjectDirectory* is the location you specified for your Twin Builder projects.

Managing Library Aliases

You can configure the library manager to consider alternate paths or alternate files for a particular library request if the initial request is unsuccessful. These alternate paths and files are called library aliases.

When a definition is requested, Twin Builder looks first in the project with the library name provided, then with the permutations generated from the prefixes and aliases set in the **Library Directory Aliases** dialog box.

Click and to determine which prefixes and aliases are tried before others. Prefixes always come before aliases. A library found with a prefix or alias is recorded by the project and the same request will always obtain the same library.

1. To manage library aliases, click **Tools** > **Library Tools** > **Manage Twin Builder Aliases**. The **Library Directory Aliases** dialog box appears.
2. Add the desired prefixes and aliases.
   a. Click **Add Prefix** to add the new path prefix to the list. >Each **Prefix** is prepended to the original library name to create an alternate search string. The library manager performs string substitutions with each **Alias** to create additional potential library paths. If the library is not found in the project, the library manager searches **personallib**, then **userlib**, and finally **syslib** using the original and all alternate library names in order.
   b. Click **/** in the **Prefixes** panel to browse and choose an alternate directory path to be prepended to the original.
   c. Click **Add Alias** to add the new alias to the list.

   Click **…** and **/** in the **Aliases** panel to browse and select file names and directories. You can also type the information directly into the text boxes.
3. The **Save as default** check box saves the currently defined prefixes and aliases so they become available when Twin Builder starts.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Components and Library Search Precedence

When you place a new component in a design, Twin Builder searches in the current project for a definition with the corresponding name. If an appropriate definition exists in the project, that definition is used to satisfy the placement *regardless of whether additional instances, even more recent ones, may exist in external libraries.*

If Twin Builder cannot locate the required definitions in the project, it searches in personal (**PersonalLib**), user (**userlib**), and system (**syslib**) library files, *in that order*. Once a definition of the correct name is located, the search ends, and that definition is used to satisfy the placement request.

> **Note:**
>
> - Once a definition is used, it is transferred to the current project, and remains in the project unless you remove it. See Removing Definitions from a Project for details. To see the definitions included in a project, expand its **Definition** folders and subfolders in the Project tree.
> - Modifying component and dependency definitions in libraries, or installing libraries with modified component and dependency definitions, does not update those definitions in projects that contain them. To update project definitions from library definitions, see Updating Project Definitions from Library Definitions.

Updating Project Definitions from Library Definitions

Once a library component, symbol, package, model, script, or material is placed into a design, the parent definition resides in the project, and the link to the source library is broken. **Update Definitions** is designed to update a project's definitions from existing libraries. For example, if a project is old and there are newer definitions available in the libraries, executing this command will bring the newer definitions into the project from the libraries.

To update a project definition with a definition from a library file:

1. Select the project you want to update in the Project tree.
2. Click **Tools** > **Project Tools** > **Update Definitions**.

   The **Update Definitions** dialog box opens, listing any library definitions that have changed since they were added to the project, the library type, and their original library locations.

3. Use the dialog box to select the items to update, and click **Update** to finish.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Model and Library Synchronization

In Twin Builder, model definitions can be used by other models in their definitions. Consequently, changes made to a model definition affect all models that are dependent upon (use) that model. If you update models that are interdependent after making changes to them, there needs to be some process to keep them in sync. Synchronization is the process by which Twin Builder reconciles these changes. Synchronization can be performed on project definitions and on library definitions.

Related Topics

Synchronizing Project Definitions

Automatic Library Synchronization

On-Demand Library Synchronization

Importing Simulation Models

Synchronizing Project Definitions

Models used in Twin Builder projects may reference other model definitions in their definitions. When you edit and update a project model definition that is used by other project models, the project synchronization process recompiles all of the dependent models. Such compilations can result in compilation errors if the updated model's interface has been modified. The following example illustrates this process and how to correct such errors.

In the following example, assume there is a VHDL-AMS model library containing several models including:

- A thermal model that defines a terminal named **m**;
- A fuse model that uses the thermal model, and references the fuse model terminal **m** in its definition. The fuse model is therefore dependent on the thermal model.
- Both the fuse and thermal models are present in the current project's **Project Manager** > **Project** > **Definitions** > **Models** folder.

1. In the **Project Manager** pane, the thermal model is opened in the VHDL-AMS model editor and terminal **m** is changed to **m1**.
2. **Update Project** is then performed on the thermal model in the model editor.

   Synchronization occurs on the fuse model, as well as on any other models in the current project that may depend upon the thermal model. During this process, the fuse model is recompiled. However, compilation will fail in this instance due to the terminal name mismatch (**m** vs. **m1**). Error messages are displayed in the **Message Manager** pane, giving the line numbers in the fuse model where the errors occurred.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. You could open the fuse model in the VHDL-AMS model editor, make the needed corrections (change **m** to **m1**), and perform **Update Project** on the fuse model to complete the process.

   Alternatively (perhaps because you mistakenly changed the thermal model), you could also correct the errors by editing the thermal model so that the terminal names agree.

Automatic Library Synchronization

Automatic library synchronization after model export in Twin Builder is enabled by default in the **Twin Builder Options** dialog box, allowing library models dependent on exported models to be synchronized.

The following example illustrates this process. In this example, assume there is a VHDL-AMS model library named **sync_demo.asmd** containing several models including:

- A thermal model that defines a terminal named **m**.
- A fuse model that uses the thermal model, and references the fuse model terminal **m** in its definition. The fuse model is therefore dependent on the thermal model.

1. With a project open, the thermal model is opened for editing in the VHDL-AMS model editor and terminal **m** is changed to **m1**.
2. **Update Project** is performed on the thermal model in the model editor.
3. The thermal model is exported from the project to the **sync_demo** library using **Export to Library**.

   Automatic synchronization occurs on the fuse model - and on any other models (in any of the libraries) that may depend upon the thermal model. During this process, the fuse model is recompiled. However, compilation will fail in this instance due to the terminal name mismatch (**m** vs. **m1**). Error messages are displayed in the **Message Manager** pane giving the line numbers in the fuse model where the errors occurred.

4. Open the fuse model in the VHDL-AMS model editor, make the needed corrections (change **m** to **m1**), and perform **Update Project** on the fuse model to complete the process.

On-Demand Library Synchronization

Automatic library synchronization after export in Twin Builder is enabled by default. If, however, **Synchronize all libraries after export** is disabled in the **Twin Builder Options** dialog box, automatic library synchronization does not occur. Instead, Twin Builder maintains a list of all exported model names for the current Twin Builder session. Click **Tools** > **Library Tools** > **Synchronize All** to synchronize all libraries on demand.

If you close Twin Builder without performing **Tools** > **Library Tools** > **Synchronize All**, the listing of exported model names is lost and manual library synchronization must be performed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Translating Legacy Libraries

Starting with Simplorer 2016 (Ansys Electromagnetics Suite 17.0), Simplorer no longer supports Simplorer 7.0 libraries. If you want to port a library from Simplorer 7.0, you must first use Simplorer Release 16.2 or earlier to translate it, then bring it into the current release of Twin Builder.

Editing Translated Components, Symbols, and Models

There are two ways you can edit the newly translated components or models:

- Add a schematic and place components from the newly translated library on it. Then use the **Definitions** folders in the **Project Manager** pane to edit the component, symbol or model. Right-click the desired element and select **Edit** *element_name*.
- Similarly, in the library translation project, right-click the **Definitions** > **Models** folder and select **Edit Library**. Choose the new library and edit the desired definitions.

Ansys Customer Portal Extension Library

The Ansys Customer Portal Extension Library is provided for the distribution of extensions and scripts to Ansys customers. Materials posted in this area have been created by Ansys or Ansys Channel Partners exclusively for use with Ansys products. Among the available libraries is a **Twin Builder Device Models Library** containing several libraries of device models for Twin Builder. Each library includes detailed documentation and an application example.

To access and download these libraries:

1. Go to the Ansys web site (http://www.ansys.com).
2. Click Customer Portal.

   You can also click https://support.ansys.com/portal/site/AnsysCustomerPortal to open the Log In window.

3. In the Customer Portal, click **Downloads** > **Ansys ACT Application Store**.
4. Select the library for downloading.

Related Topics

Library Directories

Managing Library Contents

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Managing Library Contents

Select **Tools** > **Edit Libraries** to display the contents of selected library files and to copy, modify, and export components or other library objects to user/personal libraries.

Twin Builder libraries include the following types:

- Materials
- Scripts
- Components
- Models
- Packages
- Symbols

1. Right-click the **Project Manager** > **Definitions** folder and select **Edit All Libraries** to open all libraries for editing.
2. The **Edit Libraries** dialog box appears. Choose the tab for the library type you want to edit.
3. To select an individual library, click **Tools** > **Edit Libraries** and choose the library to view.

   Alternately you may also right-click any of the **Project Manager** > **Definitions** folders and select **Edit Library**.

   After choosing a library, the **Edit Libraries** dialog box appears with a tab open to the library you selected.

   The **Edit Libraries** dialog box also filters objects available for selection based on a number of object attributes, including:

- Name
- Property
- Model type

In addition, each **Edit Libraries** dialog box includes check boxes that let you **Show Project definitions** or override filtering and **Show all libraries**.

> **Note:**
>
> When you modify a library component, the modified definition transfers to the components available to the current project. In effect, you have “checked out” the component to modify it, and it is now in the project. The original library definition is intact. To update the library definition, you must perform the extra step of exporting the modified component from the project to the library.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding Definitions to Libraries via the Project Manager

To add a definition to a library, right-click a **Project Manager** > **Definitions** folder and select **Add Definition**.

- For components, the **Edit Component** dialog box displays in which you can define a component. See *Editing Component Libraries* for additional information.
- For symbols, the **Definition Name** dialog box appears. Enter a name for the new symbol, then click **OK** to open the Symbol Editor where you can define the new symbol. See *Editing Symbol Libraries* for additional information.
- For models, the **Add Model** dialog box displays. See *Editing Model Libraries* for additional information.
- For packages, the **Add Package** dialog box displays. See *Editing Packages Libraries* for additional information.
- For materials, the **View/Edit Material** dialog box displays. See *Editing Materials Libraries* for additional information.
- For scripts, the **Get Name** dialog box appears. Enter a name for the new script, then click **OK** to open the Script Editor where you can define the new script. See *Editing Scripts Libraries* for additional information.

Exporting Definitions to User Libraries via the Project Manager

Follow this procedure to export a component, symbol, model, or package definition to a user library.

1. Right-click the desired object in the **Component**, **Symbol**, **Model**, or **Package** folder in the **Project Manager** > **Definitions** directory, and select **Export to Library**. The **Export to user library** dialog box appears.
2. Navigate to the location where you want to save the exported library.
3. Click **PersonalLib** or **UserLib** to jump to those locations. If present, select the **Encrypt** check box to encrypt the library.
4. Enter a name for the library in the **File Name** field.
5. Click **OK** to save the library and close the dialog box.
6. The **Export Hierarchy** dialog box opens. See ***Exporting Hierarchical Components*** for detailed information on working with this dialog box.

Exporting and Importing Definition Archives

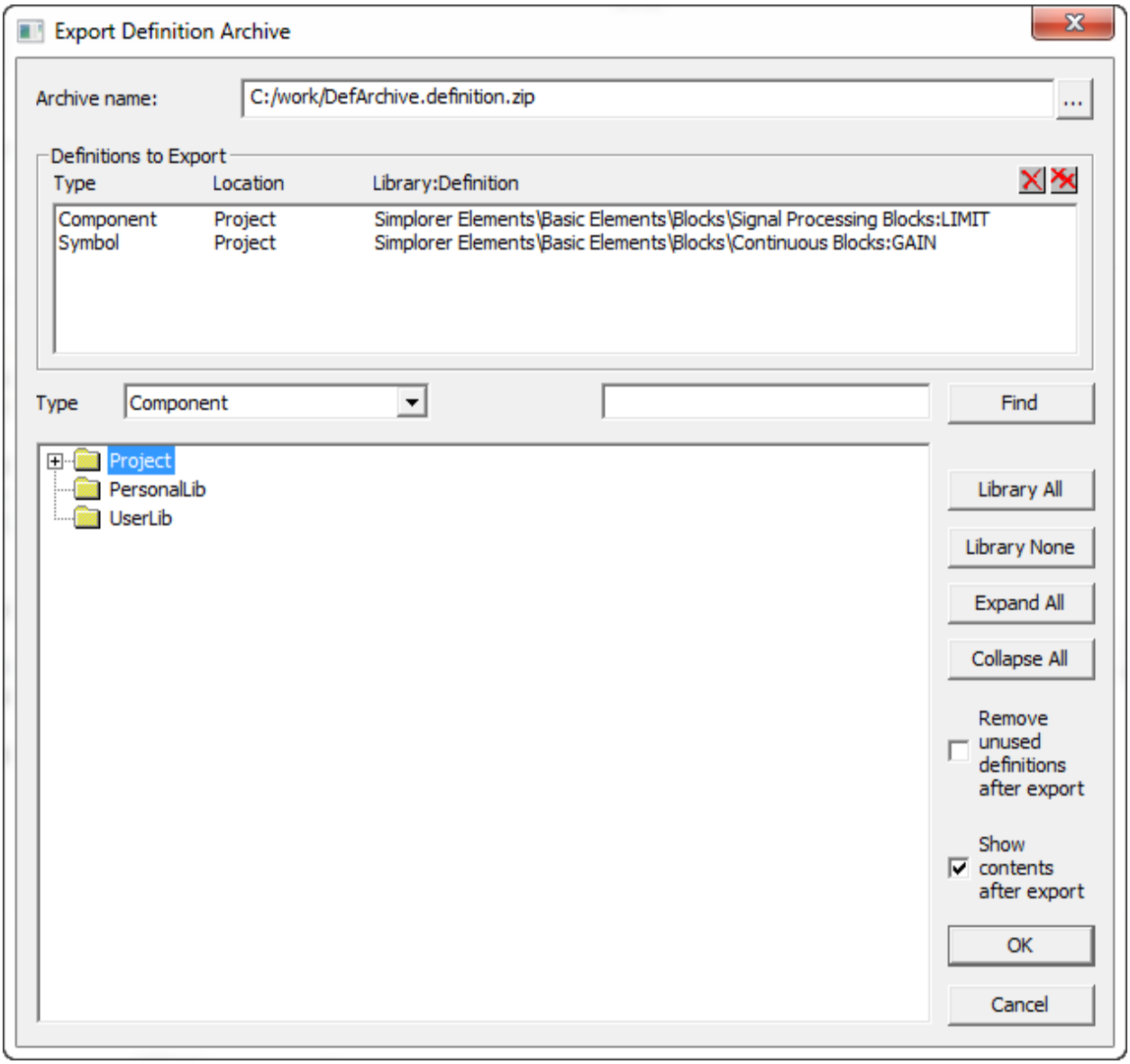
Use the **Export Definition Archive** and **Import Definition Archive** dialog boxes to select individual component, model, or other library definitions—or even entire libraries—to package and send to another user. Referenced definitions, data files and DLLs are all included. All

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

selected definitions—either those in an entire library or selected separately—are written to one set of libraries (for example, a component library and associated symbol and model libraries).

Exporting Definition Archives

Select **Tools** > **Library Tools** > **Export Definition Archive** to open the **Export Definition Archive** dialog box.



- To export a definition archive, choose an archive name. The archive name must have a **.definition.zip** file extension.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> A **.definition.zip** archive is not readable with utilities such as WinZip®.

- The definitions selected for export are listed in the **Definitions to Export** listbox. Click and to delete selected, or all, entries.
- The **Type** drop-down list controls which libraries display in the tree below it. Choices include **Component**, **Symbol**, **Model**, **Package**, **Material**, and **Script**. In the tree, each leaf element (definition) has an associated check box. Select the box to add the definition to the export list above. Clear the box to remove the definition from the export list. With a library or library element node highlighted, click **Library All** to select all definitions in that library for export. Similarly, you can click **Library None** to deselect all definitions in the selected library. **Expand All** and **Collapse All** expand or collapse the entire tree.
- Type a string into the text box next to the **Find** button to scroll to the first definition whose name matches the string. Searches are not case-sensitive. You can also use the wildcard asterisk character in search strings. For example: entering P* should find, in succession, every component that starts with P (or p). Use multiple asterisks to further control matches, for example, *p* to find any definition with a 'p' in it, including as a first or last character; or *p*p* to find "pipe" and "pump", and so on.
- Select the **Remove unused definitions after export** check box to clean up the current project after export is complete.
- When you select the **Show contents after export** check box, the **Contents** dialog box shows what was saved to the archive. This list may include multiple libraries (due to definition references) or external file references.
- Click **OK** to complete the process. The selected definitions, libraries, and referenced files are zipped, and given a double extension, **.definition.zip**. You can email or otherwise transfer the ZIP file to others. The ZIP file is not encrypted, though models and packages contained in it may be.

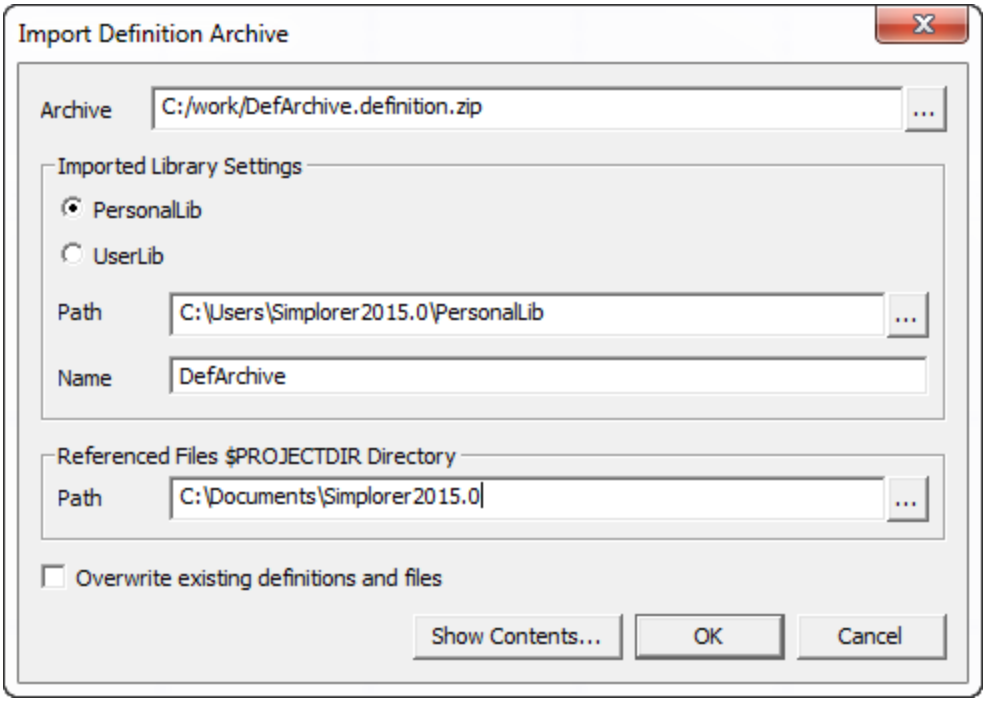
Related Topics

Importing Definition Archives

Importing Definition Archives

To import a definition archive, select **Tools** > **Library Tools** > **Import Definition Archive**. The **Import Definition Archive** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- Select a definition archive. Either type the path and definition archive file name in the **Archive** field, or click ... to browse for the file.
- In the **Imported Library Settings** section, choose the basic location for the imported archive with the **PersonalLib** or **UserLib** radio buttons. You can further choose a subdirectory **Path** under **PersonalLib** or **UserLib** as a location for the libraries.
- You can also enter a **Name** for the imported libraries. The default name is the base name of the archive file.
- On import, data files are moved to appropriate **bin** and **data** directories. External **Referenced Files** that are not appropriate to **bin** and **data** library directories will be referenced using **$PROJECTDIR**. Click ... to choose the directory for these files. The default is the project directory specified in the **Options** > **Directories** dialog box.
- Select the **Overwrite existing definitions and files** check box to replace current definitions and files with archive elements, permissions allowing.
- Click **Show contents** to view a dialog box showing what is in the archive. This list may include multiple libraries (due to definition references) or external file references.

Related Topics

Exporting Definition Archives

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Editing Materials Libraries

A material is a set of information that defines the physical properties of a substance used as the physical foundation of a component model such as a diode. Material definitions are stored in library files with an extension of **.amat**

1. To open the dialog box for a materials library, select **Tools** > **Edit Libraries** > **Materials**. The **Edit Libraries** dialog box opens on the **Materials** tab.
2. Select the material you want to manage in the **Libraries** list. Select the **Show Project definitions** check box to include material objects stored with the project.

   Use the **Search Parameters** panel to search for the material **by Name** or **by Property**. When searching by property, enter the property's *value* in the search box.

   The **Material Filters** tab provides controls for filtering the materials list.
3. Use the buttons described below to manage selected library objects:
   - **View/Edit Materials** – Edit properties or attributes for a selected material in the **Material Editor**. The resulting material may have the same name as the original, but it is saved to the current project, rather than back to the library. In effect, you have "checked out" the part to the project in order to edit it. If you want to write it back to the library, overwriting the original part, click **Export to Library**.

     Use the **Show Project Definitions** check box to include current project elements in the **Edit Libraries** dialog box.
   - **Add Material** – Create a new material which is saved to the current project.
   - **Clone Material(s)** – Create a copy of selected materials with a different name.

     The **View/Edit Material** dialog box opens. Use this dialog box to edit the cloned material's properties. The cloned material is saved to the current project.
   - **Remove Material(s)** – Remove selected materials from the library.
   - **Export to Library** – Export a selected material object to a different library, or to export edited objects from the project to the library if necessary.
4. When finished, click **OK** to save your changes and close the **Edit Libraries** dialog box.

Editing Scripts Libraries

1. To open the dialog box for a script library, select **Tools** > **Edit Libraries** > **Scripts**. The **Edit Libraries** dialog box appears.
2. Select the library file in the **Libraries** list. Use the buttons below to manage selected library objects:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Edit Script** – Edit properties or attributes for a selected model in the **Script Editor**. The resulting script may have the same name as the original, but it is saved to the current project, rather than back to the library. In effect, you have “checked out” the part to the project in order to edit it. If you want to write it back to the library, overwriting the original part, click **Export to Library**.

Use the **Show Project Definitions** check box to include current project elements in the **Edit Libraries** dialog box.

- **Add Script** – Create a new object within the selected library.
- **Clone Script(s)** – Create a copy of selected objects with a different name within the selected library.
- **Remove Script(s)** – Remove selected objects from the library.
- **Export to Library** – Export a selected object to a different library. Use also to export edited objects from the project to the library if necessary.

3. When finished, click **OK** to close the **Edit Libraries** dialog box.

Editing Component Libraries

1. Select **Tools** > **Edit Libraries** > **Components**. The **Edit Libraries** dialog box opens on the **Components** tab.

2. Click to browse for and select a library file. The selected library file appears in the **Libraries** field, and its associated components appear in the grid below. Use these buttons to manage selected library objects:

> **Note:**
>
> Select the **Show Project definitions** check box to include current project elements in the **Edit Libraries** dialog box.

- **Edit Component** – Edit properties or attributes for a selected component in the **Component Editor**. The resulting component may have the same name as the original, but it is saved to the current project, rather than back to the library. In effect, you have “checked out” the part to the project in order to edit it. If you want to write it back to the library, overwriting the original part, click **Export to Library**.
- **Add Component** – Create a new object within the selected library.
- **Clone Component(s)** – Create a copy of selected objects with a different name within the selected library.
- **Remove Component(s)** – Remove selected objects from the library.
- **Export to Library** – Export a selected object to a different library. Use also to export edited objects from the project to the library if necessary.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note	When you modify a library component, the modified definition transfers to the components available to the current project. In effect, you have “checked out” the component to modify it, and it is now in the project. The original library definition is intact. To update the library definition, you must export the modified component from the project to the library.

Hint	Hierarchical components, because of their design and the possibility of subdesigns, cannot be exported to a library. Instead, hierarchical components and their referenced objects such as models and symbols can be exported to libraries of their respective types. See ***Exporting Hierarchical Components*** for details.

3. When finished, click **OK** to close the **Edit Libraries** dialog box.

Exporting Hierarchical Components

Hierarchical components can have dependent definitions such as symbols that reside in other library types. Consequently, when exporting a hierarchical component, you can export referenced symbols, models, and packages.

> **Note:**
>
> When sending exported data such as libraries to third parties, or when relocating exported data, be sure to include all related files.

> **Note:**
>
> Project variables and project datasets used in designs represented by hierarchical components are saved to the library with the component and restored when the component is used. If the variable or dataset already exists in the project (determined by name), the library version is ignored. When creating project variables or datasets for hierarchical component designs to be saved in a library, make sure the names of the variables and datasets are appropriate and not likely to clash with those of other hierarchical components or common defaults.

1. When you select a hierarchical component in the **Edit Libraries** dialog box and click **Export to Library**, the **Export to user library** dialog box appears.
2. Choose the location and file name for the **.aclb** library file that will contain the exported components. Click **Save** to confirm your choice. If the library does not exist, you are prompted to create a new one.

   The **Export Hierarchy** dialog box lists the component and dependent object **Definitions** for the exported components. Dependent objects that can be exported include:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Symbols (exported as **.aslb** files).
- Models (exported as **.asmd** files).
- Packages (exported as **.apkg** files).

Buttons in the **Library** column open the **Export to user library** dialog box where you can browse for a library file in which to store the definition, or create a new library. The library file type and extension correspond to the **Library Type** listed for each object. The **Library** field button name is that of the library in which the definition currently is stored, and is blank if no library is yet assigned. By default, the **Library** field names are set to the same library.

Use the check boxes to choose:

- Which definitions to **Export**.
- Whether to **Overwrite** existing definitions in the target library.
- Whether to **Export Solution** data for schematic components. Solution data is saved in a **.results** folder instead of a library file.

3. Click **OK** to confirm your selections. The definitions (and solution data, if applicable) are saved in the locations and files you chose.

Editing Model Libraries

1. Select **Tools** > **Edit Libraries** > **Models**. The **Edit Libraries** dialog box appears.
2. Select the library file from the **Libraries** scrolling window. Use the buttons described below to manage selected library objects:
   - **Edit Model** – Edits properties or attributes for a selected model in the corresponding model editor: the **C-Model Editor**, the **VHDL Model Editor**, the **SPICE Model Editor**, or the **SML Model Editor**. The resulting model may have the same name as the original, but it is saved to the current project, rather than back to the library. In effect, you have “checked out” the part to the project in order to edit it. If you want to write it back to the library, overwriting the original part, click **Export to Library**.

     Use the **Show Project Definitions** check box to include current project elements in the **Edit Libraries** dialog box.

     Some models—such as those that have been encoded—are not editable. You can edit encrypted models after you enter the password.

   - **Add Model** – Creates a new model object within the selected library. The **Add Model** dialog box lets you name the model, and to choose the model type.
   - **Clone Model(s)** – Creates a copy of selected objects with a different name within the selected library. Encoded models cannot be cloned.
   - **Remove Model(s)** – Removes selected objects from the library. System library models cannot be removed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Export to Library** – Exports a selected object to a different library. Use also to export edited objects from the project to the library if necessary. System library models cannot be exported.

  > **Note:**
  >
  > When exporting SPICE models containing references to external files, the **Message Manager** pane displays an alert if the referenced files cannot be located, prompting you to copy the files into the **personallib** or **userlib** directory.

3. When finished, click **OK** to close the **Edit Libraries** dialog box.

Editing Packages Libraries

1. Select **Tools** > **Edit Libraries** > **Packages**. The **Edit Libraries** dialog box appears.
2. Select the library file from the **Libraries** scrolling window. Use the buttons below to manage selected library objects:
   - **Edit Package** – Edit properties or attributes for a selected model in the **Package Editor**. The resulting package may have the same name as the original, but it is saved to the current project, rather than back to the library. In effect, you have "checked out" the part to the project in order to edit it. If you want to write it back to the library, overwriting the original part, click **Export to Library**.

     Use the **Show Project Definitions** check box to include current project elements in the **Edit Libraries** dialog box.

     Some packages—such as those whose headers have been encoded—are not editable.

   - **Add Package** – Create a new object within the selected library. The **Add Package** dialog box lets you name the package and choose the package type.
   - **Clone Package(s)** – Create a copy of selected objects with a different name within the selected library. Encoded packages cannot be cloned.
   - **Remove Package(s)** – Remove selected objects from the library. System library packages cannot be removed.
   - **Export to Library** – Export a selected object to a different library. Use also to export edited objects from the project to the library if necessary. System library packages cannot be exported.
3. When finished, click **OK** to close the **Edit Libraries** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Editing Symbol Libraries

1. Select **Tools** > **Edit Libraries** > **Symbols**. The **Edit Libraries** dialog box appears.
2. Select the library file from the **Libraries** field. Use the buttons below to manage selected library objects:
   - **Edit Symbol** – Edit a selected symbol in the **Symbol Editor**. The resulting component may have the same name as the original, but it is saved to the current project, rather than back to the library. In effect, you have “checked out” the part to the project in order to edit it. If you want to write it back to the library, overwriting the original part, click **Export to Library**.

   Use the **Show Project Definitions** check box to include current project elements in the **Edit Libraries** dialog box.

   - **Add Symbol** – Create a new object within the selected library.
   - **Clone Symbol(s)** – Create a copy of a selected objects with a different name within the selected library.
   - **Remove Symbol(s)** – Remove selected objects from the library
   - **Export to Library** – Export a selected object to a different library. Use also to export edited objects from the project to the library if necessary.
3. When finished, click **OK** to close the **Edit Libraries** dialog box.

Twin Builder Modelica Connector Library

The Twin Builder connector library provides the convenience and ability to create Modelica models/subsystems with conservative connections in a Twin Builder schematic.

The primary goal is to simulate in the Twin Builder environment, so the connectors are mainly compatible with Twin Builder’s conservative connections – that is, only one potential variable and one flow variable in the connector are supported. You need to use or create a compatible connector in the Modelica environment.

For example, in the Modelica Standard Library, FluidPort in Fluid has more component elements defined in the connector, so you must create a Twin Builder-compatible connector to connect to the provided Twin Builder Modelica connector to successfully compile the model.

> **Note:**
>
> Because of the Modelica Standard, in a connect-equation the two connectors must have the same named component elements with the same dimensions; recursively down to the primitive components. The primitive components with the same name are matched and belong to the same connection set.

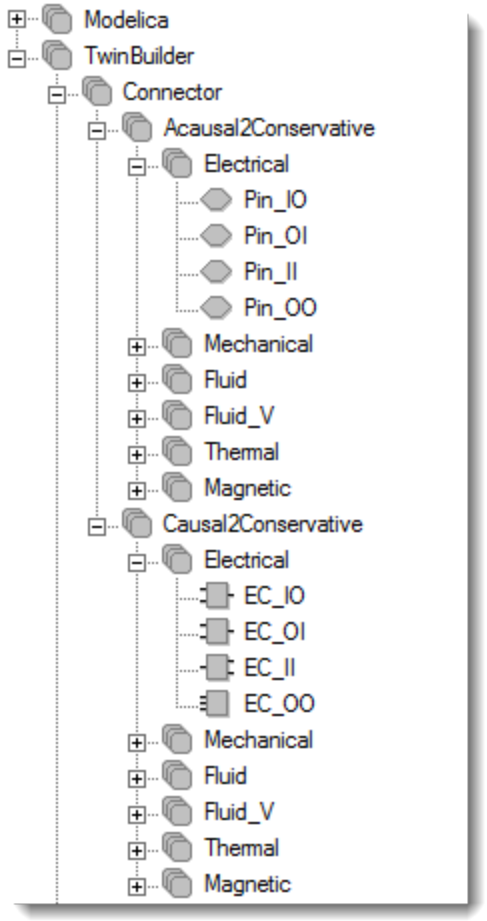
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

Modelica models that use connectors from the Twin Builder Connector library are not supported when you enable the **Use External Solver** option. In such cases, potential and flow variables are exposed through different pins in the Modelica component when placed on the schematic.

Structure of Connector Library

The Twin Builder Modelica Connector library loads into the component manager in the Twin Builder Modelica environment as shown below:



It contains two subcategories: **Acausal2Conservative** and **Causal2Conservative** Each subcategory has the following domains:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Electrical**
- **Mechanical (Rotational, RotationalV, Translational, and TranslationalV)**
- **Fluid**
- **Fluid_V**
- **Thermal**
- **Magnetic**

The **Acausal2Conservative** subcategory is for directly exposing conservative connections from Modelica's acausal connector. Two examples are shown in Examples. The connectors are named with preferred calculated directions for potential variable and flow variable based on your Modelica application. For example, Pin_IO means the preferred calculation of this electrical pin is voltage as input and current as output.

The **Causal2Conservative** subcategory is convenient if you already have non-conservative designs in Modelica with a previous version of the Modelica environment and want to connect to conservative connections in the Twin Builder schematic. The Causal2Conservative connector models have two causal pins to be connected for the Modelica side, and one domain acausal pin to be connected in the schematic after compilation.

In the Simplified Motor Driven Power Train example, the Causal2Conservative connector needs to be used together with the compatible Acausal2Conservative connector.

> **Note:**
>
> You need to find the compatible pair of the connectors based on their application for successful compilation and simulation in Twin Builder.

Examples

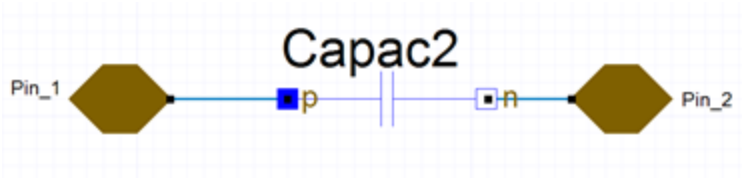
Two demo RLC examples are shown. You can either compile the RLC Modelica model separately, bring it into Twin Builder, connect and simulate it, or combine the RLC circuit as a subcircuit in the Modelica environment, compile and bring one component into Twin Builder, and simulate.

One demo Simplified Motor Driven Power Train example is shown for Causal2Conservative connection.

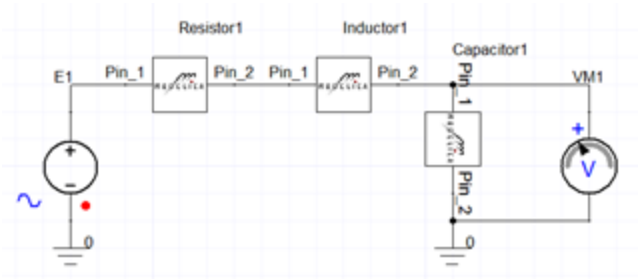
RLC Individual

To create an individual RLC model in Modelica, connect proper pins in the Twin Builder Connector library and connect them to the resistor, inductor, and capacitor models separately as shown below:

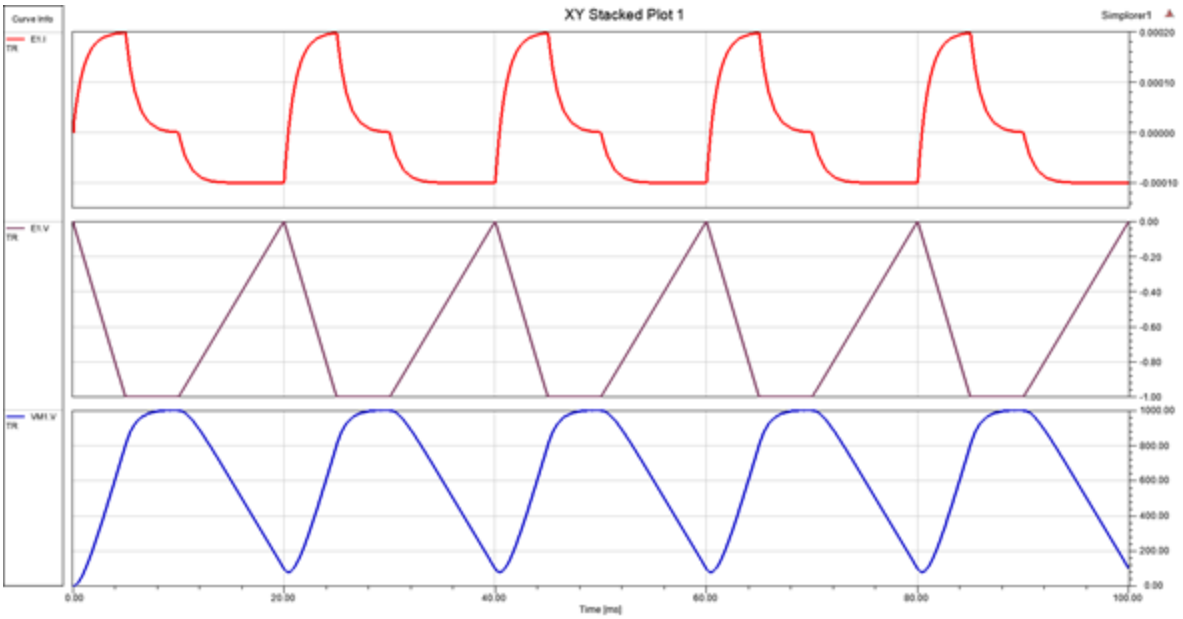
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Compile them into Twin Builder components, then connect and simulate. The Twin Builder circuit is shown below:



With R = 1000 Ohm, L = 0.01 H, and C = 1e-6 F and a periodic Trapezoidal voltage source, the simulation results are shown below:



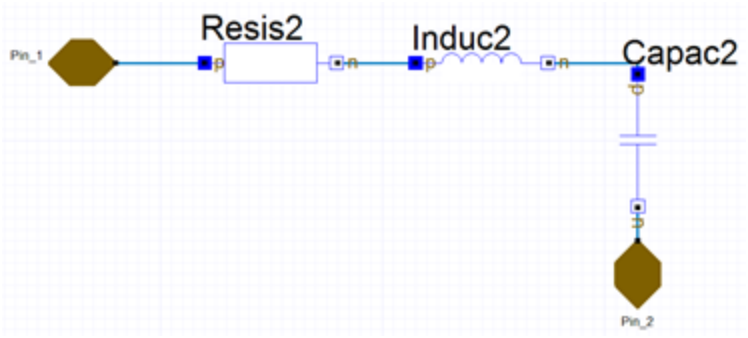
The design is in this location:

<Ansys EM Installation> **\Win64\Examples\Twin Builder\Modelica\Twin Builder Modelica Connectors\Acausal_Conservative\RLC_individual.aedt**

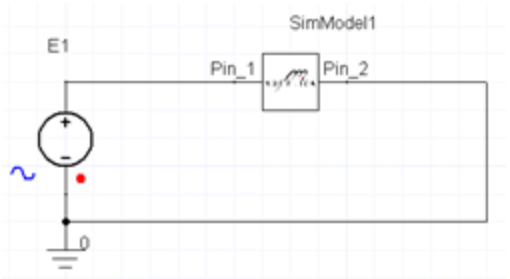
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

RLC Integrated

Another way is to combine the subcircuit in the Modelica environment.

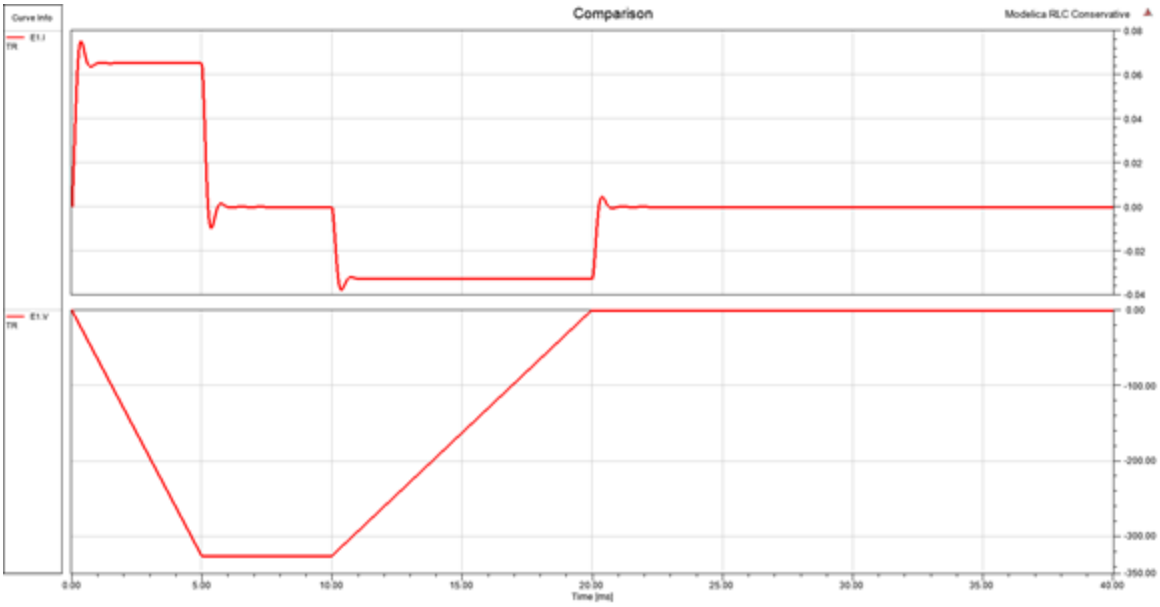


The Twin Builder circuit is simple:



With R = 1000 Ohm and with a non-periodic Trapezoidal voltage source, the simulation results are shown below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The Twin Builder design can be found in this location:

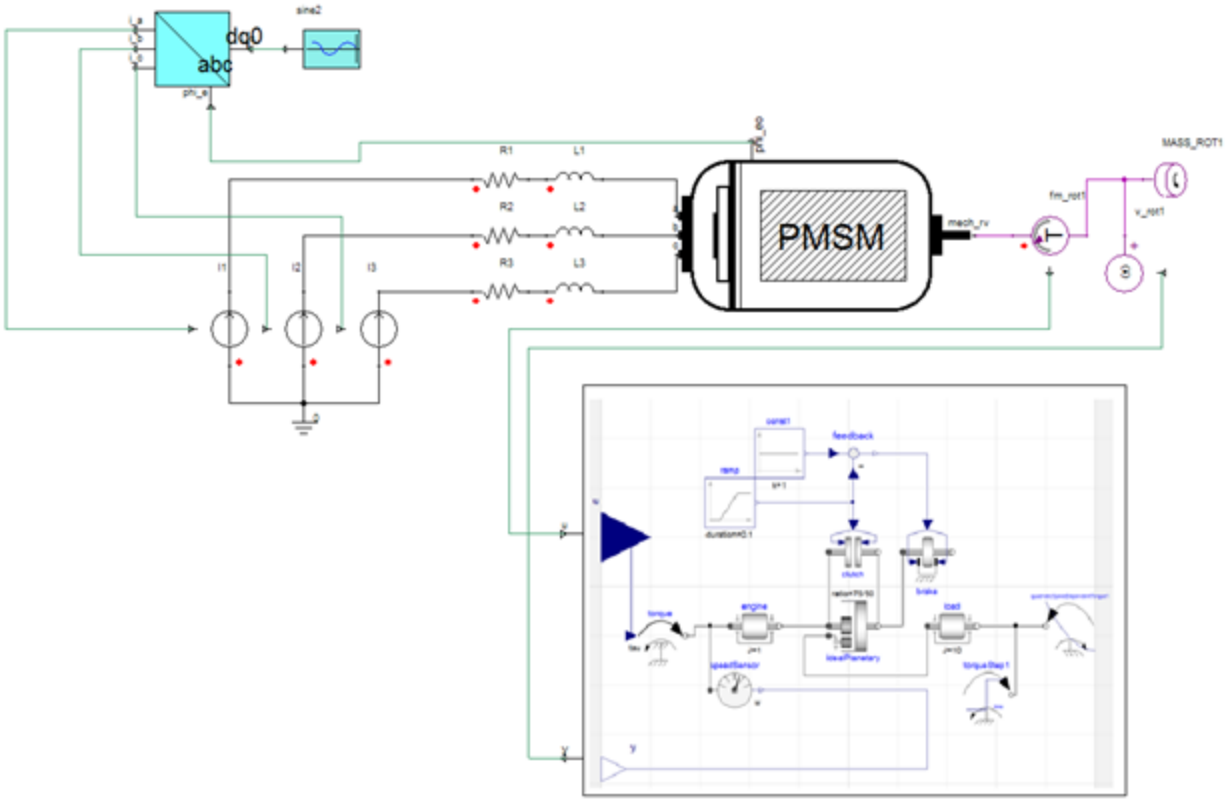
<*Ansys EM Installation*> **\Win64\Examples\Twin Builder\Modelica\Twin Builder Modelica Connectors\Acausal_Conservative\RLC_integrated.aedt**.

Simplified Motor Driven Power Train_conservative

This example demonstrates the usage of Causal2Conservative connectors. Easier to understand, an application example from Twin Builder installation (<*Ansys EM Installation*> **\Win64\Examples\Twin Builder\Applications\Modelica\ Simplified Motor Drived Power Train.aedt**) is selected.

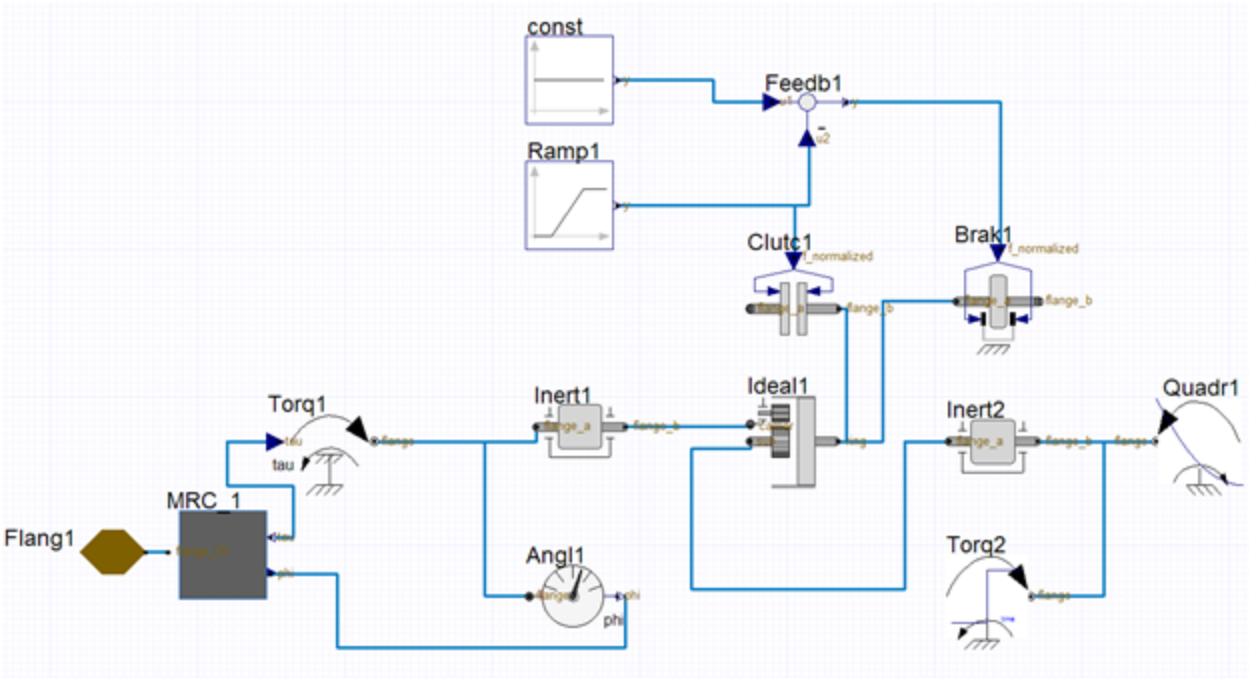
The original schematic is shown below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

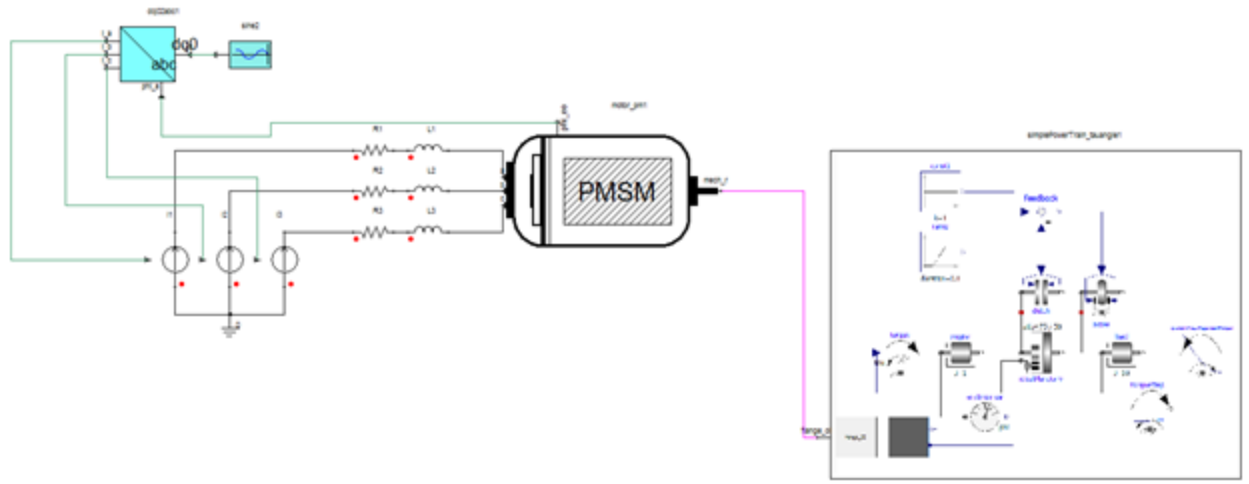


Using the Twin Builder Modelica Connector library, the acausal to conservative connection can be easily established with a pair of Causal2Conservative and Acausal2Conservative connector based on existing application. The actual Modelica model is shown below with Mechanical Rotational Causal2Conservative connector MRC and Acausal2Conservative connector Flang.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

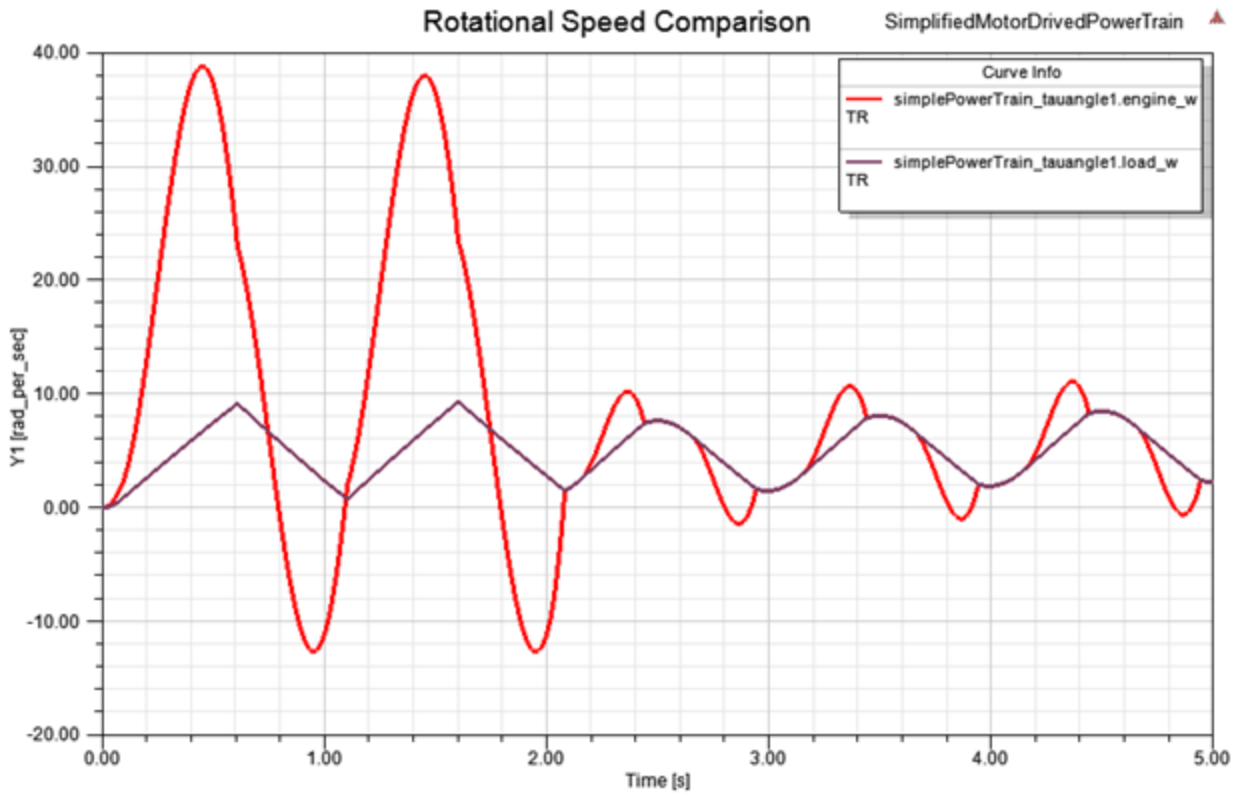


The Twin Builder design with compiled Modelica subsystem:



Rotational speed comparison of the subsystem engine and load is shown below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The Twin Builder design can be found under

*<Ansys EM Installation>***Win64\Examples\Twin Builder\Modelica\Twin Builder Modelica Connectors\Causal_Conservative\MotorDrivenPowerTrain.aedt**.

Managing Library Files

You can use **Manage Files** to change the names of the **UserLib** and **PersonalLib** libraries and directories, move libraries and directories to another location in the libraries or directories specified for **UserLib** or **PersonalLib**, or delete libraries and directories.

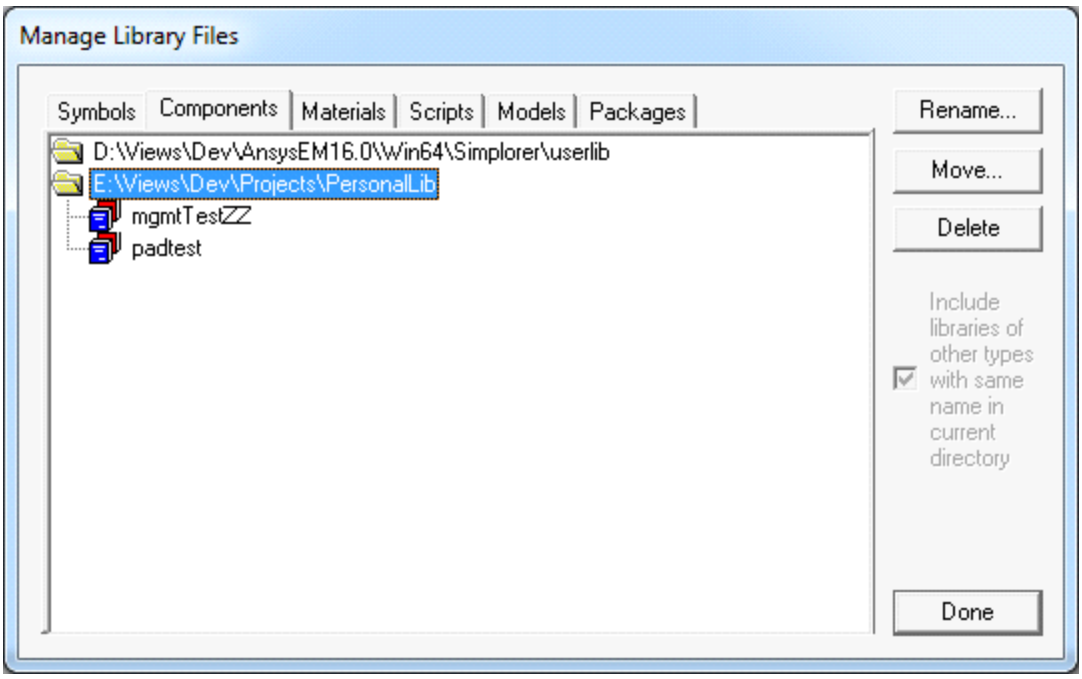
> **Note:**
>
> - You cannot move libraries out of the **UserLib** or **PersonalLib** hierarchies.
> - To move a library or directory to a subfolder, that subfolder must already exist.

Components and other definitions in libraries directly affected by an action which refer to definitions changing library name or location will have those references adjusted. Definitions in libraries indirectly affected—that is, those not having their names or locations modified by an action—with references to definitions in directly affected libraries are updated.

To manage library files:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Tools** > **Library Tools** > **Manage Files**. The **Manage Library Files** dialog box opens on the **Components** tab. There is a tab for each library type: **Symbols**, **Components**, **Materials**, **Scripts**, **Models**, and **Packages**.



2. Click the tab you want.
3. Select an individual library or a directory on which you want to perform the action.
4. Click **Rename**, **Move**, or **Delete**.
5. Select the **Include libraries of other types with same name in current directory** check box to cause the action to be performed on other types with the same name in the same directory. For example, if you select **mydefs** on the **Components** tab, the action is extended to a symbol library in the same directory called **mydefs.aslb**.
6. Click **Done** when finished managing libraries.

Exporting a Model as an FMU

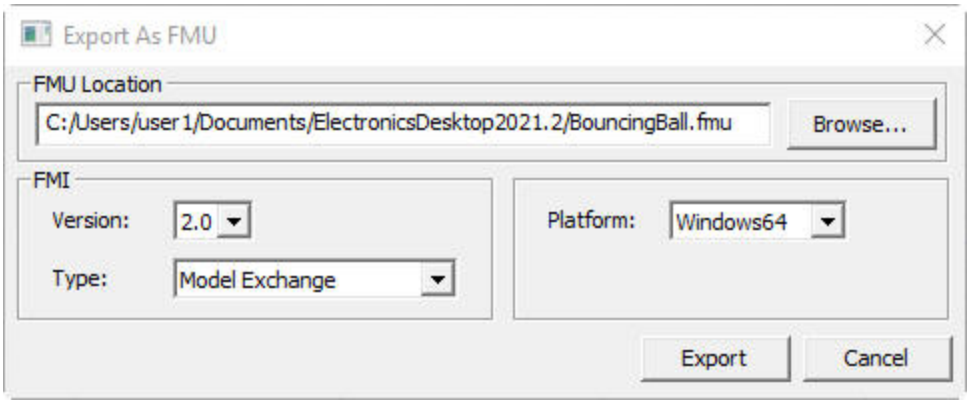
To export a model as an FMU:

1. Open the **Project Manager** pane.
2. Open the project **Definitons** > **Models** tree.
3. Select the model to export as an FMU.
4. Right-click and select **Export as FMU** to open the **Export as FMU** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> If **Export as FMU** is not available in the menu, then the selected model doesn't support export as FMU.

5. In the **Export as FMU** dialog box:



   a. Navigate to the **FMU Location** where you want to save the exported file.
   b. Select the **Version** of the FMI interface to use.
   c. Select the FMI **Type**.
   d. Select the OS **Platform** for which the FMU model is compiled.
   e. Click **Export**.

> **Note:**
>
> Exporting FMUs to a Windows 32-bit platform is useful if you want to use the model in applications other than Twin Builder. However, those models cannot be reimported and used in Twin Builder. A twin deployment from a subsheet containing an encrypted TBROM is not allowed.

To export an SML format LTI model as an FMU, follow the procedure above. During exporting, the LTI model is converted to an equivalent model in Modelica and compiled to an FMU. When the number of states in the LTI model is greater than 100, the Modelica model is written in sparse matrix format for faster compilation time; when the number of states is less than or equal to 100, the Modelica model is written in full matrix format.

Export as a Modelica Package

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To export an SML-formatted LTI model as a Modelica package, select **Exporting a model as Modelica Package**. This creates a Modelica package which can be consumed by a Modelica environment in Twin Builder.

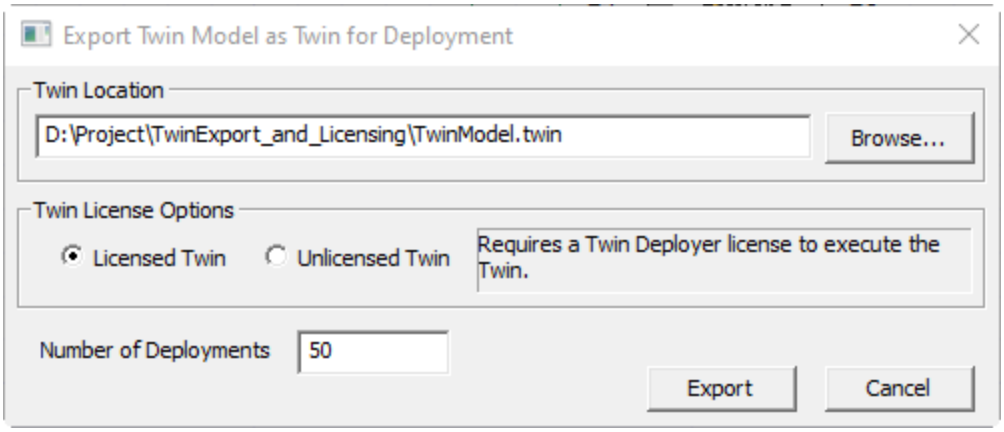
More information on the FMI standard can be found at: http://fmi-standard.org/

Exporting a Twin Model as a Twin

Follow this procedure to export a twin model as a twin.

1. Open the **Project Manager** pane.
2. Expand the project **Definitions** > **Models** tree.
3. Select the twin model to export as a **.twin** file.
4. Right-click the model and select **Export as** > **Twin for Deployment** to open the **Export Twin Model as Twin for Deployment** dialog box.

> **Note:**
>
> If **Export As** > **Twin for Deployment** is not available, then the selected model doesn't support export as a **.twin** file. See ***Creating Twin Models*** for more information. Twin models created using earlier versions of Twin Builder (R20.1 or earlier) must be recompiled as twin models using the current version of Twin Builder. A twin deployment from a subsheet containing an encrypted TBROM is not allowed.



5. Browse to the target location for the exported **.twin** file.
6. Specify whether the exported twin should check out a license at execution time.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> If the twin model is exported as unlicensed twin, a separate export license is required. The Twin Deployer licenses for the licensed twin or the export licenses must be purchased separately.

7. Set the expected number of deployments.
8. Click **Export** to export the model as a **.twin** file.

Related Topics

Creating Twin Models

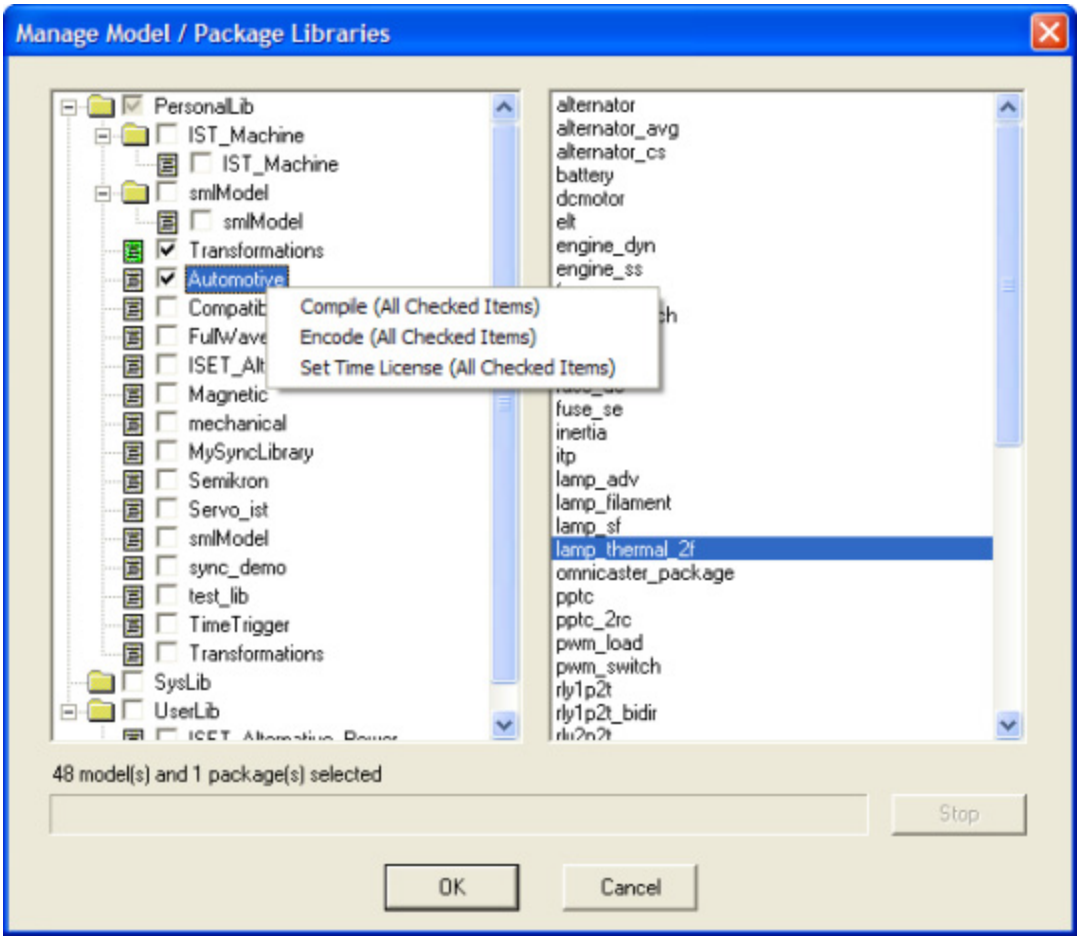
Recompiling, Encoding, and Licensing Libraries

If you make changes to binary model structure, dependent models or model libraries may need to be recompiled. You may also want to encode or license multiple libraries and/or packages. Twin Builder provides an automated way to manage these changes to model and package libraries.

1. To recompile, encode, or license model and/or package libraries, select **Tools** > **Library Tools** > **Manage Twin Builder Model Libraries**.

   The **Manage Model/Package Libraries** dialog box displays. The left panel is an expandable tree of all available model libraries and packages. Use the check boxes to select the libraries to change. Select a folder icon to select all libraries within the folder. Text appears below the left panel informing you of the number of models and packages currently selected. The right panel lists the elements in all selected libraries for informational purposes only.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. Select the desired libraries and/or packages.
3. Right-click in the library tree panel. A menu containing the following selections appears:
   - **Compile (All Checked Items)**
   - **Encode (All Checked Items)**
   - **Set Time License (All Checked Items)**

   > **Note:**
   >
   > If none of the libraries is selected, right-click one to perform operations on it. The shortcut menu items contain the library name instead of **All Checked Items**.

   a. To compile the checked items, select **Compile (All Checked Items)**.

   The progress bar below the library listing monitors the progress of the compilation. The original library and package files are saved with a **.bak** extension.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Click **Stop** during compilation to halt the process, leaving the libraries and packages in their original state.

b. To encode or encrypt the checked items, select **Encode (All Checked Items)**.

The **Encryption Settings** dialog box appears. See ***Encryption Settings Dialog Box*** for details on encryption settings. Encryption of encoded items is not supported. Encoding encrypted items results in messages informing you that “The encode password has been removed.”

c. To set a time license for the checked items, select **Set Time License (All Checked Items)**. See ***Time License Settings*** for details.

4. When finished, click **OK** to close the dialog box.

Using the Password Manager

The Password Manager lets you add and delete encryption resources using AES Encryption technology. These resources can be applied to SML and SPICE models, and to VHDL-AMS models and packages that require password access. Once encrypted, models and packages require a password before they can be viewed or used. For convenience, the same password can be applied to multiple resources.

Related Topics

Adding an Encryption Resource

Deleting an Encryption Resource

Time License Settings

Encryption Settings Dialog Box

Encrypting or Encoding a VHDL-AMS Model

Encrypting or Encoding an SML, or SPICE Model

Encrypting or Encoding a VHDL-AMS Package

Recompiling, Encoding, and Licensing Libraries

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding an Encryption Resource

1. Select **Tools** > **Password Manager**. The **Password Manager** dialog box appears.

   > **Note:**
   >
   > The **Resource Name** field lists currently defined resources. Click a resource name to display a list of all items encrypted with it in the **Encrypted Items** field.

2. Click **New**. The **Add Encryption Resource** dialog box appears.
3. Specify a name for the encryption resource in the **Resource Name** text field.
4. Assign a password to the resource by choosing one of the following options:
   - **Enter Password** – Enter and confirm a password for the new resource.

     > **Note:**
     >
     > Passwords must be at least eight characters long. Passwords may contain spaces, but may not begin or end with a space.

   - **Use Same Password As** – Select the radio button and choose an existing resource from the drop-down list. The chosen resource's password will then be used by the new resource.
5. Once you have assigned a password, click **OK** to create the resource and close the dialog box.

   The **Password Manager** dialog box now lists the new resource.
6. Repeat the above steps as needed to add encryption resources. Click **OK** when you're finished.

Deleting an Encryption Resource

1. Select an existing resource to highlight it.
2. Click to delete the resource.

   > **Note:**
   >
   > Deleting a resource merely removes it from the list of available encryption resources. Models and packages that were encrypted using that resource are unaffected.

   Click **OK** when you are finished deleting resources.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Encryption Settings Dialog Box

The **Encryption Settings** dialog box appears when you select **Encryption Settings** in either the VHDL-AMS Package Editor, or in the VHDL-AMS, SML, or SPICE Model Editors.

- **Plain Text** – No encryption or encoding is applied.
- **Encrypt** – Displays a list of available password resources in the **Resource Name** field. Select the desired resource and click **OK** to apply its settings to the model or package. You can also click **Add** to open the **Add Encryption Resource** dialog box in which you can create a new encryption resource.
- **Encode** – Click **OK** to encode the model or package, hiding the text from view.

Related Topics

Encrypting or Encoding a VHDL-AMS Model

Encrypting or Encoding an SML, or SPICE Model

Encrypting or Encoding a VHDL-AMS Package

Recompiling, Encoding, and Licensing Libraries

Time License Settings

Use time license settings to apply an expiration date beyond which a model can not be used in simulations. You can remove the date stamp from new and editable models. Apply time license settings to VHDL-AMS, SML, and SPICE models, and to VHDL-AMS packages from within their respective editors.

To add time license information:

1. Select *model_or_package* **Editor** > **License Settings**. The **License Information** dialog box appears.
2. Select the **Enable Time License settings** check box. Enter the license expiration date into the **Date** field.

   You can also click the down-arrow button to the right of the date field to reveal a pop-up calendar.

   When the desired calendar page is displayed, click the day of the month to transfer the date selection to the **Date** field in the **License Information** dialog box.
3. Click **OK** to confirm the date entry and close the dialog box. The **Encryption Settings** dialog box appears.
4. Encrypt or encode the model or package using the applicable procedure:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Encrypting or Encoding a VHDL-AMS Model
- Encrypting or Encoding an SML, or SPICE Model
- Encrypting or Encoding a VHDL-AMS Package
- Recompiling, Encoding, and Licensing Libraries

To remove the time license from a model:

1. With the model from which you intend to remove time licensing open in the model editor, click *model_or_package* **Editor** > **Time License**. The **License Information** dialog box appears.
2. Deselect **Enable Time Licensing settings**.
3. Click **OK** to remove the settings, then save the model.

Using the Component Editor

These topics cover creating new components or editing existing components within a project file. For information on managing components within the external libraries, see

Editing an Existing Component

Adding a New Component

The Edit Component Dialog Box

The Properties Dialog Box (Edit Component)

The Component Netlist String Property

Editing Components

Use the **Edit Component** dialog box to:

- Specify or change the component name, description, associated bitmap, and default property values, including those necessary for netlisting and cosimulation, as appropriate.
- Specify or re-specify a graphical symbol to represent the component in the schematic editor.
- Specify or change the component terminal properties.

The changes you make when editing a library component become part of the current project. (The edited component is saved to the project, not back to the library.) To make a new or modified component available for use in other projects, select **Export** in the **Edit Libraries** dialog box save it to a component library (**.aclb**) file.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> If you are adding a new component, it is useful to first identify and if necessary, create, the necessary dependencies before defining your component. See *Component Creation Process* for details.

Editing an Existing Component

Follow this procedure to edit an existing component.

1. Select a component.
   a. In the Project tree, expand the **Definitions**/**Components** subfolder for the project that contains the component you want to edit. Double-click the entry for the component you want to edit, or right-click the entry and select **Edit Component**.
   b. Open the **Tools** > **Edit Libraries** > **Components** dialog box, and select a component. Click **Edit Component**, or double-click the selected entry.
   c. Right-click an on-sheet component and select **Edit Component**.
2. The **Edit Component** dialog box opens, displaying the definition of the selected component.
3. When you're finished, click **OK** to save the changes.

   If you removed a Quantity property to which a symbol pin was associated, a dialog box displays informing you that "**The pins of the referenced symbol do not match the current component terminals and properties.**"

   a. If you want to adjust the current symbol (or a clone of the current symbol if another component is using the same symbol), click **Yes**.
   b. If you want to generate a symbol consistent with the edited component properties, click **No**.

Adding a New Component

Add new components by creating a model using the appropriate editor (VHDL-AMS, C-Model, SML, or SPICE), or by importing a model. You can also add new components with the **Components Library** dialog box.

Before you define your new component, identify and modify (or create, if necessary) the **Symbol** and Model or netlist that will be associated with the component.

1. To add a new component, open the **Tools** > **Edit Libraries** > **Components** dialog box and click **Add Component**. The **Edit Component** dialog box opens.
   - You can also open the **Tools** > **Edit Libraries** > **Components** dialog box and select an existing component on which you will base your new component. Click **Clone**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Component. A renamed copy of the selected component appears on the list.

Double-click the new component, or click **Edit Component**. The **Edit Component** dialog box opens.

2. When finished defining the component, click **OK** to save the changes.

The Edit Component Dialog Box

Use the **Edit Component** dialog box to edit existing components, or to create new components, in a project file. The changes you make when editing a component are saved to a definition within the current project, not to the “parent” library. If the edited component is used in any schematic within the project, all default-value instances of the parameter(s) changed are updated immediately.

Open the **Edit Component** dialog box in any of the following ways:

- Double-click a component definition on the Project tree.
- Double-click a component listed in the **Edit Library** dialog box.
- Right-click a component on a schematic and select **Edit Component**.

The **Edit Component** dialog box contains four tabs:

- **General**
- **Miscellaneous**
- **Terminals**
- **Simulation Models**

The **General** and **Terminals** tabs each contain a symbol preview window which reflects any changes you make to the symbol.

Before making a new or modified component available for use in other Twin Builder projects, you must export it to a *user* or *personal* component library (.**aclb**) file using the **Edit Libraries** dialog box. The library must be added to the circuit or schematic within the target project.

To update another project with your updated components, select its icon in the Project tree and click **Tools** > **Project Tools** > **Update Definitions**.

General Tab (Edit Component)

Use the **General** tab to specify a name and all parameter values for the component. You can also set up component dialog boxes, have a symbol created for the component, assign an existing symbol from a symbol library, and edit the symbol (You must exit the component editor to edit the symbol.) The symbol preview window at the bottom of the dialog box updates to reflect your changes.

Component Area

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Use the **Component** area to enter a **Name**for the component. The **Original library** path and name displays below the **Name**. If the **Setup Dialog** button is present, you can design and add a custom dialog box for the component using the *Component Dialog Wizard*. Click **Properties** to open the **Properties** dialog box, where you can add, edit, or remove default or local properties and values.

> **Note:**
>
> If you remove a Quantity property to which a symbol pin was associated, when you click **OK**, a dialog box displays informing you that "**The pins of the referenced symbol do not match the current component terminals and properties.**" at which time you can choose to adjust the current symbol (or a clone of the current symbol if another component is using the same symbol), or to autogenerate a symbol consistent with the edited component properties.

Symbol Area

Use the **Symbol**area to choose an existing symbol for elements in the current design from a drop-down list, and display a picture of the currently selected symbol. You can choose any symbol that has an equal or greater number of pins than the component specified in the **ComponentName** field. You can also **Auto-create** a symbol.

The component's library of origin displays beneath the component display in the **Symbol** area; if the component is **local to this Project**, that message is displayed. Click **Select** to open **Select Definition** dialog box. You can then to choose a definition for the symbol from a list. Click **Edit** to modify the symbol. A confirmation message displays with a reminder that the **Component** and **Definition** editors will close, and that all changes will then be saved if you continue. Click **Clear** at any time to clear all selections and displays in the **Symbol** area.

Model Area

Use the **Model** area to choose the current model used by the selected component from a drop-down list; the model text also displays. Click to display the **Select Definition** dialog box. Use this dialog box to add a new model to the drop-down list. A confirmation message displays with a reminder that the **Component** and **Definition** editors will close, and that all changes will be saved if you choose to continue. Click **Edit** to edit the current model. Select **Clear** to remove the currently selected model from the drop-down list.

Click **OK** to save your changes and close the dialog box, or click **Cancel** to close the dialog box without saving any changes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Miscellaneous Tab (Edit Component)

Use the miscellaneous fields to enter information such as the name, manufacturer, description, and reference designator. You can also specify a bitmap image to associate with the component listing in the Project tree, and an associated help description (if you use your own documentation).

- The top two rows of the **Miscellaneous** tab display the **Modified Date** and **Original Date** of the component. Specify the **Current Author**, **Manufacturer**, **Original Author**, and **Data Source** in those text boxes.
- To specify an **Example File** for the component, click to browse for the directory and example project file .
- Specify the component's **Reference Designator Base** value, or enter a text **Description** for the component.
- Specify the **Bitmap** image displayed with the component listing using the drop-down list.

  > **Note:**
  >
  > The bitmap image files must be located in <*Installation Directory*>**\Bitmaps**.

- The **Component Help** area displays the help file and help topic associated with the component. You can also specify a help file and topic if you want to use your own help file. The help file type must be **.chm** (compiled Microsoft HTML Help) and must be located in <*installdir*>\Help. To link a component to a help file and topic do the following:

1. Copy the **.chm** file to <*installdir*>\Help.
2. Start Twin Builder.
3. Edit the component for which you want to set up a help link.
4. In the **Edit Component** dialog box, select the **Miscellaneous** tab.
5. Choose the **.chm** file from the list in the **Help File** combo box.
6. Enter a help topic **.htm** file name in the **Help Topic** field. This file name is the **.htm** help topic file compiled in the selected **.chm** file. You must supply this information.
7. Click **Test** to verify the link. The specified .**chm** file opens, displaying the component's help topic.
8. Click **OK** to close the **Edit Component** dialog box, and export the component back to its library; or click **Cancel** to close the dialog box without saving any changes.

Terminals Tab (Edit Component)

Use the **Terminals** tab to associate each symbol pin with the proper terminal. First order the symbol pins, then select the corresponding symbol; you can then visually associate the symbol

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

pins with the netlist properties for the simulation model. The symbol preview window at the bottom of the dialog box reflects your changes.

You can select one or more terminals directly, enabling the following settings:

- **Order Symbol Pins** – Displays the terminals present in the symbol definition. These are ordered as they are written in the netlist for the corresponding model terminals. You can modify the order.
- **Model Pins** – Displays the terminals present in the model definition. The order is defined in the model text and is fixed in the user interface.
- **Nature** – Select the physical type (electrical, magnetic, and so on) associated with the pin.

- **Unconnected** – Select a corrective action if the pin remains unconnected.
- **Unconnected/default behavior** – Controls what Twin Builder does at analysis time if this pin is left unconnected in a schematic. The options are:
  - **Flag as error** – Results in the generation of netlist errors in the **Message Manager** pane.
  - **Unique net** – Writes a unique netname (for example, “unconnected0”) for the unconnected pin. This allows connections in a multi-element netlist line even when the pins of the component are unconnected.
  - **Grounded** – Assigns a user-specified **R to ground** value (see below).
  - **No net** – Results in assignment of an empty string when netlisting the terminal.

    Click **Apply to selection** to apply your choice to the selected pin.
- **R to ground (Ohms)** – If you select **Grounded** for the **Unconnected/default** behavior, you can specify the resistance to ground within the **R to Ground** entry window. The default value is 1e9 (Ohms). Click **Apply to selection** to apply your changes to the selected terminals.

Definition Parameter Tabs

The **Definition Parameter** area displays the signal, quantity, and parameter properties in a pane as:

- **Name** – Property name.
- **Value** – Property value.
- **Unit** – Units of the property value.
- **Has Pin** – Indicates whether the property has a pin.

You can also add and modify definition parameters with the **Parameter Selection** dialog box.

Click **OK** to save your changes and close the dialog box; click **Cancel** to close the dialog box without saving any changes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Simulation Models Tab (Edit Component)

Use the **Simulation Models** tab to add alternate simulation models or netlist information to the component definition.

> **Note:**
>
> The **Simulation Models** tab is not shown for components representing subcircuits.

Add Model – Select a simulation model on the **Select Definition** dialog **Models** tab.

Add Netlist Line – Open a text editor where you can enter netlist information.

The **Alternate Models** pane lists simulation models currently associated with the component. The listing shows the model **Type** (SML, VHDL, and so on), **Name**, **Architecture**, and **Library**. Click **Edit** to open the model in the appropriate editor.

The Properties Dialog Box (Edit Component)

Use the **Properties** dialog box to add, edit, and remove properties for a component. Changes made here affect all instances of the component in the current design.

To edit component properties:

1. Double-click the component definition in the Project tree to open the **Edit Component** dialog box.
2. Click **Properties** on the **General** tab to open the **Properties** dialog box containing four tabs:
   - **Parameter Defaults**
   - **Properties**
   - **Quantities**
   - **Signals**
3. When finished editing component properties on the tabs, click **OK** to close the **Properties** dialog box; click **OK** to close the **Edit Component** dialog box.

Parameter Defaults Tab (Edit Component)

This tab displays and sets the default values for component parameters for general analysis (**Value** option) or statistical analysis (**Statistics** option). You can also add, edit, or remove properties.

Value Option (Parameter Defaults Tab)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Click **Value** to list, display, and set the default values for the following component parameters used during general analysis:

- **Name** – Display the names of component parameters. Names can be edited for case only.
- **Value** – Change the initial values of parameters.
- **Unit** – Set the unit of measure for the parameter value.
- **Evaluated Value** – This read-only column displays the evaluated value of the parameter.
- **Description** – Enter a description for the parameter.
- **Netlist Unit** – Set the unit of measure used when the parameter is netlisted.
- **Callback** – Associate a callback script with the parameter.
- **Read-only** – Choose whether the parameter settings can be changed.
- **Hidden** – Choose whether the parameter is hidden by default.
- **Property Type** – Display the type (for example, Real) for generic properties.
- **Show Pin** – Show a pin for the property on the component's symbol.
- **Sweep** – Enable the value for sweep.
- **Default SDB** – Enable the value to be stored in the project SDB.

Statistics Option (Parameter Defaults and Quantities Tabs)

Click **Statistics** to list, display, and set the values for the following component parameters applicable to statistical analysis:

- **Name** – Display the names of component parameters.
- **Include** – Choose whether the associated component parameter will be varied during statistical analysis. Select **Include** for each parameter you want Twin Builder to vary during statistical analysis.
- **Distribution** – Display and choose whether the parameter value distribution is Uniform, Gaussian, lognormal, or user-defined. To change the distribution setting, click the cell to display and select an option.
- **Distribution Criteria** – Click the button in this column to open the **Edit Distribution** dialog box, in which you can set distribution criteria. The **Distribution Type** drop-down menu displays whether the distribution is Uniform or Gaussian, and you can specify values for **Cutoff Probability**, **Mean**, and **Tolerance**. Units for **Mean** and **Tolerance** can be set using their adjacent drop-down menus.

Properties Tab (Edit Component)

- **Name** – Displays the names of component parameters. Names can be edited for case only.
- **Value** – Change the initial values of parameters.
- **Unit** – Set the unit of measure for the parameter value.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Evaluated Value** – This read-only column displays the evaluated value of the parameter.
- **Description** – Enter a description for the parameter.
- **Callback** – Associate a callback script with the parameter.
- **Read-only** – Choose whether you can change the name, value, unit, description, and callback values for the property.
- **Hidden** – Choose whether the parameter is hidden by default.

Quantities Tab (Edit Component)

The **Quantities** tab lists the simulation parameters of the selected component. Parameter information displays for **Value**, **Optimization**, **Tuning**, **Sensitivity**, and **Statistics** by selecting the corresponding radio button. you can also add, edit, or remove properties.

Value

- **Name** – Displays the names of component parameters. Names can be edited for case only.
- **Value** – Change the initial values of parameters.
- **Unit** – Set the unit of measure for the parameter value.
- **Netlist Unit** – Set the unit of measure used when the parameter is netlisted.
- **Description** – Enter a description for the parameter.
- **Callback** – Associate a callback script with the parameter.
- **Direction** – Select the parameter direction (In, Out, InOut, or Don’t Care).
- **Show Pin** – Choose whether the parameter pin appears on a schematic.
- **Default SDB** – Enable the value to be stored in the project SDB.

When adding a quantity property, you must specify a **Name** and choose a **Unit Type** (angle, frequency, capacitance, and so on). Enter an initial value into the **Value** field. This can be a number, variable, or expression. Referenced project variables should be prefixed with a '$'. Examples: 22.4pF, $C1, 2*cos($x). If the value requires a multiplier unit (such as **meg** for “×1000000”) click the **Units** field to select the multiplier unit.

Optimization

Click **Optimization** to set default values for input parameters used in optimization analyses.

- **Name** – Displays the names of component parameters. Information in this column is read-only.
- **Include** – Use the parameter optimization settings.
- **Nominal Value** – Displays the value and unit currently set for the parameter. Information in this column is read-only.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Min, Max, and Unit** – Set the minimum and maximum values and their units. By default, the **Min** and **Max** values are set to plus and minus one-half the **Nominal Value**, and the **Unit** is the same as that of the **Nominal Value**.

Tuning

Click **Tuning** to set default values for input parameters used in tuning analyses.

- **Name** – Displays the names of component parameters. Information in this column is read-only.
- **Include** – Use the parameter optimization settings.
- **Nominal Value** – Displays the value and unit currently set for the parameter. Information in this column is read-only.
- **Min, Max,** and **Unit** – Set the minimum and maximum values and their units (the range over which tuning will occur). By default, the **Min** and **Max** values are set to the values from the optimization settings.
- **Step** and **Unit** – Set the size of the step for successive tuning iterations in the tuning range. By default, these values are set to 10 percent of the parameter's **Nominal Value**.

Sensitivity

Click **Sensitivity** to set default values for input parameters used in sensitivity analyses.

- **Name** – Displays the names of component parameters. Information in this column is read-only.
- **Include** – Use the parameter optimization settings.
- **Nominal Value** – Displays the value and unit currently set for the parameter. Information in this column is read-only.
- **Min, Max,** and **Unit** – Set the minimum and maximum values and their units (the range over which tuning will occur). By default, the **Min** and **Max** values are set to the values from the optimization settings.
- **Initial Disp.** and **Unit** – Set the initial displacement value. The initial displacement is the difference between the starting value and the next solved design variation. During a sensitivity analysis, Optimetrics will not consider an initial value that is greater than this step size away from the starting variable value. By default, these values are set to the parameter's **Step** value.

Statistics

Click **Statistics** to set default values for input parameters used in statistical analyses.

- **Name** – Displays the names of component parameters. Information in this column is read-only.
- **Include** – Use the parameter optimization settings.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Nominal Value** – Displays the value and unit currently set for the parameter. Information in this column is read-only.
- **Distribution** – Click in the **Distribution** column and select **Uniform**, **Gaussian**, **Lognormal**, or **User Defined**.
- **Distribution Criteria** – Opens the **Edit Distribution** dialog box, in which you can set criteria appropriate for the chosen **Distribution** type.

Signals Tab (Edit Component)

The **Signals** tab lists the names and values of signals for digital elements. Parameter information can display for **Values**, **Optimization**, **Tuning**, **Sensitivity**, and **Statistics**. you can also add, edit, or remove properties.

Value

- **Name** – Displays the names of component parameters. Names can be edited for case only.
- **Value** – Change the initial values of parameters.
- **Unit** – Set the unit of measure for the parameter value.
- **Netlist Unit** – Set the unit of measure used when the parameter is netlisted.
- **Description** – Enter a description for the parameter.
- **Callback** – Associate a callback script with the parameter.
- **Signal Type** – Shows the type of signal (for example, real or bit) set for the parameter.
- **Direction** – Select the parameter direction (In, Out, InOut, or Don't Care).
- **Show Pin** – Controls whether the parameter pin is shown on a schematic.
- **Default SDB** – Enable the value to be stored in the project SDB.

Adding and Editing Properties (Edit Component)

Click **Add** to add a property to a component definition. In the **Add Property** dialog box, you can enter the property name, value, and value type (text input, menu, check box, and so on). Depending on the value type, you can also choose a unit type (such as angle, frequency, and capacitance) and the default units (deg, GHz, pF, and so on) for the property. The **Value** field supports multi-line text which you can add either by pasting, or by direct entry (use **Ctrl+Enter** to insert carriage returns).

An informational panel provides guidance for entering data in the various fields

> **Warning:**
>
> Do not use the names of reserved system parameters for property names.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Click **Edit** to edit a component property. Editable fields are the same as those described above for the **Add Property** dialog box.

Removing Properties (Edit Component)

The **Edit Component** > **Properties** dialog box lets you remove a component property. Select the property and click **Remove**, then click **OK**. The property is deleted from the component definition (model) and from all instances in the design.

> **Warning:**
>
> Removing a property from a Twin Builder component library may produce undesirable results when the component is simulated.

Click **OK** to save your changes and close the dialog box, or click **Cancel** to close the dialog box without saving your changes.

The Component Netlist String Property

When a Twin Builder analysis is requested for a modified design, or when **Browse Netlist** is invoked, Twin Builder writes a text-based *netlist* to the project results folder that conveys to the simulation engine the connectivity, parameter values, analysis specifications, and other characteristics of the designs under analysis.

To support this process, every Twin Builder component must include a valid definition for a property called **SimplorerNetlist**.

To view the **SimplorerNetlist** for a component:

- Open its **Properties** dialog box.
- Select the **Parameter Values** tab.
- Click the **Shown Hidden** box.
- Click **Value** for **SimplorerNetlist**.

The **Edit Netlist String** dialog box opens, displaying the netlist string. Use the dialog box to edit the string.

See Also: The Netlist String Syntax Reference

Netlist String Syntax Reference

The general form for a netlist string is:

```
<Simulator>Netlist=[netlist_string]
```

where `<Simulator>` is Simplorer, and so on.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The netlist string may contain the following netlist property syntax:

%<terminal index, 0-based>

Name of net connected to terminal index.

(Syntax error if terminal index is not a positive number, within range.)

Example:

```
%0, %1
```

%_<internal node index, 0-based>

Create a unique net for an internal node. Used when a component has nodes that are not associated with symbol terminals and are not tied to a global node.

(Syntax error if node index is not a positive number, within range.)

Example:

```
%_0, %_1
```

@<propname> | @(<propname>)

Value of property named **propname**.

(Syntax error if **propname** is empty or contains spaces.)

Example:

```
@R, @IDSS
```

?<propname>(<expr1>)[:(<expr2>)]

If property named **propname** exists, substitute **expr1**, else (optionally) **expr2**. **expr1** and **expr2** may contain additional substitutions.

(Syntax error if **propname** is empty or contains spaces.)

Example:

```
?IDSS(IDSS = @IDSS):(IDSS = 0.05)
```

~<propname>(<expr1>)[:(<expr2>)]

If property named **propname** doesn't exist, substitute **expr1**, else (optionally) **expr2**.

(Syntax error if spaces in **propname** or **propname** is empty.)

Example:

```
~SUB(SUB = MS)
```

?(<propname>==<value>)(<expr1>)[:(<expr2>)]

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

?(<propname>!=<value>)(<expr1>)[:(<expr2>)]

~(<propname>==<value>)(<expr1>)[:(<expr2>)]

~(<propname>!=<value>)(<expr1>)[:(<expr2>)]

Same as “?” and “~” above except the first term is evaluated for equal to (==) or not equal to (!=) and if true, substitute **expr1** else (optionally) **expr2**.

Example:

```
?(sim==fullwave)(NSUM=@NSUM):(F0=@F0)
```

***<propname>(<expr1>)**

If property named **propname** has changed from default (definition value), then substitute **expr1**.

(Syntax error if spaces in **propname** or **propname** is empty.)

Example:

```
*IDSS(IDSS = @IDSS)
```

&(<expr>)[^(pname1,pname2, . . .)]

Add all properties that have changed from default (definition value), except those in the optional exclusion list. In expr, $ can be used to represent the property name, and # the property value.

Example:

```
&($=#)^(Model)
```

will netlist all properties that changed from their default value except the **Model** property.

- \n

A new line marker to inform the netlister to insert a new line.

Example:

```
R@ID %0 %_0 @R \n C@ID %_0 %1 @C
```

A new line marker to inform the netlister to insert a new line.

Example:

```
R@ID %0 %_0 @R \n C@ID %_0 %1 @C
```

****

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The backslash is used as an escape character. The character following the backslash is not processed but added to the netlist string as is. To add a single '\', use '\\'.

Example:

```
R:@ID %0 %1 R=\{\2\*@R\}
```

Defined Variables

$SYSLIB will be expanded to <*InstallationDirectory>/syslib* upon netlisting, where <*Installation Directory*> is the Twin Builder installation directory.

$USERLIB will be expanded to **<InstallationDirectory>/userlib** upon netlisting.

$PERSONALLIB will be expanded to **<ProjectDirectory>/PersonalLib** upon netlisting, where **<ProjectDirectory>** is the location you specified for your Twin Builder projects.

Global Reference String in Twin Builder

A global reference property will process its value the same way as the netlist string and will place the result in the top-level (global) part of the circuit file.

```
<Simulator>GlobalRef=[string]
```

where <Simulator> is Simplorer, and so on

Examples:

```
SimplorerNetlist=CPLE:@ID %0 %1 %2 %3 ?Z(Z=@Z) ?E(E=@E) ?F(F=@F) *A
(A=@A)
```

Z, E, and F use the “?” conditional because the component definition values assigned to these parameters are different from those in the engine and should always be netlisted even if you have not changed them. The A parameter uses the “*” conditional because its component definition value matches that of the engine and we just need to netlist it if you change the instance value.

```
SimplorerNetlist=R@ID %0 %1 @R ?TC1(TC1=@TC1 ?TC2(tc2=@TC2)) ?TJ
(TJ=@TJ) ?TNOM(TNOM=@TNOM)
```

In this example, TC1 will be netlisted only if TC1 exists, and TC2 will be netlisted only if TC1 and TC2 exist.

```
SimplorerNetlist=MSBENDO:@ID %0 %1 w=@Length sub=@Substrate
```

W and Sub will always be netlisted.

```
SimplorerNetlist=MOSFET:@ID %0 %1 %2 %3 MODEL=@Model &($=#)^(Model)
```

Model will always be netlisted. The &($=#)^(Model) indicates to netlist all the parameters as name=value, except for Model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
SimplorerNetlist=D@ID %0 %1 1N914
SimplorerGlobalRef =.LIB $SYSLIB/Vendor/D.lib
```

which may netlist to:

```
D22 net_3 net_4 1N914
< rest of design netlists ...>
.LIB $SYSLIB/Vendor/D.lib
SimplorerNetlist=R:@ID %0 %1 R=\{\2\*@R\
```

If ID is 1 and R is 100, the netlist will be:

```
R:1 net_0 net_1 R={2*100}
```

Using the Symbol Editor

A symbol is a set of information that defines the graphical representation and electrical connectivity of a component for inclusion in a schematic. Every component in a model library has at least one symbol, which is displayed when the component is placed on a schematic sheet. Component symbols are language-dependent. In other words, each language (SML, VHDL-AMS, C) can provide its own specific symbol. Symbol definitions are stored in library files with an extension of **.aslb**.

Creating and Editing Symbols

Creating or editing a symbol involves using the symbol editor to:

- Draw graphical primitives, such as rectangles, circles, and arcs, using options on the **Draw** menu
- Add pins for electrical connections using the **Pin** option on the **Draw** menu, and possibly modifying the properties of these pins via the **Pin List** dialog box.
- Add text labels, using the **Text** option on the **Draw** menu
- Add property displays, using the **Property Display Setup** option on the **Symbol** menu.
- Update the current project with the new or revised symbol definition using the **Update Project** option on the **Symbol** menu

You may also want to export a symbol to a symbol library (**.aslb**) file for use in other projects. For information on how to do this, see **Managing Library Contents**.

Editing an Existing Symbol

To edit an existing symbol, do either of the following:

- In the Project tree, expand the **Definitions** folder and the **Symbols** subfolder for the project that contains the symbol you want to edit.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Double-click the entry for the symbol you want to edit, or right-click the entry and select **Edit Symbol**.

The symbol editor runs and opens the selected symbol for editing.

- Click **Tools** > **Edit Libraries** > **Symbol** to open the **Edit Libraries** dialog box to the **Symbols** tab, and locate and select the Project version of the symbol to edit.

  Click **Edit Symbol**, or double-click the selected entry.

  The symbol editor opens the selected symbol for editing.

Creating a New Symbol

To create a new symbol, do either of the following:

- Click **Tools** > **Edit Libraries** > **Symbol** to open the **Edit Libraries** dialog box to the **Symbols** tab and click **Add Symbol**. The **Get Name** dialog box opens.

  Type a name for the symbol into the **Enter the name for this new Symbol** box and click **OK** or press **Enter**.

  Double-click the entry for the new symbol, or click **Edit Symbol**. The symbol editor opens.

- Click **Tools** > **Edit Libraries** > **Symbol** to open the **Edit Libraries** dialog box to the **Symbols** tab, and locate and select an existing symbol on which you would like to base your new symbol.

  Click **Clone Symbol(s)**. The **Get Name** dialog box opens.

  Type a name for the new symbol into the **Enter the name for this new Symbol** box and click **OK**.

  Double-click the entry for the new symbol, or click **Edit Symbol**. The symbol editor opens.

The Symbol Editor

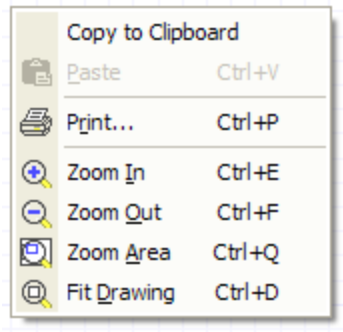
The symbol editor opens when you create or modify a symbol. Like the schematic editor, the symbol editor keeps the placement of symbol elements uniform by snapping them to points on a grid. The window controls for zooming and panning the view are the same as those in the schematic editor.

With the symbol editor, you can add, modify, and manipulate graphical primitives, text labels, pins, and property displays. You can also adjust the resolution, color, and visibility of the editor grid. You can create several symbol levels, which you can use to control symbol visibility. When you're done, you can update the current project with any changes you've made to the current symbol. Options related to these operations are available on the **Symbol** menu and the **Symbol Draw** menu.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

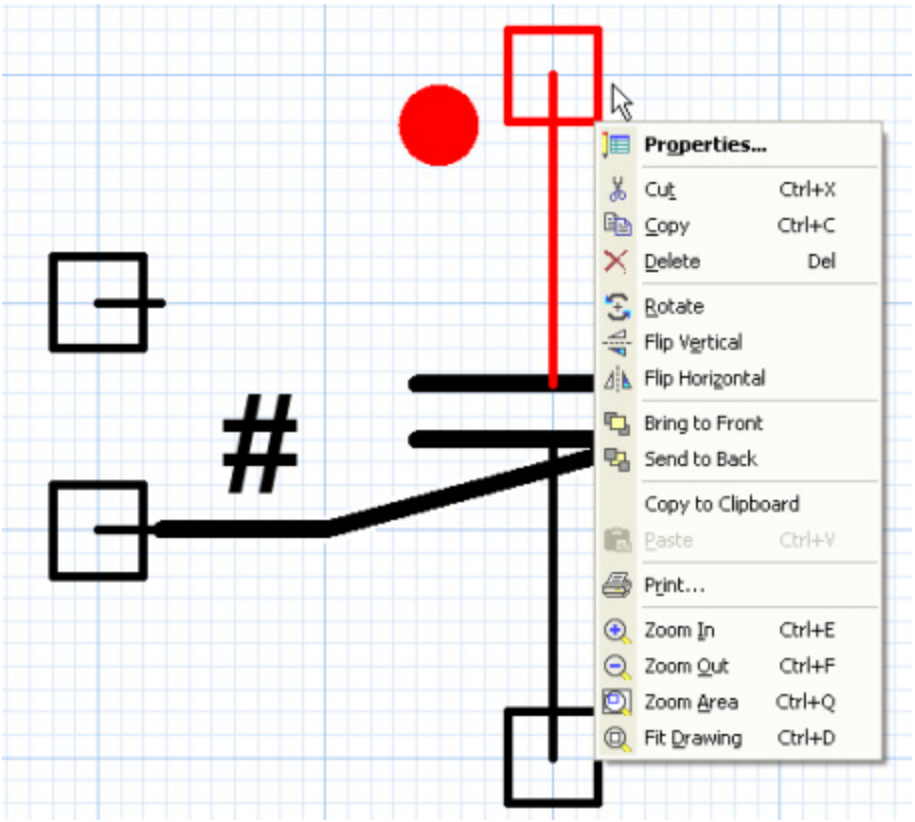
Symbol Editor Shortcut Menus

When you right-click in the symbol editor grid, this shortcut menu appears:



These options are a subset of those available on the **View** menu.

When you right-click an object in the symbol editor, this shortcut menu appears:



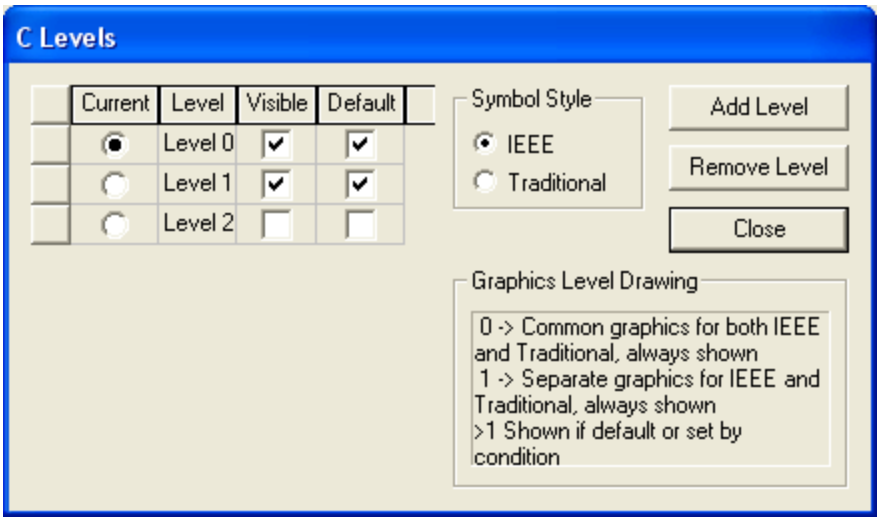
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Graphic objects placed in the symbol editor have various properties associated with them such as color, fill style, and line weight. Click **Properties** to view the properties of the selected object. The other options are a combination of entries from the **Edit**, **View**, and **Draw** menus.

Symbol Graphic Object Levels

There are two display styles available for symbols shown in the schematic editor: IEEE and Traditional. The display style is set on the **Schematic Editor Options** > **General** tab. Each style contains two or more active levels for the graphic objects that comprise a symbol. Each graphic object in a symbol is associated with a level.

Graphic object levels are managed in the Symbol Editor by selecting **View** > **Levels**, which opens the **Levels** dialog box for the symbol currently being edited.



Levels may be shown or not depending on external settings.

- Level 0 graphics are common to both IEEE and Traditional symbol styles, and are always shown.
- All symbol pins are on level 0, and thus are common to both styles.
- Objects placed on level 1 in the IEEE style are always shown on the schematic when you choose the IEEE display style. Similarly, objects placed on level 1 in the traditional style are always shown on the schematic when you choose the traditional style option.
- Levels greater than 1 are independent for each style. Click **Add Level** to add any number of levels, for IEEE, Traditional, or both styles. Visibility of graphics on additional levels may be controlled with the **Default** check box in the **Levels** dialog box, in a special component dialog box, or programmatically. In Twin Builder, you can control visibility by setting up Boolean conditions using quantities, then evaluating the conditions as a simulation progresses.

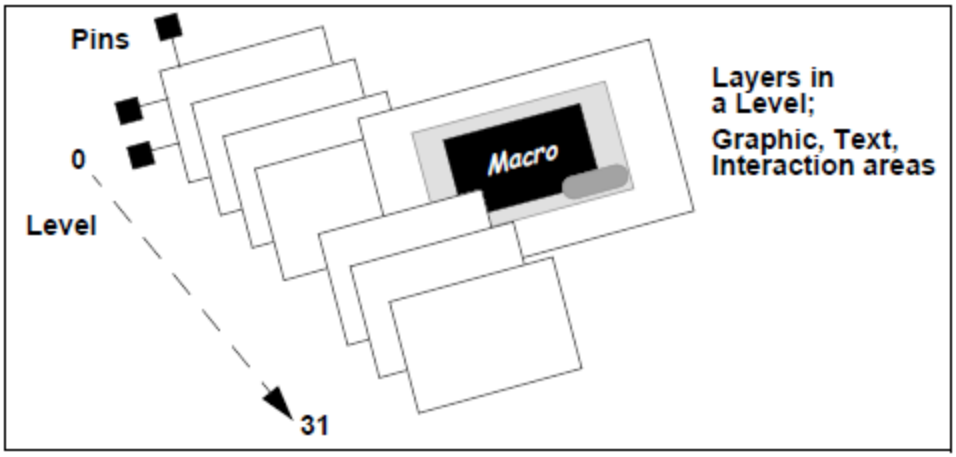
To set the level and style of a graphic object:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Choose the **Level** by selecting the radio button next to the desired level.
2. Select **IEEE** or **Traditional** to choose the desired **Symbol Style**.
3. Select and place the desired graphic object (circle, line, pin, text, and so on) from the **Symbol Draw** menu.
4. When finished, **Close** the symbol **Levels** dialog box.

Arranging Symbol Elements

You can place graphic elements and interactions on different symbol levels (0-31) and on different layers within a level. Within a level you can change the order of elements with the **Symbol Draw** menu. In the **Symbol Levels** dialog box, you can assign the level to an element. Pins are always placed on the '0' level and snapped to the grid.



The Draw Menu

The symbol editor **Draw** menu presents options for drawing graphical primitives and text, and for manipulating selected objects. The **Draw** menu options include:

- Arc
- Circle
- Line
- Polygon
- Rectangle
- Text
- Image
- Pin
- Rotate
- Align Horizontal

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Align Vertical
- Flip Vertical
- Flip Horizontal
- Bring to Front
- Send to Back

Arc

Use this option to create an arc. In addition to the **Draw** menu, you can also select **Schematic** > **Graphics** > **Arc**on the ribbon. To draw an arc, click the symbol editor grid at the two points that determine the arc's ends, then drag the arc to the desired radius.

Once you have created an arc, you can adjust its radius or move its endpoints: click the arc to select it and drag the appropriate handle.
'm in

Circle

Use this option to create a circle. In addition to the **Draw** menu, you can also select **Schematic** > **Graphics** > **Circle** on the ribbon. To draw a circle, click the symbol editor grid to select a center point and drag the circle to the desired diameter.

Once you have created a circle, you can adjust its diameter in either of two ways:

- Click a circle and drag one of its handles.
- Double-click the circle to open its **Properties** dialog box, type a new value for the **Radius** parameter and click **OK**.

By default, a new circle is hollow. You can fill a circle with solid color or parallel lines in one of several styles as follows:

a. With the circle's properties displayed, click the **Value** cell for the **FillStyle** property.
b. Select the desired fill style from the list.
c. Click **OK**.

Line

Use this option to create a line with one or more segments. In addition to the **Draw** menu, you can also select **Schematic** > **Graphics** > **Line** on the ribbon. To draw a line:

1. Click the symbol editor grid where you want the line to start, then click at one or more points to continue the line.
2. To complete the line, do one of the following after defining its final segment:
   - Press the space bar.
   - Right-click and click **Finish**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. After you have completed a line, you can change the endpoints of its segments by doing the one of the following:
   - Select the line and drag the appropriate handles.
   - Double-click the line to open its **Properties** dialog box, type new position values (in the form X: Y) for the desired vertices, and click **OK**.
4. Use the **Properties** dialog box to set the color and width of the line, as well as the line style (solid, dash, dot, dash dot, or dash dot dot).
5. You can also choose the object displayed at the beginning and ending of the line. Choices include a short perpendicular line, arrow, solid arrow, square, diamond, and circle.

Polygon

Use this option to create a polygon. In addition to the **Draw** menu, you can also select **Schematic** > **Graphics** > **Polygon** on the ribbon. To create a polygon:

1. Click the symbol editor grid to specify the position of one vertex, then click wherever you want to place additional vertices.
2. To complete a polygon, specify the position of its final vertex and do either of the following:
   - Press **SPACE**.
   - Right-click and click **Finish**.
3. Once you have drawn a polygon, you can edit its vertex positions, border width, and other properties, including its fill style, as follows:
   - Click the polygon and edit its properties in the **Properties** pane.
   - Double-click the polygon and edit its properties in the **Properties** dialog box.
4. By default, a new polygon is hollow. You can fill a polygon with solid color or parallel lines in one of several styles as follows:
   a. With the polygon's properties displayed, click in the **Value** cell for the **FillStyle** property.
   b. Select the desired fill style from the list.
   c. Click **OK**.

Rectangle

This option, also available by selecting **Schematic** > **Graphics** > **Rectangle** on the ribbon, initiates creation of a rectangle. To create a rectangle, click the editor grid to specify the position of one corner, then drag the rectangle to the desired size.

Once you have drawn a rectangle, you can edit its height and width, the position of its center, and its angle (its rotation, in degrees, relative the handles of its bounding box) as follows:

- Click the rectangle and edit its properties in the **Properties** pane.
- Double-click the rectangle and edit its properties in the **Properties** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

By default, a new rectangle is hollow. You can fill a rectangle with solid color or parallel lines in one of several styles as follows:

a. With the rectangle's properties displayed, click in the **Value** cell for the **FillStyle** property.
b. Select the desired fill style from the list.
c. Click **OK**.

Text

This option, also available by selecting **Schematic** > **Graphics** > **Text** on the ribbon, adds to the symbol an editable text label in 12-point Arial. To edit the text string immediately after placement, type the text you want, then press **Enter** or click elsewhere in the editor grid.

To change just the text of an existing label:

1. Click the label.
2. Click the label again to open its text for editing.
3. Type the desired text.
4. Press **Enter** or click elsewhere in the editor grid.

Other properties of a label, include its color, font, size, angle of rotation, and justification. You can also add a box around the text. The box's fill color and style, border color and width, the position of its center, its width, height, and angle of rotation can be set independently of the associated text. To edit these properties:

1. Do one of the following:
   - Click the label and view its properties in the **Properties** pane.
   - Double-click the label and view its properties in the **Properties** dialog box.
   - Right-click the label and select **Properties** to view its properties in the **Properties** dialog box.
2. Click in the **Value** cell for the property you want to modify.
3. Modify the value.
4. Click **OK**, or click in another **Value** cell to commit the change and keep editing values.

Image

This option, also available by selecting **Schematic** > **Graphics** > **Image** on the ribbon, lets you select an external **.bmp** or **.gif** image file and place it on the symbol grid.

Follow this procedure to add a bitmap or GIF image to a symbol:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Draw** > **Image** or click **Image** under **Graphics** on the **Schematic** ribbon.

   This displays a file browser. It defaults to your **PersonalLib** location, but lets you use buttons to select **Syslib**, **UserLib**, **Project Folder**, file path, or standard browser functions to look through your file system and network.

2. Specify the file format as bitmap (.**bmp**) or GIF (.**gif**).
3. Click **Open** to import the image to the schematic.
4. Position the cursor at the desired location and click. The image appears at the cursor location, remaining selected so you can drag the cursor to resize it. Resizing maintains the original proportions.
5. Click again to anchor the image. The image remains selected, with the drag handles on the corners visible. If necessary, you can select a handle to resize the image.

> **Note:**
>
> The image includes a **Properties** window in which you can edit the image's properties, including center coordinates, angle of rotation, width, and height. A check box toggles display of a border and enables additional border properties including border color and width. Use the additional check boxes to mirror the image left-to-right, and to establish a link to the image file.

Pin

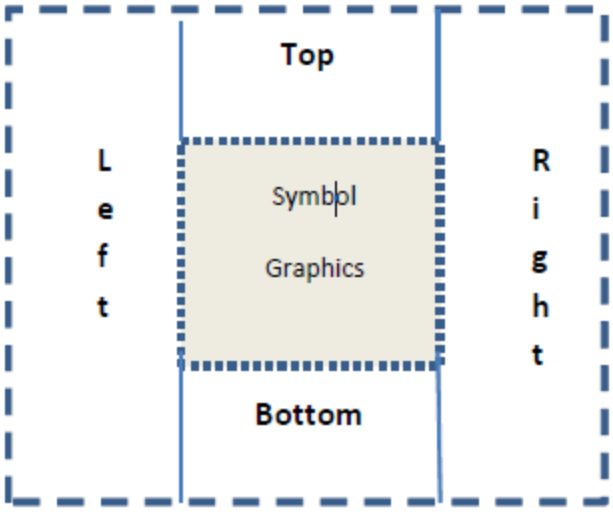
This option, also available from the **Symbol Draw** toolbar, initiates placement of a symbol pin with a default stem length of 100 mil. Press **R** to rotate a pin in 90 degree increments before placing it. Click the editor grid to finish placing the pin. Pins are always placed on the "0" level and snapped to the grid. After placement, press **Ctrl-R** to rotate the pin in 90 degree increments. You can also move and reorient pins as described below.

Adjusting Pin Orientation

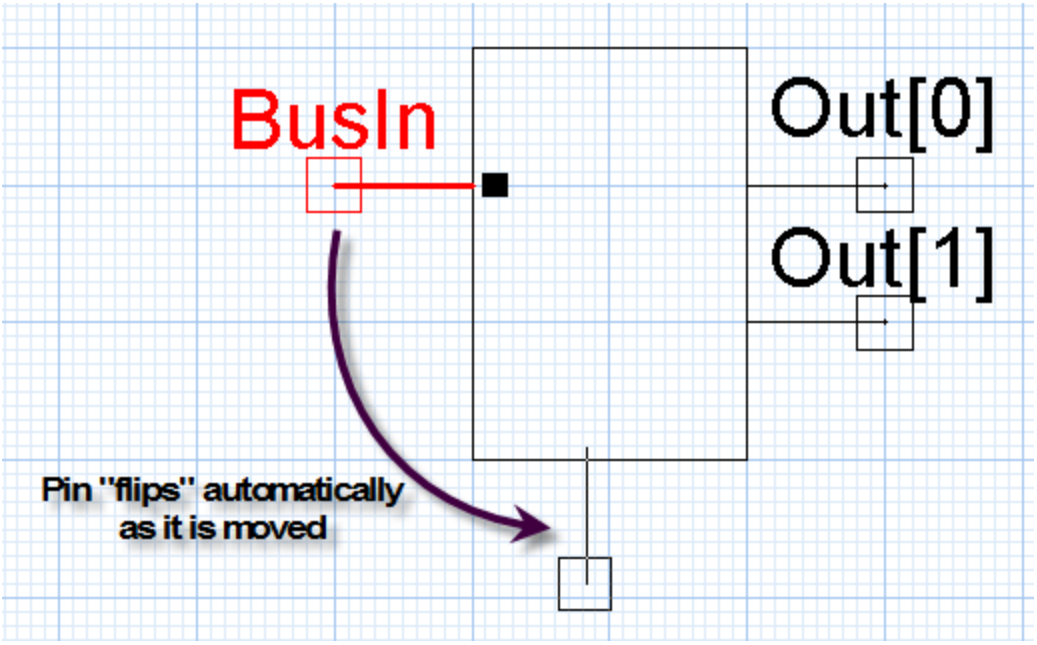
- If the **Adjust Pin Orientation on Drag** option in the ***Schematic Editor Options: Symbol Editor Options Tab*** is enabled, you can drag pins from one side of a symbol

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

to another using the mouse,



and the pins rotate or flip to the appropriate orientation.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Ctrl+click to select and drag multiple pins from one side of a symbol to another.

  > **Note:**
  >
  > If multiple selected items include symbol graphics in addition to pins, then automatic pin orientation is disabled.

- Press the Alt key while dragging the pins to temporarily disable the automatic pin orientation function if **Adjust Pin Orientation on Drag** is enabled. Conversely, if **Adjust Pin Orientation on Drag** is disabled, pressing the Alt key while dragging the pin temporarily *enables* automatic pin orientation.

Other properties of the pin include its name, color, level, type, length, visibility of the pin name and number, and the circuit node to which the pin will be connected when hidden. See ***Editing Pin Properties*** for detailed explanations of these properties.

To edit these properties:

1. Do one of the following:
   a. Click the pin and view its properties in the **Properties** pane.
   b. Double-click the pin and view its properties in the **Properties** dialog box.
   c. Right-click the pin and select **Properties** to view its properties in the **Properties** dialog box.
2. Click the **Value** cell for the property you want to modify.
3. Modify the value.
4. Click **OK**, or click in another **Value** cell to commit the change and keep editing values.

> **Note:**
>
> If you make a pin name or number visible, the text can be selected, moved, rotated, and modified like any other text label. However, a pin's name and number remain linked to the associated pin.

Rotate

This option, also available from the **Draw** menu and by pressing **Ctrl+R**, rotates a selected object or group of objects 90° counterclockwise.

Align Horizontal

This option horizontally aligns selected components, property displays, or combination of components and property displays with the first object you selected.

To align multiple objects horizontally:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Hold **Ctrl** and click the object with which you want to align the others.
2. Continue to hold **Ctrl** and click the additional objects in turn to add them to the selection.

> **Note:**
>
> The first-selected object is highlighted in red, and the subsequently selected objects are highlighted in dark red.

3. Click **Draw** > **Align Horizontal**. The selected objects align horizontally with the first selection.

Align Vertical

This option vertically aligns selected components, property displays, or combination of components and property displays with the first object you selected.

To align multiple objects vertically:

1. Hold **Ctrl** and click the object with which you want to align the others.
2. Continue to hold **Ctrl** and click the additional objects in turn to add them to the selection.

> **Note:**
>
> The first-selected object is highlighted in red, and the subsequently selected objects are highlighted in dark red.

3. Click **Draw** > **Align Vertical**. The selected objects align vertically with the first selection.

Flip Vertical

This option, also available by selecting **Schematic** > **Flip Vertical** on the ribbon, flips the selected object or group of objects around the X-axis.

Flip Horizontal

This option, also available by selecting **Schematic** > **Flip Horizontal** on the ribbon, flips the selected object or group of objects around the Y-axis.

Bring to Front

This option moves the selected object to the front of the drawing.

Send to Back

This option moves the selected object to the back of the drawing.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The Symbol Menu

The **Symbol** menu lists options for symbol naming, property displays, and file operations.

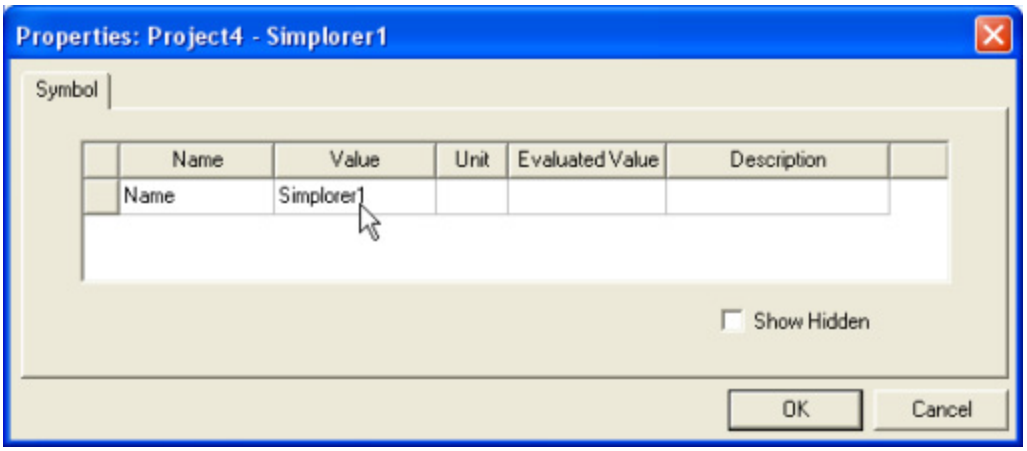
Update Project

Click **Symbol** > **Update Project** to update the current project with changes made in the current symbol.

> **Note:**
>
> **Update Project** updates your symbol changes to the current project in memory and does not save your changes to disk. To save your changes to disk, click **File** > **Save**. If you would like the changes you have made in a symbol to be available for use in other projects, you must export the symbol to a library as described in **Managing Library Contents**.

Set Name

Click **Symbol** > **Set Name** to open a **Properties** dialog box in which you can rename the current symbol.



Symbol Property Display Setup

Click **Symbol** > **Property Display Setup** to open a **Properties** dialog box in which you can add or modify how the names, values, and locations of properties associated with the component or object (such as a title block) represented by the current symbol are displayed.

The Symbol Property Displays Dialog Box

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

This dialog box lists symbol display properties and controls their presence, visibility, and position.

- **Add** – Add a new property display entry to the list.
- **Change Component Context** – Open the **Select Definition** dialog box, through which you can select the component definition on which property displays are based. The properties of the selected component are then available in the **Name** cell properties list.

  > **Note:**
  >
  > Because a symbol can be used by multiple components with differing properties, you can enter any property name. If a component using the symbol doesn't have a property with that name, the symbol property isn't used in that component.

- **Location** – Set the location (**Left**, **Top**, **Right**, **Bottom**, **Center**) of the property display relative to the symbol center.

  > **Note:**
  >
  > You can further position a property display by rotating it (**Ctrl-R**), or by dragging it to another location. When you do this for a given property display, its **Location** value is set to **Custom**.

- **Name** – Type a property display name, or select an existing property display from a list. Click **Change Component Context** to determine which property displays appear for the selected component.
- **Remove** – Remove the selected property display.
- **Visibility** – Set the visibility (**None**, **Name**, **Value**, or **Both**) of a property display.
- **Set Title Block Context** – Make the list of default page properties, and others as specified, available for display. Default properties include ProjectPath, Project, Design, Title, Author and Date. This context is selected by default if the symbol has three or more propdisplays for default page properties.

Pin List (Symbol)

Click **Symbol** > **Pin List** to access settings related to pin properties by opening the **Pin List** dialog box.

Grid Setup (Symbol)

Click **Symbol** > **Grid Setup** to access settings related to the symbol editor grid resolution, color, and visibility by opening the **Grid Setup** dialog box.

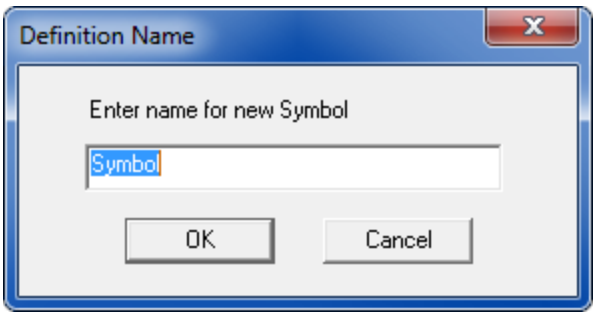
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Import File (Symbol)

Click **Symbol** > **Import File** to import an existing Scalable Vector Graphics (SVG) formatted symbol into the symbol editor. The SVG format is an open, XML-based web standard graphics format supported by many third-party tools.

To import an SVG (**.svg**) file when creating a new symbol definition:

1. In the Project Manager, right-click the **Definitions** > **Symbols** folder in the desired project and select **Add Definition**.
2. Enter a name for the new symbol in the **Definition Name** dialog box and click **OK**.



   A new blank **Symbol Editor** window opens.

3. On the main menu bar, click **Symbol** > **Import File** and choose an existing SVG file to import it into the symbol editor.

   A dialog box appears advising you that importing the **.svg** file will remove any existing symbol graphics in the editor, and asking you if you are sure you want to continue the import.

Note:

- You cannot undo the import. If you make a mistake on import, you must close the project without saving and reopen it to recover.
- You can import SVG files into any symbol editor window, including those containing pre-existing component symbols. The current symbol in the editor is replaced by the imported symbol, so make sure that you import the symbol in the desired place.

4. Click **OK** to import the SVG file for further editing, or **Cancel** to abort the operation.

   Once the new symbol is created, you can attach it to a component with Property Display Setup.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Export File (Symbol)

Click **Symbol** > **Export File** to save the current symbol to a graphics file for use in other applications. You can export to the following graphics formats:

- Microsoft Enhanced Meta File (**.emf**)
- Scalable Vector Graphics (**.svg**) – Scalable Vector Graphics (SVG) is an open, XML-based web standard supported by many third-party tools.

Edit Component (Symbol)

Click **Symbol** > **Edit Component** to open the **Edit Component** dialog box to modify the component associated with the active symbol.

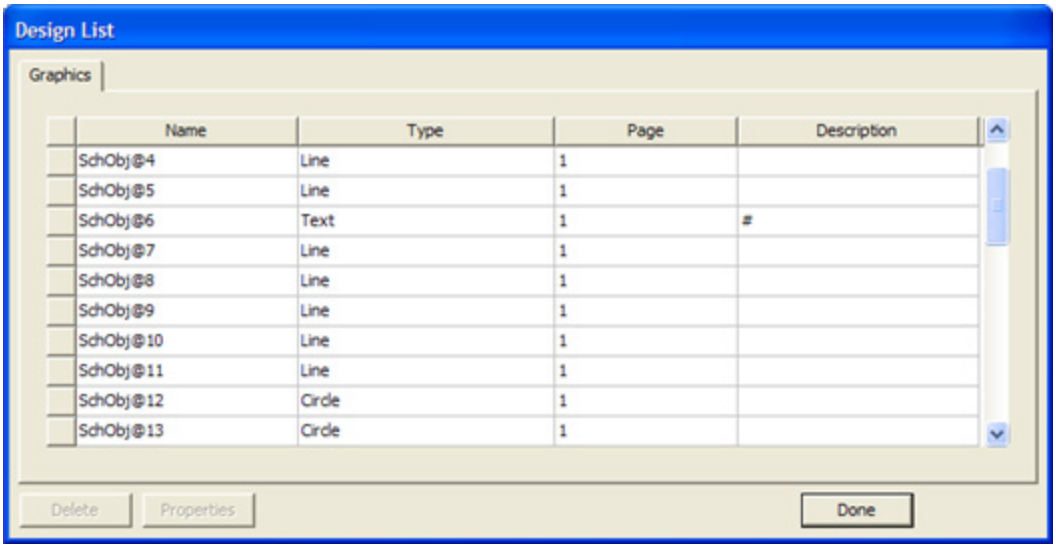
Animate

Click **Symbol** > **Animate** to open the **Animation** dialog box where you can add, edit, and delete expressions for animating the symbol. Animations embedded in a symbol change the symbol appearance during the simulation process in response to a change in value of a system quantity.

- The **Expression** control opens the **Edit Condition** dialog box in which you can add or edit conditions that trigger animation events.
- The **Activate Level(s)** control opens the **Select Levels** dialog box where you can choose which symbol levels are shown when a condition is satisfied.

List Symbol Graphic Elements

Click **Symbol** > **List** to open the **Design List** dialog box, which lists the graphics elements of a symbol.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Name** – The graphic ID.
- **Type** – The graphic type, such as circle, rectangle, arc, text, and pin.
- **Page** – The page number on which that graphic is located.
- **Description** – The descriptive text for text and pin graphic types; this column is blank for non-textual graphics.

See *Listing Design Elements* for additional information.

Editing Pin Properties

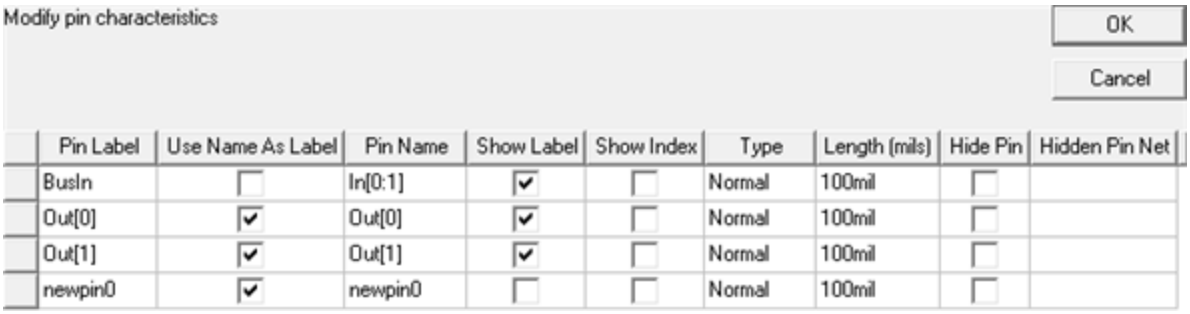
To edit a property of a symbol pin, open the symbol editor's **Pin List** dialog box by doing one of the following:

- Click **Symbol** > **Pin List**.
- Click **Symbol** > **Edit Pins**.

In addition to the pin properties described below for the **Pin List** dialog box, you can also set the pin color and level in the pin's **Properties** window and **Properties** dialog box.

The Pin List Dialog Box

The **Pin List** dialog box displays and sets values for the properties of symbol pins.



Pin Label	Use Name As Label	Pin Name	Show Label	Show Index	Type	Length (mils)	Hide Pin	Hidden Pin Net
BusIn	☐	In[0:1]	☑	☐	Normal	100mil	☐	
Out[0]	☑	Out[0]	☑	☐	Normal	100mil	☐	
Out[1]	☑	Out[1]	☑	☐	Normal	100mil	☐	
newpin0	☑	newpin0	☐	☐	Normal	100mil	☐	

Pin properties include:

- **Pin Label** – Provides a field to add a user-defined label for the chosen pin on a component symbol without having to change the actual **Pin Name** associated with the chosen pin. Pin labels for pins in a component are unique.

  To specify or change the label of a pin, click the **Pin Label** cell for that pin, type the name, and

  - Press **Enter**, or
  - Click in another cell.

  Display the **Pin Label** for the chosen pin by clearing the **Use Name As Label** check box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Use Name As Label** – Controls whether the **Pin Name** or **Pin Label** displays for the chosen pin when **Show Label** is enabled. For a new pin, **Use Name As Label** is selected by default.
- **Pin Name** – Sets the actual pin name. To specify or change the name of a pin, click in the **Pin Name** cell for that pin, type the name, and
  - Press **Enter**, or
  - Click in another cell.
- **Show Label** – Controls visibility of the either the **Pin Name** or **Pin Label** text for the pin (as determined by **Use Name As Label**). For a new pin, it is cleared by default.
- **Show Index** – Controls visibility of the pin number in the schematic editor. For a new pin, it is cleared by default.
- **Type** – Drop-down list sets the pin type. Options include **Normal**, **ANSI In**, **ANSI Out**, and **Zero Length**. To change a pin's type, click in its **Type** cell and click the desired option.
- **Length (mils)** – Sets the length of the pin stem (the graphical line associated with the pin port symbol). By default, new pins have a pin stem length of 100 mils. To change the length of a pin, click its **Length** cell, type a new value, and either:
  - Press **Enter**.
  - Click in another cell.
- **Hide Pin** – Controls whether a pin is visible in the schematic editor. For new pins, it is cleared by default.
- **Hidden Pin Net** – Editable when **Hide Pin** is selected, specifies the net (circuit node) to which a pin will be connected if it is hidden. To specify a net, click in the **Hidden Pin Net** cell, type a net name, then
  - Press **Enter**, or
  - Click in another cell.

Using Pin Labels

Pin labels provide a way to display user-defined text labels for selected pins on a component symbol without having to change the actual pin names, port names, and terminal names associated with the selected pins.

- Pin labels can be displayed on the chosen pin by changing the **PinLabel** property with a user-defined name and clearing the **UseNameAsLabel**check box.
- The **UseNameAsLabel** check box provides a mechanism to either use **PinLabel** or **PinName** as the display text on the schematic and symbol editor canvas.
- The **ShowLabel** check box controls the visibility of chosen label text on the pin.
- Pin labels for pins in a component are unique.

Changing Pin Properties

To change a pin property:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Click the cell that displays the pin property value that you want to change.
2. Modify the property value appropriate to its type as described in *The Pin List Dialog Box*.
3. Do one of the following:
   - To commit the new value and continue editing pin properties, click another property value.
   - To commit the new value and stop editing pin properties, click **OK**.

Using the Script Editor

The **Script Editor** supports the viewing, modification, saving, printing, and exporting of script file information for use by Twin Builder. After modifying a script, the information can be saved, used to render a footprint, or exported.

Related Topics

Starting the Script Editor

Printing

Script Editor Menu

Script Editor Ribbon

Text Editor Options

Starting the Script Editor

To invoke the **Script Editor** do either of the following:

- Double-click a script listed beneath the **Definitions/Scripts** folder in the **Project Manager** pane.
- Select **Edit Script** in the **Tools** > **Edit Libraries** > **Scripts** dialog box.

This opens the **Script Editor** dialog box. Text displayed within the editor is highlighted to allow for easy identification of the various structures of the displayed script.

Script Editor Menu

After you invoke the **Script Editor**, the following options are listed in the **Script** menu.

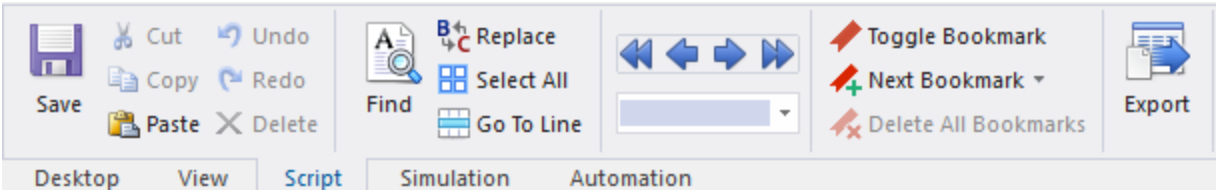
The following options are available:

- **Export Script** – Exports the script to a file. See *Exporting a Netlist or Script* for additional information.
- **Save Script** – Saves the script.
- **JavaScript/VBScript** – Controls the syntax highlighting for the selected language.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Script Editor Ribbon

When a new or pre-existing script file is opened, the **Script** ribbon displays.



Ribbon Symbol	Symbol Function	Definition
	Toggle bookmark	Inserts a bookmark at the active cursor position, or turns off an existing bookmark.
	Delete bookmarks	Deletes all bookmarks in the text.
	Next Bookmark	Moves the cursor to the next bookmark in the text.
	Previous Bookmark	Moves the cursor to the previous bookmark in the text.
	Find list	Displays the text string you have entered to find. Click the drop-down arrow to display previously searched for text strings.
	Forward search	Searches forward in the file for text entered in the find list.
	Backward search	Searches backward in the file for text entered in the find list.
	Forward search, case-sensitive	Searches forward in the file for text entered in the find list with case-sensitive control.
	Backward search, case-sensitive	Searches backward in the file for text entered in the find list with case-sensitive control.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Using the C-Model Editor

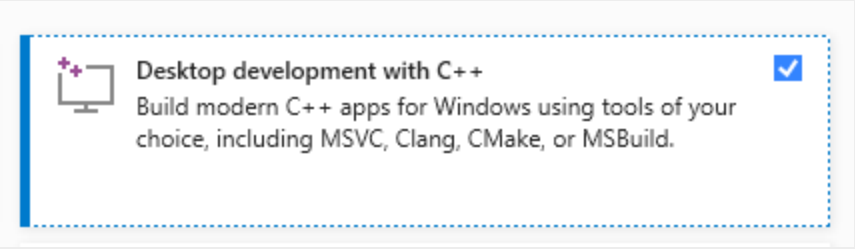
The **C-Model Editor** facilitates the development and extension of C-Models for Twin Builder by generating much of the C++ code required to implement a Twin Builder C-Model. Syntax checking and generation of Dynamically Linked Library (DLL) files for models are also managed by the **C-Model Editor**. You can also print the C-Model listing.

Developing C-Models requires that one of these C++ compilers is installed:

- Microsoft® Visual Studio Professional or Community Version
  - Visual Studio or Visual Studio Community 2022
  - Visual Studio or Visual Studio Community 2019
  - Visual Studio or Visual Studio Community 2017
  - Visual Studio or Visual Studio Community 2015

Note:

- If no professional version is available, we recommend that you use the Visual Studio Community version.
- You must install the workload package **Desktop development with C++** before using Visual Studio 2017 (or later) C++ Compiler.



- By default, the compilation of C-models in the C-model editor does not require the MFC package to be installed. If the C-model does not require MFC functionality, set **Use MFC** to **No** in the C-model editor build settings. If needed, you must manually add the MFC package to the list of packages in the C++ development package, since it is not installed automatically by the VS installer.

Models generated using the **C-Model Editor** appear under **Project Components** in the **Project Manager** components tree and are used in the same way as other Twin Builder components. Because they are components, you can edit their properties using Twin Builder's **Component Editor**.

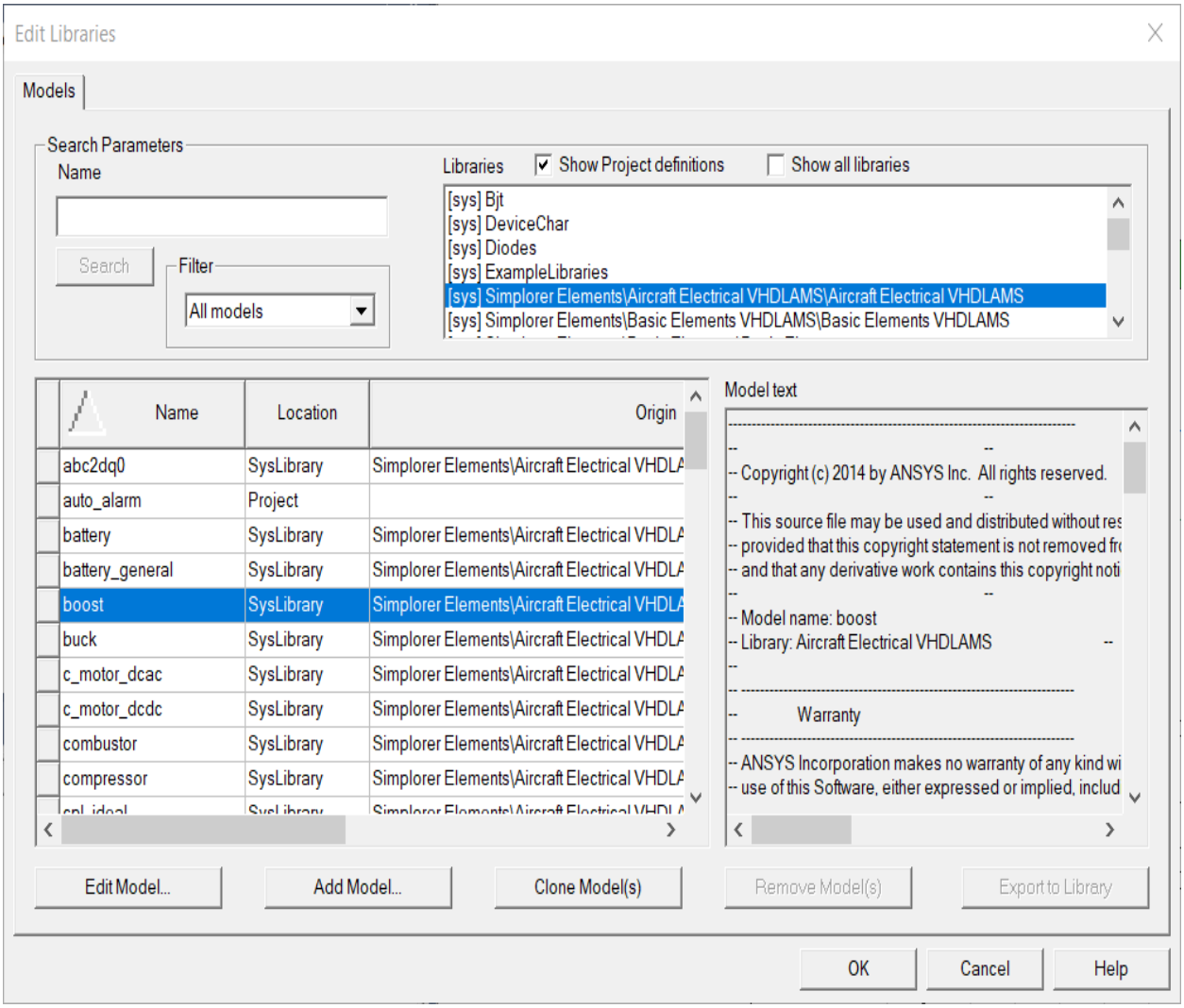
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For a detailed discussion of how to develop, program, debug, and deploy C-Models in Twin Builder, see ***C-Models in Twin Builder***.

Adding a C-Model

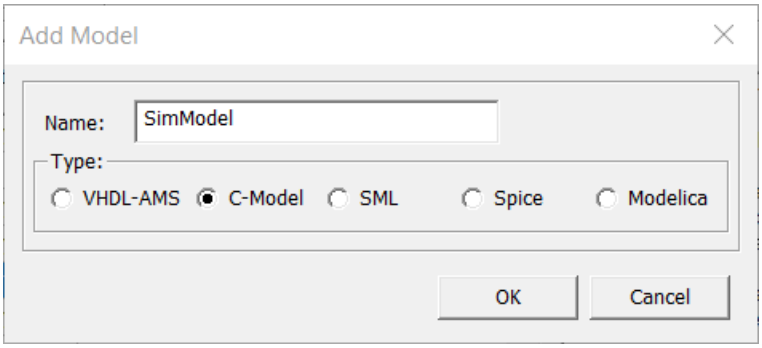
Follow this procedure to add a C-Model using Twin Builder's C-Model Editor.

1. Select **Tools** > **Edit Libraries** > **Models**. The **Edit Libraries** dialog box appears.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Click **Add Model** to open the **Add Model** dialog box.



3. Select the **C-Model** radio button.
4. Type the name of the new C-Model into the **Name** field. This name is applied to the project folder Twin Builder creates for the model's files, and also to the model's source files. The new folder and files reside in the **PersonalLib** folder. See **General Options: Directories** for information on designating a **PersonalLib** directory.

   The **C-Model Editor** opens and creates a C++ source (CPP) file for the C-Interface functions of the model, a C++ header (H) file for the macro definitions necessary for the model, and a C++ source file bearing the model name you chose above.

   By default the C++ source file for your model will already contain function declarations for the **Initialize**, **Simulate**, **Validate**, and **Close** functions for the transient analysis type.
5. Edit the model source files using the tools provided in the following topics:

   Adding Analysis Types, Nodes, States, Variables, and Models to a C-Model

   Implementing the Initialize, Simulate, Validate, and Close Functions
6. When finished editing, click **C-Model Editor** > **Compile & Update Project** to add the model to the project. Use **Export** *ModelName* **Text As** to save and export the code present in the editor.

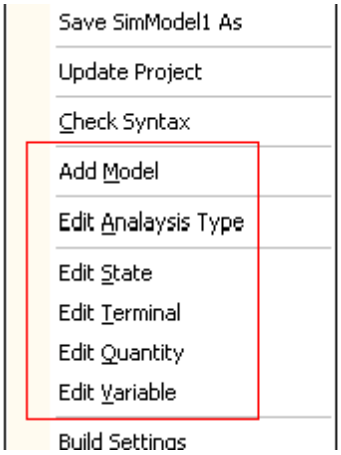
Related Topics

Importing Simulation Models

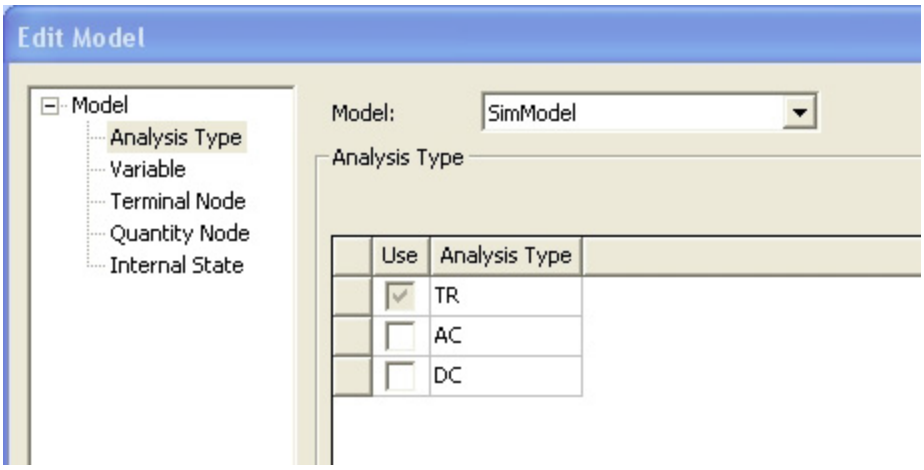
Adding Analysis Types, Nodes, States, Variables, and Models to a C-Model

You can add analysis types, terminal and quantity nodes, internal states, variables, and additional models to a C-Model project with the commands highlighted below in the **C-Model Editor** menu.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The resulting **Edit Model** dialog box provides a tree view with the selected item highlighted in the left pane. The right pane contains the corresponding data entry fields. Select items in the tree to add models, analysis types, and so on.

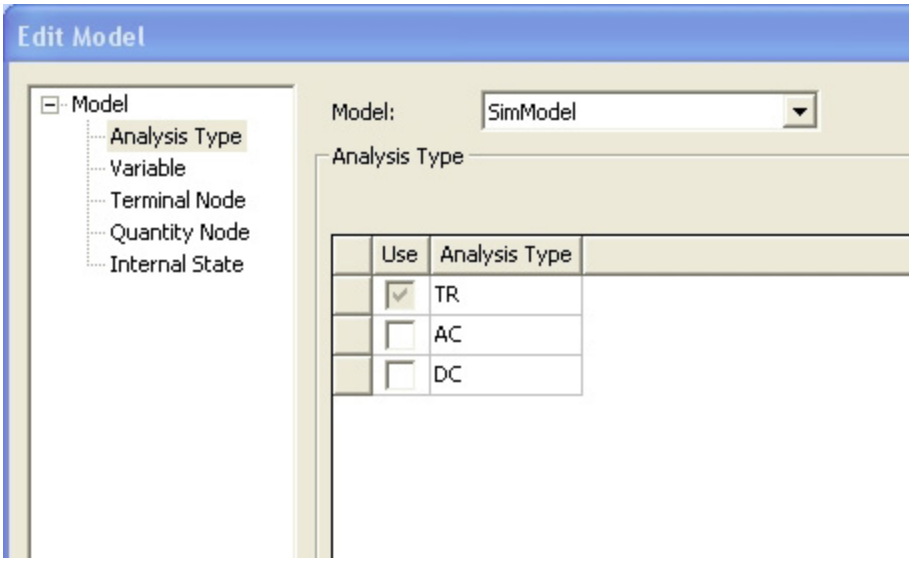


Adding Analysis Types

To add analysis types to a C-Model:

1. Click **C-Model Editor** > **Edit Analysis Type**. The **Edit Model** dialog box appears with **Analysis Type** selected in the **Model** tree.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. Select the name of the appropriate model from the **Model** list.
3. Select the **Use** check box next to the **Analysis Type** (**TR**ansient, **AC**, or **DC**) you want to add.
4. Click **OK**.

   Initialize, **Simulate**, **Validate**, and **Close** function declarations are added for each analysis type selected for the model.

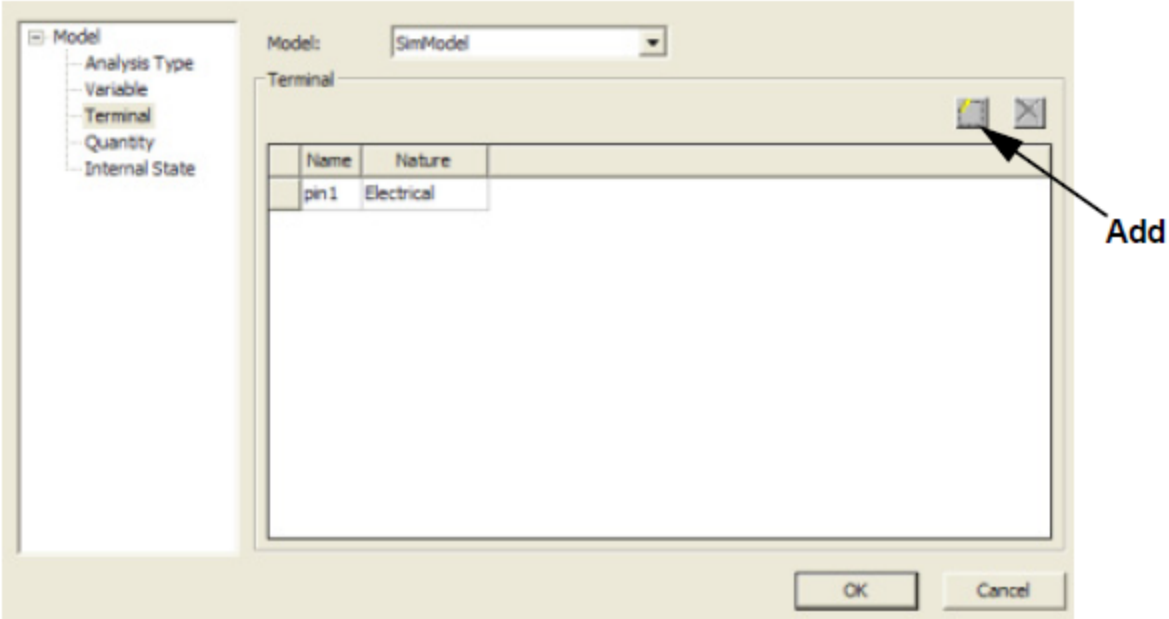
> **Note:**
>
> The *contents* of these functions must still be added. For further information about creating the contents of these functions, see **Implementing the Initialize, Simulate, Validate, and Close Functions**.

Adding Terminal Nodes

To add a terminal node to a C-Model:

1. Click **C-Model Editor** > **Edit Terminal**. The **Edit Model** dialog box appears with **Terminal** selected in the **Model** tree.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. Select the name of the appropriate model from the **Model** field.
3. Click **Add** to add a new entry to the list in the **Terminal** field. If this is the first time you have visited this field for a new model, the field will be empty.
4. Type a name for the terminal node into the **Name** field. This terminal node will be visible when the model containing the node is placed onto a Twin Builder schematic.
5. Select the nature of the terminal node from the list of available natures in the **Nature** field.
6. Repeat steps 2 through 4 to add other terminal nodes as required for your model.
7. When finished, click **OK**.

   An **AddNode__c** function call for each added terminal node will be added to the selected model's **Prepare_** function.

Editing a Terminal Node

To edit a terminal node:

1. Click **C-Model Editor** > **Edit Terminal**. The **Edit Model** dialog box appears with **Terminal** selected in the **Model** tree.
2. Select the name of the appropriate model from the **Model** field.
   - To edit existing information for the node, click the information and edit it.
   - To remove an existing node, select the line and click the red **X** to delete it.
3. Click **OK** to complete the editing process.

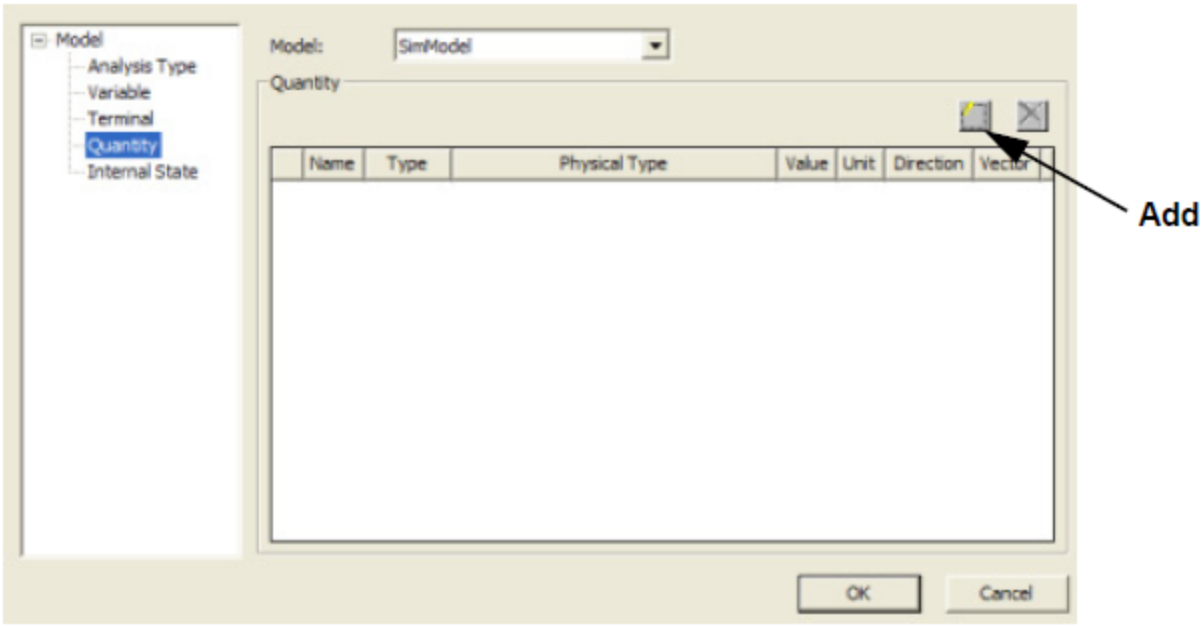
   The **AddNode__c** function calls for the edited nodes is altered to reflect the changes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding Quantity Nodes

Follow this procedure to add a quantity node to a C-Model:

1. Select **C-Model Editor** > **Edit Quantity**. The **Edit Model** dialog box appears with **Quantity** selected in the **Model** tree.



2. Select the name of the appropriate model from the **Model** field.
3. Click **Add** to add a new entry to the list in the **Quantity** field. If this is the first time you have visited this field for a new model, the field will be empty.
4. Type a name for the quantity node into the **Name** field. This quantity node appears when the model containing the node is placed onto a Twin Builder schematic.
5. Select the type of quantity node from the list of available types in the **Type** field.
6. If the **Type** chosen is **REAL_**, select the appropriate physical type from the list in the **Physical Type** field and the expected unit of measure from the list in the **Unit** field.
7. If applicable, type an initial value in the **Value** field.
8. Select the direction of the node from the **Direction** field.
   - Select **DIRIN** if the node is an input.
   - Select **DIROUT** if the node is an output.
   - Select **DIRINOUT** if the node is both an input and an output.
9. If the value of the node is a data vector, select the **Vector** check box.
10. To add more quantity nodes, repeat Steps 1 through 8 for each node. If not, click **OK** to add the new node properties.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

AddNode_nc, **SetDataTypeNode_nc**, and **SetUnitNameNode_nc** function calls are added to the appropriate model's **Prepare_** function for each quantity node created.

Editing a Quantity Node

To edit a quantity node:

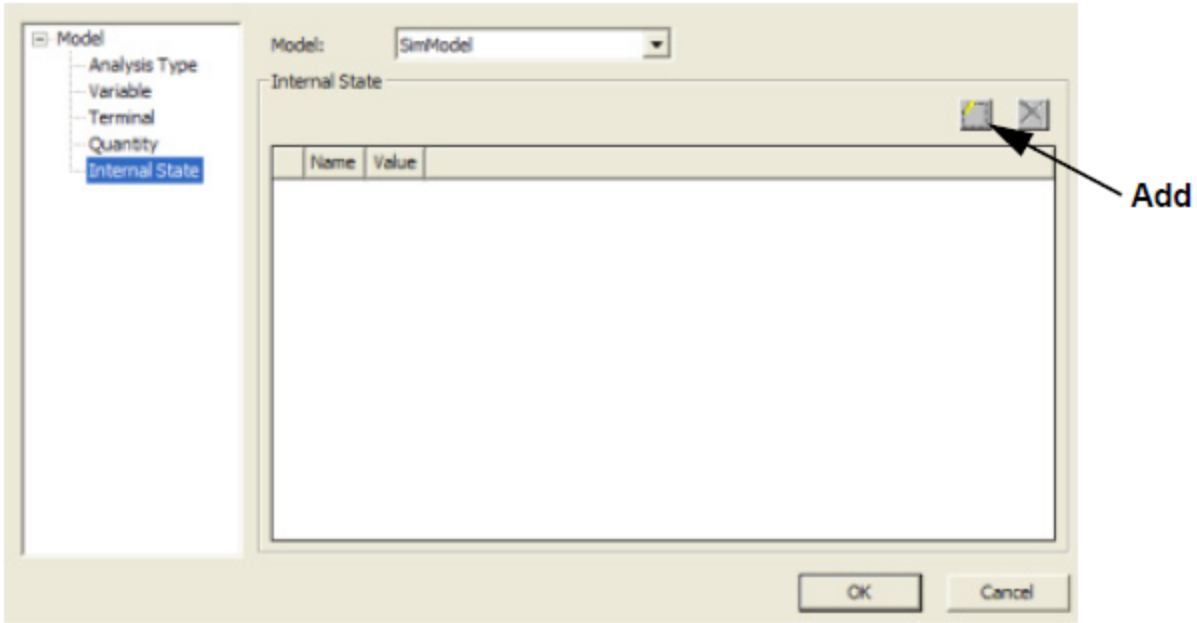
1. Click **C-Model Editor** > **Edit Quantity**. The **Edit Model** dialog box appears with **Quantity** selected in the **Model** tree.
2. Select the name of the appropriate model from the **Model** field.
   - To edit existing information for the node, click the information and edit it.
   - To remove an existing node, select the line and click the red **X** to delete it.
3. Click **OK** to complete the editing process.

   The **AddNode_nc**, **SetDataTypeNode_nc**, and **SetUnitNameNode_nc** function calls for the edited nodes are altered to reflect the changes.

Adding Internal State Definitions

To add an internal state to a C-Model:

1. Click **C-Model Editor** > **Edit State**. The **Edit Model** dialog box appears with **Internal State** selected in the **Model** tree.



2. Select the name of the appropriate C-Model from the **Model** field.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Click **Add** to add a new entry to the list in the **Internal States** field. If this is the first time you have visited this field for a new model, the field will be empty.
4. Select a name for the state and type it into the **Name** field. The name should be a valid C++ identifier.
5. Select a numerical default value for the state and type it into the **Value** field.
6. If you want to add more states, repeat Steps 1 through 4 for each state. Click **OK** to add the internal states.

   Click **OK**, and the **AddNode_State** function calls for the states are inserted in the **Prepare_** function of the appropriate model.

Editing Internal State Definitions

To edit a state definition:

1. Click **C-Model Editor** > **Edit State**. The **Edit Model** dialog box appears with **State** selected in the **Model** tree.
2. Select the name of the appropriate model from the **Model** field.
   - To edit existing information for the state, click the information and edit it.
   - To remove an existing state definition, select the line and click the red **X** to delete it.
3. Click **OK** to complete the editing process.

   The **AddNode_State** function calls for the edited states will be altered to reflect the changes.

Declaring Variables

To declare a variable:

1. Click **C-Model Editor** > **Edit Variable**. The **Edit Model** dialog box appears with **Variable** selected in the **Model** tree.
2. Select the name of the appropriate C-Model from the **Model** field.
3. Click **Add** to add a new entry to the list in the **Internal States** field. If this is the first time you have visited this field for a new model, the field will be empty.
4. Select a name for the variable and type it into the **Name** field. The name should be a valid C++ identifier.
5. Select the data **Type** from the drop-down list.
6. Enter a default value appropriate for the **Type**.
7. If you want to add more variables, repeat Steps 1 through 5 for each variable.
8. Click **OK** to add the variables.

   Declaration statements for the variables are inserted in the **Prepare_** function of the appropriate model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Editing Variable Declarations

To edit a variable declaration:

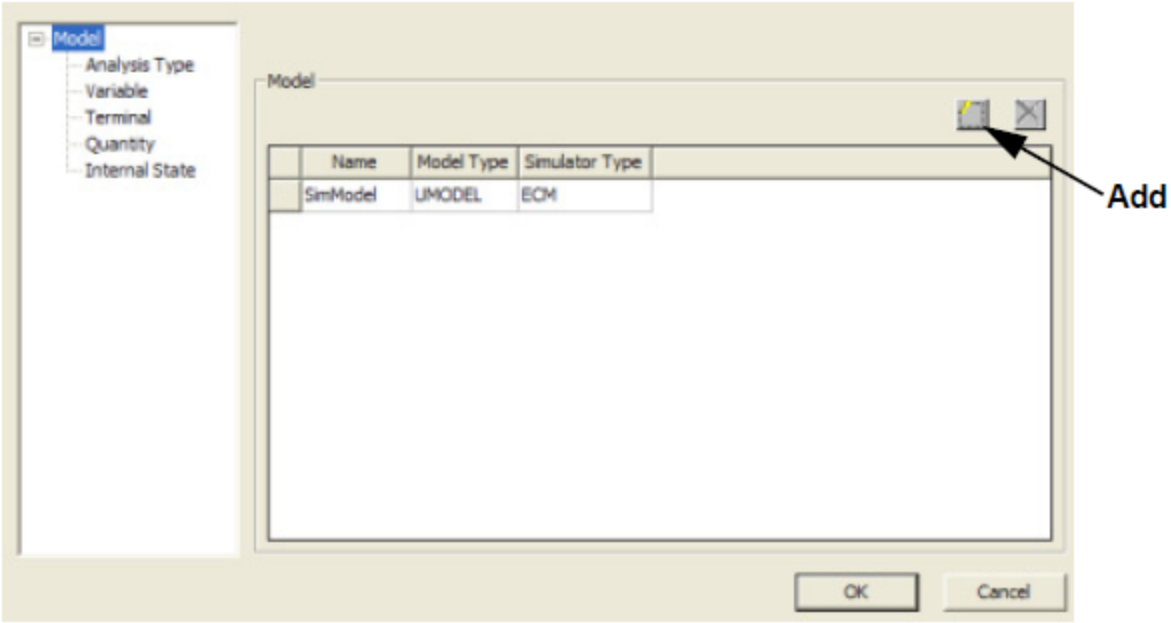
1. Click **C-Model Editor** > **Edit Variable**. The **Edit Model** dialog box appears with **Variable** selected in the **Model** tree.
2. Select the name of the appropriate C-Model from the **Model** field.
   - To edit existing information for the variable, click the information and edit it.
   - To remove an existing variable declaration, select the line and click to delete it.
3. Click **OK** to complete the editing process.

   The declaration statements for the edited variables will be altered to reflect the changes.

Adding Models

Additional models may be added to a C-Model project to create a multiple-model DLL. To add an additional model:

1. Click **C-Model Editor** > **Add Model**. The **Edit Model** dialog box appears with **Model** selected in the tree view.



2. Click **Add** to add a new entry to the list in the **Model** field.
3. Enter a name for the new model in the **Name** field.
4. Select the **Model Type** you are adding.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Select the model **Simulator Type**.
6. If you want to add more models, repeat Steps 1 through 4 for each model.
7. Click **OK** to add the models.

   Click **OK**, and the C-Model Wizard generates the **AddNode_State** function call for the internal state in the **Prepare_** function of the appropriate model.

Implementing the Initialize, Simulate, Validate, and Close Functions

This section describes how to implement the **Initialize** and **Simulate** functions of a C-Model using the **Insert Data** and **Simulator Data** menus. These menus are context sensitive; right-click the appropriate position in the source code to access them.

> **Note:**
>
> **Validate** and **Close** functions must be implemented manually. For more information on how to implement **Validate** and **Close** functions, see:
>
> *Sample C-Model Source Code*
>
> *C-Models in Twin Builder*

Setting Matrix Entries

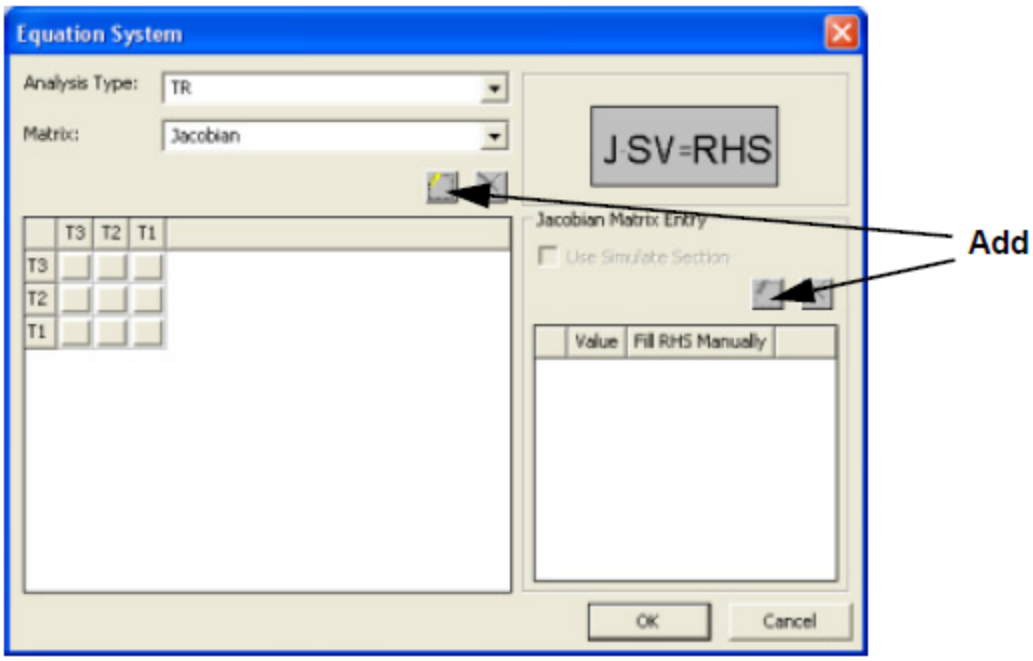
The function calls that set the matrix entries of a C-Model can be implemented using commands found in the **Simulator Data** menu.

> **Note:**
>
> For more information on how to determine the matrix entries of a C-Model, see ***C-Models in Twin Builder***.

To access the **Simulator Data** menu:

1. Right-click in the body of the **Initialize** function for one of the analysis types.
2. Select **Simulator Data** > **Set Equation System**. The **Equation System** dialog box appears.
3. In the **Analysis Type** section, select the analysis type for which the initialize and simulate functions are being implemented from the drop-down list (**TR_**, **AC_**, and **DC_** represent transient, AC, and DC analyses).
4. In the **Matrix** section, select the matrix for which entries are to be set from the list. Select **Jacobian** for the Jacobian coefficient matrix, **Solution Vector** for the solution vector, or **Right Hand Side** to enter right-side vector elements.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



By default, the dialog box shows matrix entries for each of a C-Model's terminal nodes.

5. Click **Add** to add additional variables to the matrix. New variables are called **Change Name**. To edit the name of a new variable, click in the appropriate field in the **Node** column, and edit the name.
6. Once all variables have been added to the matrix, you can add nonzero matrix entries to the matrix. To add nonzero matrix entries:
   a. Click the position of the entry in the matrix. The **Jacobian Matrix Entry** section of the **Equation System** dialog box is enabled for editing.
   b. Click **Add** in the **Jacobian Matrix Entry** section and enter the value for the matrix entry into the **Value** field. If the matrix entry contains a temporal derivative, it can be represented by **1/dt**. If the matrix entry contains a complex angular frequency term, jw can be represented by **p**.

      Select the **Fill RHS Manually** check box only if you intend to manually fill the right-side vector entry corresponding to the position of the currently selected matrix entry. Otherwise, you can leave this check box cleared.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You should manually fill the right-side vector entry only if the corresponding equation is nonlinear. For more information on setting right-side entries, see ***C-Models in Twin Builder***.

c. Enter the right-side value corresponding to the matrix entry into the **Right Hand Side Value** field only if the **Fill RHS Manually** check box is selected. If not, leave the field empty.
d. Select the **Use Simulate Section** check box if the matrix entry is not constant.
e. Once you have set all nonzero Jacobian matrix entries and all right-side vector entries, select **Jacobian** in the **Matrix** section of the dialog box to preview the completed Jacobian matrix. Select **Right Hand Side** to preview the completed right-side vector.

7. Set each **Solution Vector** entry by entering its value into the appropriate location in the **Value** column.
8. Once you have set all nonzero Jacobian matrix, right-side vector, and solution vector entries, click **OK**. The C-Model editor generates the C code to set the matrix entries in the **Initialize** and **Simulate** functions for the selected analysis.

Initializing Variables

You can initialize variables in **Initialize** and **Simulate** functions to the values of literals, expressions, functions, struct variables, quantity nodes, and solution vector values. Initializing variables to the values of literals, expressions, functions, and struct variables must be done entirely by manual editing. Initializing variables to the values of quality nodes or solution vector values, however, can be done predominantly with commands available from the **Insert Data** and **Simulation Data** menus of the C-Model editor.

To initialize a variable to the value of a quantity node:

1. In the editor window, locate the variable you want to initialize. Replace the semicolon following the variable declaration with the C++ assignment operator =.
2. Open the **Insert Data** menu. Select **GetValNode_nc**, **GetValNode_ncFile**, or **GetValNode_ncStrg** to initialize the variable to a quantity node value that is a real number, file name, or string, respectively. The **Get/Set Value** dialog box appears.
3. Select the name of the quantity node whose value the variable will be initialized with from the drop-down list in the node **Name** field.
4. Click **OK**. The C-Model editor generates the code to initialize the variable to the value of the selected quantity node.

To initialize a variable to the value of a solution vector value:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select the variable to be initialized. Replace the semicolon following the variable declaration with the C++ assignment operator =.
2. Open the **Simulator Data** menu. Select **GetSVVal** to initialize the variable to a solution vector value or **GetDSVVal** to initialize the variable to the time derivative of a solution vector value. The **Get/Set Value** dialog box appears.
3. Select the row of the solution vector element to whose value the variable will be initialized from the drop-down list in the **Name** field.
4. Click **OK**. The C-Model editor generates the code to initialize the variable to the value of the selected solution vector element.

Setting the Values of Quantity Nodes

Use the **Insert Data** menu to generate the function calls that set the values of a C-Model's quantity nodes during each simulation step.

Setting Quantity Nodes to Real or String Values

1. Right-click in the desired **Simulate** function.
2. Click **Insert Data** > **SetValNode_nc**. The **Get/Set Value** dialog box appears.
3. Select the name of the quantity node whose value the variable will be initialized with from the **Name** field.
4. If the **Value** to be assigned to the node is a variable, select the **Value** radio button and select the desired variable from the drop-down list in the adjacent field. If the value to be assigned to the node is a string, select the **Text** radio button and enter the string value into the adjacent field.
5. Click **OK** to generate the function call necessary to set the selected quantity node to the desired value.

Setting the Value of a Quantity Node to a File Name

1. Right-click in the desired **Simulate** function.
2. Click **Insert Data** > **SetValNode_ncFile**. The **Get/Set Value** dialog box appears.
3. Select the name of the quantity node for which the value is to be set from the **Name** field.
4. Enter the file name that the node is to be set to into the **File** name field, or click **Browse** to find the file.
5. Click **OK** to generate the function call necessary to set the selected quantity node to the desired file name.

Checking Syntax (C-Model Editor)

Follow this procedure to check the syntax of code you entered in the editor:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You must have Microsoft® Visual Studio 2013 or later installed to check syntax.

1. Select **C-Model Editor** > **Check Syntax** to check the syntax of the new C-Model.
2. The Microsoft® Visual Studio C++ compiler you selected in the C-Model options checks the code for syntax errors. The results of the syntax check appear in the **Message Manager** pane. If errors are reported, click them to see the associated lines of code in the editor window.

   > **Note:**
   >
   > The system generates a Dynamic Link Library (DLL) file of the model during the syntax check.

Building a C-Model Component

When all analysis types, terminal and quantity nodes, internal states, variables, and models are added to a C-Model project and all necessary Initialize, Simulate, Validate, and Close functions are implemented, the C-Model is ready to be built and added to the Twin Builder project as a component.

Related Topics

File Settings

Configuring Build Settings

Compiling and Updating a Project to Include the C-Model

C-Model Editor File Settings

Use the **File Settings** dialog box to change settings related to source files and the storage location of the C-Model DLL for the specific C-Model definition. The **File Settings** dialog box may also be used to delete project files that are already included with the project from the C-Model. To open the **File Settings** dialog box, click **C-Model Editor** > **File Settings**. The **File Settings** dialog box contains the following tabs:

- **Source File Settings**
- **Model Update Settings**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Source File Settings Tab

- **Model Source** – Set how the sources of this C-Model are managed. Choose **None** if you do not want the model to know the path to the source .**scp** file. Choose **Path to Project** to enable the model to know the path to the source .**scp** file. Choose **Zip Model Source** to have the model contain the C-Model editor file contents in ZIP format.

  > **Note:**
  >
  > If the selection is set to not store the sources with the model as a ZIP file or a link it will not be possible to edit the C-Model later on.

- **Add/Remove Files** – If any additional files are required to build the project, they may be added to the source file list. To add source files to the project, click **New File**. Enter a file name and click **OK**. The file is added to the project and displayed in the editor. To add an existing file to the source file list, click **Add File** and browse to the file. The following types of source files can be displayed in the C-Model editor and used as source files for the C-Model project: *.**cpp**, *.**c**, *.**cxx**, *.**h**, *.**hpp**, *.**hxx**. Other types of files can also be included in the list of source files; however, they will not be displayed in the C-Model Editor.

Model Update Settings Tab

The settings on this tab define the storage location of the C-Model DLL file.

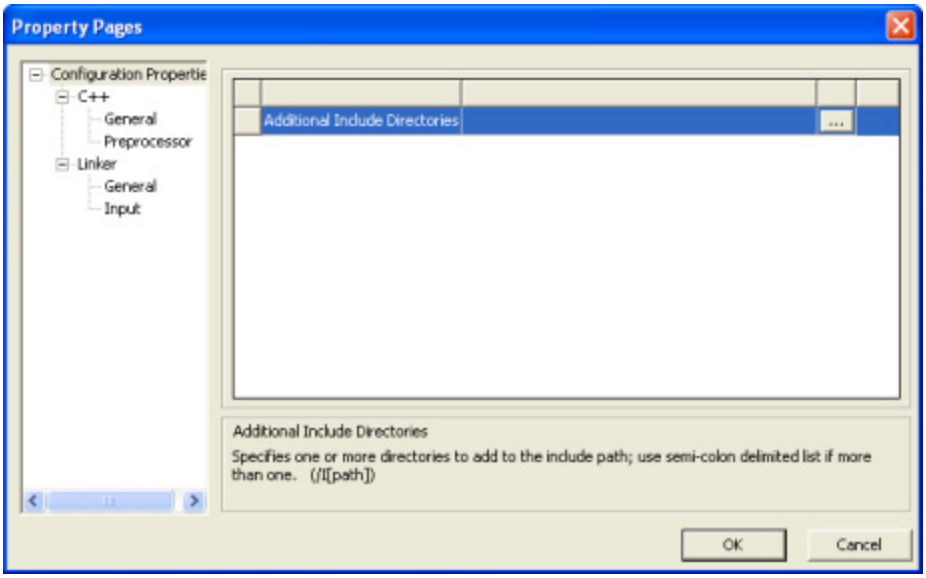
- **Keep DLL on Disc** – Defines whether the binaries of the C-Model DLL file are kept on disc or stored in the project together with the model. If the DLL is kept on disc, the location can be selected between the **PersonalLib** and the **UserLib** areas. In this case the full path of the location is displayed. If the setting is turned off, the DLL binaries are stored in the project file together with the model. For the simulation, the DLL is extracted to a temporary location.
- **Copy DLL to Bin Directory of** – Selection of the library area where the DLL file is stored. Available locations are the **PersonalLib** and **UserLib** areas.
- **Remove from Original Location** – If the location setting changed from the point when the C-Model editor was opened and the old location was on disc, this setting will remove the DLL from the old location when the model is updated.

Configuring Build Settings

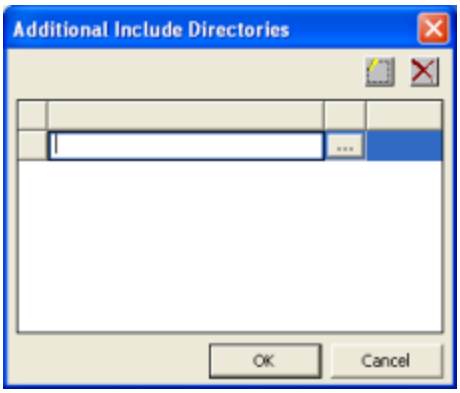
Follow this procedure to configure the build settings of a C-Model project prior to building:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **C-Model Editor** > **Build Settings**. The following dialog box appears:



2. Select **C++** > **General** in the tree view to set the paths to C++ files to include in the build. You can either type the path directly into the **Additional Include Directories** text box, or click ... to open the **Additional Include Directories** dialog box in which you can browse to and add directories.



3. Click **C++** > **Preprocessor** in the tree view to add **Preprocessor Definitions** to the build configuration.
4. You can also set library paths for builds by selecting **Linker** > **General** in the tree, then typing the path in the **Additional Library Directories** text box. You can also use ... to open the **Additional Include Directories** dialog box to browse and add directories.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Specify additional dependencies by selecting **Linker** > **Input** and entering the items to add to the link line for the build.
6. When finished modifying the build settings, click **OK**.

Compiling and Updating a Project to Include the C-Model

You can update a project to both build a Dynamic Link Library (DLL) file of your C-Model, and create a component as follows:

1. Click **C-Model Editor** > **Compile & Update Project** in the menu bar.
2. The **Import Components** dialog box appears, listing all of the models in the current C-Model project on the **Models** tab.
3. See *Importing Simulation Models* for detailed information on settings you can make in this dialog box.
4. Click **OK**. This invokes the C++ compiler of Microsoft Visual Studio to compile the C-Model source code and header files into a DLL file. If the compiler detects an error or issues a warning, it appears in the **Message Manager** window and will identify the line number of the statement that caused the error or warning. Press **Ctrl+G** to open the **Go To Line** dialog box to go to the line.

   If the compilation of the C-Model project succeeds, a message appears in the **Message Manager** window indicating that the project update succeeded.

   A component for each model is created and added to the **Project Manager Component Definitions** folder. The new models are also added to the list of **Project Components** where they can be placed onto a schematic for simulation.

Debugging a C-Model Created with the C-Model Editor

Since the **C-Model Editor** does not provide a debugger, you must use the Visual Studio C++ debugger. The C-Model project files reside in the **PersonalLib** folder. See **General Options: Directories** for information on designating a **PersonalLib** directory.

Related Topics

Debugging C-Models in Microsoft Visual Studio

Revising an Existing C-Model with the C-Model Editor

The C-Model Editor allows for easy revision of existing C-Models created using the editor. To edit the model code of a C-Model, double-click the model in the **Project Manager** > **Definitions** > **Models** folder. You can also click **Edit Model** in the context menu of the C-Model instance in the schematic.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

C-Models created in Microsoft Visual Studio cannot be revised using the C-Model Editor; they must be revised using the application in which they were created. C-Models that have no sources stored with them (as a ZIP file or a linked project directory; see ***C-Model Editor File Settings***) cannot be revised using the C-Model Editor.

Starting the Revision Process

You can use two methods to revise a C-Model. The first is to go to the **Definitions** > **Models** folder in the **Project Manager**. Right-click the desired C-Model and select **Edit Model**. You can also double-click the model listing.

C-Model projects created using the C-Model Editor may also be revised by opening the **Edit Libraries** dialog box via the **Tools** > **Edit Libraries** > **Models** menu selection. Locate and select the desired model name in the listing, and click **Edit Model** to open the project's files for editing.

Revising a C-Model

Once the files are opened, the C-Model may be revised. You can perform the following revisions on a C-Model.

- Addition of a new model.
- Addition of terminal and quantity nodes.
- Addition of internal states.
- Revision of initiate, simulate, validate, and close functions.
- Revision of terminal and quantity node properties.

For more information on how to perform these revisions on a C-Model, see the following sections:

Adding Analysis Types, Nodes, States, Variables, and Models to a C-Model

Implementing the Initialize, Simulate, Validate, and Close Functions

Implementing the Changes

To implement the revisions made to a C-Model, follow the process outlined in the section on Compiling and Updating a Project to Include the C-Model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

C-Model Editor Menus and Windows

This section gives a brief description of the available menus and windows when using the C-Model Editor.

C-Model Editor Main Menu

The **C-Model Editor** main menu contains these commands:

> **Note:**
>
> The **Compile & Update Project** and **Export** *ModelName* **Text As** menu items are common to the VHDL Package Editor, and the C-Model, VHDL-AMS, SPICE, and SML model editors.

- **Compile & Update Project**
  - For C-Models, invokes Microsoft® Visual Studio to build a Dynamic Link Library (DLL) file of your C-Model.
  - For all model types, enables creation of components for the models and their addition to the current project. For packages, enables creation of the package and addition to the current project.
- **Check Syntax** – Invokes Microsoft® Visual Studio to check the code for syntax errors. Twin Builder‘s **Message Manager** pane displays the results of the syntax check.
- **Export Model Text As** – Save and export the code present in the editor.
- **Add Model** – Add a new model to a C-Model project.
- **Edit Analysis Type** – opens a dialog box for adding a TR (transient), AC, or DC operating-point analysis to a model within a C-Model project.
- **Edit State** – Add and edit C-Model internal states within a C-Model project.
- **Edit Terminal** – Add and edit C-Model terminal nodes within a C-Model project.
- **Edit Quantity** – Add and edit C-Model quantity nodes within a C-Model project.
- **Edit Variable** – Add and edit C-Model variable declarations within a C-Model project.
- **Build Settings** – Set paths to C++ files to include in the building of a C-Model project, specifying preprocessor definitions, additional dependencies, and for setting library paths different from the default path.
- **File Settings** – Add and remove files from a C-Model project.
- **Export Project** – Choose which Visual Studio compiler to use to export the current C-Model project. The sub-menu lists all supported versions. A Visual Studio icon appears next to versions currently installed.

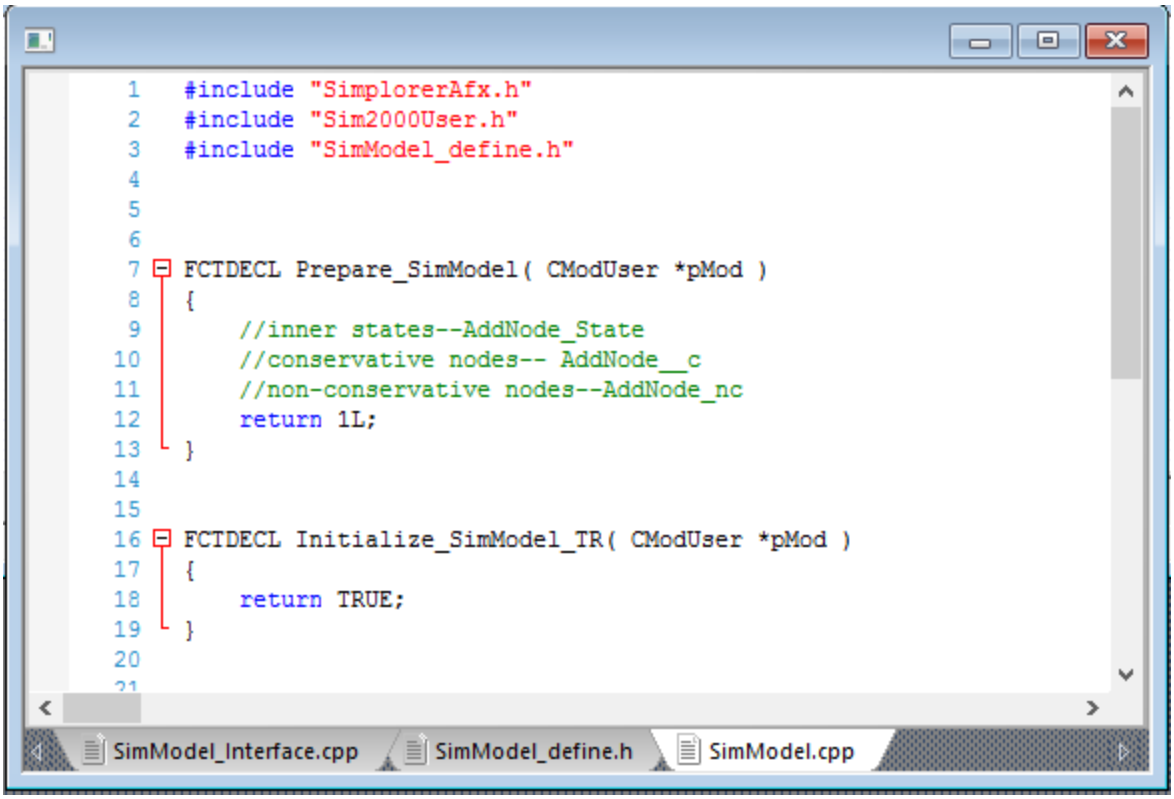
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

C-Editor Standard Functions

See Netlist Editor Tab for detailed information on standard editor functions.

Main C-Editor Window

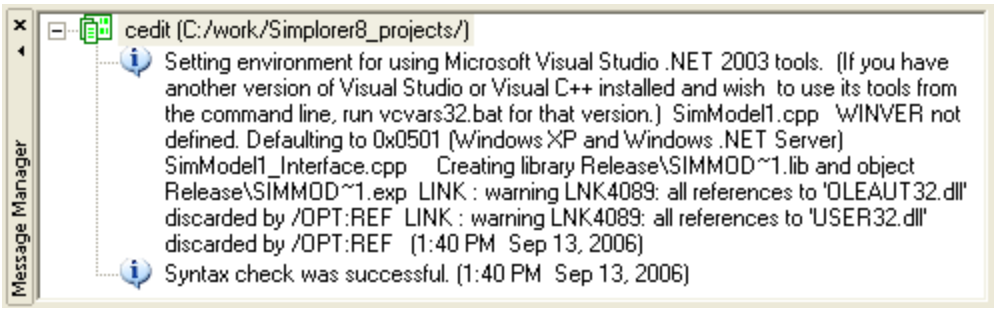
The main window (shown below) displays all C++ source and header files for the current C-Model project. Use the tabs at the bottom of the window to select a file to edit.



C-Editor Message Window

The **Message Manager** window displays messages such as compiler warnings, syntax errors, and build status.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Insert Data Menu

Right-click within the body of a C-Model function and select **Insert Data** to access the **Insert Data** menu. The menu contains these commands:

- **GetValNode_nc** – Obtain a real value from a quantity node.
- **GetValNode_ncCplx** – Obtain a complex value from a quantity node.
- **GetValNode_ncFile** – Obtain a file name from a quantity node.
- **GetValNode_ncStrg** – Obtain a string value from a quantity node.
- **GetValNode_ncState** – Obtain a state value from a quantity node.
- **SetValNode_nc** – Pass a real or string value to a quantity node.
- **SetValNode_ncFile** – Pass a file name to a quantity node.
- **SetValNode_ncState** – Pass a file name to a quantity node.
- **AllocateUserDataMemory** – Allocate memory for instance-specific data.
- **GetUserData** – Obtain user-specific data.
- **GetDCUserData** – Obtain DC user-specific data.

Simulator Data Menu

To access the **Simulator Data** menu, right click within the body of a C-Model function and select **Simulator Data**. The menu contains the following:

- **Set Equation System** – Open the **Equation System** dialog box, which is used to generate code for setting Jacobian matrix, right-side vector, and solution vector entries.
- **GetCplxSVVal** – Obtain the complex value of a solution vector element.
- **GetDSVVal** – Obtain the time derivative of a solution vector element.
- **GetSVVal** – Obtain the value of a solution vector element.

Callback Functions Menu

Right-click a C-Model function and select **Callback Functions** to access the **Callback Functions** menu. The menu contains the following commands:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **ISIM_ECM_D_DT** – Return the time derivative operator for transient analysis.
- **ISIM_ECM_IEMAX** – Return the maximum current error allowed during iterations in transient simulation.
- **ISIM_ECM_ITERAT** – Return the maximum number of iterations.
- **ISIM_ECM_LDF** – Return the local discretization error.
- **ISIM_ECM_NEW** – Return TRUE when the simulator operates smoothly and FALSE when the simulator is forced to take a step back in time.
- **ISIM_ECM_P** – Return complex frequency operator in AC analysis.
- **ISIM_ECM_SOLVER** – Return the solver type of the network simulator.
- **ISIM_ECM_UEMAX** – Return the maximum voltage error allowed during iterations in a transient simulation.
- **OSIM_ECM_REJECT** – Request a back step from the simulator.
- **OSIM_ECM_SYNC** – Request a back step from the simulator and setting of a new, user-defined step size.
- **OSIM_SYNC** – Request back step and setting new user-defined step size.
- **Report2Sim** – Display messages in the simulator output window or in pop-up windows.

Sample C-Model Source Code

Two implementations are provided:

- The initialize, simulate, validate, and close functions for the transient, AC, and DC analyses of an RLC low-pass filter.
- The initialize, simulate, validate, and close functions for a linear characteristic model.

RLC Low-Pass Filter Sample Code

This topic contains the implementation of the initialize, simulate, validate, and close functions for the transient, AC, and DC analyses of an RLC low-pass filter.

```
//================================================
FCTDECL Initialize_rlc_lowpass_TR( CModUser *pMod )
{
   //nonzero matrix entries row 0
   pMod->SetSymbolicGSEntry(0, 5, 0, RS_DONTFILL|D_DT_OPERAT);
    //row 1
   pMod->SetSymbolicGSEntry(1, 4, 0, RS_DONTFILL);
   pMod->SetSymbolicGSEntry(1, 5, 0, RS_DONTFILL|D_DT_OPERAT);
    //row 2
   pMod->SetSymbolicGSEntry(2, 2, 0, RS_DONTFILL);
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    pMod->SetSymbolicGSEntry(2, 3, 0, RS_DONTFILL);
     //row 3
    pMod->SetSymbolicGSEntry(3, 2, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(3, 3, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(3, 4, 0, RS_DONTFILL);
     //row 4
     pMod->SetSymbolicGSEntry(4, 1, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(4, 3, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(4, 4, 0, RS_DONTFILL|D_DT_OPERAT);
     //row 5
    pMod->SetSymbolicGSEntry(5, 0, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(5, 1, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(5, 5, 0, RS_DONTFILL);
     //static entries
    pMod->SetRealGSEntry(4, 1, 1);
    pMod->SetRealGSEntry(4, 3, -1);
    pMod->SetRealGSEntry(5, 0, -1);
    pMod->SetRealGSEntry(5, 1, 1);
     return TRUE;
}

FCTDECL Simulate_rlc_lowpass_TR( CModUser *pMod )
{
     //get information from nonconservative nodes
    double C = pMod->GetValNode_nc(STRG_CAP_VALUE);
    double L = pMod->GetValNode_nc(STRG_INDUCTOR_VAL);
    double R = pMod->GetValNode_nc(STRG_RESISTOR_VAL);
     //get solution vector values for rightside entries
    double V1 = pMod->GetSVVal(0);
    double V2 = pMod->GetSVVal(1);
    double V3 = pMod->GetSVVal(2);
    double V4 = pMod->GetSVVal(3);
    double PHI = pMod->GetSVVal(4);
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    double Q = pMod->GetSVVal(5);
    double dPHI = pMod->GetDSVVal(4);
    double dQ = pMod->GetDSVVal(5);

     //non-static matrix entries--row 0
    pMod->SetRealGSEntry(0, 5, -ISIM_ECM_D_DT( pMod ));

     //row 1
    pMod->SetRealGSEntry(1, 4, -1.0/L);
    pMod->SetRealGSEntry(1, 5, ISIM_ECM_D_DT( pMod ));

     //row 2
    pMod->SetRealGSEntry(2, 2, 1.0/R);
    pMod->SetRealGSEntry(2, 3, -1.0/R);

     //row 3
    pMod->SetRealGSEntry(3, 2, 1.0/R);
    pMod->SetRealGSEntry(3, 3, -1.0/R);
    pMod->SetRealGSEntry(3, 4, -1.0/L);

     //row 4
    pMod->SetRealGSEntry(4, 4, ISIM_ECM_D_DT( pMod ));
    //pMod->SetRealGSEntry(4, 5, 1.0/C);

     //row 5
    pMod->SetRealGSEntry(5, 5, -1.0/C);

     //rightside entries
    pMod->SetRealRSEntry(0, dQ);
    pMod->SetRealRSEntry(1, -dQ + PHI/L);
    pMod->SetRealRSEntry(2, -V3/R + V4/R);
    pMod->SetRealRSEntry(3, -V3/R + V4/R + PHI/L);
    pMod->SetRealRSEntry(4, -V2 + V4 - dPHI);
    pMod->SetRealRSEntry(5, V1 - V2 + Q/C);

     //send values to nonconservative output nodes
    pMod->SetValNode_nc(STRG_RESISTOR_VOLTAGE, V3-V4);
    pMod->SetValNode_nc(STRG_INDUCTOR_VOLTAGE, dPHI);

     return TRUE;
}
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
FCTDECL Validate_rlc_lowpass_TR( CModUser *pMod )
{
     /*no additional implementation of this function
    is required for this example */
    return TRUE;
}

FCTDECL Close_rlc_lowpass_TR( CModUser *pMod )
{
    /*no additional implementation of this function
    is required for this example */
    return TRUE;
}

//struct is for use of operating point values in AC simulation
struct sDCOP

{

double r, l, c, Iflux, Ccharge;

};

FCTDECL Initialize_rlc_lowpass_AC( CModUser *pMod )
{
    // Get values from DC analysis, sDCOP is defined above
    long DCsize;
    sDCOP *pUserData = (sDCOP*)pMod->GetDCUserData( &DCsize );
    double flx = pUserData->Iflux;
    double chg = pUserData->Ccharge;
     //nonzero matrix entries row 0
    pMod->SetSymbolicGSEntry(0, 5);
     //row 1
    pMod->SetSymbolicGSEntry(1, 4);
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    pMod->SetSymbolicGSEntry(1, 5);
     //row 2
    pMod->SetSymbolicGSEntry(2, 2);
    pMod->SetSymbolicGSEntry(2, 3);
     //row 3
    pMod->SetSymbolicGSEntry(3, 2);
    pMod->SetSymbolicGSEntry(3, 3);
    pMod->SetSymbolicGSEntry(3, 4);
     //row 4
     pMod->SetSymbolicGSEntry(4, 1);
    pMod->SetSymbolicGSEntry(4, 3);
    pMod->SetSymbolicGSEntry(4, 4);
     //row 5
    pMod->SetSymbolicGSEntry(5, 0);
    pMod->SetSymbolicGSEntry(5, 1);
    pMod->SetSymbolicGSEntry(5, 5);

     //static entries
    pMod->SetRealGSEntry(4, 1, 1);
    pMod->SetRealGSEntry(4, 3, -1);
    pMod->SetRealGSEntry(5, 0, -1);
    pMod->SetRealGSEntry(5, 1, 1);
     //set initial values for inductor flux and capacitor charge from
   DC operating point
    pMod->SetSVVal(4, flx);
    pMod->SetSVVal(5, chg);

    return TRUE;
}

FCTDECL Simulate_rlc_lowpass_AC( CModUser *pMod )
{

     //get values of R, L, and C from DC simulation
    long DCsize;
    sDCOP *pUserData = (sDCOP*)pMod->GetDCUserData( &DCsize );
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    double R = pUserData->r;
    double L = pUserData->l;
    double C = pUserData->c;

     //get solution vector values for rightside entries
    CComplex V1 = pMod->GetSVVal(0);
    CComplex V2 = pMod->GetSVVal(1);
    CComplex V3 = pMod->GetSVVal(2);
    CComplex V4 = pMod->GetSVVal(3);
    CComplex PHI = pMod->GetSVVal(4);
    CComplex Q = pMod->GetSVVal(5);

     //non-static matrix entries--row 0
    pMod->SetCplxGSEntry(0, 5, -ISIM_ECM_P( pMod ));

     //row 1
    pMod->SetRealGSEntry(1, 4, -1.0/L);
    pMod->SetCplxGSEntry(1, 5, ISIM_ECM_P( pMod ));

     //row 2
    pMod->SetRealGSEntry(2, 2, 1.0/R);
    pMod->SetRealGSEntry(2, 3, -1.0/R);

     //row 3
    pMod->SetRealGSEntry(3, 2, 1.0/R);
    pMod->SetRealGSEntry(3, 3, -1.0/R);
    pMod->SetRealGSEntry(3, 4, -1.0/L);

     //row 4
    pMod->SetCplxGSEntry(4, 4, ISIM_ECM_P( pMod ));

     //row 5
    pMod->SetRealGSEntry(5, 5, -1.0/C);

    //send values to nonconservative output nodes
    pMod->SetValNode_nc(STRG_RESISTOR_VOLTAGE, V3-V4);
    pMod->SetValNode_nc(STRG_INDUCTOR_VOLTAGE, V4 - V2);

     return TRUE;
}

FCTDECL Validate_rlc_lowpass_AC( CModUser *pMod )
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
{
    /*no additional implementation of this function
    is required for this example */
    return TRUE;
}

FCTDECL Close_rlc_lowpass_AC( CModUser *pMod )
{
    /*no additional implementation of this function
    is required for this example */
    return TRUE;
}

FCTDECL Initialize_rlc_lowpass_DC( CModUser *pMod )
{
    //nonzero matrix entries row 0
     //row 1
    pMod->SetSymbolicGSEntry(1, 4, 0, RS_DONTFILL);
     //row 2
    pMod->SetSymbolicGSEntry(2, 2, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(2, 3, 0, RS_DONTFILL);
     //row 3
    pMod->SetSymbolicGSEntry(3, 2, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(3, 3, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(3, 4, 0, RS_DONTFILL);
     //row 4
     pMod->SetSymbolicGSEntry(4, 1, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(4, 3, 0, RS_DONTFILL);
     //row 5
    pMod->SetSymbolicGSEntry(5, 0, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(5, 1, 0, RS_DONTFILL);
    pMod->SetSymbolicGSEntry(5, 5, 0, RS_DONTFILL);
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    //static entries
   pMod->SetRealGSEntry(4, 1, 1);
   pMod->SetRealGSEntry(4, 3, -1);
   pMod->SetRealGSEntry(5, 0, -1);
   pMod->SetRealGSEntry(5, 1, 1);

    return TRUE;
}

FCTDECL Simulate_rlc_lowpass_DC( CModUser *pMod )
{

    //get information from nonconservative nodes
   double C = pMod->GetValNode_nc(STRG_CAP_VALUE);
   double L = pMod->GetValNode_nc(STRG_INDUCTOR_VAL);
   double R = pMod->GetValNode_nc(STRG_RESISTOR_VAL);

    //get solution vector values for rightside entries
   double V1 = pMod->GetSVVal(0);
   double V2 = pMod->GetSVVal(1);
   double V3 = pMod->GetSVVal(2);
   double V4 = pMod->GetSVVal(3);
   double PHI = pMod->GetSVVal(4);
   double Q = pMod->GetSVVal(5);
   double dPHI = 0;
   double dQ = 0;

    //non-static matrix entries

    //row 1
   pMod->SetRealGSEntry(1, 4, -1.0/L);

    //row 2
   pMod->SetRealGSEntry(2, 2, 1.0/R);
   pMod->SetRealGSEntry(2, 3, -1.0/R);

    //row 3
   pMod->SetRealGSEntry(3, 2, 1.0/R);
   pMod->SetRealGSEntry(3, 3, -1.0/R);
   pMod->SetRealGSEntry(3, 4, -1.0/L);
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    //row 5
   pMod->SetRealGSEntry(5, 5, -1.0/C);

    //rightside entries
   pMod->SetRealRSEntry(0, dQ);
   pMod->SetRealRSEntry(1, -dQ + PHI/L);
   pMod->SetRealRSEntry(2, -V3/R + V4/R);
   pMod->SetRealRSEntry(3, -V3/R + V4/R + PHI/L);
   pMod->SetRealRSEntry(4, -V2 + V4 - dPHI);
   pMod->SetRealRSEntry(5, V1 - V2 + Q/C);

    //send values to nonconservative output nodes
   pMod->SetValNode_nc(STRG_RESISTOR_VOLTAGE, V3-V4);
   pMod->SetValNode_nc(STRG_INDUCTOR_VOLTAGE, dPHI);
   return TRUE;
}

FCTDECL Validate_rlc_lowpass_DC( CModUser *pMod )
{
   /*no additional implementation of this function
   is required for this example */
    return TRUE;
}


FCTDECL Close_rlc_lowpass_DC( CModUser *pMod )
{
   double res = pMod->GetValNode_nc(STRG_RESISTOR_VAL);
   double ind = pMod->GetValNode_nc(STRG_INDUCTOR_VAL);
   double cap = pMod->GetValNode_nc(STRG_CAP_VALUE);
   double flx = pMod->GetSVVal(4);
   double chg = pMod->GetSVVal(5);
    sDCOP *pUserData = (sDCOP*)pMod->AllocateUserDataMemory( sizeof
  (sDCOP) );
   pMod->SetUserData( pUserData, sizeof(sDCOP) );
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    pUserData->r = res;
   pUserData->l = ind;
   pUserData->c = cap;
   pUserData->Iflux = flx;
   pUserData->Ccharge = chg;

    return TRUE;

}
//================================================
```

Linear Characteristic Model Sample Code

This topic contains the implementation of the initialize, simulate, validate, and close functions for the linear characteristic model.

```
//================================================
FCTDECL Initialize_linear_char_TR( CModUser *pMod )

{
   //no further implementation required for this example
   return TRUE;

}


FCTDECL Simulate_linear_char_TR( CModUser *pMod )

{
   //get slope and intercept values from inputs
   double slope = pMod->GetValNode_nc(STRG_SLOPE);
   double intercept = pMod->GetValNode_nc(STRG_INTERCEPT);

    //get independent variable value
   double x = CHAR_IN(pMod);

    //variables for dependent variable and its time derivative
   double y, dy_dx;

    //calculate value of dependent variable and its time derivative
   y = slope * x + intercept;
   dy_dx = slope;

    //return dependent variable and its time derivative
   CHAR_OUT(pMod, y);
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    CHAR_OUT_DERIVE(pMod, dy_dx);
     return TRUE;
}

FCTDECL Validate_linear_char_TR( CModUser *pMod )
{
    //no further implementation required for this example
    return TRUE;
}

FCTDECL Close_linear_char_TR( CModUser *pMod )
{
    //no further implementation required for this example
    return TRUE;
}

//struct holds values of A and B used in DC for AC simulation
struct sDCOP

{

double m, b;

};



FCTDECL Initialize_linear_char_AC( CModUser *pMod )
{
    //no further implementation required for this example
    return TRUE;
}
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
FCTDECL Simulate_linear_char_AC( CModUser *pMod )
{
    //get slope and intercept from DC
    long DCsize;
    sDCOP *pUserData = (sDCOP*)pMod->GetDCUserData( &DCsize );
    double slope = pUserData->m;
    double intercept = pUserData->b;
     //get independent variable value
    double x = CHAR_IN(pMod);
     //variables for dependent variable and its time derivative
    double y, dy_dx;
     //calculate value of dependent variable and its time derivative
    y = slope * x + intercept;
    dy_dx = slope;
     //return dependent variable and its time derivative
    CHAR_OUT(pMod, y);
    CHAR_OUT_DERIVE(pMod, dy_dx);
     return TRUE;
}

FCTDECL Validate_linear_char_AC( CModUser *pMod )
{
    //no further implementation required for this example
    return TRUE;
}

FCTDECL Close_linear_char_AC( CModUser *pMod )
{
    //no further implementation required for this example
    return TRUE;
}
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
FCTDECL Initialize_linear_char_DC( CModUser *pMod )
{
    //no further implementation required for this example
    return TRUE;
}

FCTDECL Simulate_linear_char_DC( CModUser *pMod )
{
    //get slope and intercept values from inputs
    double slope = pMod->GetValNode_nc(STRG_SLOPE);
    double intercept = pMod->GetValNode_nc(STRG_INTERCEPT);
     //get independent variable value
    double x = CHAR_IN(pMod);
     //variables for dependent variable and its time derivative
    double y, dy_dx;
     //calculate value of dependent variable and its time derivative
    y = slope * x + intercept;
    dy_dx = slope;
     //return dependent variable and its time derivative
    CHAR_OUT(pMod, y);
    CHAR_OUT_DERIVE(pMod, dy_dx);
     return TRUE;
}

FCTDECL Validate_linear_char_DC( CModUser *pMod )
{
    //no further implementation required for this example
    return TRUE;
}
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
FCTDECL Close_linear_char_DC( CModUser *pMod )
{
    double sl = pMod->GetValNode_nc(STRG_SLOPE);
    double inter = pMod->GetValNode_nc(STRG_INTERCEPT);
     sDCOP *pUserData = (sDCOP*)pMod->AllocateUserDataMemory( sizeof
   (sDCOP) );
    pMod->SetUserData( pUserData, sizeof(sDCOP) );
    pUserData->m = sl;
    pUserData->b = inter;
     return TRUE;
}
//====================================
```

Using the VHDL-AMS Model Editor

The **VHDL-AMS Model Editor** automates the writing, editing, and compiling of VHDL-AMS entity declarations and architecture descriptions, generating much of the code required to implement VHDL-AMS models in Twin Builder. The **VHDL-AMS Model Editor** provides a text editor for editing and compiling the VHDL-AMS model code. The model editor manages syntax checking and component generation. You can also print the file listing.

Models generated using the **VHDL-AMS Model Editor** appear under **Project Components** in the **Project Manager** components tree, and are used in the same way as other Twin Builder components. Edit their properties with the **Component Editor**.

Related Topics

- Text Editor Options
- VHDL-AMS Models in Twin Builder
- Adding a VHDL-AMS Model
- Editing a VHDL-AMS Model
- Saving Uncompiled Model Changes
- Checking Syntax
- Printing
- Instantiating a Predefined Component in an Architecture Description
- Updating a Project to Add a VHDL-AMS Model and Component
- Encrypting a VHDL-AMS Model

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Using IEEE Standard Encryption for VHDL-AMS Models in Twin Builder
- Revising an Existing VHDL-AMS Model with the VHDL-AMS Model Editor
- VHDL-AMS Model Editor Menus and Windows

Adding a VHDL-AMS Model

Follow this procedure to add a VHDL-AMS model to a project using Twin Builder‘s VHDL-AMS Model Editor.

1. Select **Tools** > **Edit Libraries** > **Models**. The **Edit Libraries** dialog box appears.
2. Click **Add Model** to open the **Add Model** dialog box.
3. Select the **VHDL-AMS** radio button.
4. Type the new VHDL-AMS model name in the **Name** field.

   The **VHDL-AMS Model Editor** opens, displaying new VHDL-AMS model entity and architecture declarations in separate tabs bearing the model name you chose above.
5. Edit the VHDL-AMS model source code as needed.
6. When finished editing, select **VHDL Model Editor** > **Compile & Update Project** to add the model to the project. See ***Updating a Project to Add a VHDL-AMS Model and Component*** for more information..

You can use **Export** *ModelName* **Text As** to save and export the code present in the editor.

Editing a VHDL-AMS Model

Use the VHDL Model Editor to edit entity and architecture descriptions, and any libraries, packages, and models to be used with the VHDL-AMS models.

The resulting **Edit Model** dialog box provides a tree view with the selected item highlighted in the left pane. The right pane contains the corresponding data entry fields.

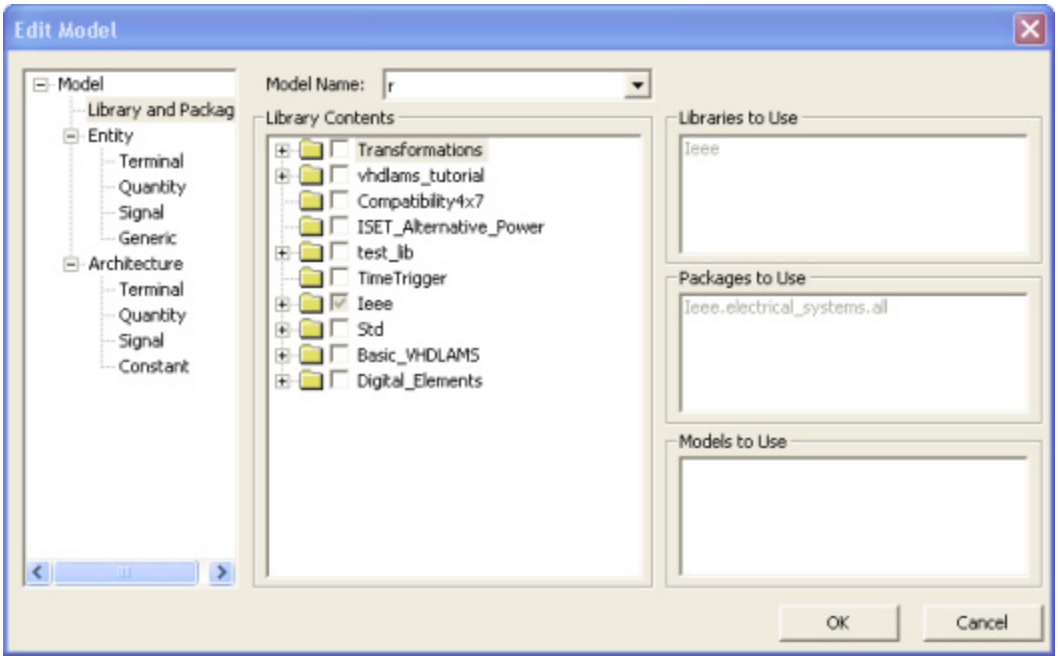
Related Topics

- Editing Libraries, Packages, and Models
- Adding Terminal, Quantity, or Signal Ports to an Entity Description
- Adding Generics to an Entity Description
- Editing an Entity
- Renaming an Entity
- Adding an Architecture
- Editing an Architecture Declaration
- Renaming an Architecture Declaration

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Editing Libraries, Packages, and Models

To include libraries, packages, and models in a VHDL model, select **Edit Libraries and Packages** in the **VHDL Model Editor** menu. The **Edit Model** dialog box appears:



1. Select the name of the appropriate model from the list in the **Model Name** field.
2. In the **Library Contents** section, select the libraries, packages, and models to include in the VHDL model.
3. Click **OK**.

   VHDL-AMS code for the necessary **LIBRARY** and **USE** statements is added to the code in the editor window.

> **Note:**
>
> Deselect libraries, packages, and models to remove them from a model.

Adding Terminal, Quantity, or Signal Ports to an Entity Description

1. Select the desired port type (**Terminal**, **Quantity**, or **Signal**) from the **VHDL Model Editor** > **Edit Entity** menu. The **Edit Model** dialog box opens with the selected port type highlighted.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You can choose the port type in the **Edit Model** dialog box by selecting the desired item in the **Entity** tree. The data entry pane to the right of the tree changes to reflect the selection.

2. Select a model from the **Model** field.
3. Click to add a new row to the list in the **Port** field. If this is the first time you have visited this field for a new model, the list will be empty.
    a. Type a name for the new port in the **Name** field.
    b. For **Quantity** or **Signal** ports, select a direction for the new port from the list available in the **Direction** field: **in**, **out**, or **inout**.
    c. For **Terminal** ports, select one of the natures supported by VHDL-AMS from the list available in the **Nature** field. For **Quantity** or **Signal** ports, select a type in the **Type** field. These lists contain a number of physical, numerical, and logical types (for example, **voltage**, **real**, **bit**).
    d. If applicable, type a default value for the port in the **Value** field.
    e. Optionally, add a description for the port in the **Description** field.
4. Repeat Steps 1 through 3 as needed for additional ports.
5. Click **OK** when finished to close the dialog box.

   The editor generates the VHDL-AMS code for the specified ports and adds it to the model code in the editor window. Syntax is checked and errors displayed in the **Message Manager** pane.

Adding Generics to an Entity Description

1. Select **VHDL Model Editor** > **Edit Entity** > **Generic**. The **Edit Model** dialog box opens with the **Generic** type highlighted.

> **Note:**
>
> You can choose the **Generic** type in the **Edit Model** dialog box by selecting **Generic** in the **Entity** tree. The data entry pane to the right of the tree changes to reflect the selection.

2. Select the name of the appropriate model from the **Model** field.
3. Click to add a new row to the list in the **Generic** field.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

a. Type a name for the new generic in the **Name** field.
b. Select a data type for the new generic from the **Type** field. This list contains a number of physical, numerical, and logical types.
c. Type a default value for the generic in the **Value** field.
d. Optionally, add a description for the generic in the **Description** field.

4. Repeat Step 3 as needed for additional generics.
5. Click **OK** when finished to close the dialog box.

   The editor generates the VHDL-AMS code for the specified generics and adds it to the model code in the editor window. Syntax is checked and errors displayed in the **Message Manager** pane.

Editing an Entity

To edit an entity:

1. Select the desired entity type (**Terminal**, **Quantity**, **Signal**, or **Generic**) from the **VHDL Model Editor** > **Edit Entity** menu. The **Edit Model** dialog box opens with the selected type highlighted.

   > **Note:**
   >
   > You can choose a type in the **Edit Model** dialog box by selecting the desired item in the **Entity** tree. The data entry pane to the right of the tree changes to reflect the selection.

2. Select the name of the appropriate model from the **Model** field.
   - To edit existing information, click the information and edit it.
   - To remove an existing definition, select the line and click the red **X** to delete it.
3. Click **OK** to complete the editing process.

Renaming an Entity

To rename an entity:

1. Select **VHDL Model Editor** > **Rename Entity**. The **Edit Model** dialog box opens with **Entity** selected in the tree view.
2. Click the entity name to edit it.
3. Click **OK** when finished.

Adding an Architecture

To add an architecture:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Click **VHDL Model Editor** > **Add Architecture**. The **Edit Model** dialog box opens with **Architecture** selected in the tree view.

   > **Note:**
   >
   > In the **Edit Model** dialog box, click **Architecture** in the tree to choose an **Architecture**. The data entry pane to the right of the tree changes to reflect the selection.

2. Click to add a new row to the list in the **Architecture** field.
   - Type a name for the new architecture in the **Name** field.
3. Repeat Step 2 as needed for additional architectures.
4. Click **OK** when finished to close the dialog box.

   The editor generates the VHDL-AMS code for the new architectures and adds a tab for each new architecture in the editor window.

Editing an Architecture Declaration

To edit an architecture declaration:

1. Select the desired object (**Terminal**, **Quantity**, **Signal** or **Constant**) from the **VHDL Model Editor** > **Edit Architecture** menu. The **Edit Model** dialog box opens with the selected object type highlighted.

   > **Note:**
   >
   > You can choose an object in the **Edit Model** dialog box by selecting the desired item in the **Architecture** tree. The data entry pane to the right of the tree changes to reflect the selection.

2. Select the name of the appropriate model from the **Model** field.
3. Select the name of the architecture you want to edit from the list in the **Architecture** field.
4. Click to add a new row to the list in the object field. If this is the first time you have visited this field for a new model, the list will be empty.
   a. Type a name for the new object in the **Name** field.
   b. For **Terminal** objects:
      - Select one of the natures supported by VHDL-AMS from the **Nature** field.
      - Optionally, add a description for the terminal in the **Description** field.
   c. For **Quantity** objects:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Select a **Category** (**free**, **across**, or **through**) for the new quantity.
- Select one of the types (for example, **voltage**, **reluctance**, **current**) available in the **Type** field.
- If applicable, type a default value for the quantity in the **Value** field.
- If the **Category** is **across** or **through**, select the appropriate **From Terminal** and **To Terminal** to be used by the quantity object.
- Optionally, add a description for the quantity in the **Description** field.

d. For **Signal** objects:

- Select one of the types (for example, **voltage**, **reluctance**, **bit**) available in the **Type** field.
- If applicable, type a default value for the signal in the **Value** field.
- Optionally, add a description for the signal in the **Description** field.

e. For **Constant** objects:

- Select one of the types (for example, **voltage**, **inductance**, **bit**) available in the **Type** field.
- If applicable, type a default value for the signal in the **Value** field.
- Optionally, add a description for the signal in the **Description** field.

5. Repeat Steps 1 through 4 as needed for additional objects.
6. Click **OK** when finished to close the dialog box.

   The editor generates the VHDL-AMS code for the specified objects and adds it to the model code in the editor window. Syntax is checked and errors displayed in the **Message Manager** window.

Renaming an Architecture Declaration

To rename an architecture declaration:

1. Select **VHDL Model Editor** > **Rename Architecture**. The **Edit Model** dialog box opens with **Architecture** selected in the tree view.
2. Click the architecture name to edit it.
3. Click **OK** to complete the editing process.

Checking Syntax (VHDL Model Editor)

Follow this procedure to check the syntax of your code:

1. Select **VHDL Model Editor** > **Check Syntax** to check the syntax of the new VHDL model.

2. Twin Builder checks the code for syntax errors. Twin Builder's **Message Manager** pane displays the results of the syntax check. Click errors to view the associated lines of code in the editor window.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Instantiating a Predefined Component in an Architecture Description

Structural VHDL-AMS architecture descriptions require the use of predefined component models. To instantiate a predefined component with the VHDL Model Editor:

1. Click **VHDL Model Editor** > **Instantiate Component**. The **Instantiate Component** dialog box opens.
2. Select the name of the **Architecture** description in which the component is to be instantiated.
3. In the **Available Components** section of the dialog box, browse to the desired component.
4. Double-click the desired component. A new instance of the model appears in the **Instantiated Components** section.
   a. Type a name for the model instance in the **Name** field.
   b. Select an entity and an architecture for the model instance from the lists available in the **Entity** and **Architecture** fields.
5. In the **Ports** section, enter a numerical value or the name of a port or generic in the **Value** field for each of the model instance's ports. Terminals should be mapped to other terminals, while quantities and signals can be mapped to ports of the same type, or to generics, or can just be given numerical values.
6. In the **Generics** section, select a value for each generic of the model. The value may be a numerical value or generic and can be typed into the field or selected from the list of available values in the **Value** field.
7. Once all desired model instances have been added to the architecture description, click **OK**. The editor generates the VHDL-AMS statements that instantiate the models and map their ports.

Updating a Project to Add a VHDL-AMS Model and Component

Once the entity description and all architecture descriptions of a model are completed, you can compile the model and add it to the Twin Builder project as a component.

You can update a project to both compile your VHDL model, and create a component as follows:

1. Click **VHDL Model Editor** > **Compile & Update Project**.
2. The **Import Components** dialog box appears, listing all of the models in the current VHDL Model project on the **Models** tab.
3. See *Importing Simulation Models* for detailed information on settings you can make in this dialog box. When finished, Twin Builder compiles the VHDL-AMS model source code.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- If the compiler detects an error or issues a warning, it appears in the **Message Manager** window along with the line number of the statement that caused the error or warning. Press **Ctrl+G** to open the **Go To Line** dialog box to go to the line.
- If the compilation of the VHDL-AMS Model succeeds, a message appears in the **Message Manager** window indicating that the project update was successful.
- A component for each model is added to the **Project Manager Component Definitions** folder. The new models appear on the list of **Project Components** from which they can be placed onto a schematic for simulation.

Related Topics

Importing Simulation Models

Encrypting or Encoding a VHDL-AMS Model

Follow this procedure to encrypt or encode a VHDL-AMS model:

1. Select **VHDL Model Editor** > **Encryption Settings**. The **Encryption Settings** dialog box appears.
2. Select the **Architecture** you want to encrypt from the drop-down list.
3. Click **OK** to confirm your choice and close the dialog box.

   The **Encryption Settings for** *<model_name>* dialog box for the selected architecture displays. Do either of the following:

   a. If you want to encrypt the model, click the **Encrypt** radio button and select the desired encryption resource in the **Resource Name** list.

   > **Note:**
   >
   > You can also click **Add** to add a new encryption resource.

   b. If you want to encode the model, click the **Encode** radio button. Encoding prevents the model text from being viewed.
4. Click **OK** to complete the process.

Related Topics

Using the Password Manager

Encryption Settings Dialog Box

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Using IEEE Standard Encryption for VHDL-AMS Models in Twin Builder

Twin Builder Encryption Public Key

Apply IEEE standard encryption to a VHDL-AMS model with the following Twin Builder public key:

`protect key_keyowner = "Ansys Incorporated"
`protect key_method = "rsa"
`protect key_keyname = "AI-TWINBUILDER-RSA-1"
`protect key_public_key
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA0d0KwK3YbsXnKu0aulNDzpQD+
KlGHT7QVzHISqMt22y+lpgsatFcyxlS8ZiNIr7BA1Gvy/cYZ8UFmu56lPeak/dEbubZmwJaBprO
1JCkKxFrZ1il/fqDBqjyrcVVYJ5U54HOyQxqglQuzFZoIwkjY6BjVTsfkkPkDvkkLUSS+RbhbVM2
C4/t03YNsgd4GpkOIndEZzVcFKWdrdYrdloqUz8NIXjmMWPuYsiovSbfYq+KDEcJDRLjIYE2n0
l3V+8TrQ1SSMc+TNaqkuHdBHA5sSGd4Dz4JTfhqjvLQG+GltbZrocHKMeCFRRqYzifolFmad3
zFxuBBd1kZVx6TRFgwwIDAQAB

Twin Builder Command-line Encryption Tool

1. In a Command Prompt window, enter:

   <*Ansys Twin Builder install path*>**TwinBuilder_encryptvhdl.exe** <*source file*>

   Limitations

   - **Ansys Twin Builder only supports encrypting the entire architecture of the model.**
     The `` `protect `` directive must be inserted just after the architecture definition, that is, `architecture <behav> of <entity>`.
     The `` `protect `` end directive must be just before the end architecture statement.
   - Ansys Twin Builder only supports the `` `protect begin `` and `` `protect end `` directives. No other `` `protect `` directives are supported and will cause an error to be output by the encryptor.
   - The Ansys tool will, by default, generate encrypted session keys for Twin Builder.
   - Ansys Twin Builder only supports the **aes-128-cbc** symmetric encryption algorithm for encrypting the model.
   - Ansys Twin Builder only supports the **RSA asymmetric encryption algorithm** for encrypting the session key.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Decryption in Twin Builder

Ansys Twin Builder supports decrypting models that have been encrypted using the Ansys Twin Builder public key above. The workflow has been tested with the Twin Builder encryption tool as well as the Mentor Graphics® encryption tool.

Limitation

Ansys Twin Builder does not support Section 9, standard 1735, **Visibility management**, of the IEEE Recommended Practice for Encryption and Management of Electronic Design Intellectual Property (IP).

When you load an encrypted model (regardless of the tool used to encrypt the model) into Ansys Twin Builder:

- Ansys Twin Builder does not hide names inside the protected part of the model.
- Ansys Twin Builder does not hide hierarchical paths through the protected part of the model.
- If an error or warning is generated from inside the protected part of the model, Ansys Twin Builder may include references to protected names as well as to original file line numbers.

Revising an Existing VHDL Model

Use the VHDL-AMS Model Editor to revise a VHDL-AMS model for which a .**vhd** file is available. To revise a model:

1. Click **Tools** > **Library Tools** > **Edit External Model** > **Information**.
2. In the **Edit Models** dialog box, browse to the .**vhd** model containing the desired model, and open it.
3. Select the desired VHDL-AMS model and click **OK** to edit the model in the VHDL Model Editor.

VHDL-AMS Model Editor Menus and Windows

This section gives a brief description of the available menus and windows when using the VHDL Model Editor.

- VHDL Model Editor Main Menu
- VHDL-AMS Edit Model Dialog Box
- VHDL Model Editor Standard Toolbar
- VHDL-AMS Editor Window
- VHDL-AMS Editor Message Window

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

VHDL Model Editor Main Menu

The **VHDL Model Editor** main menu contains these commands:

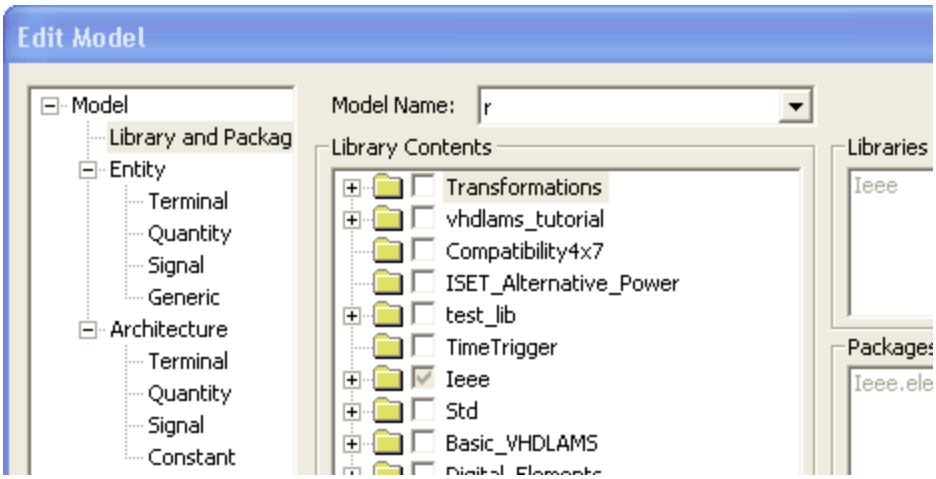
- **Compile & Update Project** – Performs a syntax check and compiles the VHDL-AMS model. Also enables creation of components for successfully compiled models, and their addition to the current project. Twin Builder's **Message Manager** window displays the results of the operation. Also see ***C-Model Editor Main Menu***.
- **Check Syntax** – Checks the code for syntax errors. Twin Builder's **Message Manager** window displays the results of the syntax check.
- **Save Uncompiled Changes** – Saves the model text to the model without having it compiled. See ***Saving Uncompiled Model Changes***.
- **Export Model Text As** – Opens a dialog box to save and export the code present in the editor.
- **Revert** *ModelName* **Changes** – Reverts current changes to the last compiled state.
- **Encryption Settings** – Opens a dialog box for selecting a VHDL **Architecture** for encryption or encoding via the Password Manager.
- **Time License Settings** – Opens a dialog box for enabling or disabling Time License Settings used to control how long a model may be used in simulations.
- **Variables & Declarations** – Opens the **Edit Model** dialog box for configuring the variables and declarations to be included in a VHDL-AMS model.
- **Edit Libraries & Packages** – Opens the **Edit Model** dialog box for configuring the libraries, packages, and models to be included in a VHDL-AMS model.
- **Edit Entity** – Contains commands that open the **Edit Model** dialog box for editing a VHDL model entity's **Terminal**, **Quantity**, **Signal**, and **Generic** definitions.
- **Rename Entity** – Opens the **Edit Model** dialog box where you can rename a VHDL model entity.
- **Add Architecture** – Opens the **Edit Model** dialog box for adding a VHDL architecture to a VHDL model.
- **Rename Architecture** – Opens the **Edit Model** dialog box for renaming a VHDL model architecture.
- **Edit Architecture** – Contains commands that open the **Edit Model** dialog box for editing a VHDL model architecture's **Terminal**, **Quantity**, **Signal**, and **Constant** definitions.
- **Instantiate Component** – Opens a dialog box for setting paths to C++ files to be included in the building of a C-Model project, specifying preprocessor definitions, additional dependencies, and for setting library paths different from the default path.

VHDL-AMS Edit Model Dialog Box

The **Edit Model** dialog box displays a tree view of a model's editable objects in its left panel. The area to the right of the tree view displays the editing options for the object currently selected in

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the tree view.



The editable content for each of the tree objects is as follows:

- **Library and Packages** – Choose the libraries, packages, and models used in the selected model.
- **Entity** – Rename a model entity.
- **Entity** > **Terminal** – Add, delete, and edit the **Name**, **Nature** (such as electrical and magnetic), and **Description** of terminal ports for the selected model entity.
- **Entity** > **Quantity** – Add, delete, and edit the **Name**, **Type** (such as capacitance, and voltage), **Value**, **Direction** (In or Out), and **Description** of quantity ports for the selected model entity.
- **Entity** > **Signal** – Add, delete, and edit the **Name**, **Type** (such as capacitance and voltage), **Value**, **Direction** (In, Out, or InOut), and **Description** of signal ports for the selected model entity.
- **Entity** > **Generic** – Add, delete, and edit the **Name**, **Type** (such as capacitance and voltage), **Value**, and **Description** of generics for the selected model entity.
- **Architecture** – Add, delete, and rename model architectures.
- **Architecture** > **Terminal** – Add, delete, and edit the **Name**, **Nature** (such as electrical and magnetic), and **Description** of terminal ports for the selected model architecture.
- **Architecture** > **Quantity** – Add, delete, and edit the **Name**, **Category** (free, across, or through), **Type** (such as capacitance and voltage), **Value**, **Direction** (In or Out), and **Description** of quantity ports for the selected model architecture.
- **Architecture** > **Signal** – Add, delete, and edit the **Name**, **Type** (such as capacitance and voltage), **Value**, **Direction** (In, Out, or InOut), and **Description** of signal ports for the selected model architecture.
- **Architecture** > **Constant** – Add, delete, and edit the **Name**, **Type** (such as capacitance and voltage), **Value**, and **Description** of constants for the selected model architecture.

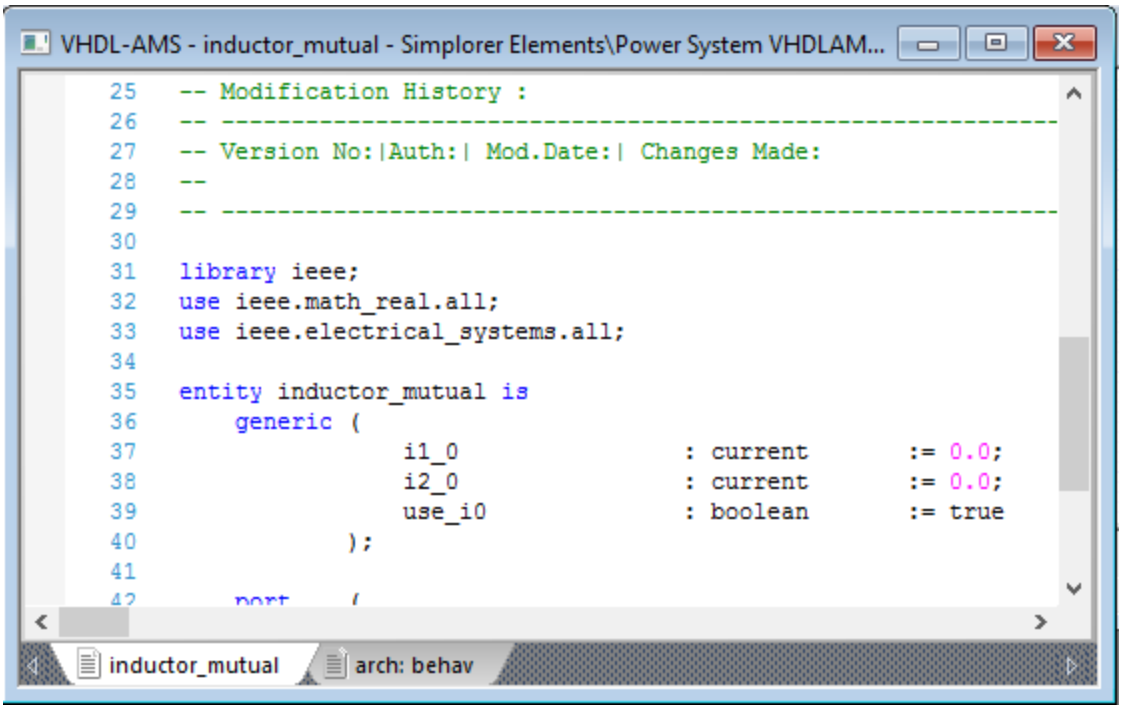
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

VHDL Model Editor Standard Toolbar

See Netlist Editor Toolbar for detailed information on standard editor toolbar functions.

VHDL-AMS Editor Window

The VHDL-AMS editor window displays entity and architecture declarations for the current VHDL-AMS Model project. Click the tabs at the bottom of the window to select an entity or architecture declaration for editing.



The editor provides automatic syntax highlighting as follows:

- VHDL-AMS statement and declaration keywords are **blue**.
- Comments are **green**.
- Numerical values are **magenta**.

Right-click in the editor window to open the VHDL-AMS editor context menu.

> **Note:**
>
> The VHDL-AMS editor context menu commands given below are also used in other Twin Builder text editors such as the SML, C, and SPICE model editors.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In addition to standard **Cut**, **Copy**, **Paste**, and **Select All** commands, other context menu commands include:

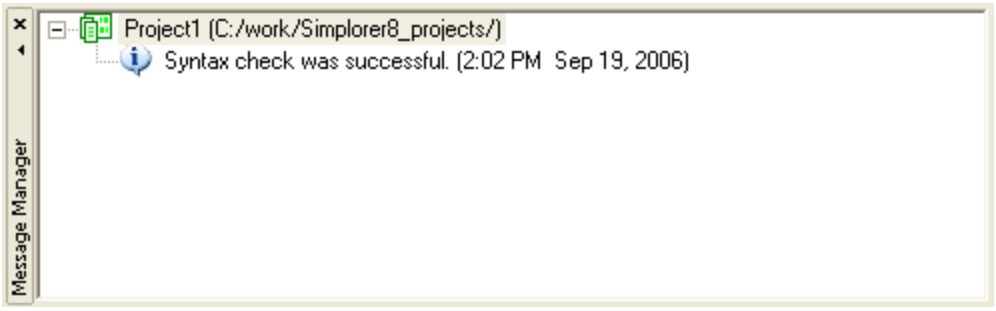
- **Comment Line(s)** – Select one or more lines of code in the editor and apply this command to change the code lines into comment statements.
- **Uncomment Line(s)** – Select one or more commented lines of text in the editor and apply this command to change the lines into code statements.
- **Find** – Open the **Find** dialog box in which you can enter strings to search for in the editor window. You can make the search case-sensitive, as well as control the direction of the search relative to the current cursor position.
- **Replace** – Similar to the **Find** command and dialog box, with the additional capability to include a **Replace with** string.
- **Go To Line** – See *Going to a Numbered Line* for details.

Related Topics

Text Editor Options

VHDL-AMS Editor Message Window

The **Message Manager** window displays messages such as compiler warnings, syntax errors, and build status.



Using the SML Model Editor

Use the **SML Model Editor** to edit SML model code, manage syntax checking, and print the code listing.

Models generated using the **SML Model Editor** are listed under **Project Components** in the **Project Manager** components tree, and can be used in the same way as other Twin Builder components. Their properties can be edited using Twin Builder's **Component Editor**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

- Text Editor Options
- Adding an SML Model
- Editing an External Model
- Saving Uncompiled Model Changes
- Inserting a Dataset in the SML Editor
- Checking Syntax
- Printing
- Encrypting or Encoding an SML, or SPICE Model
- Importing Simulation Models

Adding an SML Model

Follow this procedure to add an SML model to a project using Twin Builder's SML Model Editor:

1. Select **Tools** > **Edit Libraries** > **Models**. The **Edit Libraries** dialog box appears.
2. Click **Add Model** to open the **Add Model** dialog box.
3. Select the **SML** radio button.
4. Type the new SML model name in the **Name** field.

   The **SML Model Editor** opens, displaying new SML model entity declarations in a tab bearing the model name you chose above.
5. Edit the SML model source code as needed.
6. Optionally, you can check syntax.
7. When finished editing, select **SML Model Editor** > **Compile & Update Project** to add the model to the project. See ***Importing and Updating Twin Builder Models and Components*** for additional details.

You can use **Export** *ModelName* **Text As** to open a dialog to save and export the code present in the editor.

Related Topics

Importing Simulation Models

SML Editor Window

The SML editor window displays the current SML Model project code. Here you can directly edit the SML model code.

The editor provides automatic syntax highlighting as follows:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- SML keywords are **blue**.
- Literals are **red**.
- Numerical values are **magenta**.

Right-click the SML editor window to display its context menu.

> **Note:**
>
> The SML editor context menu commands given below are also used in other Twin Builder text editors such as the VHDL-AMS, C, and SPICE model editors.

In addition to standard **Cut**, **Copy**, **Paste**, and **Select All** commands, other context menu commands include:

- **Comment Line(s)** – Select one or more lines of code in the editor, then select **Comment Line(s)** to change the code lines into comment statements.
- **Uncomment Line(s)** – Select one or more commented lines of text in the editor, then select **Uncomment Line(s)** to change the lines into code statements.
- **Find** – Open the **Find** dialog box in which you can enter strings to search for in the editor window. you can make the search case-sensitive, as well as control the direction of the search relative to the current cursor position.
- **Replace** – Similar to the **Find** command and dialog box, with the additional capability to include a **Replace with** string.
- **Go To Line** – See *Going to a Numbered Line* for details.

Related Topics

Text Editor Options

Editing an External Model

The SML, VHDL-AMS, C, and SPICE model editors can revise models for which an external file is available. To revise a model located in an external file:

1. Select **Tools** > **Library Tools** > **Edit External Twin Builder Model**. The **Edit Models** dialog box appears.
2. Browse to the desired model file and open it. You can click **Use Path**, **SysLib**, **PersonalLib**, **UserLib**, or **Project Folder** to change the folder location for selecting the model file.

   The **Import Components** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The **Models** tab lists the names of models that can be imported for editing, provides a check box to update the project definitions, and a check box that allows existing models of the same names to be overridden. A table lists the names of models that will be imported for editing. If a model name already exists, select the **Override** check box to override the current model definition.

The **Un-Compiled** tab is present only if one or more of the components you are importing cannot be compiled. Each un-compiled component is listed along with the **Edit** check box which is selected by default. Selected entries open in the appropriate model editor when you **OK** the **Import Components** dialog box. Un-compiled models will not be imported, nor will components be created for them.

3. Select the desired model and click **OK** to import and edit the model in the appropriate model editor.

> **Note:**
>
> When referencing external files in SPICE model code, use either file names (*abcd*.**txt**) or a *relative path + file names*, where the relative path can be to the **syslib**, **userlib** or **personallib** directories. If Twin Builder cannot locate the external file in any of the above locations when you check syntax or attempt to **Compile & Update Project**, the Message Manager displays an alert prompting you to copy the referenced files into either the **personallib** or **userlib** directory. This ensures that the external files can be located during compilation.

4. When finished editing, optionally check syntax.
5. Select *model_type* **Model Editor** > **Compile & Update Project** to update the project model. Project components that use the revised model are also updated (that is, synchronized).

Related Topics

- Importing Simulation Models

Inserting a Dataset in the SML Editor

When an SML file is open in the corresponding editor, follow this procedure to insert a dataset in the code:

1. Place the cursor at the desired insertion point in the editor.
2. Right-click and select **Insert Dataset** from the context menu. The **Insert MD Dataset** dialog box displays in which you can:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- manually enter dataset values
- import dataset values from a file

a. If you want to enter dataset values manually, see Dataset Parameters Tabs.

b. If you want to import a dataset from a file, click **Import**, and choose from the following file types:

.mdx, .mda	Twin Builder characteristic format.
.txt	Text file.
.csv	Comma-separated value.
.out	Maxwell SPICE (read-only – reads data inside the KW_DATA section).
.dat	TEK Oscilloscope.

- If you select a file type other than **.txt** or **.csv** the data is imported immediately into the **Add Dataset** dialog box.
- Select a **.txt** or **.csv** file to open the **Import** dialog box in which you can specify settings for reading the data in the file for import. You can choose the **Separator**(s) and **Decimal Symbol**, as well as the line at which to begin the import. The dialog box shows both the original text and the text as it would appear when imported based on the current import settings.

d. When satisfied with the import settings, click **OK** to import the data.

e. Manually edit the imported data, if desired. See *Dataset Parameters Tabs*.

3. Click **OK** to import the dataset into the model file.

Dataset Parameters Tabs

The SML Editor contains two tabs for inserting dataset parameters in an SML file: **Input Parameters** and **Output Parameters**.

> **Note:**
>
> These tabs are also used for setting table values for multi-dimensional table components (NDSRC, NXMY, NDTAB, and NDNL).

Input Parameters Tab

- **Edit Input Channels** – Create or delete channel entries using the symbols on the upper right side. Click in the fields for a list entry to define a channel **Name** and **Unit**.
- **Channel Data**<*Input Channel Name*> – Define the corresponding characteristic data for the selected input channel. The name of the selected input channel displays in the panel name, and also in the column heading of the data table. Create, delete, or move table entries with the symbols located above the table. Click in the fields to enter values.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You cannot use variables or expressions. Only real numbers are accepted.

If the **Sweep Data** box is selected, click **Set** to replace the values in the lookup table with values generated based on the **Start Value**, **Step** size, and **Number of Samples** settings. Click **Append** to insert the new values at the end of existing table entries.

Import: (This button is present only when in the SML Editor.) Opens a file browser in which you can select the dataset values to import. See ***Inserting a Dataset in the SML Editor***.

Output Parameters Tab

- **Edit Output Channels** – Create or delete channel entries using the symbols on the upper right side. Click in the fields for a list entry to define a channel **Name** and **Unit**.
- **Edit Output Data** – Define output channel values. Output channel names appear in column headings. Input channel names and data are also shown for reference. Click in the output channel fields to enter values for output quantity.

> **Note:**
>
> You cannot use variables or expressions. Only real numbers are accepted.

- Click the various in **Tree View** to navigate the output data table.

Checking Syntax (SML, and SPICE Model Editors)

Follow this procedure to check the syntax of code you have entered in the SML or SPICE model editor:

1. Select *model_type* **Model Editor** > **Check Syntax** to check the syntax of the model.
2. Twin Builder checks the code for syntax errors. The **Message Manager** pane displays the results of the syntax check. If errors are reported, click them to go to the associated lines of code in the editor window.

Related Topics

Checking Syntax (VHDL Model Editor)

Checking Syntax (C-Model Editor)

Encrypting or Encoding an SML or SPICE Model

Follow this procedure to encrypt or encode an SML or SPICE model:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select *model_type* **Model Editor** > **Encryption Settings**. The **Encryption Settings** dialog box appears.
    a. To encrypt the model, click the **Encrypt** radio button and select the desired encryption resource from the **Resource Name** list.

    > **Note:**
    >
    > You can also click **Add** to add a new encryption resource.

    b. To encode the model, click the **Encode** radio button. Encoding prevents the model text from being viewed.
2. Click **OK** to complete the process.

Related Topics

Using the Password Manager

Time License Settings

Encryption Settings Dialog Box

Using the SPICE Model Editor

Use the **SPICE Model Editor** to edit SPICE model code, check syntax, and print the code listing.

Models generated using the **SPICE Model Editor** appear under **Project Components** in the **Project Manager** components tree, and are used in the same way as other Twin Builder components. You can edit their properties with Twin Builder's **Component Editor**.

Related Topics

- Text Editor Options
- Adding a SPICE Model
- Editing an External Model
- Saving Uncompiled Model Changes
- Checking Syntax
- Printing
- Encrypting or Encoding an SML or SPICE Model
- Importing Simulation Models

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding a SPICE Model

Follow this procedure to add a SPICE model to a project using Twin Builder's SPICE Model Editor.

1. Select **Tools** > **Edit Libraries** > **Models**. The **Edit Libraries** dialog box appears.
2. Click **Add Model** to open the **Add Model** dialog box.
3. Select the **SPICE** radio button.
4. Type the new SPICE model name in the **Name** field.

   The **SPICE Model Editor** opens displaying new SPICE model entity declarations in a tab bearing the model name you chose above.
5. Edit the SPICE model source code as needed.
6. Optionally, you can check syntax.
7. When finished editing, select **SPICE Model Editor** > **Compile & Update Project** to add the model to the project. See ***Importing and Updating Twin Builder Models and Components***.

You can use **Export** *ModelName* **Text As** to save and export the code present in the editor.

Related Topics

Importing Simulation Models

SPICE Editor Window

The SPICE editor window displays the current SPICE Model project code.

The editor provides automatic syntax highlighting as follows:

- SPICE keywords are **blue**.
- Numerical values are **magenta**.

Right-click in the editor window to display the SPICE editor shortcut menu.

> **Note:**
>
> The SPICE editor context menu commands given below are also used in other Twin Builder text editors such as the SML, VHDL-AMS, and C-Model editors.

In addition to standard **Cut**, **Copy**, **Paste**, and **Select All** commands, other context menu commands include:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Comment Line(s)** – Select one or more lines of code in the editor, then select **Comment Line(s)** to change the code lines into comment statements.
- **Uncomment Line(s)** – Select one or more commented lines of text in the editor, then select **Uncomment Line(s)** to change the lines into code statements.
- **Find** – Opens the **Find** dialog box in which you can enter strings to search for in the editor window. Use the controls to make the search case-sensitive, and to control the direction of the search relative to the current cursor position.
- **Replace** – Similar to the **Find** command and dialog box, with the additional capability to include a **Replace with** string.
- **Go To Line** – See ***Going to a Numbered Line*** for details.

Related Topics

Text Editor Options

Using the Package Editor

Use the **Package Editor** to edit VHDL package information, check syntax, and print the package code listing.

Packages generated using the **Package Editor** are listed under **Packages** in the **Project Manager** components tree, and are used in the same way as other Twin Builder VHDL-AMS packages.

Related Topics

- Text Editor Options
- Adding a Package
- Editing a Package
- Checking Package Syntax
- Printing
- Encrypting or Encoding a Package

Adding a Package

Follow this procedure to add a package to a project using Twin Builder‘s Package Editor.

1. Click **Tools** > **Edit Libraries** > **Packages**. The **Edit Libraries** dialog box appears.
2. Click **Add Package** to open the **Add Package** dialog box.
3. Select **VHDL-AMS** from the **Model Type** drop-down list.
4. Type the new package name in the **Name** field.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Twin Builder gives you the choice of adding the new model directly to the current project (for later editing, perhaps), or of adding the model using Twin Builder's **Package Editor**. Choose one of the following:
   a. To add a new model directly to the current project, clear the **Add using editor** check box and click **OK**.

      Twin Builder creates a new VHDL-AMS package bearing the chosen package name, and adds the model to the **Project Manager**'s **Definitions** > **Packages** folder.

> **Note:**
>
> To open the package for editing from the Project Manager, double-click the package, or right-click the package and select **Edit Package**.

   b. To add a new package to the current project using the **Package Editor**, select the **Add using editor** check box. Click **OK** to close the dialog box.

      The **Package Editor** opens displaying new VHDL-AMS package entity declarations in a tab bearing the package name you chose above.

6. Edit the package source code as needed.
7. When finished editing, select **Vhdl Package Editor** > **Update Project** to add the model to the project.

Editing a Package

Follow this procedure to revise a package in the Package Editor.

1. Select **Tools** > **Edit Libraries** > **Packages**. The **Edit Libraries** dialog box appears.
2. Select the package you want to edit.
3. Click **Edit Package** to open the package in the **Package Editor**.
4. Edit the package as needed.
5. When finished editing, check syntax if desired.
6. Click **Vhdl Package Editor** > **Update Project** to update the project model.

Package Editor Window

The package editor window displays the current VHDL-AMS package code.

The editor provides automatic syntax highlighting as follows:

- VHDL-AMS keywords are **blue**.
- Literals are **red**.
- Numerical values are **magenta**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Checking Package Syntax

Follow this procedure to check the syntax of code you have entered in the editor:

1. Select **Vhdl Package Editor** > **Check Syntax** to check the syntax of the SML model.
2. Twin Builder checks the code for syntax errors. The **Message Manager** window displays the results of the syntax check. If errors are reported, click them to go to the associated lines of code in the editor window.

Encrypting or Encoding a VHDL-AMS Package

To encrypt or encode a VHDL-AMS package:

1. Click **Vhdl Package Editor** > **Encryption Settings**. The **Encryption Settings** dialog box appears.
   a. To encrypt the package, click the **Encrypt** radio button and select the desired encryption resource from the **Resource Name** list.

      > **Note:**
      >
      > Click **Add** to add a new encryption resource.

   b. To encode the package, click the **Encode** radio button. Encoding prevents the package text from being viewed.
2. Click **OK** to complete the process.

Related Topics

Using the Password Manager

Time License Settings

Encryption Settings Dialog Box

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Using the Modelica Model Editor

Use the **Modelica Model Editor** to edit and compile Modelica model code, check syntax, generate components, and print the code.

You can also access the Diagram Editor for Modelica.

Components generated using the **Modelica Model Editor** appear under **Project Components** in the **Project Manager** components tree and are used in the same way as other Twin Builder components. You can edit their properties with Twin Builder's **Component Editor**.

Related Topics

- Modelica Models in Twin Builder
- Compiling & Updating a Project
- Checking Syntax (Modelica Model Editor)
- Selecting Model Interfaces
- Saving Uncompiled Model Changes
- Exporting Modelica Model Text As
- Time License Settings
- Encrypting or Encoding a Modelica Model
- Using the Diagram Editor
- Updating Modelica Library Definitions
- Printing

Modelica Models in Twin Builder

Modelica is an open standard non-proprietary, object-oriented, equation-based language used to model complex physical systems containing, for example, mechanical, electrical, electronic, hydraulic, thermal, control, electric power, or process-oriented subcomponents. The language is standardized and developed by the Modelica Association. The association also supports a large open source library developed in Modelica called the Modelica Standard Library (MSL).

For more information on Modelica, see: http://www.modelica.org.

Compiling and Updating a Project to Add a Modelica Model and Component

Once the entity description and all architecture descriptions of a model are completed, the model is ready to be compiled and added to the Twin Builder project as a component.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

You can update a project to both compile your Modelica model and create a component as follows:

1. Select **Modelica Model Editor** > **Compile & Update Project**.
2. After model compilation, select the model interfaces you want to expose in Twin Builder. See ***Selecting Component Interface***.

   > **Note:**
   >
   > Select the **Don't show interface selection during compilation** check box to skip this step in future compilations. To reenable this step, open the **Options** dialog box, select **Model Editor** > **Modelica** > **General**, and select the **Show model interfaces dialog during compilation** check box.

3. The **Import Components** dialog box appears listing all of the models in the current Modelica Model project on the **Models** tab.

   > **Note:**
   >
   > Select the **Don't show again and update component everytime** check box to skip this step in future compilations. To reenable this step, open the **Options** dialog box, select **Model Editor** > **Modelica** > **General**, and select the **Show update component dialog after compilation** check box.

4. See ***Importing and Updating Twin Builder Models and Components*** for detailed information on settings you can make in this dialog box. When finished, Twin Builder compiles the Modelica model source code.
   - If the compiler detects an error or issues a warning, it is displayed in the **Message Manager** pane and will identify the line number of the statement that caused the error or warning. Press **Ctrl+G** to open the **Go To Line** dialog box to go to the line.
   - If the compilation of the Modelica Model succeeds, a message appears in the **Message Manager** pane indicating that the project update was successful.
   - A component for each model appears in the **Project Manager Component Definitions** folder. The new models also appear in the list of **Project Components** from which they can be placed onto a schematic for simulation.

Use External Solver

You can select this option under Modelica Model editor or from the Modelica ribbon. When selected, the compiled model uses an external solver during co-simulation; the following properties are added to the created component:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **_cs_solver** – Solver used in Co-Simulation. 0 - CVode, 1 – fixed step-size explicit Euler.
- **_cs_rel_tol** – relative tolerance when CVode is used.
- **_cs_step_size** – step size for Explicit Euler.

The above three properties control the usage of the external solver for the model.

Initialization of Input Values

Initial values of inputs can be specified using two ways:

- Input pin values at time=0 by setting **UseStartValueInitialization** to “false”.
- Corresponding **_start** parameters by setting **UseStartValueInitialization** to “true”.

The first option is the default option and is appropriate if the source of the input is from constant blocks, FML_INIT block, timer function components, and so on. However, if the input is connected to a quantity that depends on other components, it could lead to initialization failure due to improper initial values. Specifying the initial conditions for the inputs (the second option) is always the preferred way to initialize the model. The start parameters have the same name of the input plus **_start** suffix.

Related Topics

Importing and Updating Twin Builder Models and Components

Using the Modelica Model Editor

Check Object Syntax

Follow this procedure to check the syntax of code you have entered in the editor:

1. Select **Modelica Model Editor** > **Check Syntax** to check the syntax of the new Modelica model.
2. Twin Builder checks the code for syntax errors. The results of the syntax check appear in the **Message Manager** pane. If errors are reported, click them to see the associated lines of code in the editor window.

Check Object Semantics

Follow this procedure to check the semantics of code you have entered in the editor:

1. Select **Modelica Model Editor** > **Check Object Semantics** to check the semantics of the new Modelica model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

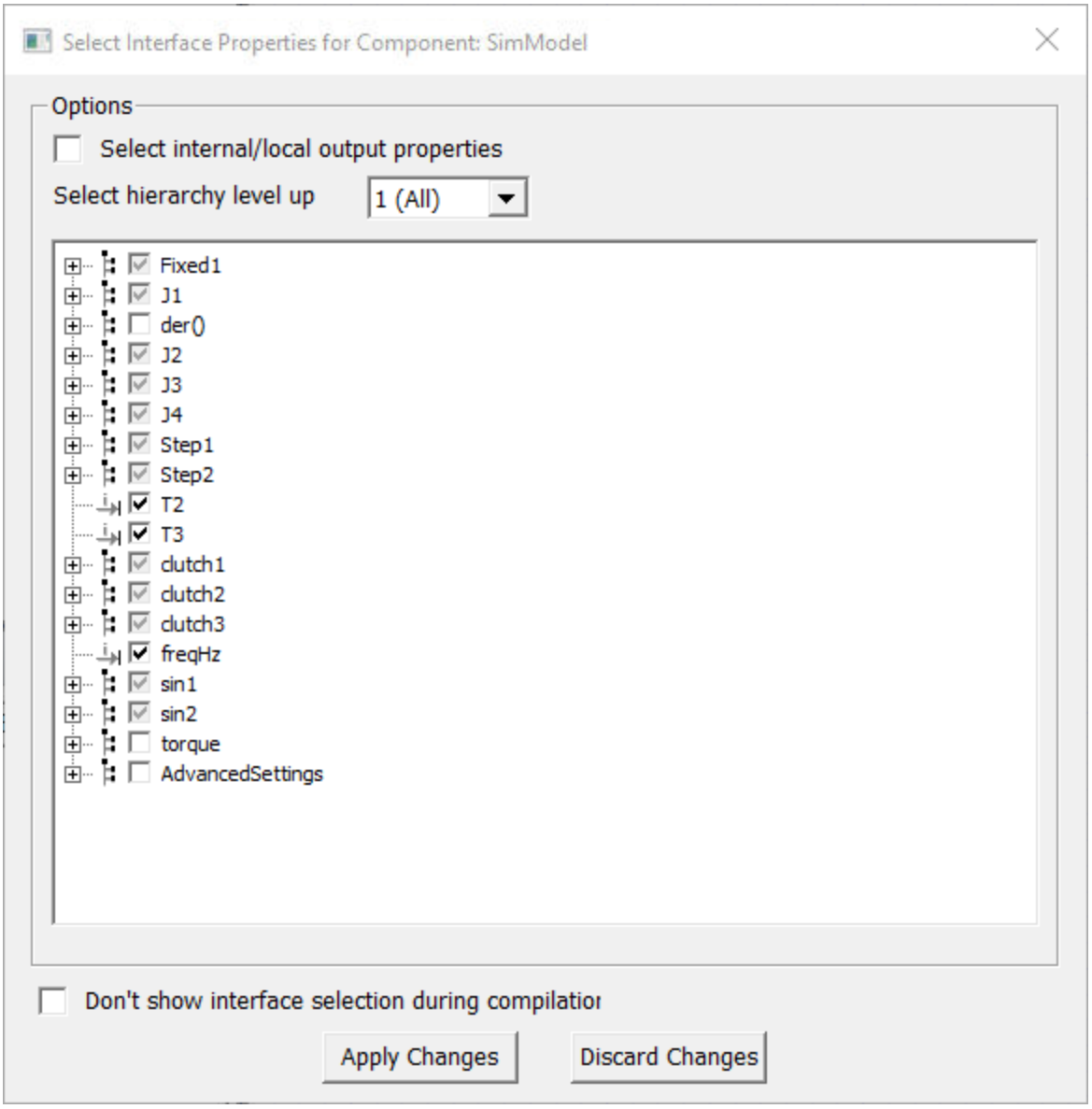
2. Twin Builder checks the code for semantics errors. The results of the semantics check appear in the **Message Manager** pane. Click any errors to highlight the associated lines of code in the editor window.

Selecting Component Interface

Use the **Select** <*model_name*> **Interface** dialog box to choose interfaces to expose in Twin Builder.

> **Note:**
>
> You can edit a component interface for compiled Modelica models from the Schematic and Modelica model editors.
>
> - **Schematic** – Right-click a Modelica component instance on schematic and select **Edit Component Interface**.
> - **Modelica model editor** – Open the Modelica model editor menu and select **Edit Component Interface**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Select the desired model interfaces with these options:

- **Select internal/local output properties** – Include internal or local output properties in the model. Clear this check box to exclude these properties from selection.
- **Select hierarchy level up to** – Choose the desired hierarchy level from the drop-down menu. The value you choose selects input and output interfaces up to the chosen hierarchy level for inclusion in the model. The default value is 1. Hierarchy selections include:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

  - - Indicates an output. When selected, the property is added as an output in the component properties.
  - - Indicates an input. When selected, the property is added as an input in the component properties.
  - - Indicates a constant. When selected, the property is added as an output in the component properties. Constants are not selected by default. You must select them to expose them as component interfaces.
- **Don't show interface selection during compilation** – Select this check box to skip this step in future compilations. To reenable the step, open the **Options** dialog box, select **Model Editor** > **Modelica** > **General**, and select the **Show model interfaces dialog during compilation** check box.

Only the selected interfaces are added to the component; these interfaces are available for you to change/plot/monitor.

> **Note:**
>
> When the number of model interfaces is large, for better performance select only necessary inputs and outputs. Recompile the model to change above selection.
>
> - Interface names that start with "_" are grouped under **AdvancedSettings**. Interface names that indicate derivative (**der(**<*name*>**)**) are grouped under **der()**.

Click **Apply Changes** to finalize your selections and create/update the Twin Builder component with the selected interfaces. Click **Discard Changes** to ignore the current changes and use the default or previous interface selections.

Related Topics

Using the Modelica Model Editor

Exporting Modelica Text As

Use **Export** *ModelName* **Text As** to save and export the code present in the editor.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

To compile Modelica text or import (.**mo** file) which is dependent on another library, specify the file path as MODELICAPATH environment variable (file path up to the folder containing the library).

Use a semicolon (;) to append multiple libraries in MODELICAPATH .

Related Topics

Modelica Model Editor

Time License Settings for Modelica

You can use time license settings to apply an expiration date beyond which a model can not be used in simulations. You can also remove the date stamp from new and editable models. Time license settings can be applied to Modelica models.

To add time license information:

1. Select **Modelica Model Editor** > **Time License Settings**. The **License Information** dialog box appears.
2. Select **Enable Time License settings** then enter the license expiration date into the **Date** field.

   You can also click the down arrow button to the right of the date field to reveal a calendar.

   - The left and right arrow buttons on the calendar change the calendar one month per click.
   - Click the month name to display a menu that lets you jump directly to any month in the currently displayed year.
   - Click directly on the year to use the associated up and down arrow buttons to move sequentially to the desired year.
   - Click **Today:***mm/dd/yyyy* at the bottom of the calendar to select the current date.

   When the desired calendar page displays, click the day of the month to transfer the date selection to the **Date** field in the **License Information** dialog box.
3. Click **OK** to confirm the date entry and close the dialog box.

   The **Encryption Settings** dialog box appears.
4. Encrypt or encode the model or package using the procedure in ***Encrypting or Encoding a Modelica Model***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To remove the time license from a model:

1. With the model from which you intend to remove time licensing open in the model editor, select **Modelica Model Editor** > **Time License Settings** in the menu bar. The **License Information** dialog box appears.
2. Clear the **Enable Time Licensing settings** check box.
3. Click **OK** to remove the settings, then save the model.

Encrypting or Encoding a Modelica Model

Follow this procedure to encrypt or encode a Modelica model:

1. Select **Modelica Model Editor** > **Encryption Settings**. The **Encryption Settings** dialog box appears.
2. To encrypt the model, click the **Encrypt** radio button and select the desired encryption resource in the **Resource Name** list.

   > **Note:**
   >
   > Click **Add** to add a new encryption resource.

3. To encode the model, click the **Encode** radio button. Encoding prevents the model text from being viewed.
4. Click **OK** to complete the process.

Related Topics

Using the Password Manager

Encryption Settings Dialog Box

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7 - Device Characterization Wizard

The semiconductor device characterization wizard aids in the setup of **NPN-type IGBT** and **Power MOSFET** device models to operate according to the device manufacturers' specifications. The wizard accepts inputs for various quantities available from the device manufacturer and fits a numerical model to the data to provide accurate device simulation over a range of device excitations and thermal conditions.

The following types of semiconductor models are available for parameter fitting:

- Average IGBT
- Basic Dynamic IGBT
- Advanced Dynamic IGBT
- Power MOSFET (Basic Dynamic model)
- Power MOSFET (Average model)
- Power Diode
- Thyristor

To start the device characterization wizard, click **Twin Builder** > **Characterize Device** > **Semiconductors** to open the **Characterize Device** dialog box in which you can either start characterization of a new device or continue a saved device characterization.

Related Topics

Characterize Device

Tutorial for Characterizing a Basic Dynamic IGBT

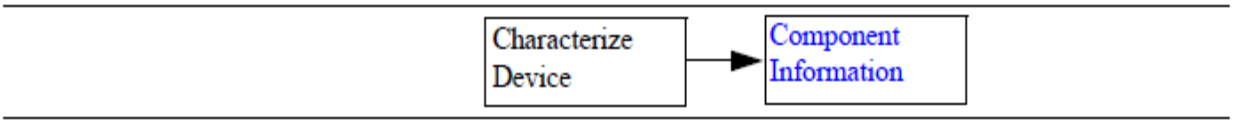
Tutorial for Characterizing a Power Diode

Tutorial for Characterizing a Power MOSFET (Average)

Tutorial for Characterizing a Power MOSFET (Basic Dynamic)

Tutorial for Characterizing a Thyristor

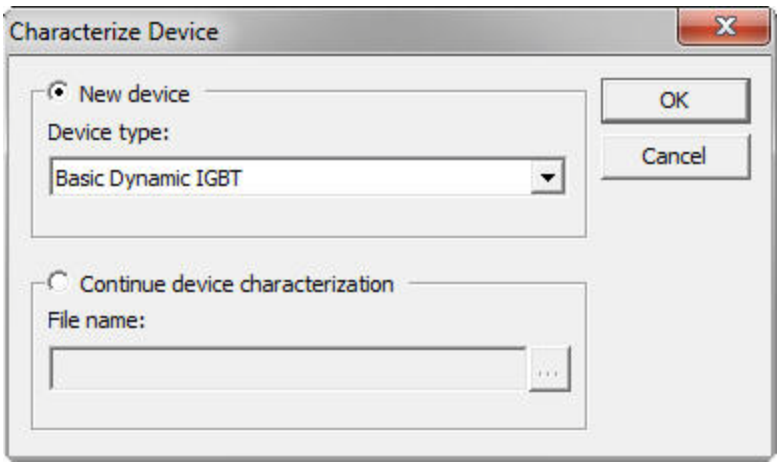
Characterize Device



Follow this procedure to begin the device characterization:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Twin Builder** > **Characterize Device** > **Semiconductors** to open the **Characterize Device** dialog box.



2. Do one of the following:
   - To characterize a new device, select the **New Device** radio button and select a device from the **Device type** drop-down list.
   - To continue device characterization from a previous session, select the **Continue device characterization** radio button and enter the file name from the previous session, or click ... to browse to the file.
3. Click **OK** to start the device characterization wizard for the specified device. The following topics describe the flow of the device characterization wizard for each device type, and provide links to detailed information for each wizard dialog box. Choose the appropriate topic for your specified device type to continue the characterization process.
   - Characterization Flow for IGBT and Power MOSFET (Average, Basic Dynamic, and Advanced Dynamic)
   - Characterization Flow for Power Diode. Details about the Power Diode Model are in the Basic Elements section in the Components help

Related Topics

Device Characterization Wizard Overview

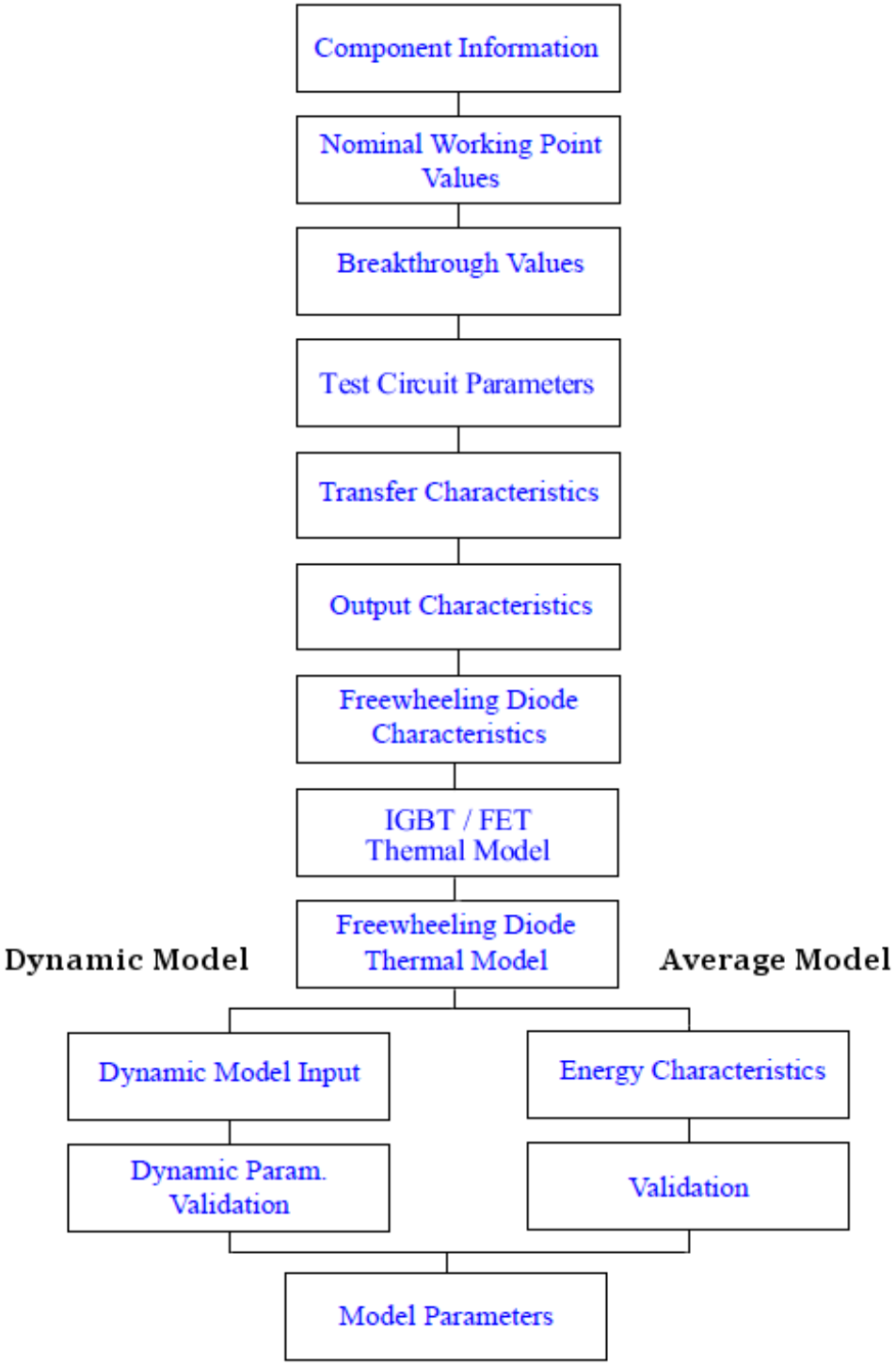
Characterization Flow for IGBT and Power MOSFET

The chart below illustrates the flow of the **Device Characterization** wizard for the **IGBT** and **Power MOSFET models** (**Average**, **Basic Dynamic**, and **Advanced Dynamic**). The topic named in each box is linked to detailed information for the corresponding wizard dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- If you are characterizing a new device, begin by choosing **Component Information**.
- If you are continuing device characterization from a previous session, choose the appropriate topic to resume the process.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Characterize Device

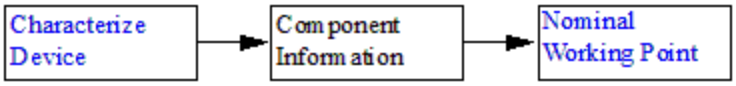
Tutorial for Characterizing a Basic Dynamic IGBT

Tutorial for Characterizing a Power MOSFET (Basic Dynamic Model)

Tutorial for Characterizing a Power MOSFET (Average Model)

Tutorial for Characterizing a Power Diode

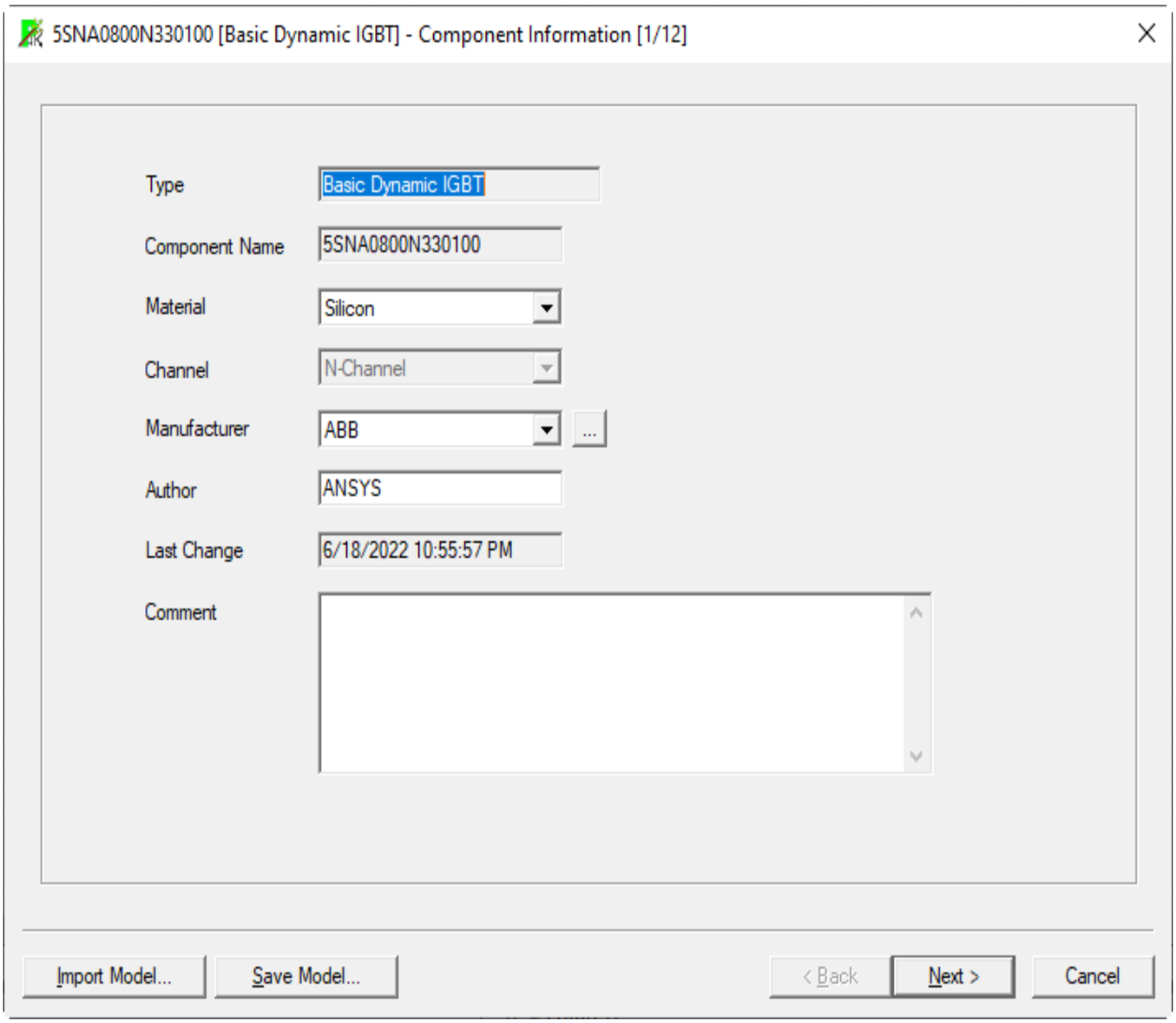
Component Information



This information will generally describe the component. The **Component Name** is used in the schematic or in the **SML** file name to identify the component for later use.

> **Note:**
>
> When naming the component, spaces are not valid characters. Using a space in the component name will result in an error. The valid characters for the component name are **a-z**, **A-Z**, **0-9**, and **_**.

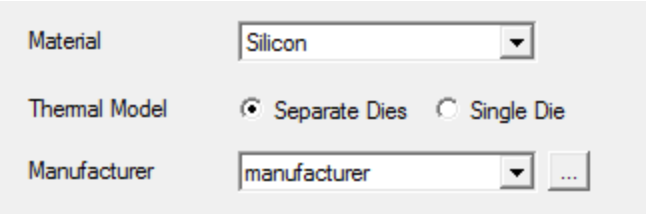
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



1. Enter the information in the fields to identify the component.
2. The selection of the material is used to account for major behavioral differences between devices made from different materials. Currently Si and SiC for wide bandgap devices are supported. The compensation for the different values Vgap is done internally through appropriate adjustments of the correction coefficients.
3. During the definition of a new device, the channel type (N-type or P-type) can be selected. For P-channel devices all values for any characteristic on following pages still must be entered as positive values. The fitting process will account for the actual sign of the value.
4. The selection of the thermal model is available for Power MOSFET models. Some MOSFETs use a common die for the FET and the freewheeling diode. This selection is used to specify the type of the die. If the device has a single die for the FET and the FWD,

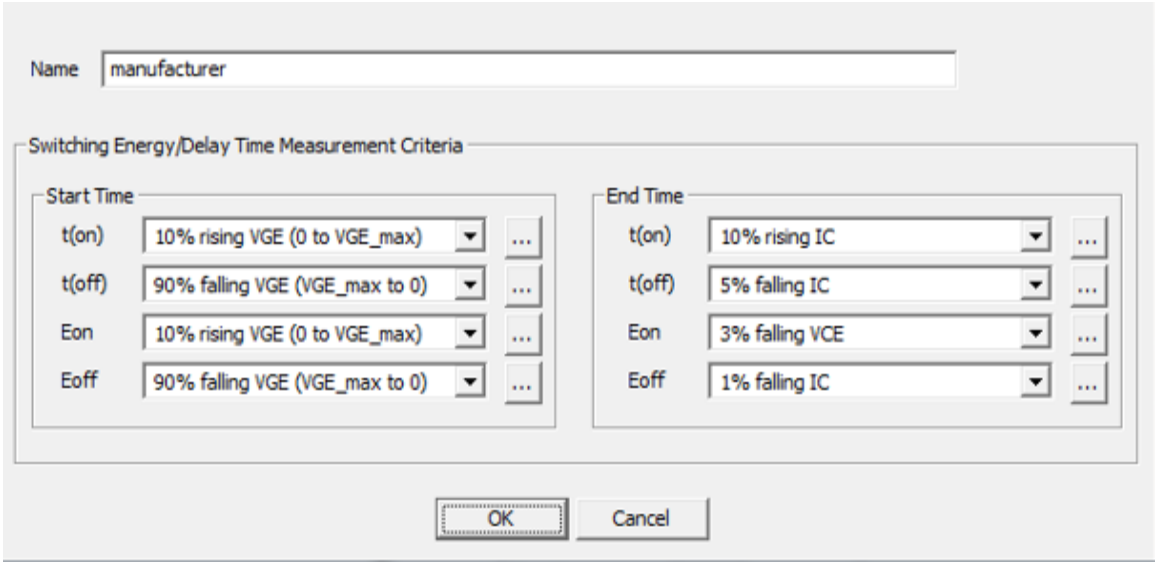
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the page for the thermal model of the FWD is not used. All parameters for the thermal behavior of the device are taken from the page for the thermal parameters of the FET.



5. To define the Switching Energy and Delay Time Measurement criteria for a Basic Dynamic or Advanced Dynamic IGBT or a Basic Dynamic Power MOSFET, click ... next to the **Manufacturer** field to open the **Manufacturer Data** dialog box in which you can set the start and stop times for **Eon**, **t(on)**, **Eoff**, and **t(off)** for the named manufacturer.

> **Note:**
>
> These criteria are not enabled for Average IGBT and Average Power MOSFET device characterization.



Use the **Manufacturer Data** dialog box to select the start and stop criteria for the switching times and energies from predefined lists. Because each manufacturer uses different criteria for these parameters, it is important to select the criteria which match the manufacturer of the device being characterized to help ensure that the configured device converges to the desired characteristics. Some choices are not yet available in the characterization tool. In

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

such cases an approximation must be made. In most cases, the off-time starts when VGE starts to drop to when IC has dropped sufficiently; the on-time starts when VGE starts to increase to when IC has sufficiently increased. Click to display a dialog box with image buttons that depict each of the possible selections. Select the image that best matches the criteria and the combo box will be updated.

Note:

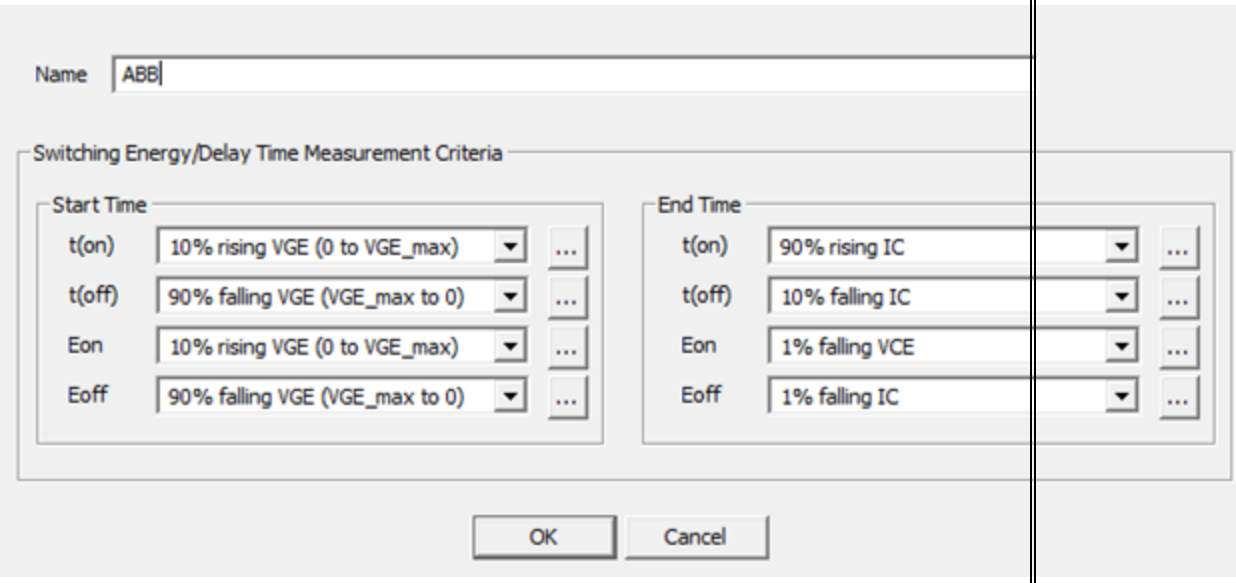
Some data sheets provide separate information for td(on/off), tr, and tf. The characterization process is not capable of fitting for td(on/off), tr, or tf separately. Therefore, it is fitting for the complete On-switch or Off-switch times, which consist of td (on) + tr = t(on) for the On-switch time and td(off) + tf = t(off) for the Off-switch time.

Note:

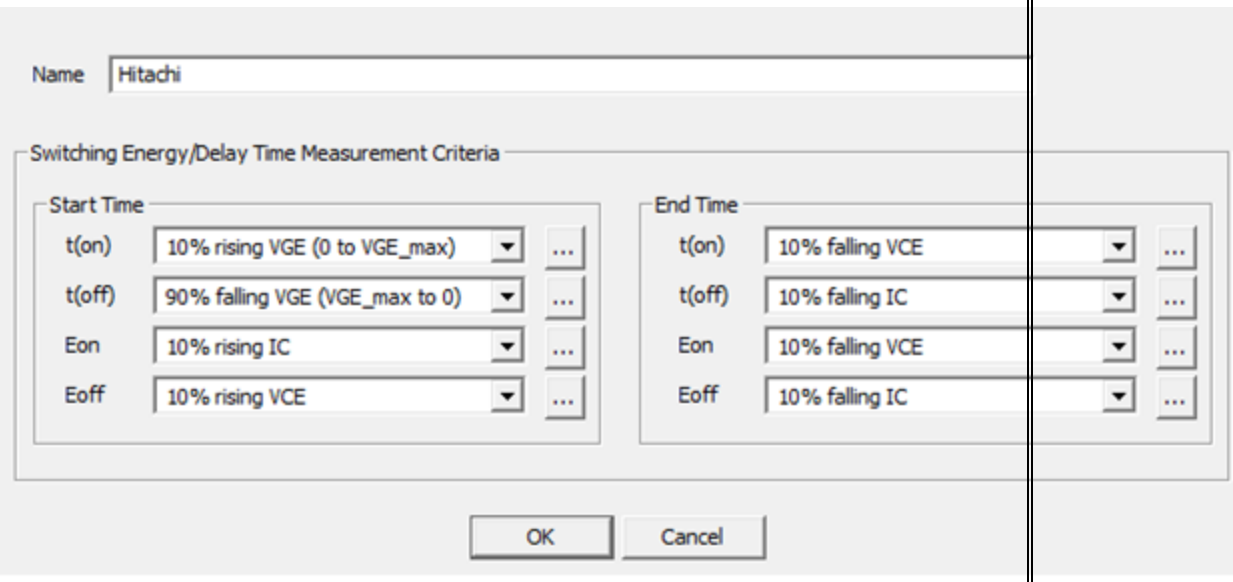
The following procedures suggest approximations for several manufacturers. If you have questions about these approximations, contact Ansys support.

Vendor	Guidelines
ABB	ABB defines its switching criteria in its *Applying IGBTs Application Note* on its website. Ansys suggests the following on-time settings: (This time is given on the datasheet as the sum of **td(on)** and **tr**.) • Set start time of **t(on)** to **10% rising VGE (0 to VGE_max)**. • Set end time of **t(on)** to **90% rising IC**. Ansys suggests the following off-time settings: • Set start time of **t(off)** to **90% falling VGE (VGE_max to 0)**. • Set end time of **t(off)** to **10% falling IC (extrapol. 90-60)**. The on-switching energy loss, **Eon**, is measured as starting at 10% after zero-crossing of rising VGE and stopping at 10 to 20 microseconds beyond this point. The off-switching energy loss, **Eoff**, is defined as starting at 90% of falling VGE and stopping at stopping at 10 to 20 microseconds beyond this point. Due to limitations of the device characterization tool, the following settings provide an *approximation* of these energy parameters: Ansys suggests the following on-switch energy settings: • Set start time of **Eon** to **10% rising VGE (0 to VGE_max)**. • Set end time of **Eon** to **1% falling VCE**.

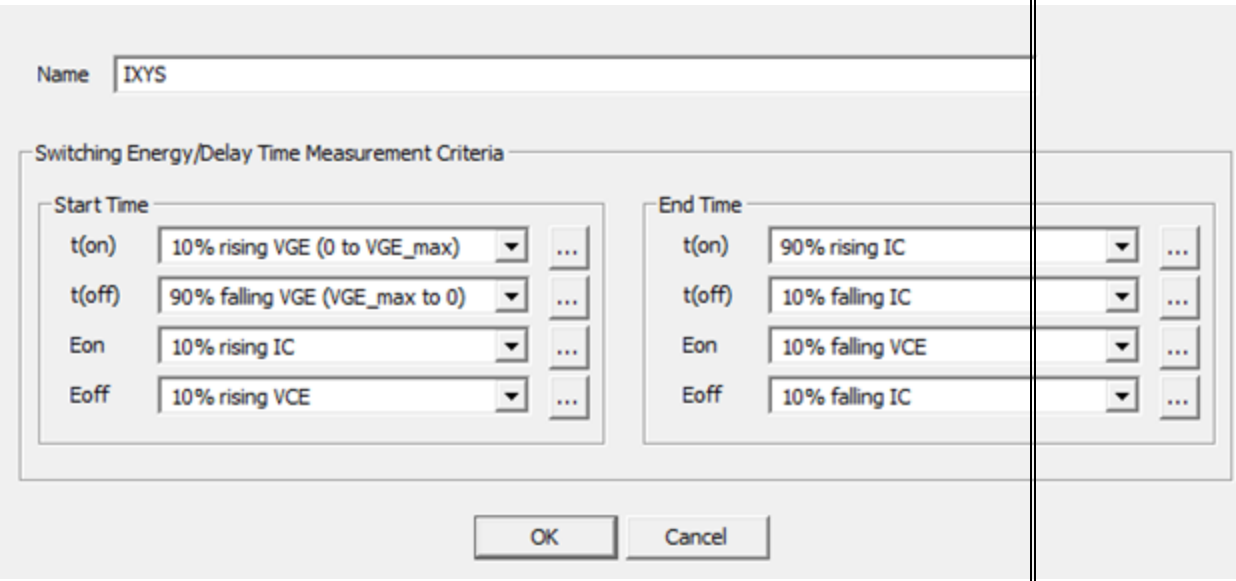
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Vendor</th><th>Guidelines</th></tr>
<tr><td></td><td>Ansys suggests the following off-switch energy settings:
Set start time of Eoff to 90% falling VGE (VGE_max to 0).
Set end time of Eoff to 1% falling IC. 
</td></tr>
<tr><td>Hitachi</td><td>Hitachi defines its switching criteria in its datasheets.
Ansys suggests the following on-time settings:
(This time is given on the datasheet as ton.)
Set start time of t(on) to 10% rising VGE (0 to VGE_max).
Set end time of t(on) to 10% falling VCE.
Ansys suggests the following off-time settings:
(This time is given on the datasheet as toff.)
Set start time of t(off) to 90% falling VGE (VGE_max to 0).
Set end time of t(off) to 10% falling IC.
Ansys suggests the following on-switch energy settings:
Set start time of Eon to 10% rising Ic.
Set end time of Eon to 10% falling VCE.
Ansys suggests the following off-switch energy settings:
Set start time of Eoff to 10% of rising VCE.
Set end time of Eoff to 10% falling IC.</td></tr>
</table>

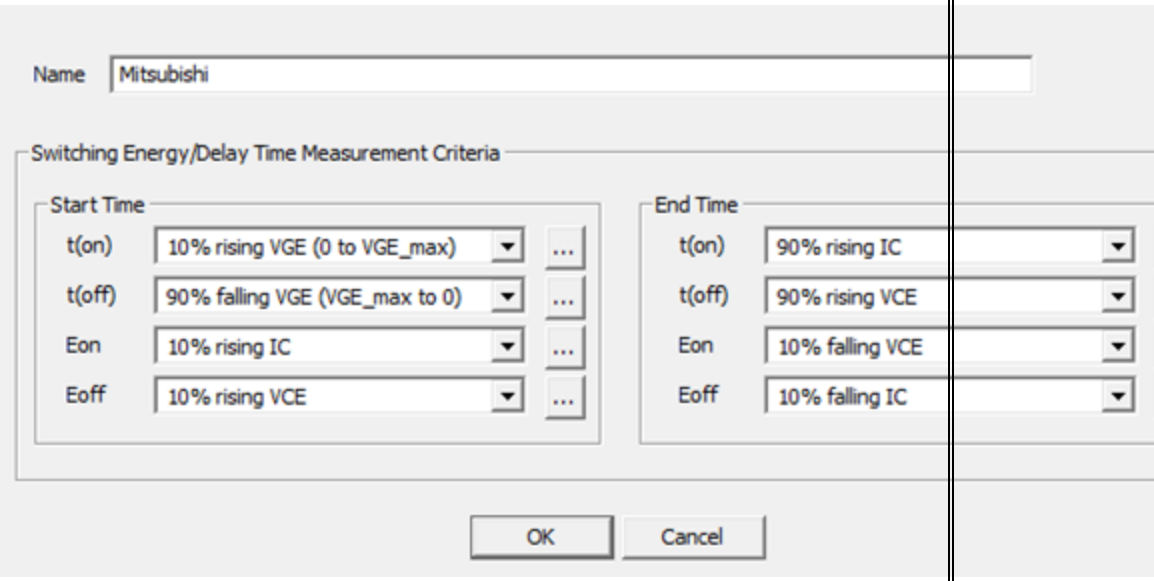
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Vendor	Guidelines
	
Infineon/Eupec	Infineon has defined its switching criteria for the switching times in the Infineon application note *AN2011-05, Figure 22*. Ansys suggests the following on-time settings: (This time is given on the datasheet as the sum of **td(on)** and **tr**.) • Set start time of **t(on)** to **10% rising VGE (0 to VGE_max)**. • Set end time of **t(on)** to **90% rising IC**. Ansys suggests the following off-time settings: (This time is given on the datasheet as **td(off) +tf**.) • Set start time of **t(off)** to **90% falling VGE (VGE_max to 0)**. • Set start time of **t(off)** to **10% falling IC**. Ansys suggests the following on-switch energy settings: • Set start time of **Eon** to **10% rising IC**. • Set end time of **Eon** to **3% falling VCE**. Ansys suggests the following off-switch energy settings: • Set start time of **Eoff** to **10% rising VCE**. • Set end time of **Eoff** to **1% falling IC**.

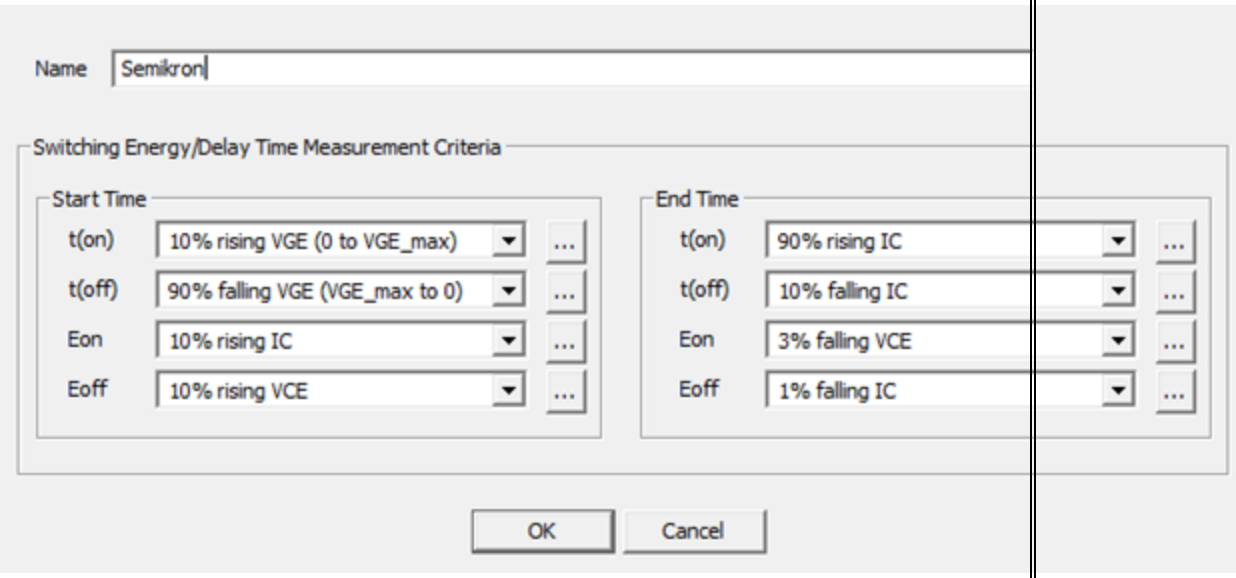
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Vendor	Guidelines
	Note: Older application notes (for example, AN2007-4, Figure 43) described the measurement for the On-switch time and Off-switch time with a time gap between the end of the delay time **td(on)** or **td(off)** and the beginning of the rise time **tr** or the fall time **tf**. This older measurement definition is no longer supported.
IXYS	IXYS has defined its switching criteria for the switching times in its IGBT application note *IXAN0063*. Ansys suggests the following on-time settings: (This time is given on the datasheet as the sum of **td(on)** and **tr**.) • Set start time of **t(on)** to **10% rising VGE (VGE_min to VGE_max)**. • Set end time of **t(on)** to **90% rising IC**. Ansys suggests the following off-time settings: (This time is given on the datasheet as **td(off)** and **tf**.) • Set start time of **t(off)** to **90% falling VGE (VGE_max to 0)**. • Set start time of **t(off)** to **10% falling IC**. Ansys suggests the following on-switch energy settings: • Set start time of **Eon** to **10% rising Ic**. • Set end time of **Eon** to **10% falling VCE**. Ansys suggests the following off-switch energy settings: • Set start time of **Eoff** to **10% rising VCE**. • Set end time of **Eoff** to **10% falling IC**. 

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Vendor	Guidelines
Mitsubishi	Mitsubishi uses varying definitions for its switching criteria. The datasheets usually supply the definitions that were used. Regrettably, many datasheets do not supply the values of **Eon** and **Eoff**, so it will not be possible to fit for these characteristics. Consequently, you need not fit for these values in the Dynamic Model Input section. Ansys suggests the following on-time settings: (This time is given on the datasheet as the sum of **td(on)** and **tr**.) • Set start time of **t(on)** to **10% rising VGE (0 to VGE_max)**. • Set end time of **t(on)** to **90% rising IC**. Ansys suggests the following off-time settings: (This time is given on the datasheet as **td(off)** and **tf**.) • Set start time of **t(off)** to **90% falling VGE (VGE_max to 0)**. • Set end time of **t(off)** to **90% rising VCE**. Ansys suggests the following on-switch energy settings: • Set start time of **Eon** to **10% rising Ic**. • Set end time of **Eon** to **10% falling VCE**. Ansys suggests the following off-switch energy settings: • Set start time of **Eoff** to **10% rising VCE**. • Set end time of **Eoff** to **10% falling Ic** or **1% falling Ic** (check datasheet). 
Semikron	Semikron defines its switching criteria in its *Technical Applications* datasheet on its website. Ansys suggests the following on-time settings:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

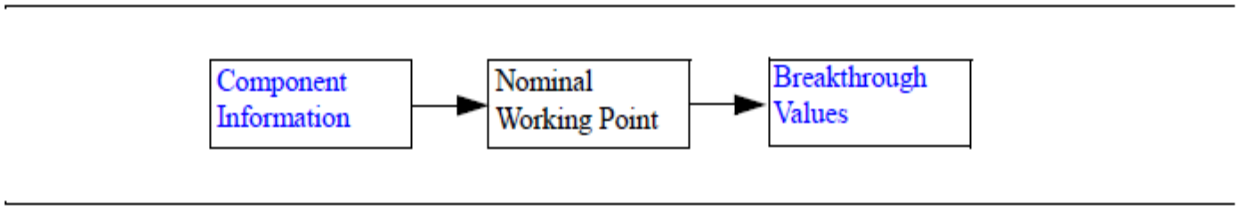
Vendor	Guidelines
	(This time is given on the datasheet as the sum of **td(on)** and **tr**.) • Set start time of **t(on)** to **10% rising VGE (0 to VGE_max)**. • Set end time of **t(on)** to **90% rising IC**. Ansys suggests the following off-time settings: (This time is given on the datasheet as **td(off)** and **tf**.) • Set start time of **t(off)** to **90% falling VGE (VGE_max to 0)**. • Set end time of **t(off)** to **10% falling IC**. Ansys suggests the following on-switch energy settings: • Set start time of **Eon** to **10% rising VGE (0 to VGE_max)**. • Set end time of **Eon** to **3% falling VCE**. Ansys suggests the following off-switch energy settings: • Set start time of **Eoff** to **90% falling VGE (VGE_max to 0)**. • Set end time of **Eoff** to **1% falling Ic**. 
Fuji	Fuji has defined its switching criteria for the switching times in its *IGBT Application notes (REH983)*. Many datasheets do not supply the values of **Eon** and **Eoff**, so it will not be possible to fit for these characteristics. You then need to not fit for these values in the **Dynamic Model Input** section. The on-time is defined as the time from the input signal rising above the threshold voltage to **90% of rising Ic**. The starting time is not yet available in the wizard. Ansys suggests starting at **10% of rising VGE** and stopping at **90% of Ic**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Vendor	Guidelines
	This time is found in the data-sheet as **toff**. The off-time is defined as starting when the input signal drops below the threshold voltage and stops at **10% of falling Ic**. The starting time is not yet available in the wizard. Ansys suggests starting at **90% of VGE** and stopping at **10% of Ic**. This time can be found in the data-sheet as **toff**. Fuji regularly does not supply the switching losses **Eon** and **Eoff**. Nor do there appear to be definitions for the losses in the application notes. A probable good approximation is to start **Eon** at **10% of rising VGE** and stop at **1% of falling VCE**. A probable good approximation is to start **Eon** at **90% of falling VGE** and stopping at **1% of falling Ic**.

3. Click **Next** to continue characterizing the device (See ***Nominal Working Point Values)***.

Nominal Working Point Values



The nominal working point of the device is the point commonly used when testing a device for switching parameters such as switching speed, rise time, and delay. Ideally, this point should be the condition in which you are planning to operate the IGBT, but it is more important to have all the data necessary for the characterization available at these conditions. Look to the area of the manufacturer's specification regarding the test conditions for the switching times and losses to locate the **Nominal Working Point** conditions.

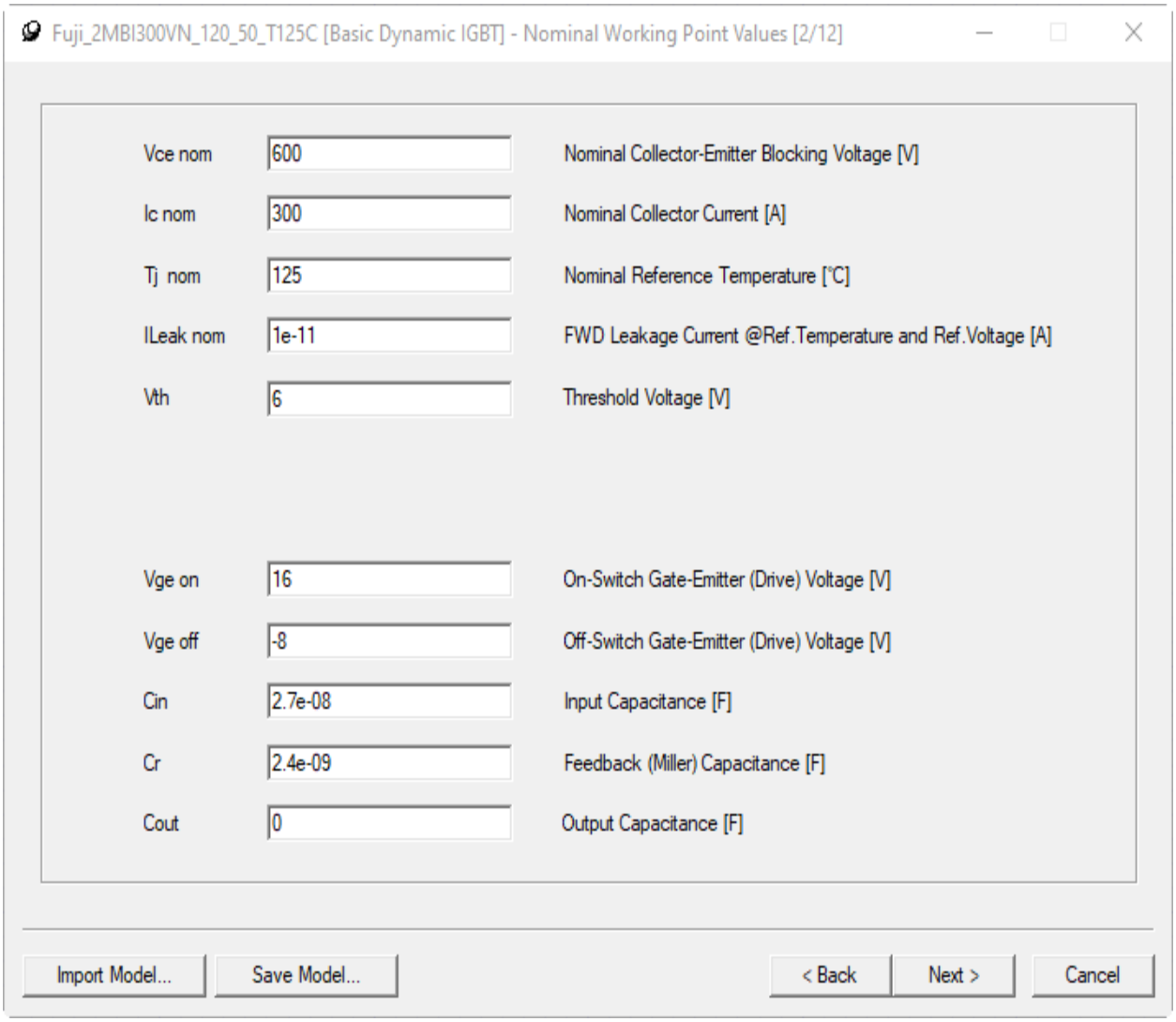
Enter the information required to define the **Nominal working point** for the device:

Nominal Working Point Values for IGBTs

Nominal Working Point Values for Power MOSFETs

Nominal Working Point Values for IGBTs

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Nominal values are the basis of working point dependency.

Note:

See *Vendor-Specific Guidelines for Nominal Working Point Values* below when making the following settings.

Nominal Collector Emitter Blocking Voltage [V]	This is listed as $\mathbf{V_{cc}}$ in the test conditions area of the switching characteristics section of the datasheet.
Nominal Collector	Listed as $\mathbf{I_c}$ in the switching characteristics test conditions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Current [A]	
Nominal Reference Temperature [^{o}C]	This is generally listed as $\mathbf{T_c}$ for the maximum temperature test conditions for the switching characteristics, for example 125^{o}C.
FWD leakage current under Nominal Conditions [A]	This value should remain at the default value unless specified in the manufacturer's specification. This parameter directly set the FWD leakage current in the extracted device model. It has little influence to static or dynamic fitting of the model.
Threshold Voltage [V]	Listed in table of characteristics.
Collector-Emitter Saturation Voltage under Nominal Condition [V]	Listed as $\mathbf{V_{ce\ sat}}$ in the switching characteristic test conditions. This field is available only for the Average IGBT characterization.
On-Switch Gate-Emitter (Drive) Voltage [V]	Listed as $\mathbf{V_{ge}}$ in the switching characteristics test conditions.
Off-Switch Gate-Emitter (Drive) Voltage [V]	This parameter may be listed in the switching characteristics test conditions; however, it often is not. A rule of thumb is to use the negative of the $\mathbf{V_{ge}}$(On) value listed above.
Input capacitance [F]	Listed in table of characteristics.
Feedback capacitance [F]	Also known as reverse transfer capacitance or Miller capacitance. Listed in table of characteristics.
Output Capacitance [F]	Listed in table of characteristics.

Vendor-Specific Guidelines for Nominal Working Point Values

See these vendor-specific guidelines when setting nominal working point values:

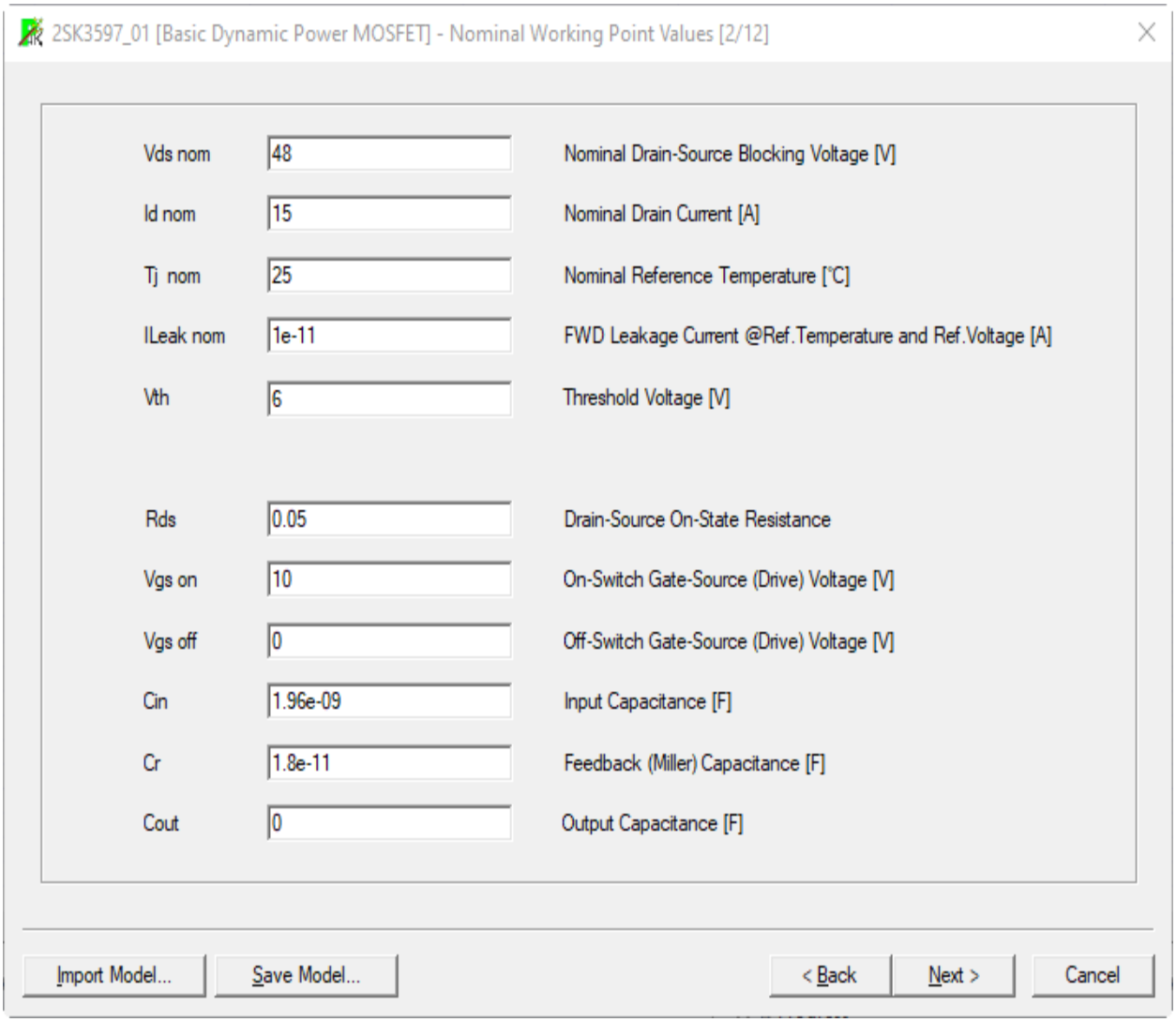
Vendor	Guidelines
ABB	Most information will be available at 125^{o} C in a datasheet. Review the conditions for which the switching times and energies are given and choose the one in which you want to operate.
Hitachi	Most information will be available at 125^{o} C in a datasheet. Review the conditions for which the switching times and energies are given and choose the one in which you want to operate.
Infineon	Most information will be available at 125^{o} C or 150^{o} C in a datasheet. Review

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Vendor	Guidelines
	the conditions for which the switching times and energies are given and choose the one in which you want to operate.
IXYS	Most information will be available at 125^{o} C in a datasheet. Review the conditions for which the switching times and energies are given and choose the one in which you want to operate.
Mitsubishi	Most information will be available at 25^{o} C in a datasheet. Review the conditions for which the switching times and energies are given and choose the one in which you want to operate. Though the information can be limited, it is best to pick the point where the most information is available.
Semikron	Most information will be available at 125^{o} C in a datasheet. It is good to look at the conditions in which the switching times and energies are given and pick the one in which you want to operate.

Nominal Working Point Values for Power MOSFETs

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



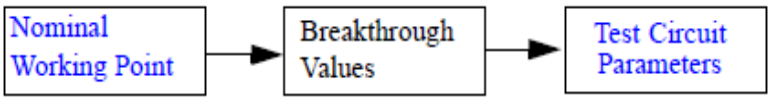
Nominal Drain-Source Blocking Voltage [V]	Listed as $\mathbf{V_{DD}}$ in the test conditions area of the switching characteristics section of the datasheet.
Nominal Drain Current [A]	Listed as $\mathbf{I_D}$ in the test conditions area of the switching characteristics section of the datasheet.
Nominal Reference Temperature [^{O}C]	This will generally be listed as $\mathbf{T_c}$ for the maximum temperature test conditions for the switching characteristics.
FWD leakage	This value should remain at the default value unless specified in the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

current under Nominal Conditions [A]	manufacturer's specification. This parameter directly set the FWD leakage current in the extracted device model. It has little influence to static or dynamic fitting of the model.
Threshold Voltage [V]	Listed in table of characteristics.
Drain-Source On-state Resistance [Ohm]	Listed in datasheet as $R_{DS(on)}$ in table of characteristics
On-Switch Gate-Source (Drive) Voltage [V]	Listed as V_{ge} in the switching characteristics test conditions.
Off-Switch Gate- Source (Drive) Voltage [V]	This parameter may be listed in the switching characteristics test conditions; however, it often is not. A rule of thumb is to use the negative of the V_{ge}(On) value above.
Input Capacitance [F]	Listed in table of characteristics.
Feedback Capacitance [F]	Also known as reverse transfer capacitance or Miller capacitance. Listed in table of characteristics.
Output Capacitance [F]	Listed in table of characteristics.

When finished, click **Next** to continue characterizing the device (See ***Breakthrough Values***).

Breakthrough Values

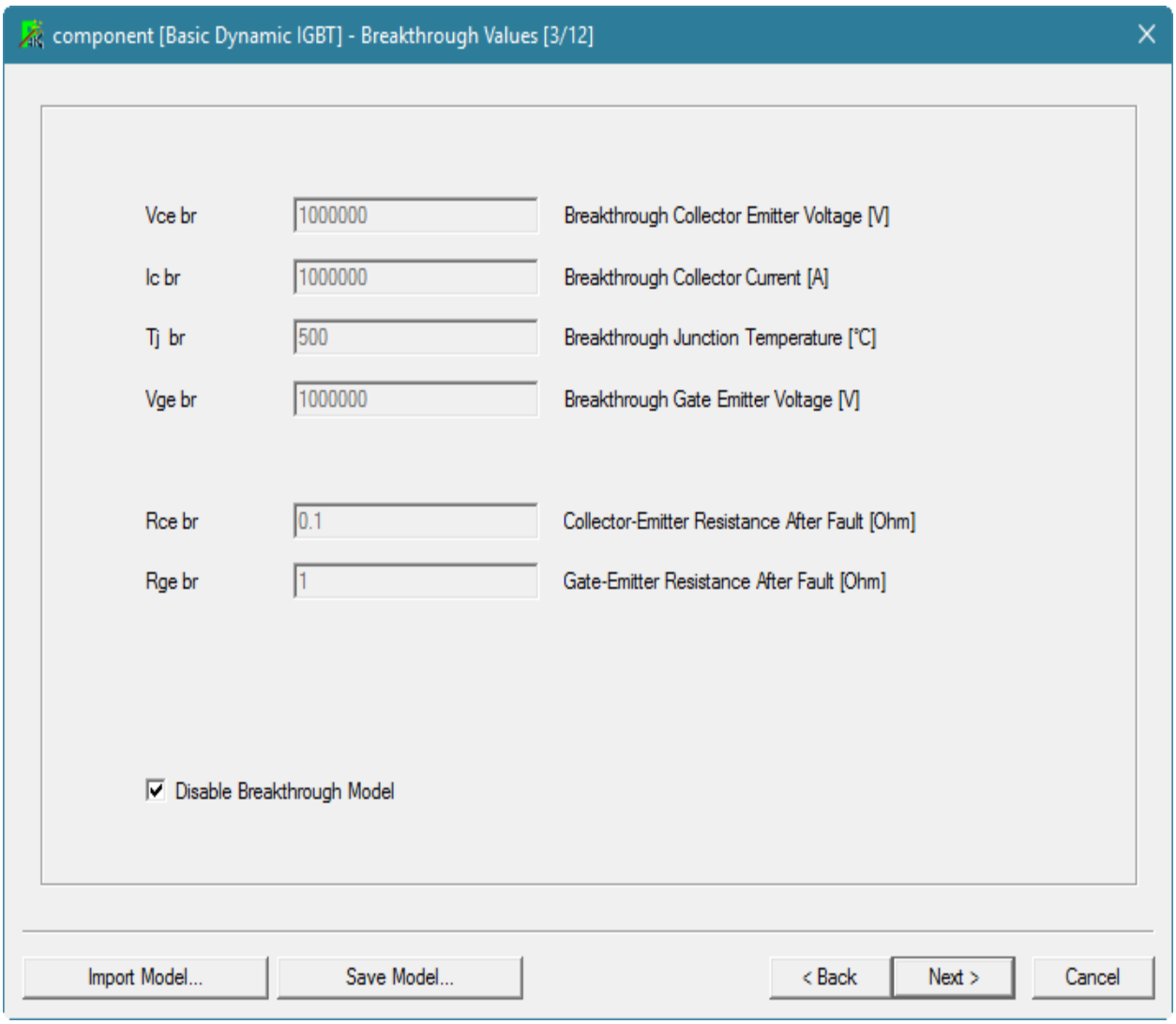


In the **Breakthrough Values** dialog box, you can enter parameters related to the device breakthrough or breakdown performance. The data needed to characterize breakthrough values is missing in most datasheets. In such cases, we recommend disabling this feature. In order to ignore the breakdown of the device, select the **Disable Breakthrough Model** check box. This

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

sets the parameters to very high values, thus preventing the model from simulating the breakthrough effects of the device.

To simulate the breakthrough effects, enter the parameters listed in the following table.



Breakthrough collector-emitter voltage (V)	This value should be greater than the maximum collector-emitter voltage (V), that is, $\mathbf{V_{ce\ br} > V_{ce\ max}}$. A good rule of thumb is to use twice the value for $\mathbf{V_{ce\ max}}$. In order to determine this value, see the **Safe Operating Area** diagram provided for the device. See Point A on the sample diagram below.
Breakthrough collector current (A)	This value should be greater than the maximum collector current, that is, $\mathbf{I_{c\ br} > I_{c\ max}}$. A good rule of thumb is to use twice the value for $\mathbf{I_{c\ max}}$. In order to determine this value, see the **Safe Operating Area** diagram provided for

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	the device. See Point B on the sample diagram below.
Breakthrough junction temperature (°C)	This value should be greater than the maximum junction temperature, that is, $\mathbf{T_{j\ br} > T_{j\ max}}$. The operating range for the Junction Temperature is generally specified in the **Absolute Maximum Ratings** section of the datasheet.
Breakthrough gate-emitter voltage (V)	This value should be set greater than the value for $\mathbf{V_{ge}}$ (On) specified in the **Nominal Working Point** dialog box.
Collector-emitter resistance after fault (Ohm)	This value represents the collector-emitter resistance of the device after a fault has occurred. Use the default value unless the device vendor can supply a specific value.
Gate-emitter resistance after fault (Ohm)	This value represents the gate-emitter resistance of the device after a fault has occurred. Use the default value unless the device vendor can supply a specific value.

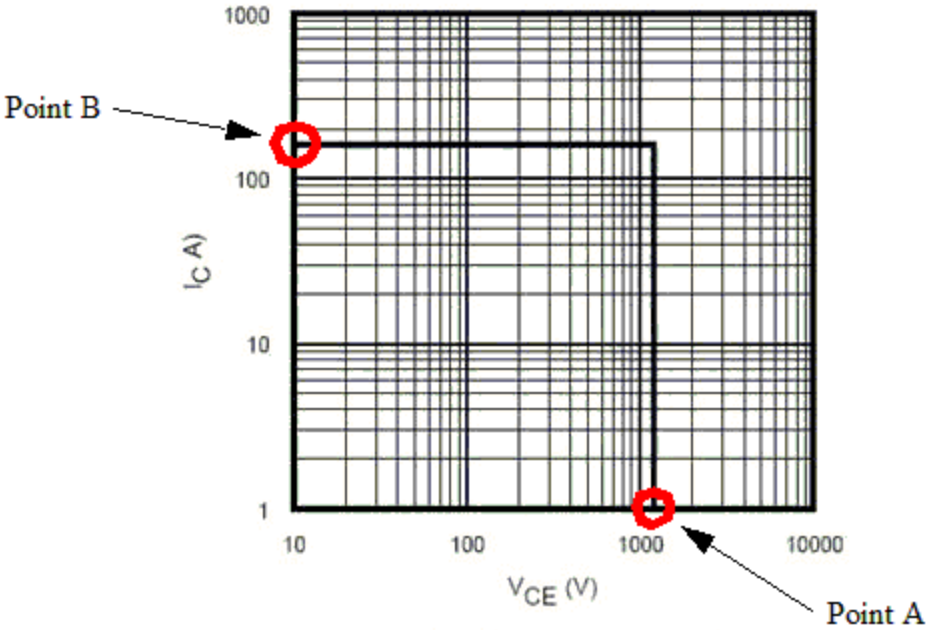
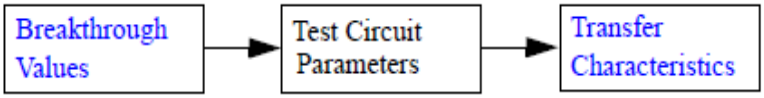


Fig. 4 - Reverse Bias SOA
T_J = 150°C; V_{GE} =15V

When finished, click **Next** to continue characterizing the device (Half-Bridge Test Circuit).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Test Circuit Conditions

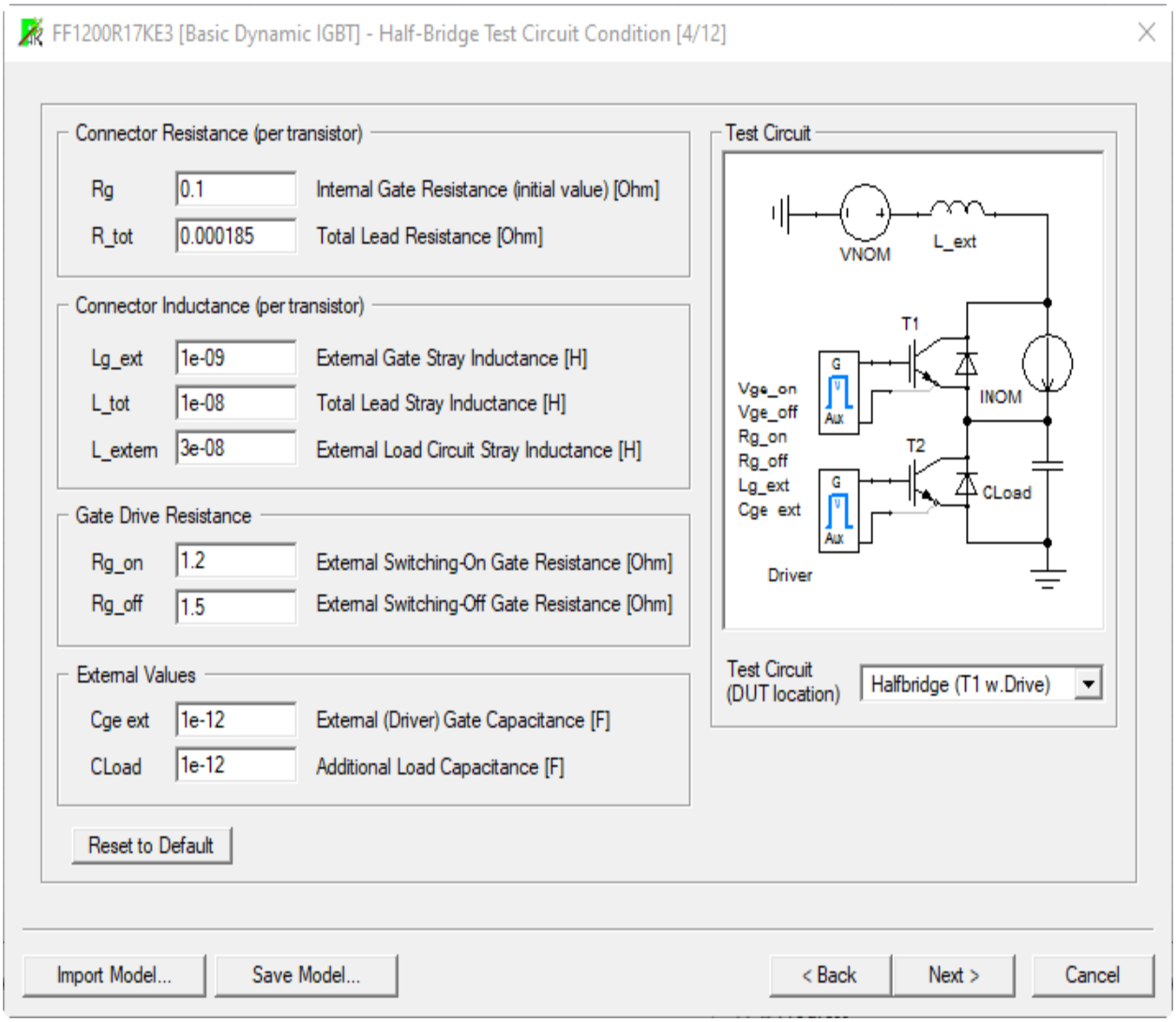


Choose the **Test Circuit Conditions** parameter settings for the IGBT type being characterized:

- Basic Dynamic or Advanced Dynamic IGBT Half-Bridge Test Circuit Conditions
- Average IGBT Half-Bridge Test Circuit Conditions
- Power MOSFET Test Circuit Conditions

Half-Bridge Test Circuit Condition for a Basic Dynamic or Advanced Dynamic IGBT

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

See ***Vendor-Specific Guidelines for Basic Dynamic IGBT Half-Bridge Test Circuit Condition*** below when making the following settings.

Internal Gate Resistance Rg (Ohm)	Listed in table of characteristics in datasheet. Default value is **0.2**.
Total Lead Resistance	**R_tot** is the lead resistance *per switch* of the IGBTs. This value is used to calculate RE and RC in the parameterized device generated by the device

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

R_tot (Ohm)	characterization wizard. If the datasheet provides the resistance for a half-bridge module, **R_tot** will be half of the value given on the datasheet. Unless specified by the manufacturer, this value should remain at the default value of **0**. Contact the device manufacturer to get a measured value for this parameter if desired. **NOTE: This value must be less than $0.250/I_{nom}$.**
External Gate Stray Inductance Lg_ext (H)	This value should remain at the default value unless specified in the manufacturer's specification. Default value is **0**.
Total Lead Stray Inductance L_tot (H)	**L_tot** is the lead stray inductance *per switch*. This is used to calculate LE, LAUX and LC in the parameterized device generated by the device characterization wizard. If the datasheet provides the stray inductance for a half-bridge module, **L_tot** will be half of the value given on the datasheet. The manufacture may provide a value for **L_tot** in the test conditions section of the switching characteristics parameters. If no value is supplied, leave the default value to prevent numerical instability. Default value is **0**. **Note**: Some manufacturers list a value for **L** which is generally the Load Inductance used for the test and should not be interpreted as **L_tot**.
Sum of External Load Path Stray Inductance L_extern (H)	**L_extern** is the sum of the external load path stray inductance. This inductance is part of the half bridge circuit, which is used to determine the switching energies and times. This value will be fit for by the dynamic characterization tool, but it is not a part of the parameterized device generated by the parameterization wizard. The value entered here is used as a starting point for the fit. Unless specified in the manufacturer's specification, the inductance value should remain at the default settings. This parameter may be specified when multiple devices are packaged in a module. Default value is **2e-009**.
Switching-On Gate Resistance Rg_on (Ohm)	This parameter will generally be listed in the test conditions section of the on-switching characteristics as $\mathbf{R_g}$. Default value is **0.001**.
Switching-Off Gate Resistance Rg_off (Ohm)	This parameter is usually listed in the test conditions section of the off-switching characteristics as **Rg**. Unless otherwise specified in the manufacturer's specification, set this parameter to the same value used for **Rg_on**. Default value is **0.001**.
External Gate Capacity Cge ext (F)	This value should remain at the default value unless specified in the manufacturer's specification. Default value is **0**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Select the **Use default values** check box to apply the default values to the device being characterized.

When finished, click **Next** to continue characterizing the device. See (***Transfer Characteristics***).

Vendor-Specific Guidelines for Basic Dynamic IGBT Half-Bridge Test Circuit Condition

Refer to the follow vendor-specific guidelines when setting values:

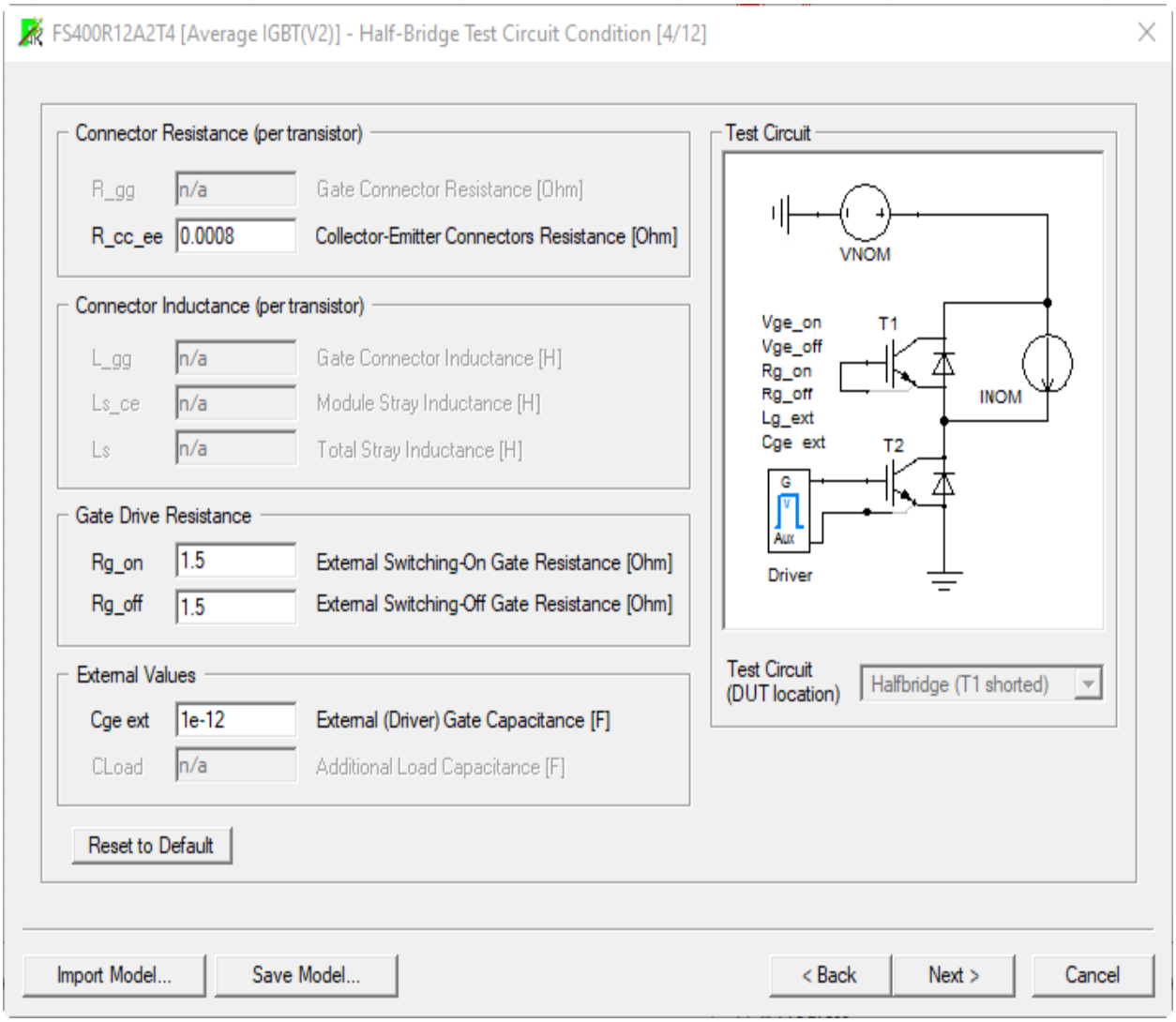
Vendor	Guidelines
ABB	**R_tot** is given by **R_cc_ee** or half of **R_cc_ee** in the datasheet, depending if the value is per switch or for the whole module. **L_tot** is given by half of **Lsce** in the datasheet, depending if the value is per switch or for the whole module. **L_extern** can be listed as **Ls** in the measurement conditions of **Eon** and **Eoff**. **Rg_on** and **Rg_off** are usually listed as **RG** in the measurement conditions of the switching times and energies. **Cge_ext** is usually not listed. If not listed, **1pF** is a good approximation.
Hitachi	**R_tot** is usually not listed in the datasheet, using **0** is a good approximation, as the DC characterization will adapt. **L_tot** is given by **Lsce** or half of **Lsce** in the datasheet, depending if the value is per switch or for the whole module. **L_extern** can be listed as **L** in the measurement conditions of **Eon** and **Eoff**. **Rg_on** and **Rg_off** are usually listed as **RG** in the measurement conditions of the switching times and energies. **Cge_ext** is usually listed as **Cge** in the measurement conditions of the switching times and energies. If not listed, **1pF** is a good approximation.
Infineon	**R_tot** is given by **R_cc_ee** or half of **R_cc_ee** in the datasheet, depending if the value is per switch or for the whole module. **L_tot** is given by **Lsce** or half of **Lsce** in the datasheet, depending if the value is per switch or for the whole module. **L_extern** can be reported as **Ls** in the measurement conditions of **Eon** and **Eoff**. **Rg_on** and **Rg_off** are usually reported as **RG** in the measurement conditions of the switching times and energies. **Cge_ext** is usually not listed by Infineon. If not listed, **1 pF** is a good approximation.
IXYS	**R_tot** is generally not supplied; a default value of **0** is then appropriate.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Vendor	Guidelines
	L_tot is generally not supplied; a default value of **0 H** is then appropriate. **L_extern** is generally not supplied; a default value of **30 nH** is then appropriate. **Rg_on** and **Rg_off** are usually given as **RG** in the measurement conditions of the switching times and energies. **Cge_ext** is usually not listed. If not listed, **1pF** is a good approximation.
Mitsubishi	**R_tot** is given by **RIead** or half of **RIead** in the datasheet, depending if the value is per switch or for the whole module. **L_tot** is generally not supplied; a default value of **0 H** is then appropriate. **L_extern** is generally not supplied; a default value of **30 nH** is then appropriate. **Rg_on** and **Rg_off** are usually listed as **RG** in the measurement conditions of the switching times and energies. **Cge_ext** is usually not listed. If not listed, **1pF** is a good approximation.
Semikron	**R_tot** is given by **R_cc_ee** or half of **R_cc_ee** in the datasheet, depending if the value is per switch or for the whole module. **L_tot** is given by **Lsce** or half of **Lsce** in the datasheet, depending if the value is per switch or for the whole module. **L_extern** can be listed as **Ls** in the measurement conditions of **Eon** and **Eoff**. **Rg_on** and **Rg_off** are usually listed as **RG** in the measurement conditions of the switching times and energies. **Cge_ext** is usually not listed. If not listed, **1pF** is a good approximation.

Half-Bridge Test Circuit Condition for an Average IGBT

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



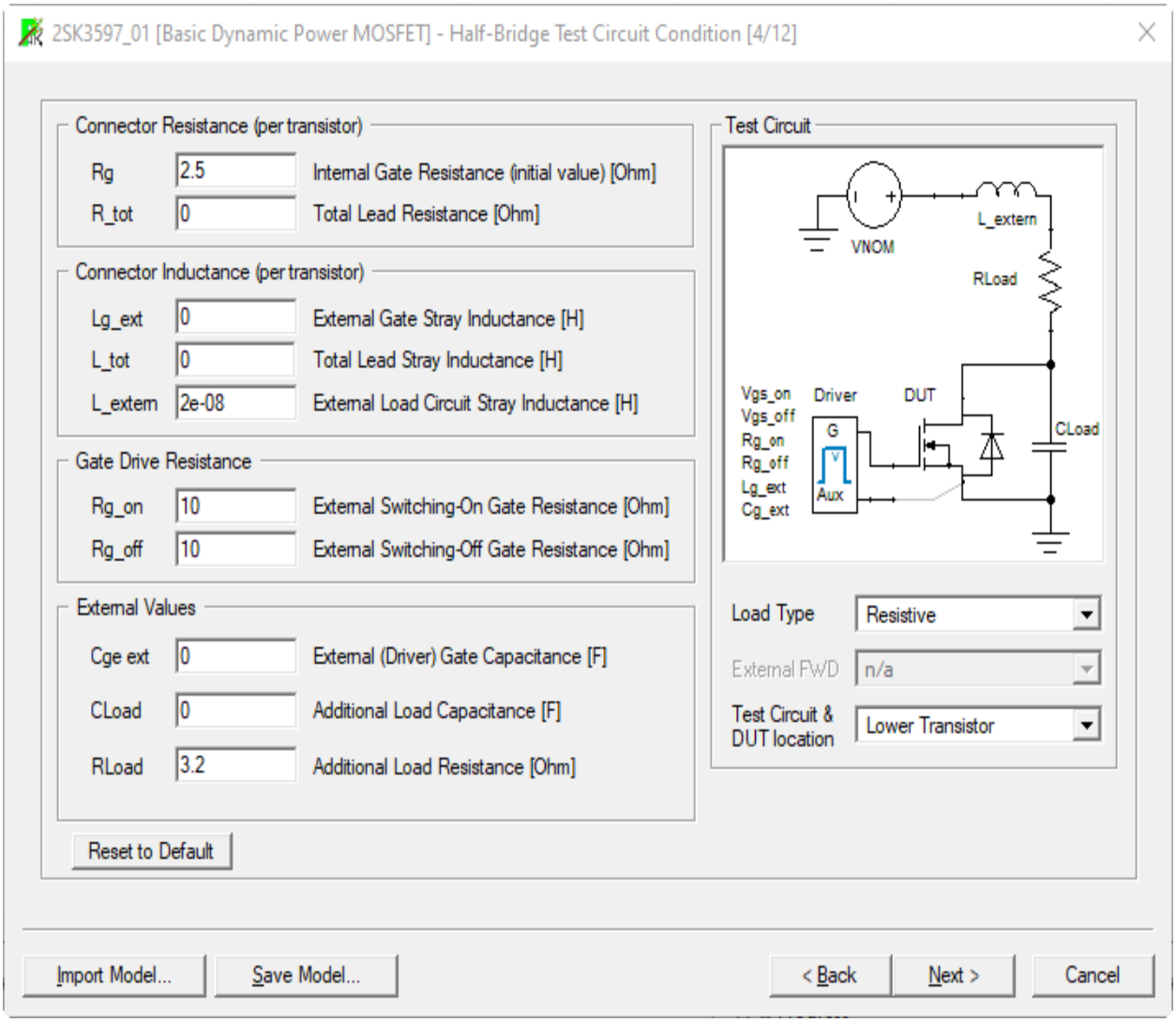
Gate Connector Resistance R_gg (Ohm)	Not applicable.
Collector-Emitter Connectors Resistance R_cc_ee (Ohm)	Unless specified by the manufacturer, this value should remain at the default value of **0**. Contact the device manufacturer to get a measured value for this parameter if desired. **Note: This value must be less than 0.250/I_{nom}.**
Gate Connector Inductance L_gg (H)	Not applicable.
Total Stray	Not applicable.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Inductance Ls (H)	
Module Stray Inductance Ls_ce (H)	Not applicable.
Switching-On Gate Resistance Rg_on (Ohm)	This parameter will generally be listed in the test conditions section of the switching characteristics as $\mathbf{R_g}$. Default value is **1**.
Switching-Off Gate Resistance Rg_off (Ohm)	Unless otherwise specified in the manufacturer's specification, set this parameter to the same value used for **Rg_on**. Default value is **1**.
External Gate Capacity Cge_ext (F)	This value should remain at the default value of **0** unless specified in the manufacturer's specification.

Test Circuit Condition for a Power MOSFET

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Internal Gate Resistance Rg (Ohm)	Listed in table of characteristics in datasheet. Default value is **0.2**.
Total Lead Resistance R_tot (Ohm)	**R_tot** is the lead resistance per switch of the MOSFET. This value is used to calculate **RE** and **RC** in the parameterized device generated by the device characterization wizard. If the datasheet provides the resistance for a half-bridge module, **R_tot** will be half of the value given on the datasheet.
External Gate Stray Inductance Lg_ext (H)	This value should remain at the default value unless specified in the manufacturer's specification. Default value is **0**.
Total Lead Stray	**L_tot** is the lead stray inductance per switch. This is used to calculate

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Inductance L_tot (H)	LE, LAUX and LC in the parameterized device generated by the device characterization wizard. If the datasheet provides the stray inductance for a half-bridge module, **L_ tot** will be half of the value given on the datasheet. The manufacturer may provide a value for **L_tot** in the test conditions section of the switching characteristics parameters. If no value is supplied, leave the default value to prevent numerical instability. Default value is **0**. **Note:** Some manufacturers list a value for **L** which is generally the Load Inductance used for the test and should not be interpreted as **L_ tot**.
Sum of External Load Path Stray Inductance L_extern (H)	**L_extern** is the sum of the external load path stray inductance. This inductance is part of the half bridge circuit, which is used to determine the switching energies and times. This value will be fit for by the dynamic characterization tool, but it is not a part of the parameterized device generated by the parameterization wizard. The value entered here is used as a starting point for the fit. Unless specified in the manufacturer's specification, the inductance value should remain at the default settings. This parameter may be specified when multiple devices are packaged in a module. Default value is **2e-009**.
Switching- On Gate Resistance Rg_on (Ohm)	This parameter will generally be listed in the test conditions section of the on- switching characteristics as **Rg**. Default value is **0.001**.
Switching- Off Gate Resistance	This parameter is usually listed in the test conditions section of the off-switching characteristics as **Rg**.
External Gate Capacity Cge ext (F)	This value should remain at the default value unless specified in the manufacturer's specification. Default value is **0**.

MOSFET devices are usually tested in a Half-Bridge circuit where the DUT is in the lower location and the upper device is replaced by a diode or a resistor. However, you can choose other configurations, like the DUT in the upper location or a real Half-Bridge circuit. Depending on the configuration some of the other settings are available or not. The test circuit picture adjusts according to the settings of the test circuit type, load type, and external FWD type and shows a simplified picture of the test circuit.

Test circuit & DUT location	Available settings are Half-Bridge, Upper Transistor, and Lower Transistor.
Load Type	Can be set when the type of test circuit is not set to Half-Bridge. Available settings are resistive or inductive.
External FWD	Type of the external freewheeling diode when the type of the test circuit is not set to Half-Bridge and the load type is inductive. The FWD can be static, dynamic, or

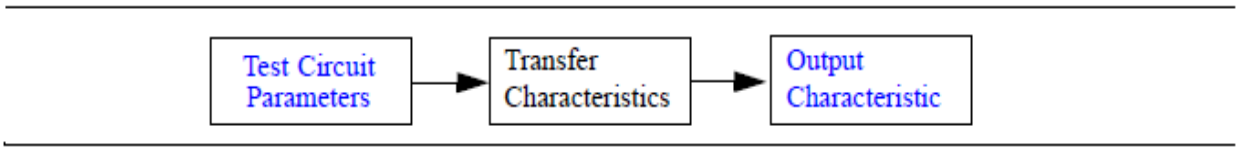
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	dynamic with reverse recovery.
RLoad	Value of the load resistance if the load type is resistive.

Select the **Use default values** check box to apply the default values to the device being characterized.

When finished, click **Next** to continue characterizing the device. See ***Transfer Characteristics***.

Transfer Characteristics



In the **Transfer Characteristics** dialog box, each tab corresponds to one transfer characteristic curve, that is, **Ic=f(Vge)** at a specific temperature. These characteristics are obtained from the typical transfer characteristic curves supplied by the manufacturer as shown in the following example.

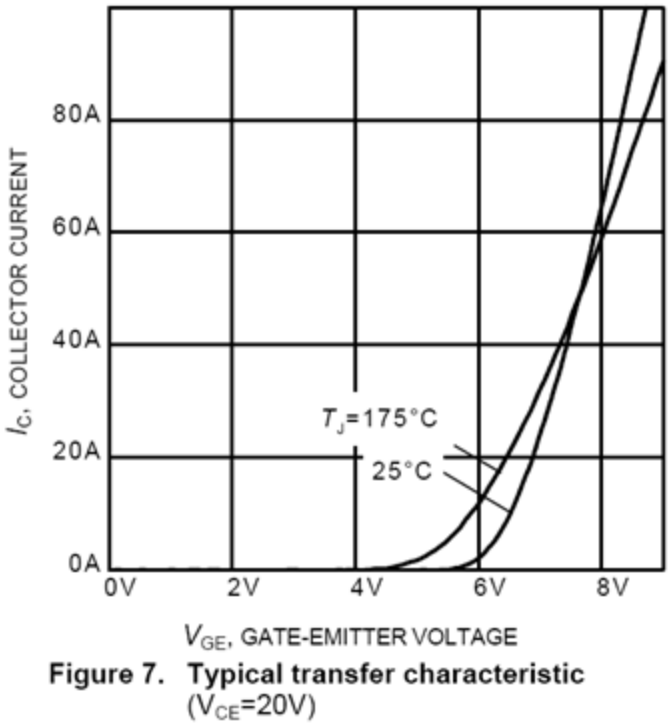


Figure 7. Typical transfer characteristic (V_{CE}=20V)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If the transfer characteristics are not supplied in the datasheet, you can usually extract enough points from the output characteristics to get a good fit. To do so, one takes the current Ic at the highest possible VCE (when the current is saturated, that is, as it ceases to increase as VCE is increased) for at least three different values of VGE.

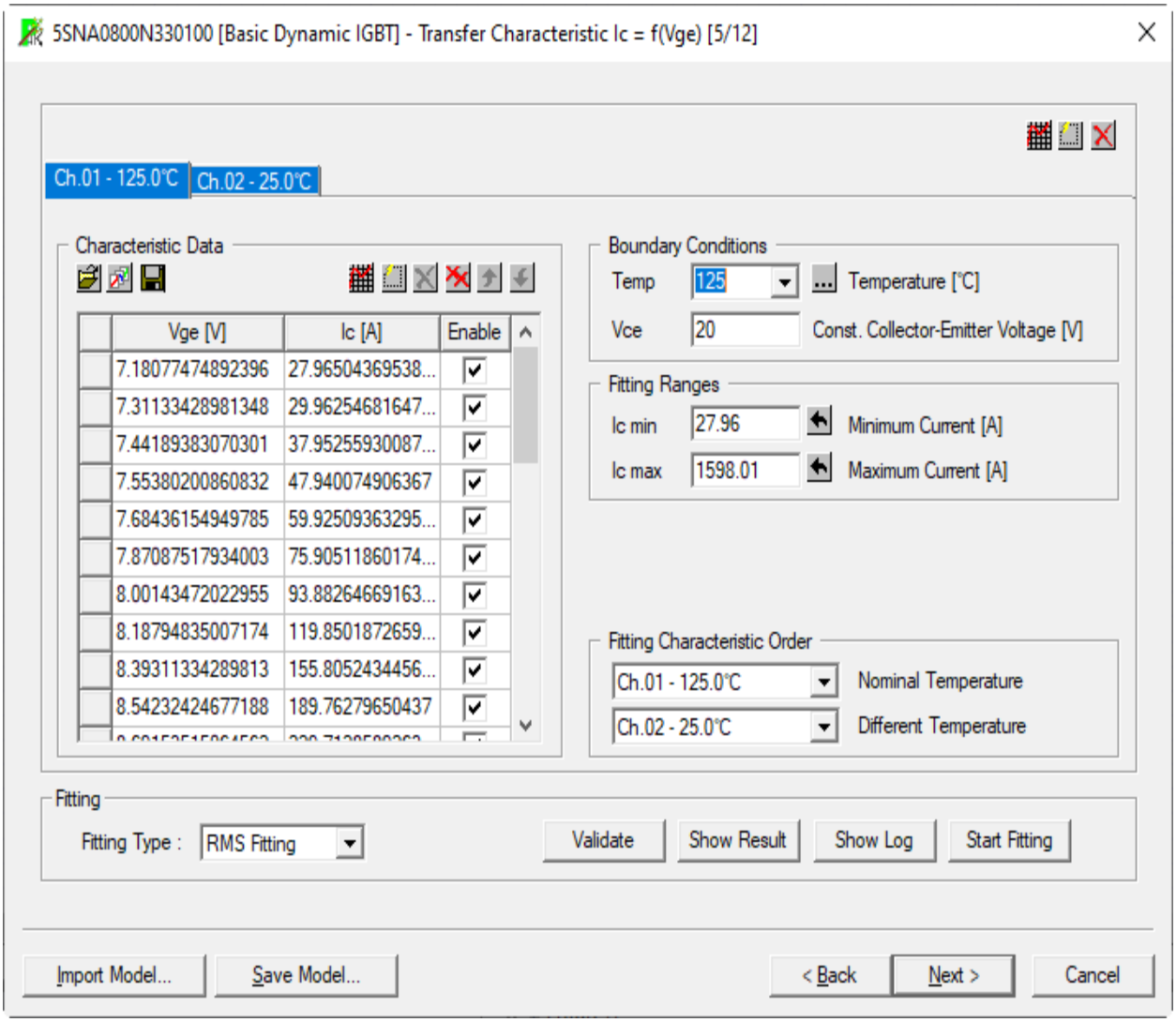
- Use the **Characteristic Data** section to enter curves or work with the data for a specific curve selected on the tabs.
- Use the **Boundary Conditions** section to specify the junction temperature (T_j) and the Collector-Emitter Voltage (V_{ce}) associated with each curve.
- Use the **Fitting Ranges** section to specify the range of the data to use when performing the fitting. Initially, the values populate with the maximum and minimum current values from the curve.
- In the **Fitting Characteristic Order** section, two sets of transfer characteristic data must be specified. The transfer characteristic under nominal junction temperature is mandatory, while the characteristic at a different temperature is used to calculate correction coefficients when the junction temperature of the device is different from the nominal temperature. If the data for the latter are not available, select **Not Used** option from the drop-down menu.

  > **Note:**
  >
  > In some cases, data sheets provide graphs for the transfer characteristic that far exceed the regular range of working conditions of the IGBT. This has a negative influence on the quality of the curve fitting by making the fitting process to focus on the high current area. Therefore, the data points used for the fitting of the transfer characteristic should be limited to approximately three times INOM.

- In the **Fitting** section, you perform the fitting of the model to the characteristic curve data and validate the results. For information about the specific fitting type see ***Fitting Types of Static Characteristics***.

For more information, click an area on the image below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



When finished, click **Next** to continue characterizing the device (Output Characteristic).

Boundary Conditions for Transfer Characteristic Data

Enter the junction temperature $\mathbf{T_j}$ and collector-emitter voltage $\mathbf{V_{ce}}$, under which the characteristic data is measured. This data is provided with the manufacturer's **Transfer Characteristic** curves as shown in **Transfer Characteristics**.

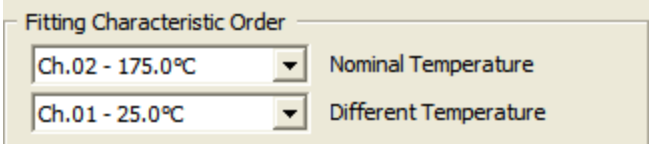
Fitting Ranges for Transfer Characteristic

When importing data for a characteristic curve, fitting range values populate with the minimum and maximum collector current values of the imported data. You can modify these values manually if a narrower fitting range is desired. Data pairs with collector current values falling into the specified fitting range are used for parameter extraction.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Fitting Characteristic Order for Transfer Characteristic

Two sets of transfer characteristic data should be specified to accurately extract a model over a range of temperatures. The transfer characteristic under nominal junction temperature is mandatory, while the characteristic at a different temperature is used to calculate correction coefficients when the junction temperature of the device is different from the nominal temperature. If the data for the latter are not available, select **Not Used** from the drop-down list, as shown in the following figure.

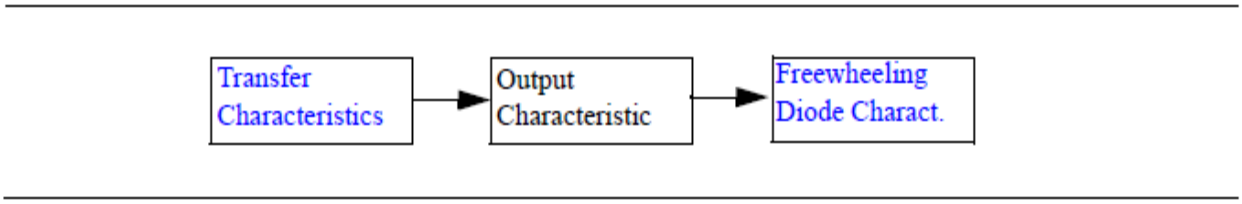


Fitting Transfer Characteristic Data

The fitting section of the **Transfer Characteristic** dialog box contains options for these functions:

- Validate
- Show Result
- Show Log
- Start Fitting

Output Characteristic



In the **Output Characteristics** dialog box, each tab corresponds to one output characteristic curve, that is, **Ic=f(Vce)** at a specific temperature. These characteristics are obtained from the output characteristic curves supplied by the manufacturer as illustrated in the example characteristic shown below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

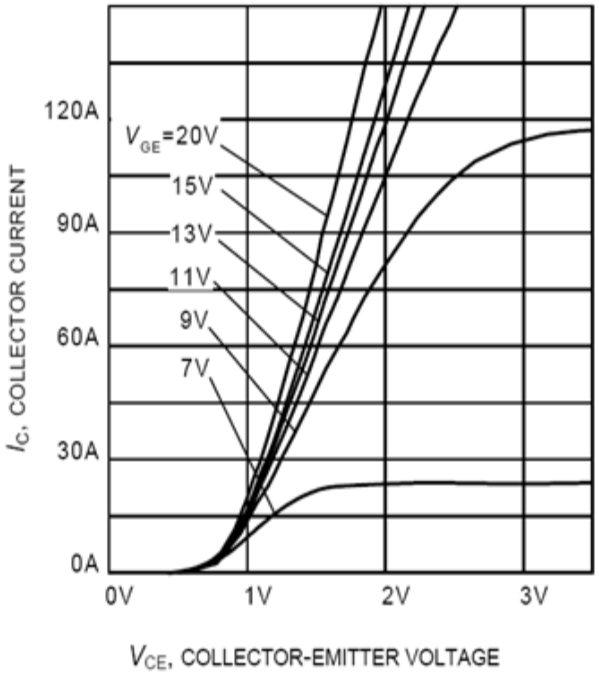


Figure 5. Typical output characteristic (T_j = 25°C)

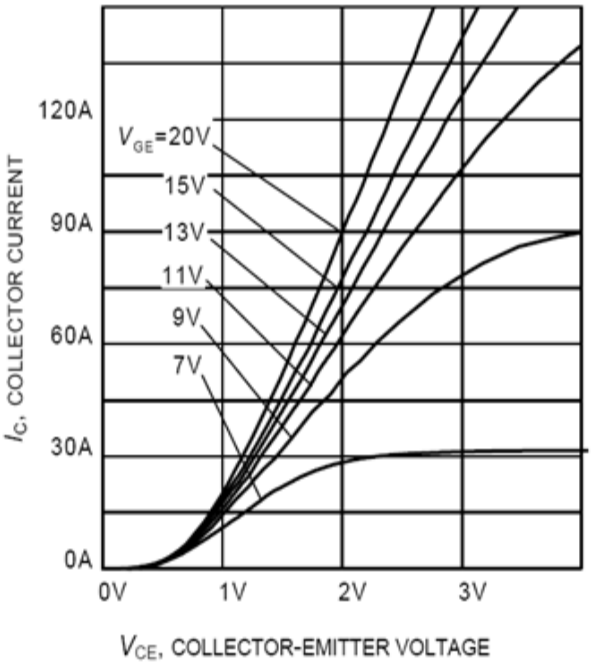


Figure 6. Typical output characteristic (T_j = 175°C)

- Use the **Characteristic Data** section to enter curves or work with the data for a specific curve selected on the tabs.
- Use the **Boundary Conditions** section to specify the junction temperature (T_j) and the Gate-Emitter Voltage (V_{ge}) associated with each curve.
- Use the **Fitting Ranges** section to specify the range of the data to use when performing the fitting. Initially, the values populate with the maximum and minimum current values from the curve.
- In the **Fitting Characteristic Order** section, you can specify these sets of transfer characteristic data.

  The **Full Saturated Branch** output characteristic under nominal junction temperature is mandatory, while the characteristic at a different temperature is used to calculate correction coefficients when the junction temperature of the device is different from the nominal temperature. If the data is available, use the value of the gate voltage which corresponds to the one used in the on-switch switching measurements (**Vge on** from the **Nominal Working Point Values** dialog box).

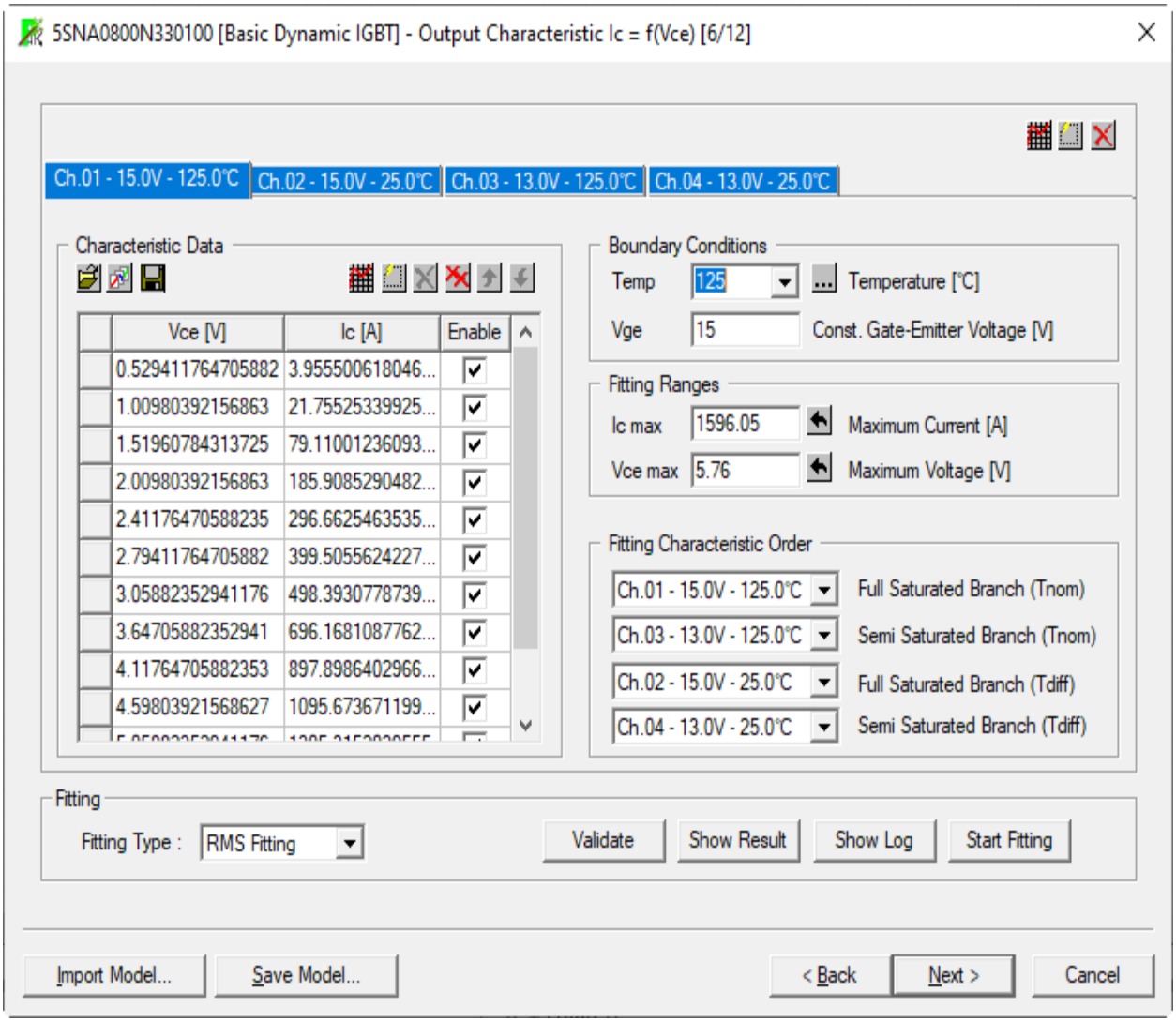
  The **Semi Saturated Branch** output data calculates correction coefficients for the output when not fully voltage saturated. The best choice for this data is a characteristic showing some flattening of the output curve as shown above for the VGE=8Vcase at 25^{o}C. Select **Not Used** option from the drop-down list for any unavailable data items. The semi-

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

saturated branch is always measured at a lower value of VGE than the fully saturated branch.

- In the **Fitting** section, you perform the fitting of the model to the characteristic curve data and validate the results.

For more information, click an area on the image below.



When finished, click **Next** to continue characterizing the device (Freewheeling Diode Characteristics).

Boundary Conditions for Output Characteristic Data

You must enter the junction temperature $\mathbf{T_j}$ and collector-emitter voltage $\mathbf{V_{ge}}$, under which the characteristic data is measured. This data is provided with the manufacturer's **Output**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Characteristic curves as shown in the following example.

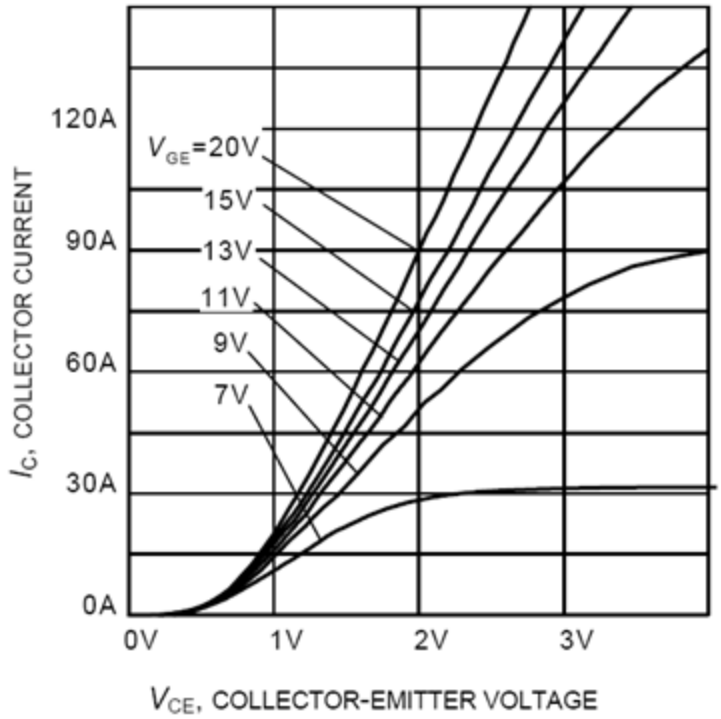


Figure 6. Typical output characteristic (T_j = 175°C)

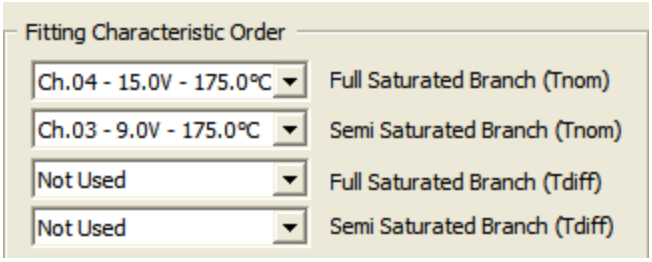
Fitting Ranges for Output Characteristic

When importing data for a characteristic curve, fitting range values populate with the maximum collector current and collector-emitter voltage values of the imported data. You can modify these values manually if a narrower fitting range is desired. Data pairs with values falling into the specified fitting range are used for parameter extraction.

Fitting Characteristic Order for Output Characteristic

You must specify four sets of transfer characteristic data to accurately extract a model over a range of temperatures. The transfer characteristic under nominal junction temperature is mandatory, while the characteristic at a different temperature is used to calculate correction coefficients when the junction temperature of the device is different from the nominal temperature. If the data for the latter are not available, select **Not Used** from the drop-down list, as shown in the following figure.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



For the Output Characteristic curves, you must enter a Full and Semi saturated curve. Looking at the example characteristic shown below, the curve for $\mathbf{V_{ge}}$=18V is the saturated curve while the $\mathbf{V_{ge}}$=15V or $\mathbf{V_{ge}}$=12V are acceptable choices for the semi-saturated case.

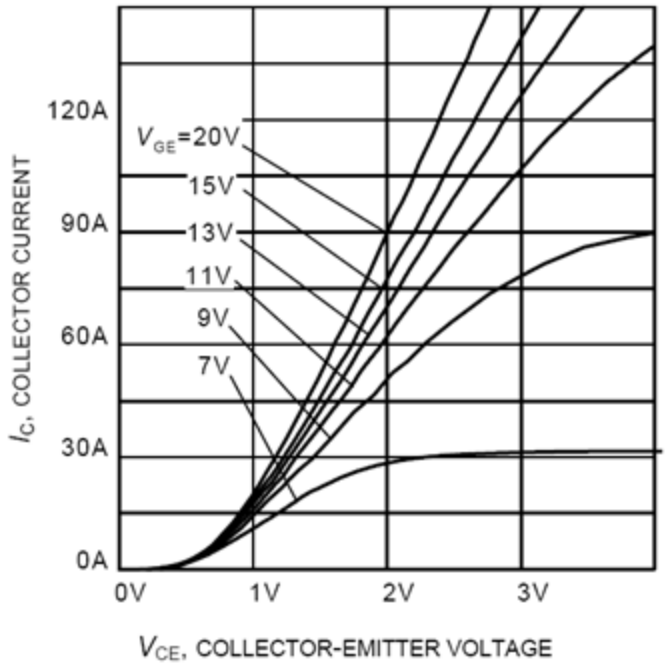


Figure 6. Typical output characteristic
(T_j = 175°C)

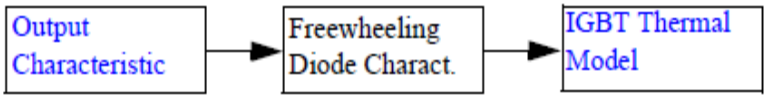
Fitting Output Characteristic Data

The fitting section of the **Output Characteristic** dialog box is identical to the **Transfer Characteristics** dialog box and contains items for these functions:

- Validate
- Show Result
- Show Log
- Start Fitting

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Freewheeling Diode Characteristic



In the **Freewheeling Diode Characteristics** dialog box, each tab corresponds to one freewheeling diode characteristic curve, that is, $\mathbf{I_f = f(V_f)}$ at a specific junction temperature $\mathbf{T_j}$. These characteristics are obtained from the forward diode characteristic supplied by the device manufacturer as illustrated below.

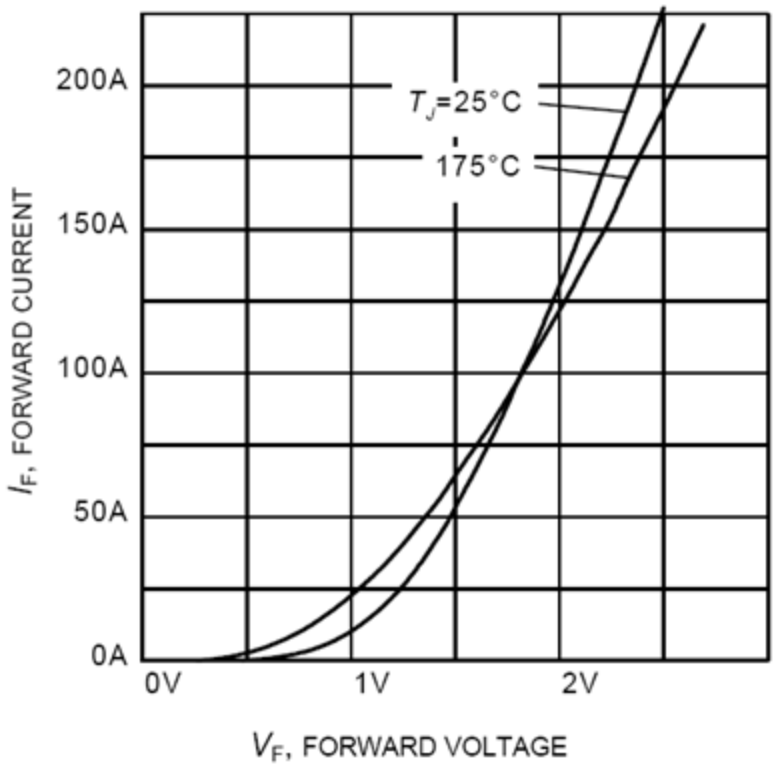


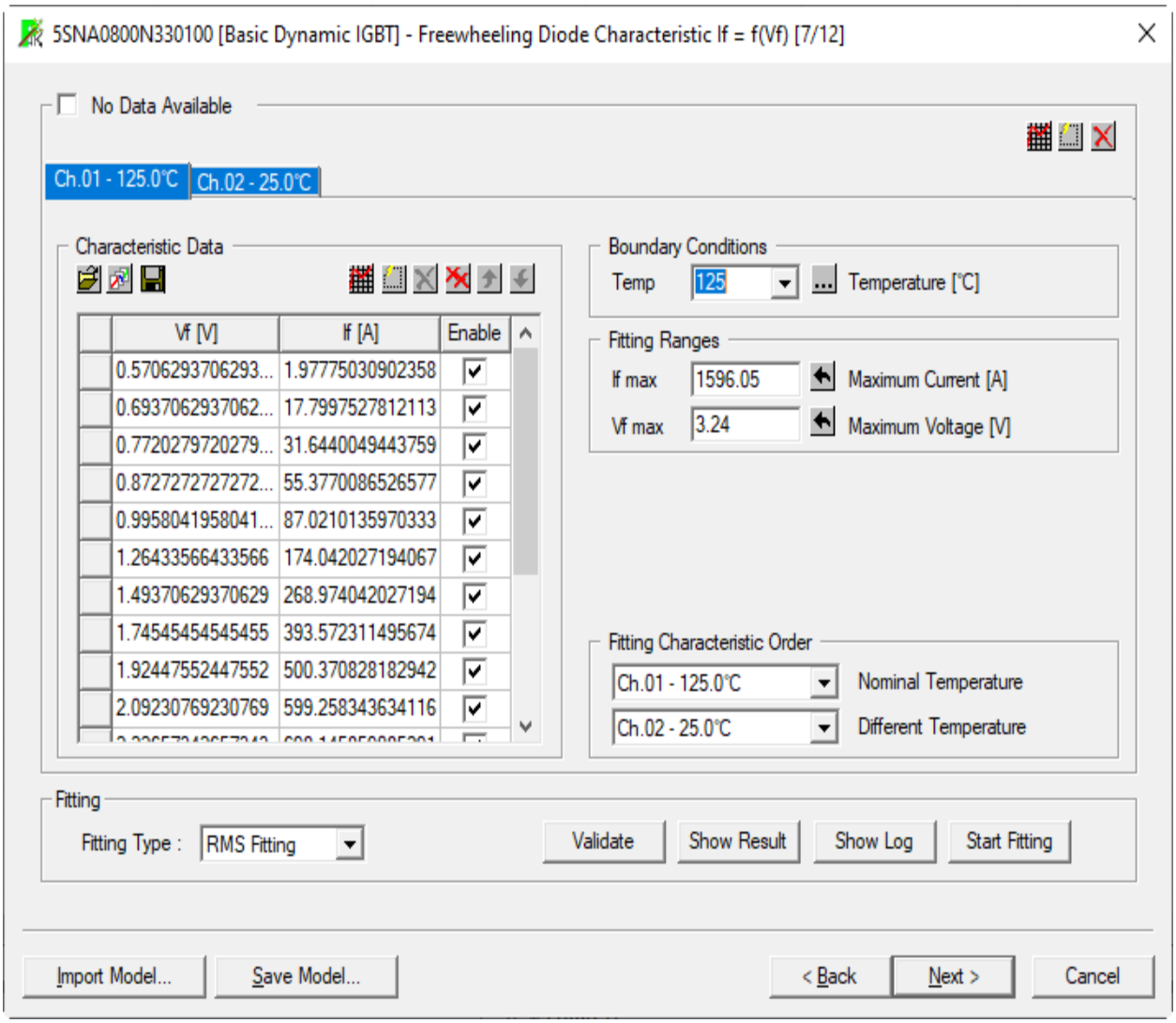
Figure 27. Typical diode forward current as a function of forward voltage

- Use the **Characteristic Data** section to enter curves or work with the data for a specific curve selected on the tabs.
- Use the **Boundary Conditions** section to specify the junction temperature (T_j) associated with each curve.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Use the **Fitting Ranges** section to specify the range of the data to use when performing the fitting. Initially, the values are filled with the maximum current and voltage values from the curve.
- In the **Fitting Characteristic Order** section, you must specify two sets of characteristic data. The characteristic under nominal junction temperature is mandatory, while the characteristic at a different temperature is used to calculate correction coefficients when the junction temperature of the device is different from the nominal temperature. If the data for the latter are not available, select **Not Used** from the drop-down list.
- In the **Fitting** section, you perform the fitting of the model to the characteristic curve data and validate the results. In addition, select the **Disable Diode** check box to disable the use of the diode in the model.

For more information, click an area on the image below.



When finished, click **Next** to continue characterizing the device (IGBT Thermal Model).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Boundary Conditions for Freewheeling Diode Characteristic

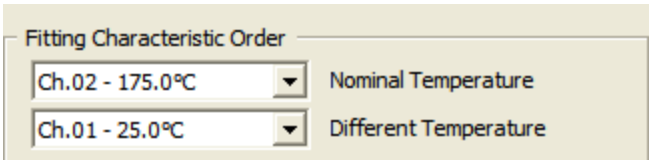
You must enter the junction temperature T_j under which the characteristic data is measured. This data is provided with the manufacturer's **Forward Diode Characteristic** curves as shown in **Freewheeling Diode Characteristic**.

Fitting Ranges for Freewheeling Diode Characteristic

When importing data for a characteristic curve, fitting range values populate with the maximum voltage and current values of the imported data. You can modify these values manually if a narrower fitting range is desired. Data pairs with values falling into the specified fitting range are used for parameter extraction.

Fitting Characteristic Order for Freewheeling Diode Characteristic

You must specify two sets of characteristic data to accurately extract a model over a range of temperatures. The diode characteristic under nominal junction temperature is mandatory, while the characteristic at a different temperature is used to calculate correction coefficients when the junction temperature of the device is different from the nominal temperature. If the data for the latter are not available, select **Not Used** from the drop-down list.



Fitting Freewheeling Diode Characteristic Data

The main functions of the **Fitting** section of the **Freewheeling Diode Characteristic** dialog box are identical to the **Transfer Characteristics** dialog box. It contains items for these functions:

- Validate
- Show Result
- Show Log
- Start Fitting

In addition, select the **Disable Diode** check box to remove the effects of a freewheeling diode from the completed model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Including Thermal Effects in the IGBT Model

Thermal effects are integrated with the IGBT model in two ways. First, using the full IGBT device thermal effects, and further using the freewheeling diode thermal effects.

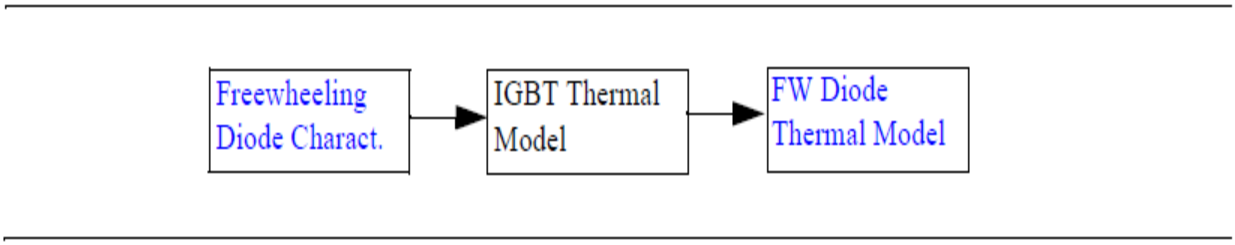
Several things should be considered when using the thermal models:

- Disabling the IGBT Thermal Model disables all thermal modeling for the device and switches the IGBT into isothermal mode.
- You may incorporate IGBT modeling but independently disable the thermal modeling of the freewheeling diode.
- You may enable thermal pins for the model, to externally model the effects of the heat sink, affecting both the IGBT and the freewheeling diode.
- You may select the topology of the internal thermal network.

For more details, see:

- IGBT Thermal Model
- Freewheeling Diode Thermal Model

IGBT Thermal Model



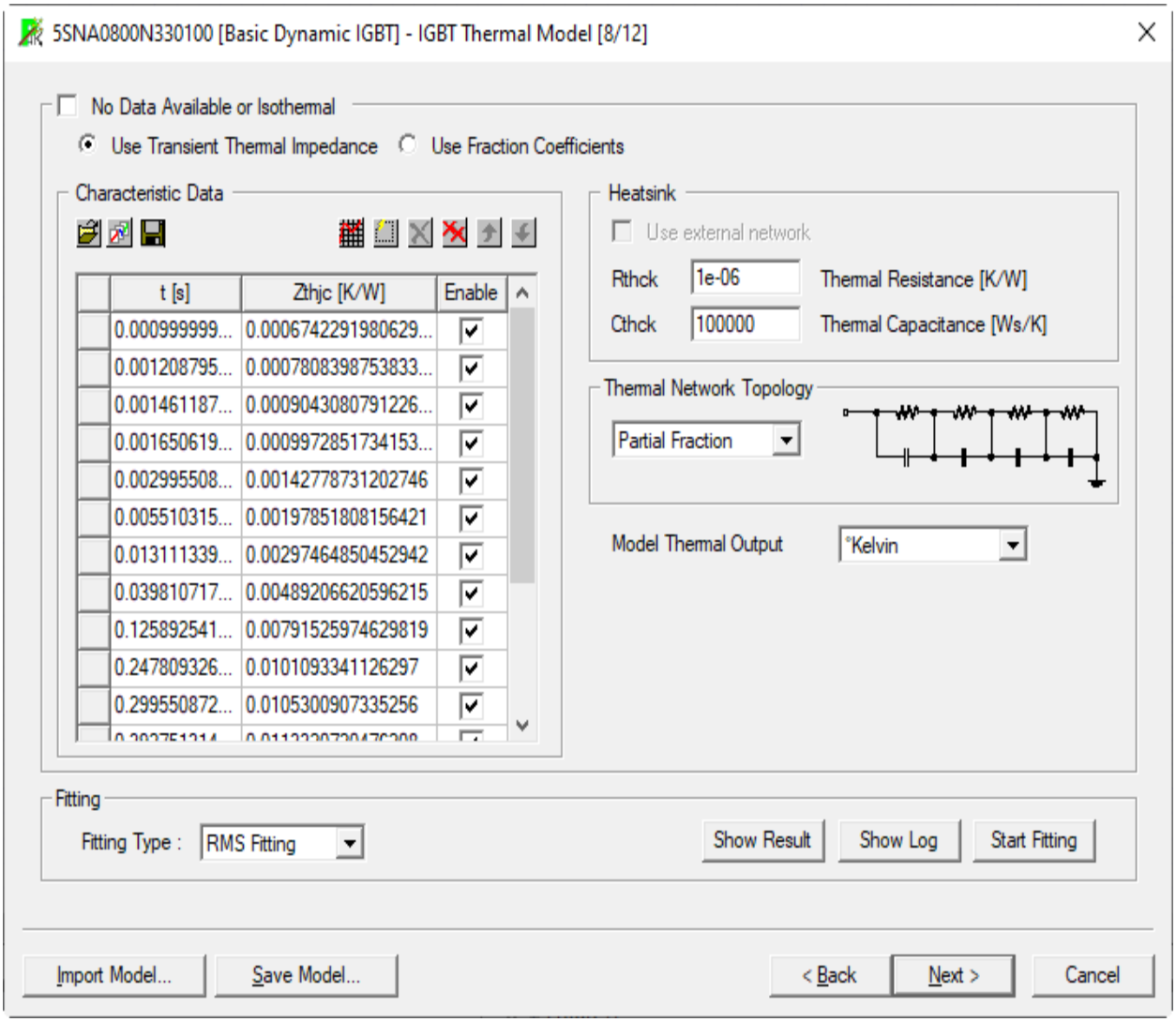
In order to model the device's thermal effects, the IGBT model uses a rational polynomial which is represented internally by a combination of four RC elements.

Use the **IGBT Thermal Model** dialog box to define the coefficients for elements of the internal thermal network by extracting them from the manufacturer's **Transient Thermal Impedance** data or by directly entering the **Fractional Coefficients** in a data table.

The table to enter the **Fractional Coefficients** is enabled by default. Select the **Use Transient Thermal Impedance** radio button to enable the entry of transient thermal impedance data into a data table.

For more information, click an area on the image below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



When finished, click **Next** to continue characterizing the device (FW Diode Thermal Model).

IGBT Thermal Fitting Mode

You can define the thermal parameters in two ways: by fitting of the transient IGBT thermal impedance waveform provided by the manufacturer, or entering the coefficient of the thermal network directly.

For more information, see:

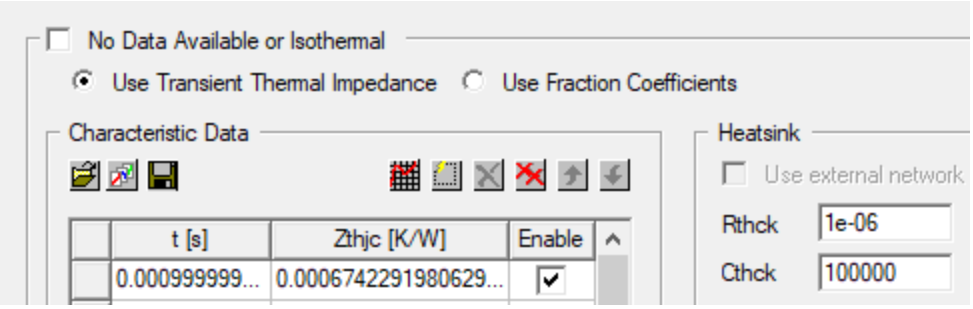
- Transient Thermal Impedance
- Fractional Coefficients

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Transient Thermal Impedance

Use **Transient Thermal Impedance** mode to enter characteristic data from the manufacturer's data sheet that describes the thermal impedance of the IGBT. Follow this procedure to use the **Transient Thermal Impedance** model:

1. Select the **Use Transient Thermal Impedance** radio button in the **Fitting Mode** section as shown below.



2. In the **Transient Thermal Impedance** characteristic data curve section of the dialog box, enter the thermal response curve as described in ***Working with Characteristic Data Curves***.

Enter the thermal impedance of the IGBT as data pairs. An example of the thermal impedance in the datasheet is shown in the following figure. Note in this case, the thermal impedance required by the device characterization tool is the thermal response under single pulse excitation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

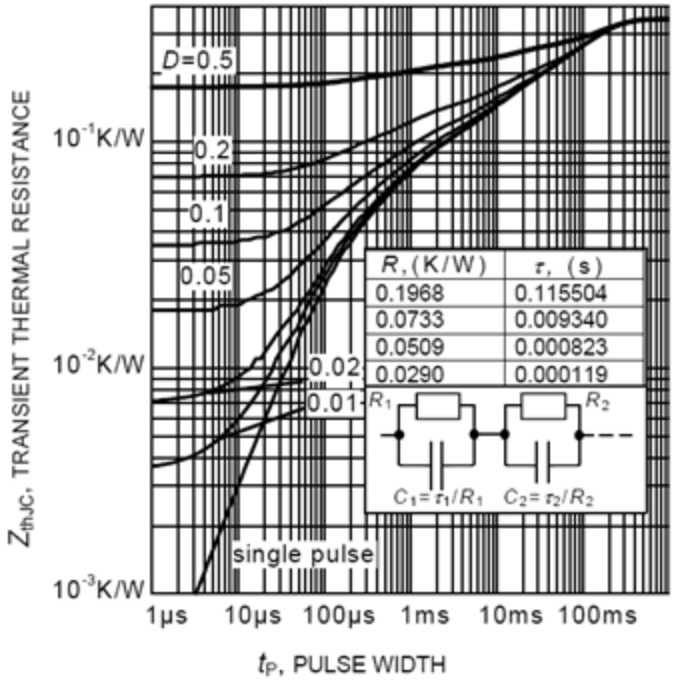
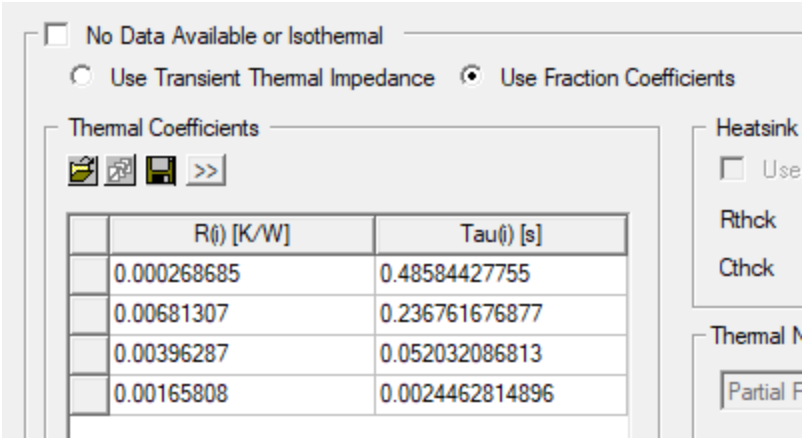


Figure 21. IGBT transient thermal resistance
($D = t_p / T$)

Fraction Coefficients Mode

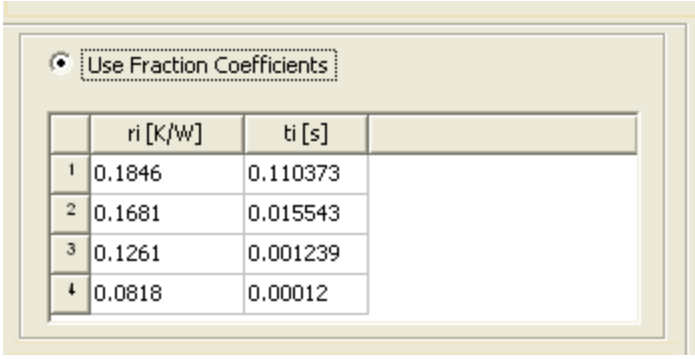
Use **Fraction Coefficients** mode to enter the thermal coefficients directly. Follow this procedure to use the **Transient Thermal Impedance** model:

1. Select the **Use Fraction Coefficients** radio button in the **Fitting Mode** area of the dialog box as shown below. The data table used to enter the thermal coefficients appears.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Enter the thermal coefficients, that is, thermal resistance $\mathbf{R_i}$ and time constant τ_i here (see the IGBT Average model reference in the component help for more information on the thermal model). These coefficients must be supplied by the device manufacturer in order to use either fractional expansion model.

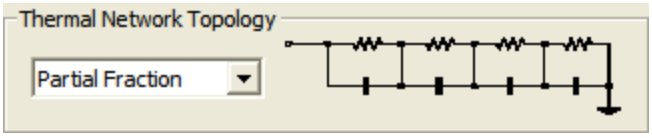


Thermal Network Topology

The **IGBT Thermal Model** can be set up to use one of two different types of fractional model topologies for the description of the thermal behavior of the IGBT and the freewheeling diode: the **partial fraction** model or the **continued fraction** model.

Partial Fraction Model

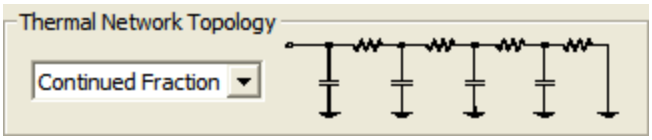
The **Partial Fraction** model treats the thermal performance of the IGBT as a rational polynomial modeled as a series of four parallel RC elements. The coefficients of the **Partial Fraction** model must be entered in the **Fraction Coefficients** section of the dialog box.



Continued Fraction Model

The **Continued Fraction** model treats the thermal performance of the IGBT as a rational polynomial modeled as a series of four RC ladder network elements. The coefficients of the **Continued Fraction** model must be entered in the **Fraction Coefficients** section of the dialog box.

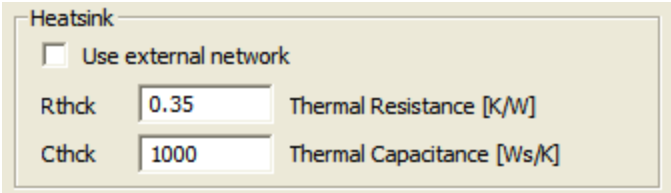
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



IGBT Heatsink Parameters

Use the **IGBT Thermal Model** dialog box to specify the parameters for the heatsink directly (internal heatsink) or to provide external thermal pins to the component in order to connect a thermal network that serves as an external heatsink. The heatsink (internal or external) affects both the IGBT and the freewheeling diode.

If you choose the internal heatsink, the **Thermal Resistance** and **Thermal Capacitance** of the external heatsink should be supplied in this section of the dialog box. This information must be supplied by the heatsink manufacturer. Use the default values if no information is available regarding the external heatsink.

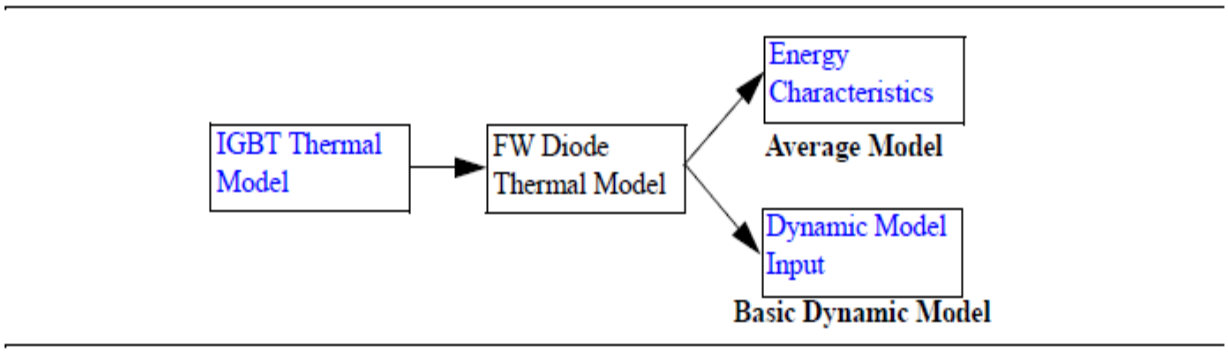


Fitting Characteristic Data Curves

The main functions of the fitting section of the **IGBT Thermal Model** dialog box are identical to the **Transfer Characteristics** dialog box. The section contains items for these functions:

- Show Result
- Show Log
- Start Fitting

Freewheeling Diode Thermal Model

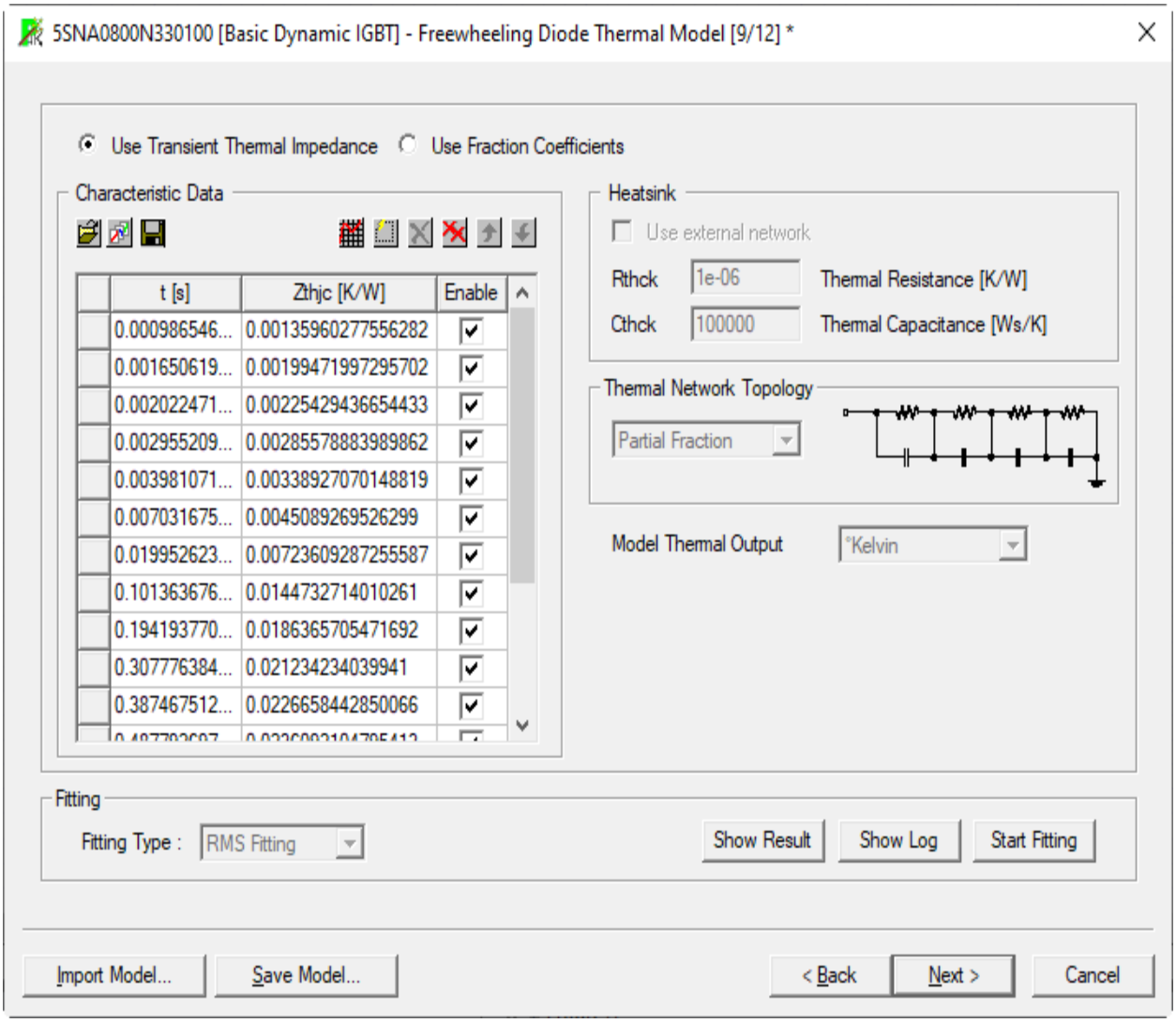


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The **Freewheeling Diode Thermal Model** dialog box is similar to the **IGBT Thermal Model**. The freewheeling diode transient thermal impedance or the fraction thermal coefficients can normally be found in the manufacturers' datasheet.

> **Note:**
>
> Certain aspects of the thermal modeling are defined commonly for the IGBT and the freewheeling diode (for example, topology of the thermal model, type, and parameters of the heatsink, and type of the thermal output). Therefore, those parameters are accessible only on the thermal parameter page for the IGBT.

Click an area on the image below for more detailed information.

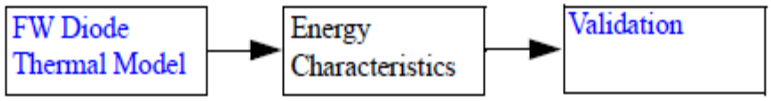


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

When finished, click **Next** to continue characterizing the device.

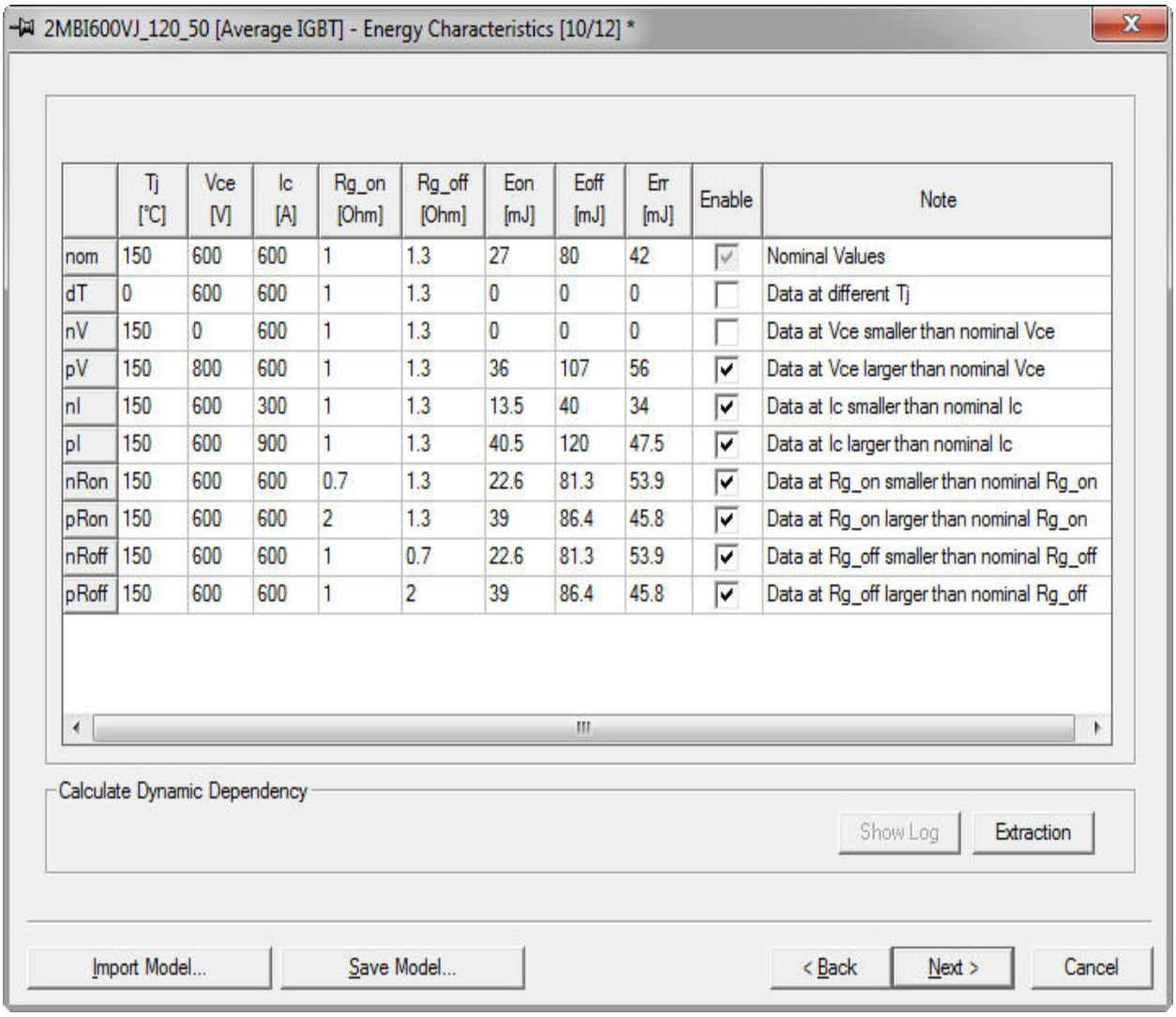
- For an Average IGBT, see ***Energy Characteristics***.
- For a Basic Dynamic IGBT, see ***Dynamic Model Input***.

Energy Characteristics (Average IGBT and Average Power MOSFET)



Use the **Energy Characteristics** dialog box to enter the data for the calculation of the energy correction coefficients for the Average IGBT and Average Power MOSFET models. The complete set of energy correction coefficients consists of switching energy data under 10 different working conditions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



	Tj [°C]	Vce [V]	Ic [A]	Rg_on [Ohm]	Rg_off [Ohm]	Eon [mJ]	Eoff [mJ]	Err [mJ]	Enable	Note
nom	150	600	600	1	1.3	27	80	42	☑	Nominal Values
dT	0	600	600	1	1.3	0	0	0	☐	Data at different Tj
nV	150	0	600	1	1.3	0	0	0	☐	Data at Vce smaller than nominal Vce
pV	150	800	600	1	1.3	36	107	56	☑	Data at Vce larger than nominal Vce
nI	150	600	300	1	1.3	13.5	40	34	☑	Data at Ic smaller than nominal Ic
pI	150	600	900	1	1.3	40.5	120	47.5	☑	Data at Ic larger than nominal Ic
nRon	150	600	600	0.7	1.3	22.6	81.3	53.9	☑	Data at Rg_on smaller than nominal Rg_on
pRon	150	600	600	2	1.3	39	86.4	45.8	☑	Data at Rg_on larger than nominal Rg_on
nRoff	150	600	600	1	0.7	22.6	81.3	53.9	☑	Data at Rg_off smaller than nominal Rg_off
pRoff	150	600	600	1	2	39	86.4	45.8	☑	Data at Rg_off larger than nominal Rg_off

These data sets must be entered following a predefined format:

- The first row is the set of data under the nominal working conditions (required).
- Each subsequent row may have only one change in the working condition; for example, temperature T is different from the value in the nominal values row. The required variation for each row is listed in the **Note** column of the table and is described in ***Energy Characteristic Default Row Description***.

If any of these data sets are not available, clear the **Enable** check box for this row and the corresponding coefficients will remain at their default values (see the IGBT Average model reference for more information on the algorithm of the switching energy calculation). Additional information provided will be used to fine tune these coefficients.

E	**Turn-On Switching Loss**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

E_{on}	
E_{off}	**Turn-Off Switching Loss**
E_{rr}	**Reverse Recovery Switching Loss**

Click **Extraction** to calculate dynamic dependency. When the extraction is complete, **Show Log** is enabled, and you can view the results after fitting.

When finished, click **Next** to continue characterizing the device (Validation).

Energy Characteristic Default Row Description

- The **nom** row is the measured data under the nominal working condition. This data provides the basis for the energy correction of other working conditions, and is required for the calculation of the correction coefficients. The **Enable** check box for this row is always selected. The nominal working conditions are populated to the second to tenth rows of data according the pre-defined format.
- The **dT** row is data at a different temperature $\mathbf{T_j}$ from the nominal value. This data is used to calculate the temperature correction coefficient. In this row, only the value of $\mathbf{T_j}$ and the switching energies are editable. The rest of the data is copied from the first row. If this set of data is not available, clear the **Enable** check box and the temperature correction coefficient will remain at its default value.
- The **nV** and **pV** rows are data at a different collector-emitter voltage $\mathbf{V_{ce}}$: one at a higher voltage than the nominal $\mathbf{V_{ce}}$ and the other at a lower value. This data calculates coefficients for doubled-sided voltage correction function (see the IGBT Average model reference in the Twin Builder Components help for more information). In these rows, only the value of $\mathbf{V_{ce}}$ and the switching energies are editable. The rest of the data is copied from the first row. If this set of data is not available, clear the **Enable** check box and the voltage correction coefficient will remain at its default value.
- The **nI** and **pI** rows are data at different collector current $\mathbf{I_c}$: one at a higher current than the nominal $\mathbf{I_c}$ and the other at a lower value. This data calculates the coefficients for doubled-sided current correction function (see the IGBT Average model reference in the component help for more information). In these two rows, only the value of $\mathbf{I_c}$ and the switching energies are editable. The rest of the data is copied from the 1st row. If any of these sets of data is not available, clear the **Enable** check box and the corresponding current correction coefficients will remain at its default value.
- The **nRgon**, **pRgon**, **nRgoff**, and **pRgoff** rows are data at different gate resistances: at a higher and lower On-Switch resistance or at a higher and lower Off-Switch resistance than the nominal $\mathbf{R_{g_on}}$ or $\mathbf{R_{g_off}}$ and the other at a lower value. This data calculates the coefficients for doubled-sided gate resistance correction function (see the IGBT Average model reference in the component help for more information). Only the value of $\mathbf{R_{g_on}}$ or$\mathbf{R_{g_off}}$ and the switching energies are editable. The rest of the data is copied from the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

first row. If any of these sets of data is not available, clear the **Enable** check box and the corresponding gate resistance correction coefficients will remain at its default value.

Fitting Types of Static Characteristics

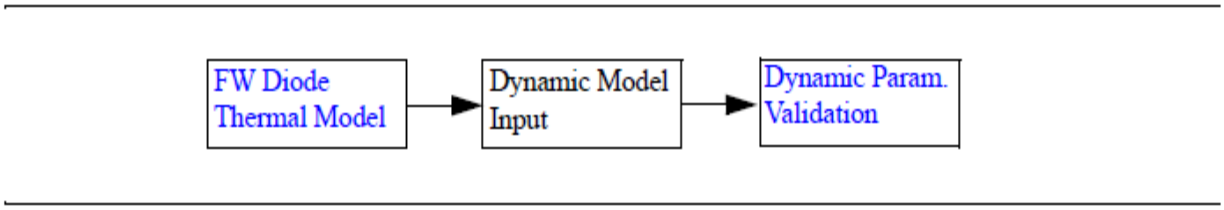
RMSFIT is the classical root mean square fitting to minimize the residue. It needs the analytical or approximate calculation of partial deviations and shows poor convergence behavior for strong parameter correlation. A good initial parameter guess is necessary. RMSFIT is the default fitting method.

GFIT is a gradient based curve fitting. A ternary parameter variation is selected on the base of a minimum residue. This process is repeated while the range of variation is reduced. The final parameter accuracy is the last range of variation. Thus, the temporary residue moves on gradient path to the minimum. This approach works well for strong parameter correlation, heavy nonlinearity, and discontinuities.

In some cases, it is advantageous to use GFIT instead of RMSFIT when:

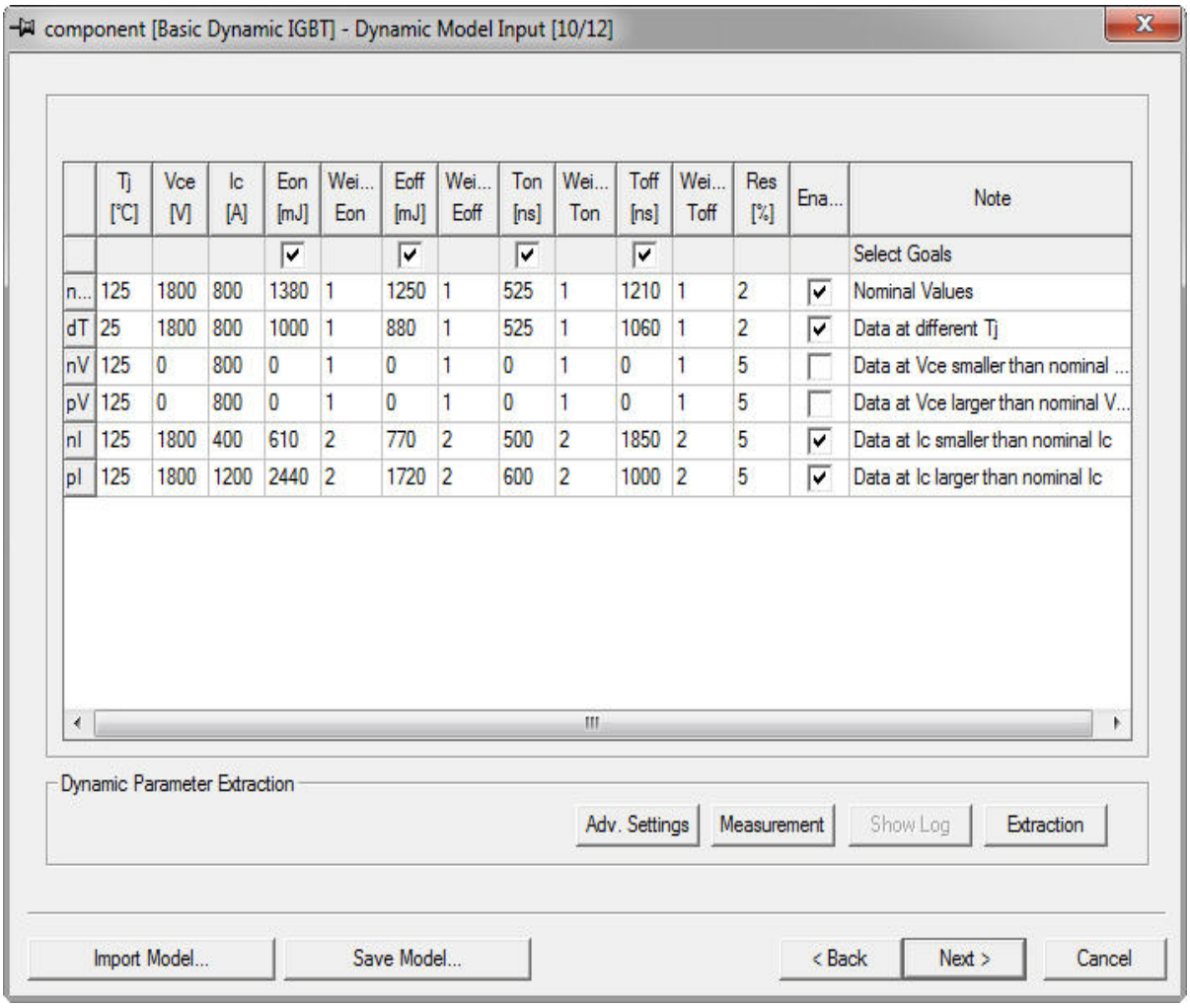
- Results are not satisfying, or fitting takes too much time.
- Some model parameters run out of the expected scope.
- Simulation using the final component shows convergence issues.
- Find the best compromise of a general unsatisfactory fitting result.

Dynamic Model Input (Basic and Advanced Dynamic IGBT and Basic Dynamic Power MOSFET)



In the **Dynamic Model Input** page, you can enter the data for the calculation of the energy correction coefficients and switching delay times for the Basic and Advanced Dynamic IGBT and the Basic Dynamic Power MOSFET models. The switching energy data under at least six different working conditions are required to calculate the complete set of energy correction coefficients.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



These six data sets must be entered following a predefined format:

- The first row is the set of data under the nominal working conditions.
- Each subsequent row may have only one change in the working condition; for example, temperature T is different from the value in the nominal values row. The required variation for each row is listed in the **Note** column.

If any of these data sets are not available, clear the **Enable** check box for the row and the corresponding coefficients will remain at their default values (see the IGBT Basic Dynamic model reference in the component help for more information on the algorithm of the switching energy calculation). Additional information provided will be used to fine tune these coefficients.

Dynamic model parameters of the Basic Dynamic IGBT model

Eon	Turn-On Switching Loss

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Eoff	Turn-Off Switching Loss
Ton	Turn-On Time (vendor-specific)
Toff	Turn-Off Time (vendor-specific)
Residue	The target error in the parameter fitting for the specified case. In general, lowering the residue results in higher accuracy at the expense of extraction time; however, lower residues are not always achievable. Increasing the residue values increases the probability of successful convergence. The default provides good model accuracy with reasonable extraction time. Fitting of the nominal working point must converge before the extraction process can proceed to other working points. Failure to converge on any other working point will warn you of the failure and provide you with the option to continue the extraction process with other working points. Change the residue to place more or less emphasis on fitting to a set of parameters.

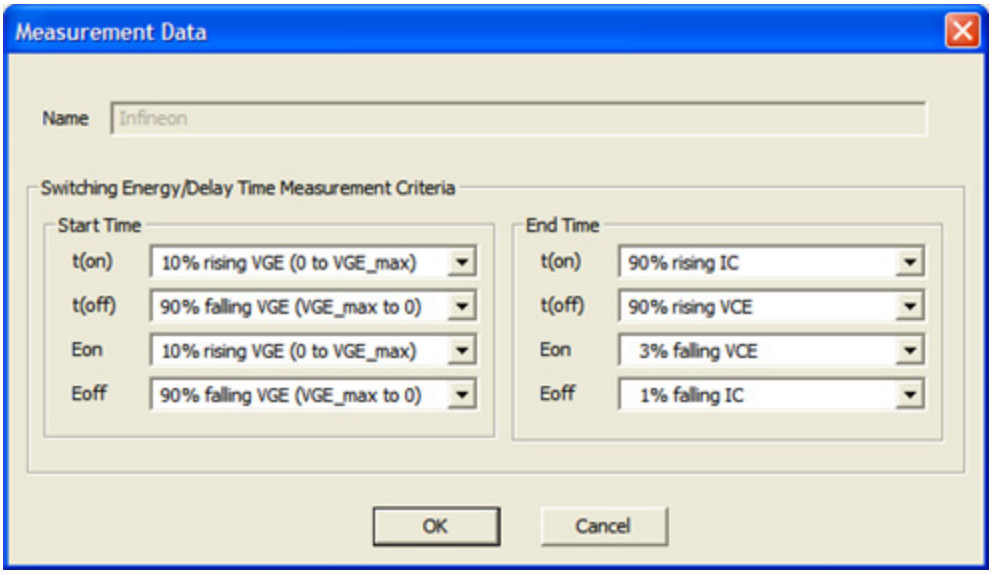
In addition to those dynamic parameters, the following table lists the parameters whose values can be determined without the need of a measuring template. These values result from either integration or direct measurement of the simulated wave form.

Qrr	Reverse recovery charge
Irr	Reverse recovery absolute current peak

You can adjust weights for the switching energies, **Eon** and **Eoff**, the delay times, **Ton** and **Toff**, and for the reverse recovery characteristics, **Qrr** and **Irr**. A higher **Weight** gives a higher importance to fit more precisely to that particular value. A default weight of 1 is set for these parameters. By default, the reverse recovery characteristics are not used. They can be turned on the **Model & Goal Settings** page in the **Advanced Settings** dialog box.

It is possible to define the Switching Energy and Delay Time Measurement criteria; click **Measurement** to open the **Measurement Data** dialog box in which you can set the start and stop times for **Eon**, **t(on)**, **Eoff**, and **t(off)** for the device being characterized.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

The **Measurement Data** dialog box contains the same settings as those in the **Manufacturer Data** dialog box (see ***Component Information***).

When finished, click **Next** to continue characterizing the device (see ***Dynamic Parameter Validation***).

Dynamic Parameters of the Advanced Dynamic IGBT Model

The first group of parameters is the same as those used for the Basic Dynamic IGBT model.

Eon	Turn-On Switching Loss
Eoff	Turn-Off Switching Loss
Ton	Turn-On Time (vendor-specific)
Toff	Turn-Off Time (vendor-specific)

In addition to the basic dynamic parameters, the following table lists the advanced dynamic parameters whose values can be determined without the need of a template function fitting. These values result from either integration or interpolation of the simulated wave form.

Qrr	Reverse recovery charge

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Flux	Off switch voltage overshoot area
Cos	Relative current overshoot at on switch
Vos	Relative voltage overshoot at off switch
TdG	Input voltage delay time before current rise
TrG	Input voltage rise time
TsG	Input voltage storage time before current fall
TfG	Input voltage fall time
TdI	Load current delay time before current rise
TrI	Load current rise time
TsI	Load current storage time before current fall
TfI	Load current fall time
TdV	Voltage wave form delay time before voltage rise (off switch)
TrV	Voltage wave form rise time (off switch)
TsV	Voltage wave form delay time before voltage fall (on switch)
TfV	Voltage wave form fall time (on switch)

Detailed descriptions of each of these are given below.

Q_{RR}

Q_{RR} is an integral value. The integration starts above 100% of the collector current which is equal to the DC like load current at this moment and goes until the current overshoot has settled down to 100% collector current again, or until the next off switch trigger occurs. The equation or the measurement instruction is:

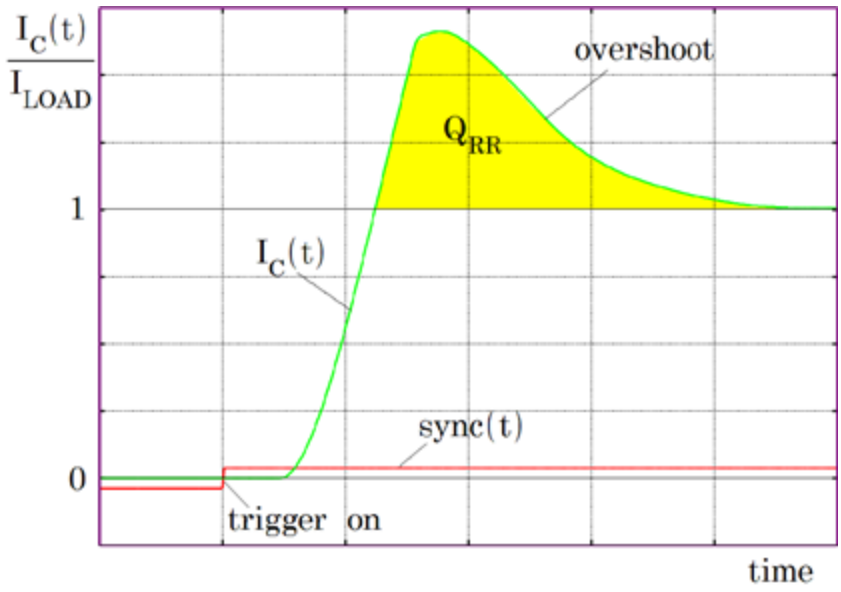
$$Q_{RR} = \int (I_C - I_{LOAD}) dt$$

if the current overshoot,
$\Delta I_C = I_C - I_{LOAD}$

is positive.

The value **Q_{RR}** includes the charge of the capacitive components of the upper half bridge transistor and the excess charge storage of the free wheeling diode parallel to the upper transistor at the moment of I_{RR} zero crossing. It represents the remaining excess charge at the crossing time.

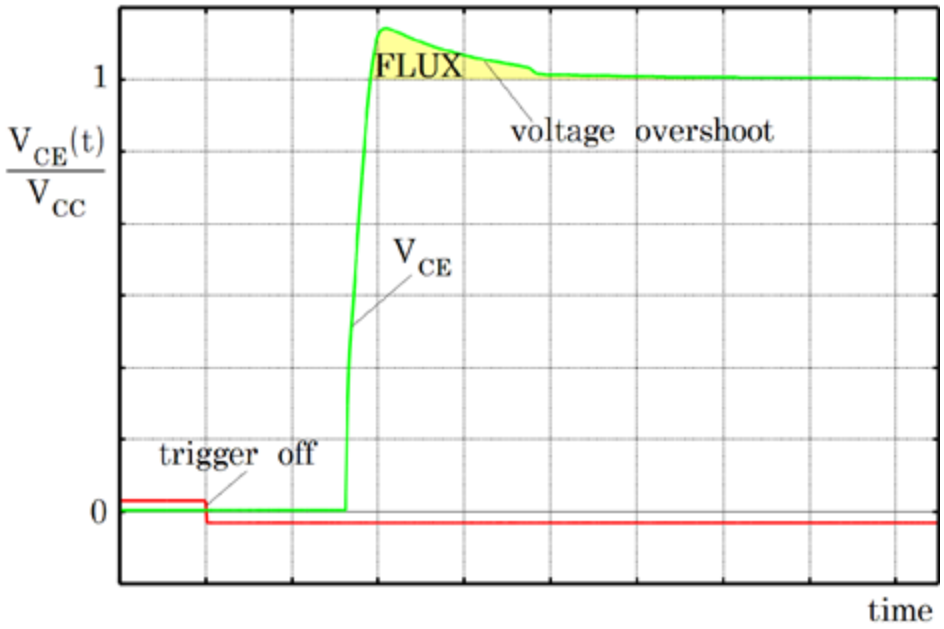
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Flux

Another integral value is the area under the voltage overshoot. Define the voltage overshoot as $\Delta V_{CE} = V_{CE} - V_{CC}$, then $FLUX = \int \Delta V_{CE}(t)\,dt$ if the voltage overshoot is positive or maximum until the next on switch trigger occurs. This overshoot is caused by stray inductances of the circuit from the power source to the half bridge, throughout the half bridge back to the power source ground. From FLUX it is possible to calculate the effective stray inductance L_σ. Using FLUX module stray inductances can be adapted to the best fit.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

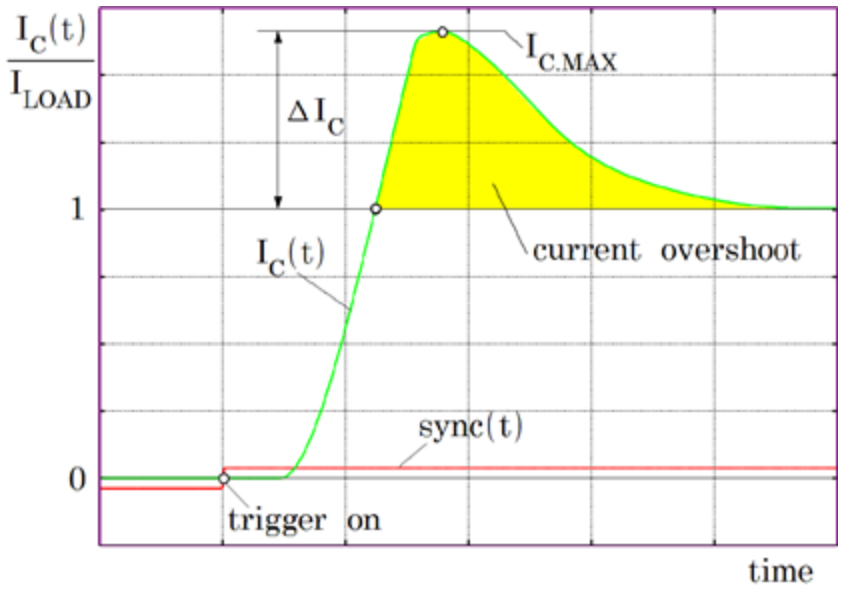


Cos

To adopt the parasitic capacitances and the charge controlling life time parameters **TAUFD** and **TAUBE** according to given $\frac{di}{dt}$ values of I_C this relative collector current overshoot goal value $\mathrm{COS} = \frac{\Delta I_C}{I_{LOAD}(100\%)}$ is useful.

COS strongly depends on the reverse recovery current slope $\frac{di}{dt}$ at the moment of zero crossing and thus on the total inductance L_{TOT} of the current commutation path. Both values, **COS** and the overshoot duration, fit to preserve the excess charge $\mathbf{Q_{RR}}$.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

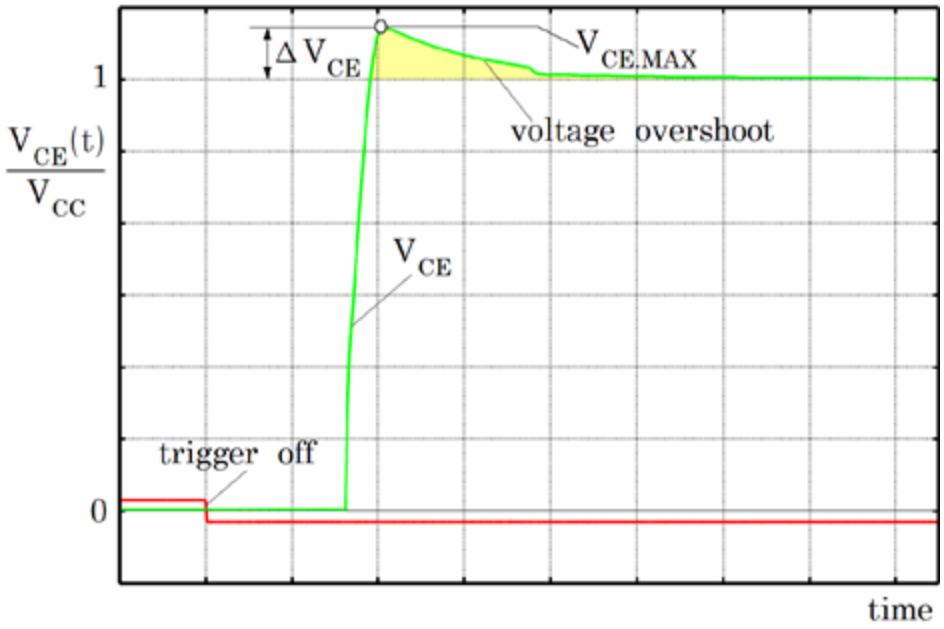


Vos

The voltage overshoot is sensitive to the stray inductances L_σ and $\frac{di}{dt}$ values. It can be used to fit for the most appropriate circuit stray inductance.

Its definition is $\mathrm{VOS} = \frac{\Delta V_{CE}}{V_{CC}(100\%)}$.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



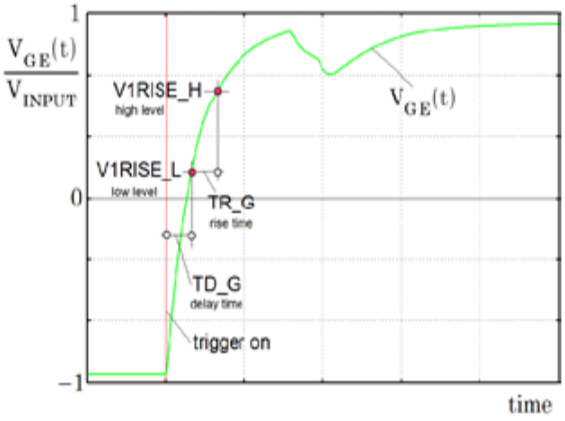
T_xG, T_xI, T_xV

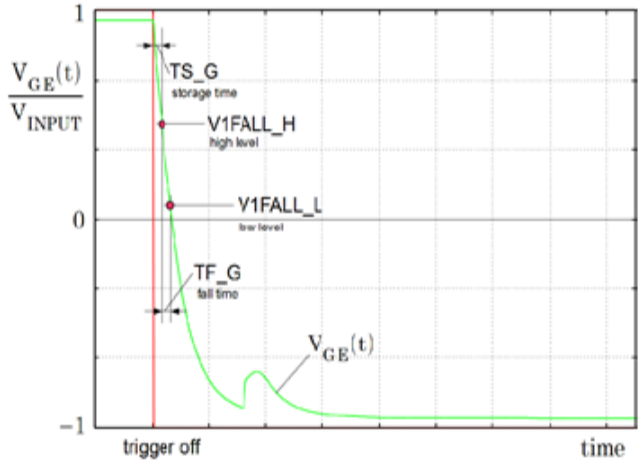
These advanced parameters provide full flexibility for device parameter optimization. Every switched signal is characterized by typical trigger levels and the corresponding time values. Usually a signal rise time is defined from 10% of rise beginning to 90% of signal maximum. This is only one example for more possible definitions.

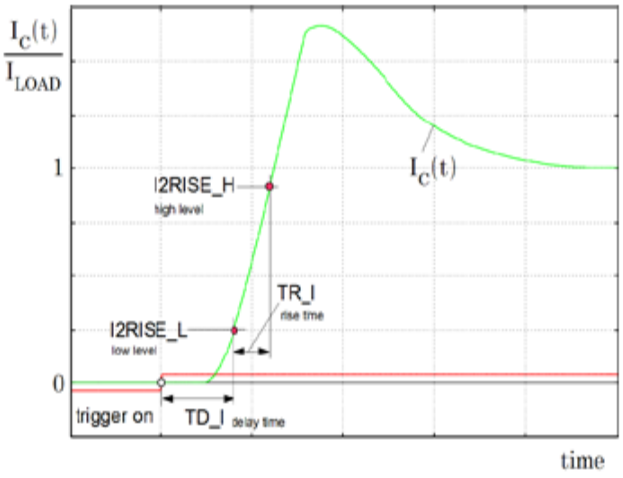
Here four trigger points and their times used. Starting from Low come delay, rise, storage and fall time. This simple definition is applied to the input wave form $V_{GE}(t)=V_1(t)$, the output current $I_C(t)=I_2(t)$ and the output voltage $V_{CE}(t)=V_2(t)$.

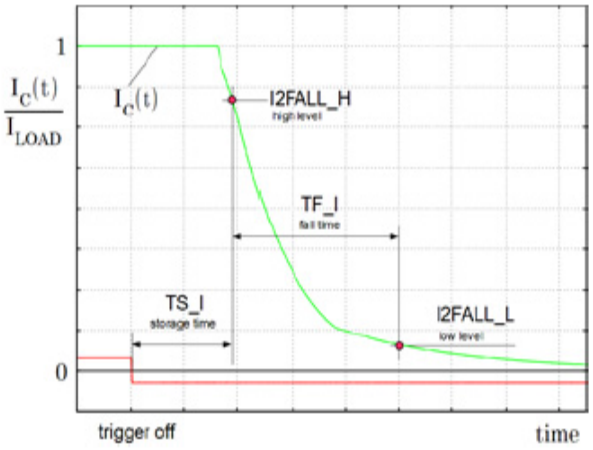
As constraints the device characterization must be provided the level definition. Each trigger level can accept values in the range of 0.0 to 1.0. The resulting time values are all related to the on switch or the off switch trigger. By combination of these twelve time durations, it is possible to generate very different measurement conditions which are used as fitting targets or for comparison between measurement and simulation. The following images give an overview of the existing definitions:

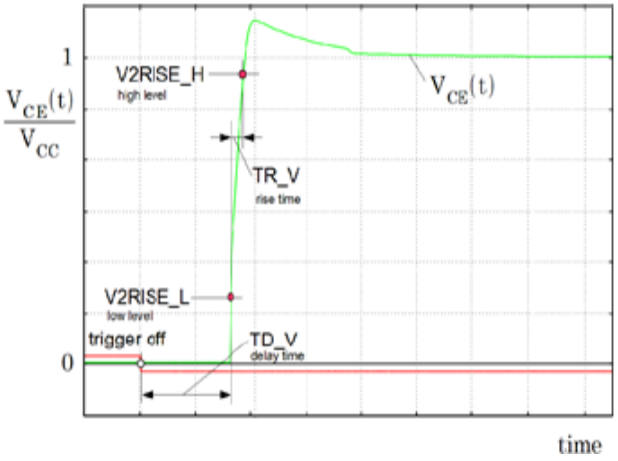
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

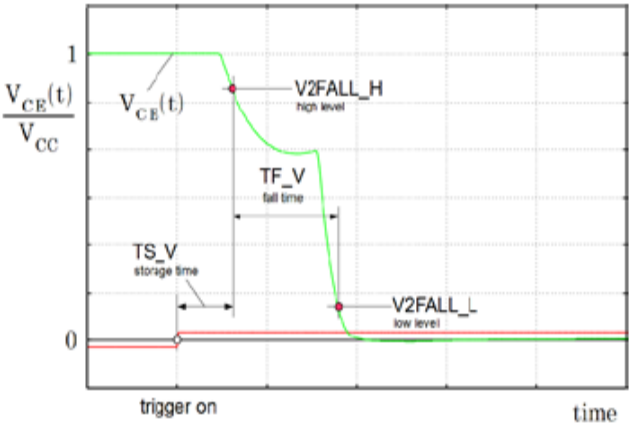






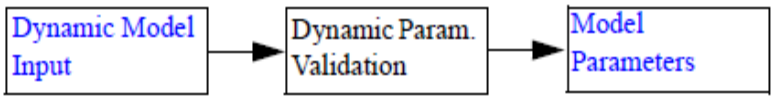






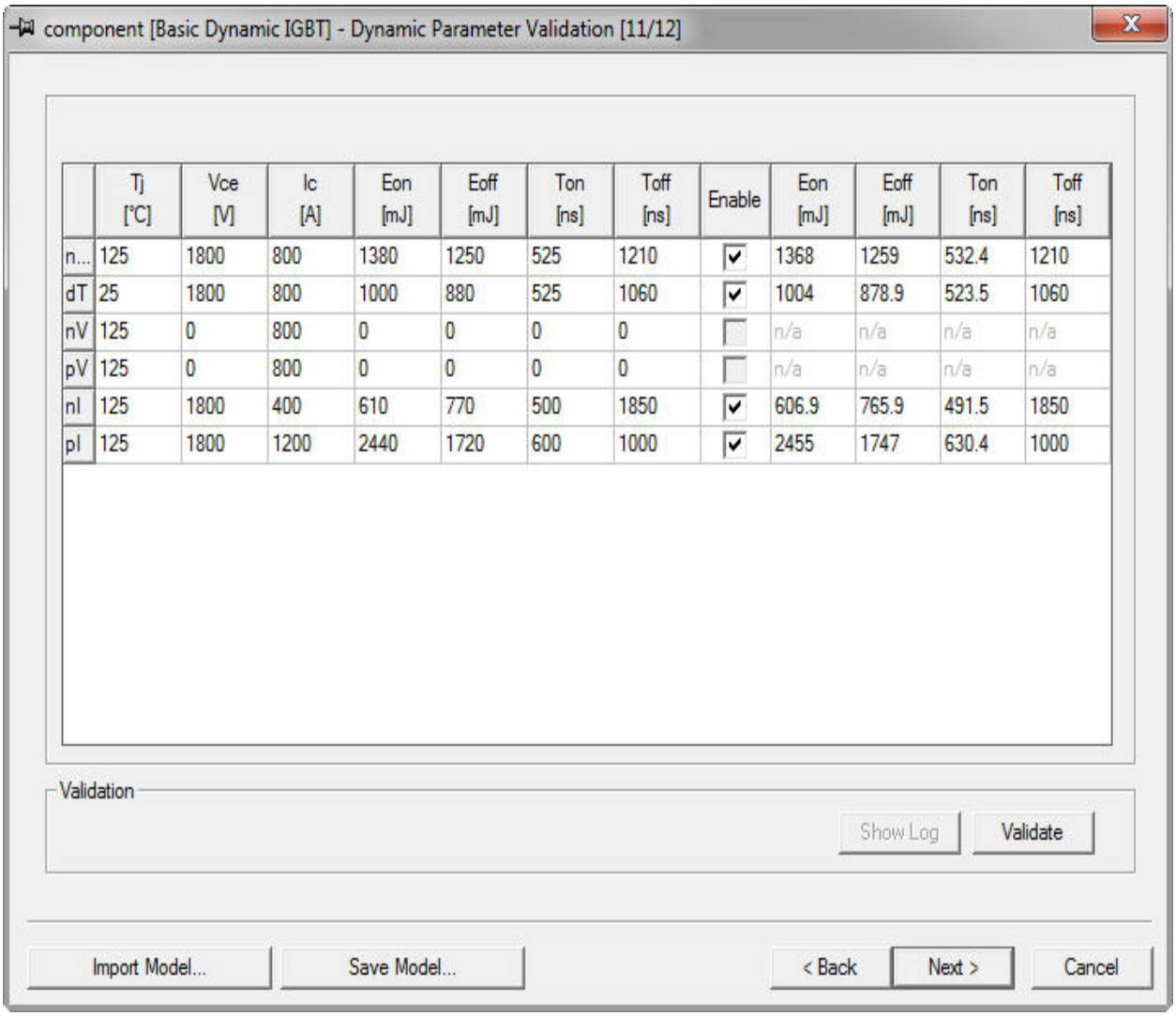
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Dynamic Parameter Validation (Basic and Advanced Dynamic IGBT and Basic Dynamic Power MOSFET)



In the **Dynamic Parameter Validation** page, you can check the accuracy of the parameterized component through the validation of switching energies. The table populates with data at the nominal and difference temperatures used to create the model. To the left of the **Enable** button are the switching energies entered for the model creation; to the right are those calculated through the dynamic model for validation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



	Tj [°C]	Vce [V]	Ic [A]	Eon [mJ]	Eoff [mJ]	Ton [ns]	Toff [ns]	Enable	Eon [mJ]	Eoff [mJ]	Ton [ns]	Toff [ns]
n...	125	1800	800	1380	1250	525	1210	☑	1368	1259	532.4	1210
dT	25	1800	800	1000	880	525	1060	☑	1004	878.9	523.5	1060
nV	125	0	800	0	0	0	0	☐	n/a	n/a	n/a	n/a
pV	125	0	800	0	0	0	0	☐	n/a	n/a	n/a	n/a
nI	125	1800	400	610	770	500	1850	☑	606.9	765.9	491.5	1850
pI	125	1800	1200	2440	1720	600	1000	☑	2455	1747	630.4	1000

To validate the model:

1. Click **Validate** to calculate the energies in a test configuration.

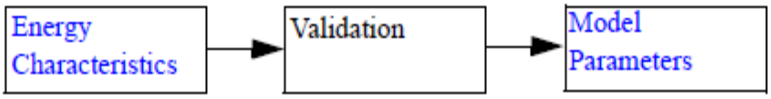
   Entries on the right hand side of the **Enable** check box populate after the validation process ends. The validation is performed through a series of transient analyses under the specified working conditions.

   > **Note:**
   >
   > Select or clear the **Enable** check box to enable or disable the validation process for any row in the table.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

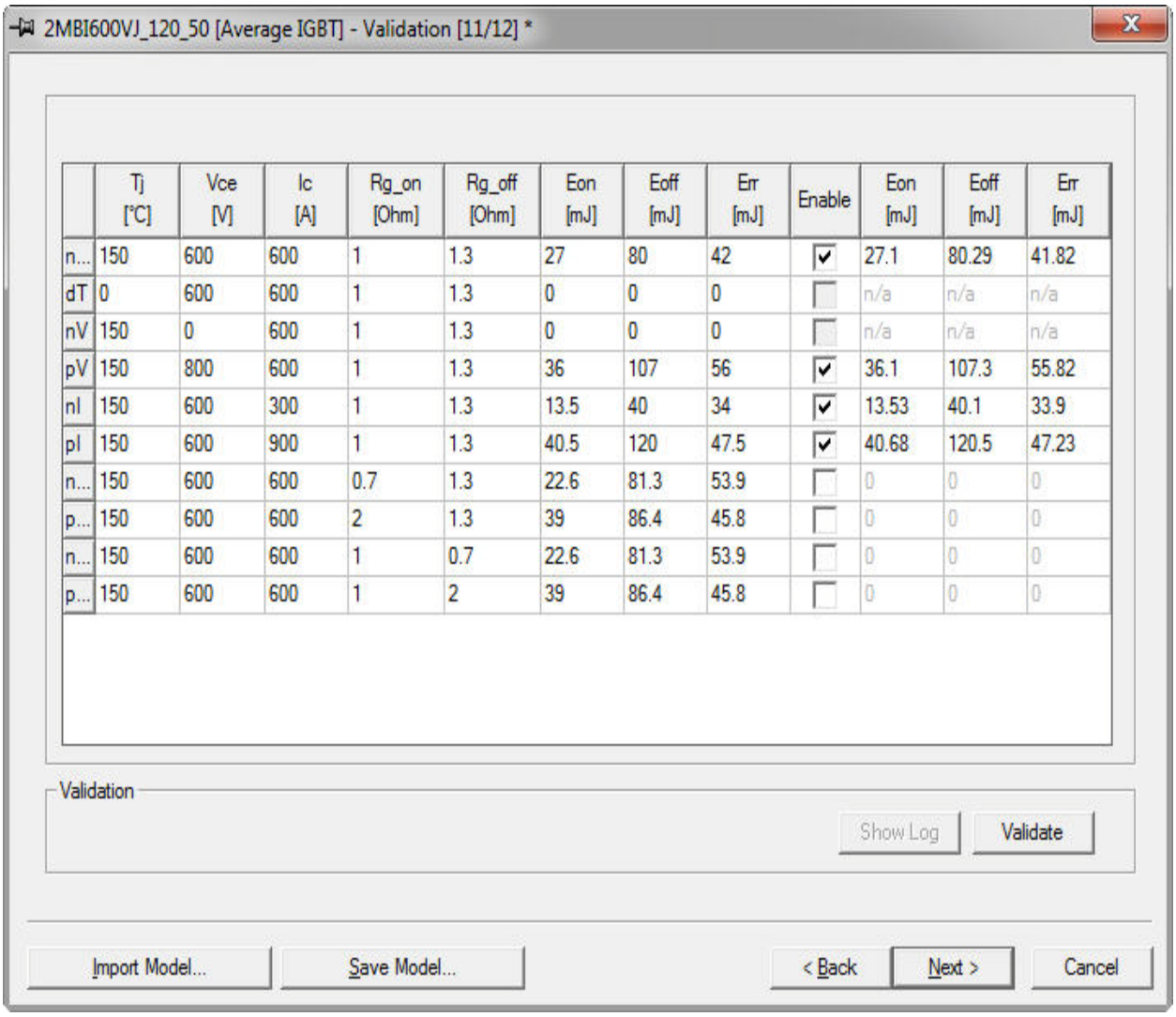
2. Click **Show Log** to view a copy of the simulation log showing both the **Data sheet value** and the **Simulation results** as well as the **Error** percentage between the two sets of data.
3. When finished, click **Next** to continue characterizing the device (see ***Model Parameters***).

Validation (Average IGBT and Average Power MOSFET)



In the Average IGBT **Validation** page, you can check the accuracy of the parameterized component through the validation of switching energies. All input values transfer from the energy fitting page. If the energy fitting for a particular set of working conditions was not run, this set of working conditions is not available for validation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



	Tj [°C]	Vce [V]	Ic [A]	Rg_on [Ohm]	Rg_off [Ohm]	Eon [mJ]	Eoff [mJ]	Err [mJ]	Enable	Eon [mJ]	Eoff [mJ]	Err [mJ]
n...	150	600	600	1	1.3	27	80	42	☑	27.1	80.29	41.82
dT	0	600	600	1	1.3	0	0	0	☐	n/a	n/a	n/a
nV	150	0	600	1	1.3	0	0	0	☐	n/a	n/a	n/a
pV	150	800	600	1	1.3	36	107	56	☑	36.1	107.3	55.82
nI	150	600	300	1	1.3	13.5	40	34	☑	13.53	40.1	33.9
pI	150	600	900	1	1.3	40.5	120	47.5	☑	40.68	120.5	47.23
n...	150	600	600	0.7	1.3	22.6	81.3	53.9	☐	0	0	0
p...	150	600	600	2	1.3	39	86.4	45.8	☐	0	0	0
n...	150	600	600	1	0.7	22.6	81.3	53.9	☐	0	0	0
p...	150	600	600	1	2	39	86.4	45.8	☐	0	0	0

To validate the model:

1. Click **Validate** to use the model with the parameters entered and to calculate the energies in a test configuration.

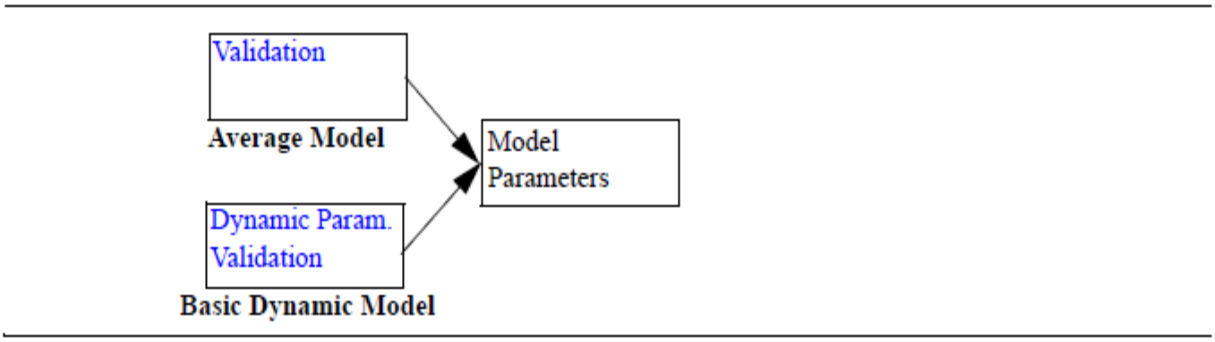
   Entries on the left-hand side of the **Enable** check box transfer from the previous (Energy fitting) page, and the entries to the right of the check box populate after the validation process ends. The validation is performed through a series of transient analysis under the specified working conditions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Select or clear the **Enable** check box to enable or disable the validation process for any row in the table.

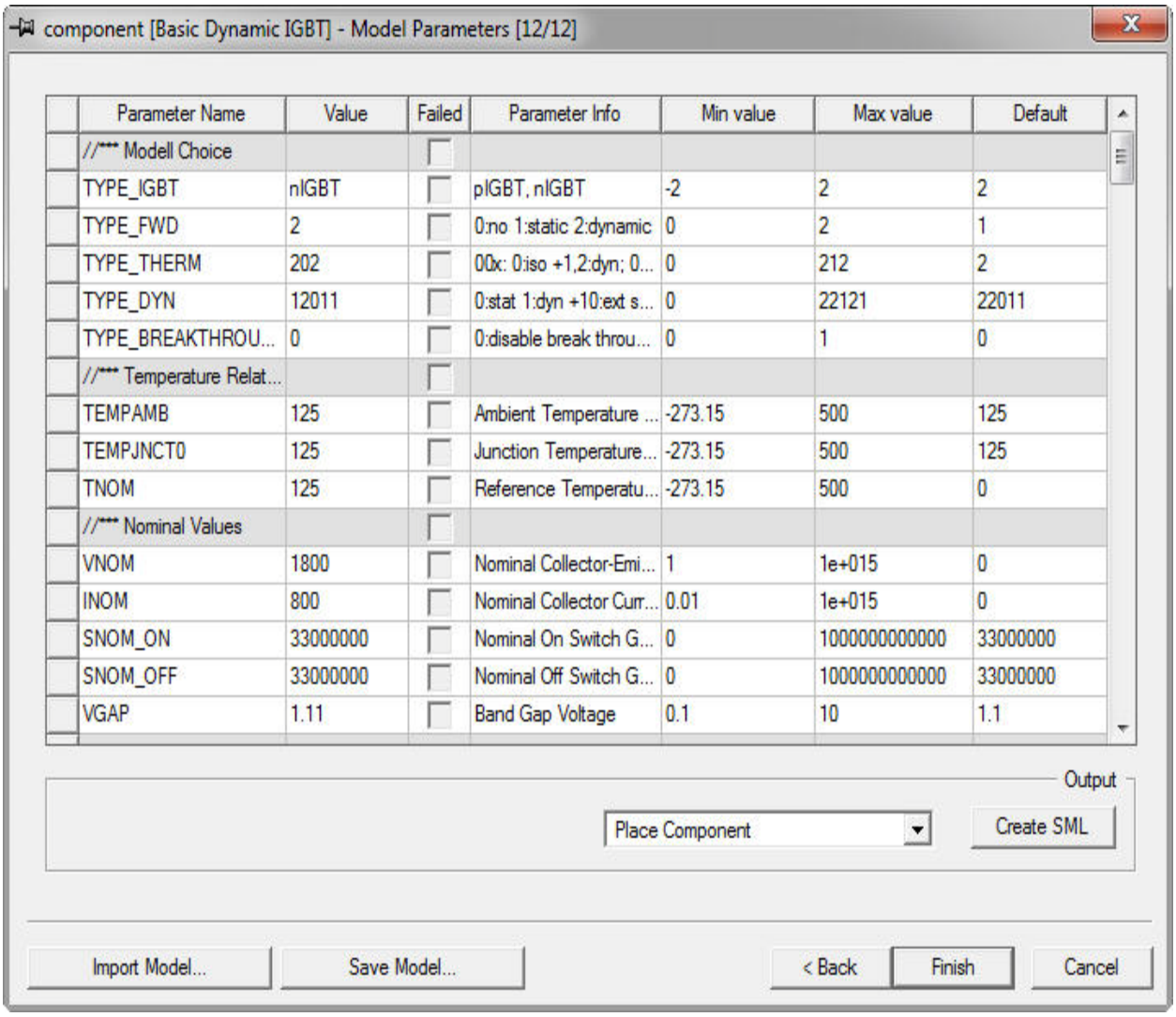
2. Click **Show Log** to view a copy of the simulation log showing both the **Data sheet value** and the **Simulation results** as well as the error percentage between the two sets of data.
3. When finished, click **Next** to continue characterizing the device (Model Parameters).

Model Parameters

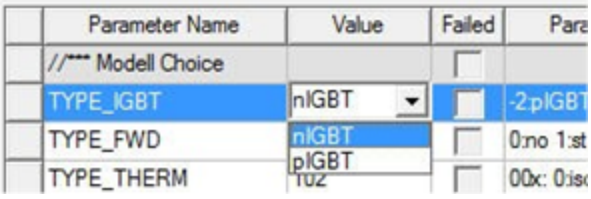


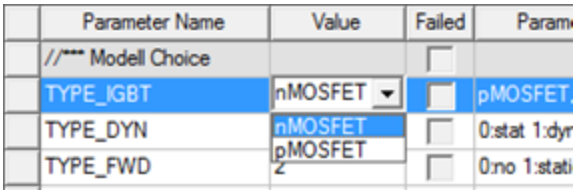
The **Model Parameters** page shows the extracted values of the parameters, their allowed range, default values, and a brief description.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Select the N- or P-type of an IGBT or Power MOSFET. By default, an N-type IGBT component is created. To create a P-type IGBT, click the value field of the **TYPE_IGBT** line and switch the value to **pIGBT** or **pMOSFET**.





Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You cannot change this setting after the characterization is finished and the new component is created.

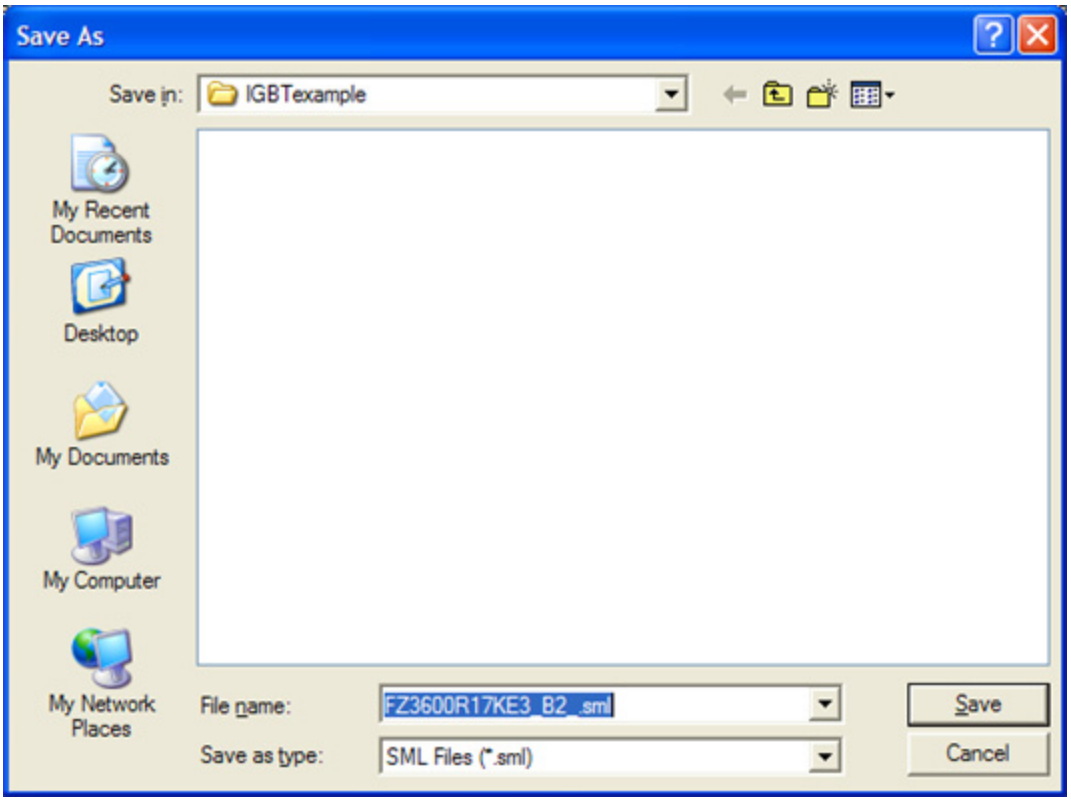
To complete the device characterization process:

1. To complete the characterization process and place the model in the schematic:
   a. Click **Finish**. The **Model Parameters** dialog box closes.
   b. Click and drag the component to a location on the schematic.
   c. Click the mouse at the desired location to place the component.
   d. Press **Esc** to exit placement mode.

   The component has now been successfully characterized and placed on the schematic. A copy of the component definition is available in the **Components** tab of the **Component Libraries** window.

2. To complete the characterization process and create a test circuit for the characterized component:
   a. Select a type of test circuit in the combo box. The test circuits that are available for a particular component depend on the type of the component and its configuration.
   b. Click **Finish**. The **Model Parameters** dialog box closes, and the test circuit with the characterized component appears in a new project window.
3. To complete the characterization process and create a stand-alone **SML** model of the device:
   a. Click **Create SML** to create a parameterized IGBT or Power MOSFET component. The **Save As** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



b. The component name specified in the **Component Information** dialog box is the file name.

c. Browse to the location you want to store the component model and click **Save**. A stand-alone SML model is now available for import or to distribute.

4. Click **Finish** to complete the **Characterize Device** wizard and place the component on the schematic, or click **Cancel** to complete the wizard without placing the component.

Adding/Deleting Characteristic Data Curves

You can use multiple characteristic data curves to create a model for an IGBT. Use the buttons at the top right of the dialog box to add or delete entire curves, or to plot the curves currently available.

	Display the **Choose Characteristics** dialog box for selecting and plotting the current characteristic curves.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	Add a new characteristic tab.
	Delete the currently selected characteristic tab and all associated data.

Related Topics

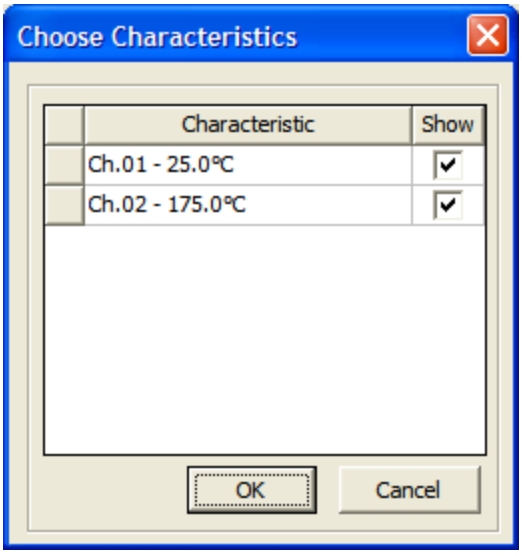
Transfer Characteristics

Output Characteristic

Freewheeling Diode Characteristic

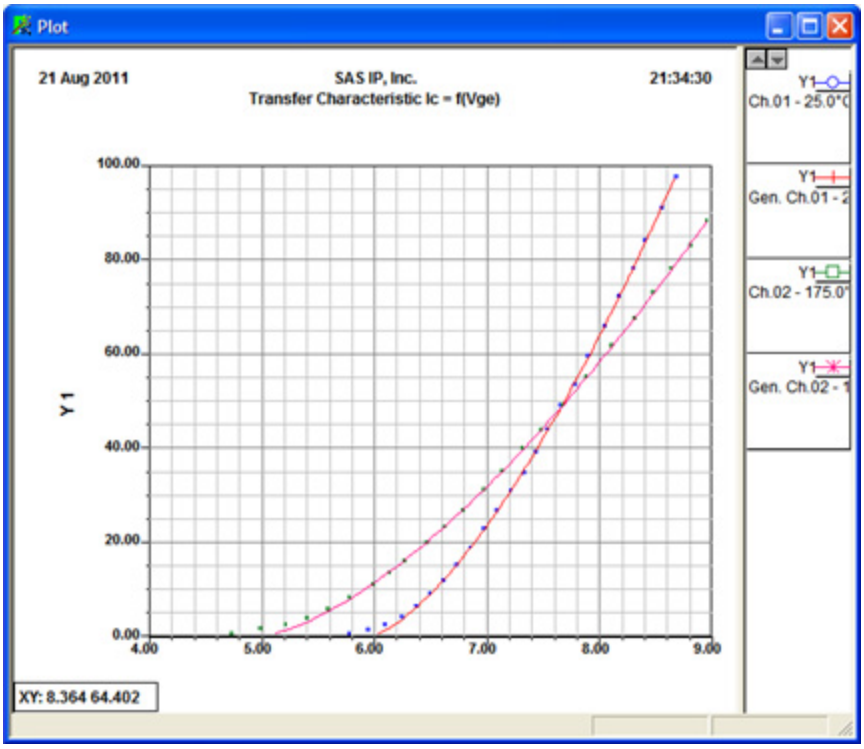
Plotting Characteristics Data

1. Click **Show Results** on the Transfer, Output, or FWD wizard pages. The **Choose Characteristics** dialog box appears.
2. Use the check box in the **Show** column to select or exclude the characteristic data to display for any curve.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Click **OK** to create a plot using the selected curves.

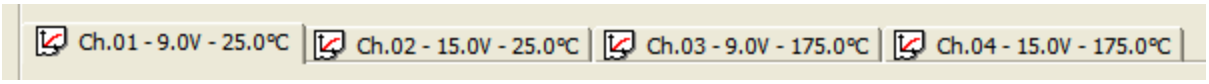


Selecting a Characteristic Curve

Use the tabs at the top of the **Transfer Characteristics**, **Output Characteristic**, and **Freewheeling Diode Characteristics** dialog boxes to switch between individual characteristic curves to review the data or modify data in a specific curve as necessary for the fitting process.

Each curve is assigned a numerical index based on the order it was added to the model. The numerical index along with the temperature associated with the curve, and the $\mathbf{V_{ge}}$ value for **Output Characteristic**, are listed in the tabs to uniquely identify the curve.

Click a tab to load the curve of interest into the table and make it available for editing.



Related Topics

Transfer Characteristics

Output Characteristic

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

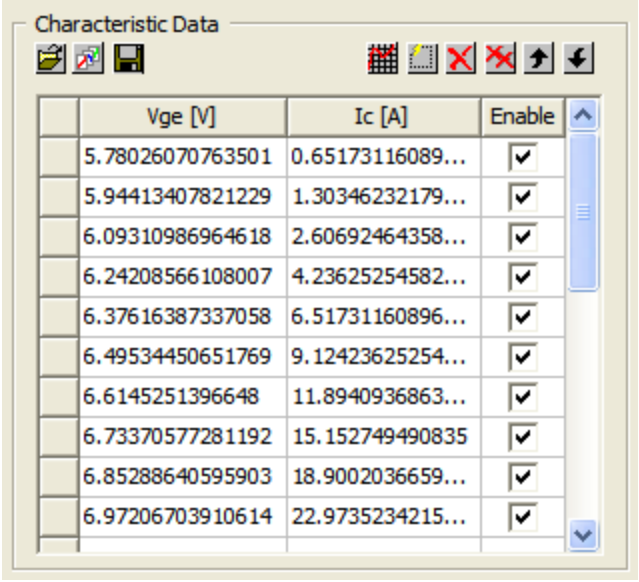
Freewheeling Diode Characteristic

Working with Characteristic Curve Data

Working with Characteristic Data Curves

Enter data into the **Characteristic Data** section of the dialog box to describe the performance of the device representing either the **Transfer Characteristic**, **Output Characteristic**, or **Freewheeling Diode Characteristic**. In addition, you can manipulate the data by changing specific data points, adding new data, or excluding data from consideration when performing curve fitting and model parameter extraction.

> **Note:**
>
> The information in this section on working with data curves also applies to the **Transient Thermal Impedance** section in the **IGBT Thermal Model** dialog box.



You can open, save, or import characteristic curves using the buttons listed in the following table.

	Open an existing characteristic data file (.**mdx** file).
	Open the **Dataset Manager** dialog box to allow manual data point entry, data entry via the sheetscan tool, or import of a dataset from a file.
	Save the characteristic data to a file (.**mdx** file).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To manipulate data in the curve:

- Click any data cell to make the cell editable, and change the data for that entry in the table.
- Select the check box next to any data point in the table to **Enable/Disable** the data point during the fitting.
- Click one of the buttons described in the following table.

	Plot the selected characteristic curve.
	Create a new row of data for the selected characteristic.
	Delete the selected row of data.
	Delete all the data for the selected characteristic.
	Move the selected row of data upward.
	Move the selected row of data downward.

Related Topics

Transfer Characteristic

Output Characteristic

Freewheeling Diode Characteristic

IGBT Thermal Model

Fitting Data to the Model

Use the **Fitting** area to fit the model parameters to the entered data.

Related Topics

Validate

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Show Result

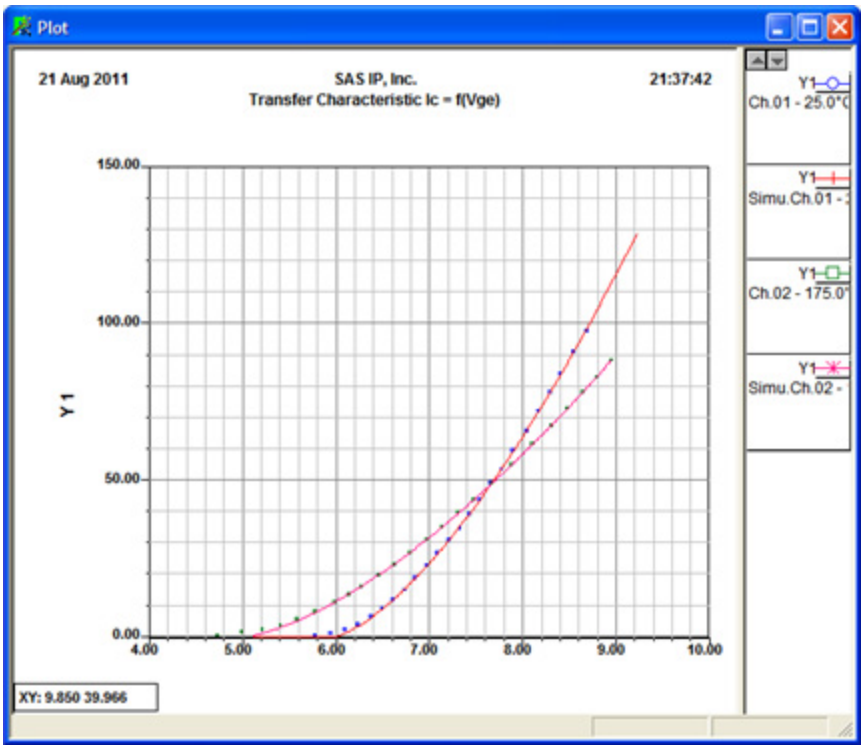
Show Log

Start Fitting

Validate

To validate the extracted parameters, place the parameterized IGBT model in the same testing circuit configuration used by the manufacturer. Simulation results appear in the same plot as the measurement data for easy comparison. To perform a validation:

1. Click **Validate** after performing a fitting function. The **Choose Characteristics** dialog box appears with the characteristics used for the fitting process selected for display.
2. In the **Choose Characteristics** dialog box, choose any additional characteristics for plotting by selecting or clearing the **Show** check box next to the characteristic name.
3. Click **OK** to dismiss the dialog box and plot the results of the validation as illustrated below.



In the legend, traces started with **Simu** are simulation results, and the others are measurement data.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

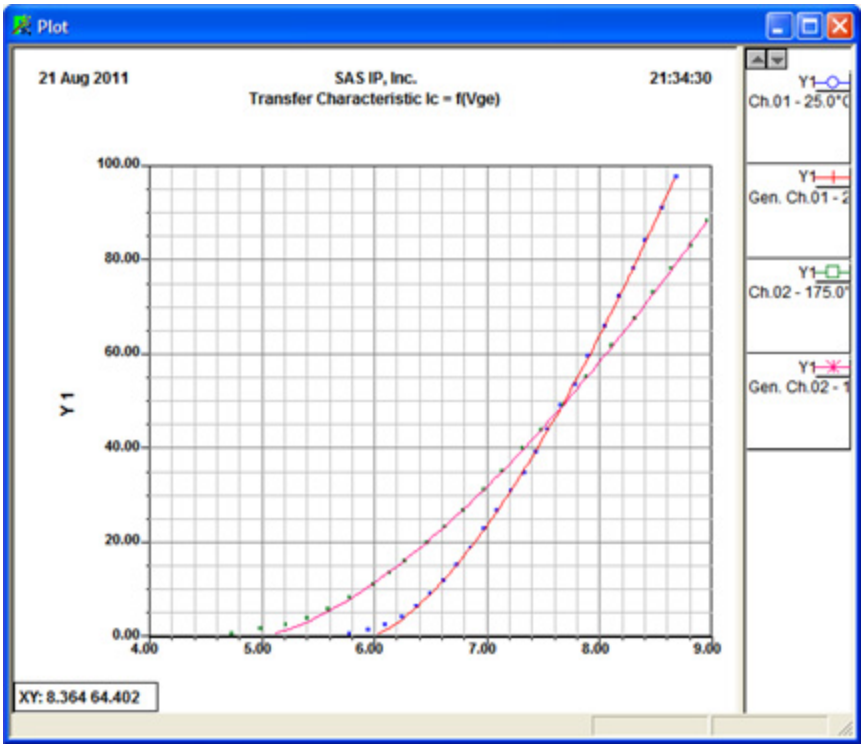
Transfer Characteristic

Output Characteristic

Freewheeling Diode Characteristic

Show Result

Use the **Show Results** functionality to visually compare the measurement and simulation data. The results plotted are the data input from measurement, and the results generated using the fitted polynomials from the fitting process. The plot is identical to the plot shown after completing the fitting process.



Related Topics

Transfer Characteristic

Output Characteristic

Freewheeling Diode Characteristic

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

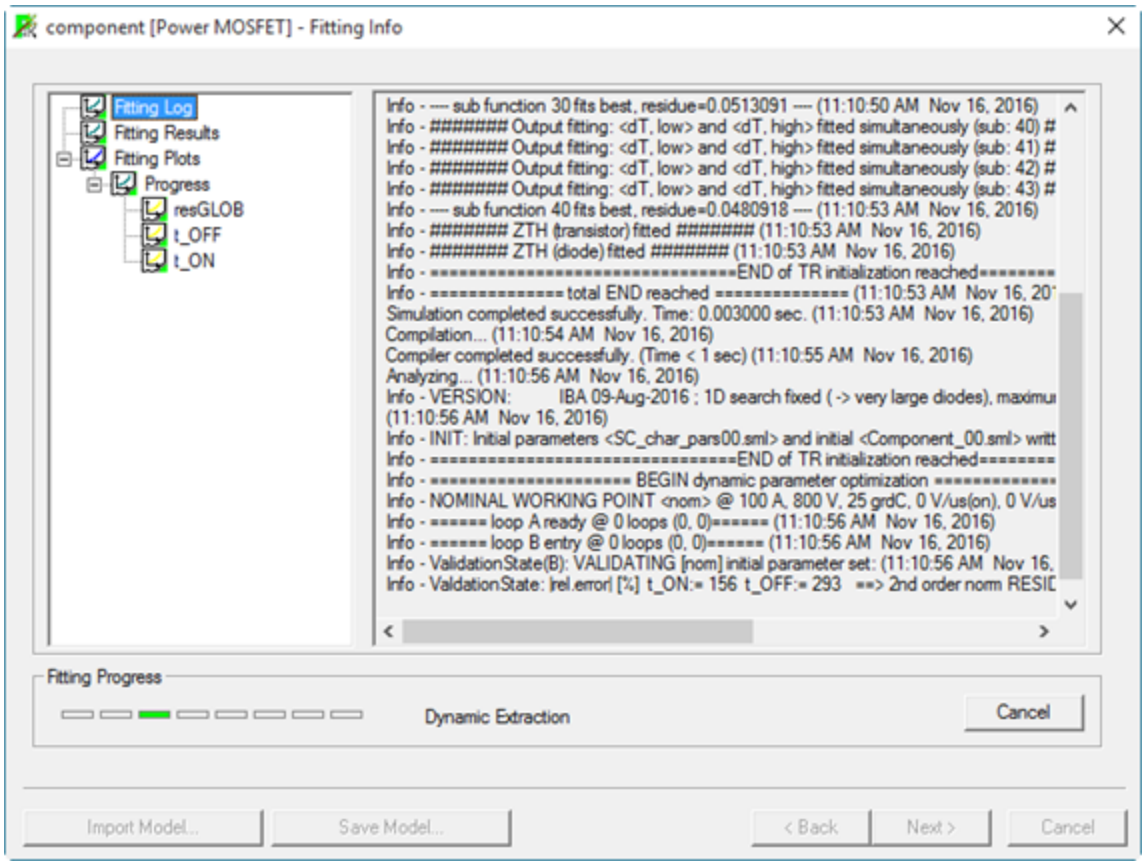
Show Log

Show Log is enabled after the curving fitting or extraction process is finished. Click it to open the **Fitting Info** page where information about the fitting/extraction process, the final fitted parameter values as well as the accuracy information, and curves of the fitted characteristics display.

Fitting Info

The **Fitting Info** page displays information during the extraction process and results after the fitting or extraction process ends.

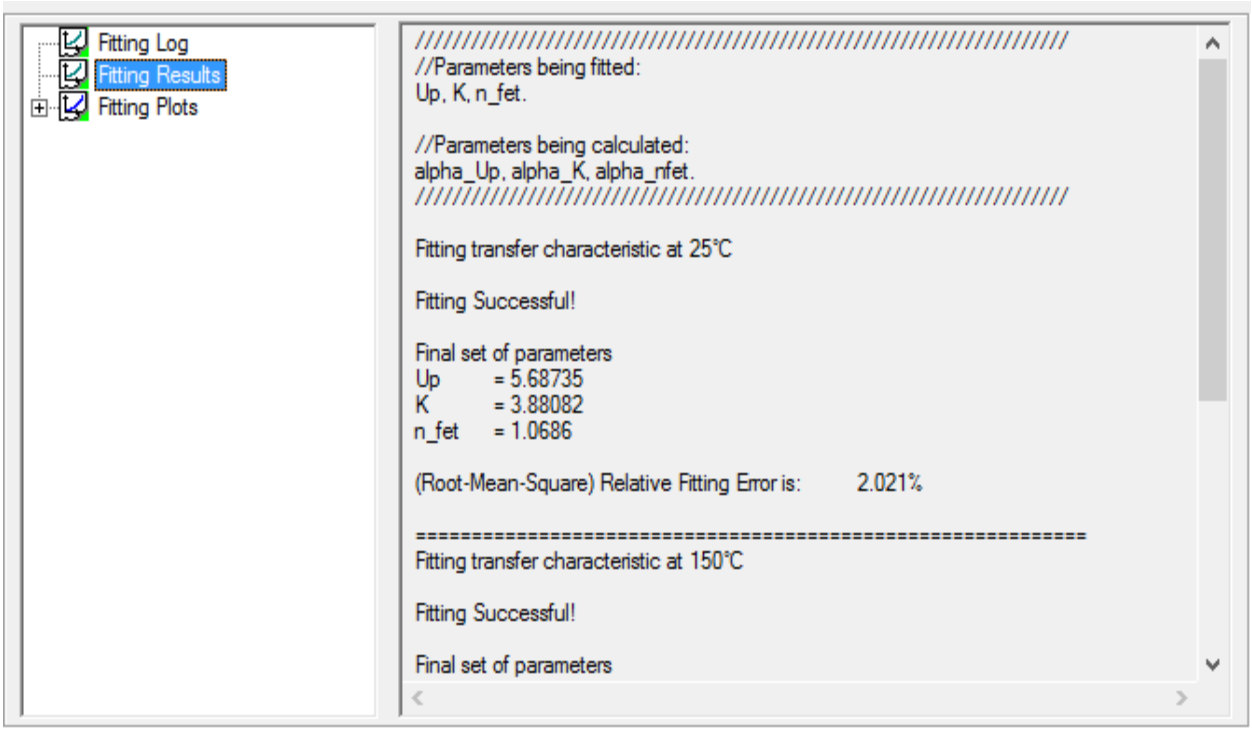
- **Fitting Log** – Displays log messages during the fitting or extraction process.



- **Fitting results** – Select **Fitting Results** to display the fitting status (**Fitting Successful** or **Fitting Failure**), the model parameters that were fitted for this type of characteristic, and the values for each parameter. In this example, the transfer characteristic data at both nominal working temperature (**Tj=25°C**) and a different working temperature (**Tj=150°C**) are fitted to derive the values of the set of parameters, which determines the transfer characteristics of the IGBT model, and the working point correction coefficients. If only the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

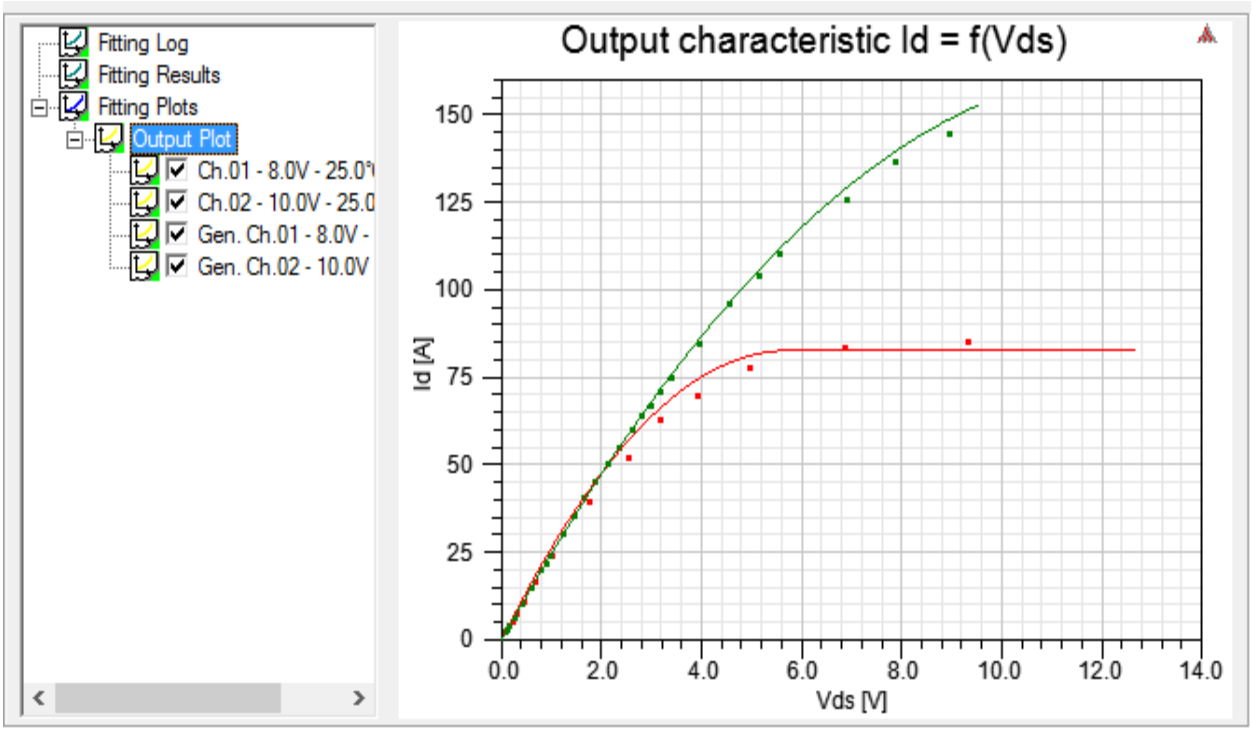
characteristic data at the nominal temperature are provided, the working point correction coefficients will be set to **0.0**.



- **Fitting Plots** – Select **Fitting Plots** to display the plots of the fitted characteristics for static fittings or the plots of goal value development and global residue development during dynamic parameter extraction.

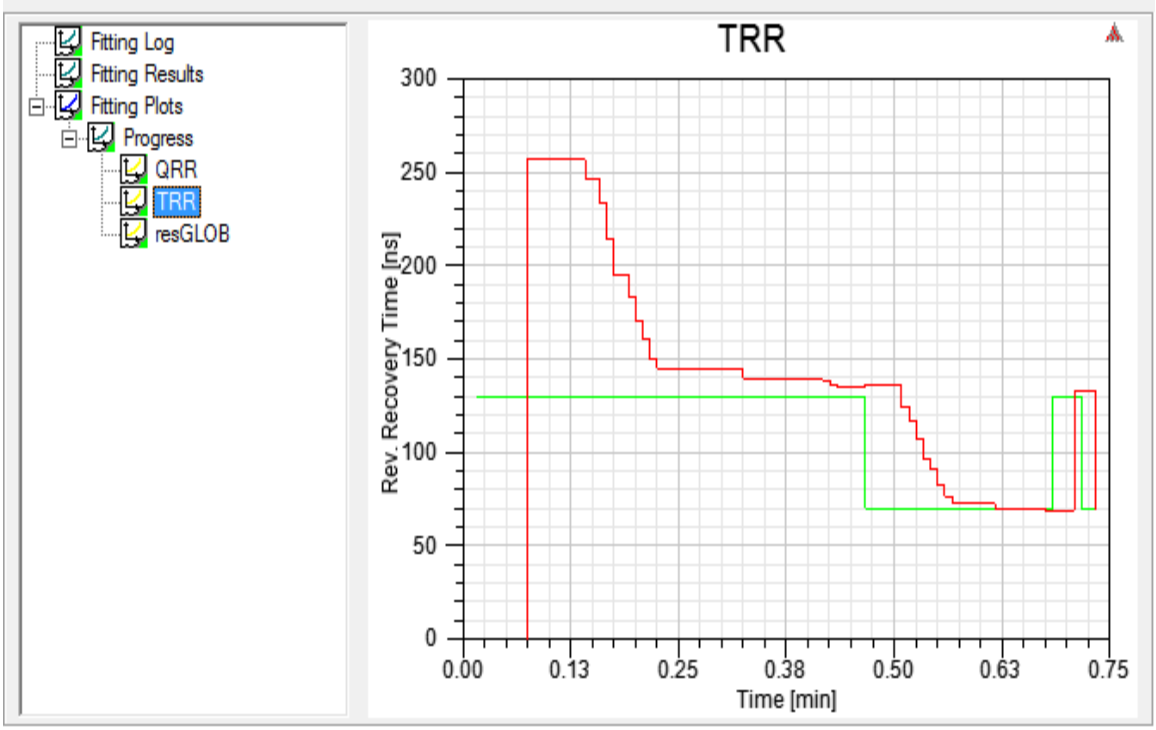
  The plots of the static fittings show a comparison between the characteristic input data (dotted) and the curve generated by the model using the extracted model parameters (continuous).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The plots of the goal value development during dynamic parameter extraction are useful to assess whether the system is converging to a solution. In the case that the solution is not converging, the dynamic extraction can be canceled. Make changes to the **Advanced Extraction Settings** to reach a better convergence.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Related Topics

Transfer Characteristic

Output Characteristic

Freewheeling Diode Characteristic

Start Fitting

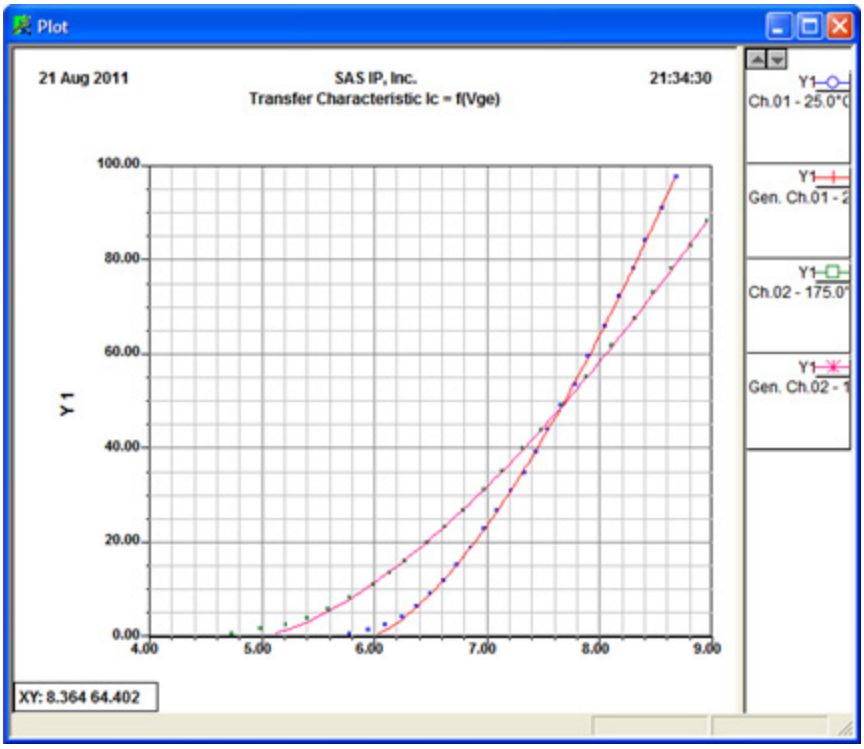
The fitting process involves adjusting a set of standard parameters to the selected curves. Once the parameters are adjusted to fit to the characteristic curves, the parameters become the basis for the IGBT model. The curves selected in the **Fitting Characteristic Order** section of the dialog box are used for the fitting process.

1. Click **Start Fitting** to perform curve fitting. An information dialog box appears when fitting is complete.
2. Click **OK** to dismiss the **Fitting Complete** dialog box.
3. The **Choose Characteristics** dialog box appears with the fitted curve pre-selected.
4. Click **OK** to dismiss the dialog box and display the fitted curves as shown below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The characteristics data both from the measurement and the calculation appear in the same plot for comparison.

In the legend, traces started with **Gen** are calculated results using the derived polynomials, and the others are measurement data.



Related Topics

Transfer Characteristic

Output Characteristic

Freewheeling Diode Characteristic

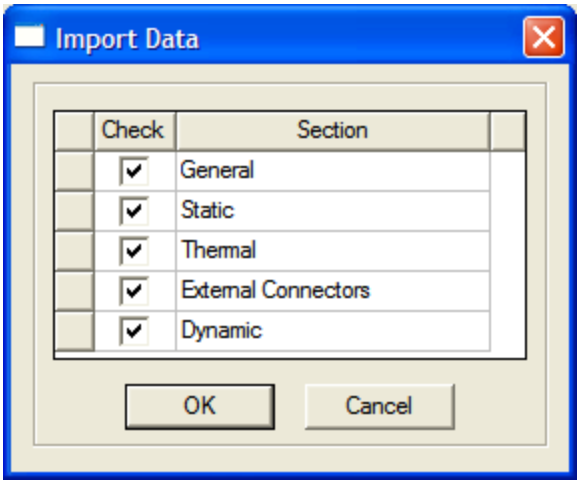
Import Model

Follow this procedure to import data from an existing parameterization wizard file.

1. Click **Import Model**.
2. Use the file browser to locate and open the **.ppm** file with the desired data.

   The **Import Data** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Section items in the dialog box depend upon the type of device model and the data available in the selected file.

3. Select one or more check boxes in the **Check** column to indicate what data is to be imported.
4. Click **OK** to import the data or **Cancel** to exit without importing data.

Related Topics

Save Model

Save Model

Follow this procedure to save the state of the Device Characterization wizard at any point:

1. Click **Save Model**. The **Save As** dialog box appears.
2. Enter a **File Name** for the saved parameterization file and click **Save**.

You can recover the parameterization at a later time. Use the **Continue Device Characterization** option in the **Characterize Device** dialog box, or click **Import Model** to import sections of data.

Close the Device Characterization Wizard

To stop the **Device Characterization Wizard**:

1. Click **Cancel**.
2. Click **Yes** in the confirmation dialog box to **Abort the Changes** and close the wizard. Click **No** to return to the wizard.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Tutorial for Characterizing a Basic Dynamic IGBT

Use the IGBT Device Characterization Wizard to parameterize a Basic Dynamic IGBT model to match both the DC and dynamic characteristics of a device using information from the manufacturer's data sheet. The information available in the data sheet limits the accuracy of parameterized IGBT model. For example, if the data sheet does not provide a device's transfer characteristics, the accuracy of the gate-voltage dependence of the output will be reduced.

The wizard characterizes a wide array of IGBTs. The available information can vary from vendor to vendor, and even between devices from the same vendor. The following procedures include vendor-specific information for help in characterizing models for several vendors' devices. If you have any additional questions, contact Ansys support.

Related Topics

Preliminary Considerations Using Data Sheet Information

Using the Device Characterization Wizard for a Basic Dynamic IGBT

Advanced Settings for Characterizing a Basic Dynamic IGBT

Preliminary Considerations using Basic Dynamic IGBT Data Sheet Information

Before starting a device characterization, study the manufacturer's data sheet for the IGBT you want to characterize. Each manufacturer has a unique design and naming convention for their data sheets. Different manufacturers also measure device characteristics differently.

- Determine the nominal conditions for the device. Ideally, these are the conditions at which the device will be operated. These conditions should be those for which you have the most information closest to the desired operating conditions. Search the data sheet and look at all available data to decide on the nominal point. The measurement conditions for the dynamic characteristics (**Eon**, **Eoff**, **td(on)**, and **td(off)**) are generally good choices.
- DC characteristics are parameterized using the transfer and output characteristics, which can be found as plots in the sheet.
- Dynamic behavior is characterized using the switching energies and times, which are found either in a table or read from a plot.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> The IGBT wizard documentation includes information and recommended settings for several manufacturers. Make sure to consult these sections in the documentation when characterizing a device.

Related Topics

Using the Device Characterization Wizard for a Basic Dynamic IGBT

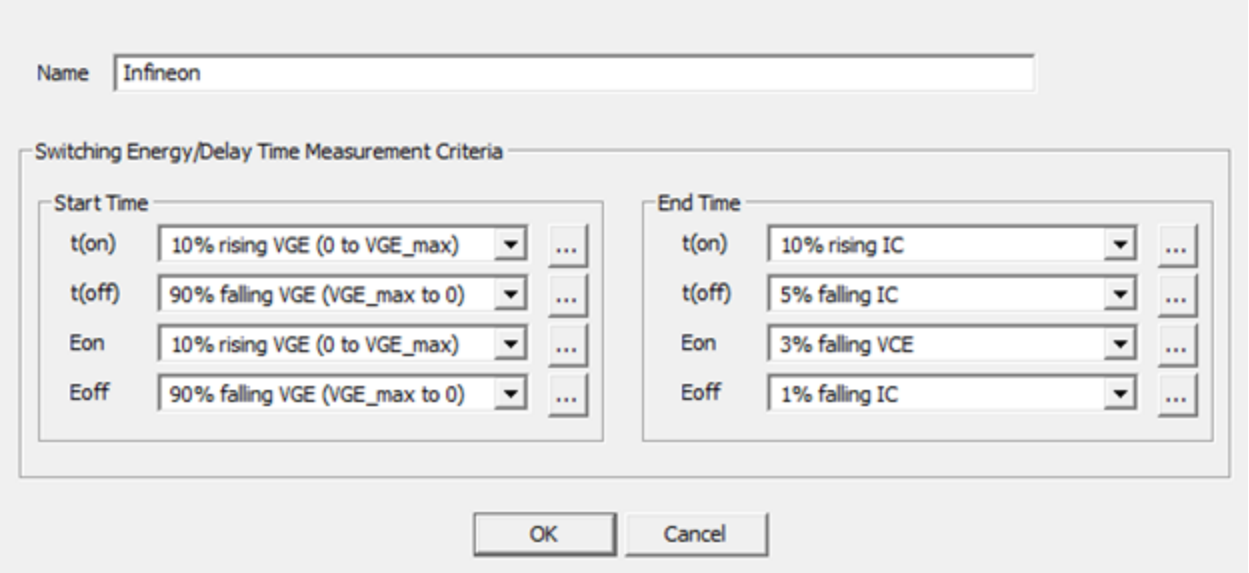
Using the Device Characterization Wizard for a Basic Dynamic IGBT

The following procedures use an Infineon device as an example. The process is similar for other manufacturer's devices.

> **Note:**
>
> At any time during the characterization process, click **Save Model** to save your progress.
>
> You can load a characterization with the **Import Model** tab.

1. In the **Component Information [1/12]** dialog box, enter the component name and define the manufacturer data. The detailed documentation describes some specific settings for several manufacturers. These are the measurement conditions for the dynamic characteristics. For example, Infineon/Eupec uses the following settings:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. The **Nominal Working Point Values [2/12]** dialog box sets nominal values at the device's working point. As mentioned above, ideally, these are the operating conditions for the device in the design. Practically, these values should be the ones on the manufacturer data sheet with the most information closest to the operating conditions. These are the conditions for which the parameterized device will best match the measured characteristics. For Infineon/Eupec, generally pick the measurement conditions for the dynamic characteristics. Pick the values of **VCE** and **IC** mentioned in the table next to the value of the turn-on energy, **Eon**. Select the temperature for which you have the output characteristics in the plots section. **VGE_on** and **VGE_off** are gate voltages used in manufacturer's test circuit in which switching dynamics are measured. In the datasheet, **VGE_on** and **VGE_off** are next to the value of **Eon**. **Cin**, the input capacitance, is in the datasheet. **Cr**, the Feedback (Miller) capacitance can be found in the datasheet, it is sometimes referred to as reverse transfer capacitance. If the values cannot be found in the datasheet, **0** is a good default value, as the characterization tool will then pick a value using the other device characteristics. Click **Next** to continue.
3. On the **Breakthrough Values [3/12]** dialog box, select the **Disable Breakthrough Model** check box, as the breakdown characteristics data is rarely available in a data sheet. Click **Next** to continue.
4. The data needed for the **Half-Bridge Test Circuit Conditions [4/12]** dialog box is in the tabular section of the data sheet.

   For example, for an Infineon/Eupec device:

   - **Rg** is the internal gate resistance found in the datasheet. This value is used as an initial value during the dynamic characterization, rather than the internal gate resistance the characterized device ends up with.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **R_tot** is given by **R_cc_ee**. If the value is given for the module instead of the single switch, divide it by two. If the value is not on the data sheet, **0** is a good default value.
- **Ltot**, is given by **Lsce**. If the value is given for the module instead of the single switch, divide it by two. If the value is not on the data sheet, **0** is a good default value.
- **L_extern** can be given by **Ls** in the test conditions for **Eon**. If the value is not there, 30 nH is a good value to use. The characterization wizard will parameterize this value to find a good fit with the dynamic data.
- **Rg_on** and **Rg_off** are next to the values of the switching energies **Eoff** and **Eon**.
- **Cge** is generally omitted and a default value of 1e-12F can be used.
- **Cload** is sometimes found on the datasheet as load capacitance. Use **0** if it is missing.
- When finished setting values, click **Next** to continue.

5. Use the **Transfer Characteristic [5/12]** dialog box to parameterize the threshold voltages and transconductance of the IGBT. Enter the data with SheetScan. To do so, use the **Load characteristics from Dataset Manager** ( ) above the table and click **SheetScan**. Load the graphs in graphic files captured from the manufacturer data sheet using the **Picture** > **Load picture** command.
   - For an Infineon/Eupec device, the data sheet plot shows **IC=f(VGE)**.
   a. Select a coordinate system (**Coordinate System** > **New**) and define it on the plot. To do so, you must select three points. Typically, you should use the bottom-right, bottom-left and top-right points of the plot grid. Click **Point1**, **Point2** and **Point3** and select the corresponding points on the graph. Enter the corresponding X-Values and Y-Values for these points (read from the plot's X-axis and Y-axis labels) in the table and click **OK** to finish defining the coordinate system.
   b. To define the first characteristic, select **Curve** > **New** and give names to the X-axis and Y-axis in the **Curve Settings** dialog box. Click **OK** when finished.
   c. Make sure to note the temperature given on the plot. Then select several points on the curve starting with the lowest X-value. Pick at least 4 points. When done, select **File** > **Export**, then click **Dataset** in the resulting **Save** dialog box.
   d. If a plot at a different temperature is available, repeat steps **a**, **b**, and **c** to record additional data.
   e. When finished, click **File** > **Exit** to exit the SheetScan tool, saving the scan setup information if desired.
   f. In the **Datasets** dialog box, select the data you want to use at the nominal temperature and click **Done**. The data is transferred to the **Characteristic Data** table in the **Transfer Characteristic [5/12]** dialog box. Make sure to enter the correct values for temperature and Vce, which you recorded during the SheetScan measurements.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

g. If you also recorded plot data for a different temperature, click **Add new characteristic** ( ) in the top right corner, then click **Load characteristics from Dataset Manager** ( ) above the table to load the additional data. Select the plot in the first tab for the **Nominal Temperature** field of the **Fitting Characteristic Order** panel. If you added a second set of data for a different temperature, you can select this plot for the **Different Temperature** field, or select **Not Used**, if data is available only for one temperature.

h. Click **Start Fitting** to fit the characteristics, then examine the resulting plot to check the match of the fit. Click **Next** to continue.

6. Use the **Output Characteristic [6/12]** dialog box to parameterize more of the IGBT's DC characteristics. Ideally, enter the output characteristics at **Vge=VGE_on** (**Full Saturated Branch**) and for Vge at a lower voltage (**Semi Saturated Branch**), for both the nominal temperature and at a different temperature using SheetScan as described in step **5**. Make sure to note the value of **Vge** and the corresponding temperature for each curve.

   - For Infineon/Eupec, the plot shows **IC=f(VCE)**. The output characteristics of the semi-saturated branch are often missing at a different temperature.

   a. Add up to four characteristics, making sure to identify them with the correct values of **VGE** and temperature.

   b. Select the plot at the nominal temperature and with **Vge=VGE_on** for the **Full Saturated Branch (Tnom)** field in the **Fitting Characteristic Order** panel, and one at a lower value of **Vge** for the **Semi Saturated Branch (Tnom)** field, or select **Not Used**, if the data is not available. Repeat this at a different temperature for **Vge=VGE_on** and at a lower Vge in the corresponding **Full Saturated Branch (Tdiff)** and **Semi Saturated Branch (Tdiff)** fields. The fit will be better if the current does not saturate much for the semi-saturated curve, but the fitting algorithm will work with either option.

   c. Click **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

7. Use the **Freewheeling Diode Characteristic [7/12]** dialog box to parameterize the DC characteristics of the diode. Ideally, you should enter the diode forward characteristic If (Vf) at both the nominal temperature and at a different temperature using SheetScan as outlined in step **5**. Make sure to note the value of the temperature for each curve.

   - For Infineon/Eupec, the data sheet plot shows **IF=f(VF)**.

   a. Generate two characteristic data sets, one at the nominal temperature and one at a different temperature, being sure to identify them with their corresponding temperatures.

   b. Select the plot at the nominal temperature in the **Nominal Temperature** field of the **Fitting Characteristic Order** panel, and the plot at the different temperature in the **Different Temperature** field, or select **Not Used**, if the data is not available.

   c. Click **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

8. Use the **IGBT Thermal Model [8/12]** dialog box to parameterize the thermal impedance of the IGBT. Do one of the following:
   - If the data sheet provides extracted values for **ri** and **ti**, which is the case for Infineon/Eupec, enter these four value pairs in the **Continued/Partial Fraction Coefficients** table, click **Start Fitting**, check the plot, and click **Next** to continue.
   - If the data sheet does not provide extracted values for **ri** and **ti**, select **Use Transient Thermal Impedance** and enter the plot data for the thermal characteristics using SheetScan per the instructions given in step **5**. The plot shows the impedance as a function of time. Make sure to adjust the scale of the coordinate system in SheetScan to logarithmic if needed. Select **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

   > **Note:**
   >
   > Sample the "single pulse" curve when multiple transient thermal impedance curves are presented in the data sheet.

9. Use the **Freewheeling Diode Thermal Model [9/12]** dialog box to parameterize the thermal impedance of the diode.
   - If the data sheet provides extracted values for **ri** and **ti**, which is the case for Infineon/Eupec, enter these four value pairs in the **Continued/Partial Fraction Coefficients** table, click **Start Fitting**, check the plot, and click **Next** to continue.
   - If the data sheet does not provide extracted values for **ri** and **ti**, select **Transient Thermal Impedance** and enter the plot data for the thermal characteristics using SheetScan per the instructions given in step **5**. The plot shows the impedance as a function of time. Make sure to adjust the scale of the coordinate system in SheetScan to logarithmic if needed. Select **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.
10. Use the **Dynamic Model Input [10/12]** dialog box to parameterize the dynamic characteristics of the IGBT. You can select to fit for the switching energies (**Eon** and **Eoff**) and switching times (**T(on)** and **T(off)**). The nominal point must be fit, so the data must be available. The **Advanced Settings** button allows for more control over the characterization process, but should not be necessary to change for most devices. Check the Advanced Settings documentation for more information.
    a. Click **Measurement** to open the **Measurement Data** dialog box and make sure the settings correspond to the ones for the manufacturer of the device being characterized.
    b. If data is available at different temperature, add it to the **dT** row.
    c. If data is available at a different value of VCE, add the lower value of VCE to the **nV** row and the higher value of VCE to the **pV** row.
    d. If data is available at a different value of Ic, add the lower value of Ic to the **nI** row and

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the higher value of Ic to the **pl** row.

e. Click **Extraction** to start the fit and click **Next** to continue.

Advanced Settings

- **TR** and **TF** are the rise and fall time from the data sheet. These should be set to **0** if one is uncertain, as their value can affect the convergence.
- There are three loops. The first loop uses a 1D search method to find a better solution. To enable it, set **LOOPS_A** to an integer number larger than 0. **MASKPAR_A** contains the names of the parameters changed by the characterization tool to find a better fit. **RESORD_A** sets how the residue is defined: **0** for the maximum error, **1** for the average error, and **2** for the root mean square error. **RESTOL_A** defines the value under which the residue must get to leave the loop with a good solution.
- The second and third loops (B and C) use a Jacobian matrix method. **LOOPS_B** and **LOOPS_C** set the number of pre-loops which are taken for the second and third loop; usually one should suffice. **MASKPAR_B** and **MASKPAR_C** contain the names of the parameters changed by the characterization tool to find a better fit during the second and third loop. **MATRIX_B** and **MATRIX_C** set the maximum number of Jacobian recalculations, while **ZEROFN_B** and **ZEROFN_C** set the maximum number of constant Jacobian calculations. **RESLOC_B** and **RESLOC_C** set the maximum number of relaxation which occurs within a constant Jacobian calculation. **RESORD_B** and **RESORD_C** set how the residue is defined: **0** for the maximum error, **1** for the average error, and **2** for the root mean square error. **RESTOL_B** and **RESTOL_C** define the value under which the residue must get to leave the second and the third loop with a good solution; if it is zero, it gets this value from the worksheet.

11. Use the **Dynamic Parameter Validation [11/12]** dialog box to validate the dynamic extraction. The actual values of the switching times and energies for the parameterized device will be calculated. **Enable** the conditions that should be checked, then click **Validate**. Click **Next** to continue.
12. Use the **Model Parameters [12/12]** dialog box to browse the extracted parameters.
    a. To generate an **.sml** file of the model, click **Create SML**.
    b. Click **Finish** to close the dialog box. The characterized device is generated for you to place in the Twin Builder project.

Related Topics

Preliminary Considerations Using Data Sheet Information

Using SheetScan

Advanced Settings for Characterizing a Basic Dynamic IGBT

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Advanced Settings for Characterizing a Basic Dynamic IGBT

Click **Adv. Settings**in the **Dynamic Model Input** [10/12] dialog box to access the advanced settings of basic dynamic IGBT characterization. There are two tabs: **Extraction Settings** and **Model & Goal Settings**.

- Extraction Settings

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Name	Value	Parameter Info
H_ON	5e-06	Initial On-switch signal acquisition time (2*time from trigger to 1% falling VCE) [s]
H_OFF	1e-05	Initial Off-switch signal acquisition time (2*time from trigger to 1% falling IC) [s]
INTEGRATION	0	Integration method: 0:Euler, 1-Adaptive, 2-explicit Euler, 3-Trapezoid
MEASURE_FLAG	1	Measurement mode: 0:continuous 1:discontinuous
MEASURE_RE...	1	Measurement tool output: 0:none 1:waves and goal values 2:+convergence 3:+fitting
FITTINGREPORT	1	Fitting report output: 0:none 1:fitting check 2:fitting details
LOOPS_A	1	Number of pre-loops using 1D search
TOL_JACO_A	0	-1: simple 1D search; >=0: Matrix search - Relative Jacobian entry threshold for parameter change
NUM_JACO_A	10	Number of most important Jacobian entries considered for matrix search
MASKPAR_A	CR0 CR1 TAUFWD TAUTAIL ...	Set parameters for optimization, CIN0 CIN1 TAUFWD ...
MATRIX_A	3	Maximum number of Jacobean recalculations, matrix loops A (for TOL_JACO_A>0)
ZEROFN_A	15	Maximum number of constant Jacobean iterations, Samanskii factor (for TOL_JACO_A>0)
RESLOC_A	100	Maximum number of local relaxation within ZEROFN
RESORD_A	2	Order of 1D residue 0:MAX 1:AV 2:RMS
RESTOL_A	0.05	1D residue tolerance (0.1=10%)
LOOPS_B	1	Number of pre-loops using Jacobean matrix
TOL_JACO_B	0.1	Relative Jacobian entry threshold for parameter change
NUM_JACO_B	10	Number of most important Jacobian entries considered for matrix search
MASKPAR_B	CIN0 CIN1 CR0 DELTACR0 C...	Set parameters for optimization, CIN0 CIN1 TAUFWD ...
MATRIX_B	5	Maximum number of Jacobean recalculations, matrix loops B
ZEROFN_B	15	Maximum number of constant Jacobean iterations, Samanskii factor
RESLOC_B	1000	Maximum number of local relaxation within ZEROFN
RESORD_B	0	Residue order 0:MAX 1:AV 2:RMS
RESTOL_B	0.05	Loop B residue tolerance (0.1=10%, 0=take TOL from WP)
LOOPS_C	1	Number of main Jacobean matrix loops
TOL_JACO_C	0.01	Relative Jacobian entry threshold for parameter change
NUM_JACO_C	10	Number of most important Jacobian entries considered for matrix search
MASKPAR_C	CIN0 CIN1 CR0 DELTACR0 C...	Set parameters for optimization, CIN0 CIN1 TAUFWD ...
MATRIX_C	15	Maximum number of Jacobean recalculations, matrix loops C
ZEROFN_C	10	Maximum number of constant Jacobean iterations, Samanskii factor
RESLOC_C	1000	Maximum number of local relaxation within ZEROFN
RESORD_C	0	Residue order 0:MAX 1:AV 2:RMS
RESTOL_C	0	Loop C residue tolerance (0.1=10%, 0=take TOL from WP)

Figure 7-1 Advanced Parameter Setting Dialog Box

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

These settings give you more control over the characterization process, but it should not be necessary to change them for most devices.

Parameter	Information
H_ON and H_ OFF	Before preprocessing and interpolation of the on/off switch wave forms, a series of simulated data points must be stored temporarily. These parameters set the acquisition time for the data points. This time interval setting limits the amount of acquired data and must fit to the longest possible time constants at the on/off switch together with a remaining time buffer to allow the Jacobean entry calculation which sweeps every parameter in a wide range. The sensitivity analysis for the 1D search in loop A varies actual parameters from p-25% ... p+50% and the Jacobean-based method does a parameter variation in the range p-10% ... p+10%. This is repeated several times to calculate either the actual best fitting single parameter or a parameter to goal sensitivity coefficient. Thus, the transient wave form duration will vary in a wide range. **H_ON** and **H_OFF** are the initial settings for the On-switch and OFF-switch signal acquisition times. The actual **Hon** and **Hoff** values change during the characterization. **Hon** and **Hoff** derive from the measured switching times and length of the reverse recovery. 5x…10x of the On-switch/reverse recovery time should be a good value. For SiC devices the values should be much smaller than for Si devices. If the initial values are not good, the extraction process will not determine the sensitivity of parameter changes to the output signal and the extraction will not start to converge. You can observe the convergence of the extraction on the **Fitting** info page and adjust the initial values for **H_ON** and **H_OFF** if necessary.
INTEGRATION	Integration method (0 - best for switching devices).
MEASURE_ FLAG	Measurement mode (1 – allows for stop of measurement cycles as soon as good result is reached).
MEASURE_ REPORT	Measurement tool output (should stay 1).
FITTINGREPO RT	Fitting report output (should remain set to 1).
LOOPS_A, **LOOPS_B**, and **LOOPS_C**	Approaching the optimal parameter set is an iterative procedure. This iteration is split into three sequenced loops: A, B and C. Each of the sequences can be repeated any number of times according to **LOOPS_***x*. This is especially helpful in the case of loop A, which is a predefined sequence of one dimensional parameter sweeps. In some special cases, an increased number **LOOPS_A** can result in much better initial parameter values before starting the Newton-Raphson-based matrix loops B and C. But the repeated 1D parameter sweep will take time and a global

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>convergence is not guaranteed. In most cases the initial values calculated internally from the nominal conditions are good enough to start loop B immediately without time-consuming loop A.

Default values are LOOPS_A:=0, LOOPS_B:=1, and LOOPS_C:=1.</td></tr>
<tr><td>MASK_PAR_A</td><td>This is a list of dynamic model parameters which take part in the 1D parameter sweep sequence. The input is interpreted as a parameter mask for parameter selection during the characterization. Parameter names in bold mark the values included by default.

<table>
<tr><th>Parameter</th><th>Observed impact to goals</th></tr>
<tr><td>CIN0</td><td>Eon, Eoff</td></tr>
<tr><td>CIN1</td><td>Ton</td></tr>
<tr><td>TAUS</td><td>Toff</td></tr>
<tr><td>CR0</td><td>Ton, Toff</td></tr>
<tr><td>CR1</td><td>Ton, Toff</td></tr>
<tr><td>TAUFWD</td><td>Eon</td></tr>
<tr><td>DAMPING</td><td>Eoff</td></tr>
<tr><td>RG</td><td>Ton</td></tr>
<tr><td>DELTATAIL</td><td>Toff, Eoff</td></tr>
<tr><td>TAUTAIL</td><td>Toff, Eoff</td></tr>
<tr><td>L_EXTERN</td><td>Eon, Eoff</td></tr>
</table>
A repeated application of loop A helps to find a good initial parameter guess for the following Newton-Raphson-based loops B and C.

The above table also shows the most likely parameter-to-goal impact which is useful in case of manual parameter optimization.</td></tr>
<tr><td>MASK_PAR_B and MASK_ PAR_C</td><td>Lists of dynamic model parameters which take part in the Newton-Raphson-based parameter optimization by error minimization. The following parameters are recognized. The parameters included by default are in bold letters.

<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td>CIN0 CIN1</td><td>The constant input capacitances are split into two values which take effect either at the on-switch edge (1) or at the off-switch edge (0). Generally, the input capacitance is expected to be a constant value but depends really on the state of the</td></tr>
</table></td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>
<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td></td><td>MOSFET channel and the applied GS- and DS voltages, which is highly complicated. By splitting CIN into two independent parts, the parameter optimization gets less complicated. The transition from CIN0 to CIN1 takes place at the trigger moment before any change by transition. Internally the charge balance is preserved to avoid unexpected current injection.

CIN0 and CIN1 should change during the optimization in a similar range and stay much larger than the corresponding feedback capacitance CR0 and CR1 to ensure stable simulation results.

If one of CIN0 and CIN1 changes greatly, especially toward smaller values, then this CINx should be deleted from the list before restarting the extraction.

In some cases the whole characterization becomes unstable because CIN0+CIN1 gets much too small compared to CR0+CR1. In this case, you can exclude CIN0 and CIN1 ; the solution must then be found using the initial CIN values. Normally you can take the data sheet values as initial values. If CIN is not available, input zero so the initial value can be derived from the nominal values.</td></tr>
<tr><td>CR0 CR1</td><td>Feedback capacitance CR is also split into off(0) and on(1) parts (see CIN) to reach a better match with the optimization targets. It is selectable if CR is a constant or a voltage dependent value, under Model & Goal Settings. The data sheet values can be taken as initial values. If CR is not available, set zero for the initial value calculation derived from the nominal values (see CIN). In general the initial input CIN:=0 and CR:=0 will work and result in a parameter optimum. But to save time a good initial guess for CIN and CR is recommended.</td></tr>
<tr><td>DELTACR0 DELTACR1</td><td>For a rough calculation or to find a better initial guess DELTACR values can be excluded. Excluding parameters enhances the optimization speed. For example, you can exclude DELTACRx in loops A and B if the refinement loop C will follow.

If DELTACRs are enabled or part of the parameter list, more</td></tr>
</table>
</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Parameter	Information	
	Parameter	**Meaning**
		accurate optimization results for different working points are expected. More parameters, which must have non-vanishing impact to at least one goal value, result in a larger freedom of the optimization problem and faster convergence. However, there is no clear rule if a maximum number of free parameters will accelerate simulation and increase the accuracy. The most important rule is that the parameter impacts-to-goal values should have minimum correlation. This information is not available before the optimization run, so sometimes trial and error is a useful way to find an optimum solution.
	TAUFD	This parameter directly influences the amount of excess charge storage of the active freewheeling diode. Thus, the impact can be seen in the on switch current overshoot area. Eon of pn- junction devices strongly depend on TAUFD setting – if a freewheeling diode is used and really active at the considered switching edge. Resistive switching time test circuits don't show this effect. Thus, the parameter **TAUFD** becomes meaningless and must be excluded from the parameter list. If in general a meaningless parameter or a parameter having very low impact to any selected goal value takes part in the optimization, it takes much longer and the convergence path may deteriorate. Shottky junction devices or modules having a Shottky freewheeling diode anti-parallel to the semiconductor switch don't store excess carriers in the junction region. You can use at least a minimum TAUFD value to be able to adjust the capacitive reverse recovery.
	SF	The larger the soft factor **SF**, the smaller the current overshoot gets and the longer the reverse recovery time **t_ RR** is. In most cases **SF** can stay at default. If the optimization for loss energies and delay times together becomes difficult or the process of optimization takes much more time than expected, the inclusion of **SF** may solve the problem. The optimization result should be in the range 1 … 7.
	TAUTAIL	Both parameters are intended to control the current tail effect

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>
<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td>DELTATAIL</td><td>due to excess carrier storage in the active switching device at the off-switch edge. Assuming the current tail as a falling exponential function starting at the off-switch trigger moment, the current duration can be controlled by the time constant at 1/e = 36.8% of the starting value. Thus, TAUTAIL must be in the same range as the off switch delay Toff and the storage time TAUS. Much too large TAUTAIL problems with H_OFF arise and much too small initial values will practically exclude TAUTAIL from the parameter optimization because of negligible impact.

DELTATAIL sets the starting value compared to the load current just before the off-switch. It can be even larger than 1 to reshape the off-switch edge and thus to control the off-switch energy in any case. You should include both parameters in the list for any material and technology.</td></tr>
<tr><td>RG</td><td>RG is the internal gate resistance due to the spreading resistance of poly silicon. Sometimes a separate resistor is added to limit the input current at the switching edges.

It impacts all goal values because it controls the charging and discharging speed.

If external gate resistances given in the data sheet RG should be set to 20% of (RG_ON + RG_OFF)/2. RG is treated as a free parameter to find the best trade-off between all goal values. RG should be part of the characterization in all cases.</td></tr>
<tr><td>TAUS</td><td>This parameter has direct feed through to the off-switch delay time. It controls the miller level duration and thus the off-switch storage time which is the main part of the off-switch delay time Toff. All devices must include this parameter.</td></tr>
<tr><td>DAMPING</td><td>Calculated from the stray inductance and the feedback (Miller) capacitance, each component gets an internal RC-snubber parallel to the output terminals. For a rough characterization, set DAMPING:=1 and leave it untouched. DAMPING is useful for solution refinement especially in loop C.</td></tr>
<tr><td>L_EXTERN</td><td>If the external stray inductance of the test circuit is not known</td></tr>
</table>
</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>
<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td></td><td>(as in most cases), it is possible to include this value in the parameter optimization. L_EXTERN influences the distribution of the total switching energy across the on and off switch edge.

If L_EXTERN is included, the convergence speed improves. By default, a fix value of 2nH is assumed instead if changing L_EXTERN.</td></tr>
<tr><td>LAUX</td><td>The value of the inner emitter inductance has impact to di/dt and dv/dt. This parameter is especially useful to slow down the voltage rise at switch off. For parallel connected devices it controls the amount of ringing.</td></tr>
</table>
</td></tr>
<tr><td>NUM_JACO_A, NUM_JACO_B, and NUM_JACO_C</td><td>This parameter defines which entries from the sensitivity matrix are considered for the matrix search:

-1: largest column, 0: apply threshold >0: this number of largest entries

For Loops_A when using a simple 1D search, this parameter is ignored.</td></tr>
<tr><td>TOL_JACO_A, TOL_JACO_B, and TOL_JACO_C</td><td>This parameter defines the relative threshold for Jacobian entries to be used in the matrix search (when NUM_JACO_x = 0).

For LOOPS_A it can be set to -1 which will switch LOOPS_A to simple 1D search.</td></tr>
<tr><td>RESORD_A, RESORD_B, and RESORD_C</td><td>This is the order of the residual. The number range is 0 .. 2:

0 – The maximum error is the residual value (the error number).

1 – Absolute error average.

2 – Root mean square error value. If the initial parameter set is far off the optimum solution (for example, RESORD_A) this parameter should be 2. toward the optimal solution, to achieve a certain error limit (for example, RESORD_C) this parameter should be 0.</td></tr>
<tr><td>RESTOL_A, RESTOL_B, and RESTOL_C</td><td>Desired accuracy of the residual according to the RESORD setting. To quickly move the model parameters toward the optimum, the tolerance can be set as large as 0.2 = 20%. The number of necessary loops A, B and C will decrease strongly. Use the final accuracy only in the last main loop (usually loop C).

Relaxing the accuracy can result in a noticeable reduction of simulation time.</td></tr>
<tr><td></td><td>Within A (when matrix search is used) and within loops B and C the</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Parameter	Information
MATRIX_A, **MATRIX_B**, and **MATRIX_C**	recalculation of the Jacobean matrix will be done this number unless the residue falls within the accuracy region. **MATRIX_B** and **MATRIX_C** are the maximum numbers of Jacobean recalculation, which takes most of the simulation time. In terms of pure mathematics, the recalculation should be done in every iteration cycle to stay close on the direct path to the optimum. It turns out to be a good trade-off between the shortest path and the simulation duration to leave the Jacobean untouched for some iterations before its recalculation by simulation. For loop A when using a simple 1D search, this parameter is ignored.
ZEROFN_A, **ZEROFN_B**, and **ZWROFN_C**	This is the number of iterations without changing the Jacobean (system) matrix. Only the right side of the balance system—the so-called zero function—is changed in every iteration step. This residue calculation is much faster and drives the parameter changes in roughly the right direction toward the optimum solution (the optimum parameter set). These values are called Samanski factors. For loop A, when using a simple 1D search, this parameter is ignored.
RESLOC_A, **RESLOC_B**, and **RESLOC_C**	Numbers of local residue (under-relaxation values differ between parameters) use before the system becomes stiff and hard to solve. After **RESLOC_***x*, total iterations for under-relaxation and boosting the main diagonal of the Jacobean, no longer is each parameter treated separately but all together globally the same way (under-relaxation values equal for all parameters), which slows down the convergence considerably but forces the solution path back to the optimum path. The additional convergence control stays unused by large **RESTOL_***x* numbers to save time and to make the optimization feasible. For loop A, when using a simple 1D search, this parameter is ignored.

- Model & Goal Settings

  You can choose dynamic fitting goals in **Select Goals to Display**. The checked goals appear in the **Dynamic Model Input** dialog box [10/12], so they are included as fitting goals during extraction.

  You can also adjust the dynamic model type in **Adv. Model Settings**. Each drop-down list corresponds to one digit of parameter **TYPE_DYN** in the basic dynamic IGBT model. For more detail, see the documentation for the Basic Dynamic IGBT Model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Tutorial for Characterizing a Power MOSFET (Basic Dynamic Model)

Use the power MOSFET Device Characterization Wizard to parameterize a power MOSFET model to match both the DC and dynamic characteristics of a device using information from the manufacturer's data sheet. The information available in the data sheet limits the accuracy of the parameterized power MOSFET model. For example, if the data sheet does not provide a device's transfer characteristics, the accuracy of the gate-voltage dependence of the output is reduced.

The power MOSFET device characterization tool performs the following tasks:

- **Static parameter fitting**

  The static parameter fitting uses the following user-supplied information to determine the value of static core model parameters:

  - The MOSFET’s transfer characteristic.
  - Its output characteristic.
  - Its thermal characteristic.
  - The forward and thermal characteristics of the MOSFET’s source to drain diode, if used.
- **Dynamic parameter extraction**

  The dynamic parameter extraction uses additional user-supplied information–namely the turn-on time, turn-off time, turn-on energy, and turn-off energy–to determine the values of the MOSFET’s dynamic parameters. The datasheet test conditions for these parameters, as supplied by the manufacturer, serve as the nominal conditions you must input into the device characterization tool to correctly characterize the device.

- **Validation**

  When the static parameter fitting and dynamic extraction are complete, the device characterization tool uses the static and dynamic parameters to perform a validation based on the test conditions provided.

  The wizard characterizes a wide array of power MOSFETs. The available information varies from vendor to vendor, and even between devices from the same vendor. The following procedures include vendor-specific information for help with characterizing models for several vendors' devices. If you have any additional questions, contact Ansys support.

Related Topics

Preliminary Considerations Using Power MOSFET Data Sheet Information

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Using the Device Characterization Wizard for a Power MOSFET

Preliminary Considerations Using Power MOSFET Data Sheet Information

Before starting a device characterization, study the manufacturer's data sheet for a power MOSFET to characterize. Each manufacturer has a unique design and naming convention for their data sheets. Different manufacturers also measure device characteristics differently.

- First, determine the nominal conditions for the device. Ideally, these are the conditions at which the device will be operated. Practically, these conditions are those for which you have the most information closest to the desired operating conditions. Search the data sheet and look at all available data to decide on the nominal point. The measurement conditions for the dynamic characteristics (**Eon**, **Eoff**, **td(on**), and **td (off)**) are generally good choices.
- DC characteristics are parameterized using the transfer and output characteristics, which are found as plots in the sheet.
- Dynamic behavior is characterized using the switching energies and times, which can be found either in a table or read from a plot.

> **Note:**
>
> The power MOSFET wizard documentation includes information and recommended settings for several manufacturers. Consult these sections in the documentation before characterizing a device.

Related Topics

Using the Device Characterization Wizard for a Power MOSFET

Advanced Settings for Power MOSFET Characterization

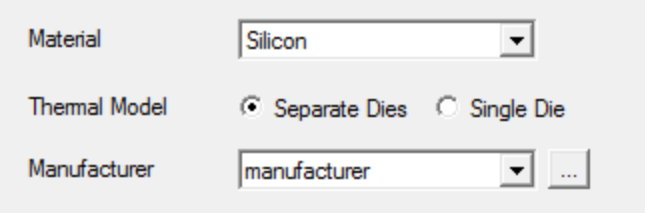
Using the Device Characterization Wizard for a Power MOSFET

The following procedures use an International Rectifier® device as an example. The process is similar for other manufacturer's devices.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> At any time during the characterization process, click **Save Model** to save your progress.
>
> A characterization can be loaded on the **Import Model** tab.

1. In the **Component Information [1/12]** dialog box, enter the component name and material and define the Manufacturer Data. There are three important settings that influence the characterization result:



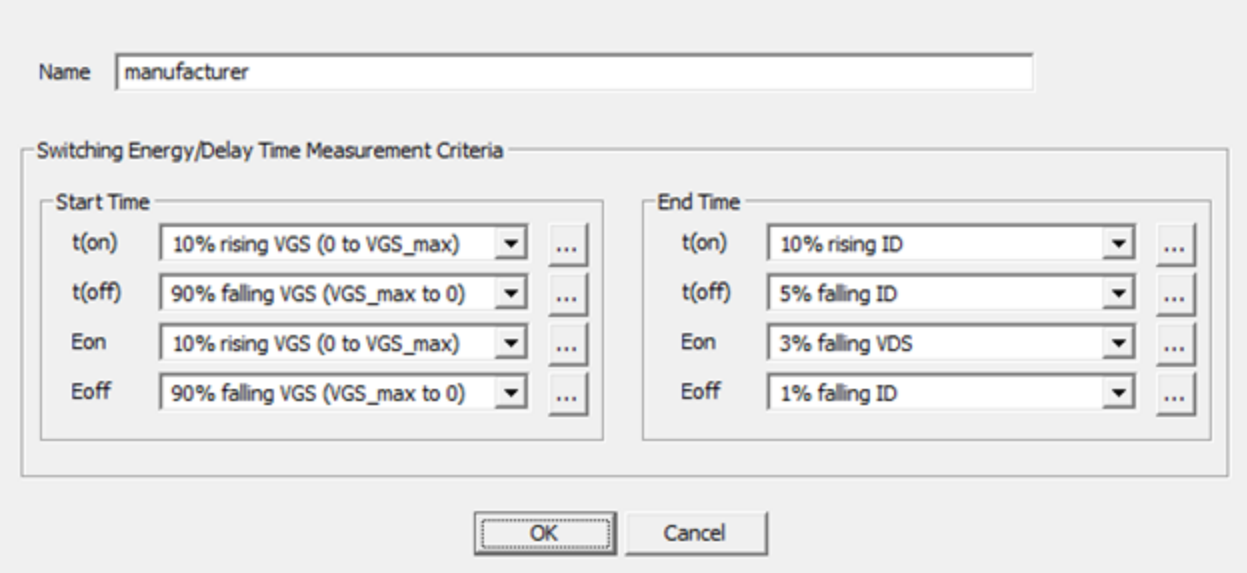
- **Material** – Si and SiC MOSFETs are currently supported.
- **Thermal Model** – This option specifies whether the model contains one or two thermal networks, which influence the correctness of temperature simulation. It has no impact if thermal simulation is not being used.
  - If the diode in MOSFET is on a separate die, choose **Separate Dies** and the extracted model contains two internal thermal networks, one for transistor and one for diode.
  - If the diode in MOSFET is its own body diode, choose **Single Die** and the extracted model contains one thermal network.

    > **Note:**
    >
    > Manufacturers usually specify in the datasheet if there is a body diode. One can also determine the number of dies by whether the datasheet contains joint or separate thermal impedance plots for MOSFET and diode.

- **Manufacturer** – These are measurement criteria of dynamic characteristics defined by manufacturers. The information is usually found in the datasheet or device application note from the manufacturer. The detailed documentation describes the specific settings

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

for several vendors. For example, International Rectifier uses the following settings:



This manufacturer data is critical for the correct characterization of the device. Because each manufacturer uses its own standard to define **Ton**, **Toff**, **Eon**, and **Eoff**, it is important to measure these quantities the same way in the characterization wizard when fitting the model to output the same. If the settings in this dialog box are different from the actual criteria used by the vendor, the extracted model may be inaccurate even with good validation results in [11/12] the wizard.

Note:

If the manufacturer uses a start or stop criterion that is not available in the options, select the point that is closest to that criterion.

2. The **Nominal Working Point Values [2/12]** dialog box sets nominal values at the device's working point. As mentioned above, ideally, these are the operating conditions for the device in the design. Practically, these values are the test conditions on the manufacturer data sheet specified for the switching time and energy information obtained and used for dynamic parameters extraction. These are the conditions for which the parameterized device will best match the measured characteristics. For International Rectifier, generally pick the measurement conditions for the dynamic characteristics. Pick the values of **VDD** and **ID** that are mentioned in the table next to the value of the turn-on delay time, **td(on)**. Select the temperature for which you have the output characteristics in the plots section. **VGS_on** and **VGS_off** are gate voltages used in manufacturer's test circuit in which switching dynamics are measured. In the datasheet, **VGS_on** and **VGS_off** can be found next to the value of **td(on)**. **Cin**, the input capacitance, can be found in the datasheet. **Cr**, the Feedback (Miller) capacitance can be found in the datasheet. It is

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

sometimes referred to as reverse transfer capacitance. If the values cannot be found in the datasheet, **0** (zero) is a good default value, as the characterization tool will then pick a value using the other device characteristics. Click **Next** to continue.

3. On the **Breakthrough Values [3/12]** dialog box, check **Disable Breakthrough Model**, as the breakdown characteristics data is rarely available in a datasheet. Click **Next** to continue.
4. The **Half-Bridge Test Circuit Condition [4/12]** dialog box sets up the test circuit under which the model's dynamic characterizations are measured. It is important to have the same test circuit used by the vendor when the datasheet values are measured.

   The required data can generally be found in the datasheet and device application note from the vendor.

- **Rg** is the internal gate resistance found in the datasheet. This value is used as an initial value during the dynamic characterization. It is not the internal gate resistance the characterized device ends up with.
- **R_tot** is generally omitted in the datasheet; **5% of Rds in [2/12]** is a good default for MOSFET. If R_tot is provided in the datasheet and the device is a MOSFET module that contains two MOSFETs in series between two external pins, use half of the R_tot value in the datasheet.
- **L_tot** is generally omitted; **0** (zero) is a good default. If L_tot is provided in the datasheet and the device is a MOSFET module that contains two MOSFETs in series between two external pins, use half of the L_tot value in the datasheet.
- **L_extern** is given by **Ls** in the test conditions for **td(on)**. If the value is not there, **2 nH** is a good value to use. The characterization wizard parameterizes this value to find a good fit with the dynamic data.
- **Rg_on** and **Rg_off** are found next to the values of the switching energies **td(on)** and **Eon**.

- **Cge ext** is generally omitted and a default value of **1e-12F** used.
- **CLoad** is sometimes found on the datasheet as load capacitance. Use **0** (zero) if it is missing.
- **RLoad** is the load resistance value when a resistive load test circuit is used. When its check box is cleared, use an inductive load test circuit.
- **Test Circuit (DUT location)** defines the location of the device under test (DUT) in the test circuits. Together with **RLoad**, it is used to define different configurations of the test circuit.
  - If the **RLoad** check box is selected, the drop-down options are **Lower Transistor** and **Upper Transistor**.
    - If the **RLoad** check box is cleared, the drop-down options are **Lower Transistor**, **Upper Transistor**, and **Module (Halfbridge)**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

You can see the test circuits used after selecting **Test Circuit** in the last step,then **Finish**.

> **Note:**
>
> Load type and Test Circuit (DUT location) usually appear in either the datasheet or application note from the manufacturer. When multiple test circuits are provided, use the one for dynamic characteristics (switching time and energy) measurement.
>
> In general, the Module (Halfbridge) setting should be used only for MOSFET modules that contain two MOSFETs in series between two external pins.

- When finished setting values, click **Next** to continue.

5. Use the **Transfer Characteristic Id=f(Vgs) [5/12]** dialog box to parameterize the threshold voltages and transconductance of the MOSFET. Enter the data with SheetScan.

   To do so, click **Load characteristics from Dataset Manager** above the table and click **SheetScan**. Load the graphs in graphic files captured from the manufacturer data sheet using the **Picture** > **Load picture** command.

For an International Rectifier device, the data sheet plot shows **ID=f(VGS)**.

a. Select a coordinate system (**Coordinate System** > **New**) and define it on the plot. To do so, you must select three points. Typically, you should use the bottom-right, bottom-left and top-right points of the plot grid. Click **Point1**, **Point2** and **Point3** and select the corresponding points on the graph. Enter the corresponding X-Values and Y-Values for these points (read from the plot's X-axis and Y-axis labels) in the table and click **OK** to finish defining the coordinate system.
b. To define the first characteristic, select **Curve** > **New** and give names to the X-axis and Y-axis in the **Curve Settings** dialog box. Click **OK** when finished.

   > **Note:**
   >
   > You can save SheetScan work for later adjustment and reuse.

c. Make sure to note the temperature given on the plot. Then select several points on the curve starting with the lowest X-value. Pick at least four points. When done, click **File** > **Export** and click **Dataset** in the resulting **Save** dialog box.
d. If a plot at a different temperature is available, repeat steps **a**, **b**, and **c** to record additional data.
e. When finished, click **File** > **Exit** to exit SheetScan, saving the scan setup information, if desired.
f. In the **Datasets** dialog box, select the data to use at the nominal temperature and click **Done**. The data is transferred to the **Characteristic Data** table in the **Transfer**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Characteristic [5/12] dialog box. Make sure to enter the correct values for temperature and Vce, which you recorded during the SheetScan measurements.

g. If you also recorded plot data for a different temperature, click **Add new characteristic** ( ) in the top right corner and click **Load characteristics from Dataset Manager** ( ) above the table to load the additional data. Select the plot in the first tab for the **Nominal Temperature** field of the **Fitting Characteristic Order** panel. If you added a second set of data for a different temperature, select this plot for the **Different Temperature** field, or select **Not Used**, if data is available only for one temperature.

h. Click **Start Fitting** to fit the characteristics and examine the resulting plot to check the match of the fit. Click **Next** to continue.

6. Use the **Output Characteristic Id=f(Vds) [6/12]** dialog box to parameterize more of the MOSFET's output characteristics. Ideally, enter the output characteristics at **Vgs=VGS_on (Full Saturated Branch**) and for Vgs at a lower voltage (**Semi Saturated Branch**), for both the nominal temperature and at a different temperature, using SheetScan as described in step **5**. Make sure to note the value of **Vgs** and the corresponding temperature for each curve.

   For International Rectifier, the datasheet plot shows **ID=f(VDS)**. The output characteristics of the semi-saturated branch are often missing at a different temperature.

   a. Add up to four characteristics, making sure to identify them with the correct values of VGS and temperature.

   b. Select the plot at the nominal temperature and with **Vgs=VGS_on** for the **Full Saturated Branch (Tnom)** field in the **Fitting Characteristic Order** panel, and one at a lower value of Vgs for the **Semi Saturated Branch (Tnom)** field, or select **Not Used**, if the data is not available. Repeat this at a different temperature for **Vgs=VGS_on** and at a lower Vgs in the corresponding **Full Saturated Branch (Tdiff)** and **Semi Saturated Branch (Tdiff)** fields. The fit will be better if the current does not saturate much for the semi-saturated curve, but the fitting algorithm will work with either option.

   c. Click **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

7. Use the **Drain-Source Diode Characteristic If=f(Vf) [7/12]** dialog box to parameterize the DC characteristics of the diode. Ideally, you should enter the diode forward characteristic If(Vf) at both the nominal temperature and at a different temperature using SheetScan as outlined in step **5**. Make sure to note the value of the temperature for each curve.

   For International Rectifier, the datasheet plot shows **ISD=f(VSD)**.

   a. Generate two characteristic data sets, one at the nominal temperature and one at a different temperature, being sure to identify them with their corresponding temperatures.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

b. Select the plot at the nominal temperature in the **Nominal Temperature** field of the **Fitting Characteristic Order** panel, and the plot at the different temperature in the **Different Temperature** field, or select **Not Used**, if the data is not available.

c. Click **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

8. Use the **MOSFET Thermal Model [8/12]** dialog box to parameterize the thermal impedance of the MOSFET. Do one of the following:
   - If the data sheet provides extracted values for **ri** and **ti**, which is the case for International Rectifier, enter these four value pairs in the **Use Fraction Coefficients** table, click **Start Fitting**, check the plot and press **Next** to continue.
   - If the data sheet does not provide extracted values for **ri** and **ti**, select **Use Transient Thermal Impedance** and enter the plot data for the thermal characteristics using SheetScan per the instructions given in step **5**. The plot will show the impedance as a function of time. Make sure to adjust the scale of the coordinate system in SheetScan to logarithmic if needed. Select **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

   For additional information about the MOSFET Thermal Model, see IGBT Thermal Model because the information is valid for both thermal models.

> **Note:**
>
> Sample the "single pulse" curve when multiple transient thermal impedance curves are presented in the data sheet.

9. Use the **Drain-Source Diode Thermal Model [9/12]** dialog box to parameterize the thermal impedance of the diode.
   - If the data sheet provides extracted values for **ri** and **ti**, which is the case for International Rectifier, enter these four value pairs in the **Use Fraction Coefficients** table, click **Start Fitting**, check the plot, and click **Next** to continue.
   - If the data sheet does not provide extracted values for **ri** and **ti**, select **Use Transient Thermal Impedance** and enter the plot data for the thermal characteristics using SheetScan per the instructions given in step **5**. The plot will show the impedance as a function of time. Make sure to adjust the scale of the coordinate system in SheetScan to logarithmic if needed. Select **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.
10. Use the **Dynamic Model Input [10/12]** dialog box to parameterize the dynamic characteristics of the power MOSFET. You can select to fit for the switching energies (**Eon** and **Eoff**) and switching times (**Ton** and **Toff**). The nominal point has to be fit, so the data has to be available. For more information about the input settings, see Dynamic Model Input. The **Advanced Settings** button allows more control over the characterization process, but should not be necessary to change for most devices. Check the Advanced

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Settings documentation for more information. For more information about the model settings, see Dynamic Model Input.

a. Click **Measurement** to open the **Measurement Data** dialog box and make sure the settings correspond to the ones for the manufacturer of the device being characterized. For example, for International Rectifier, use **Eon** and **Eoff** from the data sheet as the goal settings for **Eon** and **Eoff**. Use **td(on)+tr** as the goal setting for **t(on)**, and **td(off)+tf** as the goal setting for **t(off**).

b. Set weights and desired residue for the goals. By default, weights are set equally to 1, and residue is 5% to provide good trade-off between accuracy and extraction time.

c. If data is available at different temperature, add it to the **dT** row.

d. If data is available at a different value of VDS, add the lower value of VDS to the **nV** row and the higher value of VDS to the **pV** row.

e. If data is available at a different value of **Id**, add the lower value of **Id** to the **nI** row and the higher value of **Id** to the **pI** row.

f. Optionally, the **Adv. Settings** button allows for more control over the characterization process, but should not be necessary to change for most devices. Check the more detailed description for more information.

Advanced Settings

- **T_ON_MAX** – total On-switch signal settling time (trigger to 1% falling VDS)
- **T_OFF_MAX** – total Off-switch signal settling time (trigger to 1% falling ID)
- **H_ON** – On-switch signal acquisition time
- **H_OFF** – Off-switch signal acquisition time

For low and medium power devices, the default settings should be sufficient to ensure correct data measurement. For high power devices, the maximum value might be increased (2-5 times). The actual acquisition time is increased by the process if necessary.

- The parameter extraction process uses three optimization routines to determine values for the dynamic device parameters. The first routine is an iterative loop that uses a 1D search method to progressively refine its approximation. To enable it, set **LOOPS_A** to an integer number larger than 0 (zero). **MASKPAR_A** contains the names of the optimization parameters to reach a good convergence. **RESORD_A** sets how the residue is defined 0 (zero) for the maximum error, 1 for the average error and 2 for the root mean square error. **RESTOL_A** defines the value under which the residue must get to leave the loop with a good solution.
- The second and the third loop (B and C) use a Jacobian matrix method. **LOOPS_B** and **LOOPS_C** set the number of pre-loops taken for the second and third loop; usually 1 will suffice. **MASKPAR_B** and **MASKPAR_C** contain the names of the parameters changed by the characterization tool to find a better fit during the second and third loop. **MATRIX_B** and **MATRIX_C** set the maximum number of

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Jacobian recalculations, while **ZEROFN_B** and **ZEROFN_C** set the maximum number of constant Jacobian calculations. **RESLOC_B** and **RESLOC_C** set the maximum number of relaxation which occurs within a constant Jacobian calculation. **RESORD_B** and **RESORD_C** set how the residue is defined: 0 (zero) for the maximum error, 1 for the average error and 2 for the root mean square error. **RESTOL_B** and **RESTOL_C** define the value under which the residue must get to leave the second and the third loop with a good solution. However, if the values of **RESTOL_B** and **RESTOL_C** are left at zero, the simulation will use the Res [%] values from the Dynamic Model Input table.

f. Click **Extraction** to start the fit. A dialog box showing the progress of the extraction appears. Upon completion, another dialog box informs you that extraction is complete. Click **OK** to return to the **Dynamic Model Input [10/12]** dialog box.

g. Click **Next** to continue.

11. Use the **Dynamic Parameter Validation [11/12]** dialog box to validate the dynamic extraction. The actual values of the switching times and energies for the parameterized device are calculated. **Enable** the conditions that you need to check and click **Validate**. Click **Next** to continue.
12. Use the **Model Parameters [12/12]** dialog box to browse the extracted parameters.
    a. To generate an **.sml** file of the model, click **Create SML**.
    b. To place a characterized device in the Twin Builder project, click **Place Component** and click **Finish**.
    c. To generate a test circuit using the characterized device, click **Testcircuit** and click **Finish**.

Advanced Settings for Characterizing a Power Mosfet

Click **Adv. Settings**in the **Dynamic Model Input** [10/12] dialog box to access the advanced settings of basic dynamic IGBT characterization. There are two tabs: **Extraction Settings** and **Model & Goal Settings**.

- Extraction Settings

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Name	Value	Parameter Info
H_ON	5e-06	Initial On-switch signal acquisition time (2*time from trigger to 1% falling VCE) [s]
H_OFF	1e-05	Initial Off-switch signal acquisition time (2*time from trigger to 1% falling IC) [s]
INTEGRATION	0	Integration method: 0:Euler, 1-Adaptive, 2-explicit Euler, 3-Trapezoid
MEASURE_FLAG	1	Measurement mode: 0:continuous 1:discontinuous
MEASURE_RE...	1	Measurement tool output: 0:none 1:waves and goal values 2:+convergence 3:+fitting
FITTINGREPORT	1	Fitting report output: 0:none 1:fitting check 2:fitting details
LOOPS_A	1	Number of pre-loops using 1D search
TOL_JACO_A	0	-1: simple 1D search; >=0: Matrix search - Relative Jacobian entry threshold for parameter change
NUM_JACO_A	10	Number of most important Jacobian entries considered for matrix search
MASKPAR_A	CR0 CR1 TAUFWD TAUTAIL ...	Set parameters for optimization, CIN0 CIN1 TAUFWD ...
MATRIX_A	3	Maximum number of Jacobean recalculations, matrix loops A (for TOL_JACO_A>0)
ZEROFN_A	15	Maximum number of constant Jacobean iterations, Samanskii factor (for TOL_JACO_A>0)
RESLOC_A	100	Maximum number of local relaxation within ZEROFN
RESORD_A	2	Order of 1D residue 0:MAX 1:AV 2:RMS
RESTOL_A	0.05	1D residue tolerance (0.1=10%)
LOOPS_B	1	Number of pre-loops using Jacobean matrix
TOL_JACO_B	0.1	Relative Jacobian entry threshold for parameter change
NUM_JACO_B	10	Number of most important Jacobian entries considered for matrix search
MASKPAR_B	CIN0 CIN1 CR0 DELTACR0 C...	Set parameters for optimization, CIN0 CIN1 TAUFWD ...
MATRIX_B	5	Maximum number of Jacobean recalculations, matrix loops B
ZEROFN_B	15	Maximum number of constant Jacobean iterations, Samanskii factor
RESLOC_B	1000	Maximum number of local relaxation within ZEROFN
RESORD_B	0	Residue order 0:MAX 1:AV 2:RMS
RESTOL_B	0.05	Loop B residue tolerance (0.1=10%, 0=take TOL from WP)
LOOPS_C	1	Number of main Jacobean matrix loops
TOL_JACO_C	0.01	Relative Jacobian entry threshold for parameter change
NUM_JACO_C	10	Number of most important Jacobian entries considered for matrix search
MASKPAR_C	CIN0 CIN1 CR0 DELTACR0 C...	Set parameters for optimization, CIN0 CIN1 TAUFWD ...
MATRIX_C	15	Maximum number of Jacobean recalculations, matrix loops C
ZEROFN_C	10	Maximum number of constant Jacobean iterations, Samanskii factor
RESLOC_C	1000	Maximum number of local relaxation within ZEROFN
RESORD_C	0	Residue order 0:MAX 1:AV 2:RMS
RESTOL_C	0	Loop C residue tolerance (0.1=10%, 0=take TOL from WP)

Figure 7-2 Advanced Parameter Setting Dialog Box

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

These settings allow for more control over the characterization process. It should not be necessary to change them for most devices.

Parameter	Information
H_ON and **H_ OFF**	Before preprocessing and interpolation of the on/off switch wave forms, a series of simulated data points must be stored temporarily. These parameters set the acquisition time for the data points. This time interval setting limits the amount of acquired data and must fit to the longest possible time constants at the on/off switch together with a remaining time buffer to allow the Jacobean entry calculation which sweeps every parameter in a wide range. The sensitivity analysis for the 1D search in loop A varies actual parameters from p-25% ... p+50% and the Jacobean-based method does a parameter variation in the range p-10% ... p+10%. This process repeats several times to calculate either the actual best fitting single parameter or a parameter to goal sensitivity coefficient. Thus, the transient wave form duration will vary in a wide range. **H_ON** and **H_OFF** are the initial settings for the On-switch and OFF-switch signal acquisition times. The actual **Hon** and **Hoff** values change during the characterization. **Hon** and **Hoff** derive from the measured switching times and length of the reverse recovery. 5x…10x of the On-switch/reverse recovery time should be a good value. For SiC devices, the values should be much smaller than for Si devices. If the initial values are not good, the extraction process can not determine the sensitivity of parameter changes to the output signal, and the extraction will not start to converge. You can observe the convergence of the extraction on the Fitting info page and adjust the initial values for **H_ON** and **H_OFF** if necessary.
INTEGRATION	Integration method (**0** - best for switching devices).
MEASURE_ FLAG	Measurement mode (**1** – allows for stop of measurement cycles as soon as good result is reached).
MEASURE_ REPORT	Measurement tool output (should stay **1**).
FITTINGREPO RT	Fitting report output (should remain set to **1**).
LOOPS_A, **LOOPS_B**, and **LOOPS_C**	Approaching the optimal parameter set is an iterative procedure. This iteration is split into three sequenced loops A, B and C. Each of the sequences can be repeated any number of times according to **LOOPS_***x*. This is especially helpful in the case of loop A, which is a predefined sequence of one dimensional parameter sweeps. In some special cases an increased number **LOOPS_A** results in much better initial parameter values before starting the Newton-Raphson-based matrix loops B and C. But the repeated 1D parameter sweep takes time, and a global convergence

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>is not guaranteed. In most cases the initial values calculated internally from the nominal conditions are good enough to start loop B immediately without time-consuming loop A.

Default values are LOOPS_A:=0, LOOPS_B:=1, and LOOPS_C:=1.</td></tr>
<tr><td>MASK_PAR_A</td><td>This is a list of dynamic model parameters which take part in the 1D parameter sweep sequence. The input is interpreted as a parameter mask for parameter selection during the characterization. Parameter names in bold mark the values included by default.

<table>
<tr><th>Parameter</th><th>Observed impact to goals</th></tr>
<tr><td>CIN0</td><td>Eon, Eoff</td></tr>
<tr><td>CIN1</td><td>Ton</td></tr>
<tr><td>TAUS</td><td>Toff</td></tr>
<tr><td>CR0</td><td>Ton, Toff</td></tr>
<tr><td>CR1</td><td>Ton, Toff</td></tr>
<tr><td>TAUFWD</td><td>Eon</td></tr>
<tr><td>DAMPING</td><td>Eoff</td></tr>
<tr><td>RG</td><td>Ton</td></tr>
<tr><td>DELTATAIL</td><td>Toff, Eoff</td></tr>
<tr><td>TAUTAIL</td><td>Toff, Eoff</td></tr>
<tr><td>L_EXTERN</td><td>Eon, Eoff</td></tr>
</table>
A repeated application of loop A helps to find a good initial parameter guess for the following Newton-Raphson-based loops B and C.

The above table also shows the most likely parameter-to-goal impact which is useful in case of manual parameter optimization.</td></tr>
<tr><td>MASK_PAR_B and MASK_ PAR_C</td><td>Lists of dynamic model parameters which take part in the Newton-Raphson-based parameter optimization by error minimization. The following parameters are recognized. The parameters included by default are in bold letters.

<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td>CIN0 CIN1</td><td>The constant input capacitances are split into two values which take effect either at the on-switch edge (1) or at the off-switch edge (0). Generally, the input capacitance is a constant value but depends really on the state of the</td></tr>
</table></td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>
<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td></td><td>MOSFET channel and the applied GS- and DS voltages, which is highly complicated. By splitting CIN into two independent parts, the parameter optimization gets less complicated. The transition from CIN0 to CIN1 takes place at the trigger moment before any change by transition. Internally the charge balance is preserved to avoid unexpected current injection.

CIN0 and CIN1 should change during the optimization in a similar range and stay much larger than the corresponding feedback capacitance CR0 and CR1 to ensure stable simulation results.

If one of CIN0 and CIN1 changes greatly, especially toward smaller values, then this CINx should be deleted from the list before restarting the extraction.

In some cases the whole characterization becomes unstable because CIN0+CIN1 gets much too small compared to CR0+CR1. In this case, you can exclude CIN0 and CIN1 and the solution must be found using the initial CIN values. Normally the data sheet values are taken as initial values. If CIN is not available, input 0 to derive the initial value from the nominal values.</td></tr>
<tr><td>CR0 CR1</td><td>Feedback capacitance CR is also split into off(0) and on(1) parts (see CIN) to reach a better match with the optimization targets. It is selectable if CR is a constant or a voltage dependent value, under Model & Goal Settings. You can take the data sheet values as initial values. If CR is not available, set 0 for the initial value calculation derived from the nominal values (see CIN). In general the initial input CIN:=0 and CR:=0 will work and result in a parameter optimum. But to save time a good initial guess for CIN and CR is recommended.</td></tr>
<tr><td>DELTACR0 DELTACR1</td><td>For a rough calculation or to find a better initial guess, you can exclude DELTACR values. Since excluding parameters clearly enhances the optimization speed, you can exclude DELTACRx in loops A and B if the refinement loop C will follow.

If DELTACRs are enabled or part of the parameter list, more</td></tr>
</table>
</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>
<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td></td><td>accurate optimization results for different working points are expected. More parameters, which must have non-vanishing impact to at least one goal value, result in a larger freedom of the optimization problem and faster convergence. However, there is no clear rule if a maximum number of free parameters will accelerate simulation and increase the accuracy. The most important rule is that the parameter impacts-to-goal values should have minimum correlation. This information is not available before the optimization run, so sometimes trial and error is a useful way to find an optimum solution.</td></tr>
<tr><td>TAUFD</td><td>This parameter directly influences the amount of excess charge storage of the active freewheeling diode. Thus, the impact in the on switch current overshoot area is evident. Eon of pn- junction devices strongly depend on TAUFD setting – if a freewheeling diode is used and really active at the considered switching edge.

Resistive switching time test circuits don't show this effect. Thus, the parameter TAUFD becomes meaningless and must be excluded from the parameter list. If in general a meaningless parameter or a parameter having very low impact to any selected goal value takes part in the optimization, it takes much longer and the convergence path may be deteriorated.

Shottky junction devices or modules having a Shottky freewheeling diode anti-parallel to the semiconductor switch don't store excess carriers in the junction region. You can use at least a minimum TAUFD value to be able to adjust the capacitive reverse recovery.</td></tr>
<tr><td>SF</td><td>The larger the soft factor SF, the smaller the current overshoot gets and the longer the reverse recovery time t_ RR is. In most cases SF can stay at default. If the optimization for loss energies and delay times together becomes difficult or the process of optimization takes much more time than expected, the inclusion of SF may solve the problem. The optimization result should be in the range 1 - 7.</td></tr>
<tr><td>TAUTAIL
DELTATAIL</td><td>Both parameters control the current tail effect due to excess carrier storage in the active switching device at the off-switch</td></tr>
</table>
</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Parameter	Information	
	Parameter	**Meaning**
		edge. Assuming the current tail as a falling exponential function starting at the off-switch trigger moment, the current duration can be controlled by the time constant at 1/e = 36.8% of the starting value. Thus, **TAUTAIL** must be in the same range as the off switch delay Toff and the storage time **TAUS**. Much too large **TAUTAIL** problems with **H_OFF** arise and much too small initial values will practically exclude **TAUTAIL** from the parameter optimization because of negligible impact. **DELTATAIL** sets the starting value compared to the load current just before the off-switch. It can be even larger than 1 to reshape the off-switch edge and thus to control the off-switch energy in any case. Both parameters should be included in the list for any material and technology.
	RG	**RG** is the internal gate resistance due to the spreading resistance of poly silicon. Sometimes a separate resistor is added to limit the input current at the switching edges. It impacts all goal values because it controls the charging and discharging speed. If external gate resistances given in the data sheet **RG** should be set to 20% of (**RG_ON + RG_OFF**)/2. **RG** is treated as a free parameter to find the best trade-off between all goal values. **RG** should be part of the characterization in all cases.
	TAUS	This parameter has direct feed through to the off-switch delay time. It controls the miller level duration and thus the off-switch storage time which is the main part of the off-switch delay time **Toff**. All devices need to include this parameter.
	DAMPING	Calculated from the stray inductance and the feedback (Miller) capacitance, each component gets an internal RC-snubber parallel to the output terminals. For a rough characterization, set **DAMPING:=1** and leave it untouched. **DAMPING** is useful for solution refinement especially in loop C.
	L_EXTERN	If the external stray inductance of the test circuit is not known (as in most cases), it is possible to include this value in the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>
<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td></td><td>parameter optimization. L_EXTERN influences the distribution of the total switching energy across the on and off switch edge.

If L_EXTERN is included, the convergence speed improves. As default, a fix value of 2nH is assumed instead if changing L_EXTERN.</td></tr>
<tr><td>LAUX</td><td>The value of the inner emitter inductance has impact to di/dt and dv/dt. This parameter is especially useful to slow down the voltage rise at switch off. For parallel connected devices it controls the amount of ringing.</td></tr>
</table>
</td></tr>
<tr><td>NUM_JACO_A, NUM_JACO_B, and NUM_JACO_C</td><td>This parameter defines which entries from the sensitivity matrix are considered for the matrix search:

-1: largest column, 0: apply threshold >0: this number of largest entries.

For Loops_A when simple 1D search is used, this parameter is ignored.</td></tr>
<tr><td>TOL_JACO_A, TOL_JACO_B, and TOL_JACO_C</td><td>This parameter defines the relative threshold for Jacobian entries used in the matrix search (when NUM_JACO_x = 0).

For LOOPS_A, set the parameter to -1 which switches LOOPS_A to simple 1D search.</td></tr>
<tr><td>RESORD_A, RESORD_B, and RESORD_C</td><td>This is the order of the residual. The number range is 0 .. 2:

0 – The maximum error is the residual value (the error number).

1 – Absolute error average.

2 – Root mean square error value. If the initial parameter set is far off the optimum solution (for example, RESORD_A) this parameter should be 2. toward the optimal solution, to achieve a certain error limit (for example, RESORD_C) this parameter should be 0.</td></tr>
<tr><td>RESTOL_A, RESTOL_B, and RESTOL_C</td><td>Desired accuracy of the residual according to the RESORD setting. To quickly move the model parameters toward the optimum, the tolerance can be set as large as 0.2 = 20%. The number of necessary loops A, B and C will decrease strongly. It is suggested to use the final accuracy only in the last main loop (usually loop C).

Relaxing the accuracy can result in a noticeable reduction of simulation time.</td></tr>
<tr><td>MATRIX_A,</td><td>Within A (when matrix search is used) and within loops B and C, the recalculation of the Jacobean matrix will be done this number unless the</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Parameter	Information
MATRIX_B, and **MATRIX_C**	residue falls within the accuracy region. **MATRIX_B** and **MATRIX_C** are the maximum numbers of Jacobean re-calculation, which takes most of the simulation time. In terms of pure mathematics, the recalculation should be done in every iteration cycle to stay close on the direct path to the optimum. It turns out to be a good trade-off between the shortest path and the simulation duration to leave the Jacobean untouched for some iterations before its recalculation by simulation. For loop A when simple 1D search is used, this parameter is ignored.
ZEROFN_A, **ZEROFN_B**, and **ZWROFN_C**	This is the number of iterations without changing the Jacobean (system) matrix. Only the right side of the balance system—the so-called zero function—is changed in every iteration step. This residue calculation is much faster and drives the parameter changes in roughly the right direction toward the optimum solution (the optimum parameter set). These values are called Samanski factors. For loop A when simple 1D search is used, this parameter is ignored.
RESLOC_A, **RESLOC_B**, and **RESLOC_C**	Numbers of local residue (under-relaxation values differ between parameters) use before the system is considered to become stiff and hard to solve. After **RESLOC_***x* total iterations for under-relaxation and boosting the main diagonal of the Jacobean, no longer is each parameter treated separately but all together globally the same way (under-relaxation values equal for all parameters), which slows down the convergence considerably but forces the solution path back to the optimum path. The additional convergence control stays unused by large **RESTOL_***x* numbers to save time and to make the optimization feasible. For loop A when simple 1D search is used, this parameter is ignored.

- Model & Goal Settings

  You can choose dynamic fitting goals in **Select Goals to Display**. The checked goals appear in the table of the **Dynamic Model Input** dialog box [10/12], so it can be included as fitting goals during extraction.

  You can also adjust the dynamic model type in **Adv. Model Settings**. Each drop-down list corresponds to one digit of parameter TYPE_DYN in the basic dynamic IGBT model. For more detail, see the documentation for the Power MOSFET Model (Basic Dynamic).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Tutorial for Characterizing a Power MOSFET (Average)

Use the power MOSFET Device Characterization Wizard to parameterize an average power MOSFET model to match both the DC and switching energies of a device using information from the manufacturer's data sheet. The information available in the data sheet limits the accuracy of the parameterized power MOSFET model. For example, if the data sheet does not provide a device's transfer characteristics, the accuracy of the gate-voltage dependence of the output will be reduced.

The power MOSFET device characterization tool performs the following tasks:

- **Static parameter fitting**

  The static parameter fitting works by using the following user-supplied information to determine the value of static core model parameters:

  - The MOSFET’s transfer characteristic.
  - Its output characteristic.
  - Its thermal characteristic.
  - The forward and thermal characteristics of the MOSFET’s source to drain diode, if used.
- **Energy Characteristics**

  The average power MOSFET model is a static model capable of switching loss estimation. Swithching energies are interpolated/extrapolated according to working conditions at the time of each switching. The parameter extraction uses additional user-supplied information—namely, the t turn-on energy, turn-off energy, and reverse recovery energy—to determine the values of the MOSFET’s switching energy related parameters. The datasheet test conditions as supplied by the manufacturer serve as the nominal and alternative working conditions. You must input into the device characterization tool data at multiple working points to correctly characterize the device.
- **Validation**

  When the static parameter fitting and switching energy extraction are complete, the device characterization tool uses the full set of parameters to perform a validation based on the provided test conditions.

  The wizard can characterize a wide array of power MOSFETs. The available information varies from vendor to vendor, and even between devices from the same vendor. If you have any additional questions, contact Ansys support.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Preliminary Considerations Using Power MOSFET (Average) Data Sheet Information

Using the Device Characterization Wizard for a Power MOSFET (Average)

Preliminary Considerations Using Power MOSFET (Average) Data Sheet Information

Before starting a device characterization, study the manufacturer's data sheet for the power MOSFET you want to characterize. Each manufacturer has a unique design and naming convention for their data sheets. Different manufacturers also measure device characteristics differently.

- First, determine the nominal conditions for the device. Ideally, these are the conditions at which the device will be operated. Practically, these conditions should be those for which you have the most information closest to the desired operating conditions. Search the data sheet and look at all available data to decide on the nominal point. The measurement conditions for switching energies (**Eon**, **Eoff**) are generally good choices.

- DC characteristics are parameterized using the transfer and output characteristics, which can be found as plots in the sheet.
- Dynamic behavior is characterized using the switching energies at multiple working conditions, found either in a table or read from a plot. Switching energies of average model is estimated through interpolation/extrapolation based on known data. It is important to provide switching energies at multiple working points during characterization.

Related Topics

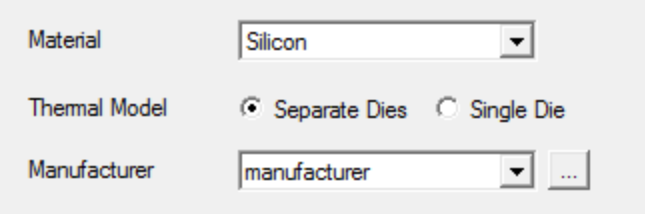
Using the Device Characterization Wizard for a Power MOSFET (Average)

Using the Device Characterization Wizard for a Power MOSFET (Average)

> **Note:**
>
> Click **Save Model** at any time during the characterization process to save your progress.
>
> You can load a characterization on the **Import Model** tab.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. In the **Component Information [1/12]** dialog box, enter the component name and material. There are two important settings that influence the characterization result:
   - **Material** – Si and SiC MOSFETs are currently supported.



   - **Thermal Model** – Specify whether the model contains one or two thermal networks, which influence the correctness of temperature simulation. It has no impact if thermal simulation is not in concern.
     - If the diode in MOSFET is on a separate die, choose **Separate Dies** and the extracted model contains two internal thermal networks, one for transistor and one for diode.
     - If the diode in MOSFET is its own body diode, choose **Single Die** and the extracted model contains one thermal network.

> **Note:**
>
> Manufacturers usually specify in the datasheet if there is a body diode. You can also determine the number of dies by whether the datasheet contains joint or separate thermal impedance plots for MOSFET and diode.

2. The **Nominal Working Point Values [2/12]** dialog box sets nominal values at the device's working point. Ideally, these are the operating conditions for the device in the design. Practically, these values should be the test conditions on the manufacturer data sheet specified for the switching energy loss data. Pick the values of **VDD** and **ID** mentioned in the table next to the value of the turn-on switching loss, **Eon**. Select the temperature for which you have the output characteristics in the plots section. **VGS_on** and **VGS_off** are gate voltages used in manufacturer's test circuits in which switching dynamics are measured. In the datasheet, **VGS_on** and **VGS_off** are next to the value of **Eon**. **Cin**, the input capacitance, found in the datasheet. Click **Next** to continue.
3. On the **Breakthrough Values [3/12]** dialog box, select **Disable Breakthrough Model**, as the breakdown characteristics data is rarely available in a datasheet. Click **Next** to continue.
4. The **Half-Bridge Test Circuit Condition [4/12]** dialog box sets up the test circuit under which the model's dynamic characterizations are measured. It is important to have the same test condition used by the vendor when the datasheet values are measured.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The required data can generally be found in the datasheet and device application note from the vendor.

- **Rg** is the internal gate resistance found in the datasheet. This value is an initial value during the dynamic characterization. It will not be the internal gate resistance the characterized device ends up with.
- **R_tot** is generally omitted in the datasheet; **5% of Rds in [2/12]** is a good default for MOSFET. If **R_tot** is provided in the datasheet and the device is a MOSFET module that contains two MOSFETs in series between two external pins, use half of the **R_tot** value in the datasheet.
- **Rg_on** and **Rg_off** are next to the values of the switching energies **Eoff** and **Eon**.

When finished setting values, click **Next** to continue.

5. Use the **Transfer Characteristic Id=f(Vgs) [5/12]** dialog box to parameterize the threshold voltages and transconductance of the MOSFET. Enter the data with SheetScan.

   To do so, click **Load characteristics from Dataset Manager** ( ) above the table and click **SheetScan**. Load the graphs in graphic files captured from the manufacturer data sheet using **Picture** > **Load picture**.

   a. Select a coordinate system (**Coordinate System** > **New**) and define it on the plot. To do so, you must select three points. Typically, you should use the bottom-right, bottom-left and top-right points of the plot grid. Click **Point1**, **Point2** and **Point3** and select the corresponding points on the graph. Enter the corresponding X-values and Y-values for these points (read from the plot's X-axis and Y-axis labels) in the table and click **OK** to finish defining the coordinate system.

   b. To define the first characteristic, select **Curve** > **New** and give names to the X-axis and Y-axis in the **Curve Settings** dialog box. Click **OK** when finished.

      > **Note:**
      >
      > You can save your SheetScan work for later adjustment and reuse.

   c. Make sure to note the temperature given on the plot. Then select several points on the curve starting with the lowest X-value. Pick at least four points. When done, click **File** > **Export** and click **Dataset** in the resulting **Save** dialog box.

   d. If a plot at a different temperature is available, repeat steps **a**, **b**, and **c** to record additional data.

   e. When finished, click **File** > **Exit** to exit SheetScan, saving the scan setup information if desired.

   f. In the **Datasets** dialog box, select the data you want to use at the nominal temperature and click **Done**. The data is transferred to the **Characteristic Data** table in the **Transfer Characteristic [5/12]** dialog box. Make sure to enter the correct values for temperature and **Vce**, which you recorded during the SheetScan measurements.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

g. If you also recorded plot data for a different temperature, click **Add new characteristic** ( ) in the top right corner, and click **Load characteristics from Dataset Manager** ( ) above the table to load the additional data. Select the plot in the first tab for the **Nominal Temperature** field of the **Fitting Characteristic Order** panel. If you added a second set of data for a different temperature, select this plot for the **Different Temperature** field, or select **Not Used**, if data is available only for one temperature.

h. Click **Start Fitting** to fit the characteristics and examine the resulting plot to check the match of the fit. Click **Next** to continue.

6. Use the **Output Characteristic Id=f(Vds) [6/12]** dialog box to parameterize more of the MOSFET's output characteristics. Ideally, enter the output characteristics at **Vgs=VGS_on (Full Saturated Branch**) and for **Vgs** at a lower voltage (**Semi Saturated Branch**), for both the nominal temperature and at a different temperature, using SheetScan as described in step **5**. Make sure to note the value of **Vgs** and the corresponding temperature for each curve.

   a. Add up to four characteristics, making sure to identify them with the correct values of VGS and temperature.

   b. Select the plot at the nominal temperature and with **Vgs=VGS_on** for the **Full Saturated Branch (Tnom)** field in the **Fitting Characteristic Order** panel, and one at a lower value of **Vgs** for the **Semi Saturated Branch (Tnom)** field, or select **Not Used**, if the data is not available. Repeat this at a different temperature for **Vgs=VGS_on** and at a lower Vgs in the corresponding **Full Saturated Branch (Tdiff)** and **Semi Saturated Branch (Tdiff)** fields. The fit will be better if the current does not saturate much for the semi-saturated curve, but the fitting algorithm will work with either option.

   c. Click **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

7. Use the **Drain-Source Diode Characteristic If=f(Vf) [7/12]** dialog box to parameterize the DC characteristics of the diode. Ideally, you should enter the diode forward characteristic If(Vf) at both the nominal temperature and at a different temperature using SheetScan as outlined in step **5**. Make sure to note the value of the temperature for each curve.

   a. Generate two characteristic data sets, one at the nominal temperature and one at a different temperature, being sure to identify them with their corresponding temperatures.

   b. Select the plot at the nominal temperature in the **Nominal Temperature** field of the **Fitting Characteristic Order** panel, and the plot at the different temperature in the **Different Temperature** field, or select **Not Used** if the data is not available.

   c. Click **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

8. Use the **MOSFET Thermal Model [8/12]** dialog box to parameterize the thermal impedance of the MOSFET. Do one of the following:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- If the data sheet provides extracted values for **ri** and **ti**, which is the case for International Rectifier, enter these four value pairs in the **Use Fraction Coefficients** table, click **Start Fitting**, check the plot and click **Next** to continue.
- If the data sheet does not provide extracted values for **ri** and **ti**, select **Use Transient Thermal Impedance** and enter the plot data for the thermal characteristics using SheetScan per the instructions given in step **5**. The plot will show the impedance as a function of time. Make sure to adjust the scale of the coordinate system in SheetScan to logarithmic if needed. Select **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

> **Note:**
>
> Sample the "single pulse" curve when multiple transient thermal impedance curves are presented in the data sheet.

9. Use the **Drain-Source Diode Thermal Model [9/12]** dialog box to parameterize the thermal impedance of the diode. If **Single Die** is selected in [1/12], this page becomes inactive.
   - If the data sheet provides extracted values for **ri** and **ti**, which is the case for International Rectifier, enter these four value pairs in the **Use Fraction Coefficients** table, click **Start Fitting**, check the plot, and click **Next** to continue.
   - If the data sheet does not provide extracted values for **ri** and **ti**, select **Use Transient Thermal Impedance** and enter the plot data for the thermal characteristics using SheetScan per the instructions given in step **5**. The plot shows the impedance as a function of time. Make sure to adjust the scale of the coordinate system in SheetScan to logarithmic if needed. Select **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.
10. Use the **Energy Characteristics [10/12]** dialog box to parameterize switching loss at nominal working point and energy correction coefficients for estimating switching energy in different working conditions. For the Power MOSFET (Average) model, provide as much data as possible on this page. It would improve energy loss estimation at conditions other than nominal working point.

E_{on}	Turn-On Switching Loss
E_{off}	Turn-Off Switching Loss
E_{rr}	Reverse Recovery Switching Loss

The data must be entered following a predefined format:

- The first row is the set of data under the nominal working conditions (required).
- Each subsequent row may have only one change in the working condition; for example, temperature T is different from the value in the nominal values row. The required

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

variation for each row is listed in the **Note** column and is described here, same as for the IGBT Average model.

If any of these data sets are not available, clear the **Enable** check box for this row; the corresponding coefficients will remain **0**.

Click **Extraction** to calculate dynamic dependency. **Show Log** is enabled after extraction is completed so you can view the results after fitting.

When finished, click **Next** to continue characterizing the device.

11. Use the **Validation [11/12]** dialog box to check the accuracy of the parameterized component through the validation of switching energies. **Enable** the conditions that need to be checked and click **Validate**. Click **Next** to continue.
12. Use the **Model Parameters [12/12]** dialog box to browse the extracted parameters.
    a. To generate an **.sml** file of the model, click **Create SML**.
    b. To place a characterized device in the Twin Builder project, click **Place Component** and click **Finish**.

Tutorial for Characterizing a Power Diode

Use the Power Diode Device Characterization Wizard to parameterize a power diode model to match both the DC and dynamic characteristics of a device using information from the manufacturer's data sheet. The information available in the data sheet limits the accuracy of the parameterized power diode model if certain characteristics are not provided.

The wizard characterizes a wide array of power diodes. The available information varies from vendor to vendor, and even between devices from the same vendor. The following procedures include vendor-specific information for help in characterizing models for several vendors' devices. If you have any additional questions, contact Ansys support.

> **Note:**
>
> Details about the Power Diode Model are in the Basic Elements section in the Components help.

Related Topics

Preliminary Considerations Using Data Sheet Information

Using the Device Characterization Wizard for a Power Diode

Advanced Settings for Characterizing a Power Diode

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Preliminary Considerations using Power Diode Data Sheet Information

Before starting a device characterization, study the manufacturer's data sheet for the power diode to characterize. Each manufacturer has a unique design and naming convention for their data sheets. Different manufacturers also measure device characteristics differently.

- First, determine the nominal conditions for the device. Ideally, these are the conditions at which the device will be operated. Practically, these conditions should be those for which you have the most information closest to the desired operating conditions. Search the data sheet and look at all available data to decide on the nominal point. The measurement conditions for the dynamic characteristics (**Qrr, Trr**) are generally good choices.
- DC characteristics are parameterized using the forward characteristics, found as plots in the sheet.
- Dynamic behavior is characterized using the reverse recovery characteristics, found either in a table or read from a plot.

> **Note:**
>
> The Power Diode wizard documentation includes information and recommended settings for several manufacturers. Make sure to consult these sections in the documentation when characterizing a device.

Related Topics

Using the Device Characterization Wizard for a Power Diode

Using the Device Characterization Wizard for a Power Diode

> **Note:**
>
> Details about the Power Diode Model are in the Basic Elements section in the Components help.

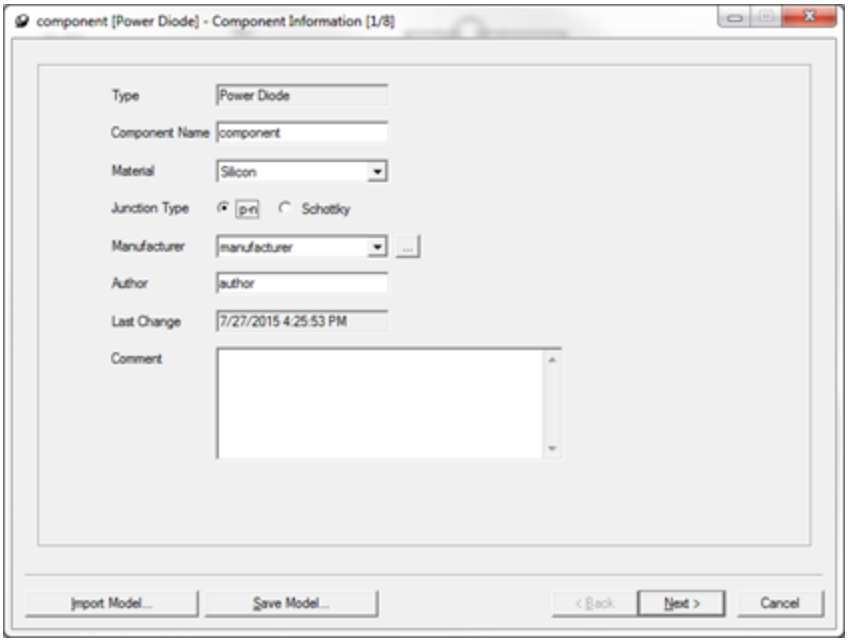
The following procedures use an Infineon device as an example. The process is similar for other manufacturers' devices.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

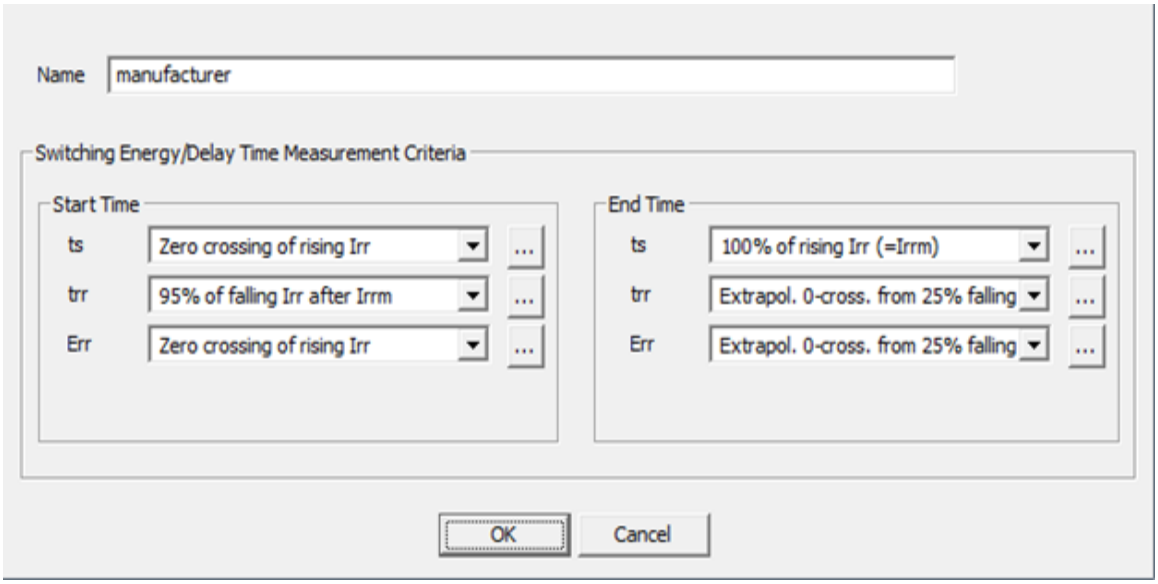
- Click **Save Model** at any time during the characterization process to save your progress.
- A characterization can be loaded on the **Import Model** tab.

1. In the **Component Information [1/8]** dialog box, enter the component name, material, and junction type and define the manufacturer data.
   - **Material** – Si and SiC Diodes are currently supported.
   - **Junction Type** – p-n junction or Schottky barrier diode (SBD). Note that almost all commercial SiC diodes are SBDs.



   - **Manufacturer Data** – These are measurement criteria of dynamic characteristics defined by manufacturers. The information is usually found in the data sheet or device application note from the manufacturer. The documentation describes specific settings for several manufacturers. These are the measurement conditions for the dynamic characteristics. For example, Infineon uses the following settings:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

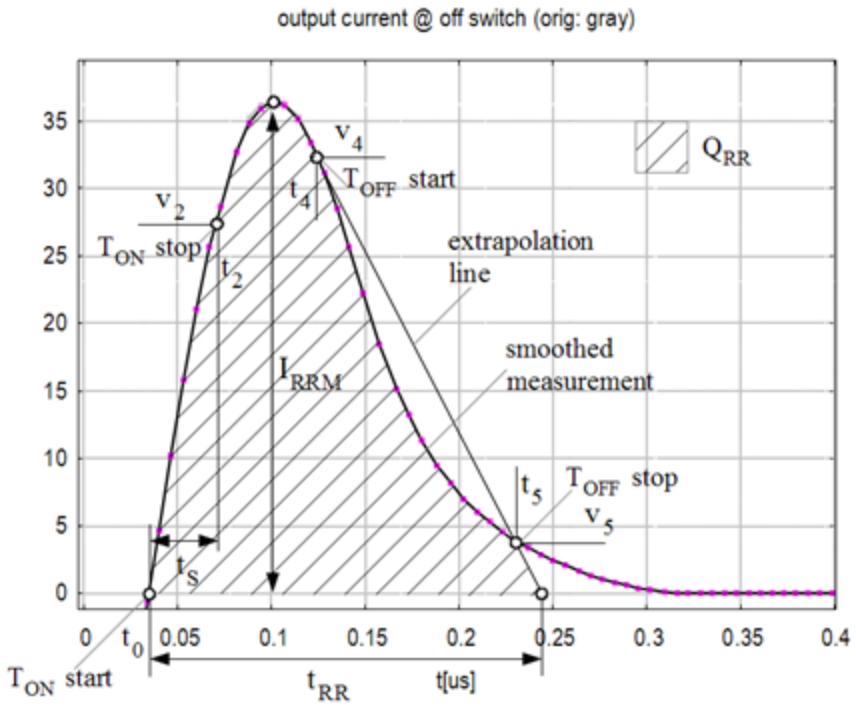


This manufacturer data is critical for the correct characterization of the device. Because each manufacturer uses its own standard to measure the diode reverse recovery parameters, if a dynamic measurement is included as fitting goal in [6/8] of the wizard, it is important to measure the quantity the same way during characterization, which fits the model to output the same.

The most important setting in this dialog box is **trr**. It always starts at the current zero crossing point, while the end point is extrapolated on two helping points, **t4** and **t5** , as in the following figure. Enter **t4** in the first input box next to **trr** and **t5** in the second, according to the vendor's specification. **Qrr** uses the same start and end time as **trr**.

The start and end times of **ts** and **Err** are often not specified by vendors. In this case, use the default settings for these two, as long as they are not included as fitting goals in [6/8] of the wizard.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. The **Nominal Working Point Values [2/8]** dialog box sets nominal values at the device's working point. As mentioned above, these are ideally the operating conditions for the device in the design. Practically, these values should be the test conditions on the manufacturer's data sheet specified for the switching time and energy information obtained and used for dynamic parameters extraction. These are the conditions for which the parameterized device will best match the measured characteristics. For Infineon, generally pick the measurement conditions for the dynamic characteristics. Pick the values of **VR**, **IF** and **diF/dt** mentioned in the table next to the value of the reverse recovery time, **trr.** Select the temperature for which you have the forward characteristics in the plot's section. Use the default breakdown characteristics, as the breakdown characteristics data is rarely available in a data sheet. Click **Next** to continue.

   > **Note:** Breakthrough limits are by default set to very high out-of-range values to disable the breakthrough model. The default values do not represent the operating range of the model.

3. Use the **Diode Forward Characteristic [3/8]** dialog box to parameterize the forward current characteristics of the diode. Enter the data with **SheetScan**. To do so, click **Load characteristics from Dataset Manager** ( ) above the table and click **SheetScan**. Load the graphs in the graphic files captured from the manufacturer data sheet using **Picture** > **Load picture**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For an Infineon device, the data sheet plot shows $I_F=f(V_F)$.

a. Select a coordinate system (**Coordinate System** > **New**) and define it on the plot. To do so, you must select three points. Typically, you should use the bottom-right, bottom-left and top-right points of the plot grid. Click **Point1**, **Point2** and **Point3**, and select the corresponding points on the graph. Enter the corresponding X-values and Y-values for these points (read from the plot's X-axis and Y-axis labels) in the table, and click **OK** to finish defining the coordinate system.

b. To define the first characteristic, select **Curve** > **New**, and give names to the X-axis and Y-axis in the **Curve Settings** dialog box. Click **OK** when finished.

c. Make sure to note the temperature given on the plot. Then select several points on the curve starting with the lowest X-value. Pick at least four points. When done, click **File** > **Export** and click **Dataset** in the resulting **Save** dialog box.

d. If a plot at a different temperature is available, repeat steps **a**, **b**, and **c** to record additional data.

e. When finished, click **File** > **Exit** to exit SheetScan, saving the scan setup information if desired.

f. In the **Datasets** dialog box, select the data to use at the nominal temperature and click **Done**. The data is transferred to the **Characteristic Data** table in the **Diode Forward Characteristic [3/8]** dialog box. Make sure to enter the correct values for temperature, which you recorded during the SheetScan measurements.

g. If you also recorded plot data for a different temperature, click **Add new characteristic** ( ) in the top right corner, and click **Load characteristics from Dataset Manager** ( ) above the table to load the additional data. Select the plot in the first tab for the **Nominal Temperature** field of the **Fitting Characteristic Order** panel. If you added a second set of data for a different temperature, you can select this plot for the **Different Temperature** field, or select **Not Used**, if data is available only for one temperature.

h. Click **Start Fitting** to fit the characteristics and examine the resulting plot to check the match of the fit. Click **Next** to continue.

4. Use the **Diode Thermal Model [4/8]** dialog box to parameterize the thermal impedance of the diode. Do one of the following:
   - If the data sheet provides extracted values for **ri** and **ti**, enter these four value pairs in the **Continued/Partial Fraction Coefficients** table, click **Start Fitting**, check the plot, and click **Next** to continue.
   - If the data sheet does not provide extracted values for **ri** and **ti**, select **Use Transient Thermal Impedance**, and enter the plot data for the thermal characteristics using **SheetScan** per the instructions given in step **3**. The plot shows the impedance as a function of time. Make sure to adjust the scale of the coordinate system in SheetScan to

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

logarithmic if needed. Select **Start Fitting** to start the fit, and examine the plot to check the match of the fit. Click **Next** to continue.

> **Note:**
>
> Sample the "single pulse" curve when multiple transient thermal impedance curves are presented in the data sheet.

5. Use the **Junction Capacitance Characteristics [5/8]** dialog box to parameterize the diode junction capacitance vs. the reverse voltage.

   If the data sheet provides junction capacitance vs. reverse voltage characteristics, clear **Disable Cj Characteristics,** and enter the junction capacitance data with **SheetScan** per the instructions given in step **3**. Select **Start Fitting** to start the fit and examine the plot to check the match of the fit. Click **Next** to continue.

   For Schottky barrier diodes, when **Schottky** is selected for the junction type, this characteristic is crucial for correct dynamic behavior.

6. Use the **Dynamic Model Input [6/8]** dialog box to parameterize the dynamic characteristics of the power diode. Select to fit for the reverse recovery charge (**Qrr**) and time (**Trr**). The nominal point has to be fit, so the data must be available. The **Advanced Settings** button allows for more control over the characterization process, but it should not be necessary to change for most devices. Check the Advanced Settings documentation for more information.
   a. Click **Measurement** to open the **Measurement Data** dialog box, and make sure the settings correspond to the ones for the manufacturer of the device being characterized.
   b. If data is available at different temperature, add it to the **dT** row.
   c. If data is available at a different value of current $\mathbf{I_F}$, add the lower value of current to the **nI** row and the higher value of current to the **pI** row.
   d. If data is available at a different value of current rate of change $\mathbf{di_F/dt}$, add the lower value of $\mathbf{di_F/dt}$ to the **nX** row and the higher value of $\mathbf{di_F/dt}$ to the **pX** row.
   e. Click **Extraction** to start the fit and click **Next** to continue.

Advanced Settings

- **H_ON** - On-switch signal acquisition time
- **H_OFF** - Off-switch signal acquisition time

  For low and medium power devices, the default settings should be sufficient to ensure correct data measurement. For high power devices, the **H_ON** and **H_OFF** values might be increased (2-5 times).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- The parameter extraction process uses three optimization routines to determine values for the dynamic device parameters. The first routine is an iterative loop that uses a 1D search method to progressively refine its approximation. To enable it, set **LOOPS_A** to an integer number larger than 0 (zero). **MASKPAR_A** contains the names of the optimization parameters to reach a good convergence. **RESORD_A** sets how the residue is defined: **0** (zero) for the maximum error, **1** for the average error, and **2** for the root mean square error. **RESTOL_A** defines the value under which the residue must get to leave the loop with a good solution.
- The second and the third loops (B and C) use a Jacobian matrix method. **LOOPS_B** and **LOOPS_C** set the number of pre-loops which are taken for the second and third loop: usually **1** should suffice. **MASKPAR_B** and **MASKPAR_C** contain the names of the parameters changed by the characterization tool to find a better fit during the second and third loops. **MATRIX_B** and **MATRIX_C** set the maximum number of Jacobian recalculations, while **ZEROFN_B** and **ZEROFN_C** set the maximum number of constant Jacobian calculations. **RESLOC_B** and **RESLOC_C** set the maximum number of relaxations that occur within a constant Jacobian calculation. **RESORD_B** and **RESORD_C** set how the residue is defined: **0** for the maximum error, **1** for the average error, and **2** for the root mean square error. **RESTOL_B** and **RESTOL_C** define the value under which the residue must get to leave the second and the third loops with a good solution. If it is zero, it gets this value from the worksheet

7. Use the **Dynamic Parameter Validation [7/8]** dialog box to validate the dynamic extraction. The actual values of the switching times and energies for the parameterized device will be calculated. **Enable** the conditions that need to be checked and click **Validate**. Click **Next** to continue.
8. Use the **Model Parameters [8/8]** dialog box to browse the extracted parameters.
   a. To generate an **.sml** file of the model, click **Create SML**.
   b. To place a characterized device in the Twin Builder project, click **Place Component** and click **Finish**.
   c. To generate a test circuit using the characterized device, click **Testcircuit** and click **Finish**.

   The generated test circuit contains an RC snubber circuit. The RC values are calculated by the following equations:

$$C_s = 4 * C_0$$

   and

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$R_s = 4\sqrt{\frac{L_d}{C_s}}$$

in which C_0 is the zero voltage junction capacitance of the diode, with parameter name **C0_JNCT** in the diode model; L_d is the inductor in the same test circuit which is calculated based on specified $\frac{di}{dt}$.

Advanced Settings for Characterizing a Power Diode

Click **Adv. Settings**in the **Dynamic Model Input** [6/8] dialog box to access advanced settings of power diode characterization. There are two tabs: **Extraction Settings** and **Model & Goal Settings**.

- **Extraction Settings**

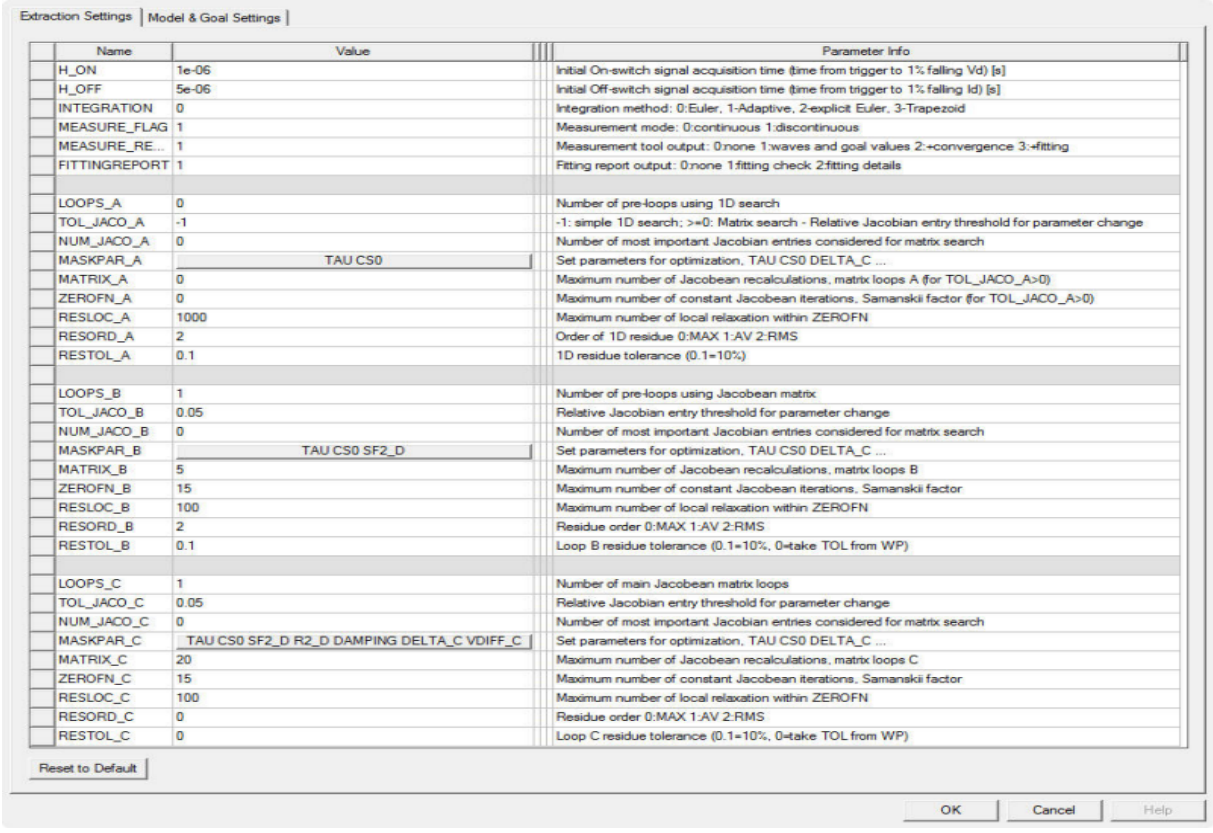


Figure 7-3 Advanced Parameter Setting Dialog Box

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

These settings allow for more control over the characterization process, but it should not be necessary to change them for most devices. The diode model used for characterization also appears as a sub model of the dynamic IGBT and MOSFET models.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td>H_ON and H_OFF</td><td>H_ON and H_OFF are the initial settings for the On-switch and OFF-switch signal acquisition times. The actual Hon and Hoff values change during the characterization. Hon and Hoff derive from the measured switching times and length of the reverse recovery. 5x…10x of the On-switch/reverse recovery time should be a good value. For SiC devices the values should be much smaller than for Si devices. If the initial values are not good, the extraction process will not be able to determine the sensitivity of parameter changes to the output signal and the extraction will not start to converge. You can observe the convergence of the extraction on the Fitting info page and adjust the initial values for H_ON and H_OFF if necessary.</td></tr>
<tr><td>INTEGRATION</td><td>Integration method (0 is best for switching devices).</td></tr>
<tr><td>MEASURE_FLAG</td><td>Measurement mode (1 allows for stop of measurement cycles as soon as a good result is reached).</td></tr>
<tr><td>MEASURE_REPORT</td><td>Measurement tool output (should stay 1).</td></tr>
<tr><td>FITTINGREPORT</td><td>Fitting report output (should stay 1).</td></tr>
<tr><td>LOOPS_A, LOOPS_B, and LOOPS_C</td><td>Approaching the optimal parameter set is an iterative procedure. This iteration is split into three sequenced loops: A, B, and C. Each sequence can repeat any number of times according to LOOPS_x. This is especially helpful in the case of loop A, which is a predefined sequence of one dimensional parameter sweeps. In some special cases an increased number LOOPS_A can result in much better initial parameter values before starting the Newton-Raphson-based matrix loops B and C.
But the repeated 1D parameter sweep will take time and a global convergence is not guaranteed. In most cases, the initial values calculated internally from the nominal conditions are good enough to start loop B immediately without time-consuming loop A.

Default values are LOOPS_A:=0, LOOPS_B:=1 and LOOPS_C:=1.</td></tr>
<tr><td>MASK_PAR_A</td><td>This is a list of dynamic model parameters which take part in the 1D parameter sweep sequence. The input is interpreted as a parameter mask for parameter selection during the characterization.
<table>
<tr><th>Parameter</th><th>Observed impact to goals</th></tr>
<tr><td>TAU</td><td>QRR, IRRM</td></tr>
</table>
</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>
<table>
<tr><td>CS0</td><td>QRR, TRR</td></tr>
<tr><td>DELTA_C</td><td>QRR</td></tr>
<tr><td>SF2_D</td><td>TRR</td></tr>
<tr><td>R2_D</td><td>QRR, TRR</td></tr>
<tr><td>DAMPING</td><td>QRR, TRR</td></tr>
</table>
A repeated application of loop A helps to find a good initial parameter guess for the following Newton-Raphson-based loops B and C.
The above table also shows the most likely parameter-to-goal impact which is useful in case of manual parameter optimization. If a parameter appears sequenced, then different combined and direct impacts are tested. In some cases diodes are already characterized just after loop A.
</td></tr>
<tr><td>MASK_PAR_B and MASK_ PAR_C</td><td>
Lists of dynamic model parameters which take part in the Newton-Raphson-based parameter optimization by error minimization. The following parameters are recognized. The parameters included by default are in bold letters.
<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td>TAU</td><td>This parameter has direct feed through to the reverse recovery time. All devices must include this parameter.</td></tr>
<tr><td>CS0</td><td>Zero voltage semiconductor depletion capacitance.

You can take the data sheet value as the initial value. If this value is not given, zero input triggers an initial guess calculation derived from the nominal values.

CS0 and DELTA_C get major importance for Shottky junction devices because the minority excess charge is zero and the reverse recovery wave form is caused by the displacement current of the depletion capacitance with its parameters CS0 and DELTA_C.</td></tr>
<tr><td>DAMPING</td><td>Calculated from the stray inductance and the feedback (Miller) capacitance, each component gets an internal RC- snubber parallel to the output terminals. For a rough characterization, set DAMPING:=1 and leave it untouched.
DAMPING is useful for solution refinement especially in loop C.</td></tr>
<tr><td>R1_D</td><td>For reverse recovery current two templates are predefined. In and the meaning of the shape parameters R1_D, R2_D, R3_</td></tr>
</table>
</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Parameter	Information
	Parameter / **Meaning** **D**, **SF1_D**, and **SF2_D** are shown in the figures as r_1, r_2, r_3, SF_1 and SF_2. The templates are scalable so that very different reverse responses can be modeled. The point t_B, r_1 marks the transition from exponential current wave at discharging the diffusion capacitance to a sinusoidal approach to the peak current. The range is 0 … 0.995 but zero is the best setting to avoid voltage oscillation at the transition point. **R1_D** is excluded from parameter optimization. **R2_D**: **R2_D** marks the inflection point r_2 of the wave form. It can be smooth transition from cosine to an exponential decay in case of **TYPE_FWD:=2 ()** or a kink in case of **TYPE_FWD:=3 ()**, if the measurements are available. Otherwise, **TYPE_FWD:=2** is preferred. To get a tradeoff between **TRR** and (**QRR** or **IRRM**), include **R2_D** in the optimization. R3_D: This shape parameter has to be set according to the measurement conditions and thus **R3_D** is excluded from parameter optimization. It is shown as r_2 in the plots SF1_D: Meaningful only for **Irr** wave forms with linear current tail according to SF_1 in to control the softness and thus the voltage overshoot at off switch. Together with **R2_D**, **SF1_D** provides additional freedom to fit for **TRR**, **QRR** and **IRRM** together. **SF2_D**: The meaning depends on the freewheeling diode setting with exponential or linear current tail, shown as SF_2 in plots. For **TYPE_DIODE:=3 ()**, **SF_2** is merely a resulting value without specific meaning. But for **TYPE_DIODE:=2 ()**, this value is the commonly used soft factor. The larger the soft factor **SF2_D**, the smaller the current overshoot gets and so does the reverse recovery time **TRR**. In most cases **SF2_D** can stay at default. If the optimization for loss energies and delay times together becomes difficult or the process of optimization takes much more time than expected, then the inclusion of **SF2_D** may solve the problem. The optimization result should be in the range 1 … 7.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

<table>
<tr><th>Parameter</th><th>Information</th></tr>
<tr><td></td><td>
<table>
<tr><th>Parameter</th><th>Meaning</th></tr>
<tr><td>DELTA_C</td><td>DELTA_C is the relative minimum depletion capacitance $\delta_c = \frac{C_{s.min}}{C_s(0V)}$. If the C(V) characteristic is given, you can derive DELTA_C from the curve. If not, the input zero triggers an internal initial guess for this value.</td></tr>
<tr><td>VDIFF_C</td><td>Diffusion voltage of the depletion capacitance.</td></tr>
</table>
<table>
<tr><th>Goal</th><th>Meaning</th></tr>
<tr><td>ERR</td><td>Energy loss for the diode’s reverse recovery.
Measurement conditions differ and can be set by users.</td></tr>
<tr><td>TRR</td><td>Reverse recovery time. Measurement conditions are different for manufacturers.</td></tr>
<tr><td>IRR</td><td>Reverse recovery absolute current peak.
The magnitude depends on the soft factor settings.</td></tr>
<tr><td>QRR</td><td>QRR is the reverse recovery current area $|\int I_{RR}\,dt|$ and thus a combination of displacement charge and excess carrier charge in the free wheeling diode.</td></tr>
</table>
</td></tr>
<tr><td>NUM_JACO_A
NUM_JACO_B, and
NUM_JACO_C</td><td>This parameter defines which entries from the sensitivity matrix are considered for the matrix search:
-1: largest column, 0:apply threshold >0: this number of largest entries
For Loops_A when simple 1D search is used, this parameter is ignored.</td></tr>
<tr><td>TOL_JACO_A
TOL_JACO_B, and
TOL_JACO_C</td><td>This parameter defines the relative threshold for Jacobian entries to be used in the matrix search (when NUM_JACO_x = 0).
For LOOPS_A it can be set to -1 which will switch LOOPS_A to simple 1D search.</td></tr>
<tr><td>RESORD_A, RESORD_B, and RESORD_C</td><td>This is the order of the residual. The number range is 0 .. 2:
0 – The maximum error is the residual value (the error number).
1 – Absolute error average.
2 – Root mean square error value. To observe the system approaching from far the optimum toward the optimum solution number 2 is best suitable. Finally we claim that every goal value has a smaller error than a certain limit. Here we have to use 0. This is the reason for RESORD_C:=0 in every</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Parameter	Information
	characterization.
RESTOL_A, **RESTOL_B**, and **RESTOL_C**	Desired accuracy of the residual according to the **RESORD** setting. To quickly move the model parameters toward the optimum, the tolerance can be set as large as 0.2 = 20%. The number of necessary loops A, B and C will decrease strongly. It is suggested to use the final accuracy only in the last main loop (usually loop C). Relaxing the accuracy can result in a noticeable reduction of simulation time.
MATRIX_A **MATRIX_B**, and **MATRIX_C**	Within loops A (when matrix search is used) and within loops B and C the recalculation of the Jacobean matrix will be done this number unless the residue falls within the accuracy region. The value for **MATRIX_***x* are the maximum numbers of Jacobean re-calculation, which takes most of the simulation time. In terms of pure mathematics, the recalculation should be done in every iteration cycle to stay close on the direct path to the optimum. It turns out to be a good trade-off between the shortest path and the simulation duration to leave the Jacobean untouched for some iterations before its recalculation by simulation. For Loops_A when simple 1D search is used, this parameter is ignored.
ZEROFN_A **ZEROFN_B**, and **ZWROFN_C**	This is the number of iterations without changing the Jacobean (system) matrix. Only the right side of the balance system—the so-called zero function—is changed in every iteration step. This residue calculation is much faster and drives the parameter changes in roughly the right direction toward the optimum solution (the optimum parameter set). These values are called Samanski factors. For Loops_A when simple 1D search is used, this parameter is ignored.
RESLOC_A **RESLOC_B**, and **RESLOC_C**	Numbers of local residue (under-relaxation values differ between parameters) use before the system is considered to become stiff and hard to solve. After **RESLOC_***x* total iterations for under-relaxation and boosting the main diagonal of the Jacobean, no longer is each parameter treated separately but all together globally the same way (under-relaxation values equal for all parameters), which slows down the convergence considerably but forces the solution path back to the optimum path. The additional convergence control stays unused by large **RESTOL_***x* numbers to save time and to make the optimization feasible. For Loops_A when simple 1D search is used, this parameter is ignored.

- Model & Goal Settings

  Use the **Dynamic Behavior** drop-down list to choose a reverse recovery model. The mapping between the menu options and value of parameter **TYPE_DYN** is:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Dynamic Behavior	TYPE_DYN
Dyn0 (no reverse recovery)	1
Dyn1 (+exponential Irr)	2
Dyn2 (+linear Irr)	3

Description of parameter **TYPE_DYN** and different reverse recovery models can be found in . Note that in the final characterized model, **TYPE_DYN** is added by 10 so it takes external synchronization input for accuracy of switching behavior (see **TYPE_DYN=x+10** in the detailed documentation). That is why we see value of **TYPE_DYN** is **11**, **12** or **13** in characterized power diodes.

You can choose dynamic fitting goals in **Select Goals to Display**. The checked goals appear in the table of dialog box **Dynamic Model Input** [6/8], so it can be included as fitting goals during extraction.

Tutorial for Characterizing a Thyristor

Use the Thyristor Device Characterization Wizard to parameterize a thyristor model to match both the DC and reverse recovery characteristics of a device using information from the manufacturer's data sheet.

The Thyristor model has digital gate ignition, so gate related dynamics cannot be characterized. However, delay time **Tgd** and turn-off time **Tq** are modeled as timers in the model, which can be set according to data sheet values.

The information available in the data sheet limits the accuracy of the parameterized thyristor model if certain characteristics are not provided in the data sheet.

The wizard can characterize a wide array of thyristors. The available information varies from vendor to vendor, and even between devices from the same vendor. If you have any additional questions, contact Ansys support.

> **Note:**
>
> Details about the Thyristor Model are in the Basic Elements section in the Components help.

Related Topics

Preliminary Considerations Using Thyristor Data Sheet Information

Using the Device Characterization Wizard for a Thyristor

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Preliminary Considerations using Thyristor Data Sheet Information

Before starting a device characterization, study the manufacturer's data sheet for the thyristor to characterize. Each manufacturer has a unique design and naming convention for their data sheets. Different manufacturers also measure device characteristics differently.

- First, determine the nominal conditions for the device. Ideally, these are the conditions at which the device will be operated. Practically, these conditions should be those for which you have the most information closest to the desired operating conditions. Search the data sheet and look at all available data to decide on the nominal point. The measurement conditions for the dynamic characteristics **Qrr** are generally good choices.
- DC characteristics are parameterized using the On-State characteristics, found as plots in the sheet.
- Dynamic behavior is characterized using the reverse recovery characteristics, found either in a table or read from a plot.

Related Topics

Using the Device Characterization Wizard for a Thyristor

Using the Device Characterization Wizard for a Thyristor

> **Note:**
>
> Details about the Thyristor Model are in the Basic Elements section in the Components help.

The following procedures use an ABB device as an example. The process is similar for other manufacturers' devices.

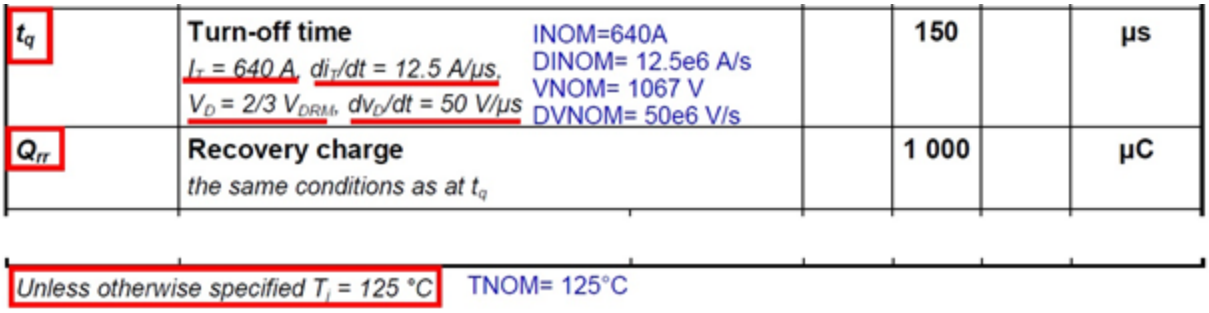
> **Note:**
>
> - Click **Save Model** at any time during the characterization process to save your progress.
> - Open the saved **.ppm** file opened through "Continue device characterization" or load it using the **Import Model** tab.

1. In the **Component Information [1/9]** dialog box, enter the component name, and define the manufacturer's measurement criteria. If thermal or depletion capacitance characteristics are available, clear the **Hide & Disable** option in this page.
2. The **Nominal Working Point Values [2/9]**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Nominal Values** – This section sets nominal values at the device's working point. These values should be the test conditions of **Qrr** or **Tq** on the manufacturer's data sheet. The test condition is later used during model extraction, when model parameters are optimized to match the model's characteristics in the same test condition with datasheet values. When **Qrr** data is available for multiple temperatures, select the **Tj nom** for which you have the on-state characteristics in the plot section.

  I leak_r and **I leak_bl** are the leakage current in reverse and blocking mode. Use default values if not known.



- **Device & Circuit Values** – If specified by the datasheet, input the snubber and stray inductance values; otherwise, use **0**, then snubber parameters will calculate during extraction.

Click **Next** to continue.

3. The **Device and Limit Values [3/9]** dialog box.

- **Trigger Definition**

  Only the digital trigger option is available for now. Input parameters set the trigger condition of the model; values can be found in the datasheet.

  V trig - Gate trigger voltage, model turns on when gate voltage is above this value.

  I hold - Holding current.

  Td_q - Gate turn-on delay time.

  Tq - Circuit commutated turn-off time.

- Limit Values

  Use the default breakthrough characteristics, as the data is rarely available in a datasheet.

4. Use the **On-State Characteristic [4/9]** dialog box to parameterize the on-state current characteristics of the thyristor. At least one set of input data at nominal temperature (input

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

in dialog box [2/9]) is required. A second set of data at a different temperature is preferred in order to characterize temperature dependency parameters.

Choose one of the two ways to input: **Use Data Table** or **Use Equation Coefficients**. If both types of data are available, choose the type with data at both nominal temperature and a different temperature.

- **Use Equation Coefficients** - If provided, these coefficients are usually in a table of A, B, C, D parameters.

Maximum On-state Characteristics

Analytical function for on-state characteristics:

$$v = A + B \cdot i + C \cdot \sqrt{i} + D \cdot \ln(i+1)$$

T_j (°C)	A	B	C	D
25	6.739E-1	3.730E-4	-2.519E-3	7.669E-2
125	5.754E-1	5.005E-4	-4.021E-3	7.925E-2

a. From the drop-down list of analytical functions, select one that matches the equation specified in data sheet.
b. Input the value of A, B, C and D, and set the temperature.
c. In the **Fitting Range** panel, define the effective i_T range of the analytical function. It is usually provided with the function, as shown below.

Durchlasskennlinie 300 A ≤ i_T ≤ 6000 A
on-state characteristic

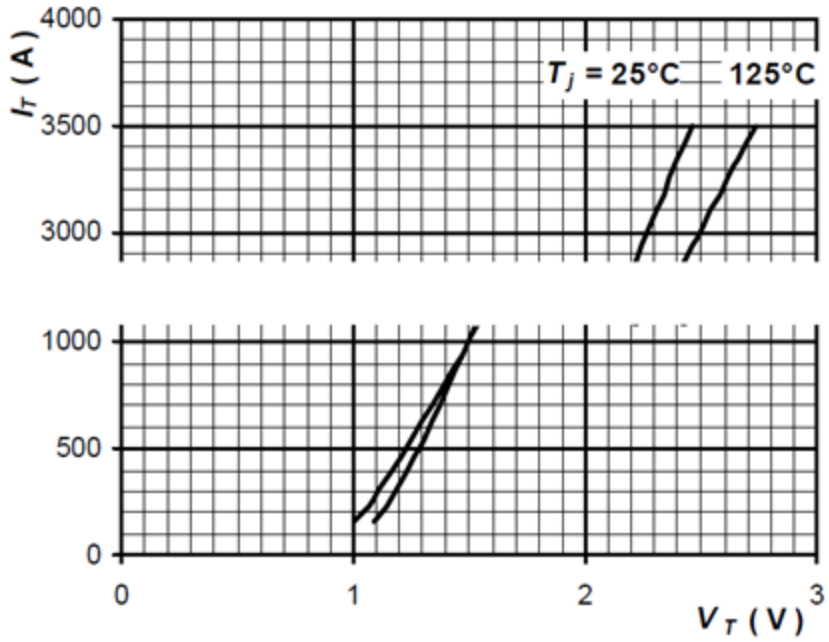
$$v_T = A + B \cdot i_T + C \cdot \ln(i_T + 1) + D \cdot \sqrt{i_T}$$

$T_{vj} = T_{vj\,max}$

A= 7,528E-01
B= 1,682E-04
C= -1,035E-02
D= 6,509E-03

You can also look for range of i_T in the on-state characteristics figure. For example, in the plot below, the effective range of i_T is roughly 200A to 3500A. Note that the analytical function is usually not valid close to i_T=0A, so it is important to define the lower range of fitting.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



If none of the above information is available in regards to effective range, click the two arrows to let the wizard estimate proper current range.

d. To add a characteristic at a different temperature, click **Add new characteristic** in the top right, then repeat steps **a**, **b**, and **c**.

Proceed to Fitting the On-State Characteristic.

- **Use Data Table** - Sample data from the on-state characteristics plots.

  Enter the data with SheetScan. To do so, click **Load characteristics from Dataset Manager** above the table and click **SheetScan**. Load the graphs in the graphic files captured from the manufacturer data sheet using **Picture** > **Load**.

a. Select a coordinate system (**Coordinate System** > **New**) and define it on the plot. To do so, you must select three points. Typically, you should use the bottom-right, bottom-left and top-right points of the plot grid. Click **Point1**, **Point2** and **Point3**, and select the corresponding points on the graph. Enter the corresponding X-values and Y-values for these points (read from the plot's X-axis and Y-axis labels) in the table, and click **OK** to finish defining the coordinate system.
b. To define the first characteristic, select **Curve** > **New**, and give names to the X-axis and Y-axis in the **Curve Settings** dialog box. Click **OK** when finished.
c. Make sure to note the temperature given on the plot. Then select several points on the curve starting with the lowest X-value. Pick at least four points. When done, click **File** > **Export** and click **Dataset** in the resulting **Save** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

d. If a plot at a different temperature is available, repeat steps **a**, **b**, and **c** to record additional data.

e. When finished, click **File** > **Exit** to exit SheetScan, saving the scan setup information if desired.

f. In the **Datasets** dialog box, select the data you want to use at the nominal temperature and click **Done**. The data is transferred to the **Characteristic Data** table in the **On-State Characteristic** [4/9] dialog box. Make sure to enter the correct temperature values which you recorded during the SheetScan measurements.

g. If you also recorded plot data for a different temperature, click **Add new characteristic** in the top right, then click **Load characteristics from Dataset Manager** above the table to load the additional data.

- Fitting the On-State Characteristic

a. In the **Fitting Characteristic Order** panel of either tab for input data, select the Nominal Temperature, which is the same as **Inom** input in [2/9].

b. If you added a second set of data for a different temperature, you can select the temperature for the **Different Temperature** field, or select **Not Used**, if data is available only for one temperature. If **Not Used**, the model's static parameters for temperature dependency will be **0**.

c. Click **Start Fitting** to fit the characteristics, and examine the resulting plot to check the match of the fit. Click **Next** to continue.

5. Use the **Thyristor Thermal Model [5/9]** dialog box to parameterize the thermal impedance of the thyristor. If no thermal data is available, or thermal behavior is not interested in simulation, select **No Data Available or Isothermal** to skip this step. Otherwise, if this dialog box is not displayed, make sure **Hide & Disable Thermal Characteristics** in **[1/9]** is cleared. To fit thermal characteristics, do one of the following:

- **Use Fraction Coefficients** - If the data sheet provides extracted values for **Ri** and **Taui**, enter these value pairs in the **Thermal Coefficients** table. Click **Add new point** on the top right of the data table to add orders as provided in data sheet. Click **Start Fitting**, check the plot, and click **Next** to continue.

- **Use Transient Thermal Impedance** - If the data sheet does not provide extracted values for **ri** and **ti**, select **Use Transient Thermal Impedance** and enter the plot data for the thermal characteristics using SheetScan per the instructions given in step **4**. The plot shows the impedance as a function of time. Make sure to adjust the scale of the coordinate system in SheetScan to logarithmic if needed.

Click **Start Fitting**, check the plot, and click **Next** to continue.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Pay attention to the unit of **Ri** or **Zthjc** in the datasheet. Some datasheets provide values in K/kW, while the UI takes value in K/W.

6. Use the **Depletion Capacitance Characteristics [6/9]** dialog box to parameterize the thyristor junction capacitance vs. the reverse voltage. In most cases where such data is not available, skip this step.

   If the data sheet provides junction capacitance vs. reverse voltage characteristics, make sure **Hide & Disable Depletion Capacitance Characteristics** in **[1/9]** is cleared, so this dialog box displays. In dialog box **[6/9]**, clear **Disable Cv Characteristics**, and enter the junction capacitance data with SheetScan. Select **Start Fitting** to start the fit, and examine the plot to check the match of the fit. Click **Next** to continue.

7. Use the **Dynamic Model Input [7/9]** dialog box to parameterize the dynamic characteristics of the thyristor. You can select to fit for the reverse recovery measurements. The nominal point has to be fit, so some data at this working condition must be available.

   By default, reverse recovery charge **Qrr** and reverse recovery time **Trr** are the displayed goals to fit. If only **Qrr** is available in the datasheet, clear the other goals to disable them. If other measurements are available in the datasheet but not displayed in the wizard, for example, reverse recovery peak current **Irm**, click **Advanced Settings** and go to the **Model & Goal Settings** tab to add fitting goals by checking them in the **Select Goals to Display** panel.

   Turn-off time **Tq** does not play a role in dynamic extraction, since the thyristor model has only digital ignition. **Tq** is modeled as a timer set directly by parameter TQ, which is input in dialog box **[3/9]**. Note that **Tq** is not equivalent to reverse recovery time **Trr**: it is usually much longer (3-5 times) than **Trr**.

   The **Advanced Settings** button allows for more control over the characterization process. Some parameters are important for extraction to succeed. See the Advanced Settings documentation for more information.

   a. Click **Measurement** to open the **Measurement Data** dialog box and make sure the settings correspond to the ones for the manufacturer of the device being characterized.
   b. If data is available at different temperature, add it to the **dT** row.
   c. If data is available at a different value of current $\mathbf{I_d}$, add the lower value of current to the **nI** row and the higher value of current to the **pI** row.
   d. Click **Extraction** to start the fit, then click **Next** to continue.

Advanced Settings

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

Some suggested settings are calculated based on **Trr**. When **Trr** is not available, use 0.2***Tq** as a rough estimation of **Trr**.

- **H_MIN** – Minimum step width during extraction. Suggested value: **Trr**/1000. **H_MIN** should be small enough to have good resolution of reverse recovery waveform. Suggested **H_MIN** during extraction is smaller than what should be used in the final simulation because during extraction, dynamic parameters are being tuned and reverse recovery waveform is changing and could be a much smaller **Trr** measurement than the goal value. The **H_MIN** setting here should still give enough resolution in such cases.
- **H_MAX** – Maximum step width during extraction. Suggested value: 100***H_MIN**.
- **PERIOD** – Switching period during extraction. Suggested value: 3***Tq**.
- **H_ON** - Reverse recovery signal acquisition time, should be set to 2-5 times of **Trr**. This parameter is crucial for extraction success.
- **H_OFF** - Not used.
- The parameter extraction process uses three optimization routines to determine values for the dynamic device parameters. Thyristor extraction uses the first routine, which is an iterative loop that uses a 1D search method to progressively refine its approximation. **LOOPS_A** sets the number of loops in this routine; the default value is usually sufficient. **MASKPAR_A** contains the names of the optimization parameters to reach a good convergence, and the default setting enables both parameters, which is usually good. **RESORD_A** sets how the residue is defined: **0** (zero) for the maximum error, **1** for the average error, and **2** for the root mean square error. **RESTOL_A** defines the value under which the residue must get to leave the loop with a good solution.
- The second and the third loops (B and C) use a Jacobian matrix method. These loops are usually not used for thyristor extraction.

8. Use the **Dynamic Parameter Validation [8/9]** dialog box to validate the dynamic extraction. The actual reverse recovery characteristics for the parameterized device are simulated and measured. **Enable** the conditions that need to be checked and click **Validate**. Click **Next** to continue.
9. Use the **Model Parameters [9/9]** dialog box to browse the extracted parameters.
   a. To generate an **.sml** file of the model, click **Create SML**.
   b. To place a characterized device in the Twin Builder project, click **Place Component** and click **Finish**.
   c. To generate a test circuit using the characterized device, click **Testcircuit** and click **Finish**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

8 - Power Module Characterization Wizard

The Power Module Characterization wizard aids in the building of various types of power module models so they can operate to a device manufacturer's specifications. The wizard accepts inputs for various quantities available from the device manufacturer and fits a numerical model to the data to provide accurate device simulation over a range of device excitations and thermal conditions.

- Click **Twin Builder** > **Characterize Device** > **Power Modules** to start the Power Module Characterization Wizard.

  > **Note:**
  >
  > Microsoft .NET version 2.0, Service Pack 1 or later is required to open and use the Power Module Characterization Wizard.

The wizard includes a library of **DC-DC Converter** models from various manufacturers. These models can be used as is, or as the basis for developing new, user-defined models. For a detailed discussion of the approach to DC-DC Converter modeling, see *Modeling of DC-DC Converters Based on Hybrid Wiener-Hammerstein Structure*.

Related Topics

Using System Library Power Modules

Modifying System Library Power Modules

Creating a Power Module Behavioral Model

Using System Library Power Power Module Models

The Power Modules Characterization wizard includes an extensive library of DC-DC converter models from various manufacturers. You can view the characteristic data for these models in the wizard. You can also to import these models into Twin Builder projects, to save models as discrete VHDL-AMS model files, and enable generation of sub-architectures for models.

1. To view the characteristics of power module models supplied in the System Library, select the desired models in the **System Libraries** model browser.

   The model editing panel displays the summary tab for the model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

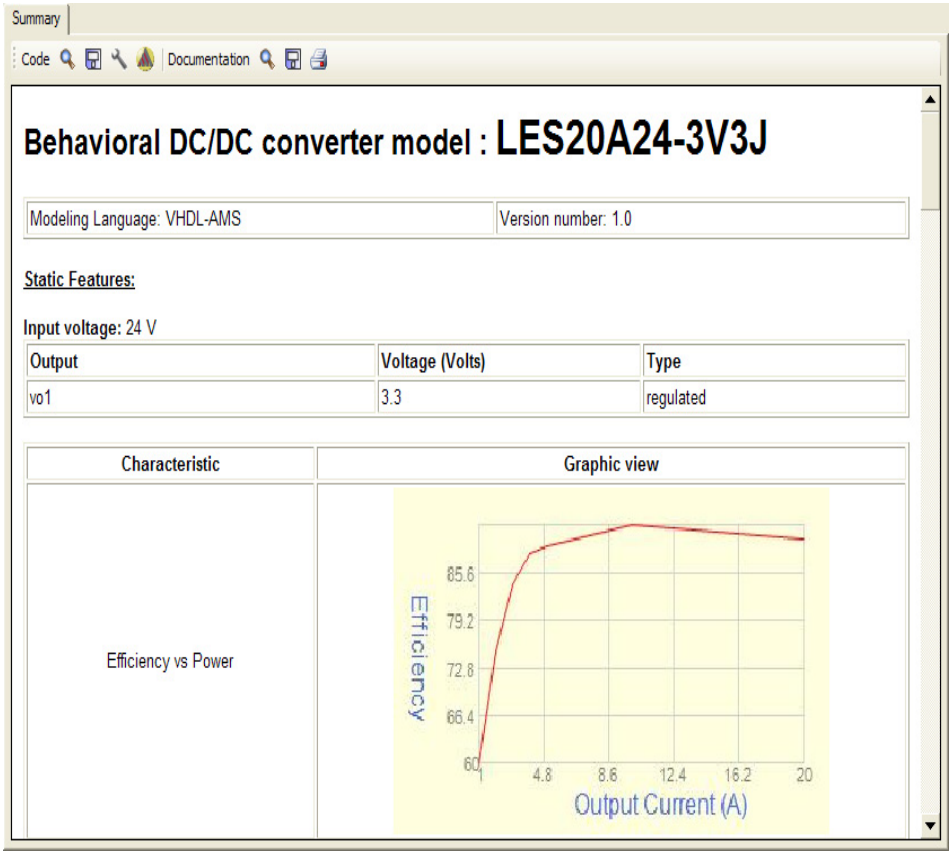
2. To view the model code (read only) in the summary tab, click the **Code** icon.

3. To save the model in VHDL-AMS format, click the **Code** icon. In the resulting dialog box you can enter the model file name and choose a save location.

4. To enable generation of sub-architectures, click the **Code** icon, check **Enable**, then select the sub-architectures to include in the model.

5. Click the **Documentation** icon in the tab title bar to view the documentation in the tab.

   You can also print the documentation with the **Documentation** icon, or export the documentation as an HTML file with the **Documentation** icon.



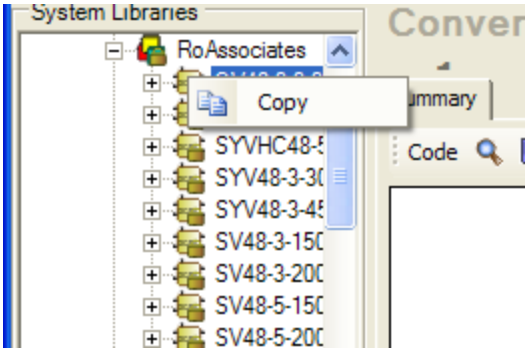
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. To import the model into the active Twin Builder project, click the **Code** icon. The message panel displays **Model sent to Simplorer** upon completion of the operation.

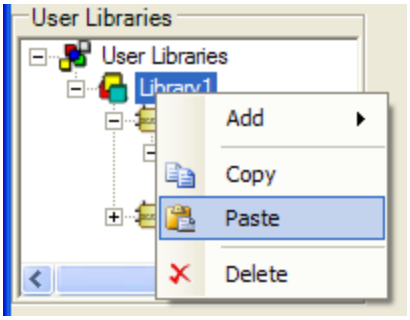
Modifying System Library Power Module Models

The models provided in the DC-DC Converters System Library cannot be modified directly. If you want to modify any of these models you must first place a copy of the desired model in the **User Libraries**. To modify a System Library DC-DC Converter:

1. Right-click the model you want to copy and select **Copy**.



2. Right-click the target library and select **Paste**.



3. Once the model is in the **User Libraries**, you can modify the model as shown in *Creating a Power Module Behavioral Model*. You can import it into Twin Builder when you're done.

Creating a Power Module Behavioral Model

Use the Power Module Characterization wizard to create new behavioral models and model libraries in which to store them.

Related Links

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Creating a New Power Module Model Library

Adding a New Behavioral DC/DC Converter Model

Defining a Static Converter Model

Generating VHDL-AMS Code and Importing the Model into Twin Builder

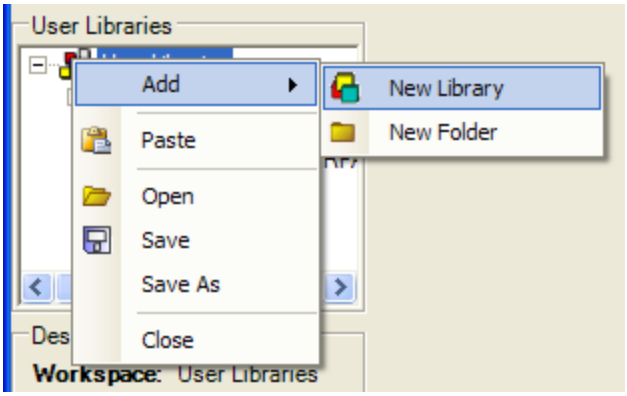
Defining the Converter Model Dynamic Behavior

Defining Event Driven Behavior

Enabling Current Sharing and Thermal Modeling

Creating a New Power Module Model Library

1. Click **File** > **Add New** > **Library** to create a new model library. A library with a default name appears under the **User Libraries** model browser.

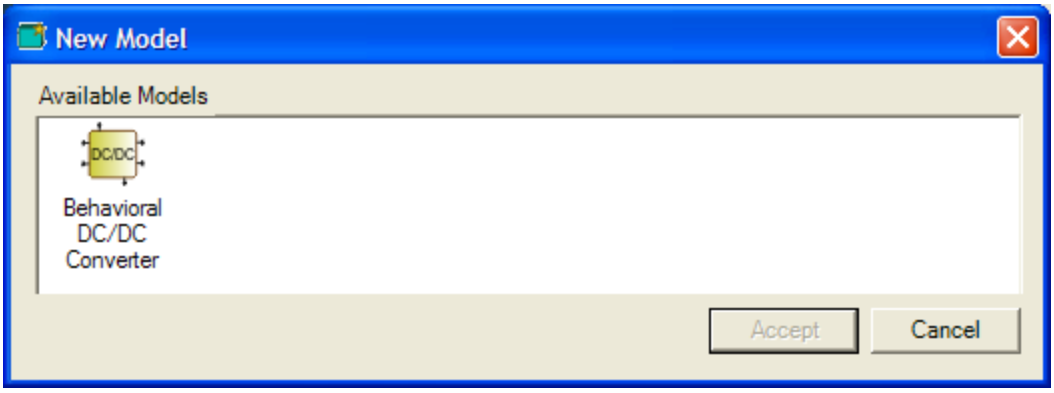


2. Click a library name to select it, then click it again to rename it.

Adding a New Behavioral DC/DC Converter Model

1. Right-click a library in the model browser, then click **Add** > **New Model**. The **New Model** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. Select the **Behavioral DC/DC Converter** model and click **Accept**. A model with a default name appears under the selected library.
3. Click the model name to select it, then click it again to edit the name.

Every newly created DC/DC model has default parameters that correspond to a 12V input and 5V output voltage with a constant efficiency.

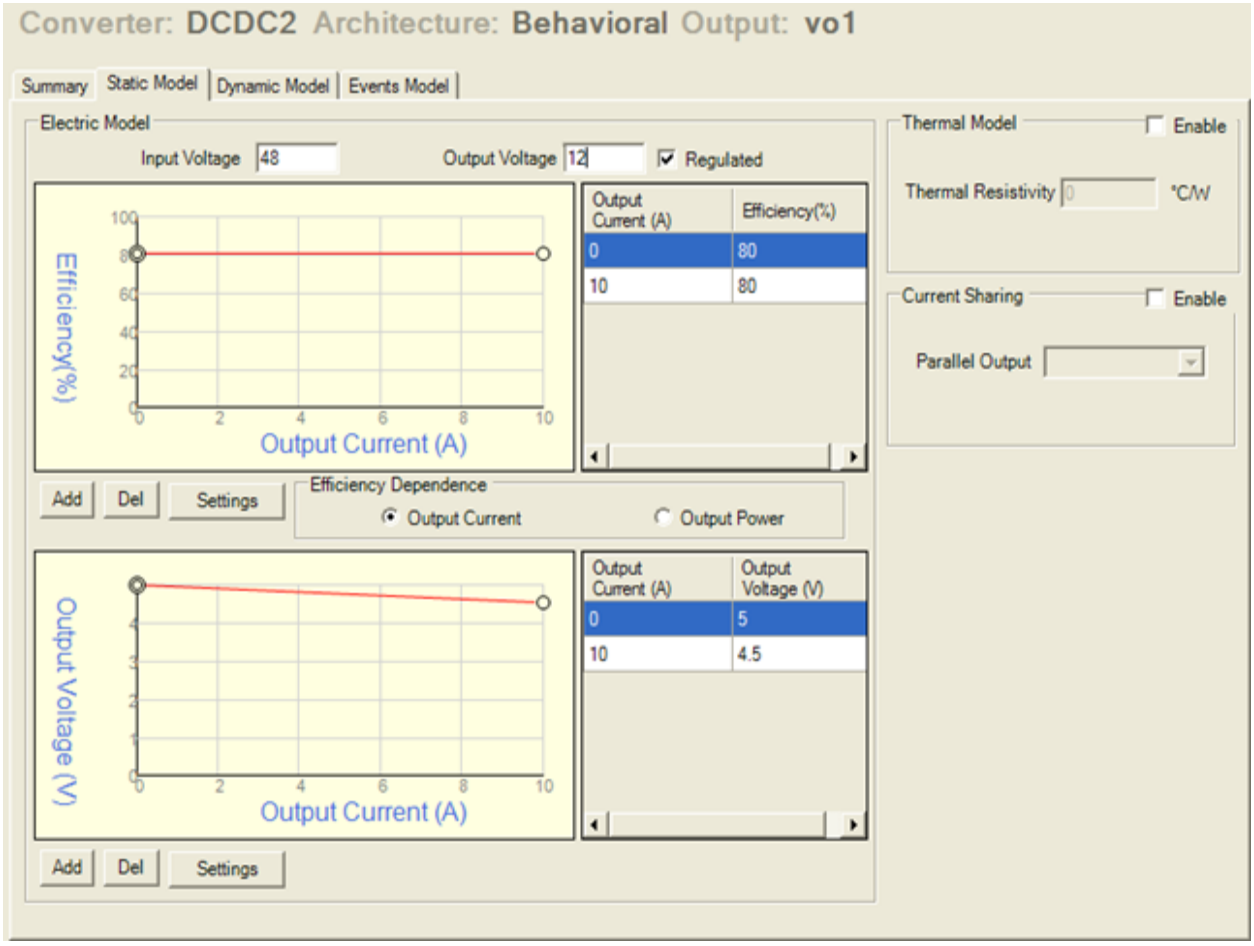
Defining a Static Converter Model

The following example describes how to define a static converter model having a 48V input voltage and a 12V output voltage. The efficiency is assumed to be a constant value.

1. In the **User Libraries** browser, select the DC-DC converter model that you want to edit. An editing panel with four tabs—**Summary**, **Static Model**, **Dynamic Model**, and **Events Model**—appears.
2. Select the **Static Model** tab.
3. On the **Static Model** tab, you can define the basic characteristic of the converter. Change

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

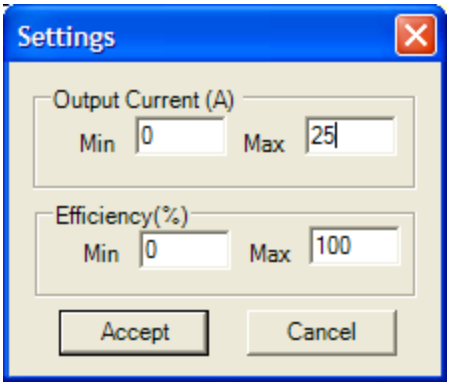
the value of the input voltage to **48V** and the output voltage to **12V**.



The **Static Model** editing panel shows two graphs: Efficiency vs. Output Current and Output Voltage vs. Output Current.

4. The first step to define a characteristic graph is to set its maximum and minimum values. Click **Settings** on the Efficiency vs. Output Current graph. The **Settings** dialog box appears.
5. In the **Settings** dialog box, set the **Output Current (A)Max** value to **25** (we are assuming that the converter supports a maximum output current of 25A). Similarly, enter **100** as the **Efficiency(%)Max** value. This corresponds to a maximum efficiency of 100%. Click **Accept** to close the dialog box.

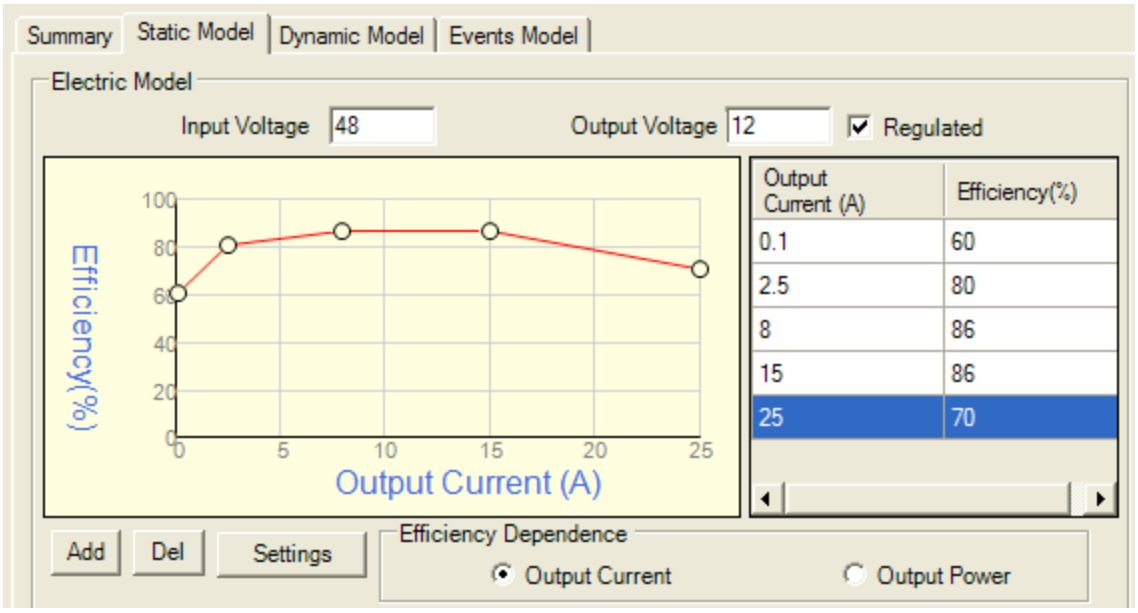
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



6. Click **Add** on the efficiency graph to insert as many points as you need to define the behavior of your converter. You can also right-click in the graph and select **Add**.

   In this example, three more points will be added. Set the values of the points by dragging the added points directly on the graphic, or specify the values directly in the table view. (Note that as you add points, the table entries sort in ascending order of output current magnitude.)

7. Enter the following values for the efficiency curve: **(0.1, 60)**, **(2.5,80)**, **(8,86)**, **(15,86)**, and **(25,70)**.

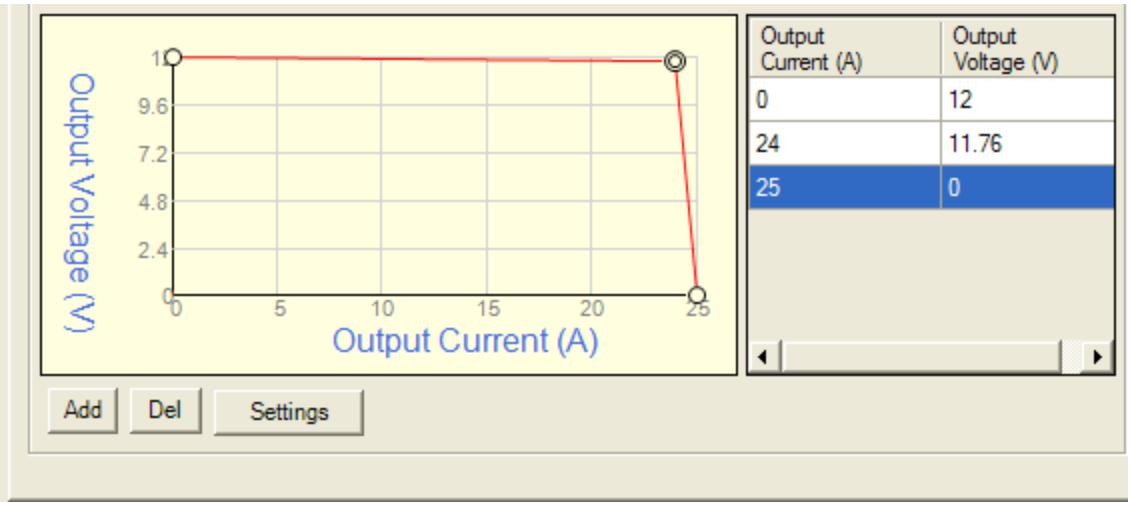


8. Similarly, you can define the behavior of the Output Voltage vs. Output Current graph. Click **Settings** on the Output Voltage vs. Output Current graph and enter a maximum

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

value of **25** for the **Output Current (A)**, and a maximum value of **12** for the **Output Voltage (V)** (the output voltage value corresponds to the nominal output voltage).

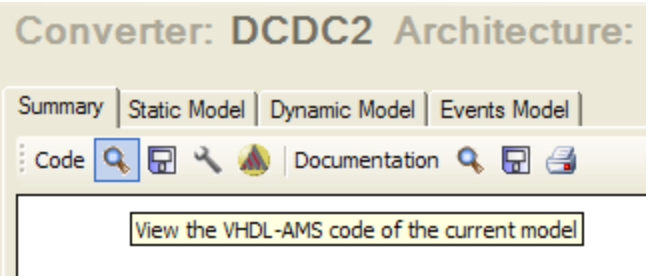
9. In this example a load regulation of 2% at 24A is being defined. This means that the output voltage of the converter drops 0.24V when the converter is supplying an output current of 24A. The values to enter are **(0,12)**, **(24,11.76)**, and **(25,0)**. Note that the converter also has output protection that makes the voltage drop when output current exceeds 24A.



At this point the minimum characteristic values needed to create a simulation model are defined, VHDL-AMS code for the model can be generated, and the model exported to Twin Builder.

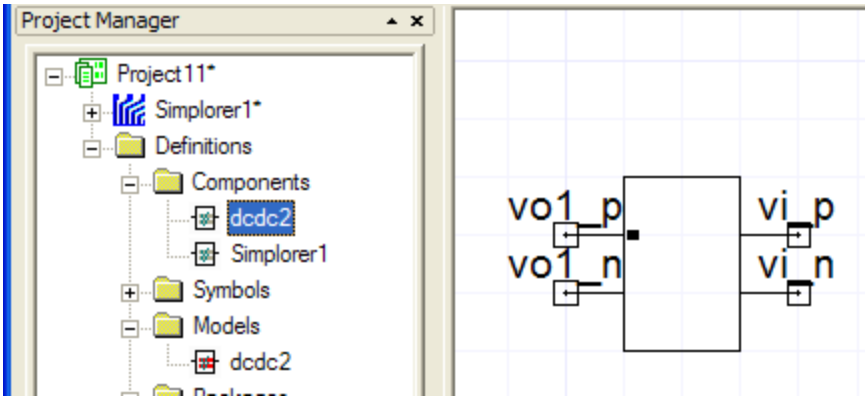
Generating VHDL-AMS Code and Importing the Model into Twin Builder

1. Click the **Summary** tab in the editing panel. The **Summary** tab contains options for the most common functions for code and documentation generation.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

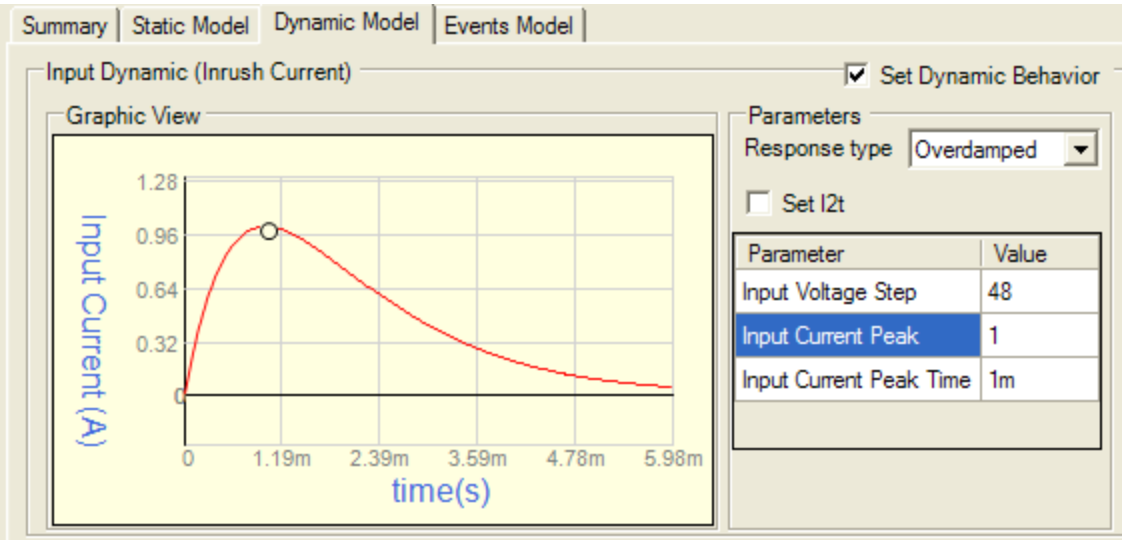
2. In the main menu, select **Behavioral DC/DC** > **Code** > **View** or click the **Code** icon. The VHDL-AMS code for the converter model appears (read-only) in the viewer.
3. To import this model into Twin Builder, select **Behavioral DC/DC** > **Code** > **Send To Simplorer**. The system adds the model to the currently active project in Twin Builder.
4. Return to Twin Builder. In the **Project Manager** pane, find the imported model under **Definitions** > **Components**, then drag and drop the model onto a schematic.



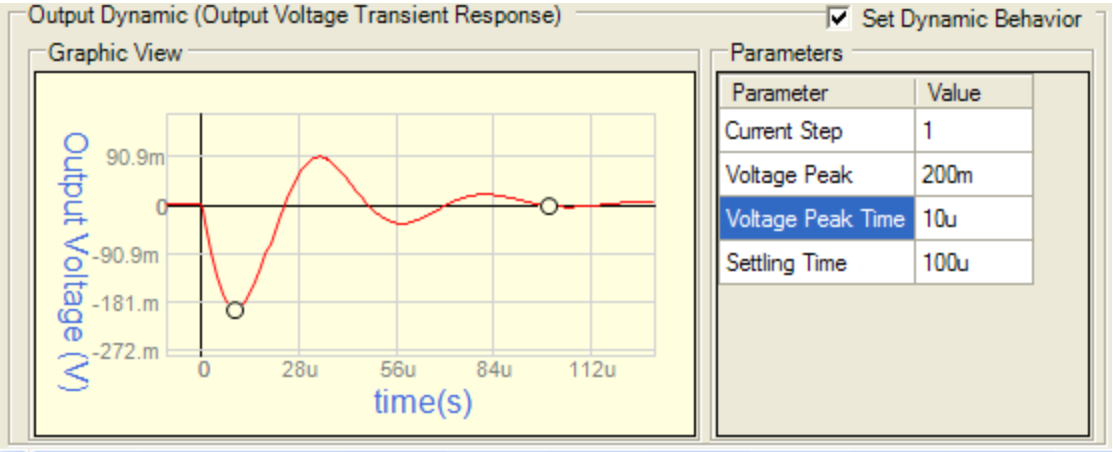
Defining the Converter Model Dynamic Behavior

1. Select the **Dynamic Model** tab in the editing panel. In the **Input Dynamic (Inrush Current)** panel, select the **Set Dynamic Behavior** check box.
2. Set the **Input Voltage Step** parameter to **48V**. This value corresponds to a step in the input voltage when the converter is powered. Set the **Input Current Peak** parameter to **1A** and the **Input Current Peak Time** to **1ms**. You can also drag the points to graphically define the behavior of the input current.
3. Right-click the input dynamic graph and select **Automatic Scale** to display the input current graph in its best representation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. In the **Output Dynamic (Output Voltage Transient Response)** panel, select the **Set Dynamic Behavior** check box to include the behavior of the converter under load steps.
5. Set the **Current Step** value to **1A**, this value corresponds to the load step under which the test was performed. Set the **Voltage Peak** to **200mV**. This value is the voltage drop peak when the load step occurs. Set the **Voltage Peak Time** to **10us**. This is the time when the voltage peak occurs. Finally, set the **Settling Time** value to **100us**. This value represents the time that the converter takes to recover from the load step.
6. Right-click the graph and select **Automatic Scale** to display the graph in its best representation.

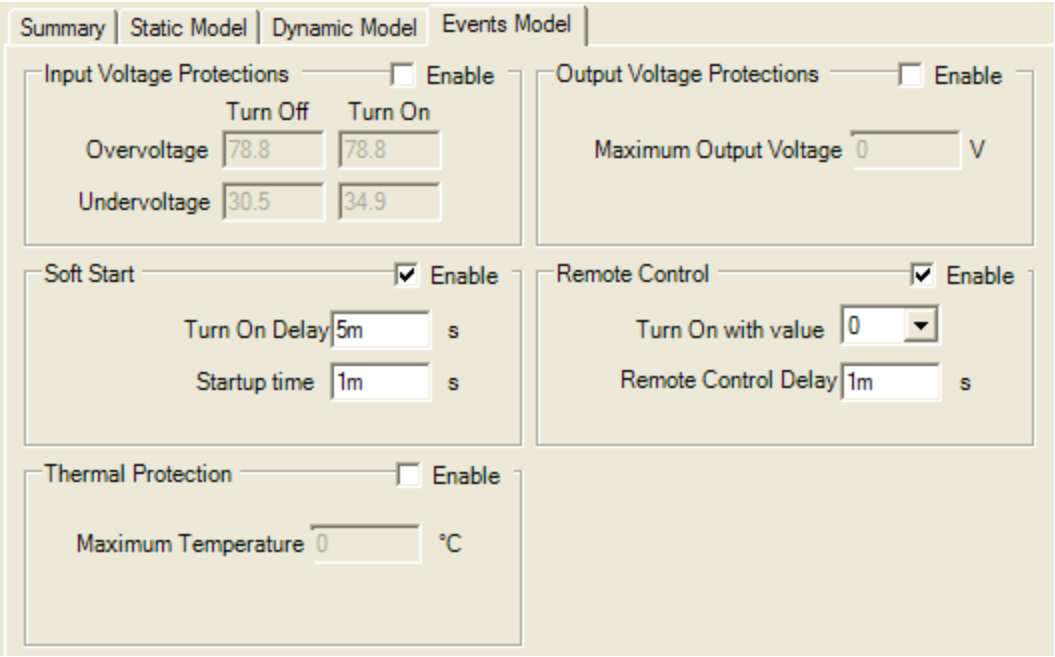


7. On the **Summary** tab, generate the model and import it to Twin Builder as needed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Defining Event Driven Behavior

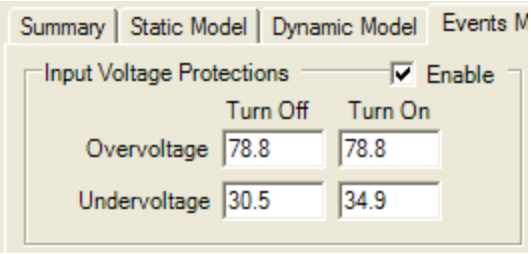
1. Click the **Events Model** tab in the editing panel. Select the **Soft Start** and **Remote Control** panel **Enable** check boxes to enable these features.



2. In the **Soft Start** panel, set the value of **Turn On Delay** to **5ms** and the **Startup time** to **1ms**. **Turn On Delay** represents the time after power-up before the converter begins supplying voltage. The **Startup time** is the time that takes the converter to raise the output voltage from zero to its nominal value.
3. When **Remote Control** is enabled, a terminal is added to the Twin Builder model to control the converter. The **Turn On with value** combo box defines the value used to turn on the converter. Set the **Remote Control Delay** to **1ms**. This value specifies the time that it takes for the converter to turn on or off after the remote control signal is toggled.
4. Select the **Enable** check box in the **Input Voltage Protections** panel. This function defines the input voltage operating range of the converter. Set **Undervoltage Turn On** to **34.9V** and **Undervoltage Turn Off** to **30.5V**. These values mean that when the converter is turned off it needs at least 34.9 V to turn on; and once it is turned on, if the voltage drops below 30.5V the converter turns off.
5. Set **Overvoltage Turn Off** and **Overvoltage Turn On** to **78.8V**. In this case, when the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

input voltage is above 78.8V the converter will turned off and if it is below it will turn on.

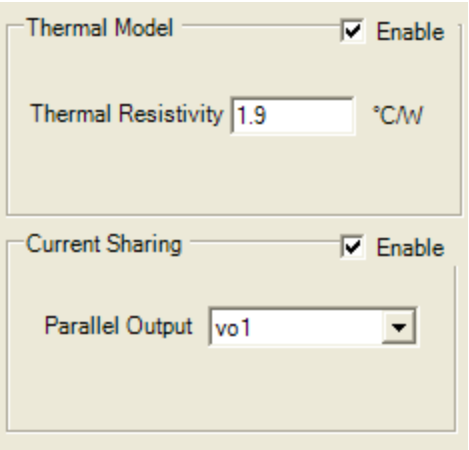


You can also enable other types of protection on the **Events Model** tab, such as **Output Voltage Protection** and **Thermal Protection**. In this example these protections are not used.

Enabling Current Sharing and Thermal Modeling

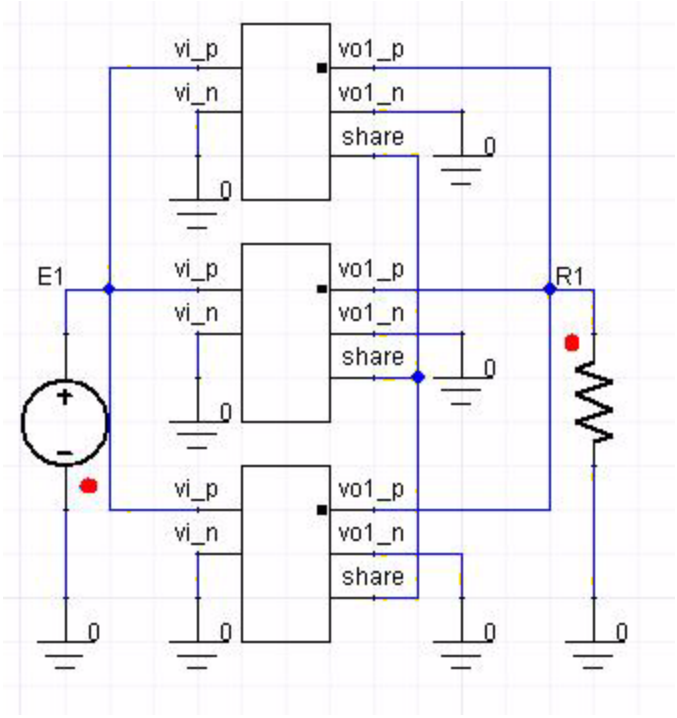
1. Click the **Static Model** tab in the editing panel and select the **Current Sharing** check box.

2. Select a **Parallel Output** from the drop-down list.
3. Generate the code and import it to Twin Builder as described in ***Generating VHDL-AMS Code and Importing the Model into Twin Builder***.



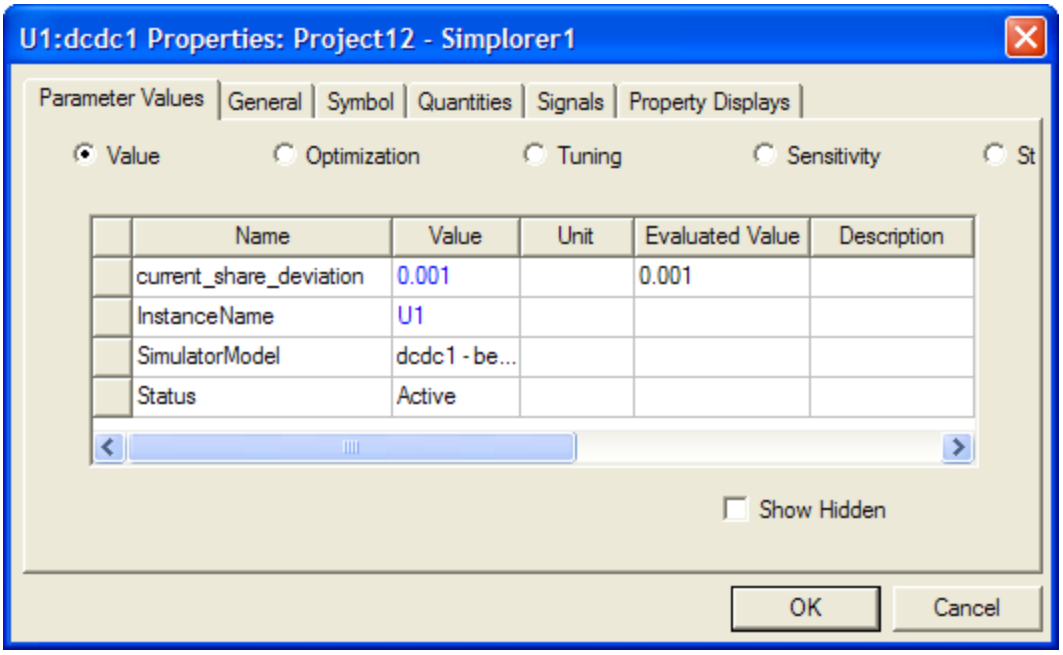
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Insert the model in a schematic in Twin Builder to create a simulation model.



5. In reality, two similar converters cannot be perfectly equal; they have physical differences. When you add the current sharing feature to a model, a corresponding *current_share_deviation* parameter is also included in the model. This value defines a factor of deviation that simulates a real and imperfect current sharing among converters. This parameter is set in Twin Builder as shown below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



6. Similarly, to add the effects of thermal resistivity to the model, select the **Thermal Model** check box and supply the desired **Thermal Resistivity** value.

Modeling of DC-DC Converters Based on Hybrid Wiener-Hammerstein Structure

Most modeling approaches of DC-DC converters are based on some kind of averaging technique that requires a knowledge of the topology, the control strategy, and the parameters of all the components. These kinds of techniques are very useful for design of the control loops but, because commercial DC-DC converters do not supply this information (it is part of their intellectual property), these modeling approaches cannot be applied to extract a model of a commercial DC-DC converter.

Other modeling approaches try to model the behavior of the converter based on the small-signal frequency identification of the converter. This approach is very accurate from the point of view of small-signal stability but it does not account for efficiency as a function of the input voltage and load, output voltage as a function of load or protections such as overvoltage, over-current, under-voltage and temperature. The model proposed below overcomes all of those limitations. The trade-off between simulation time and accuracy of DC-DC converters is successfully addressed by the use of a hybrid model as shown in the following figure:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

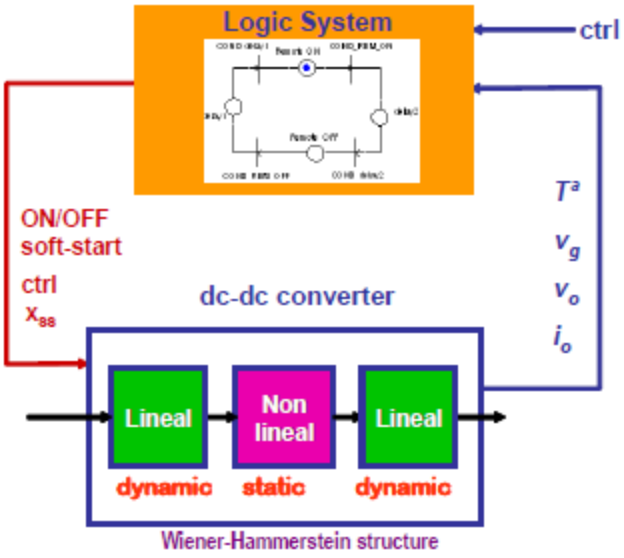


Figure 8-1 ***Proposed DC-DC Converter Model Structure***

Proposed Hybrid Wiener-Hammerstein Structure

Control signals, protection, and start-up behavior are extremely important in the assessment of the stability of distributed power systems. These parameters modify the behavior of the converter and must be considered in the model.

Taking into account the hybrid nature of DC-DC converters, the model is partitioned in two blocks:

The **logic system** (event driven behavior) manages the protection and remote control features. It is modeled by means of a state diagram. Its implementation on a circuit simulator depends on the modeling capabilities of the simulator. It measures the analog variables that can modify the behavior of the DC-DC converter (output voltage, output current, input voltage, temperature, and so on) and, based on its current state and the analog inputs, it generates the control signal for the power stage and control.

The **continuous/discrete system** includes the power stage and its control - independently if it is implemented digitally or analogy.

The **power stage and control** is modeled by means of a Wiener-Hammerstein structure (see below). This structure has the advantage of gathering all the non-linear static behavior of the converter, usually provided in datasheets, by means of one block. This non-linear behavior includes:

- Variation of the output voltage with respect to the output current and input voltage.
- Variation of the efficiency with the load and the input current.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Detailed Wiener-Hammerstein Structure

One of the most critical steps in the modeling process is to select the model structure. If the selected structure is wrong, it does not matter what optimization algorithm is used to fit the model – it will never find a correct solution.

In this case, the problem is how to find a model structure valid for all kinds of DC-DC converters (with and without zeros in the Right Hand Plane (RHP), hard switched PWM converters or frequency controlled resonant converters). The proposed model for the power stage is based on a Wiener-Hammerstein structure shown below.

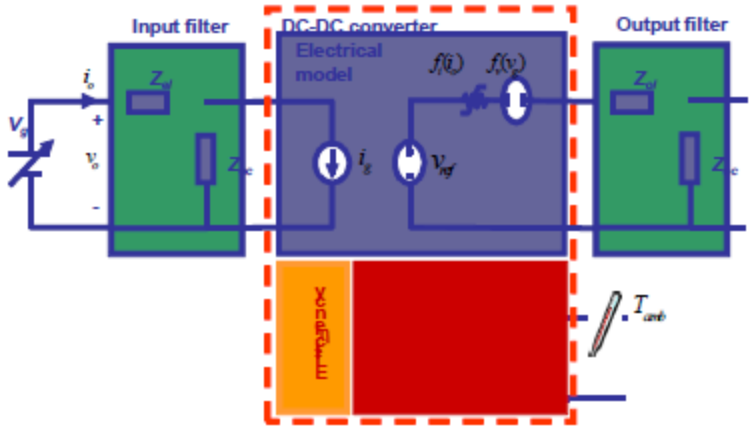


Figure 8-2 ***Proposed Wiener-Hammerstein structure for the dc-dc converter model***

It consists of:

- A non-linear static model that represents the steady-state behavior of the converter.
  a. Efficiency as a function of input voltage and output current.
  b. Output voltage as a function of the reference, the input voltage and the output current.
- A dynamic input block that models the high-frequency input impedance behavior of the converter and its initial inrush current.
- A dynamic output block that models the transient behavior of the converter under load changes.

Additionally, a simple thermal model accounts for the effect of thermal protection.

The equations for the static, nonlinear model are given by:

$$v_o = v_{ref} + f_v(v_g) - f_z(i_o) \quad (1)$$

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$i_{in} = \frac{1}{\eta(v_g, i_o)} \frac{v_o \cdot i_o}{v_g} \qquad (2)$$

The implementation of these equations in a circuit-oriented simulator based on information given by the manufacturers is straightforward. The dependence of the output voltage on the input voltage and output current can be implemented either by a 2-D lookup table, or by means of two 1-D lookup tables.[1] This solution, though an approximation, provides very good results in terms of accuracy and convergence.

Additionally, dependence of efficiency on the input voltage and output current also can be implemented by means of a 2-D lookup table or two 1-D lookup tables.

Linear dynamic output network

To propose a suitable network to consider the output dynamics of a generic converter the following assumptions are made:

1. **There is a capacitive element place at the output**. Actually, this is always the case to reduce output voltage ripple and to reduce output impedance. This capacitive element need not be ideal, and the model can include the equivalent series resistance (ESR) and inductance (ESL) of the capacitor bank.
2. **Output voltage of the converter is controlled through the injected current**. This is almost true for current mode controlled converters up to the crossover frequency of the current loop. In voltage mode controlled converters seems less obvious but, as will be shown the approach is also applicable - but with higher error.

Using these assumptions, the small-signal model of the output stage of a general current controlled converter can be represented as shown below, where:

- Z_C is the equivalent impedance of the output capacitors (including parasitics).
- K_V is the output voltage controller transfer function.
- $\tilde{i}_c$ is the injected current.
- $\tilde{v}_{ref}$ is the reference voltage.
- $\tilde{i}_0$ is the current demanded by the load.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

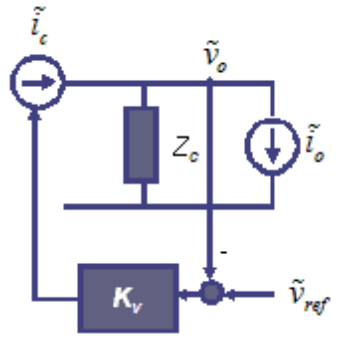


Figure 8-3 ***Equivalent Small-signal Circuit of the Output Stage of a Current-Controlled Converter***

According to the equivalent circuit shown above, and after some calculation, it is possible to arrive at the following expression for the output voltage (3):

$$i_{in} = \frac{1}{\eta(v_g, i_o)} \frac{v_o \cdot i_o}{v_g} \qquad (3)$$

This transfer function is also the same as the transfer function of the equivalent circuit of the following figure, just by making the series impedance Z_L equal to the inverse of the voltage loop regulator (4):

$$Z_L(s) = \frac{1}{K_v(s)} \qquad (4)$$

As a consequence, the proposed dynamic network to fit the output dynamics of the converter is the one shown below.

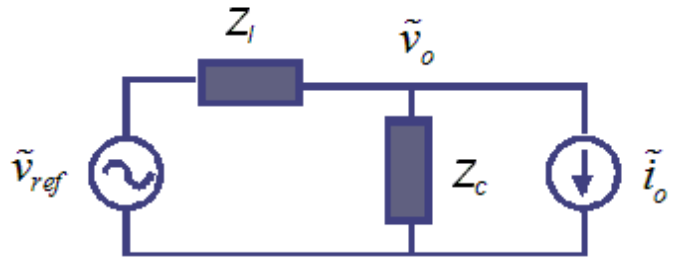


Figure 8-4 ***Equivalent small signal circuit of the output stage***

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Because the output network does not change the static behavior of the converter, which is accounted for by the non-linear network, the condition that must be satisfied by this network is:

$$\frac{Z_L(0)}{Z_C(0)} = 0 \qquad (5)$$

The impedance of Z_L/Z_C must be 0. It can be achieved with $Z_C(0)$=infinite (capacitor) and/or $Z_L(0) = 0$ (inductance).

From the point of view of the parametric identification it is necessary to determine the structure of the series impedance, Z_L, to identify its components. To do that, the form of the transfer function K_V will be analyzed.

Using the above assumptions and the small signal model shown above, the output voltage can be controlled by means of a PI controller:

$$K_v(s) = \frac{\omega_i}{s}\cdot\left(1+\frac{s}{\omega_z}\right) \qquad (6)$$

where:

- ω_i adjusts the crossover frequency and
- ω_z sets the phase margin.

This controller, according to equation (4) yields the following equivalent series impedance:

$$Z_i(s) = \frac{1}{K_v(s)} = \frac{\frac{1}{\omega_i}s}{1+\frac{s}{\omega_z}} \qquad (7)$$

The network that fits into this transfer function consists of an inductance in parallel with a resistor of the following values:

$$L = \frac{1}{\omega_i}$$

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$L = \frac{1}{\omega_i} \qquad (8)$$

$$R_d = \frac{\omega_z}{\omega_i} \qquad (9)$$

Applying this procedure to the current-controlled converter with equivalent output capacitors impedance (C, R_{ESR}), the linear output network will have the structure shown below. This output network can be easily identified based on the current step-response data provided by the manufacturer (or an equivalent measurement).

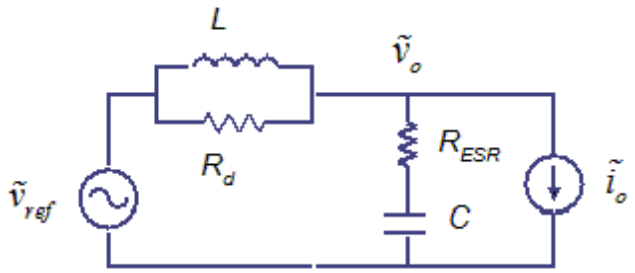
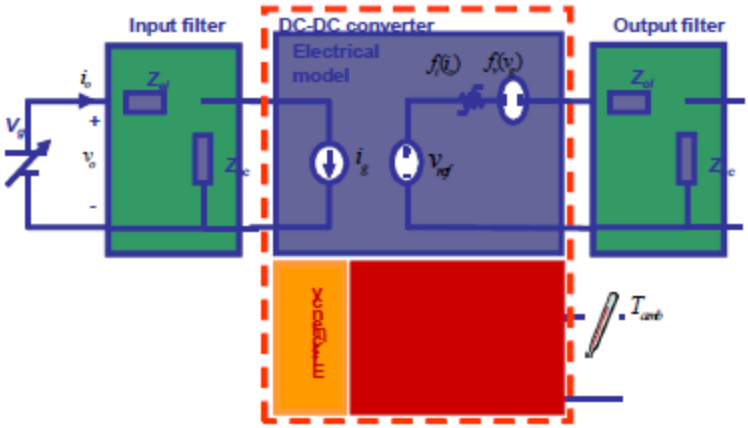


Figure 8-5 ***Equivalent Closed Loop Circuit***

Linear dynamic input network

The proposed structure of the input network is labeled **Input filter** below. The selection of this network is based on the typical configuration of the EMI input filters for DC-DC converters.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

This network must be selected under the constraint of not modifying the static behavior of the converter, given by the nonlinear static block. As a consequence, it must satisfy the same condition in steady state as the output network equation (5).

One possible network that can be fitted based on inrush current data is shown below [1-2]. This network is valid for both over-damped and under-damped inrush current behavior.

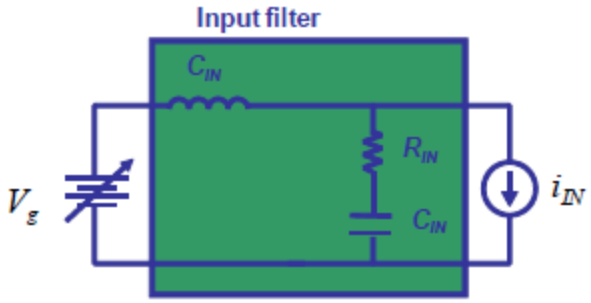


Figure 8-6 ***Candidate input filter network***

Event-driven Behavior

The behavior of the converter depends on its state (ON, OFF, PROTECTION, and so on). It is possible to include this dependency on the behavior thanks to the mixed-signal capabilities of current circuit simulators. In this example, the state of the converter is modeled by means of the state diagram shown below that controls the values of the dependent voltage and current sources.

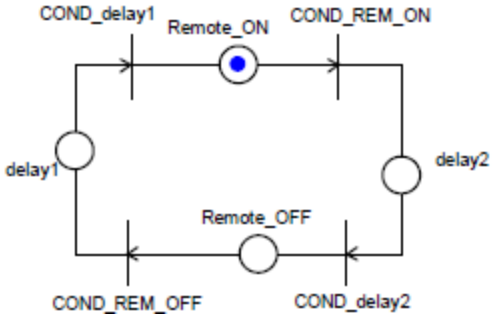


Figure 8-7 ***Remote ON-OFF state diagram***

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

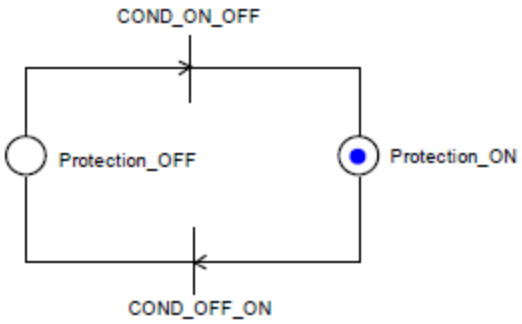


Figure 8-8 ***Protection state diagram - State Diagram for Event Driven Behavior Description***

In this example, most common events are included based on a careful search of many DC-DC converter manufacturers. The events included in this model are:

- **Remote control** - This signal is usually provided to DC-DC converters to enable or disable them.
- **Input voltage protection** - This protection disables the operation of the converter when the input voltage is out of the operational limits of the converter, to avoid malfunction.
  - **Under-voltage protection** (with and without hysteresis) - Below this voltage, the converter will be turned off to avoid high currents that can destroy it.
  - **Over-voltage protection** (with and without hysteresis) - This voltage setting assures that all the components of the converter are working with designed voltage ranges.
- **Output voltage protection** - This is usually an upper output voltage value that disables the converter to protect it against overvoltage caused by an external source.
- **Thermal protection** - In the case of high-temperature operation of the converter, this protection turns it off if the high-temperature limit is exceeded.
- **Over-current protection** - This protection tries to avoid high internal current due to high external loads. This protection is included inside the event-driven behavior in case of turning-off the converter. The protection based on lowering the output voltage as a function of the output voltage must be included in the static non linear model since it does not create an event.

References

[1] J. A. Oliver, R. Prieto, V. Romero, and J. A. Cobos, "*Behavioral modeling of dc-dc converters for large-signal simulation of distributed power systems,"* IEEE Applied Power Electronics Conference and Exposition, APEC 2006.

[2] J. A. Oliver, R. Prieto, V. Romero, and J. A. Cobos, "*Behavioral Modeling of Multi-Output DC-DC Converters for Large-Signal Simulation of Distributed Power Systems,"* Power Electronics Specialists Conference, 2006.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

9 - Component Dialog Wizard

The Twin Builder **Component Dialog Wizard** facilitates the development of component dialog boxes for user-defined models, as well as many other models (for example, some VHDL-AMS components and subcircuit components such as Icepak) that do not already have their own component dialog boxes. The wizard features an easy-to-use graphical interface you can use to design, preview, and test component dialog boxes.

This section describes how to create, edit, and test component dialog boxes with the **Component Dialog Wizard**. It also provides a thorough explanation of the menus and toolbars in the wizard, and a detailed step-by-step example of how to create a component dialog box using the wizard.

Note: In Twin Builder, model text compilation typically creates a component dialog box with a default layout. This layout is overwritten whenever you update any new model or component compilation. After you use the **Edit Component** dialog box to edit and save the wizard layout, it will no longer generate a new default layout for that component.

Related Topics

Creating a New Component Dialog Box Using the Component Dialog Wizard

Adding Static Information and Control Elements to a Component Dialog Box

Determining a Non-conservative Input Node's Value Using Free Values

Aligning and Sizing Component Dialog Box Elements

Setting the Tab Order of a Component Dialog Box

Setting the Display Behavior of Component Dialog Box Elements

Testing a Component Dialog Box

Saving and Closing a Component Dialog Box

Using a Component Dialog Box

Revising a Component Dialog Box

Component Dialog Wizard Window Items

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

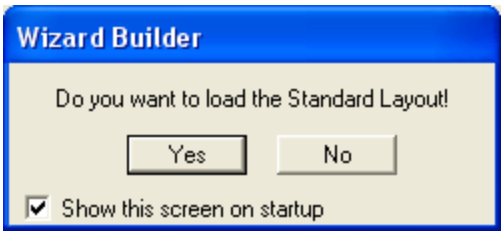
Creating a New Component Dialog Box Using the Component Dialog Wizard

To create a new component dialog box, it is first necessary to start the **Component Dialog Wizard**.

1. Open the **Edit Component** dialog box. Right-click a component on a schematic and select **Edit Component**.

   You can also right-click a component listed on the Component Libraries window **Components** tab and select **Edit Component**, or you can right-click a component definition in the **Project ManagerProject** tree and select **Edit Component**.

2. In the **Edit Component** dialog box, click **General** > **Setup Dialog**. The **Component Dialog Wizard** appears. The **Wizard Builder** dialog box may also appear.



   - If you select **Yes** in this dialog box, the **Component Dialog Wizard** loads the standard layout, which is a partially completed component dialog box. The standard layout is useful for many component dialog boxes because it creates the basic structure of the dialog box, decreasing your work and saving you time.
   - If you select **No** in the **Wizard Builder** dialog box, the **Component Dialog Wizard** loads a blank component dialog box.

Hint	If the **Wizard Builder** dialog box does not appear upon starting the **Component Dialog Wizard**, the wizard loads a blank component dialog box. You can still add the standard layout to the blank dialog box; select **Layout** > **Create Standard Layout** on the wizard's menu bar.

Adding Static and Control Elements to a Component Dialog Box

Component dialog boxes created with the **Component Dialog Wizard** can include static information such as node and output names, as well as control elements used to change the input values and appearance of a model. The topics in this section describe the different methods available for adding static information and control elements to a component dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Start the **Component Dialog Wizard** before adding information or control elements to a component dialog box. See ***Creating a New Component Dialog Box Using the Component Dialog Wizard*** for more information.

Hint	When adding static information or a control element to a component dialog box, you may need to resize the component dialog box to accommodate the new element.

Related Topics

Changing the Name of a Component Dialog Box

Adding Groups to a Component Dialog Box

Adding Statics to a Component Dialog Box

Adding Text Edits to a Component Dialog Box

Adding ComboBoxes to a Component Dialog Box

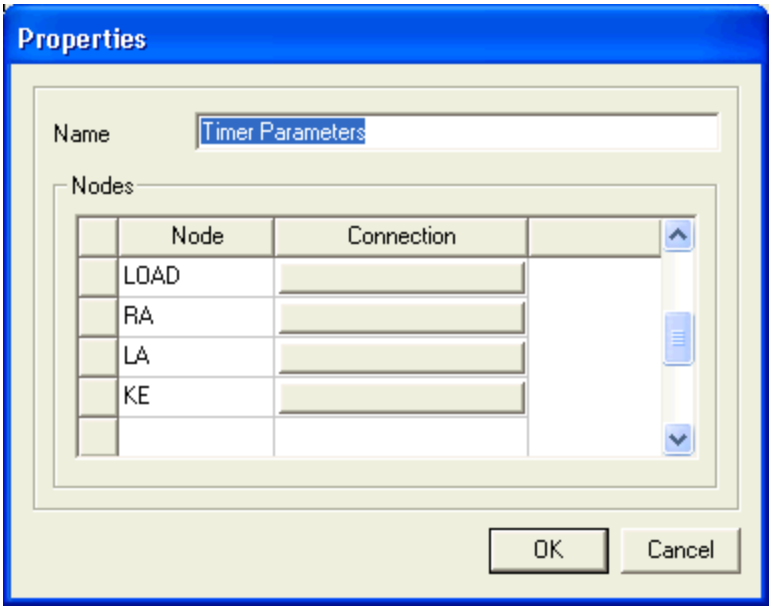
Adding Check Boxes to a Component Dialog Box

Adding Radio Buttons to a Component Dialog Box

Changing the Name of a Component Dialog Box

To change the name of a component dialog box:

1. Right-click an empty portion of the component dialog box and select **Properties**. The **Properties** dialog box opens.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Type the name of the component dialog box in the **Name** field.
3. Click **OK**.

   The new name becomes visible in the title bar of the component dialog box.

Adding Groups to a Component Dialog Box

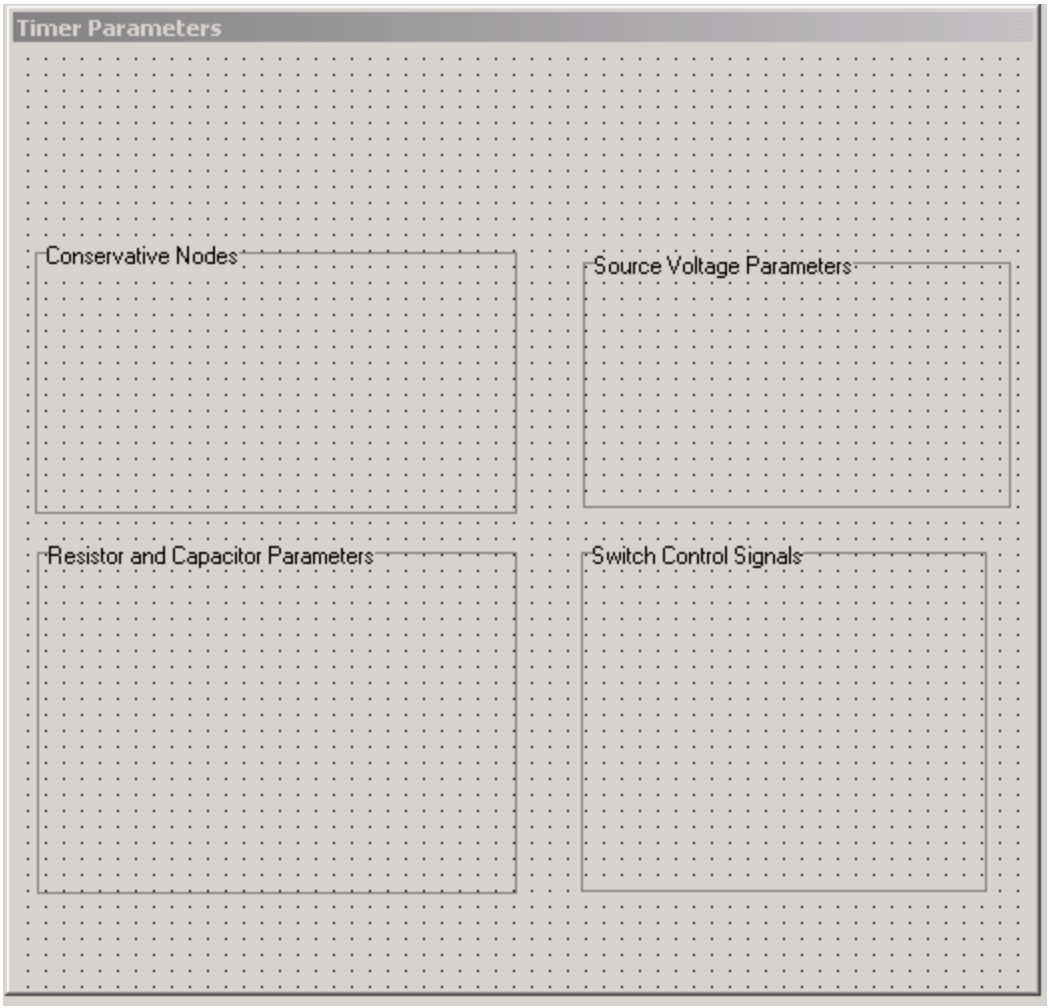
Groups are static information elements that provide visual organization in a component dialog box. Because groups are typically the largest elements added to a component dialog box, you should add them first.

Follow this procedure to add a group to a component dialog box:

1. Select **Controls** > **Group** from the **Component Dialog Wizard** menu bar.
   - You can also click **Add Group** on the **Standard** toolbar.
2. Click the component dialog box to place the new group.
3. Drag the group to its desired location in the component dialog box.
4. Right-click a group border. The **Properties** dialog box opens.
5. Edit the group name in the **Name** field on the **Group Data** tab.
6. Click **OK**.

   Several completed groups are shown in the following illustration:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Adding Statics to a Component Dialog Box

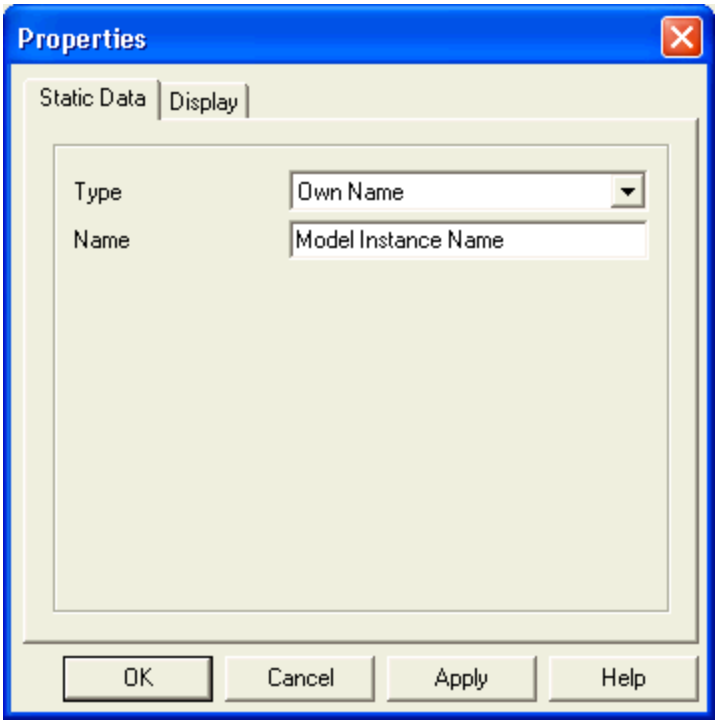
Statics are static information elements that display text. They can display simple labels that accompany control elements such as text edits and **Combo Boxes**, or node information. The following is an example of a static as it appears in the **Component Dialog Wizard**:

To create a static for displaying a simple label:

1. Select **Controls** > **Static** from the **Component Dialog Wizard** menu bar.
2. Click the component dialog box to place the static.
3. Drag the static to its desired location in the component dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Right-click the static. The **Properties** dialog box opens.

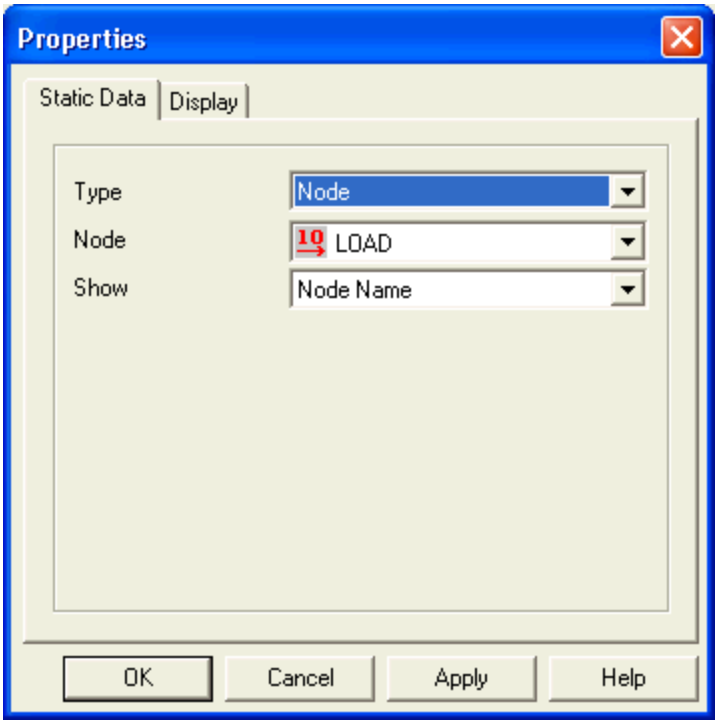


5. Select **Own Name** from the **Type** list on the **Static Data** tab.
6. Type the text to be displayed by the static in the **Name** field.
7. Click **OK**.

To create a static for displaying node information:

1. Select **Controls** > **Static** from the **Component Dialog Wizard** menu bar.
2. Click in the component dialog box to place the static.
3. Drag the static to its desired location in the component dialog box.
4. Right-click the static. The **Properties** dialog box opens.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



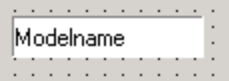
5. Select **Type** > **Node** on the **Static Data** tab.
6. Select the name of the node for which information is to be displayed from the **Node** list.
7. Select the type of node information to be displayed from the **Show** list.
   - **Description** - The node's description is shown on the component dialog box as [Description].
   - **Description(Node Name+Unit)** - The node's name and unit are shown on the component dialog box as [Node Name, Unit].
   - **Node Name** - The node's name is shown on the component dialog box as Node Name.
   - **Node Name+Unit** - The node's name and unit are shown on the component dialog box as Node Name [Unit].
   - **Node Name+Unit+Description** - The node's name, unit, and description are shown on the component dialog box as [Node Name, Unit, Description].
8. Click **OK**.

Adding Text Edits to a Component Dialog Box

Text edits are control elements used for entering information to a model via its component dialog box. Use text edits to enter the name of an instance of a model, to enter values into nonconservative input nodes, and to enter free values. For more information on free values, see ***Determining a Nonconservative Input Node's Value Using Free Values***.

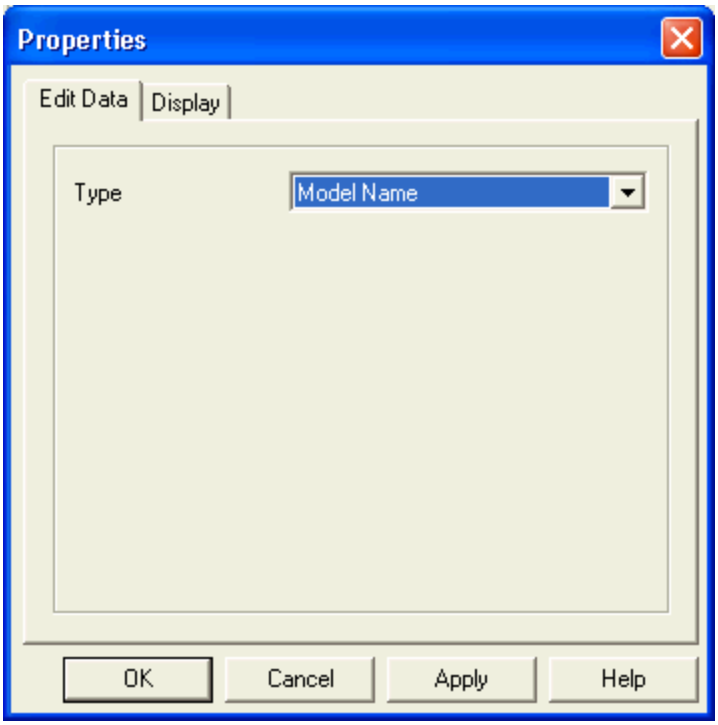
The following is an example of a text edit as it appears in the **Component Dialog Wizard**:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



To create a text edit for entering a model instance name:

1. Select **Controls** > **Edit** from the **Component Dialog Wizard** menu bar.
2. Click the component dialog box to place the text edit.
3. Drag the text edit to its desired location in the component dialog box.
4. Resize the text edit to its desired height and width.
5. Right-click the text edit. The **Properties** dialog box opens.



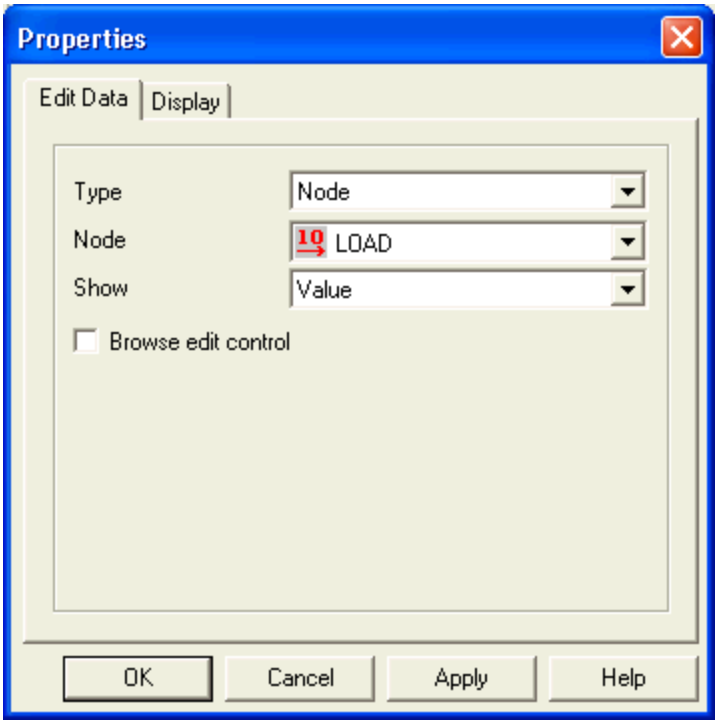
6. Select **Type** > **Model Name** on the **Edit Data** tab.
7. Click **OK**.

To create a text edit for entering a value into a nonconservative input:

1. Select **Controls** > **Edit** from the **Component Dialog Wizard** menu bar.
2. Click in the component dialog box to place the text edit.
3. Drag the text edit to its desired location in the component dialog box.
4. Resize the text edit to its desired height and width.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Right-click the text edit and select **Properties**.The **Properties** dialog box opens.



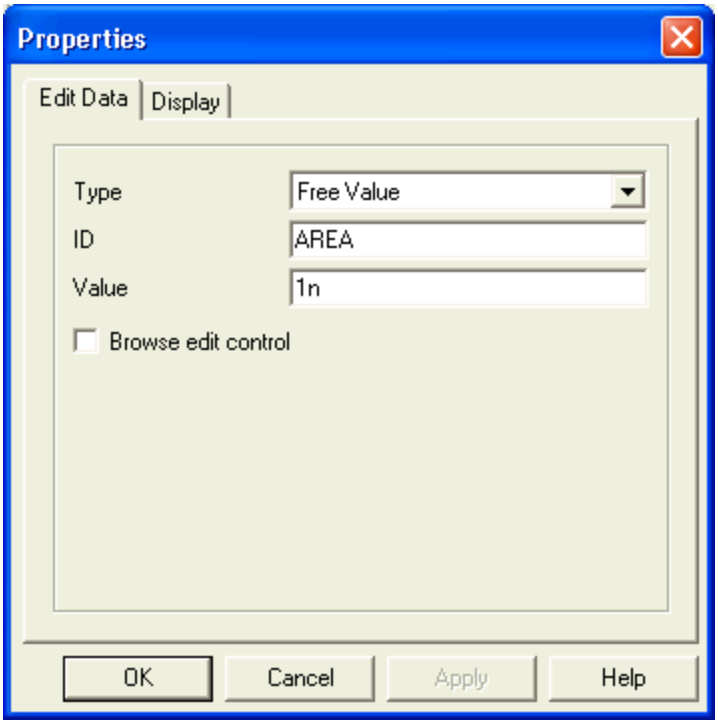
6. Select **Type** > **Node** on the **Edit Data** tab.
7. Select the name of the desired node from the **Node** list.
8. Select **Show** > **Value** to enter a numerical value through the component dialog box, or select **Information** to enter other information.
9. Select the **Browse edit control** check box to enter a file name.
10. Click **OK**.

To create a text edit for entering a free value:

1. Select **Controls** > **Edit** from the **Component Dialog Wizard** menu bar.
2. Click in the component dialog box to place the text edit.
3. Drag the text edit to its desired location in the component dialog box.
4. Resize the text edit to its desired height and width.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Right-click the text edit and select **Properties**. The **Properties** dialog box opens.



6. Select **Type** > **Free Value** from the **Edit Data** tab.
7. In the **ID** field, type a valid node name for the free value.

> **Note:**
>
> Valid node names must start with a letter or underscore, and may contain any combination of uppercase and lowercase letters (A-Z, a-z), the numerals 0 through 9, and underscores.

8. Type a default value for the free value in the **Value** field.
9. Select the **Browse edit control** check box to enter a file name.
10. Click **OK**.

Adding Combo Boxes to a Component Dialog Box

A combo box is a control element that allows a choice of entries from a predefined list. In the **Component Dialog Wizard**, combo boxes are used for the following model information:

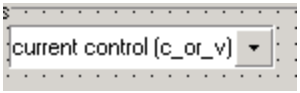
- values of non-conservative input nodes
- units of non-conservative input and output nodes
- labeling of conservative and non-conservative pins

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- free values

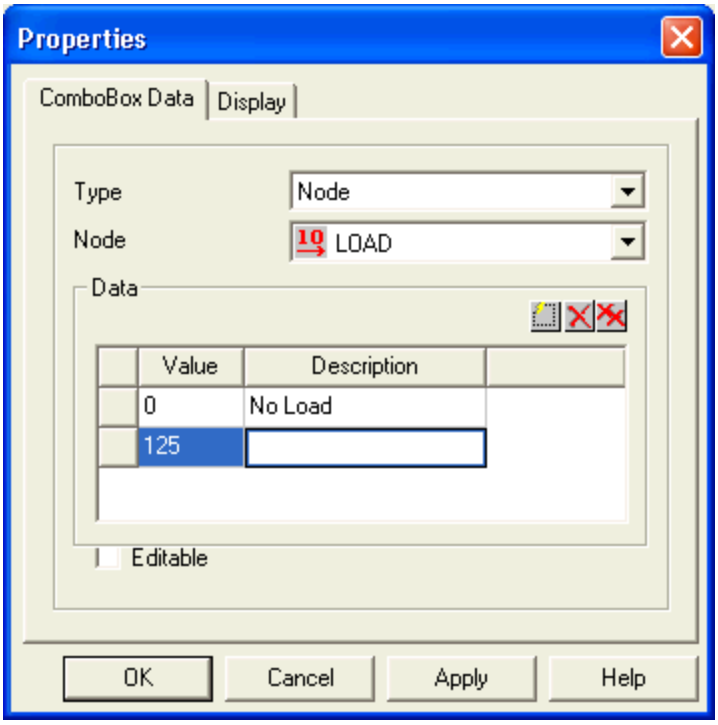
> **Note:**
>
> For more information on free values see ***Determining a Non-conservative Input Node's Value Using Free Values***.

The following is an example of a combo box as it appears in the **Component Dialog Wizard**:



To create a combo box for selecting the value of a non-conservative input node:

1. Select **Controls** > **ComboBox** from the **Component Dialog Wizard** menu bar.
2. Click the component dialog to place the combo box.
3. Drag the combo box to its desired location on the component dialog box.
4. Resize the combo box to its desired height and width.
5. Right-click the combo box and select **Properties**. The **Properties** dialog box opens.

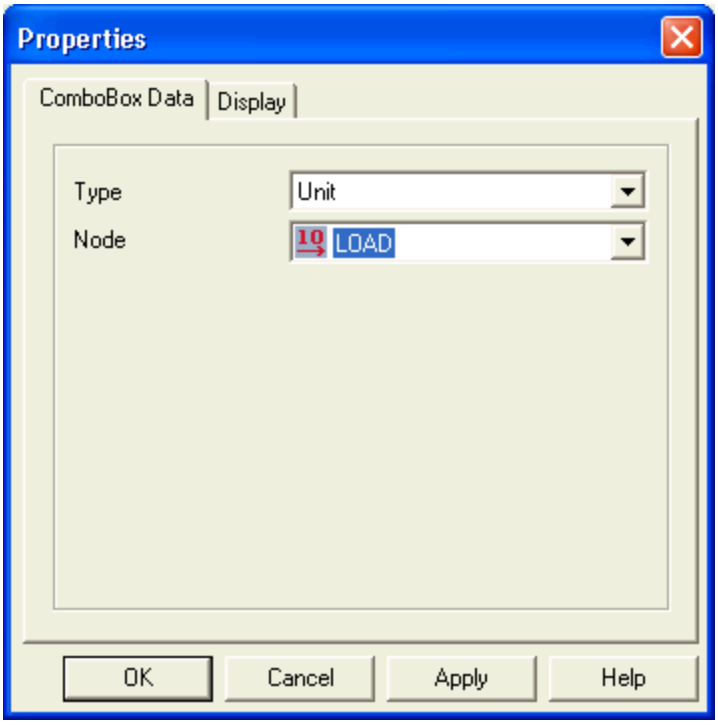


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. Select **Type** > **Node** on the **ComboBox Data** tab.
7. Select the name of the desired non-conservative input node from the **Node** list.
8. Click to create a predefined entry for the combo box.
9. Enter the description that will be listed in the component dialog box into the **Description** field.
10. Enter a value in the **Value** field that will be passed to the non-conservative input node when the corresponding description is selected in the component dialog box.
11. Repeat steps **9** and **10** for the remaining predefined entries of the combo box.
12. Select the **Editable** check box to make the contents of the combo box editable.
13. Click **OK**.

To create a combo box for selecting the unit of a non-conservative node:

1. Select **Controls** > **ComboBox** from the **Component Dialog Wizard** menu bar.
2. Click the component dialog box to place the combo box.
3. Drag the combo box to its desired location on the component dialog box.
4. Resize the combo box to its desired height and width.
5. Right-click the combo box and select **Properties**. The **Properties** dialog box opens.



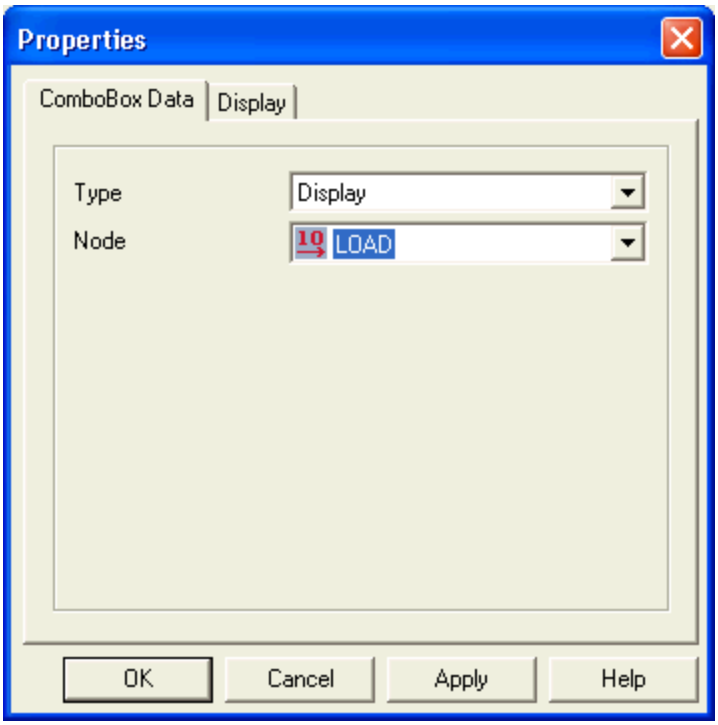
6. Select **Type** > **Unit** on the **ComboBox Data** tab.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7. Select the name of the non-conservative input node from the **Node** list.
8. Click **OK**.

To create a combo box for selecting the labeling of a conservative or non-conservative pin:

1. Select the **Controls** > **ComboBox** from the **Component Dialog Wizard** menu bar.
2. Click in the component dialog box to place the combo box.
3. Drag the combo box to its desired location in the component dialog box.
4. Resize the combo box to its desired height and width.
5. Right-click the combo box and select **Properties**. The **Properties** dialog box opens.



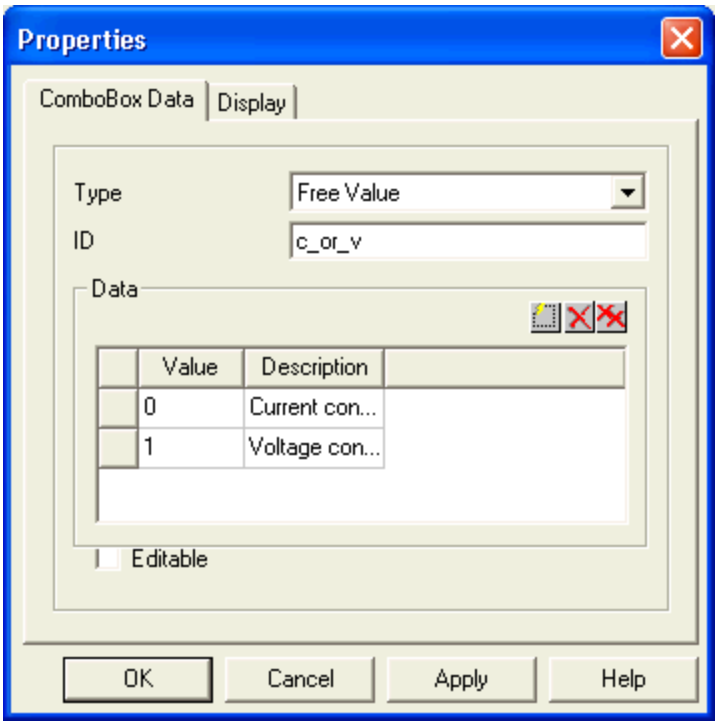
6. Select **Type** > **Display** on the **ComboBox Data** tab.
7. Select the name of the desired non-conservative or conservative node from the **Node** list.
8. Click **OK**.

To create a combo box for entering a free value:

1. Select **Controls** > **ComboBox** from the **Component Dialog Wizard** menu bar.
   - You can also click **Add Combobox** on the **Standard** toolbar.
2. Click in the component dialog box to place the combo box.
3. Drag the combo box to its desired location in the component dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Resize the combo box to its desired height and width.
5. Right-click the combo box and select **Properties**. The **Properties** dialog box opens.



6. Select **Type** > **Free Value** on the **ComboBox Data** tab.
7. Type a valid node name for the free value in the **ID** field.

> **Note:**
>
> Valid node names must start with a letter or underscore, and may contain any combination of uppercase and lowercase letters (A-Z, a-z), the numerals 0 through 9, and underscores.

8. Click to create a predefined entry for the combo box.
9. Enter a description that will be listed in the component dialog box in the **Description** field.
10. In the **Value** field, enter a value that will be passed to the free value when the corresponding description is selected in the component dialog box.
11. Repeat steps **10** and **11** for the remaining predefined combo box entries.
12. Select the **Editable** check box to edit the contents of the combo box.
13. Click **OK**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding Check Boxes to a Component Dialog Box

Check boxes are control elements that provide a simple either-or input. Using the **Component Dialog Wizard**, you can add check boxes to provide the following choices regarding model information:

- Toggle display of a model instance's name.
- Toggle display of a non-conservative pin. (Conservative pins are always shown.)
- Select or deselect model outputs.
- Assign a value to a non-conservative input node.
- Define a free value.

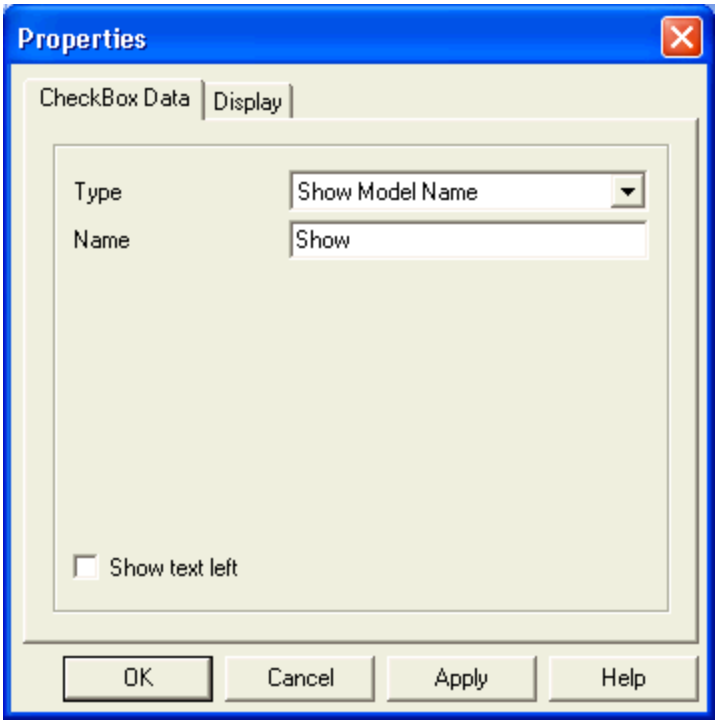
An example of a check box as it is appears in the **Component Dialog Wizard**:

To create a check box for displaying or not displaying a model instance name:

1. Select the **Controls** > **CheckBox** from the **Component Dialog Wizard** menu bar.
   - You can also click **Add Checkbox** on the **Standard** toolbar.
2. Click the component dialog box to place the check box.
3. Drag the check box to its desired location in the component dialog box.
4. Resize the check box to its desired height and width.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Right-click the check box and select **Properties**. The **Properties** dialog box opens.

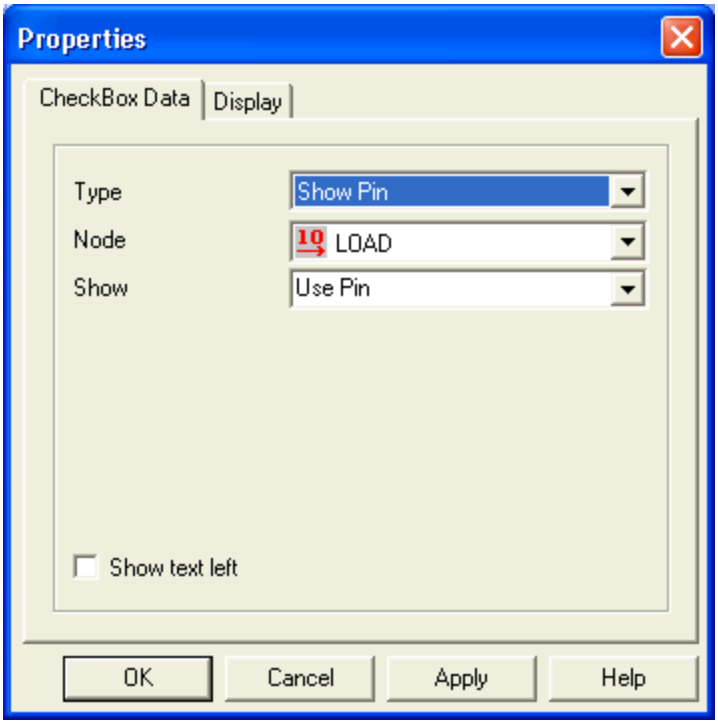


6. Select **Type** > **Show Model Name** on the **CheckBox Data** tab
7. Type a name for the check box in the **Name** field.
8. Select the **Show text left** check box to display the name to the left of the check box. If this option is cleared, the name displays to the right of the check box.
9. Click **OK**.

To create a check box for displaying or not displaying a pin:

1. Select **Controls** > **CheckBox** from the **Component Dialog Wizard** menu bar.
   - You can also click **Add CheckBox** on the **Standard** toolbar.
2. Click in the component dialog box to place the check box.
3. Drag the check box to its desired location in the component dialog box.
4. Resize the check box to its desired height and width.
5. Right-click the check box and select **Properties**. The **Properties** dialog box opens.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



6. Select **Type** > **Show Pin** on the **CheckBox Data** tab
7. Select the name of the desired node from the **Node** list.

> **Note:**
>
> **Show Pin** applies only to non-conservative nodes. Pins for conservative nodes are always visible.

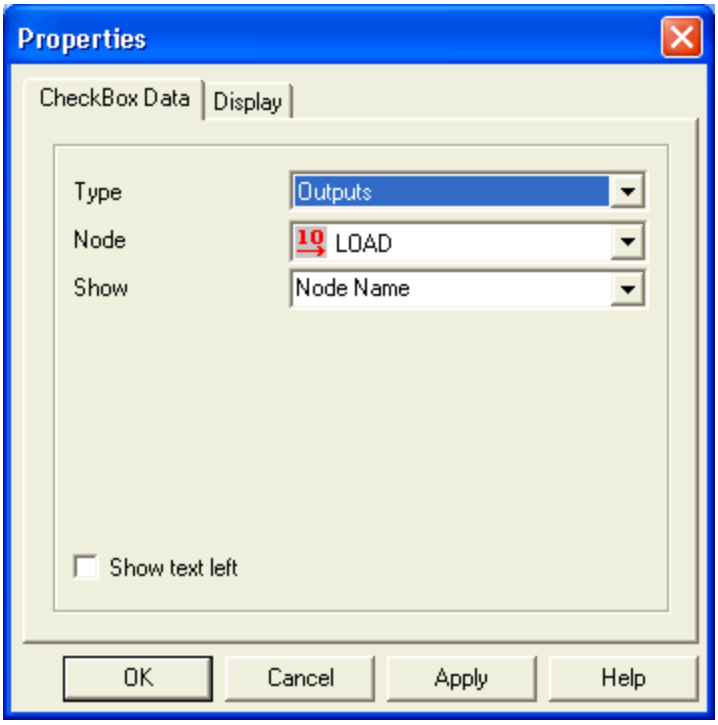
8. Select the manner in which the text accompanying the check box displays in the **Show** list. There are two choices:
   - **Use Pin[Node]** - A check box appears with the phrase “Use Pin” and the name of the node in brackets [Node Name].
   - **Use Pin** - A check box appears with the phrase “Use Pin” only.
9. Select **Show text left** to display the text to the left of the check box. If cleared, the text displays to the right of check box.
10. Click **OK**.

To create a check box for selecting or deselecting a non-conservative node as a model output:

1. Select the **Controls** > **CheckBox** from the **Component Dialog Wizard** menu bar.
   - You can also click **Add Checkbox** on the **Standard** toolbar.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Click in the component dialog box to place the check box.
3. Drag the check box to its desired location in the component dialog box.
4. Resize the check box to its desired height and width.
5. Right-click the check box and select **Properties**. The **Properties** dialog box opens.

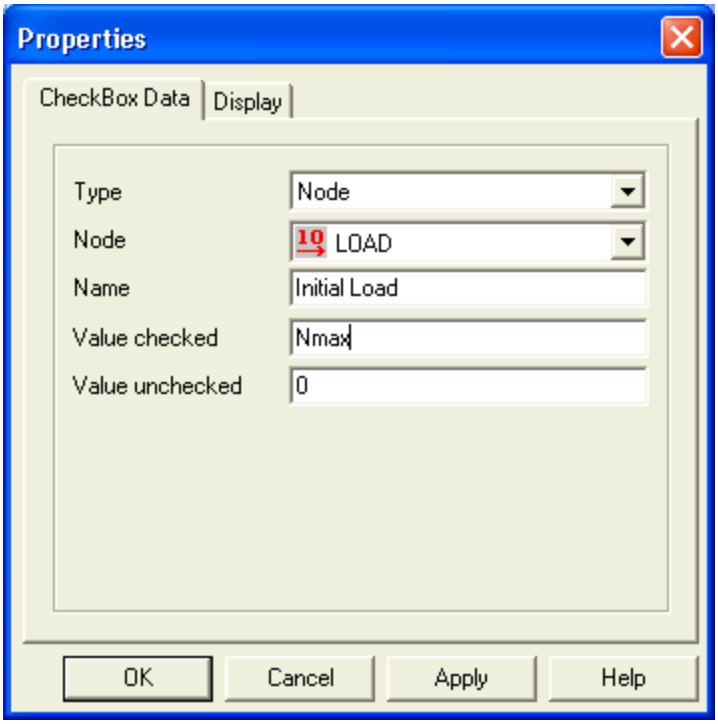


6. Select **Type** > **Outputs** on the **CheckBox Data** tab.
7. Select the name of the desired node from the **Node** list.
8. Select the manner in which the text accompanying the check box is displayed from the **Show** list. There are five choices:
   - **Nodename** - The check box appears with the node name only: *Node Name*.
   - **Node Name+Unit** - The check box appears with node name and expected unit in brackets: *Node Name* **[***Unit***]**.
   - **Output[Node]** - The check box appears with the word “Output” and the node name in brackets: **Output [***Node Name***]**.
   - **Description** - The check box appears with the node description only: *Description*.
   - **Description+Unit** - The check box appears with the node description and expected unit in brackets: *Description* **[***Unit***]**.
9. Select **Show text left** to display the text to the left of the check box. If cleared, the text displays to the right of check box.
10. Click **OK**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To create a check box for assigning one value or another to a non-conservative input node:

1. Select **Controls** > **CheckBox** from the **Component Dialog Wizard** menu bar.
   - You can also click **Add Checkbox** on the **Standard** toolbar.
2. Click in the component dialog box to place the check box.
3. Drag the check box to its desired location in the component dialog box.
4. Resize the check box to its desired height and width.
5. Right-click the check box and select **Properties**. The **Properties** dialog box opens.



6. Select **Type** > **Node** on the **CheckBox Data** tab.
7. Select the desired node from the **Node** list.
8. Type a name for the check box in the **Name** field.
9. Type a value to be assigned to the non-conservative node when the box is checked into the **Value checked**field.
10. Type a value to be assigned to the non-conservative node when the box is not checked into the **Value unchecked** field.
11. Click **OK**.

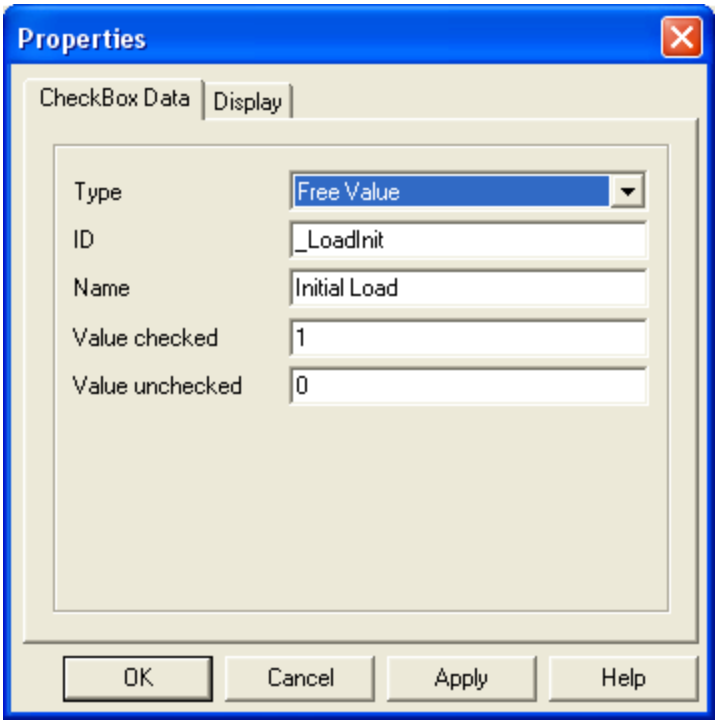
To create a check box for assigning one value or another to a free value:

1. Select **Controls** > **CheckBox** from the **Component Dialog Wizard** menu bar.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- You can also click **Add Checkbox** on the **Standard** toolbar.

2. Click in the component dialog box to place the check box.
3. Drag the check box to its desired location in the component dialog box.
4. Resize the check box to its desired height and width.
5. Right-click the check box and select **Properties**. The **Properties** dialog box opens.



6. Select **Type** > **Free Value** on the **CheckBox Data** tab.
7. Type a valid node name for the free value in the **ID** field.

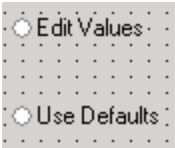
> **Note:**
>
> Valid node names must start with a letter or underscore, and may contain any combination of uppercase and lowercase letters (A-Z, a-z), the numerals 0 through 9, and underscores.

8. Type a name for the check box in the **Name** field.
9. Type a value to be assigned to the free value when the box is checked into the **Value checked** field.
10. Type a value to be assigned to the free value when the box is not checked into the **Value unchecked** field.
11. Click **OK**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding Radio Buttons to a Component Dialog Box

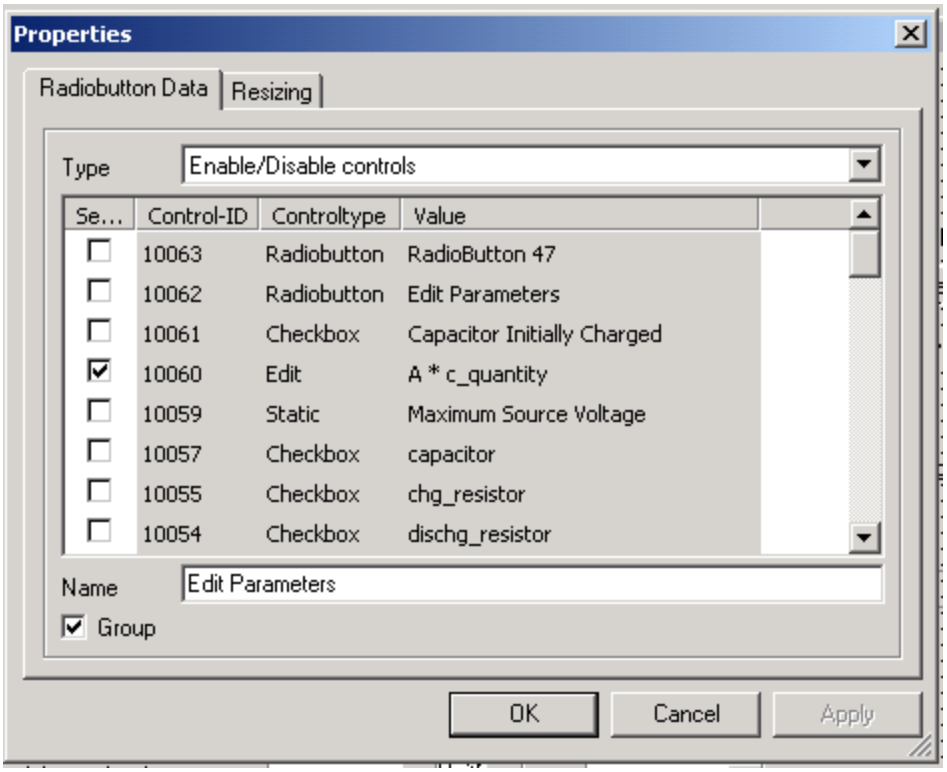
Radio buttons are control elements that provide a set of mutually exclusive choices. In the **Component Dialog Wizard**, you can create radio buttons for two purposes: for enabling and disabling other control elements, and for entering the value of a non-conservative input node. Following is an example of a set of radio buttons as it appears in the **Component Dialog Wizard**.



To create a set of radio buttons for enabling and disabling other control elements within a component dialog box:

1. Select **Controls** > **RadioButton** from the **Component Dialog Wizard** menu bar.
   - You can also click **Add Radio Button** on the **Standard** toolbar.
2. Click the component dialog box to place the radio button.
3. Drag the radio button to its desired location in the component dialog box.
4. Resize the radio button to its desired height and width.
5. Repeat steps **1** through **4** for the remaining radio buttons in the set.
6. Select **Layout** > **Set Tab Order** from the **Component Dialog Wizard** menu bar.
   - You can also click **Set Tab Order** on the **Layout** toolbar.
7. Note the tab order number of each newly placed radio button in the group.
8. Right-click the radio button with the lowest tab order number. The **Properties** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



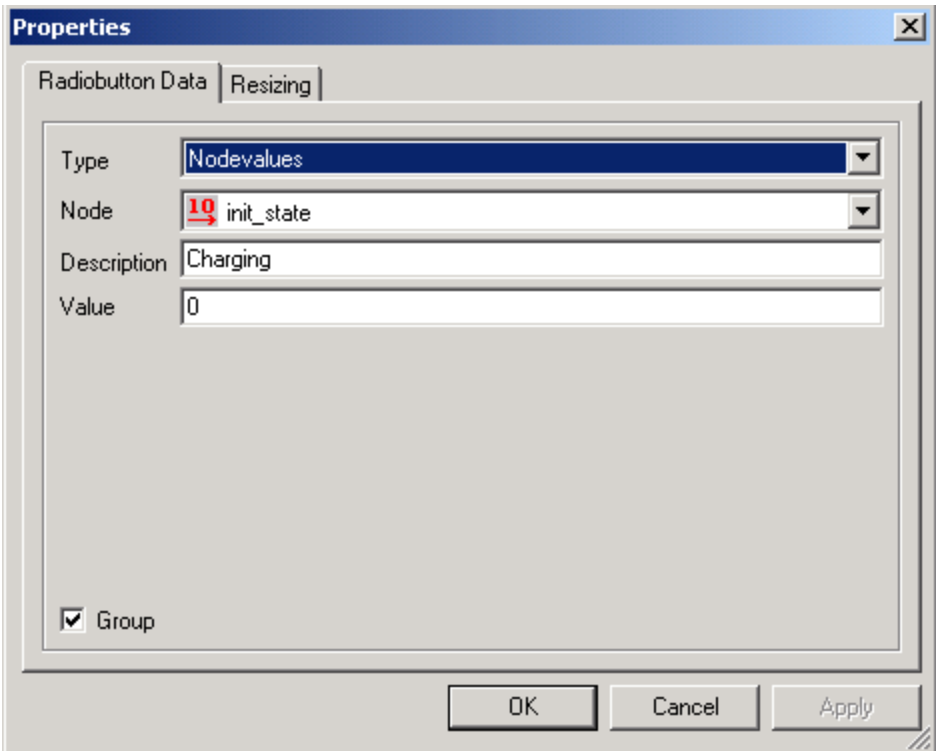
9. Click the **Radiobutton Data** tab.
10. Select **Type** > **Enable/Disable controls**.
11. Select a check box in the **Select** column for each control element you want to enable when the radio button is selected. If you want no controls to be enabled, select no boxes in this column.
12. Type a name to accompany the radio button in the **Name** field.
13. Select the **Group** check box.
14. Repeat steps **1** through **12** for each remaining radio button in the set, but do not select the **Group** check box for any of these buttons. Only the radio button with the lowest tab order number in the set should have its **Group** box selected.

To create a set of radio buttons for entering the value of a non-conservative input node:

1. Select **Controls** > **RadioButton** from the **Component Dialog Wizard** menu bar.
   - You can also click **Add Radio Button** on the **Standard** toolbar.
2. Click in the component dialog box to place the radio button.
3. Drag the radio button to its desired location in the component dialog box.
4. Resize the radio button to its desired height and width.
5. Repeat steps **1** through **4** for the remainder of the radio buttons in the set.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. Select **Layout** > **Set Tab Order** on the **Component Dialog Wizard** menu bar, or click **Set Tab Order** on the **Layout** toolbar.
7. Note the tab order number of each newly placed radio button in the group.
8. Right-click the radio button with the lowest tab order number. The **Properties** dialog box opens.



9. Select **Type** > **Nodevalues**.
10. Select the name of the non-conservative input node to be controlled by the set of radio buttons from the **Node** list.
11. Select a name for the radio button and type it into the **Description** field.
12. Select the value for the chosen non-conservative input node when the radio button is selected and type it into the **Value** field.
13. Select the **Group** check box.
14. Repeat steps **1** through **12** for each remaining radio button in the set, but do not select the **Group** check box for any of these buttons. Only the radio button with the lowest tab order number in the set should have its **Group** box selected.

Adding List Boxes to a Component Dialog Box

A list box is a control element that provides a table of component parameter information and controls. In the **Component Dialog Wizard**, list boxes can display, edit, and control model

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

information such as:

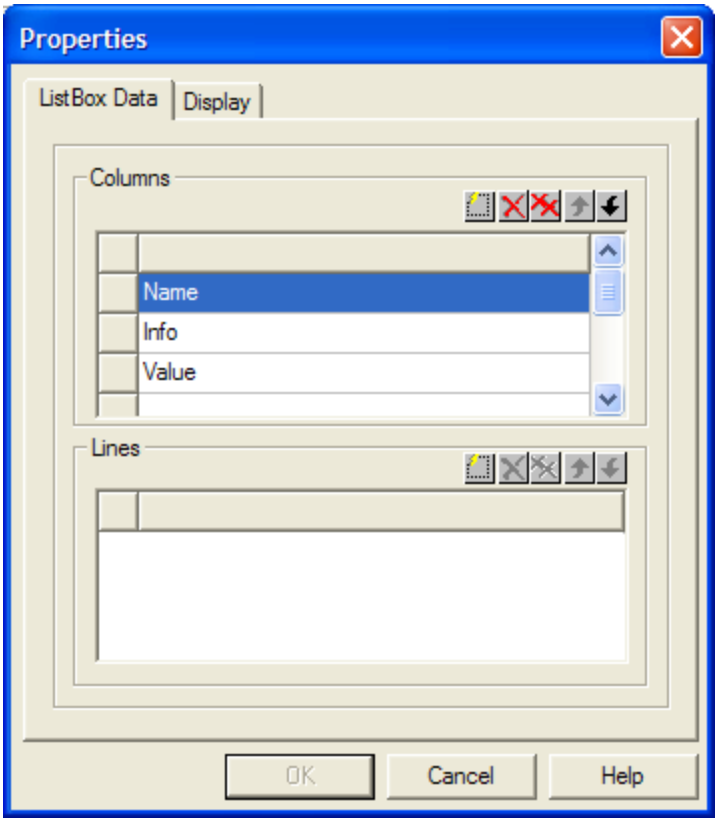
- Values of non-conservative input nodes.
- Units of non-conservative input and output nodes.
- Descriptions of conservative and non-conservative pins.

The following is an example of a list box as it appears in the **Component Dialog Wizard**:

To create a ListBox for displaying and/or changing the values of parameters:

1. Select **Controls** > **ListBox** from the **Component Dialog Wizard** menu bar.
2. Click the component dialog box to place the list box.
3. Drag the list box to its desired location on the component dialog box.
4. Resize the list box to its desired height and width.
5. Right-click the list box and select **Properties**. The **Properties** dialog box appears, displaying two lists on the **ListBox Data** tab.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



6. In the **Columns** list, click and choose the columns that will appear in the actual dialog box from the menu that appears.
   - deletes the currently selected list item.
   - deletes all entries in the list.
   - moves the currently selected list item up or down in the list. You can also drag and drop list items.
7. Similarly, in the **Lines** list, click and choose the parameters that will appear in the actual dialog box from the menu that appears.
   - deletes the currently selected list item.
   - deletes all entries in the list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- moves the currently selected list item up or down in the list. You can also drag and drop list items.

8. Repeat steps **6** and **7** for the remaining entries of the **ListBox**.
9. Click **OK**.

Determining a Non-conservative Input Node's Value Using Free Values

Component dialog boxes created using the **Component Dialog Wizard** allow values to be passed to a model's non-conservative input nodes by two methods.

- Direct passing or assignment of values by control elements such as text edits and check boxes.
- Indirect passing or assignment of values using **Free Values**: parameters that are passed to the component dialog wizard, which then uses the values of these parameters to calculate the values passed to some or all of the model's non-conservative nodes.

This section describes the procedure for assigning a value to a non-conservative input node using free values.

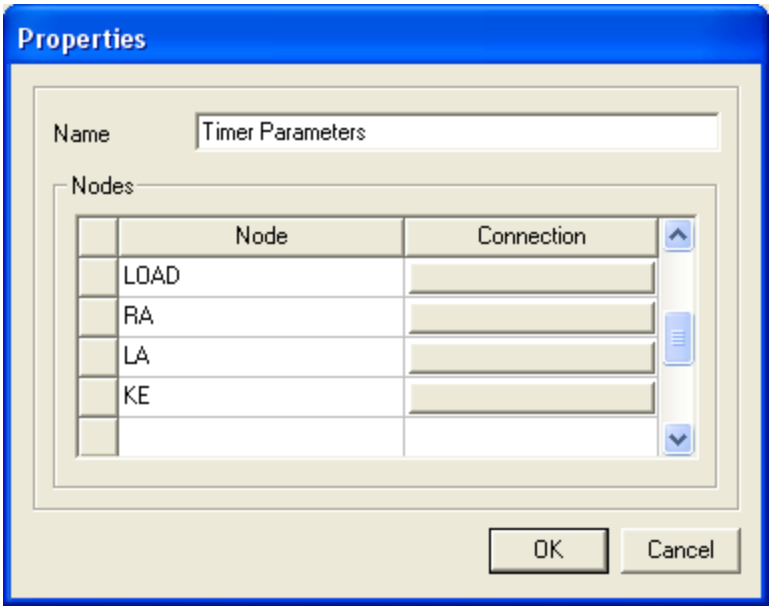
Create free values in a component dialog box using the text edit, ComboBox and check box control elements (for a detailed explanation of how to create free values using these control elements, see ***Adding Static Information and Control Elements to a Component Dialog***). Once you have added free values, they can determine the value of any non-conservative input node not already connected to another control element in a component dialog box.

To determine the value of a non-conservative input node using free values:

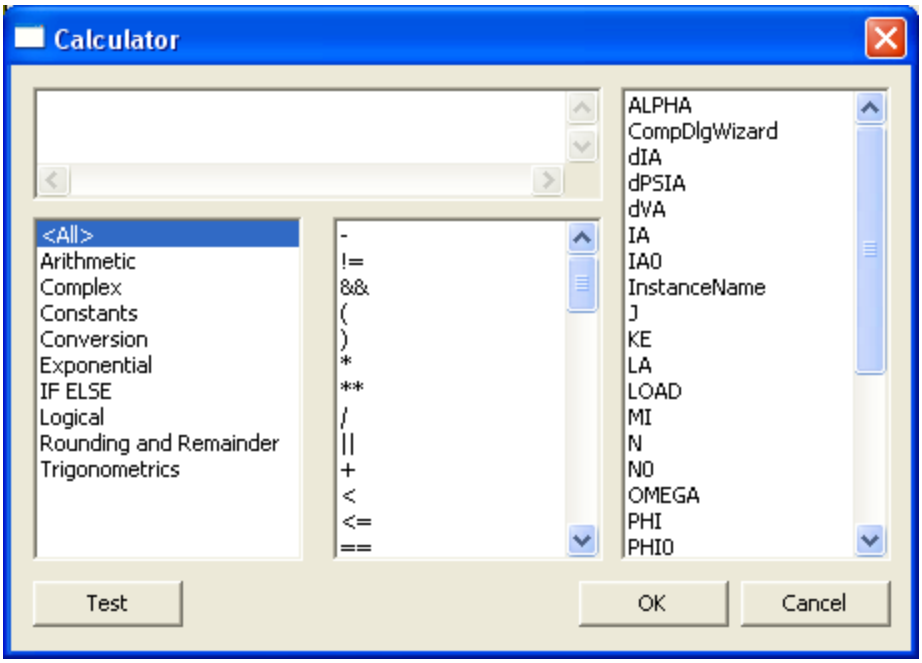
1. Start the **Component Dialog Wizard**.
2. Add any necessary free values.
3. Right-click an empty portion of the component dialog and select **Properties**.

   The **Properties** dialog box opens, and a list of the model's non-conservative input nodes appears in the **Nodes** panel. In the **Connection** column, there is a browse edit button to the right of any input node not currently connected to any other control element. The values for such input nodes can be determined by free values. For example, in the illustration below, the values for nodes LOAD, RA, LA, and KE all can all be determined using free values.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. Click the browse edit button to the right of the input node that is to have its value determined using free values. The **Calculator** dialog box opens.



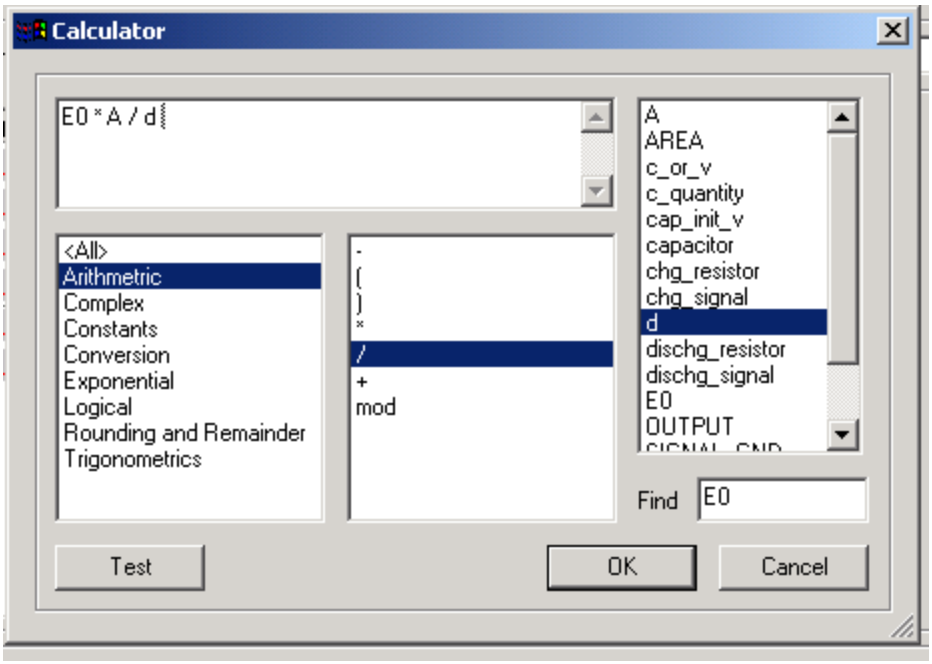
5. Enter an expression for the value of the non-conservative input node in the upper-left panel. The expression should be written in terms of free values, non-conservative input values, variables, and Twin Builder mathematical and logical operators, functions, and constants. Type or enter these expressions with the utilities provided in the **Calculator**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

dialog box:

- The lower-left panel contains a list of the various categories of Twin Builder operators, functions, and constants. Click a category to filter the display in the center window to the operators, functions, and constants associated with that category. For example, if you click **Trigonometrics** in the left window, the list of Twin Builder trigonometric functions appears in the center window.
- The center panel contains a list of all Twin Builder mathematical and logical operators, functions, and constants. Double-click a symbol in this window to add it to the expression.
- The upper-right panel contains a list of all conservative nodes, non-conservative nodes, and free values. Double-click a name in this window to add it to the expression.

In the example shown below, the expression (E0 * A)/d is assigned to the non-conservative node capacitor. The name of the non-conservative node does not appear in the upper-left window with the rest of the expression.



6. Click **Test** when the expression has been entered. If any SML syntax errors exist in the expression, the **Component Dialog Wizard** indicates the source of the error so that you can correct it.
7. Click **OK** when the expression is error free.
8. Repeat steps **1** through **7** for all non-conservative input nodes whose values are to be calculated using free values.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Aligning and Sizing Component Dialog Box Elements

The **Component Dialog Wizard** uses three utilities to facilitate precise alignment and sizing of component dialog box elements:

- The grid.
- Lock controls.
- Size controls.

Component Dialog Box Grid

The grid provides a set of convenient reference points for placing information and control elements on the control dialog box, ensuring consistent spacing, and sizing elements.

- By default, the grid appears when the **Component Dialog Wizard** starts. If the grid does not display, select **Layout** > **Show Grid**. Select **Show Grid** to toggle display of the grid. Select **Wizard** > **Options** from the **Component Dialog Wizard** menu bar to adjust spacing between points on the grid.

The **Properties** dialog box opens.

In this dialog box, enter an integer between **2** and **100** in the **Grid Size** field to adjust the spacing between grid points. A value of **2** provides the smallest grid spacing, **100** the largest.

Component Dialog Box Lock Controls

Lock controls help you align one or more component dialog box elements along a single guide element. You can align elements with the top, bottom, left, or right of the guide element.

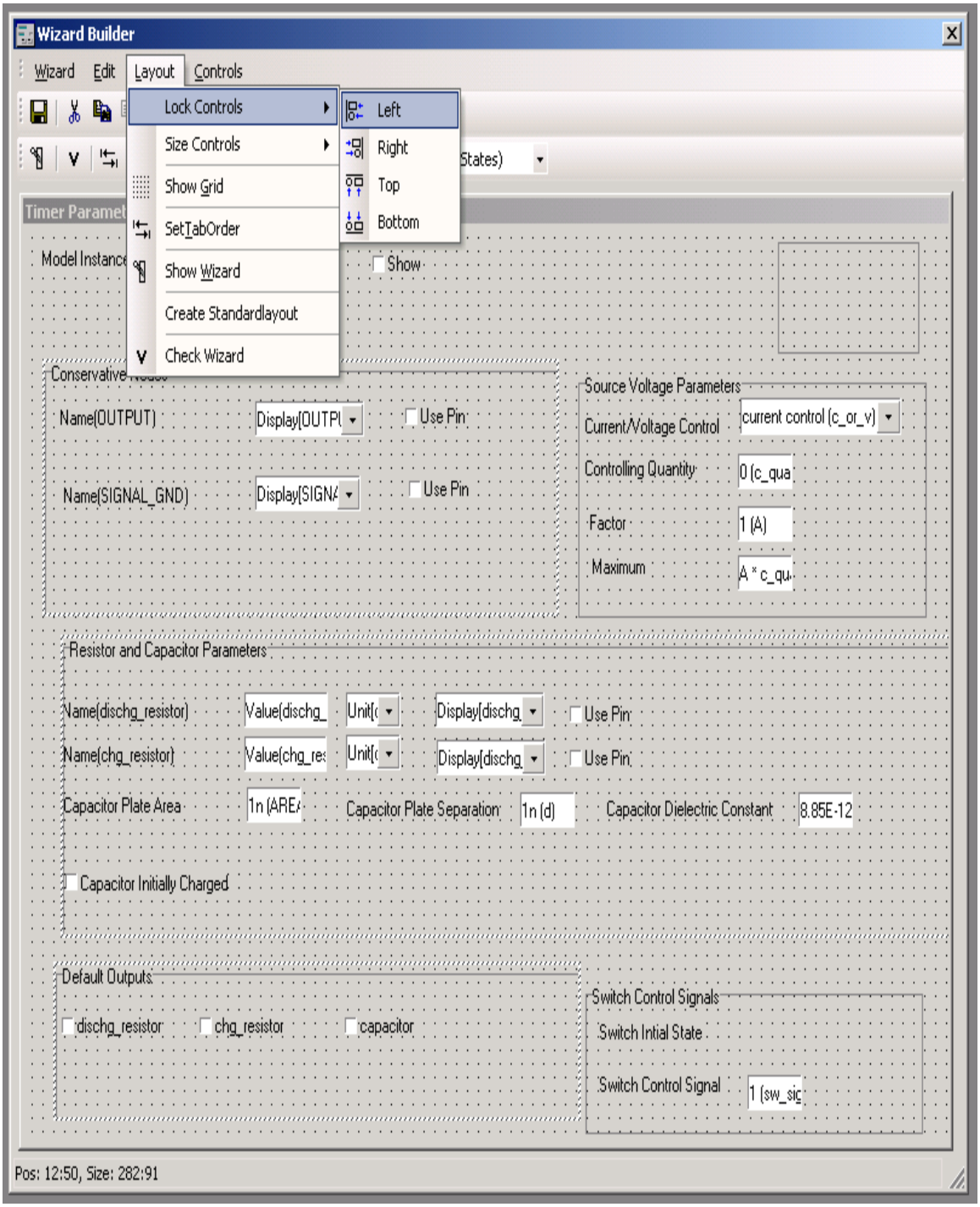
To use the lock controls:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select the element or elements to be aligned with the single selected element. To do this, use the mouse to draw a rectangle around the desired elements, or press Shift and click each element individually.
2. Select the guide element. To do this, press Shift while the other elements are selected and click the guide element. The guide element must be selected last.
3. Choose a lock control by selecting **Layout** > **Lock Controls**, or click the appropriate lock control on the **Layout** toolbar.
4. Four lock controls are available:
   - **Left** – The left edges of the selected elements become aligned with the left edge of the guide element
   - **Right** – The right edges of the selected elements become aligned with the right edge of the guide element.
   - **Top** – The tops of the selected elements become aligned with the top of the guide element.
   - **Bottom** – The bottoms of the selected elements become aligned with the bottom of the guide element.

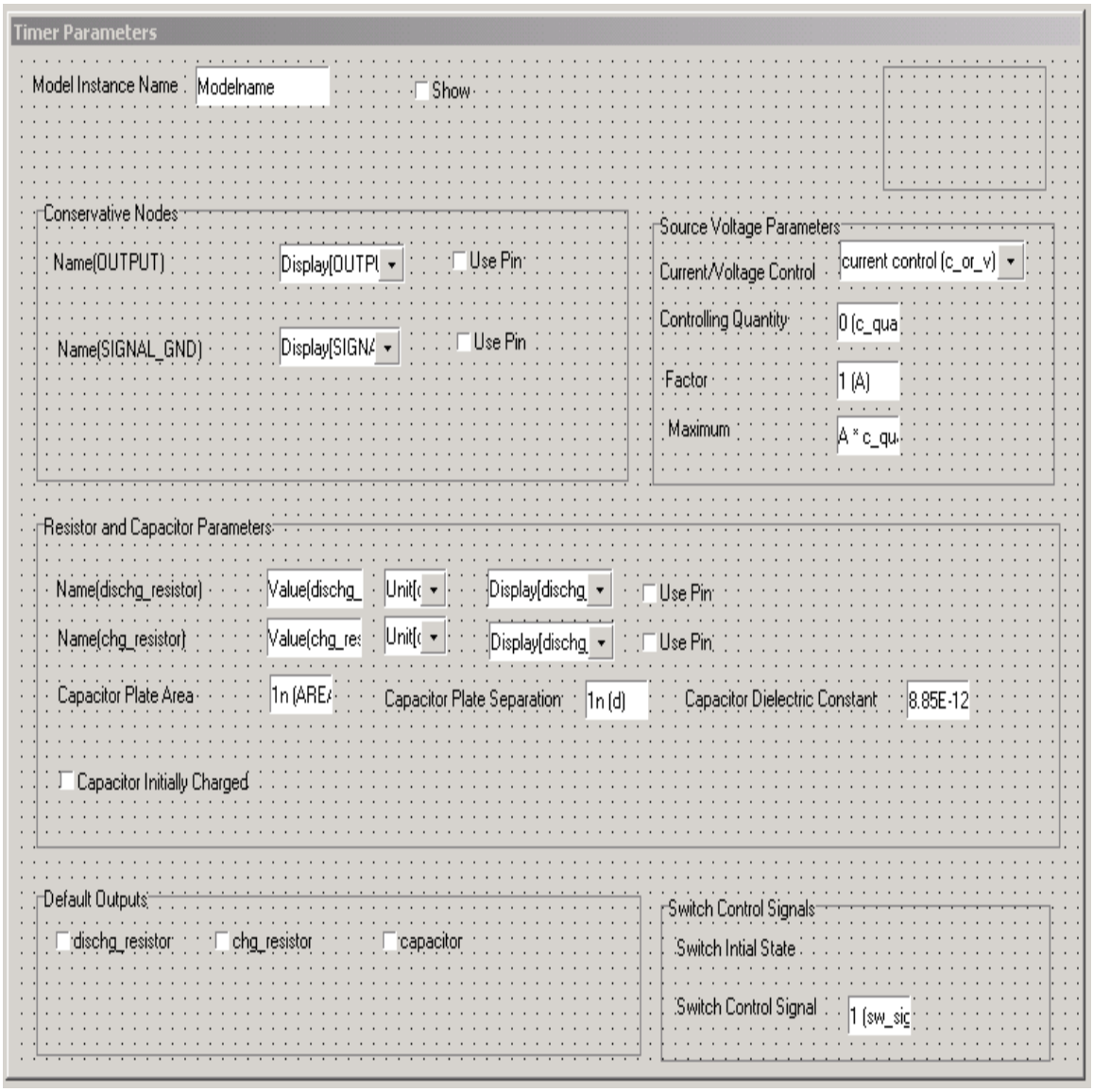
   The first illustration in the following example shows the groups **Resistor and Capacitor Parameters** and **Default Outputs** selected to be aligned with the left edge of the group **Conservative Nodes**. The **Left** lock control is selected from the lock control menu to perform the alignment.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The second illustration shows the three groups in left alignment after the lock to the left has occurred.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Component Dialog Wizard Size Controls

The size controls facilitate the sizing of component dialog box elements so that they are the same size as a selected reference element.

To use the size controls:

1. Select the element or elements to be sized to the reference element. Draw a rectangle around the desired elements, or press Shift and click each element individually.
2. Select the reference element. Press Shift while the other elements are selected and click the reference element.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> The reference element must be selected last.

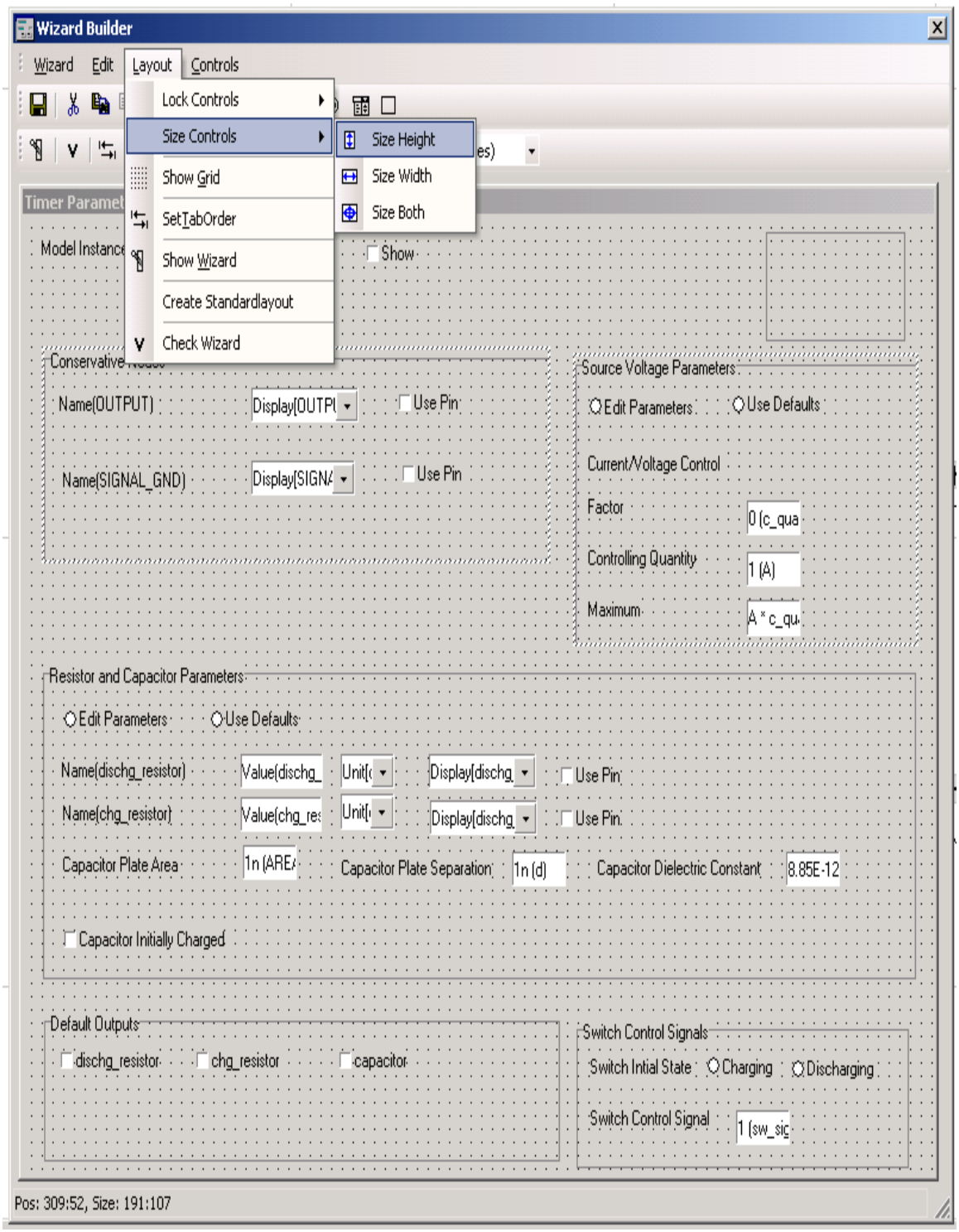
3. Select **Layout** > **Size Controls**to choose a size control. You can also click the appropriate size control on the **Layout** toolbar.

   There are three size controls available:

   - **Size Height** – The selected elements will have the same height as the reference element.
   - **Size Width** – The selected elements will have the same width as the reference element.
   - **Size Both** – The selected elements will have the same height and width as the reference element.

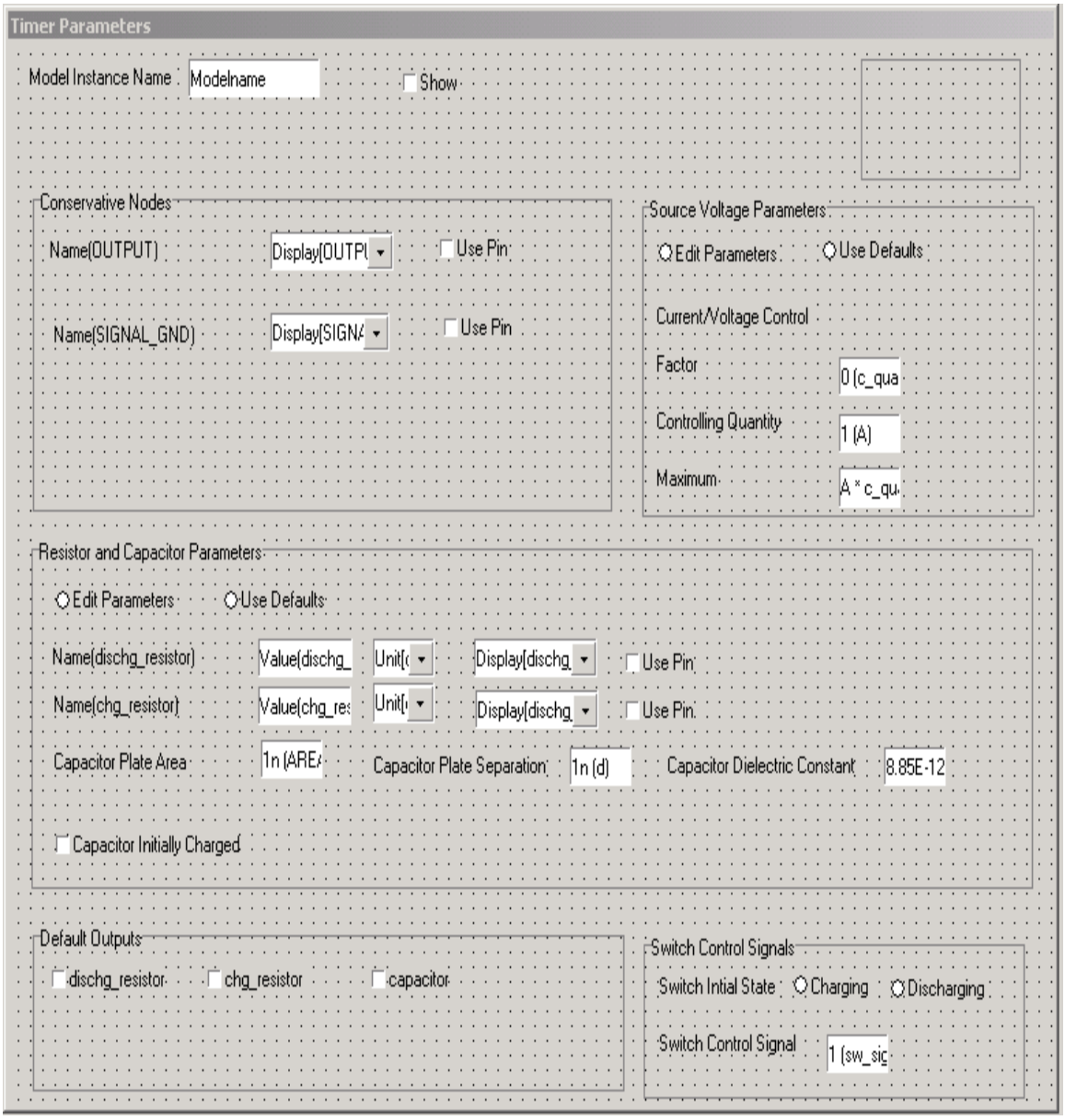
   The first illustration in the following example shows the group **Conservative Nodes** selected to be sized to the height of the group **Source Voltage Parameters**. Select **Size Height** from the size control menu to accomplish the size change.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The second illustration shows the two groups with equal height after the sizing has occurred.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

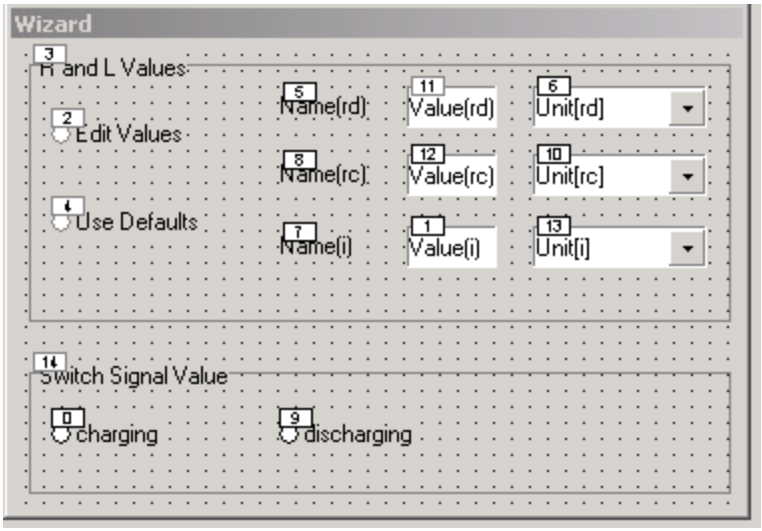


Setting the Tab Order of a Component Dialog Box

The tab order of a component dialog box is the order in which the control elements are selected when you press Tab. The **Component Dialog Wizard** provides a utility for changing the tab order of a component dialog box created with the Wizard.

The illustration below shows the tab order of a component dialog box. The radio button **charging** is first in the order, and the group **Switch Signal Value** is last.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



To set the tab order of a component dialog box:

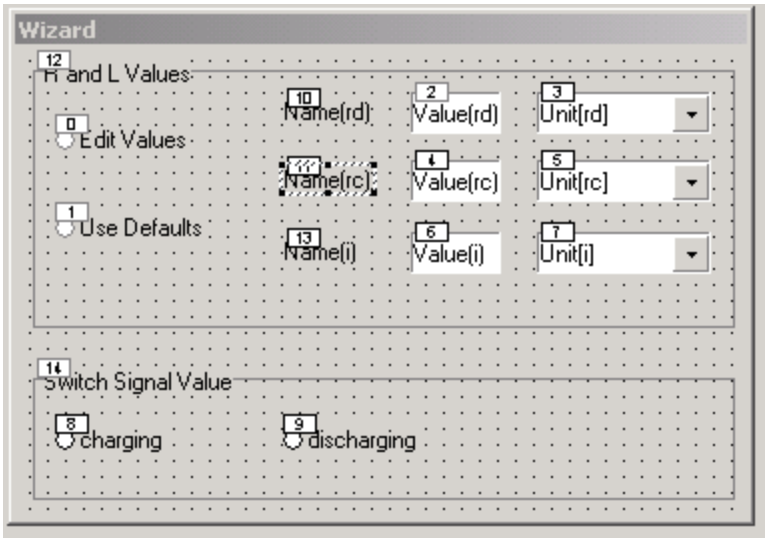
1. Start the **Component Dialog Wizard**.
2. Select **Layout** > **SetTabOrder**.
   - Alternately, click **Set Tab Order** on the **Layout** toolbar (for more information on the **Component Dialog Wizard** menu bar and toolbars, see Section XII: Reference).

   The current tab order of the component dialog box is shown. The first element in the tab order is **0**, and the elements selected with each subsequent pressing of **Tab** are numbered in ascending numerical order.
3. Click the component element that is to be first in the new tab order.

   The number of the element should change to **0**.
4. Click the component that is to be second in the new tab order.

   The number of the element should change to **1**.
5. Continue selecting elements until all elements are in the correct order. The illustration below shows the same component dialog box shown above with a newly set tab order:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



6. Click an empty portion of the component dialog to exit the tab order utility.

Setting the Display Behavior of Component Dialog Box Elements

The **Component Dialog Wizard** makes it possible to control how the sizes and positions of component dialog box elements change when the component dialog box containing them is resized. It also enables control of when component dialog box elements are disabled or hidden. This section describes the dialog box display behaviors available for component dialog box elements, and how to apply these behaviors to individual elements.

Component Dialog Box Display Behaviors

The three dialog box display behaviors available for component dialog box elements are:

- **Lock** – The element maintains a constant distance between its borders and selected borders of the component dialog box. The element may change size in order to do this.
- **Relative** – The element maintains its size and relative distance from the selected borders of the component dialog box as the dialog box is resized.
- **Default** – The element maintains its size. An element with this behavior does not maintain its relative distance from other elements with default behavior as the dialog box is resized.

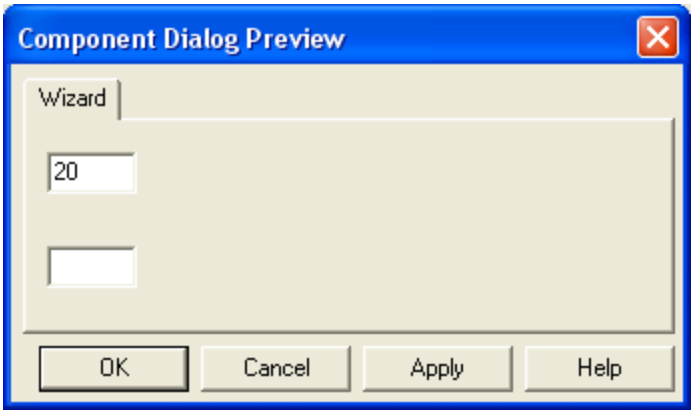
Lock Behavior

Lock behavior causes a dialog box element to maintain a fixed distance from the borders of the dialog box. The first pair of illustrations shows a text edit element with **Lock** behavior selected for

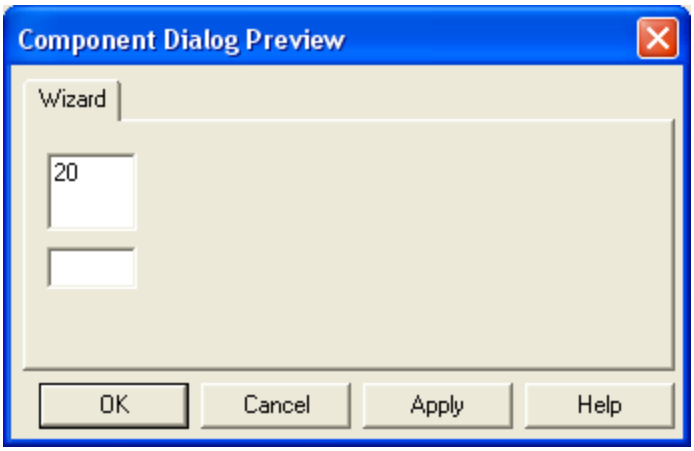
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the top and bottom dialog box borders. For comparison, a second text edit element of equal size with **Default** behavior is also shown. The upper text edit is the one for which **Lock** behavior has been selected.

Lock Behavior - Before Resizing



Lock Behavior - After Resizing



The dialog box has been resized by dragging its bottom border downward. In response, the upper text edit has maintained its position relative to the top and bottom borders of the dialog box. It has changed size in order to maintain a constant distance from both the top and bottom borders of the dialog box. In contrast, the text edit with **Default** behavior has moved in the direction of the resizing of the dialog box, and the distance between it and the top and bottom borders of the dialog has changed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Relative Behavior

Relative behavior causes a locked component dialog box element to maintain its relative distance from the selected dialog box border. The next pair of illustrations shows two combo box elements with relative behavior selected with respect to the left border of the dialog box. For comparison, two combo boxes of equal size with **Default** behavior are also shown. The upper combo boxes are the ones for which **Relative** behavior has been selected. These combo boxes are set to maintain their left edges at the same relative distance from the left border of the dialog box as it is resized.

Relative Behavior - Before Resizing

Relative Behavior - After Resizing

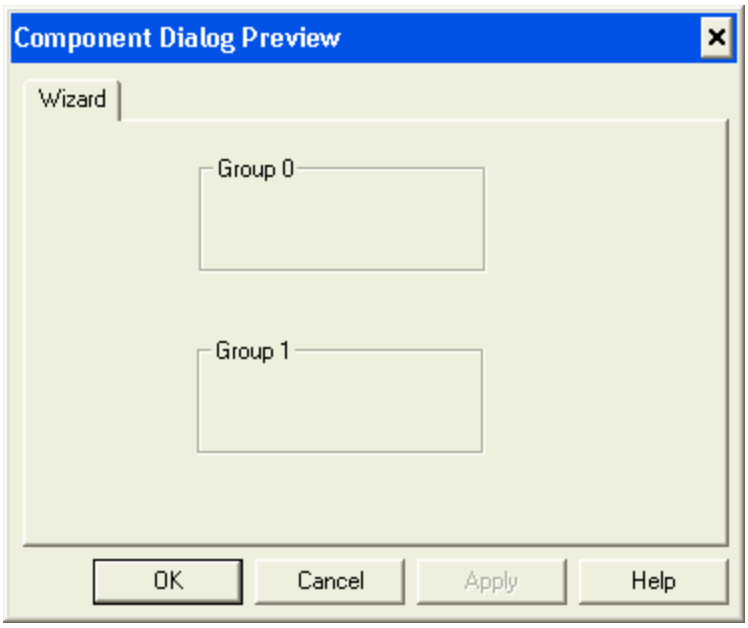
The dialog box has been resized by dragging its right border to the right. In response, the upper combo boxes have moved so that their relative distances to the left border of the dialog box remains the same. In contrast, the lower combo boxes set for **Default** behavior have not moved.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Default Behavior

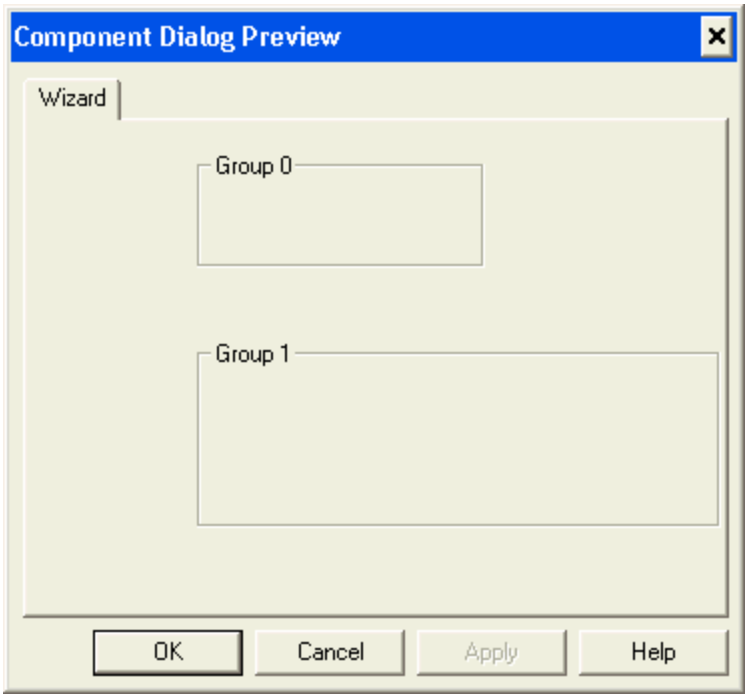
Default behavior, which is the behavior shown by a dialog box element if neither **Lock** nor **Relative** behaviors are selected, causes a dialog box element to maintain its size and its position relative to the dialog box top and left edges when the dialog box containing it is resized. The next pair of illustrations shows a group element with **Default** behavior. For comparison, a group element with **Lock** and **Relative** behaviors is shown. The group with **Default** behavior is the upper group in the illustration.

Default behavior - Before Resizing



Default Behavior - After Resizing

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The dialog box has been resized by dragging its bottom border downwards and by dragging its right border to the right. The upper group with **Default** behavior has not changed in size and has also maintained its position relative to the left and top edges of the dialog box. The bottom group with **Lock** and **Relative** behavior has changed its size in proportion to the change in size of the dialog box and has maintained its position relative to the left edge of the dialog box.

Selecting the Dialog Box Display Behavior of a Dialog Element

Setting the display behavior of dialog box elements includes:

- Resizing behavior.
- Conditions for display.

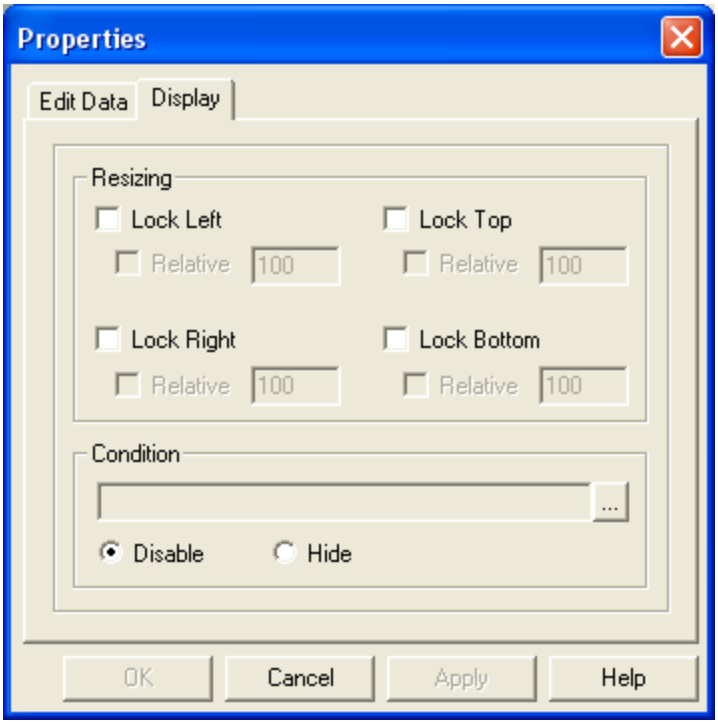
Selecting Dialog Box Resize Behavior

To select the resize behavior of a dialog box element:

1. Right-click the element and select **Properties**. The **Properties** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Select the **Display** tab.



3. Choose the desired resizing behaviors by selecting the appropriate check boxes. You can select **Relative** only if **Lock** is enabled. The **Relative** behavior number is the percentage of the dialog box width (or height) maintained by the element's edge relative to the chosen dialog box edge. For example, if **Lock Right** is selected, and its **Relative** number is set to **75**; as the dialog box is resized horizontally, the right edge of the element will move so that it is always **75** percent of the distance between the left and right edges of the dialog box. If no boxes are selected, the element exhibits **Default** behavior in both the width and height dimensions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

- It is possible to achieve more than one resize behavior for a dialog box element. For example, an element can be set to **Lock** to the left dialog box border and to resize in the height dimension by also choosing to **Lock** to the top (or bottom) border.
- Dialog box elements do not exhibit their resize behaviors when the dialog box constructed in the **Component Dialog Wizard** is resized. They will exhibit their resize behaviors only when a component dialog box is resized in the Wizard's preview mode (See Section VIII: Testing a Component Dialog for more details), or when you access a component dialog box from a Twin Builder schematic.
- Preview the dialog box to be sure that the intended resize behavior has been achieved.

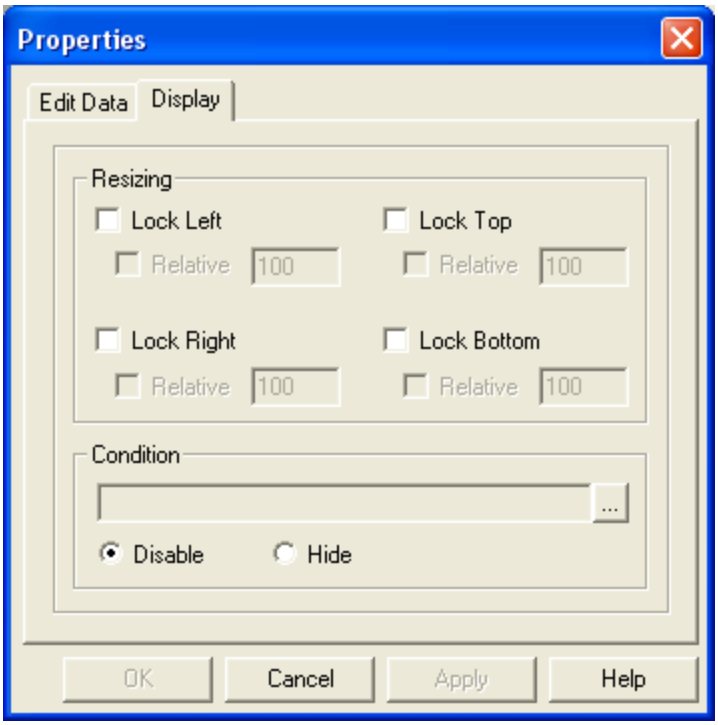
4. When finished, click **OK**.

Setting Dialog Box Element Conditions for Display

By default, dialog box elements are both enabled for use and visible. To set a **Condition** for disabling or hiding a dialog box element:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

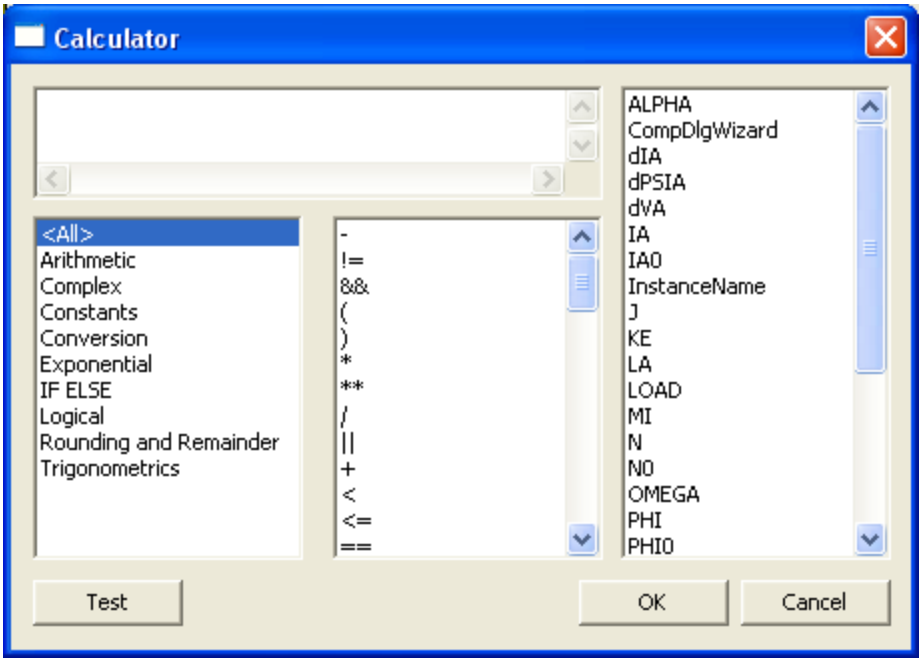
1. Right-click the element and select **Properties**. The **Properties** dialog box appears.
2. Click the **Display** tab.



3. Select the action you want to apply to a dialog box element when the **Condition** is satisfied. **Disable** keeps the element visible while preventing you from making changes to it. **Hide** removes the element from view.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Click [**...**]. The **Calculator** dialog box appears

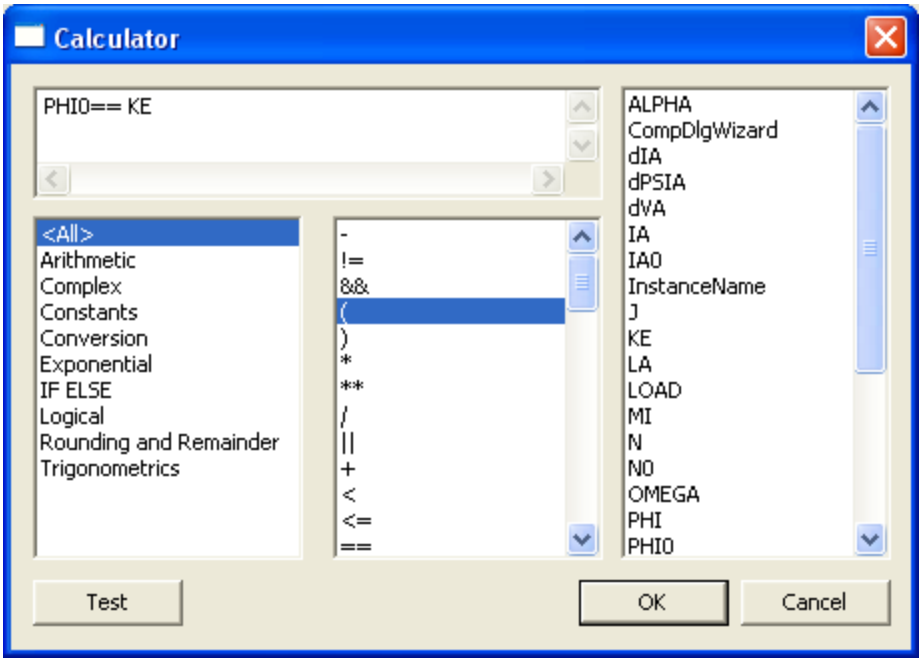


5. Enter an expression for the desired condition for disabling or hiding the element in the upper-left panel. Compose the expression in terms of free values, non-conservative input values, variables, and Twin Builder mathematical and logical operators, functions, and constants. Enter or type the expression with the utilities provided in the **Calculator** dialog box. These utilities are:
   - The lower-left panel lists the various categories of Twin Builder operators, functions, and constants. Click a category to filter the display in the center panel to the operators, functions, and constants associated with that category. For example, if you click **Trigonometrics** in the left panel, the list of Twin Builder trigonometric functions appears in the center panel.
   - The center panel lists all Twin Builder mathematical and logical operators, functions, and constants. Double-click an entry to add it to the expression.
   - The panel on the right lists all conservative nodes, non-conservative nodes, and free values. Double-click a name to add it to the expression.

   > **Note:** For a component property with multiple choices, you can create a condition using an index of choice. For example, for property **SimulationModel**, the condition could be **SimulationModel == 1**.

   In the example shown below, the expression **PHIO==KE** has been entered. This condition is met whenever the value of PHIO is identical to that of KE.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



6. Click **Test** when the expression is complete. If any SML syntax errors exist in the expression, the wizard indicates the source of the error.
7. Click **OK** when the expression is error free.
8. Repeat steps **1** through **7** for all non-conservative input nodes whose values are to be calculated using free values.

Testing a Component Dialog Box

The **Component Dialog Wizard** provides two utilities for testing component dialog boxes created with it:

- **Show Component Dialog** tests a dialog box's appearance.
- **Check Component Dialog** tests its completeness.

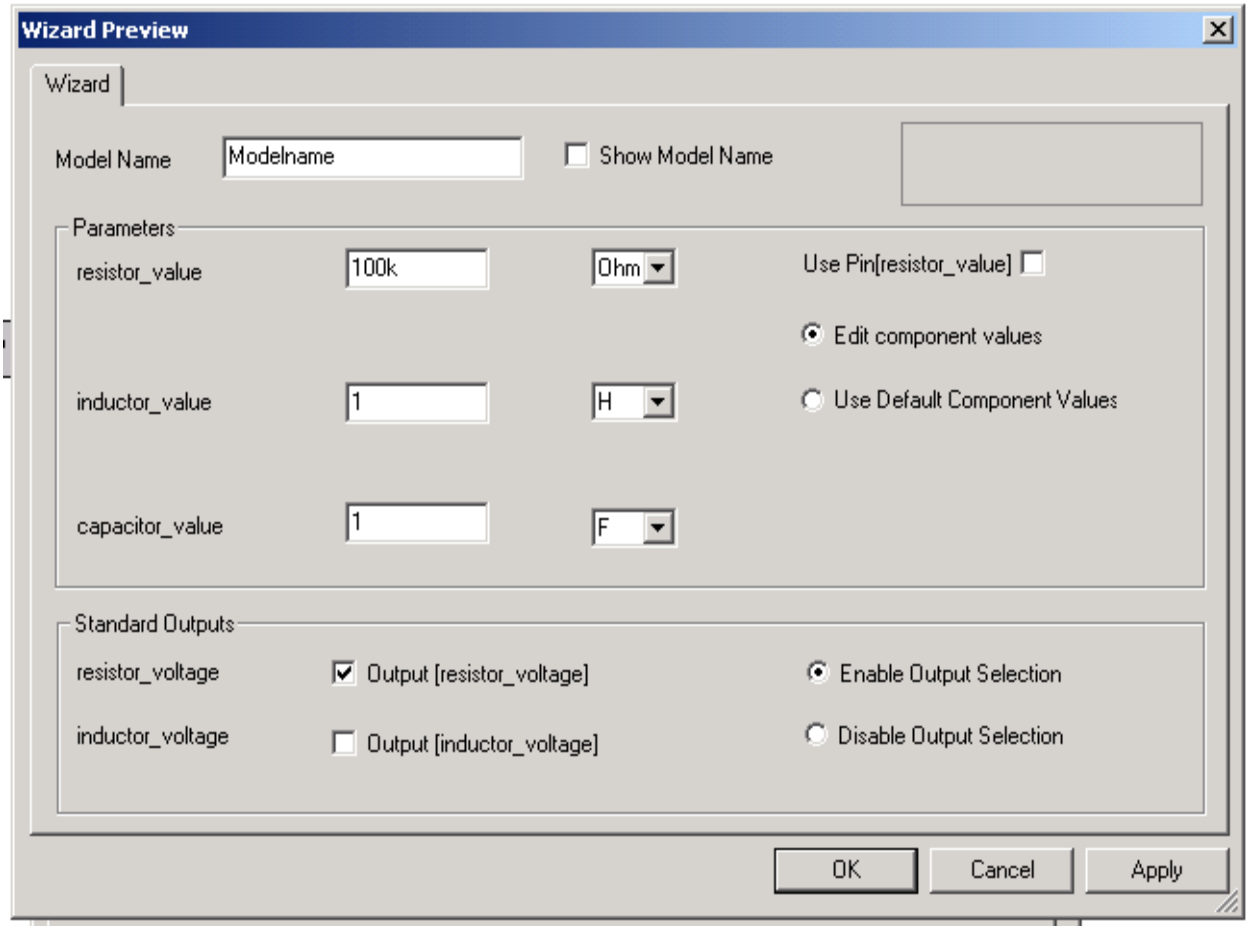
Using Show Component Dialog

The **Show Component Dialog** utility generates a preview of the component dialog box as it would appear when opened from a schematic.

1. Select **Layout** > **Show Component Dialog** on the **Component Dialog Wizard** menu bar. You can also click **Show Component Dialog** on the Wizard's **Layout** toolbar.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The **Component Dialog Preview** dialog box opens, displaying a preview of the component dialog box.



Use the preview to test the functionality of the component dialog box. For example, values in text edits may be retyped, radio buttons and check boxes may be selected and cleared, and values in combo boxes may be changed. You can also test the dialog box resize behavior by resizing the preview window.

2. When finished testing, click **OK** to save your work and close the **Component Dialog Preview** dialog box, click **Apply** to save your work and continue to work within the dialog box, or click **Cancel** to close the dialog box without saving your changes.

Using Check Component Dialog

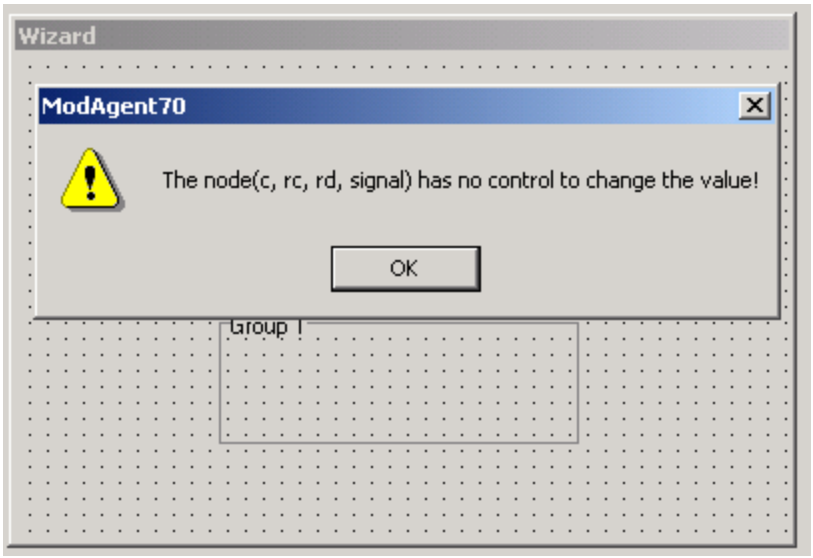
The **Check Component Dialog** utility is used to determine the overall completeness of a component dialog box. Specifically, the utility determines if the component dialog box provides access to all of a model's non-conservative input nodes, and if all of the dialog box's free values

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

are used. If the component dialog box is not complete, the **Component Dialog Wizard** returns an error message indicating what is not complete in the dialog box.

1. Start the **Check Component Dialog** utility by selecting **Layout** > **Check Component Dialog** on the **Component Dialog Wizard** menu bar.
   - You can also click **Check Component Dialog** on the Wizard's **Layout** toolbar.

   The illustration below shows an example of an error message returned by the **Check Component Dialog** utility:



   The message indicates that the non-conservative input nodes **c**, **rc**, **rd**, and **signal** are not accessible through any control element available on the component dialog box.

2. To make the component dialog box complete, add control elements that connect to these non-conservative nodes.

Saving and Closing a Component Dialog Box

When a component dialog box is complete, save and close it.

1. To save the dialog box, select **Wizard** > **Save** on the **Component Dialog Wizard**'s menu bar.
   - Alternately, click **Save** on the Wizard's standard toolbar.
2. To close the **Component Dialog Wizard** select **Wizard** > **Close** on the Component Dialog Wizard's menu bar.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Using a Component Dialog Box

After a component has been placed on a schematic, use its component dialog box to change many of its parameters.

1. Double-click the desired component on a schematic sheet. The component dialog box appears.
2. To use a component dialog box, right-click the desired component on a schematic sheet and select **Properties**. The model's **Properties** dialog box appears.
3. On the **Parameter Values** tab, click **CompDlgWizard** to open the component dialog box.
4. Once the component dialog box opens, all of the functionality of the component dialog box is available for use.

Revising a Component Dialog Box

Component dialog boxes created with the **Component Dialog Wizard** may be revised at any time. To change the contents of a component dialog box, start the **Component Dialog Wizard** and change the appearance using the methods described in the previous sections of this tutorial.

Component Dialog Wizard Window Items

This section describes the menus and toolbars of the **Component Dialog Wizard**.

Related Topics

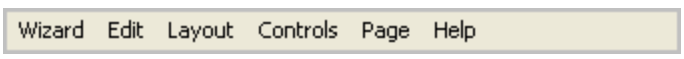
Component Dialog Wizard Menu Bar

Component Dialog Wizard Standard Toolbar

Component Dialog Wizard Layout Toolbar

Component Dialog Wizard Menu Bar

The menu bar contains menus for the **Component Dialog Wizard**. Each menu is described in the following sections.

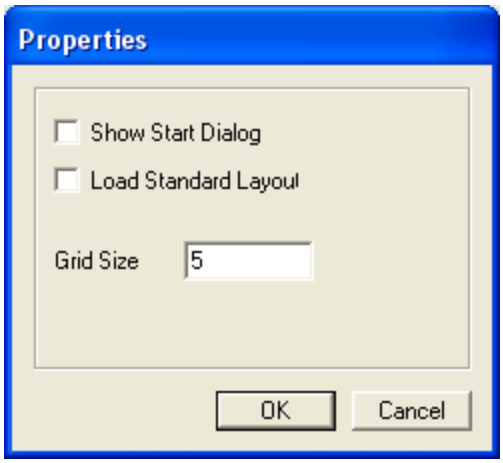


Wizard Menu (Component Dialog Wizard)

The Wizard menu contains the following general commands:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Delete** – Removes the component dialog box created with the Wizard from subsequently placed instances of the model. The changes made to the component dialog box with the Wizard are destroyed.
- **Save** – Saves all changes made to the element dialog box.
- **Options** – Opens a dialog box (shown below) that features three controls for the Component Dialog Wizard.



  - **Show Start Dialog** – Choose whether to load the standard layout when the **Component Dialog Wizard** starts.
  - **Load Standard Layout** – Load the standard layout when the **Component Dialog Wizard** starts.
  - **Grid Size** – Enter an integer between 2 and 100 to determine the spacing of the **Component Dialog Wizard**'s grid, where 2 is the smallest (closest) spacing, and 100 is the largest.
- **Close** – Close the **Component Dialog Wizard**.

Edit Menu (Component Dialog Wizard)

Use the **Edit** menu to select, copy, cut, paste, and delete component dialog box elements.

Layout Menu (Component Dialog Wizard)

The **Layout** menu provides the following commands for positioning and sizing component dialog box elements, and for testing a component dialog box:

- **Lock Controls** – Contains the **Left**, **Right**, **Top**, and **Bottom Lock** commands.
- **Size Controls** – Contains the **Size Width**, **Size Height**, and **Size Both** commands.
- **Show Grid** – Toggles display of the **Component Dialog Wizard** grid.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **SetTabOrder** – Starts the utility for setting the tab order of component dialog box elements.
- **Show Component Dialog** – Generates a preview of the component dialog box.
- **Create Standard Layout** – Replaces all elements currently in the component dialog box with the standard layout.
- **Check Component Dialog** – Starts the utility for checking if the component dialog box is complete with respect to non-conservative input nodes and free values.

Controls Menu (Component Dialog Wizard)

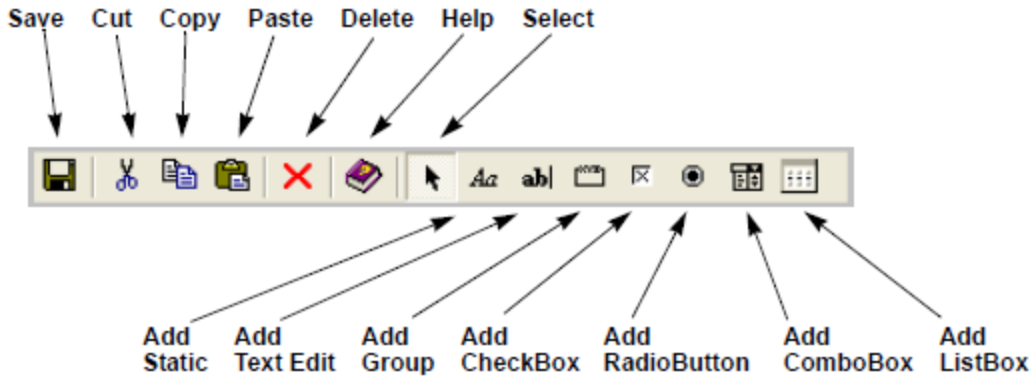
Use the **Controls** menu to add the seven available control elements—**CheckBox**, **ComboBox**, **ListBox**, **Edit**, **Group**, **RadioButton**, and **Static**—to a component dialog box.

Page Menu (Component Dialog Wizard)

Use the **Page** menu to **Add**, **Delete**, and **GoTo** pages in the wizard. Each page in the wizard represents a tab in the finished component dialog box.

Component Dialog Wizard Standard Toolbar

You can use the standard toolbar to perform these functions in the **Component Dialog Wizard**:



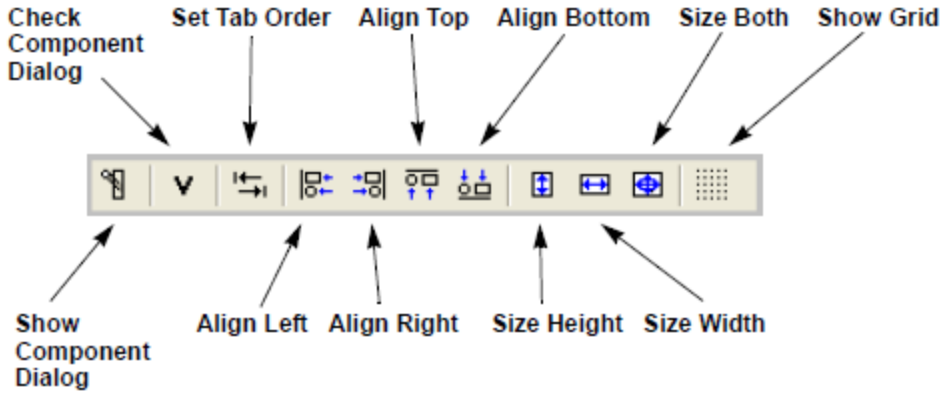
- **Save** – Save all changes made to the component dialog box.
- **Cut**, **Copy**, **Paste**, **Delete** – Perform basic editing functions.
- **Help** – Start help for the **Component Dialog Wizard**.
- **Select** – Return to select object mode.
- **Add Static** – Create a static for the component dialog box.
- **Add Text Edit** – Create a text edit for the component dialog box.
- **Add Group** – Create a group for the component dialog box.
- **Add Checkbox** – Create a check box for the component dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Add Radio Button** – Create a radio button for the component dialog box.
- **Add ComboBox** – Create a combo box for the component dialog box.
- **Add ListBox** – Create a list box for the component dialog box.

Component Dialog Wizard Layout Toolbar

The **Layout** toolbar contains commands for the Component Dialog Wizard.



- **Show Component Dialog** – Generate a preview of the component dialog box.
- **Check Component Dialog** – Start the utility for checking if the component dialog box is complete with respect to non-conservative input nodes and free values.
- **Set Tab Order** – Start the utility for setting the tab order of component dialog box elements.
- **Align Left** – Align all selected dialog box elements with the left edge of a reference element.
- **Align Right** – Align all selected dialog box elements with the right edge of a reference element.
- **Align Top** – Align all selected dialog box elements with the top edge of a reference element.
- **Align Bottom** – Align all selected dialog box elements with the bottom edge of a reference element.
- **Size Height** – Size all selected elements to have the same height as a reference element.
- **Size Width** – Size all selected elements to have the same width as a reference element.
- **Size Both** – Size all selected elements to have the same height and width as a reference element.
- **Show Grid** – Toggle display of the **Component Dialog Wizard**'s grid.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

10 - Working with Variables

Properties and quantities in Twin Builder may be represented by alphabetic or alphanumeric variables. For example, the capacitance property of a capacitor may be set to the value of the variable C1, rather than set to a numeric quantity, like 1.0uF. A property that has been set to a variable is said to be *parameterized*.

In some cases, the use of variables is mandatory. For example, if you want to sweep the current level of a source during analysis, you must set the current level equal to a variable, so the simulator can identify and vary the level as the solution proceeds.

Variable Types

A variable is a numerical value, mathematical expression, or mathematical function that can be assigned to a design parameter in Twin Builder. Variables are useful in the following situations:

- You expect to change a parameter often.
- You expect to use the same parameter value often.
- You intend to run a parametric analysis, in which you specify a series of variable values within a range to solve.
- You intend to optimize a parameter value by running an optimization analysis.

The following types of variables are available in Twin Builder:

<table>
<tr><td>Project Variables</td><td>Project variables are available:
For use in any design in a project.
In design properties or component instance properties.
Project variables are evaluated before simulation and use constants, functions and other project variables in their expressions.
A project variable is identified by its dollar sign ($) prefix, as in $C1. If a project variable $C1 is defined as equal to 4.32pF, a capacitance property for a component anywhere in that project can be set equal to 4.32pF by typing $C1 in the appropriate Value field:<table><tr><th></th><th>Name</th><th>Value</th><th>Unit</th><th>Description</th></tr><tr><td></td><td>C</td><td>$C1</td><td></td><td>Capacitance</td></tr></table>You can manually include the <b>$ symbol in the project variable’s name, or Twin Builder will append the project variable’s name after you define the variable.</td></tr>
<tr><td>Design Properties</td><td>A local variable is usable within a design and is not in a design's public interface (that is, it is not visible on the component representing the design</td></tr>
</table>

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	in a parent circuit). These variables are evaluated *before* simulation and use constants, functions, project variables, design parameters and other design variables in their expressions. For example, if a local variable, **R2**, is defined as equal to 4316 Ohms, a resistance property anywhere in that design can be set equal to 4316 Ohms by entering **R2** in the appropriate **Value** field. • A *parameter default* is similar to a local variable but is in the design's public interface. It has a default value that can be overridden in instances of a design. For example, if three subcircuit instances contain a parameter default C1 that is defined as equal to 11.3pF, C1 may be overridden as 11.8pF in the first instance, overridden as 10.9pF in the second, and left at its default value of 11.3pF in the third. A property value set by means of a parameter default is called a *passed parameter*. These variables are evaluated before simulation. • A *quantity* is similar to a parameter default and is in the design's public interface. The difference is that quantities are evaluated during simulation, based on their “Calculate” setting. Quantities can use other quantities, as well as the basic variable types, in their expressions. *Output* or *in/out* quantities can route circuit component instance or net quantities up into the parent circuit. • A *signal* is similar to a quantity and is in the design's public interface. Design signals are restricted to being of *real* type and *input* only. **Note**: The order of variable declaration is important. A variable that is used before being defined has a value of **0** at that use. Property lists are written to the netlist in sorted order, so that variables that reference other variables in the same property list (Local Variables, Design Parameters, and so on) are written *after* the variables on which they depend.
Object Property Variables	An object property variable is composed of the object's instance name (for example, “C1”, “N001”) plus a period (“.”) plus the name of the property. The property can be any quantity or signal, or any real-valued parameter that is part of the object. Object property variables are usable only within the design containing the object. Examples are “R1.V”, “E2.I”, “N001.S” (for a wire connecting a translational nature pin).
FML_INIT Variables	FML_INIT variables are defined within FML_INIT components and are available for use in expressions in the local design, similar to local variables. These are evaluated once before the start of simulation and cannot use simulation-dependent variables in their expressions. Analysis control variables can be set within an FML_INIT component and will override those values in the solution setup. Care must be taken to avoid unintended consequences if these values are changed. Note that setting Tend in this way may cause problems with spectral plots, continuing a simulation

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	based on **Enable continue to solve** in the setup, or continuing a simulation with a state (KRN) file. **Note**: The order of variable declaration is important, and this order is under user control in FML_INIT components. A variable that is used before being defined has a value of **0** at that use. **Note**: FML and FML_INIT variables override the local variables in design properties if they use the same name. However, this is not recommended and a warning will be issued during netlisting. Reports always give a precedence to a design variable of the same name while showing the simulation results. In such cases, if you want to see results for the FML variables in plots, you can optionally assign it to a new non-conflicting FML variable in the same block, then plot its results.
FML Variables	FML variables are defined within FML components and are available for use in expressions in the local design, also similar to local variables. These are evaluated for each iteration at a point in the sequence of evaluation of your choice, and can use simulation-dependent variables in their expressions. Analysis control variables can be set within an FML component and will override those values in the solution setup. As an example, setting Hmin and Hmax based on a specific time value or circuit conditions may be very useful. Care must be taken to avoid unintended consequences if these values are changed. Note that setting Tend in this way may cause problems with spectral plots, continuing a simulation based on **Enable continue to solve** in the setup, or continuing a simulation with a state (KRN) file. **Note**: The order of variable declaration is important, and this order is under user control in FML components. A variable that is used before being defined has a value of **0** at that use. **Note**: FML and FML_INIT variables override the local variables in design properties if they use the same name. However, this is not recommended and a warning will be issued during netlisting. Reports always give a precedence to a design variable of the same name while showing the simulation results. In such cases, if you want to see results for the FML variables in plots, you can optionally assign it to a new non-conflicting FML variable in the same block, then plot its results.

If you define a variable as an expression that evaluates to a constant, whether a project, local or parameter variable, the expression is retained in the variable list, rather than being evaluated and replaced with a constant. You can then identify and modify the expression in the future.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

Whether you are defining project, local, or parameter variables, intrinsic names (such as freq and lb) are reserved and cannot be used or entered into the source and port dialogs. See ***Reserved Variable Names*** for details.

Defining a Project Variable

A project variable can be assigned to a parameter value in the Twin Builder project in which it was created. Twin Builder differentiates project variables from other types of variables by prefixing the variable name with the **$** symbol. You can manually include the **$** symbol in the project variable's name when you create it, or Twin Builder will append the project variable's name with the symbol after you define the variable.

1. On the **Project** menu, click **Project Variables**.
   - You can also right-click the project name in the Project tree and click **Project Variables**. You can also access the project variables from a menu in the lower-left corner of the following **Optimization** dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Project Variables**.

   The **Properties** dialog box appears, containing three tabs: **Project Variables**, **Intrinsic Variables**, and **Constants**.

Note:

The **Intrinsic Variables** and **Constants** tabs provide non-editable reference listings of Twin Builder's predefined internal variables and constants.

2. Under the **Project Variables** tab, click **Add**. The **Add Property** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. In the **Name** text box, type the name of the variable.

   Note:

   - Project variable names must start with the **$** symbol followed by a letter. Variable names may include only alphanumeric characters and underscores (_).
   - Names of reserved variables, reserved (intrinsic) functions and pre-defined constants cannot be used as variable names.
   - Twin Builder prepends the **$** symbol to each project variable name you specify. The **$** symbol becomes an essential part of the project variable name and must be included whenever you specify an existing project variable as a parameter value.
   - In netlists, the **$** prefix is changed to **Pjt_**. For example, project variable **$RNom** appears as **Pjt_RNom** in the netlist.

4. In the **Unit Type** text box, use the drop-down list to select from the available unit types. **None** is the default.

   When you select a unit type, the choices in the **Units** drop-down list adapt to that unit type. For example, selecting **Length** as the unit type causes the **Unit** menu to show a range of metric and English units for length. Similarly, if you select the **Resistance** unit type, the **Units** drop-down list shows a range of standard Ohm units.

5. In the **Value** text box, type the quantity that the variable represents. Optionally, include the units of measurement.

   Warning:

   If you include the variable's units in its definition (in the **Value** text box), do not include the variable's units when you enter the variable name for a parameter value.

   The quantity can be a numerical value, a mathematical expression, or a mathematical function. The quantity entered is the *current,* (or *default) value* for the variable. If the mathematical expression includes a reference to an existing variable, the system treats this variable as a dependent variable. The units for a dependent variable change to those of the independent variable on which the value depends. Additionally, dependent variables, though useful in many situations, cannot be the direct subject of optimization/Design of Experiments, sensitivity analysis, tuning, or statistical analysis.

6. Click **OK** to return to the **Properties** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The new variable and its value appear in the table. If the value is an expression, the evaluated value displays. Updating the expression also changes the evaluated value display.

7. Optionally, type a description of the variable in the **Description** text box.
8. Optionally, select **Read Only**. The variable's name, value, unit, and description cannot be modified when **Read Only** is selected.
9. Optionally, select **Hidden**. If you clear the **Show Hidden** option, the hidden variable will not appear in the **Properties** dialog box.

   The new variable can now be assigned to a parameter value in the project in which it was created.

Related Topics

Viewing and Editing Project Variables

Deleting Project Variables

Add/Edit Property Dialog Boxes

Viewing and Editing Project Variables

Follow this procedure to edit a project variable:

1. Select **Project** > **Project Variables**, or right-click the project name in the Project tree and select **Project Variables**. The **Properties** dialog box appears.
2. On the **Project Variables** tab, you can edit the variable name, value, unit, and description. Click the appropriate cell and make your changes.

   > **Note:**
   > - Clear the **Read Only** check box before attempting to edit.
   > - Select the **Show Hidden** check box to display all defined variables.

3. When finished editing, click **OK** to save your changes and close the **Properties** dialog box.

Deleting Project Variables

Follow this procedure to delete a project variable:

1. Remove all references to the variable in the project.
2. Save the project to erase the command history.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Select **Project** > **Project Variables** to display the **Properties** dialog box with a list of variables. You can also access the project variables from a menu in the lower-left corner of the following **Optimization** dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Project Variables**.
4. Select the variable and click **Remove Selected** > **OK**. The variable can be removed if it is not in use.

To remove all unused project variables that are not in use (that is, not in the undo/redo command history):

1. Remove all references to unused project variables, including dependent variables.
2. **Save** the project to erase the command history.
3. Select **Project** > **Project Variables** to display the **Properties** dialog box with a list of variables. You can also access the project variables from a menu in the lower-left corner of the following **Optimization** dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Project Variables**.
4. Click **Remove** and from the drop-down list, select **Remove All Unused** and click **OK**. All variables that are not in use (not in undo/redo history) are removed.

To force remove all unused design variables:

1. Remove all references to unused project variables, including dependent variables.
2. Click **Project** > **Project Variables** to display the **Properties** dialog box with a list of variables. You can also access the project variables from a menu in the lower-left corner of the following **Optimization** dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Project Variables**.
3. Click **Remove** and from the drop-down list, select **Force Remove All Unused** and click **OK**. A warning message asks whether you want the unapplied changes in the property dialog to be applied, and clear undo/redo history. If you select **No**, nothing happens. If you select **Yes**, all unapplied changes are applied first, and undo/redo history is cleared; all variables that are not used are removed (including those that are only in undo/redo history before this command is executed).

To remove all unused project variables that are not in use (that is, not in the undo/redo command history):

1. Remove all references to unused project variables, including dependent variables.
2. **Save** the project to erase the command history.
3. Click **Project** > **Project Variables** to display the **Properties** dialog box with the list of variables.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Click **Remove** and from the drop-down list, select **Remove All Unused** and click **OK**. All variables that are not in use (not in undo/redo history) are removed.

To force remove all unused design variables:

1. Remove all references to unused project variables, including dependent variables.
2. Click **Project** > **Project Variables** to display the **Properties** dialog box with the list of variables.
3. Click **Remove** and from the drop-down list, select **Force Remove All Unused** and click **OK**. A warning message asks whether you want the unapplied changes in the **Property** dialog box to be applied, and clear undo/redo history. If you select **No**, nothing happens. If you select **Yes**, all unapplied changes are applied first, and undo/redo history is cleared. Then all variables that are not used are removed, including those that are only in undo/redo history before this command is executed.

Defining Local (Design) Variables

You can define a local variable in two ways:

- At the design level.
- From within a component.

Related Topics

Deleting Local Variables

Adding an Array of Values for a Local Variable

Adding a Local Variable at the Design Level

A design variable is associated with a Twin Builder design. You can assign a design variable to a parameter value in the Twin Builder design in which it was created.

1. On the **Twin Builder** menu, click **Design Properties**.
   - You can also right-click the design name in the Project tree and click **Design Properties**. The **Properties** dialog box appears.
2. Under the **Local Variables** tab, click **Add**. The **Add Property** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. In the **Name** text box, type the name of the variable.

   > **Note:**
   >
   > - Variable names must start with a letter, and may include alphanumeric characters and underscores (_).
   > - Names of reserved variables, reservoir (intrinsic) functions and pre-defined constants cannot be used as variable names.

4. In the **Unit Type** text box, use the drop-down list to select from the list of available unit types. **None** is the default.

   When you select a unit type, the choices in the **Units** drop-down list adapt to that unit type. For example, selecting **Length** as the unit type causes the **Unit** menu to show a range of metric and English units for length. Similarly, if you select the **Resistance** unit type, the **Units** drop-down list displays a range of standard Ohm units.

5. In the **Value** text box, type the quantity that the variable represents. Optionally, include the units of measurement.

   > **Note:**
   >
   > If you include the variable's units in its definition (in the **Value** text box), do not include the variable's units when you enter the variable name for a parameter value.

   The quantity can be a numerical value, a mathematical expression, or a mathematical function. The quantity entered will be the *current* (or *default value)* for the variable. If the mathematical expression includes a reference to an existing variable, this variable is treated as a dependent variable. The units for a dependent variable will change to those of the independent variable on which the value depends. Additionally, dependent variables, though useful in many situations, cannot be the direct subject of optimization, sensitivity analysis, tuning, or statistical analysis.

   > **Note:**
   >
   > Complex numbers are not allowed for variables to be used in an Optimetrics sweep, or for optimization, statistical, sensitivity, or tuning setups.

6. Click **OK** to return to the **Properties** dialog box.

   The new variable and its value appear in the table. If the value is an expression, the evaluated value displays. Updating the expression also changes the evaluated value display.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7. Optionally, type a description of the variable in the **Description** text box.

   The new variable can now be assigned to a parameter value in the design in which it was created.

> **Note:**
>
> Local Variables are *not* usable in the design's subcircuits and *are not* in the design's public interface (that is, they are not visible on the component representing the design when it is used as a subdesign).

Related Topics

Adding an Array of Values for a Local Variable

Adding a Local Variable from a Component

To define a local variable from a component:

1. Display the properties of the component that will use the local variable to define. To do this, do any one of the following:
   - Right-click the component and select **Properties**. The **Properties** dialog box opens.

     Click the **Parameter Values** or the **Quantities** tab.
   - Select the component and view its properties in the **Param Values** or **Quantities** tab of the desktop's **Properties** window.
     a. Select the component.
     b. Click **Edit** > **Properties**. The **Properties** dialog box opens.
     c. Click the **Parameter Values** tab.
2. Click the **Value** field for the parameter you want to set equal to the local variable.
3. Type the variable name. The name may not duplicate a reserved (intrinsic) variable name. See Reserved Variable Names for details.
4. Click **OK**. The **Add New Variable** dialog box appears.
5. Type a valid numerical quantity in the **Value** box.
6. Click **LocalVariable**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7. Click **OK**.

> **Note:**
>
> Local variables are *not* usable in the design's subcircuits and *are not* in the design's public interface (that is, they are not visible on the component representing the design when it is used as a subdesign).

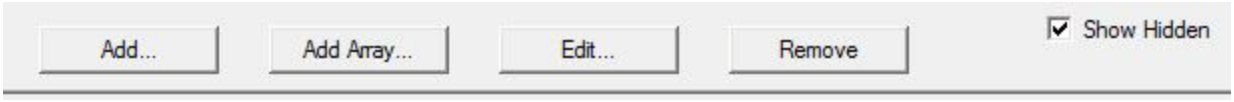
8. Click **Add** to add another local variable, or click **OK** to stop adding local variables.

Adding an Array of Values for a Local Variable

A local variable is associated with a Twin Builder design. You can also add a variable defined with an array of values.

1. Select **Twin Builder** > **Design Properties**.
   - You can also right-click the design name in the Project tree and select **Design Properties**.

   From the **Local Variables** tab in the **Properties** dialog box, you can **Add**, **Add Array**, **Edit**, or **Remove** variables. This section describes **Add Array**.



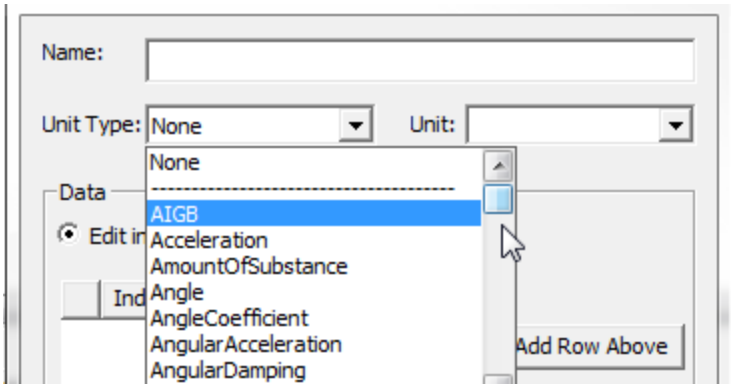
   Any existing local variables are listed in the **Properties** dialog box with the name followed by cells for **Value**, **Unit**, **Evaluated Value**, **Type**, **Description**, and **Read-only** and **Hidden** check boxes. The **Show Hidden** check box controls the display of any hidden variables.

   Initially, leave the radio button with **Value** selected until you define a variable. Use the other radio buttons to enable regular variables for **Optimization/Design of Experiments**, **Tuning**, **Sensitivity**, or **Statistics**. Array variables cannot be enabled for Optimetrics use.

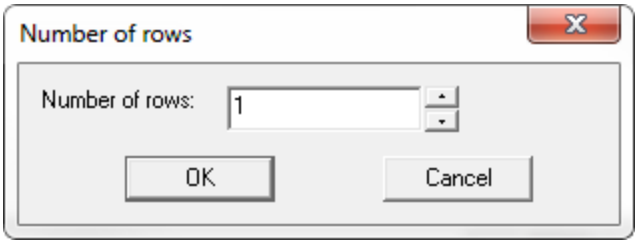
2. Click **Add Array**. The **Add Array** dialog box appears.
3. Specify a variable **Name** in the text field.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Select a **Unit Type** and **Units** from the drop-down lists.



5. To specify the array with **Edit in grid** selected, click **Append Rows** to display the **Number of Rows** dialog box.



6. Specify a value and click **OK** to display a list of indexed data rows in the **Add Array** dialog box. You can type any data value in the cells. If you enter alphanumeric text in a cell, it must be delimited by double quotes. Click **Add Row Above**, **Add Row Below**, or **Remove Row** to edit the rows relative to a row selection. All cells must contain a value.
7. When you have completed the array, click **OK** to close the dialog box.

   The array variable is listed in the **Design Properties** dialog box as a local variable. The array variable value field includes the array contents in brackets with the unindexed data values delimited by commas.

   If you elected to edit the array **Edit in plain text field** in the **Add Array** dialog box, the bracketed and comma delimited format is used.

Deleting Local Variables

To delete a local variable:

1. Remove all references to the variable in the design.
2. Save the project to erase the command history.
3. Select **Twin Builder** > **Design Properties** to display the **Properties** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Click the **Local Variables** tab.
5. Select the desired variable, click **Remove Selected**, and click **OK**.

To remove all unused local variables (that is, not in the undo/redo command history):

1. Remove all references to unused variables in the design, including dependent variables.
2. **Save** the project to erase the command history.
3. Select **Twin Builder** > **Design Properties** to display the **Properties** dialog box.
4. Click the **Local Variables** tab.
5. Select the variable and click **Remove** and from the drop-down list, select **Remove All Unused**, and click **OK**. All unused variables (not in undo/redo history) are removed.

To force remove all unused local variables:

1. Remove all references to unused variables in the design, including dependent variables.
2. Click **Twin Builder** > **Design Properties** to display the **Properties** dialog box.
3. Click the **Local Variables** tab.
4. Select the variable and click **Remove** and from the drop-down list, select **Force Remove All Unused** and click **OK**. A warning message asks whether you want the unapplied changes to be applied, and clear undo/redo history. If you select **No**, nothing happens. If you select **Yes**, all unapplied changes are applied first, and undo/redo history is cleared; all unused variables are removed (including those that are only in undo/redo history before this command is executed).

Defining an Expression

Expressions are mathematical descriptions that typically contain intrinsic functions, such as sin (x), and arithmetic operators, such as +, -, *, and /, as well as defined variables. Expressions may also contain pre-defined constants, but such constants may not be reassigned a new value. Defining one variable in terms of another makes a dependent variable. Dependent variables, though useful in many situations, cannot be the subject of optimization, sensitivity analysis, tuning, or statistical analysis.

Numerical values may be entered in Ansys Electromagnetics' shorthand for scientific notation. For example, $5x10^7$ could be entered as **5e7**.

Operator Precedence

Operators used to define an expression or function have a pre-defined sequence in which they are performed. The following list shows both the valid operators and the sequence in which they are executed, listed in decreasing precedence. Operators of equal precedence are evaluated left-to-right, except for exponentiation, which is evaluated right-to-left.

( )	parenthesis	1

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

!	logical NOT (negation)	2
^ (or **)	exponentiation (If you use “**” for exponentiation, as in previous software versions, it is changed to “^”.)	3
-	unary minus	4
*****	multiplication	5
/	division	5
+	addition	6
-	subtraction	6
==	equal to (compares two values for equality)	7
!=	not equal	7
>	greater than	7
<	less than	7
>=	greater than or equal to	7
<=	less than or equal to	7
&&	logic AND (conjunction)	8
||	logic OR (disjunction)	8

Range Functions

Range Functions are special functions that use a 2D dataset as input, along with zero or more additional parameters. Range functions produce a user-created 2D report display that is a collection of traces and their attributes. The trace collection and attributes generate a portion of a report definition that generates a family of curves in the report window.

For more information, as well as a table which lists available range functions, see Defining Traces Using Range Functions.

Using Piecewise Linear Functions in Expressions

The following piecewise linear intrinsic functions are accepted in expressions:

```
pwl (dataset_expression, variable)
```

```
pwl(arrayVariable[indexVariable], variable)
```

```
pwl_periodic (dataset_expression, variable)
```

The **pwl** function interpolates along the X-axis and returns a corresponding y value. The **pwl_periodic** function also interpolates along the X-axis but periodically.

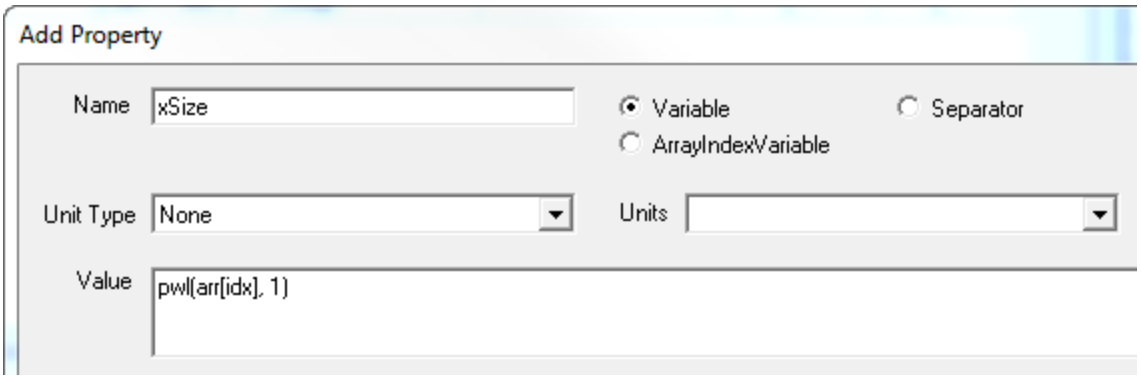
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

You can use **pwl** in an expression that uses array variables and datasets for uses such as a frequency dependent material property. For example, you specify BulkConductivity as:

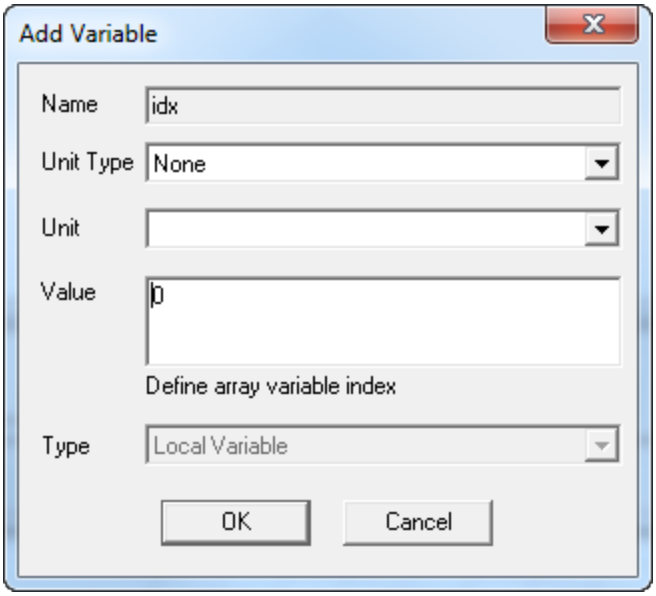
`pwlx($dsArr[$dsIndex], Freq)` where `$dsArr=["$ds1", "$ds2"]`

You can create a design variable representing dimension `xSize` as `pwl(arr[idx], 1)` where "`arr`" is an array variable and `idx` is an array index variable.

In this case, creating a variable named xSize with pwl(arr[idx], 1) like this:

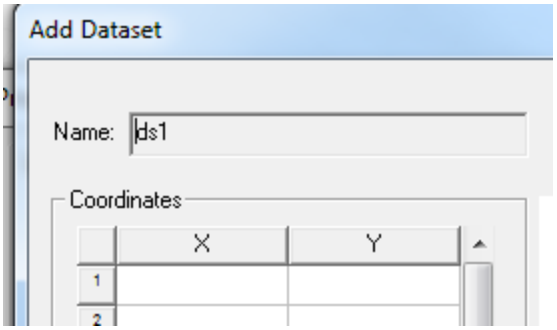


This value for xSize opens a dialog box first for the idx index variable:

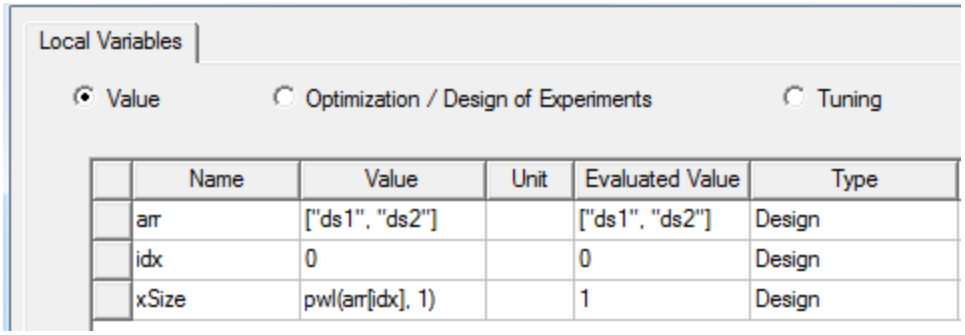


Dialog boxes then open for each dataset variable implicit for the predefined array variable:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The **Design Properties** dialog box then appears as follows:



Related Topics

Adding Datasets

Editing Datasets

Using Standard Mathematical Functions in Expressions

Twin Builder recognizes a set of standard mathematical functions that can be used to define expressions. These function names are reserved, and cannot be used as variable names.

> **Note:**
>
> If you do not specify units, all trigonometric functions interpret their arguments in radians. Likewise, inverse trigonometric functions' return values are in given in radians. When the argument to a trigonometric expression is a variable, the units are assumed to be radians. If you want values interpreted in degrees, supply the argument with the unit name **deg**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Constants Tab (Project Variables Dialog Box)

The **Constants** tab in the **Project Variables** dialog box lists the available pre-defined constants. These may not be reassigned a new value. See ***Predefined Constants*** for additional information.

Reserved Variable Names

> **Note:**
>
> When you define a new project variable or local variable, the name you assign must not be the same as one of the reserved variable names listed on the **Intrinsic Variables** tab.

To view the list of reserved variable names, open the **Project** drop-down list—or right-click a project—select **Project Variables**, and click the **Intrinsic Variables** tab:

Intrinsic Variable Name	Description
_Empty	Empty value, taken to be model default by simulator (Twin Builder).
_I1 to _I9	Terminal current in interpretive user-defined model (A).
_t	Variable to define parametric equation based curve.
_u	Variable to define parametric equation based surface.
_v	Variable to define parametric equation based surface.
_V1 to _V9	Port voltage in interpretive user-defined model (V).
Ang	Post-processing variables; not settable by user.
Budget_Index	Post-processing variables; not settable by user.
Distance	...
ElectricalDegree	Electric Degree of the rotating machine. Not settable by the user.
F1	Frequency of tone 1 in harmonic balance analysis (Hz).
F2	Frequency of tone 2 in harmonic balance analysis (Hz).
F3	Frequency of tone 3 in harmonic balance analysis (Hz).
Fend	...
FNoi	Offset noise frequency in harmonic balance noise analysis (Hz).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Intrinsic Variable Name	Description
Freq	Frequency of circuit/system analysis (Hz).
Fstart	...
Fstep	...
H	...
Hmax	...
Hmin	...
Ia	Post-processing variables; not settable by user.
Ib	Post-processing variables; not settable by user.
Index	Post-processing variables; not settable by user.
IWavePhi	Post-processing variables; not settable by user.
IWaveTheta	Post-processing variables; not settable by user.
NormalizedDeformation	Post-processing variables; not settable by user.
NormalizedDistance	Post-processing variables; not settable by user.
OP	Post-processing variables; not settable by user.
Pass	Post-processing variables; not settable by user.
Phase	Post-processing variables; not settable by user.
Phi	Post-processing variables; not settable by user.
R	Post-processing variables; not settable by user.
Rho	Post-processing variables; not settable by user.
RSpeed	Speed of the machine. Not settable by the user.
Spectrum	Post-processing variables; not settable by user.
Temp	Analysis temperature (deg).
Tend	...
Theta	Post-processing variables; not settable by user.
Time	Time point in transient analysis.
Vac	Post-processing variables; not settable by user.
Vbe	Post-processing variables; not settable by user.
Vce	Post-processing variables; not settable by user.
Vds	Post-processing variables; not settable by user.
Vgs	Post-processing variables; not settable by user.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Intrinsic Variable Name	Description
X	Post-processing variables; not settable by user.
Y	Post-processing variables; not settable by user.
Z	Post-processing variables; not settable by user.
ZAng	Post-processing variables; not settable by user.
ZRho	Post-processing variables; not settable by user.

Defining Parameter Defaults

You can define a new parameter default at the design level.

The current local values of a design's parameter defaults appear in the design's **Passed Parameters** list.

Defining a New Parameter Default by Starting at the Design Level

To define a new parameter default by starting at the design level:

1. In the Project tree, right-click the design for which you want to define a parameter default and select **Design Properties**.
2. Select the **Parameter Defaults** tab.
3. Select **Value**.
4. Click **Add**. The **Add Property** dialog box appears.
5. Select **Variable**.
6. In the **Name** box, type a variable name.
7. In the **Value** box, type a valid numerical quantity.
8. Click **OK**.
9. Click **Add** to add another parameter default, or click **OK** to stop adding parameter defaults.

Assigning Variables

To assign a variable to a parameter in Twin Builder:

- Type the variable name or mathematical expression in place of a parameter value in a **Value** text box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If you typed a variable name (without the **$** prefix) that has not been defined, the **Add Variable** dialog box appears, enabling you to define the variable either as a **Local Variable** or as a **Parameter Default**.

If you typed a variable name that included the **$** prefix, but that has not been defined, the **Add Variable** dialog box appears, enabling you to define the project variable.

> **Note:**
>
> You can assign a variable to nearly any design parameter assigned a numeric value in Twin Builder. Search the help for the specific parameter you want to vary to determine if can be assigned a variable.

Choosing a Variable to Optimize

Before optimizing a variable, you must specify that you intend to use it during an optimization analysis in the **Properties** dialog box.

1. To choose a variable to optimize:
   a. If the variable is a design variable, click **Twin Builder** > **Design Properties**. You can also access the design variables from a menu in the lower left corner of the following Optimization dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Design Variables**.
   b. If the variable is a project variable, click **Project** > **Project Variables**. You can also access the project variables from a menu in the lower left corner of the following Optimization dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Project Variables**.

      The **Properties** dialog box appears.
2. Click the tab that lists the variable you want to optimize.
3. Click the row containing the variable you want to optimize.

   > **Note:**
   >
   > Dependent variables cannot be optimized.

4. Select **Optimization/Design of Experiments**.
5. For the variable you want to optimize, select **Include**.

   The selected variable will now be available for optimization in an Optimetrics setup defined in the current design or project.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Complex numbers are not allowed for variables to be used in an Optimetrics sweep, or for optimization, statistical, sensitivity or tuning setups.

6. Optionally, override the default minimum and maximum values that Optimetrics will use for the variable in every optimization analysis. During optimization, the optimizer will not consider variable values that lie outside of this range.

Related Topics

Setting up an Optimization Analysis

Including a Variable in a Sensitivity Analysis

Before including a variable in a sensitivity analysis, you must specify that you intend to use it during a sensitivity analysis in the **Properties** dialog box.

1. To include a variable in a sensitivity analysis:
   a. If the variable is a design variable, click **Twin Builder** > **Design Properties**. You can also access the project variables from a menu in the lower left corner of the following Optimization dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Design Variables**.
   b. If the variable is a project variable, click **Project** > **Project Variables**. You can also access the project variables from a menu in the lower left corner of the following Optimization dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Project Variables**.

   The **Properties** dialog box appears.
2. Click the tab that lists the variable you want to include in the sensitivity analysis.
3. Click the row containing the variable you want to include in the sensitivity analysis.

> **Note:**
>
> Dependent variables cannot be included in a sensitivity analysis.

4. Select the **Sensitivity** option above.
5. For the variable you want to include in the sensitivity analysis, select **Include**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The selected variable will now be available for sensitivity analysis in a sensitivity setup defined in the current design or project.

> **Note:**
>
> Complex numbers are not allowed for variables to be used in an Optimetrics sweep, or for optimization, statistical, sensitivity or tuning setups.

6. Optionally, override the default minimum and maximum values that Optimetrics will use for the variable in every sensitivity analysis. During sensitivity analysis, Optimetrics will not consider variable values that lie outside of this range.
7. Optionally, override the default initial displacement value that Optimetrics will use for the variable in every sensitivity analysis. During sensitivity analysis, Optimetrics will not consider a variable value for the first design variation that is greater than this step size away from the starting variable value.

Related Topics

Setting up a Sensitivity Analysis

Choosing a Variable to Tune

Before tuning a variable, you must specify that you intend to tune it in the **Properties** dialog box.

1. To choose a variable to tune:
   a. If the variable is a design variable, click **Twin Builder** > **Design Properties**. You can also access the project variables from a menu in the lower left corner of the following Optimization dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Design Variables**.
   b. If the variable is a project variable, click **Project** > **Project Variables**. You can also access the project variables from a menu in the lower left corner of the following Optimization dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**. and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Project Variables**.

   The **Properties** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Click the tab that lists the variable you want to tune.
3. Click the row containing the variable you want to tune.

> **Note:**
>
> Dependent variables cannot be tuned.

4. Select the **Tuning** option above.
5. For the variable you want to tune, select **Include**.

> **Note:**
>
> Complex numbers are not allowed for variables to be used in an Optimetrics sweep, or for optimization, statistical, sensitivity or tuning setups.

The selected variable will now be available for tuning in the **Tune** dialog box.

Related Topics

Tuning a Variable

Including a Variable in a Statistical Analysis

Before including a variable in a statistical analysis, you must specify that you intend to use it during a statistical analysis in the **Properties** dialog box.

1. To include a variable in a statistical analysis:
   a. If the variable is a design variable, click **Twin Builder** > **Design Properties**. You can also access the project variables from a menu in the lower left corner of the following Optimization dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Design Variables**.
   b. If the variable is a project variable, click **Project** > **Project Variables**. You can also access the project variables from a menu in the lower left corner of the following Optimization dialog boxes: **Parametric**, **Optimization**, **Sensitivity**, **Statistical**, **Design of Experiments**, and **DesignXplorer Setup**. Click **Edit Variables** > **Edit Project Variables**.

   The **Properties** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Click the tab that lists the variable you want to include in the statistical analysis.
3. Click the row containing the variable you want to include in the statistical analysis.

> **Note:**
>
> Dependent variables cannot be included in a statistical analysis.

4. Select the **Statistical** option above.
5. For the variable you want to include in the statistical analysis, select **Include**.

   The selected variable will now be available for statistical analysis in a statistical setup defined in the current design or project.

> **Note:**
>
> Complex numbers are not allowed for variables to be used in an Optimetrics sweep, or for optimization, statistical, sensitivity or tuning setups.

6. Optionally, override the distribution criteria that Optimetrics will use for the variable in every statistical analysis.

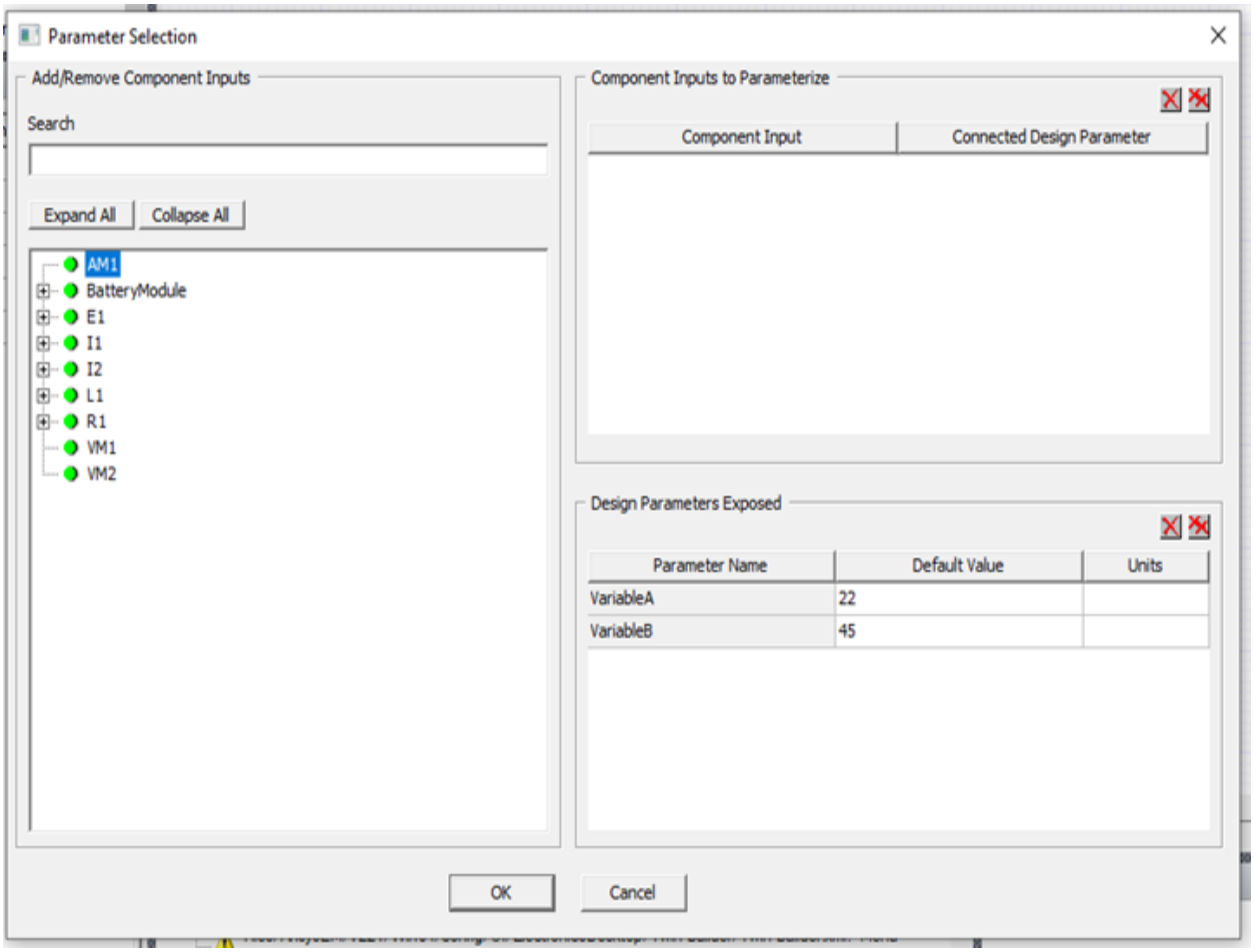
Related Topics

Setting up a Statistical Analysis

Assigning Parameters to Component Quantities

Use the **Parameter Selection** dialog box to assign design parameters to component input quantities. You can only parameterize input quantities that do not show a pin. Select **Twin Builder** > **Parameter Selection Dialog**, or in the **Project Manager** pane, right-click a design or subsheet and select **Parameter Selection Dialog**. The **Parameter Selection** dialog box appears as pictured below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The **Parameter Selection** dialog box has three panels.

- **Add/Remove Component Inputs** – Select the component quantity to parameterize. Select a component quantity from the tree to add it to the **Component Inputs to Parameterize** panel. Use the **Search** box to search for quantities in the tree.
- **Component Inputs to Parameterize** – Assign a parameter to a component quantity. The **Component Input** column contains the quantity name and the **Connected Design Parameter** column contains the parameter name. When adding a quantity to the grid for the first time, a new parameter is created and added to the **Design Parameter Exposed** panel. Edit the text in the **Connected Design Parameter** column to rename a newly-created parameter. Edit the text in the **Connected Design Parameter** column to assign an existing parameter to a quantity. To delete a quantity assignment, select its row and click . Click to delete all assignments.
- **Design Parameters Exposed** – Assign values to existing and new parameters. In this panel, the **Parameter Name** column displays the parameter name, the **Default Value**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

column displays the parameter value, and the **Units** column contains the value units. Edit the text in the **Default Value** column to change the parameter value; use the drop-down list in the **Units** column to change the units. To delete a parameter, select its row and click . Click to delete all parameters.

Click **OK** to implement your changes and close the dialog box; click **Cancel** to close the dialog box without making any changes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

11 - Schematic Editor

Use the schematic editor to create and edit *designs*. A design graphically represents and captures the electrical, mechanical, and functional characteristics of a circuit or multi-domain system. A design may be *hierarchical*: it may contain symbols that represent other designs. A design may consist of one or more *pages*: multiple, associated graphical workspaces that share a local namespace at the same hierarchical level. You create a *design* by starting the schematic editor and placing components, ports, connectors, and wires (connections) into a default empty schematic.

Setting Schematic Editor Options

Set these options under **Schematic Editor** in the **Options** dialog box.

To set **Schematic Editor** options:

1. Click **Tools** > **Options** > **General Options**.
2. Click **Schematic Editor**.
3. Make your changes on the appropriate panels:
   - General
   - Fonts
   - Colors
   - Wiring
   - Multiple Placement
   - Symbol Editor
   - Twin Builder Schematic
4. Click **OK**.

Schematic Editor Options: General Tab

Set these options in the **General** tab under **Schematic Editor** in the **Options** dialog box.

- **Symbol Graphics** – Specify whether to use IEEE or traditional graphic symbol style on schematics. Each style contains two or more active levels for the graphic objects that comprise a symbol. See Symbol Graphic Object Levels for additional information.
- **SubCircuit Pin Spacing** – Specify the pin spacing in grid units for pins on a subcircuit component symbol. See Adding a Subcircuit and Creating a Subcircuit from a Selection Area for additional information.
- **Show Pin Labels** – Specify whether to display pin labels in the **Schematic Editor**.
- **Property Display Angle Follows Symbol Angle** – Specify whether displayed text properties rotate when the associated component rotates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Update Property Display on Definition Update** – Specify that when component symbol definitions update, the displayed properties of instances of the updated component on the schematic also update. If not enabled, only the symbol graphics are updated.
- **Auto Scroll when close to edges** – Specify whether the **Schematic Editor** display scrolls when the cursor is near the edge of the editor window.
- **Show advanced property data** - Toggle the display of less often used component property tabs such as **Symbol** and **General**.
- **Net name display** – Set the net name property display distance from the net.
- **Symbol Scaling Factor** – Third-party vendors often follow a different symbol dimensioning system than that used in Twin Builder. Symbol sizes that look appropriate in a third-party application may not be appropriate for Twin Builder. To mitigate such issues, use symbol scaling during import to scale the incoming symbol graphics for symbol formats such as SVG by the specified amount. Similarly, when exporting, the symbol can be rescaled to the original dimensions.
- **Selection Colors** – sets the colors of the first and subsequent objects as they are selected on a schematic.

Schematic Editor Options: Twin Builder Schematic Tab

Set these options in the **Twin Builder Schematic** panel under **Schematic Editor** in the **Options** dialog box.

Select the desired quantities and signal types that will be available for selection by Probes:

- Inputs
- Outputs
- Inouts

Schematic Editor Options: Wiring Tab

Set these options in the **Wiring** panel under **Schematic Editor** in the **Options** dialog box.

Connectivity Options:

- **Show Merge Wire dialog before combining wires** – Display the **Merge Wire** dialog box before combining wires to allow for changes to the merge configuration.
- **Show Split Wire dialog before separating wires** – Display the **Split Wire** dialog box before separating wires to allow for changes to the split configuration.
- **When renaming a physically separate piece of wire** – Choose from:
  - **Split into a different net**.
  - **Keep connected (name all pieces the same)**.
- **Show GlobalPort Disconnect dialog before separating** – Display the **GlobalPort Disconnect** dialog box before separating wires to allow for changes to the global port

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

configuration.

Schematic Editor Options: Multiple Placement Tab

Use multiple placement to place several objects of the same type in the schematic. Set these options in the **Multiple Placement** panel under **Schematic Editor** in the **Options** dialog box.

Select the desired objects in the **Items for multiple placement** list.

- **Components**
- **Interface Ports**
- **Grounds**
- **Page Connectors**
- **Global Ports**

Schematic Editor Options: Fonts Tab

Set these options in the **Fonts** panel under **Schematic Editor** in the **Options** dialog box.

Set the **Font Name** and **Size** for text used on schematics on the **Fonts** tab of the **Schematic Editor Options** dialog box. A **Sample Text** display window shows the appearance of the specified font and size. **Apply this font to all property displays in the active schematic** specifies that all displays in the schematic editor are to use the font style and size you selected.

Schematic Editor Options: Colors Tab

Set these options in the **Color** panel under **Schematic Editor** in the **Options** dialog box.

Use the **Schematic Objects** panel to set the color for each **Object Type**.

- **Object Type** – Select the object type whose colors you want to modify.
- **Set Color** – Implement the color you selected for the object type.
- **Clear Color** – Reset the color of the selected object type to its default value.

Use the **Individual Definitions** panel to set the color for specified **Components** and **Wire Domains**.

- **Component Name** – Identify the component whose colors you want to modify. Visible only when you select **Components Object Type**.
- **Add** – Add a component whose colors you want to define. Visible only when you select **Components Object Type**.
- **Wire Domain** – Select the wire domain whose color you want to modify. Each wire domain is assigned a unique color by default.
- **Set Color** – Implement the color you selected for the specified component. All future instances of the component added to the active schematic window display will possess the selected color.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Clear Color** – Reset the color of the component to its default value.
- **Apply these color settings to the current schematic** – Apply the color settings for the selected **Object Type** to objects of that type in the active schematic window.

Schematic Editor Options: Symbol Editor Tab

Set these options in the **Symbol Editor** panel under **Schematic Editor** in the **Options** dialog box.

- **Adjust Pin Orientation on Drag** – Enable symbol pin orientation to be adjusted when dragging pins to the desired side of a symbol in the symbol editor.

Starting the Schematic Editor

To start the schematic editor and begin editing a circuit or system design, do one of the following:

- In the **Project Manager** Project tree, double-click for an existing design.
- In the **Project Manager** Project tree, right-click a project icon and select **Insert** > **Insert Twin Builder Design**.
- In the **Project Manager** Project tree, select the project that will contain the new design, then select **Project** > **Insert Twin Builder Design**, or click in the ribbon.

Related Topics

Working with Twin Builder Projects

The Schematic Editor Window

Use the **Schematic Editor** window to place components and wire them together.

- Click and drag components to move them.
- **Copy** and **Paste** components and their wires within the schematic editor.
- Copy and paste to other schematics.

As you place the cursor near a pin of a component, it changes from an arrow to a large **X**, indicating the schematic editor is in wiring mode. In wiring mode, click to start drawing a wire, and click again to end the wire.

Place commonly used items such as ports, grounds, and page connectors in the schematic; either click their toolbar icons or use the **Draw** menu.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Controls to **Zoom In**, **Zoom Out**, and **Fit Drawing** to the editor window are available on the **View** menu, the toolbar, and on the shortcut menu that opens when you right-click a schematic. Shortcut key combinations (hotkeys) for these controls are also available. For example, you can use **Ctrl** while clicking and dragging with the middle mouse button to zoom in and out (assuming you have *not* selected *Use Legacy View Navigation* in the General > User Interface options. Alternatively, if you chose to select *Enable legacy view navigation*, use the combination **Alt** + **Shift** + **drag** (with the left mouse button) to zoom in or out.

Zoom in and out of the schematic with the mouse wheel. Use the **Zoom with mouse wheel** check box in the **Options** dialog box to customize mouse wheel behavior. Select **Tools** > **Options** > **General Options** to open the **Options** dialog box. Select **Schematic Editor** > **General**.

If you select the **Zoom with mouse wheel** check box:

- Press **Shift** while rolling the mouse wheel to scroll vertically.
- Press **Ctrl** while rolling the mouse wheel to scroll horizontally.

If you clear the **Zoom with mouse wheel** check box:

- Roll the mouse wheel up and down to scroll vertically.
- Press **Shift** while rolling the mouse wheel to zoom in and out.

The arrow keys scroll the view up, down, left, or right in small increments. The **Page Up** and **Page Down** keys scroll the view up or down in larger increments. Select **View** > **Fit Drawing** (or Ctrl+D) to re-center the entire design, resized to fill the window.

Schematic Editor Shortcut Keys

Ctrl + **Z**	Undo
Ctrl + **Y**	Redo
Ctrl + **X**	Cut
Ctrl + **C**	Copy
Ctrl + **V**	Paste
Ctrl + **A**	Select all
Ctrl + **"+"**	Zoom in (if using the "=\+" key, instead of the "+" key on the numeric keypad, do **not** press *Shift*)
Ctrl + **"-"**	Zoom out
Ctrl + **Q**	Zoom area
Ctrl + **D**	Fit drawing
Ctrl + **R**	Rotate
Ctrl + **W**	Add wire

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ctrl + **K**	Cross-probe for selected components in layout editor
Ctrl + **B**	Fit border
Ctrl + **G**	Place ground
Ctrl + **N**	New project
Ctrl + **O**	Open project
Ctrl + **S**	Save project
Ctrl + **T**	Add text
Ctrl + **Shift** + **drag** (on selection)	Move the selection without maintaining connections. Can be used to remove a component from its connections, and put another in its place.
Ctrl + **MMB** + **drag**	Pan (MMB = **middle mouse button** click)
Shift + **MMB** + **drag**	Zoom (MMB = **middle mouse button** click)
Ctrl + **Shift** + **click**	Select an entire net when a segment is clicked.
F3	Find elements
Roll mouse wheel	Zoom in and out if **Zoom with mouse wheel** check box is selected. Scroll vertically if **Zoom with mouse wheel** check box is cleared. See ***The Schematic Editor Window*** for more information on setting mouse wheel behavior.
Shift + **roll mouse wheel**	Scroll vertically if **Zoom with mouse wheel** check box is selected. Zoom in and out if **Zoom with mouse wheel** check box is cleared. See ***The Schematic Editor Window*** for more information on setting mouse wheel behavior.
Ctrl + **roll mouse wheel**	Scroll horizontally if **Zoom with mouse wheel** check box is selected. See ***The Schematic Editor Window*** for more information on setting mouse wheel behavior.

Grid Setup

Adjust the visibility, color, resolution, and other characteristics of both the Schematic Editor and Symbol Editor grids.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> The **Schematic Editor** and the **Symbol Editor** grid settings are independent of each other.

1. To access these settings, do one of the following:
   a. If using the Schematic Editor, click **Schematic** > **Grid Setup**.
   b. If using the Symbol Editor, click **Symbol** > **Grid Setup** on the **Symbol** menu.
   c. Click the **Grid Setup** icon on the **View** ribbon.

   The **Grid Setup** dialog box opens displaying the settings described below.

   Grid Lines Group

   - **Pin grid** – Display and specify the spacing of the pin grid lines. To specify a different value, type a new value into the **Pin grid** box. You can also change the unit of measure. Because symbol pins always snap to the pin grid, changes to this setting in the Schematic Editor affect the size or scale of displayed symbols.

     > **Note:**
     >
     > You can see the effect of changes to this setting more clearly if you apply a border to the schematic with **Schematic** > **Page Borders**.

   - **Major** – Display and specify the spacing of the major grid lines. To specify a different major value, type a new value into the **Major** box. To change the color of the major grid lines, click **Major**, specify a color in the **Color** dialog box and click **OK** to close the **Color** dialog box.
   - **Minor** – Display and specify the number of minor grid divisions between major grid lines. To specify a different value, type a new value into the **Minor** box. To change the color of the minor grid lines, click **Minor**, specify a color in the **Color** dialog box and click **OK** to close the **Color** dialog box.
   - **Show Grid** – Toggles grid line visibility. Select **Show Grid** to make the grid lines visible, or clear **Show Grid** to turn grid line display off.
   - **Snap Text and Graphics to Grid** – Control whether text and graphics (arcs, circles, lines, polygons, rectangles) snap to the nearest grid intersection on placement.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> To ensure connectivity among schematic elements, symbol pins always snap to the pin grid regardless of the **Major** and **Minor** grid line settings. This snapping cannot be disabled.

Other Group

- **Background Color** – To set the editor background color, click the color box. The **Color** dialog box opens. Specify a color and click **OK**.
- **Save as Default** – Select this box to save the current **Grid Setup** values as defaults for use across Twin Builder sessions.
- **Defaults** – Click to restore the **Major**, **Minor**, and **Background Color** settings to their factory defaults.
- **OK** – Click to commit changes made in this dialog box and close it.
- **Cancel** – Click to close this dialog box without committing changes made in it.

2. Make the desired settings and click **OK** to apply them and exit the dialog box.

Schematic Page Setup

Adjust the page size, orientation, borders, title blocks, and other characteristics of the schematic editor pages. Click **Schematic** > **Page Borders and Title Blocks** to open the **Page Border, Title Block, Page Properties** dialog box, which contains the following three tabs:

- **Page Borders**
- **Title Block**
- **Page Properties and Display**

Page Borders Tab

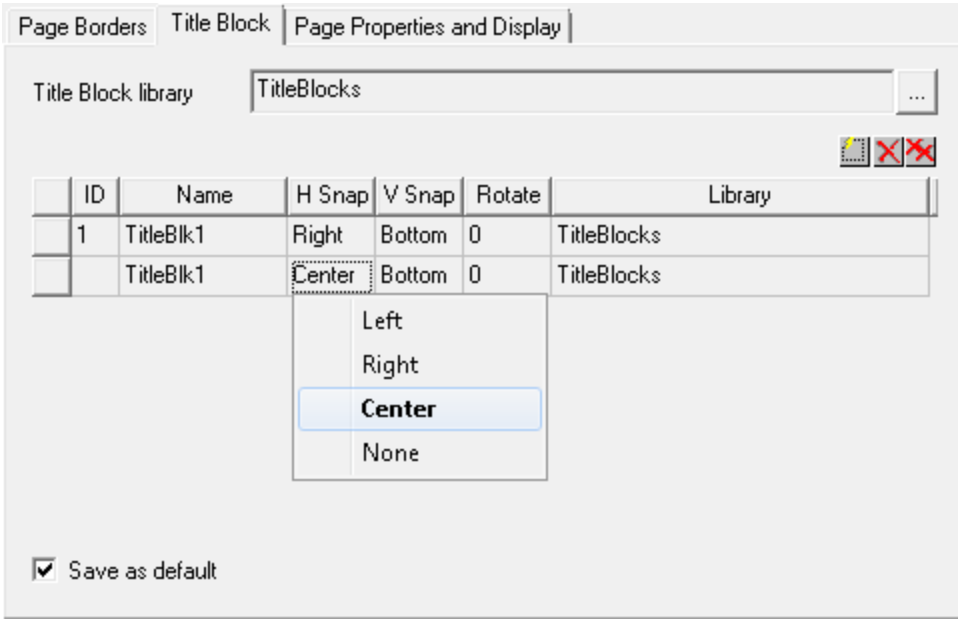
Use the **Page Borders** tab to set the border type, page size, page margins, and ANSI Zones for the schematic editor display. Select **Save as default** to save the current tab settings as defaults for new schematic pages.

- **Border Type**
  - **None** – No border displays. All other settings in the tab are disabled when **None** is selected. If title blocks have been defined, they will be hidden.
  - **Outline only –** Display a simple outline of the specified **PageSize**. **Margins** can also be adjusted.
  - **ANSI Zones –** Display fully ANSI-compliant borders of the specified **PageSize**. The specified **Number of Zones** are added to the border.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

  - **ISO Zones –** Display fully ISO-compliant borders of the specified **PageSize**. The specified **Number of Zones** are added to the border.
  - **DIN Zones –** Display fully DIN-compliant borders of the specified **PageSize**. The specified **Number of Zones** are added to the border.
- **Page Size –** Choose from a number of preset vertical and horizontal page sizes. Select **Custom** to set your own page **Width** and **Height**.
- **Number of Zones –** Set the number of **Vertical** and **Horizontal** zones to draw. The minimum and maximum number of zones allowed varies with page size. If you try to set a number outside of the allowable range, a reminder appears displaying the allowable range of values.
- **Margins –** Set the **Vertical** and **Horizontal** margins in inches.

Title Block Tab



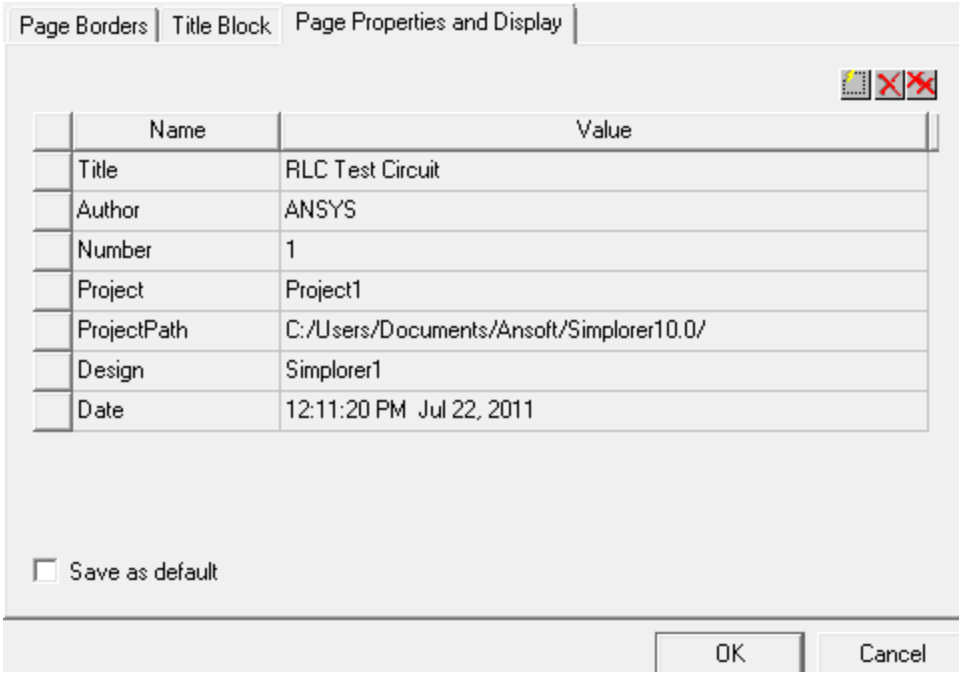
- On the **Title Block** tab, you can accept the default library or choose a **Title Block Library**. Click ... to locate a title block **aslb** library.
- To add a title block to the grid, click the yellow “lightning bolt” icon, and select a title block symbol from the list.
- Within the grid control, the title block can be set to snap along the bottom, top, or either side, or not snap at all. You can **Rotate** the title block 0, 90, or 270 degrees.
- You can move title blocks on the schematic much like any other element, though title blocks have optional snapping constraints. If a title block is set to snap (any choice but **None**), it can be moved between snapping points: Left, Center, and Right for horizontal,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Top, Center, and Bottom for vertical. The snap property on the title block will be changed accordingly. The snap location can also be changed through the property window or the **Page Borders and Title Blocks** dialog box.

- You can add more than one title block to a page. To delete unwanted title blocks from the page, click to delete a single entry, or to delete all entries. An ID property on the title block corresponds to the number in the **ID** column of this tab, so you can identify which title block to delete.
- Click **OK** to add all title blocks in the list to the schematic page, The blocks in the list snap to the border as specified, or as previously placed if the snap specification is **None**. If blocks were removed from the list, they are removed from the schematic. Any changes made to blocks in the list are applied to preexisting title blocks.
- Select **None** for **Border Type** after setting up border and title blocks to cause title blocks to be remembered but not drawn.
- You can create a symbol with company logo, *propdisplay* positioning, and as much additional information and graphics as needed and store it in a symbol library where it will be available for any schematic page.
- Select the **Save as default** check box to cause the title block information to be saved to the registry and used as a default for subsequent pages.

Page Properties and Display Tab



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Page properties include fields such as **ProjectPath**, **Project**, **Design**, **Title**, **Author**, and **Date**. Click a **Value** field and enter the desired information. Data entered in fields used by the title blocks appear in the title block on the schematic. Page properties are saved if the **Save as default** check box is selected, then initialized when you create a new page. Click to create additional page properties.

In the symbol editor, a *propdisplay* may be added to a title block symbol, and the title block *propdisplay* will then show the value of the page property with the same name.

Zooming and Panning the View

You can magnify (zoom in) or shrink (zoom out) the contents in the view window, or in a rectangular area in the view window. You can pan (scroll) the view in any direction. You can also fit the contents within a window and redraw a window's contents.

Zooming In and Out on the Window Contents

1. Follow this proceduret o magnify or shrink the contents in the view window:
   - Click **Zoom In** or **Zoom Out** on the **View** menu.
   - Right-click in the schematic window and select **Zoom In** or **Zoom Out**.
   - Click the **Zoom In** or the **Zoom Out** icon on the **View** ribbon.
   - Use **Ctrl** + **middle mouse button** and drag to zoom in and out.

     Alternatively, if you are using the **Enable Legacy View Navigation** option in **General** > **User Interface**, you can also use the legacy mouse-button and hotkey combination for zooming. Hold down the **Alt** + **Shift** keys as you click and drag using the left mouse button.

   - Roll the mouse wheel. Make sure to select the **Zoom with mouse wheel** check box in the **Options** dialog box, **Schematic Editor** > **General** section. See ***The Schematic Editor Window*** for more information on interacting with the Schematic Editor.

     Alternatively, if you choose to clear the **Zoom with mouse wheel** option, then hold **Shift** while rolling the mouse wheel to zoom in or out. Rolling the wheel without *Shift* will scroll the view in this case.
2. The view zooms to the chosen magnification. The absolute size of the model does not change.
3. Repeat the operation until you achieve the desired magnification.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Zooming in and out is limited to approximately 20 steps. The **Zoom In** or the **Zoom Out** icon on the **View** ribbon is dimmed when you reach the corresponding zoom limit.

Zooming In on a Rectangular Area

1. To magnify a specific rectangular area in the view window do one of the following:
   - Select **View** > **Zoom Area**.
   - Right-click the schematic window and select **Zoom Area**.
   - Select **View** > **Zoom Area** on the ribbon.

   The cursor changes to a magnifying glass.
2. Draw a rectangle to increase magnification in that area.

Restoring a Previous Zoom View

After zooming in or out, select **View** > **Zoom Previous** to return to the previous magnification level.

Fitting the Drawing to the Window

1. To scale the contents to fit within the current window, do one of the following:
   - Click **View** > **Fit Drawing**.
   - Press Ctrl+D.
   - Right-click the schematic window and select **Fit Drawing**.
   - Click on the toolbar.
2. The view zooms to fit the contents of the drawing within the bounds of the window. The absolute size of the model does not change.

Fitting the Drawing Border to the Window

1. When a border is added to a drawing, click **View** > **Fit Border** to scale the bordered contents to fit within the current window. You can also click Ctrl+B to fit the border.
2. The view zooms to fit the bordered contents of the drawing within the bounds of the window. The absolute size of the model does not change.

Panning the View

To pan (scroll) the view in any direction:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Select **View** > **Pan**. The cursor changes to a hand as you move it over the schematic window. Click and drag to pan the view in any direction.
- **Ctrl + middle-click** and drag to pan the view in any direction.
- If the *Enable Legacy View Navigation* option is selected in the User Interface Options, you can also use the legacy combination (**Shift + click**) and drag the cursor to pan the view in any direction.
- Click on the ribbon. The cursor changes to a hand as you move it over the schematic window. Click and drag to pan the view in any direction.
- In the Symbol and Schematic Editors, hold Shift or Ctrl and roll the mouse wheel to pan vertically or horizontally. Make sure to select the **Zoom with mouse wheel** check box in the **Options** dialog box, **Schematic Editor** > **General** section. See ***The Schematic Editor Window*** for more information.

Redrawing a View

Click **View** > **Redraw** to redraw the current window.

Closing the Schematic Editor

Click the standard window close box to close the **Schematic Editor**.

Creating Simulation Models Using the Schematic Editor

In general, simulation models are represented by schematic designs consisting of components, elements, and simulation (analysis) parameters.

Components	Constituent parts of a complete simulation model. Some examples of components include: resistors, valves, motors, switches, logic gates, measuring devices, function blocks, sensors. Twin Builder includes many pre-defined components in its component libraries. Component types include: internal components, C-Models, VHDL-AMS models, SML models, and SPICE models. You can also construct and import their own components.
Elements	Connections, display of component names and parameter on the sheet, drawing elements, text elements, reports.
Simulation parameters	Simulation settings for the simulator and compiler.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The following sequence of steps describes how to create a simulation model using the schematic editor.

1. Select and place components on a schematic sheet. See the documentation about the components for detailed information on the types of components you can use to create simulation models.

   > **Note:**
   >
   > These components can also include subcircuit models coupled to external applications.

2. Define component parameters and properties.
3. Connect the components.
4. Specify simulation (analysis) parameters.
5. Add drawing elements (including text and imported graphics), and reports as needed.

Selecting and Placing Components

Components are contained in libraries. The **Components** tab of the Component Libraries lists the installed libraries in folders. Each of these folders can contain one or more libraries. See the ***The Components Tab*** section for additional information.

To place a component on the schematic, select the schematic that will contain it, then:

1. In the Component Libraries window, click the **Components** tab.
2. In the component list, locate the component you want to place, opening library files and folders as necessary to find it.

Hint	Searching for components in the installed libraries can be time-consuming. The **Component Libraries****Search** feature minimizes the complexity of the search and allows quick access to all models.

3. Do one of the following:
   - Double-click the component, then move your mouse cursor into the schematic. Inside the schematic window, the cursor is accompanied by the component symbol for placement.
   - Drag the component to the desired location.
   - Right-click the component and select **Place Component**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Position the cursor where you want to place the component, and click.

Hint	• You can rotate a component *before placing it* by repeatedly pressing **R** on your keyboard. Each press rotates the component 90º counterclockwise. After placement use **Ctrl+R** to rotate a component. • Similarly, you can flip a component *before placing it* by pressing the **X** key to flip it left-to-right; or by pressing the **Y** key to flip it top-to-bottom. After placement use **Ctrl+X** and **Ctrl+Y** to flip a component.

If Multiple Placement is turned on for components in the **Schematic Editor Options** dialog box, click additional locations to place additional instances of the component.

To stop placing components during multiple placement, do either of the following:

- Press **Enter**, the **SPACEBAR**, **Backspace**, or **Esc** on your keyboard.
- Right-click and select **Place and Finish** or **Finish**.

Hints	To help facilitate the easy copying of design materials, you can drag and drop components to the **Schematic Editor**. In particular, you can select and drag *components*: • From the Project tree Definitions folder to a schematic • From the Component Libraries Search tab component list to a schematic • From the Component Libraries Components tab component list to a schematic

Note:

1. The first time you place a component in a project, entries for it are added under the Component tab tree's **Most Recently Used** and **Project Components** headings. To save time as your work progresses, you can double-click these icons to place new instances of a component.

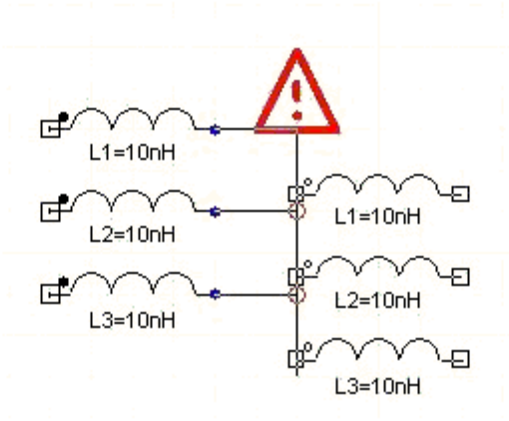
   • For security reasons, encrypted components are not saved in the **Most Recently Used** list or the **Favorites** list.

2. To ensure electrical connectivity among schematic elements, the pins of placed components snap to a 100-mil (2.54-millimeter) grid. This snapping cannot be turned off, and the spacing of the connectivity grid cannot be adjusted.
3. To move an existing component, select and drag the component to a new position. Wiring to the component is adjusted; that is, the wiring "follows" the component. To retain the in-place wiring, hold down the **Alt** key as you drag the component to a new position.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Warning:

When dragging a component to a new position, if the wires attached to it make an unintended connection, a red exclamation mark is displayed as a warning. The figure below illustrates the red circles used to indicate where connections will be made and the exclamation point warns that there may be a connection that is not what you intended.



Editing Component Parameters

You can use Twin Builder to enter component parameter values in multiples of standard SI units such as millivolts, nanoamperes, and kilometers, as well as in non-SI units such as pounds per square inch, degrees Fahrenheit, and feet. Twin Builder performs all of the necessary unit conversions, reducing the time needed to calculate component parameters.

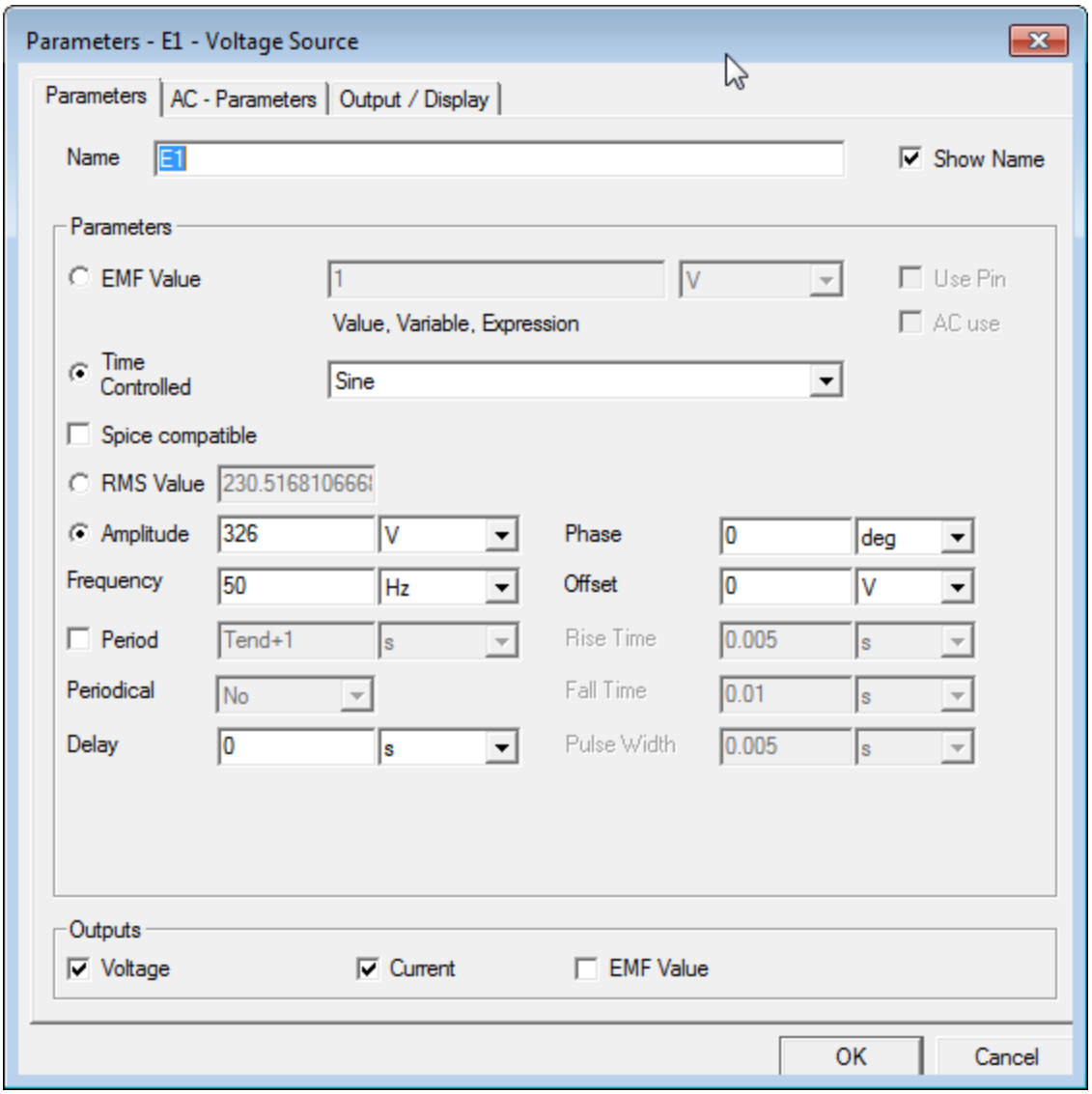
Component parameters that are physical quantities can be assigned expected units of measure. These default units are the ones associated with the parameters during simulation. In addition to the expected (default) unit of measure, an additional set of units known as “used units” can also be specified for the same physical quantity. For example, the amplitude of a force source may have an expected unit of *Newtons* and a set of “used units” that includes *pounds* and *dynes*.

Note:

Twin Builder internal models have predefined default units for their parameters. Parameters of user-defined models such as C-Models and text macros can be assigned expected units when they are created.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To display the editable parameters of a component, either right-click the component and select **Display Component Dialog**, or double-click the component. A typical component **Parameters** dialog box is shown below.



A component's **Parameters** dialog box opens with its left-most tab selected by default; and with the component instance **Name** selected.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

Twin Builder assigns a unique **Name** to each component instance you place on a schematic.

- If you choose to *rename* a component instance, *do not duplicate the name of any other component instance* (for example, **R1** and **r1**). Duplicate instance names result in netlist errors when attempting to compile a circuit prior to analysis.
- The first character of a name must always be a letter.
- Vowel mutations (for example, umlauts) are not allowed.
- Spaces are not allowed.

The parameters and associated units of measure shown in the component dialogs vary by component. The dialog box may also contain additional parameter settings tabs - depending on the component type. It also contains an **Output/Display** tab on which you can enable parameters as outputs, and control how they are displayed on schematics.

Note:

- Component **Properties** dialog boxes also provide access and control of the parameters shown in the **Parameters** dialog boxes.
- You must use the **Properties** dialog boxes to edit parameters of components that do not have a **Parameters** dialog box.

Use the various radio buttons, check boxes, drop-down lists, and text boxes to edit parameter values and - where applicable - to select the unit of measure. When new units for a component's parameters have been selected and the component is simulated, Twin Builder converts quantities expressed in the "used units" to equivalent quantities of the default (expected) units. In other words, component parameters can be expressed in units from different systems of measurement, and the simulation still produces accurate results.

You can also specify parameter values via input from an external component. To do this, select the **Use Pin** check box next to the desired parameter. The parameter's value field is grayed-out to show that the value of the parameter will now be determined by the component to which the pin is connected.

Related Topics

Copying and Pasting Properties

The Parameters Dialog Box Output/Display Tab

Use the **Output/Display** tab to control:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- If a parameter pin displays.
- If a parameter may be swept.
- If a parameter's values are stored in the project database.
- How parameters and their values display on a schematic page.

Descriptions of the column headings for the **Output/Display** tab are as follows:

- **Name** - Non-editable field listing the names of component parameters.
- **Description** - Non-editable field that provides a brief description of the parameter.
- **Show Pin** - Show or hide a parameter pin.
- **Sweep** - When selected, enables the parameter value to be swept during an analysis.
- **SDB** - When selected, the parameter data is saved in the project database **.sdb** file.
- **Direction** - describes whether a parameter is an Input or an Output.
- **Visibility** and **Location** - Control the display and location of parameter information labels on a schematic sheet. These settings are the same as those found on the Property Display Tab of a component's Properties dialog box.

Editing Component Properties

You can edit component properties using any of the following methods:

- **Properties** window — Click a component to select it. The **Properties** window displays the properties of the selected component on a series of tabs where they may be edited.
- **Properties** dialog box — Right-click a component and select **Properties**. In addition to the tabs available on the **Properties** window, the Properties dialog box also contains a **Property Displays** tab, which can be used to control the display of properties as text labels on the schematic.

> **Note:**
>
> - If you select multiple components, only those properties common to all selected components are shown in the Properties window or dialog box. Changes made to any of these common properties will affect all selected components. For example, you can use this feature to select and deactivate several components simultaneously.
> - Select the **Show Hidden** check box to view hidden properties of the component. Hidden properties contain system-defined values and rules for interpreting predefined component parameters. Modifying hidden properties requires specialized knowledge of the component, and is not needed for normal operation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- In-place editing — Double-click a property that is displayed near the component to start an in-place edit of the property value. You can also move the property text box by dragging it to the desired location.

Related Topics

Copying and Pasting Properties

Copying and Pasting Properties

You can copy and paste properties for primitive drawing elements (graphical objects) and components.

Primitive drawing elements (graphical objects)

You can copy and paste the common properties of the following primitive drawing elements (graphical objects):

- Arcs
- Circles
- Lines
- Rectangles
- Polygons
- Text
- Images

Components

For components, you can copy either the value of properties or the value and attribute of properties. Properties are copied only when property names match.

Parameters, quantities, signals, and property displays are copied along with properties and their attributes.

> **Note:**
>
> Property displays are copied to same location as the source if they are on the left, bottom, right, top or center. Custom location property displays are copied to the default bottom location.

There are two types of components: components that have user-defined parameters, and those that do not have user-defined parameters. User defined properties are created in two ways:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Properties created by users in equations such as FML and FML_INIT
- Properties that are dynamically created depending on certain parameters. An example is the GS component, where coefficient parameters are created dynamically based on the values of numerator(n) and denominator(d).

The options available in the **User Defined Properties of Component** pane differ according to the type of component, as shown in the following table.

Component type	User-Defined Properties Information	Examples
Component has user-defined parameters	Components for which copy properties allow replacing only existing parameters.	GS, NXMY, NDTAB, and DES
	Components for which copy properties allow either replacing existing parameters or adding parameters to the existing list. Note: Only components that have equations allow adding more to their existing properties.	FML, FML_INIT, EQUBL, and State components, such as STATE_01, STATE_10, STATE_33, and STATE_Flexible
Component does not have user-defined parameters	The **User Defined Properties of Component** pane is unavailable when there are no user-defined properties for the component.	R, L, G, C, E, D, 2, DELAY, DIFF, and GAIN

Related Topics

To Copy and Paste Selected Properties for Components

To Copy and Paste Common Properties for Primitive Drawing Elements

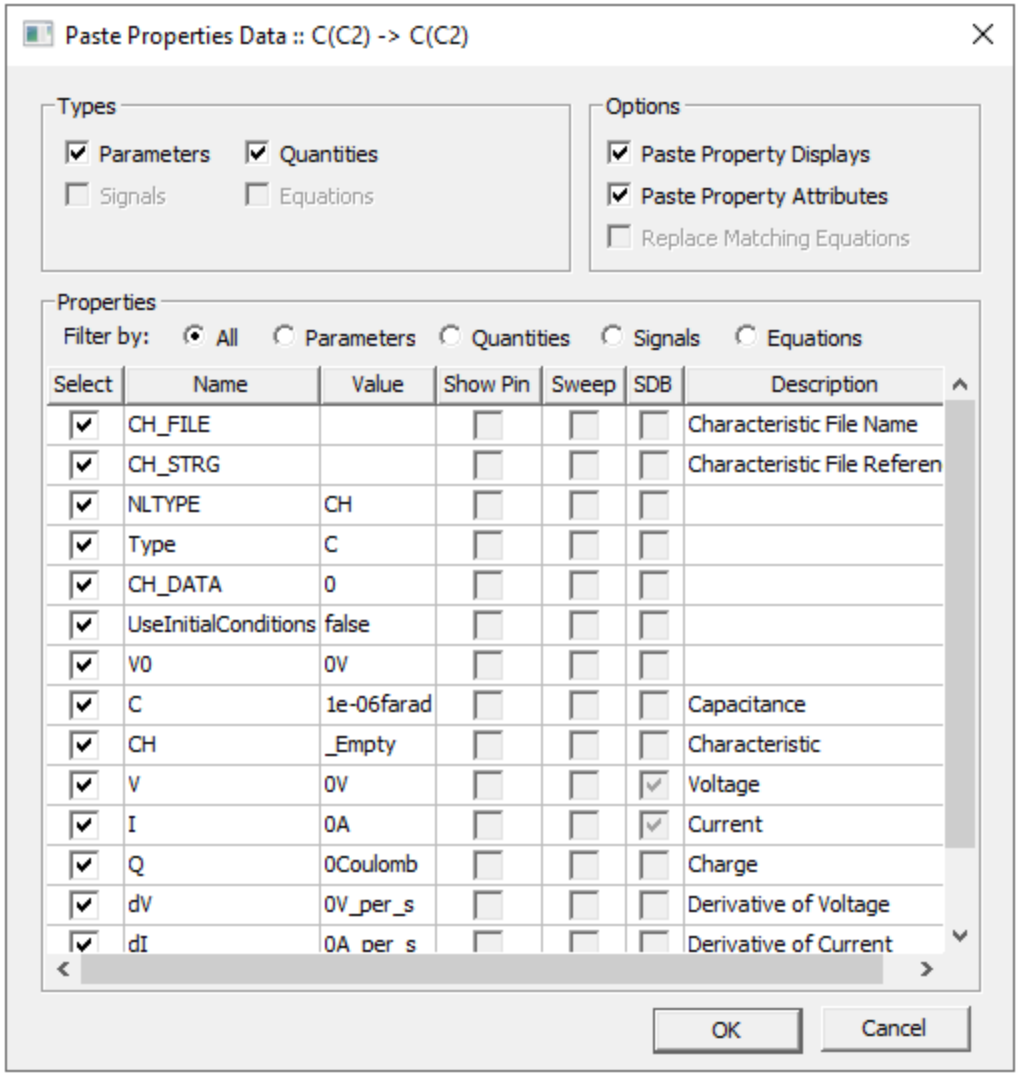
To Copy and Paste Selected Properties for Components

Follow this procedure to copy and paste selected properties from one component to one or more components.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select the component you want to copy from.
2. Use one of the following methods to copy data:
   - Select **Edit** > **Copy Data**.
   - Right-click the object and select **Copy Data**.
   - Press Ctrl+Shift+C.
3. Select the components you want to paste to.
4. Paste the properties by using one of the following methods:
   - Select **Edit** > **Paste Data**.
   - Right-click the component and select **Paste Data**.
   - Press Ctrl+Shift+V.

   The **Paste Properties Data** dialog box appears.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. In the **Copy Data** pane, select either **Value and attributes** or **Only value**. When you select **Only value**, the attributes columns are hidden.
6. In the **Options** pane, select an option. The pane is unavailable when there are no user-defined properties for the component.
   - **Paste Properties Displays** – Copy property display settings.
   - **Copy Property Attributes** – Copy property attributes.
   - **Replace Matching Equation** – Select this check box to replace all target equations with the source equations; clear the check box to add source equations to the target without deleting the existing target equations.
7. Do one of the following to select the properties to copy:
   - Select one or more of the following check boxes: **Parameters**, **Quantities**, **Signals**, or **Equations**.
   - Click **Select** in the first column to toggle between selecting all and clearing all.
   - Select the **Select** check box for the property. If you do not want to include the property, clear the **Select** check box.

   In the **Filter by** options, you can filter the view by selecting **All**, **Parameters**, **Quantities**, **Equations**, or **Signals**.

   Select the **Show Description** check box to display the descriptions for each property.
8. Click **OK** to paste the properties into the component.

Related Topics

Copying and Pasting Properties

To Copy and Paste Common Properties for Primitive Drawing Elements

To Copy and Paste Common Properties for Primitive Drawing Elements

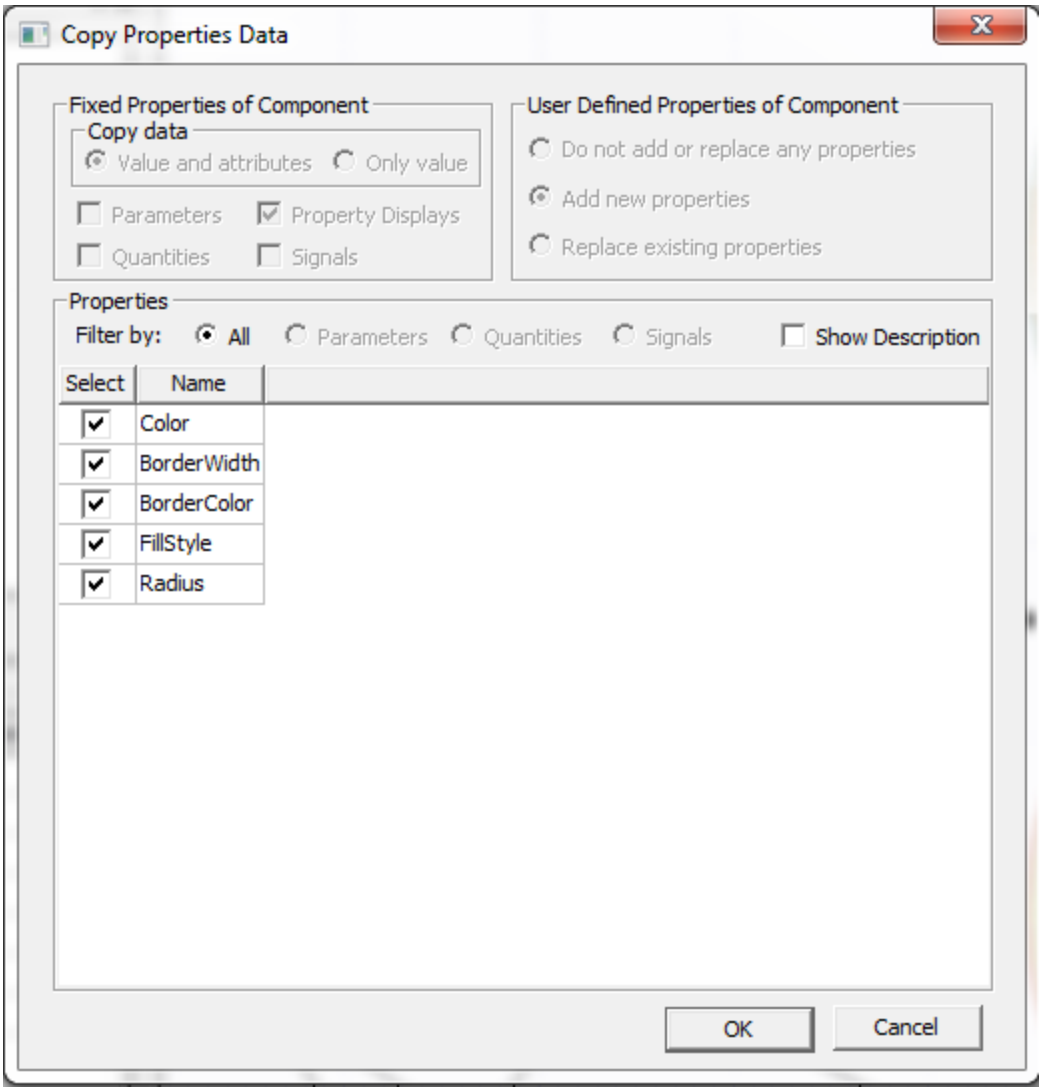
Follow this procedure to copy and paste the common properties from a primitive drawing element (graphical object) to another:

1. Select the object you want to copy from.
2. Use one of the following methods to access Copy Data:
   - Select **Edit** > **Copy Data**.
   - Right-click the object and select **Copy Data**.
   - Press Ctrl+Shift+C.
3. Select the object you want to paste to.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Paste the properties by using one of the following methods:
   - Select **Edit**> **Paste Data**.
   - Right-click the component and select **Paste Data.**
   - Press Ctrl+Shift+V.

   The **Copy Properties Data** window appears.



5. In the **Properties** pane, keep the default selections or clear the check boxes for the properties you do not want to copy.

   Select the **Show Description** check box to display the description for each property.

6. Click **OK**. The properties are pasted into the object.

Related Properties

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Copying and Pasting Properties

To Copy and Paste Selected Properties for Components

Favorites and Most Recently Used Components

When the **Schematic Editor** is the active window, the **Components** tab in the Component Libraries window keeps track of the components that have been **Most Recently Used**. You can also add components you use frequently to the **Favorites** list.

- To add a component to the **Favorites** list, right-click the component in the listing and select **Add to Favorites**.

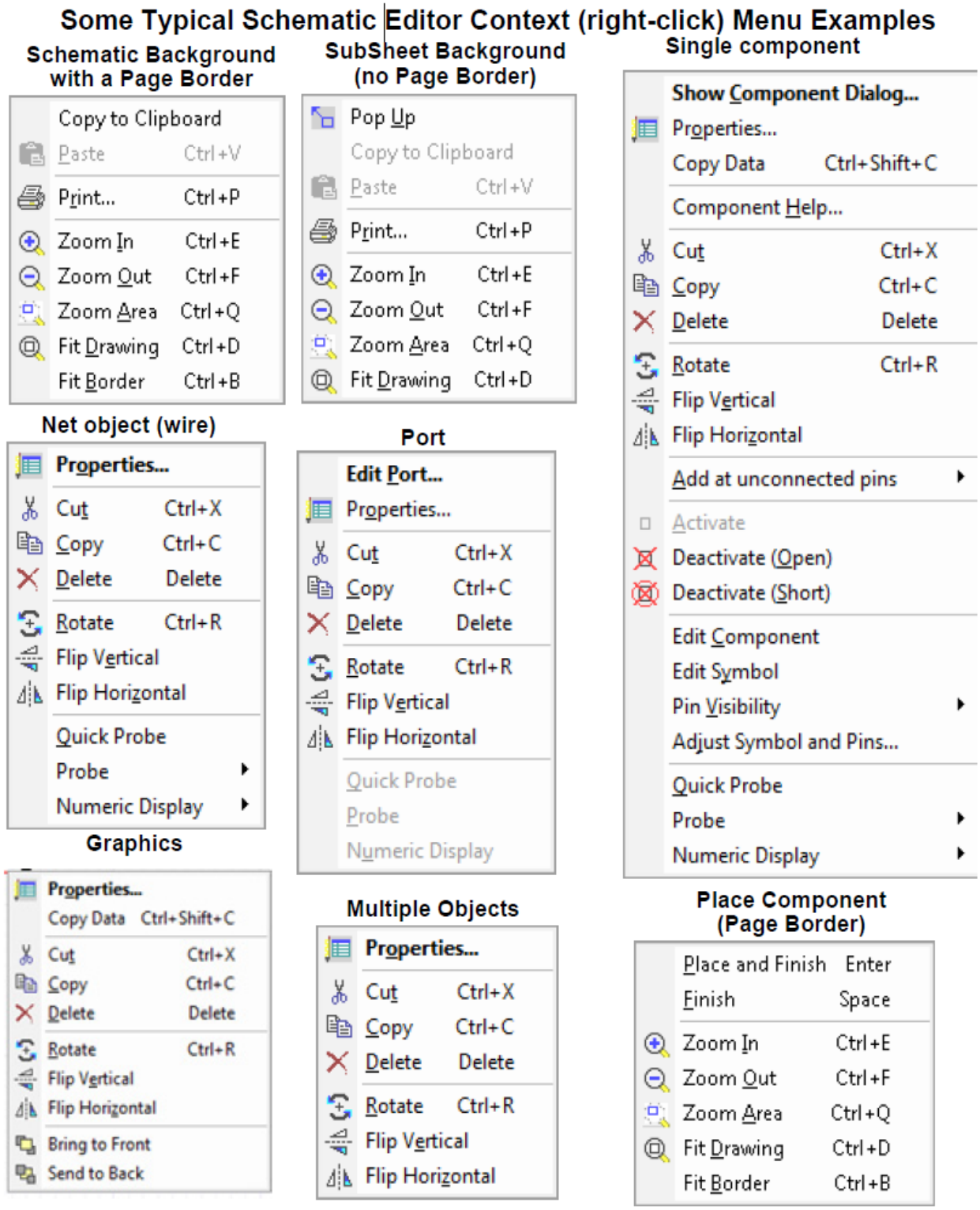
  > **Note:**
  >
  > For security reasons, encrypted components are not saved in the **Most Recently Used** list or the **Favorites** list.

- To remove a component from the **Favorites** list, right-click the component in the **Favorites** list and select **Remove from Favorites**.

Operations on Components and Design Objects in the Schematic Editor

Right-click a component or other design object in the schematic editor to select the object and open a context menu. Some context menu examples are shown below. Default commands (typically those that are invoked when you double-click an object) are at the top and **bold**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The following describes the possible **Schematic Editor** context menu commands.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Properties** – Opens the **Properties** dialog box for the selected component.
- **Component Help** – Opens the help topic for the selected component.
- **Show Component Dialog...** – Opens the component Parameters dialog box for the component.
- **Add at unconnected pins** – Adds either **Page Connectors**, **Interface Ports**, or **Grounds** to unconnected pins of the component.
- **Cut** – Deletes the selected component or wire segment, and retains a copy for pasting into a schematic in the same application.
- **Copy** – Creates a local copy for pasting into a schematic in the same application.
- **Delete** – Deletes the component.
- **Rotate** – Rotates the component 90 degrees counterclockwise. (See **Rotate** in ***Drawing Operations***.)
- **Flip Vertical** – Flips the component in the Y-direction. (See **Flip Vertical** in ***Drawing Operations***.)
- **Flip Horizontal** – Flips the component in the X-direction. (See **Flip Horizontal** in ***Drawing Operations***.)
- **Bring to Front** – Moves the selected object to the front of the drawing. (See **Bring to Front** in ***Drawing Operations***.)
- **Send to Back** – Moves the selected object to the back of the drawing. (See **Send to Back** in ***Drawing Operations***.)
- **Activate** – Restores a deactivated component to the circuit.
- **Deactivate (Open)** – Temporarily converts the component into an open circuit. This is displayed graphically with a red X over the circuit element.
- **Deactivate (Short)** – Temporarily converts the component into a short circuit. This is displayed graphically with a circled red X over the circuit element. Deactivating (short) a component with multiple conservative pins will connect all the conservative pins to the same net. Deactivating (short) a block (which contains non-conservative pins) does not work the same as the circuit component, and the output of the shorted block will be 0.
- **Edit Component** – Lets you edit the component using the Component Editor.
- **Edit Symbol** – Lets you edit the component symbol using the Symbol Editor.
- **Copy as New Design** – Present if the component represents a subcircuit. Lets you copy hierarchical designs (subcircuits) such that the pasted copy is independent of the original source design. This menu item is also present in the context menu for a subdesign in the **Project Manager** pane. (See also: ***Copying a Twin Builder Design into Another Design***)
- **Edit Model** – Lets you edit the component model using the appropriate model editor: SML Model Editor, VHDL-AMS Model Editor, C-Model Editor, SPICE Model Editor. The **Message Manager** informs you if the model is not editable.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Alternatively, you can open the model for editing. Right-click the model symbol in the Project Manager Project tab, and select **Model** from the context menu.

- **Pin Visibility** – Lets you toggle visibility of individual component pins.
- **Adjust Symbol and Pins** – Lets you move unconnected pins around the circumference of the bounding rectangle of a selected component (such as coupling components and components that represent subcircuits). Also lets you resize the bounding rectangle. (See ***Adjusting Symbols and Pins*** for additional information.)
- **Revert to default symbol** – present if changes have been made to a component symbol using the **Adjust Symbol and Pins** command. Lets you revert to the component's original symbol. (See ***Adjusting Symbols and Pins*** for additional information.)
- **Push Down** – Moves down one level in the project design.
- **Pop Up** – Moves up one level in the project design.
- **Quick Probe** – Inserts a 2D rectangular plot-on-schematic of a parameter in the list of defined outputs for the component. Quick Probe first looks for **V** if it exists, then **I**, then the first quantity not starting with a lower case "**d**", then the first quantity (in that order).
- **Probe** – Inserts a 2D rectangular plot-on-schematic of a component parameter chosen from a list of defined outputs for the component. Default parameter types can be selected on the Schematic Editor Options>Twin Builder Schematic panel.
- **Numeric Display** – Adds a data table to the schematic. You can choose which quantity of the selected component to display from the submenu. By default, only the last value calculated is displayed.
- **Copy to Clipboard** – Creates a global copy on the clipboard for pasting into a different application.
- **Paste** – Puts the local object from the previous copy or cut into the schematic.
- **Print** – Prints the schematic in the active window.
- **Zoom In** – Decreases the area of the schematic in the view.
- **Zoom Out** – Increases the area of the schematic in the view.
- **Zoom Area** – Specify the new view area. Click to specify the upper left and lower right corners.
- **Fit Drawing** – Changes the view to include all the objects currently present.
- **Fit Border** – Changes the view to include the page border. (See ***Page Borders Tab***.)

Right-click a component in the **Components** tab of the Component Libraries window to open a menu containing these commands:

- **Add to Favorites** – Adds the component to your list of frequently-used components (see ***Favorites and Most Recently Used Components***).
- **Place Component** – Attaches the component symbol to the cursor for placement in the schematic (see ***Placing Components on a Schematic***).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Edit Component** – Opens the Component Editor dialog (see ***Editing Component Libraries*** for details).
- **View Component Help** – Opens the help topic for the component (see ***Help for Components***).
- **Load Example** – Loads the example project associated with the component.

Characteristics in Simulation Models

Some components use characteristics to specify their model properties. Use characteristics to create a unique transfer relationship between the output and input of a model. The user-defined characteristic curve can have virtually any shape, however, it must be monotonic in X-value. Component types that use characteristics include: nonlinear passive components, nonlinear sources, system-level semiconductors, nonlinear transfer blocks, and nonlinear DC machines.

Twin Builder provides the means to use both predefined and user-defined characteristics.

Predefined characteristics are located in the **Basic Elements** > **Tools** > **Characteristics** folder on the **Component Libraries** window **Components** tab.

User-defined characteristics can be assigned to a component via a component dialog box itself, or via separate characteristic components such as the 2D and Multidimensional Lookup Table components. You can also enter characteristics data directly into datasheets, or import data from a separate .mdx, .mda, .mdk, .xls, .xlsx, .txt, .csv, .out, .cfg, or .dat file.

User-defined characteristics can also be generated using the SheetScan tool, which renders a characteristic curve graphic into a simulator-legible format.

Characteristics in Component Dialog Boxes

To define a characteristic directly in a component's dialog box, click **Characteristic**. This button is only available for components that exhibit nonlinear behavior such as nonlinear passive elements and sources.

When the Characteristic dialog box opens, select either an internal or external reference option for the characteristic. Internal references include Component references and Datasets. External references can be to variables or an external data file.

Characteristic Reference via Separate Component

To define a characteristic by name reference to a external component in the dialog of a component, select the **Internal Reference** option. Then select **Component** and enter the name of the qualifier of the external component (such as XY1.VAL) in the text box. The characteristic reference cannot be a number or expression. It must be _Empty or the VAL output of a characteristic component such as: XY, EXP, HYP, CDI, CSP, EQUL, PO2; or the VAL array element of NDNL, NDTAB, or NXMY. If you attempt to enter an invalid reference, an error dialog box displays informing you of the invalid entry, and recommending possible remedial actions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> If you attempt to enter an invalid reference, an error dialog box displays informing you of the invalid entry, and recommending possible remedial actions.

This reference may also be accomplished using a pin. To do this, select the **Use Pin** check box in the component's dialog box rather than opening the **Characteristic** dialog box. The pin to connect the external component becomes visible on the sheet.

Characteristic Reference via Datasets

If datasets are already defined in the project, they can be selected from the drop-down list. Click **Datasets** to open the Datasets dialog box where you can add, edit, and import datasets. You can also use SheetScan to define characteristic datasets from images of datasheets.

Characteristics in Files

To define a characteristic using an external file, select **External Reference**, then select **File Name**. You may browse to the external file using the available buttons.

Using Pins

Pins are graphical connectors for linking nodes or parameters with a wire. There are two different pin types: *conservative nodes* (also referred to as terminals) and *non-conservative nodes* (which include parameters, inputs, and outputs).

Conservative	• Conservative pins belong to components of physical domains (electrical, fluidic, magnetic, mechanical, thermal). Each conservative pin has a nature type —such as: electrical, thermal, or fluidic — assigned to it. • Conservative pins have no direction attribute. • You can connect only pins with the same nature type directly. • To connect pins of different natures use the Domain-to-Domain (D2D) component located in the **Tools>Nature Transformation** library. • To connect conservative pins or wires to non-conservative nodes, use the Conservative-to-Non-Conservative (C2NC) component, also located in the **Tools>Nature Transformation** library. The C2NC component functions as an output and supplies the across quantity from the conservative domain (such as voltage) to the non-conservative input(s). • Conservative pins can be displayed or hidden. If you hide a conservative pin, you can define the connection through a name reference.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Non-conservative	• Non-conservative pins can belong to circuit as well as block diagram and state graph components. Non-conservative pins represent many different data types. A real-valued quantity (for the magnitude of a voltage source), a record of mixed data (for a VHDL-AMS model), or a std_logic enum (for a digital component) are examples of data types. The data type includes size for arrays, so that an 8-wide array of real values is not the same type as a 16-wide array of real values. • Non-conservative pins can have one of the following direction attributes: IN, OUT, or IN/OUT. • You can connect only pins of the same data type directly. • To connect different data types, **OmniCaster** components, located in the **Tools** library may be used for many situations. For size-related changes, **Bus Entries** may be used to specify an appropriate subset for connection between wider and narrower nodes. • Non-conservative pins can be displayed or hidden. If you hide a component pin, you can define the connection through a name reference.

Related Topics

Displaying and Hiding Pins

Adjusting Symbols and Pins

Displaying and Hiding Pins

When an instance of a component has been placed on a Schematic sheet, you can show or hide its conservative or non-conservative pins through the right-click **context menu**, its **Properties Dialog**, or its **Special Component Dialog**.

- If a non-conservative pin does not exist on the default symbol for the component, or if a conservative pin's visibility is changed (conservative pins always exist on the symbol), a copy of the symbol will be made and used for this component instance.
- New pins are added on the left and right periphery of the graphics bounding box, alternately. There are a limited number of available pin locations for a symbol unless it has a rectangle enclosing the symbol graphics. If all the available spaces are full, the context menu shows only pins that can be hidden or made visible without adding a new pin. If the symbol has a rectangle enclosing its graphics, however, the rectangle can be extended downward to accommodate any number of pins, and all possible pins are shown in the context menu. Once extended, the rectangle will not be reduced if pins are hidden.
- When a conservative pin is hidden, a property **Net**(*pinname*) is added to the component instance's parameters so the pin can be connected to an existing net, by name. The value

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

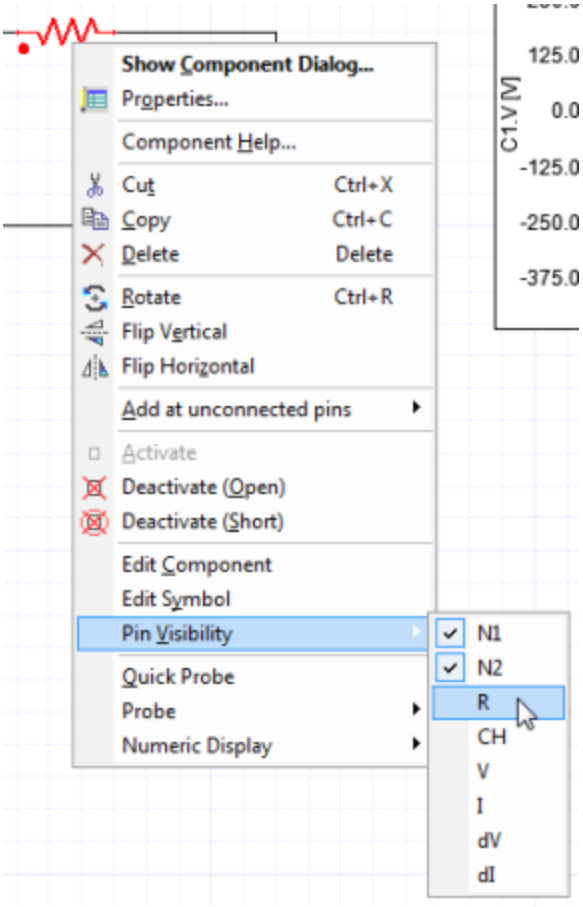
is checked for correct net name syntax. If the name corresponds to a current net in the schematic, the pin is connected. If the name does *not* correspond to a current net, a global port with that name is created. Multiple hidden pins can be connected by using the same name for the value of their **Net**(*pinname*) property.

- When a non-conservative pin is displayed, its property value is determined by its connection. All **Inputs** take the value of a single **Output** property on the net. **Output** property values are determined by the simulator and generally are set to 0 in the schematic, regardless of pin visibility - with the exception that during simulation, **Output** values may be set to evaluate symbol animation conditions. For an **InOut** pin, the property value is set to an **InOut** design parameter. This is necessary for all **InOut** pins to be connected in the simulator, which will ensure input/output validity for the net.
- When a non-conservative **Input** or **InOut** pin is hidden, the property value becomes editable. An **Input** value may be set to an expression involving variables, functions, and any number of output parameters from components in the schematic. An **InOut** value must be an **InOut** design parameter, and all **InOut** pins sharing this value are connected.

Context Menu

On the right-click context menu for a component, the **Pin Visibility** menu lists pins with checkmarks to show visibility. Select a pin to toggle it's visibility.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Properties Dialog (Show Pins)

1. To open the Properties dialog box of an instance of a component, right-click it and choose Properties from the context menu.
2. Once the Properties dialog is open, select the Parameters, Quantities or Signals tab. These tabs contain a list of all of the component's properties, most of which can be pins.
3. To show a pin, select the **Show Pin** check box for the desired pin. To hide a pin, clear the check box.

> **Note:**
>
> The **Show Pin** check boxes are disabled for properties that cannot be displayed as pins. Connections to hidden pins can be defined via a name reference.

4. Click **OK**. All selected parameters and nodes receive a pin on the sheet.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Special Component Dialog (Show Pins)

1. To open the Special Component Dialog of an instance of a component, double-click the component or right-click and choose Show Component Dialog from the context menu.
2. Once the dialog is open, select the Output / Display tab. This tab contains a list of all of the component's properties, and a **Show Pin** check box column.
3. As above, to show a pin, select the **Show Pin** check box for the desired pin. To hide a pin, clear the check box.

   > **Note:**
   >
   > The **Show Pin** check boxes are disabled for properties that cannot be displayed as pins. Connections to hidden pins can be defined via a name reference.

4. Click **OK**. All selected properties receive a pin on the sheet.

Names at Pins

The Property Displays tab controls the display of component properties associated with pins on the schematic. Displayed properties appear as text labels on the schematic.

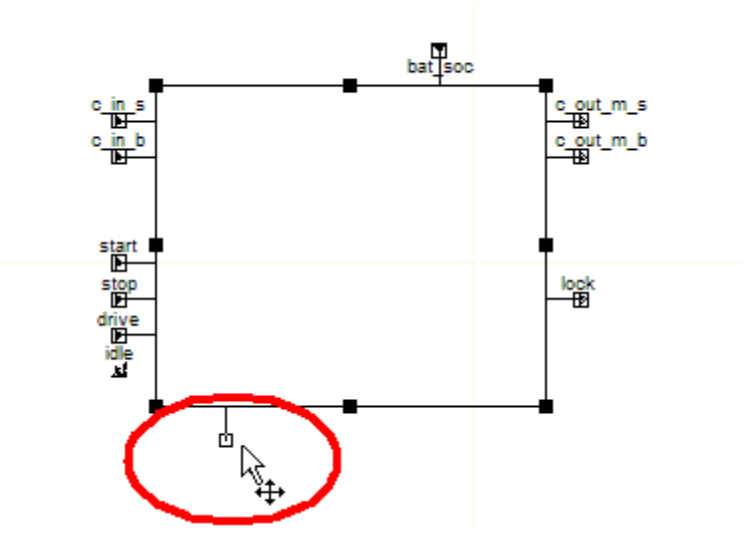
Adjusting Symbols and Pins

The **Adjust Symbol and Pins...** menu item appears in the context menu when a single component is selected and the component's symbol has a rectangle that matches the graphic extent such as those for auto-generated symbols, including those for coupling components and components representing subcircuits.

Use **Adjust Symbol and Pins...** to move unconnected pins around the circumference of the bounding rectangle. You can also resize the rectangle.

- You can move a pin to any vacant grid location (a grid location not occupied by any other of the selected component's pins). Its pin stem rotates depending on the side to which the pin is moved.

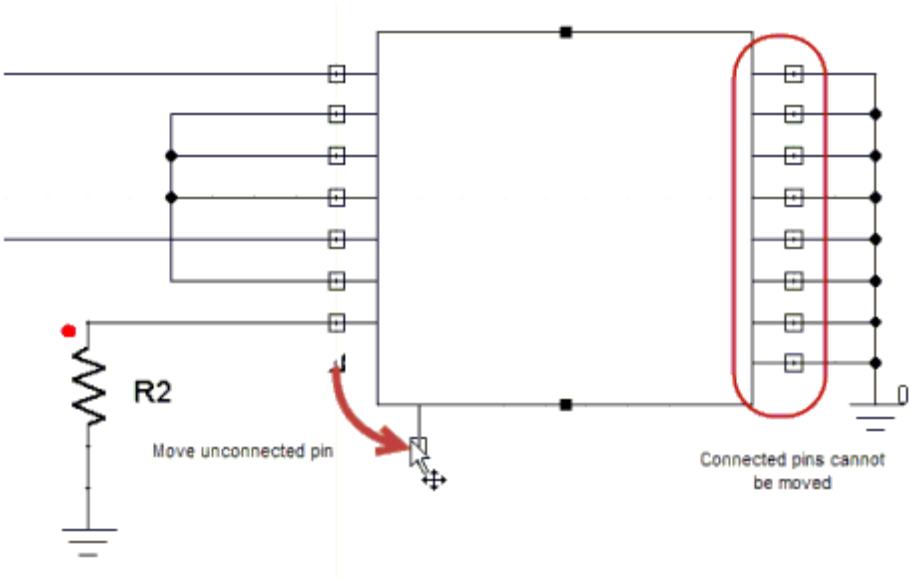
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



> **Note:**
>
> Pins with connections cannot be moved. The **Message Manager** pane displays a reminder if you attempt to move a connected pin.

- Moving an unconnected pin onto a wire or pin of another component and accepting the changes causes a connection to be made.
- With no pins connected, there are eight handles for resizing the rectangle. When the rectangle is resized, any unconnected pins will move proportionately, snapping to the nearest grid point.
- Connected pins constrain to how the rectangle may be resized. A connected pin on one side of the component prevents that side from moving (no handles appear on that side or its corners), though you can lengthen or shorten that side of the rectangle by moving a corner on the opposite side.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- If the symbol has a picture whose size matches the rectangle (such as for coupling components), the picture is resized as the symbol is adjusted.

To accept the changes, press the **Enter** key, or right-click and select either **Finish** or **Place and Finish**. To abort the command and discard changes, press either the **Backspace** or **Esc** key, or right-click and select either **Back** or **Cancel**.

The changes you make using this command are applied to a *copy* of the component's original symbol, which is used uniquely by the selected schematic component instance. To revert to the original symbol, right-click the component and select **Revert to default symbol**.

Related Topics

Using Pins

Connecting Components

Connect components on a sheet using wire mode, or by dragging one pin of a component over another pin of the same type.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Warning:

By default unconnected component pins (terminals) will generate errors (displayed in the **Message Manager** pane) when a netlist is generated. Simulation is not possible until the errors are corrected.

You can change this default component behavior Using the Component Editor so that either “No Action” is taken when unconnected terminals are netlisted; or that unconnected terminals are “Grounded”.

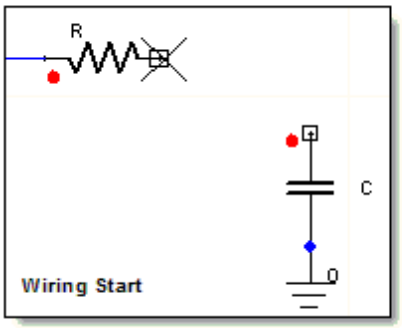
Note:

Direct connections between interface ports, or between an interface port and ground, are not allowed

Connections (wires) for the various domains are distinguished by default colors. For example, green for hydraulic, orange for magnetic, black for electrical, and so on. Connections between different domains are blue. These default colors can be set on the Colors tab in the Schematic Editor Options dialog box.

To start drawing a wire, do one of the following:

- On the desktop’s **Draw** menu, click **Wire**.
- Press Ctrl+W.
- Click the desktop’s **Wire** icon .
- Move the mouse cursor over a component pin to display the **X**-shaped wiring cursor:



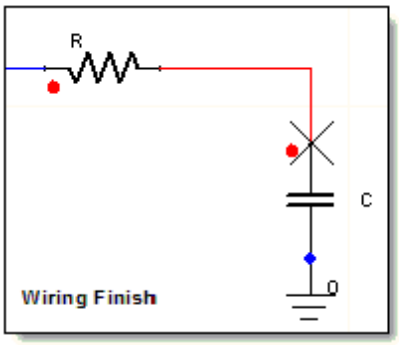
To draw connections:

- Click at the point where you want the wire to start, and move the cursor to extend the wire. As you move the cursor, click wherever you want to turn a corner.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To stop drawing a wire, do one of the following:

- Click, if the wire ends at a port or component pin.
- Double-click, if the wire does not end at a port or component pin.
- Press the spacebar to terminate the wire operation and leave the net unselected.
- Press Enter to terminate the wire operation and leave the net selected.
- Right-click and select **Place and Finish**.



Selecting a Wire

Although you may draw a wire that includes bends in a single operation, Twin Builder treats each portion of a wire between bends as a separate segment during selection operations.

- To select a single wire segment, click it.
- To select all segments of a wire (net), press and hold **Shift** and **Ctrl** and click any of its segments.
- To select multiple wire segments, contiguously or otherwise, press and hold **Ctrl** and click each segment in turn.
- To cancel a selection, click outside it in the schematic window.

Related Topics

Removing Unconnected Wires

Displaying Wire Properties

To display the properties of a wire, double-click it. Its **Properties** dialog appears.

NetName and **PinCount** are two wire properties of particular interest.

- **NetName** is the name of the unique interconnecting node, or *net*, a wire represents. NetNames correspond to the node names shown in the schematic's corresponding circuit file, or *netlist*.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **PinCount** reflects how many component pins a wire interconnects, with one exception: Port connections (interface and ground) are listed separately and thus are not included in PinCount values. A wire connecting a port and three component pins will therefore have a PinCount of 3.

Related Topics

Removing Unconnected Wires

Checking Connectivity

The Electric Rule Check (ERC) feature checks the circuit for valid connectivity. ERC conducts rule checking for ports, connections, dangling (unconnected) wire segments, and components of the active schematic.

1. To test for connectivity, select **Schematic** > **Electric Rule Check** to open the **Electric Rule Check** dialog box.
2. Select **Check subcircuits** to run the electric rule check on subcircuits of the active schematic display.
3. Click **Run ERC** to begin the error check.
4. If an error appears in the **Results** window, double-click the error message or select the message and click **Goto Error** to go directly to the object in the **Schematic Editor** that caused the error.
5. Click **Copy Results to Clipboard** to copy the **Results** text to the clipboard for pasting into another application such as an email program.
6. When finished, click **Close** to exit the dialog box.

Related Topics

Removing Unconnected Wires

Removing Unconnected Wires

In addition to deleting unconnected wire segments by selecting them individually, you can use **Remove Unconnected Wires** to remove all unconnected wires from a schematic.

- To remove all unconnected (“dangling”) wires and net segments from a schematic, select **Schematic** > **Remove Unconnected Wires**.

> **Note:** Merged nets (two or more nets with the same name) are not removed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Selecting a Wire

Displaying Wire Properties

Nets, Buses, and Bundles

A net or node is a single wire segment or a connected set of wire segments. A node name can contain any alphanumeric characters except the space ( ), ampersand (&), and asterisk (*). See ***Connecting Components***.

Buses and bundles are ways to name multiple related wires, pins, and ports conveniently. This section presents the rules for using bus and bundle wire names, and for the separation of sets of connected wire segments into different wires based on different names.

Bus Format

A bus is a collection of schematic wires that are indexes of a base name, for example, Data[0-31]. A bus is a schematic concept only. In the circuit and layout, all signals are individual.

A bus name consists of :

- A base name. A base name can contain any alphanumeric characters except the space ( ), ampersand (&), and asterisk (*).
- A square or angle open bracket ('[', '<']
- A number, a range of numbers specified with a hyphen or colon (*n1*-*n2* or *n1*:*n2*), or a comma-separated list of numbers or ranges.
- A square or angle close bracket

Here are two examples:

```
DATA[1:5] \\ A bus with five signals
inputbus[1,3-5,11-22] \\ A bus with 15 signals
```

Bundle Format

A bundle is a collection of schematic wires including individual wires and buses, for example, A,B,C[7-0]. A bundle is a schematic concept only. In the circuit and layout, all signals are individual.

A bundle name consists of a comma-separated list of single wire names and/or bus names. For example:

```
DATA[1-5],node5
```

Creating a Bus or Bundle

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To create a bus or a bundle, select **Draw** > **Wire** to create a wire; then change the wire's name to specify more than one signal, in either bus or bundle format. The width of the wire will change to a wide appearance when the number of signals is more than one.

- When a bus or bundle is copied and pasted, its name is lost and it becomes a single-signal wire.
- Drawing a wire by starting at a bus or bundle vertex will extend/add segments to the bus or bundle.
- Drawing a wire by ending on a bus or bundle will extend or add segments to the bus/bundle, unless the wire started at a named wire, bus or bundle. If it started at a named wire, that named wire will be extended, and it will connect to the bus or bundle if the bus or bundle contains the signal(s) in the named wire.

Deleting Wire Segments from a Bus or Bundle

Deleting one or more wire segments will result in the remaining segments of the wire being assembled into connected sets (based on physical and port connections) with each unique set being an independent wire. Only one of the sets will retain the original name. The others become single-signal wires with assigned names.

Naming a Wire in a Bus or Bundle

Selecting one or more connected segments of an existing wire and assigning a name results in the following:

- The wire is broken into sets as if the newly named segment(s) had been deleted. One set retains the original name. The selected segments become a new wire and are assigned the new name. The new name will spread to named connected segments, or those that have fewer signals than the new name, and those segments will be added to the new wire. Several different wires may result from this operation, each having a different name.
- Thus, renaming a segment of bus A[0-3] to A[7-0] will rename the entire wire. Renaming a segment of bus B[31-0] to B[0-63] with a sub-bus connected to it, B[16-31], will rename the connected segments of B[0-31] but will not rename the sub-bus, since it is a separate wire. Renaming the end segment of wire A[0-31] to A[0-15] will make that end segment into a sub-bus that is a different wire than A[0-31].
- If a wire with one or more pageports attached is renamed to a different width, the wire will follow naming rules, and any pageports physically attached will also be renamed.

Naming a Pin or Port on a Bus or Bundle

- The name of a pin or port can be a simple signal name, or can specify multiple signals through bus or bundle format.
- Changing the name of a port will change the name of attached wires.

Connections using Buses and Bundles

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Connections between wires, between pins and ports, and between wires and pins/ports depend on widths and names.
- Wire to wire connections are made if one wire contains the signals of the other. Thus, a wire A[0-7] would have 4 signals electrically connected to physically connected wire A [7-4], namely A[4], A[5], A[6] and A[7]. A bundle A,B,C would have one signal electrically connected to physically connected wire A. A wire A,B,C,D would have two signals connected to physically connected wire C,D,E,F. (See ***Drawing Bus Entry Objects*** for information on adding visual representations of bus connections between a bus and individual wires.)
- Wire to pin or port connections are determined by the number of signals in the wire and pin/port. If the numbers of signals are the same, electrical connections are made in order. If the widths are different, there is no connection.
- Pin/port to pin/port connections are identical to wire to pin/port connections. If the widths are the same, all signals are connected in order. If the widths are not the same, none are connected.
- Changing the name of a wire that is attached to a pin, a port, or another wire may cause connections to be made or broken, based on the considerations above.

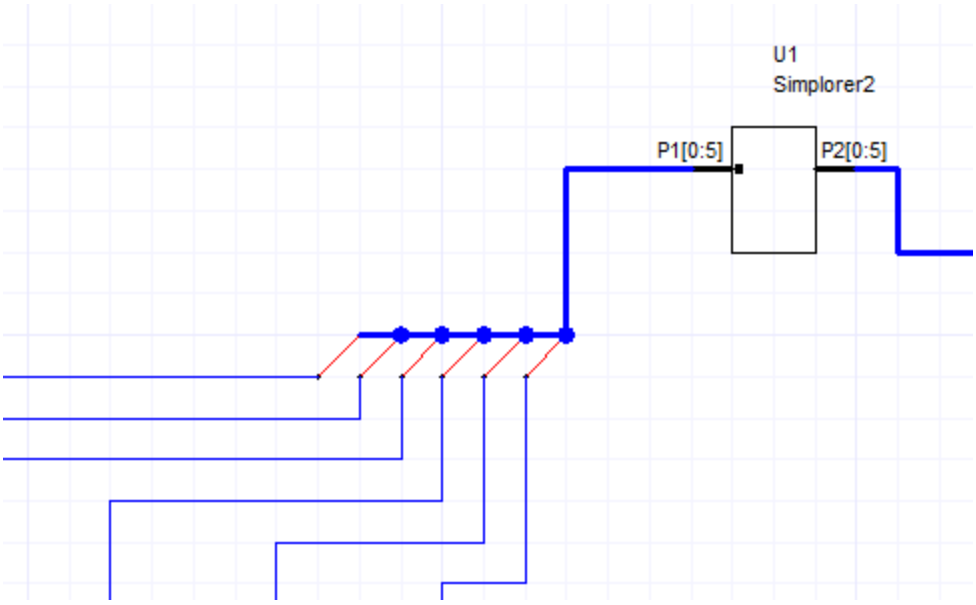
Grounding Buses and Bundles

A bus or bundle can be grounded by renaming it with zeros. For example, to ground a bus named A[0:2], you would name it as 0,0,0.

Drawing Bus Entry Objects

Select **Draw** > **Bus Entry** to add *visual representations* of connections from a bus to an individual net or a sub-bus. On a schematic, a bus may be shown with individual nets (wire) or with wires where the bus width is greater than 1 (drawn with a thick blue line). The **Bus Entry** object provides a mechanism for *visually representing* individual wires connected to a bus consisting of multiple nets as shown in the following example. In this example, the P1[0:5] bus is visually “connected” to individual wires via the bus entry objects (the diagonal red lines).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

- **Bus Entry** objects provide only a *visual representation* of connections on the schematic - not actual connections. You still must ensure that connectivity is maintained by properly naming the wire segments drawn out of a bus.
- Because they are merely visual representations of connections, **Bus Entry** objects have no significance or effect on the actual circuit, and thus do not appear in the net list or in any results.
- **Bus Entry** objects are used to show connections to nets of any width and any domain.
- Both ends are only attached to nets and cannot be directly attached to other pins or ports.

To add bus entry objects to an existing schematic containing a bus:

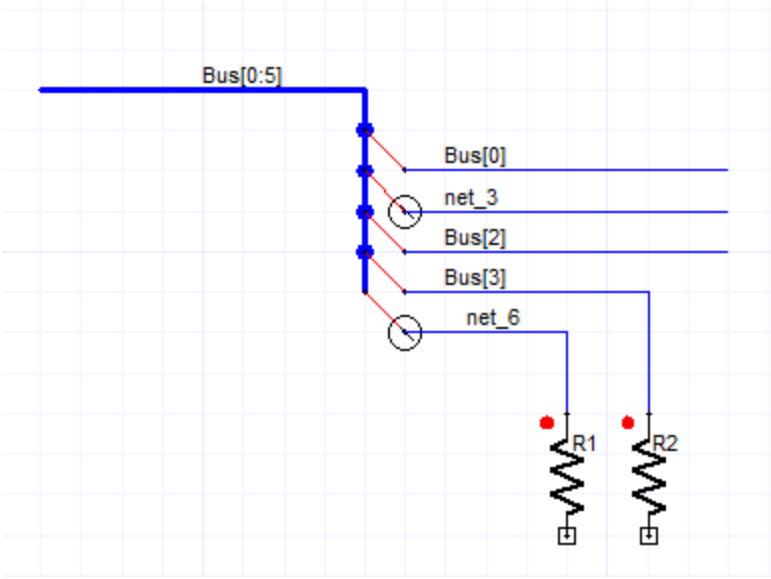
1. Add a bus to the schematic by drawing a wire and naming it as a bus. For example: **DataBus[0:2]**.
2. Place a **Bus Entry** from the Draw menu such that one end intersects the bus.
3. Draw another wire from the other end of the Bus Entry object.
4. Rename this new wire as **DataBus[0]** to use that element of the bus as the individual net.
5. Repeat as needed for the remaining bus elements.

Disconnects

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

A disconnect is a circular visual indicator at the pin of a bus entry object. Its presence indicates that two nets – though visually connected through the bus entry object – are actually disconnected as indicated by the differing names.

If a bus entry has both pins connected to two different nets (which can be of different sizes) and if neither of the named nets are a subset of the other named net, a disconnect circle will be drawn.



The figure above shows a bus named Bus[0:5], which has five nets that are branched out through bus entry objects. Three nets are named properly as: Bus[0], Bus[2] and Bus [3]. However two nets: net_3 and net_6 are not connected to the bus by name - and thus are shown drawn with circular disconnect objects.

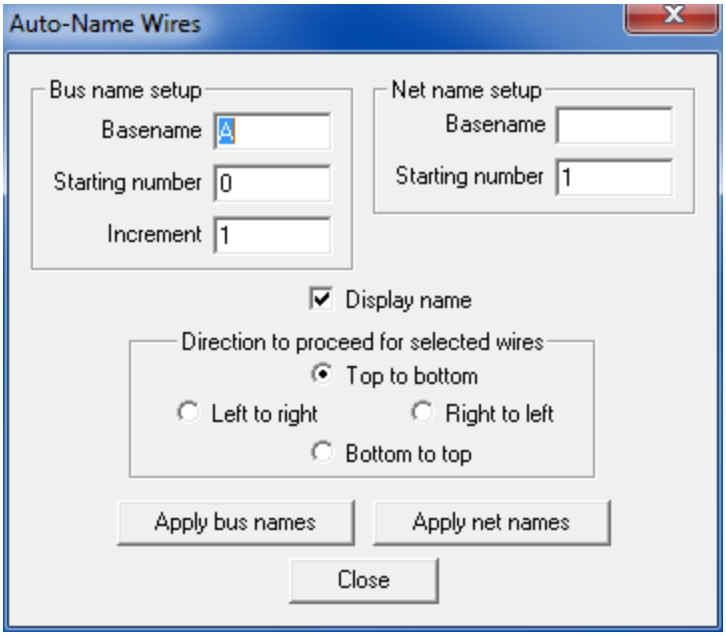
Net and Bus Auto-Naming

Selected wires can be named as a group with either bus-style names (for example, A[0]) or net-style names (for example, W1), using the **Auto-Name Wires** dialog.

1. In the Schematic Editor, select the wires to be auto-named.
2. Click **Schematic** in the top menu bar and select **Auto-Name Wires** .

   The **Auto-Name Wires** dialog box opens.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



3. Adjust the **Bus name setup** or **Net name setup**, depending on which naming style you want to use.
4. Toggle **Display name** off if you do not want the wire names to display on the schematic.
5. Select the direction for applying the names to the selected wires.
6. Click **Apply bus names** or **Apply net names** to set the names of the selected wires.
7. Click **Close** to close the dialog box.

Placing Ports

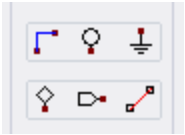
The following types of ports are available in Twin Builder: *interface*, *ground*, *global*, and *page*.

- An interface port serves as a connector into or out of a given design, and may contain termination and signal source definitions. The names of interface ports must be unique within a given design but may be duplicated from design to design.
- All ground ports connect to the reference or ground node, node 0.
- A global port serves as a means of common connection – in effect, a connection to a bus – within a given design, across hierarchical levels. You can define one or more circuit nodes to connect to global ports that cross hierarchy and schematic page boundaries. Within a design, all global ports with the same name are treated as if they are connected. The nature of the port, electrical, fluidic, thermal, and so on, is determined by the first connection made to the port. Once a global port has a specific nature, additional connections in the same or other levels of hierarchy can only be made to pins of that nature.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- A page port (connector) serves as a named connection to a signal that is common to two or more pages of the schematic.

The ports described in the next several topics are available from the **Draw** menu of the Schematic Editor, and via icons under **Graphics** on the **Schematic** ribbon:



Interface Ports

Ground Ports

Global Ports

Page Connector

Interface Ports information

Interface ports provide connectivity between different levels of hierarchy and serve as named connectors in or out of a design. The interface port parameters specify the port name, its impedance, a reference node, and (optionally) one or more power, current, or voltage source specifications.

An interface port can have a source associated with it that provides excitation when the parent circuit is not part of the simulation. When a parent is simulated, port sources in a subcircuit have no effect and communication occurs between the parent and the child through the port. When the subcircuit is simulated directly, rather than as part of a parent simulation, excitation at ports (if any) is provided by port sources.

Similarly, a conservative interface port may have a sink associated with it. This may be a ground reference or an impedance (to ground). These have no effect when a parent is simulated, but are part of direct simulation of the subcircuit.

For conservative ports, the available sources and impedances depend on the port's nature. See ***Twin Builder Options: Port Options Tab*** for detailed information on setting up lists of port source and impedance components.

For non-conservative ports in a subcircuit, the symbol pin for the component shows the type as Input or Output.

Related Topics

Renaming an Interface Port

Setting Interface Port Properties

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Placing Interface Ports

To place an interface port, select the schematic that will contain it. Then do one of the following:

- On the **Draw** menu, click **Interface Port**.
- On the **Schematic** ribbon, click the **Interface Port** icon .

The cursor, now associated with an interface port symbol for placement, moves to the center of the schematic window. To place the port, click at the desired location.

Hint	You can rotate an interface port *before placing it* by repeatedly pressing **R** on your keyboard. Each press rotates the port 90º counterclockwise. After placement use **Ctrl+R** to rotate the port.

If Multiple Placement is turned on for interface ports in the Schematic Options dialog box, click additional locations to place additional ports. To stop placing interface ports, do one of the following:

- Press **Enter**, the **SPACEBAR**, or **Esc**.
- Right-click and select **Place and Finish**, **Finish**, or **Cancel**.

> **Note:**
>
> To ensure electrical connectivity among schematic elements, the pins of placed interface ports snap to a 100-mil (2.54-millimeter) grid. This snapping cannot be turned off, and the spacing of the connectivity grid cannot be adjusted.

> **Note:**
>
> Direct connections between interface and/or global ports, or between an interface or global port and ground, are not allowed. Attempting to do produces cautionary messages in the **Message Manager** pane.

> **Note:**
>
> Unconnected interface ports adjust to the domain they are being connected to in the subcircuit. However if the corresponding top-level pins in the parent circuit are wired, then automatic adjustment does not occur, in which case domain conversion is applied instead.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding Interface Ports at Pins

You can add an interface port directly to a component's unconnected pins. Right-click a component and select **Add at unconnected pins** > **Interface Ports**.

Alternatively, select **Draw** > **Add at unconnected pins** > **Interface Ports** to add interface ports to the currently selected components.

Related Topics

Renaming an Interface Port

Setting Interface Port Properties

Renaming an Interface Port

Each interface port you place also appears as a Port object in the Ports folder under the selected design's icon in the Project tree. The name of the port is the same as the name of the node or net to which it is connected. Node names of the form "net_*n*" generate during the wiring operation, but they can be renamed.

> **Note:**
>
> Do not use definition properties such as *ID*, *Representation*, and *Owner* as the names for ports.

- To rename an interface port, select it by doing one of the following:
  - Right-click its icon in the Project tree and select **Edit**.
  - Right-click its schematic symbol and select **Properties** to open its Properties dialog box.
  - In the Properties window, Click the port name button in the **Port Name** field to open the **Interface Port Name** dialog box. Enter the desired name (including a bus name) and click **OK**. The Properties window reflects the new name. Note that the port number (pnum) does not change.
- If the port is placed so that it connects to an existing node, the node is renamed to the name of the port (even if the node was manually renamed by the user). You can also change a port name attached to a node. Select the node and click the node name in its property window. A **Net Name** dialog box similar to the one above opens for you to specify a new name. Changing the net name changes the names of any ports attached to that node.
- If you name a port the same as an existing net, the two nodes are merged into a single node with that name. A dialog box appears for you to confirm the merge operation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- If you delete an interface port, the node to which it was attached is renamed to a system-defined name (“net_*n*”).

Related Topics

Nets, Buses, and Bundles

Setting Interface Port Properties

1. Select the port and either double-click the port symbol, or right-click the port symbol and select **Edit Port**. you can also right-click the port icon in the Project tree and select **Edit**.

   The **Port Properties** dialog box opens.

2. You can specify a **Port Name** (including a bus name), and set **Terminal Attributes** by selecting from the following drop-down menus:
   - **Domain** - Conservative, or the following non-conservative domains, where no physical conservation laws apply to the interface: Quantity, Parametric, Signal.
     - Conservative - where laws of conservation of the physical domain of interest are respected and the value can change during a single simulation.
     - Quantity - typically should be used for a non-conservative, real valued interface port for which the value can change during a single simulation.
     - Parametric - typically used in real valued ports to pass design parameters. It is non-conservative and the value remains constant during a single simulation.
     - Signal - typically should be used for a non-conservative, language-specific enumerated (fixed set of values) interface port for which the value can change during a single simulation.
   - **Type** – Conservative **Domain** types (or natures): Electrical, Magnetic, Fluidic, Translational, Translational_Velocity, Rotational, Rotational_Velocity, Radiant, Thermal, Compressible_Fluidic.
   - Quantity or Parametric **Domain** type: real.
   - Signal **Domain** types: bit, std_logic.
   - **Direction** – Present when the **Domain** is Quantity, Parametric, or Signal. Choices are: In, Out, InOut. The interface port symbol contains a red-filled triangle for In, a blue-filled triangle for Out, and both red and blue-filled triangles for InOut.
   - **Port Value** – Present when the **Domain** is Quantity, Parametric, or Signal. Enter a port value.
   - **If Unconnected** – Present when the **Domain** is Conservative. Choices are: No action, Flag as error, Unique net, Grounded.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

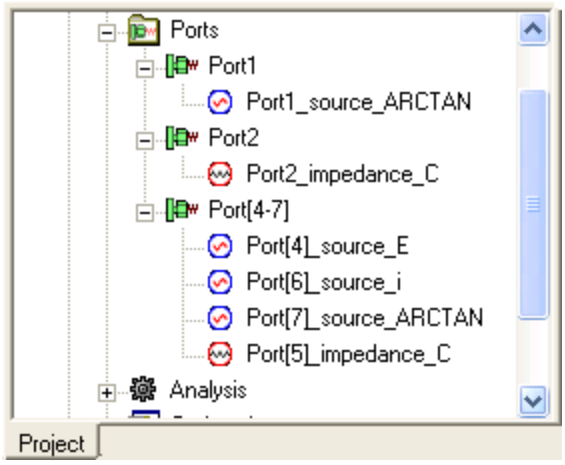
If the Port Name designates it as a bus port, you can also choose:

- **Common Attributes** – when checked, assigns the same **Terminal Attributes** and Port Source or Impedance to all **Bus Elements**.
- **Bus Element** - drop-down list allows unique **Terminal Attributes** and Port Sources or Impedances to be assigned to individual bus elements.

3. You can specify a **Port Source or Impedance** for conservative ports by selecting the appropriate radio button: **None**, **Source**, or **Impedance**. Similarly, you can specify a **Port Source** for non-conservative ports by selecting the **Source** radio button.
    - The **Source** drop-down contains a list of source components appropriate to the nature (shown in the **Terminal Attributes Type** field) of the port.
    - Similarly, the **Impedance** drop-down shows appropriate impedance components.
    - The lists of available source and impedance components are set up in the **Tools>Options>General Options>Twin Builder>Port Options**. For more information, see **Port Options**.
    - Click **Properties** to open the special component dialog for the selected component, allowing complete specification of the element's properties.

The instance name of a port source has the form: <*port name*>_source_<*comp name*>. For example, if you place an interface port named **Port2** and select a voltage source **E** as its port source, the instance name for this port source would be: **Port2_source_E**. Similarly, the name of an port impedance would be <*port name*>_impedance_<*comp name*>. Instance names for port sources and impedances cannot be modified, and the Output/Display tab on the special component dialog box is disabled.

Icons for defined port sources and impedances appear in the Project Manager tree under the ports with which they are associated.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Double-click a port source or impedance icon to open the associated component dialog box for editing. You can also right-click an icon if you want to delete the instance.

4. When finished setting port properties, click **OK**.

Related Topics

Nets, Buses, and Bundles

Twin Builder Options: Port Options Tab

Global Ports

A global port serves as a means of common connection – in effect, a connection to a bus – within a given design, across hierarchical levels. You can define one or more circuit nodes to connect to global ports that cross hierarchy and schematic page boundaries. Within a design, all global ports with the same name are treated as if they are connected.

The nature of the port, electrical, fluidic, thermal, and so on, is determined by the first connection made to the port. Once a global port has a specific nature, additional connections in the same or other levels of hierarchy can only be made to pins of that nature.

> **Note:**
>
> Direct connections between interface and/or global ports, or between an interface or global port and ground, are not allowed. Attempting to do produces cautionary messages in the Message Manager.

- Bringing in a model with global variables or ports will generally add those variables or ports to the project or design, unless they are already present. If a model's global port has one nature and a same-named global port in the target design has another, the model (and referencing component) will be rejected with an error message.
- Similarly, pasting a design into another may cause a similar conflict and will be also rejected with an error message.

To place a global port, select the schematic that will contain it. Then do one of the following:

- On the **Draw** menu, click **Global Port**.
- On the **Schematic** ribbon, click .

The cursor, now associated with a global port symbol for placement, moves to the center of the schematic window. To place the port, click at the desired location.

Hint	You can rotate a global port *before placing it* by repeatedly pressing **R** on your

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	keyboard. Each press rotates the port 90º counterclockwise. After placement use **Ctrl+R** to rotate the port.

If Multiple Placement is turned on for global ports in the Schematic Options dialog box, click additional locations to place additional ports. To stop placing global ports, do one of the following:

- Press **Enter**, the **SPACEBAR**, or **Esc** on your keyboard.
- Right-click and select **Place and Finish**, **Finish**, or **Cancel**.

> **Note:**
>
> To ensure connectivity among schematic elements, the pins of placed global ports snap to a 100-mil (2.54-millimeter) grid. This snapping cannot be turned off, and the spacing of the connectivity grid cannot be adjusted.

> **Note:**
>
> Direct connections between interface ports and/or global ports, or between a global port and ground, are not allowed.

Renaming a Global Port

Each global port you place also appears as a Port object in the Ports folder under the selected design's icon in the Project tree. Port names of the form **G_*n***, where ***n*** is an integer, are generated when placing them; but they can be renamed.

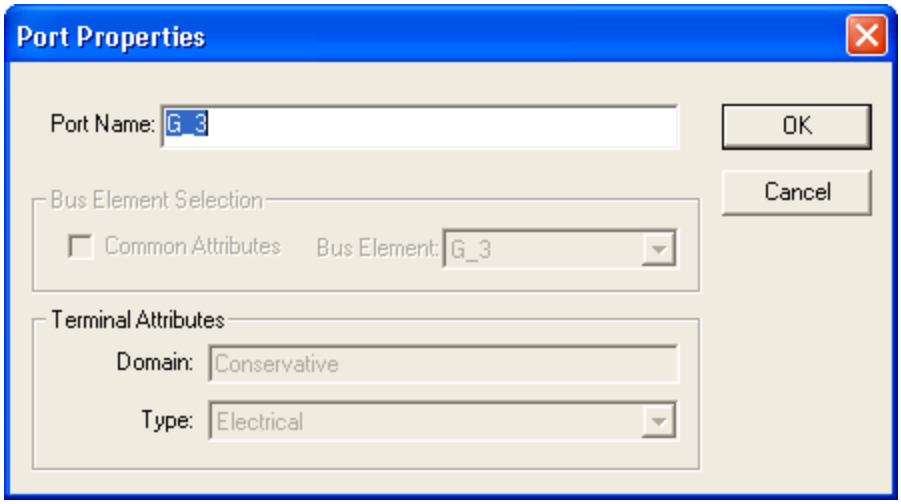
The name of a global port cannot be changed to **0** (which is reserved for ground ports) and the name of a ground cannot be changed to something else. A ground port is a special example of a global port, and it serves as reference for any or all natures.

To rename a global port, select it by doing one of the following:

- Right-click its icon in the **Project** tree and select **Edit**.
- In the Properties window, click the port name button in the **Port Name** field to open the **Global Port Name** dialog box. Enter a name and click **OK**. The Properties window reflects the new name.
- Right-click its schematic symbol, and select **Properties** to open a **Port Properties**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

dialog box.



The terminal attributes of the global port shown in the dialog are determined from the connections and are provided for informational purposes.

Page Connector

A page connector serves as a named connection to a signal that is common to two or more pages of the schematic. Page connectors thus serve as graphic reminders that a signal may be present on elsewhere on the page, or on another page. Any node with a given name (system-generated or user-defined) is connected across all pages that reference it.

> **Note:**
>
> When translating legacy schematics having wires that are too close to pins to which they are not connected, page connectors are added to the pins on the translated schematic to maintain proper connectivity.

To place a page connector in the schematic:

1. Click on the **Schematic** ribbon, or select **Draw** > **Page Connector**.

   The cursor, carrying a page connector symbol for placement, can be moved to the schematic window.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. To place a page connector, click the desired location on the schematic.

   > **Note:**
   >
   > Press R to rotate a page connector *before placing it*. Each press rotates the page connector 90º counterclockwise. Press Ctrl+R to rotate the page connector after placement.

   By default, the port is named **Pageport_***n*, where *n* is an arbitrary integer.

The name of the connector is the same as the name of the node or net to which it is connected. Node names of the form **net_***n* are generated during the wiring operation; select the wire and click the **PortName** in the **Parameter Values** tab to rename it. If the connector is placed so that it connects to an existing node with a system-assigned name (**net_***n*), the node is renamed to the name of the port. If you manually named the node, the node retains its user-defined name and the page connector gets that name.

> **Note:**
>
> If you paste an existing page port to an existing net, the page port takes the name of the net.

To rename a connector (and the node to which it connects):

1. Click the connector to select it and view its properties in the property window.
2. Click the connector name in the **PortName** field to open the **Page Port Name** dialog box.
3. Enter the desired name and click **OK**.

   The **Properties** dialog box reflects the new name.

You can also change a connector name attached to a node. Select the node and click the node name in its property window. A **Net Name** dialog box opens for you to specify a new name. Changing the net name changes the names of any connectors attached to that node.

If you name a connector the same as an existing net, the two nodes are merged into a single node with that name. A dialog box appears for you to confirm the merge operation.

Adding Page Connectors at Pins

You can add a page connector directly to a component's unconnected pin. Right-click the pin and select **Add at unconnected pins** > **Page Connectors**.

Alternatively, select **Draw** > **Add at unconnected pins** > **Page Connectors** to add page connectors to the currently selected components.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ground Ports

A ground port serves as a connection to the reference or ground node, node 0 in Twin Builder circuit designs. Ground ports cannot be renamed.

To place a ground:

1. Select the schematic that will contain it.
2. On the **Schematic** ribbon, click the **Ground** icon . Alternatively, click **Ground** on the **Draw** drop-down list.

   The cursor, carrying a ground symbol for placement, can be moved to the schematic window.
3. To place a ground, click the desired location on the schematic.

   > **Note:**
   >
   > You can rotate a ground *before placing it* by repeatedly pressing **R** on your keyboard. Each press rotates the ground 90º counterclockwise. After placement use **Ctrl+R** to rotate the ground.

   Twin Builder supports multi-domain grounds for conservative pins and nets. Based on the type of net connection, Twin Builder changes to the appropriate ground symbol and net (wire) color. For example, the net (wire) color for hydraulic connections is green and the default hydraulic ground symbol is .

   If two domains are merged on a common ground, the net connection defaults to the normal blue color and the default ground symbol .

   Net connection colors and ground symbols revert to the appropriate states when the merged domains are separated by deleting the connecting wires.

If **Multiple Placement** is turned on for grounds in the Schematic Options dialog box, click additional locations to place additional grounds.

To stop placing grounds during multiple placement, do either of the following:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

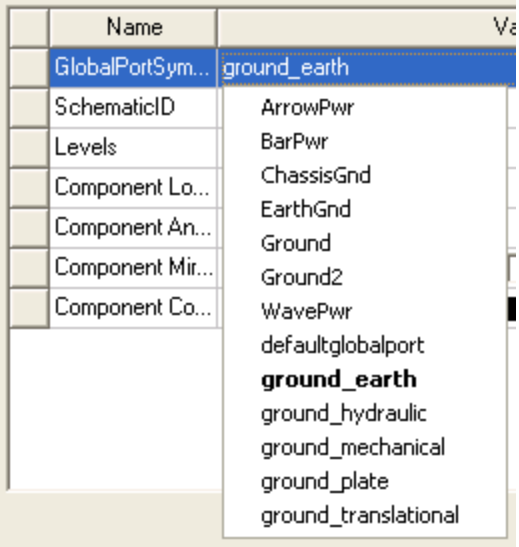
- Press **Enter**, the **SPACEBAR**, **Backspace**, or **Esc** on your keyboard.
- Right-click and select **Place and Finish**, **Finish**, **Cancel**, or **Back**.

> **Note:**
>
> To ensure electrical connectivity among schematic elements, the pins of placed grounds snap to a 100-mil (2.54-millimeter) grid. This snapping cannot be turned off, and the spacing of the connectivity grid cannot be adjusted.

Selecting a Ground Symbol

Several symbols are available to represent a ground node. To select a symbol other than the default:

1. Double-click the ground symbol to open the **Properties** dialog box.
2. Select the **Symbol** tab.
3. Click the **Value** field for **GlobalPortSymbol** to display a drop-down menu with the symbol choices.



4. Select a symbol and click **OK** to close the **Properties** dialog box.

   The new symbol is displayed in the schematic.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> All of the global port symbols except the **defaultglobalport** symbol can be edited in the Symbol Editor (part of the Library Editor).

Adding Ground Connectors at Pins

You can add a ground connector directly to a component's unconnected pin. Right-click the pin and select **Add at unconnected pins** > **Grounds**.

Alternatively, select **Draw** > **Add at unconnected pins** > **Grounds** to add grounds to the currently selected components.

How Ports Affect Node Names

When you connect a port to a node or net, the net name changes to that of the port. The rules for net renaming depend on the kinds of ports attached to the node. The precedence for port types is global ports > interface ports > page ports.

Global Ports

- If a global port is attached to a net, the net name is the global port name, and all connected page ports have that name as well.

Interface Ports

- If an interface port is attached to a net, the net name is the interface port name, and all connected page ports have that name as well.
- If more than one interface port is attached to a net, the net name is the first interface port name.
- If you delete a wire connecting an interface port to a net (no page port connected), the interface port retains its name and the rest of the net is assigned a name.

Page Ports

- If a page port is attached to a net (but no interface ports), the net name is the page port name.
- If more than one page port is attached to a net, they will all share the same name.
- If you delete a wire connecting a page port to a net (no interface ports connected), the page port retains its name and the rest of the net is assigned a name.
- If you delete a wire that connects an interface port and connected page port to a net, the ports retain their names, and the rest of the net is assigned a name.
- If you move a page port (interface port connected) and break the page port's connection to a net using **Ctrl+Shift**, the interface port retains its name and the page port is assigned a name.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding Ports

- If a global port is added to a net, the name of the net and page ports on the net are changed to the global port name.
- If an interface port is added to a net the name of the net and page ports directly connected are changed to the interface port name.
- If a page port is added to a net (without any interface ports) the name of the net and pageports directly connected are changed to the page port name.

Changing Node Names

You can change the name of a node (net) in the schematic. Double-click a node to open its **Properties** dialog box, then click the **Value** field for the **Name** property. Click **OK** to apply the new name.

Here are some guidelines to keep in mind when changing the name of a node:

- Net names may not contain spaces, ampersands (&), or asterisks (*).
- When you try to change an interface port name to that of an existing net, the existing net of the same name as the new port name will be connected to the new port's net. A dialog box appears so that you can resolve the name conflict by splitting the net into two nets, or allowing the nets to remain connected.
- When you try to change an interface port name to that of an existing net, the existing net of the same name as the new port name connects to the new port's net. A dialog box appears so that you can resolve the name conflict by splitting the net into two nets, or by allowing the nets to remain connected.
- When you try to change a page port name to that of an existing net, the existing net of the same name as the new port name will be connected to the new port's net. A dialog box appears so that you can resolve the name conflict by splitting the net into two nets, or allowing the nets to remain connected.
- If an interface port name is changed, the name of the net it is attached to and any connected page ports are changed.
- If a global port name is changed, the name of the net it is attached to and any connected page ports are changed.
- If a page port name is changed, the net is examined to see what is physically connected. Wires and pins that are directly attached to the renamed page port are connected on the net with the page port. If there is another net with a page port of the new name, the page port and directly connected objects are connected to that net. If there is no such other net, the page port and directly connected objects become their own net.
- If a net name is changed, the ports on the net that had the same name as the old net name will be changed to the new name.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The Effect of Deleting Ports

- If a global port is deleted from a net, the name of the net is determined by the remaining global or interface ports, or becomes a new unique name.
- If an interface port is deleted from a net, the name of the net is determined by the remaining interface ports, or becomes a new unique name.
- If a page port is deleted from a net, the name of the net is determined by the remaining interface ports, or becomes a new unique name.

Creating Twin Models

Follow these steps to create a Twin model using the schematic editor.

> **Note:**
>
> An example of this procedure is located in **AnsysEM***version***\Win64\Examples\Twin Builder\Applications\Twin Generation\CoupledClutches.aedt**.

> **Note:**
>
> Only subcircuits that contain one or more of the following components can be compiled into a Twin model:
>
> - FMU
> - Modelica
> - Static ROM
> - Dyna ROM
> - UnitDelay (located in the **Simplorer Elements\Basic Elements VHDLAMS\Blocks\Discrete Blocks** library)

1. Open a Twin Builder design.
2. Open the top level design in the schematic editor.
3. Select the desired subcircuit design instance on the schematic that you want to compile as a Twin component.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

When compiling a Twin, you must break any algebraic loop formed when a feedback loop exists between components on the subsheet.

The **UnitDelay** component holds and delays its input by a sample time parameter (**ts**) that you specify.

Follow these steps to add a unit delay component to a feedback loop.

a. Open a Twin Builder design.

b. Go to the Component Library and find the **udelay** component (the Unit Delay Model in VHDL-AMS under **Simplorer Elements\Basic Elements\VHDLAMS\Blocks\Discrete Blocks**).

c. Select the **udelay** library and create an instance on the subsheet where a feedback loop exists.

d. Connect the pins of the **udelay** instance to the pins of the feedback loop.

e. Set the sample time parameter (**ts**) according to your transient analysis; for example, a **ts** equal to the time step at which your twin is being simulated.

4. Right-click the selected subcircuit instance and select **Compile As Twin Model**. The **Compile Twin Model** dialog box appears.
5. In the **Compile Twin Model** dialog box, you can change the Twin model name and various model-dependent options (solver method, step size, relative and absolute tolerance).

Note:

If compiling an FMU, the full path and file name must not exceed 256 characters or the compilation will fail. If the Twin model did not generate, check your local **TEMP** folder to see if the **dtrg_error_*.log** file generated. This log can help identify and explain common errors in Twin compilation.

6. Click **Compile** to compile the subcircuit as a Twin model. which you then add to the project under **Project definitions** > **Components**. It will be also added to mouse cursor ready to be placed anywhere on the schematic.

Recompiling and Update Existing Twin Component

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In the **Compile Twin Model** dialog box, use the previously created Twin model name to update the existing Twin component.

Related Topics

Exporting a Twin Model as a Twin

Creating and Exporting Co-Simulation FMU Models

Use the co-simulation FMU functionality to export a Twin Builder design in a Subsheet as an FMU. You can import FMUs with any tool that supports FMI co-simulation version 2.0. For the simulation, the importing tool is the primary solver, while the Twin Builder solver in stand-alone mode is the secondary solver.

Follow these steps to create and export an FMU model using the schematic editor.

1. Open a Twin builder design.
2. Open the top-level design in the schematic editor.

   > **Note:**
   >
   > Transient (TR) analysis settings (Hmin, Hmax, and Tend) and the TR options (integration formula, tolerances, and so on) from this top level will be applied when the co-simulation is running. Make sure these values are appropriate for the co-simulation run before compiling the FMU.

3. Select the desired subcircuit design instance on the schematic that you want to compile as an FMU component.
4. Right-click the selected subcircuit instance and select **Compile as Co-Simulation FMU**. The **Compile as Co-Simulation FMU** dialog box appears.
5. In the **Compile as Co-Simulation FMU** dialog box, enter the name of the FMU as the model name (the default name is **FMUModel**).
6. Specify whether the exported FMU should check out a runtime license at execution time.

   > **Note:**
   >
   > If the FMU is exported as an unlicensed FMU, a separate export license is required. The runtime licenses for the licensed FMU or the export licenses must be purchased separately.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7. Select the dependent data and binary files to include in the FMU.

> **Note:**
>
> When creating an FMU for co-simulation, Twin Builder includes all necessary dependent binary and data files in the FMU's resource folder. This includes binaries for the solver as well as binaries and data files for C-Models or sub-FMU models.

   a. Select **AutoDetect** if the subsheet to compile as a co-simulation FMU contains only models from system libraries, C-Models, FMU models, or any other model definitions that do not load binaries or data files manually.
   b. Select **Complete** if the subsheet to compile as a co-simulation FMU contains user-defined models that manually load any binaries or data files from the **userlib** or **personallib** folders. By default, all files from the **userlib** and **personallib** folders are included in the FMU. Instead of including the files from the standard **userlib** and **personallib** folders in the FMU, you can specify alternate locations in a configuration file. Include these lines:

```
[Windows]

UseAltLibDirs=<Yes/No>

AltUserLibDir=<Alternate UserLib folder for Co-Simulation>

AltPersLibDir=<Alternate PersonalLib folder for Co-Simulation>
```

> **Warning:**
>
> If you need to share the FMU with someone else when selecting this option, make sure the library folders do not contain any files with sensitive data such as intellectual property. In such situations, either remove those files, or configure an alternate **userlib** and **personallib** folders without any sensitive data.

8. Click **Compile** to compile the FMU model. The model is then added to the project **Definitions**.
9. Under the project **Definitions** > **Models**, right-click the compiled model and select **Export as FMU** to export it to a desired location.

Creating Cross-Platform Co-Simulation FMUs

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For the creation and simulation of cross-platform (Windows64/Linux64) co-simulation FMUs, download the Twin Builder Linux Redistributable package from the Download area in the Ansys Customer Portal. Select the release version accordingly to your Twin Builder release version and the Linux x64 operating system. The package can be found under the Add-On Packages section of the page.

After downloading and extracting the package, follow the provided documentation for a comprehensive guideline on how to create a Linux-compatible co-simulation FMU, as well as how to simulate it on a Linux environment.

1. In the **Compile as Co-Simulation FMU** dialog box, set **Include Linux Files** to **Yes**.
2. Specify the location where you installed the downloaded Linux package in **Linux Solver Folder**.
3. If applicable, specify the locations for the Linux versions of model DLLs of **PersonalLib** and **UserLib** in the fields for **Linux PersonalLib Folder** and **Linux UserLib Folder**.

Component Support

Not all the components in a Subsheet can be supported in the exported FMU. Mainly VHDL models, co-simulation components, and dynamic components that need to connect/simulate with other Ansys Electronics Desktop products before launching Twin Builder simulation are not supported. The following table provides the details about component support for Windows and Linux platforms.

Component type	Windows support	Linux support	Remarks
Internal elements	Yes	Yes	
SML models	Yes	Yes	All SML models including generated SML models such as state-space models, Maxell Equivalent Circuit (ECE model), Ansys Mechanical, Icepak, PExprt Static component, Fluent LTI components are supported.
VHDL	No	No	
C-Models	Yes	Yes	For user-defined models, dependencies should be placed in appropriate directories.
FMU	Yes	Yes	For Linux, FMU must be cross-platform (support both Windows and Linux).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Modelica	Yes	No	For Linux, Modelica model must be compiled as a cross platform FMU.
Co-simulation components	No	No	Components such as Simulink co-simulation, Maxwell transient co-simulation, Fluent, and RBD are not supported.
Dynamic components	No	No	Components such as HFSS, Q3D, SIWave, and Maxwell Magnetostatic, Electrostatic and Eddy Current are not supported.
Twin Builder ROM components	Yes	Yes	Visualization is not supported for ROMs.
Data connector	No	No	
Python component	No	No	
Excitation component	No	No	

> **Note:**
>
> Even though Simulink co-simulation is not supported, Simulink Coder generated Twin Builder C-models are supported.

Exporting a Subsheet Containing any Modelica Model as a Linux Compatible Co-Simulation FMU

If a Subsheet contains any Modelica model, perform these operations to replace the Modelica model with a cross-platform FMU:

- Export the Modelica model as an FMU (this will be a Windows only FMU).
- Copy the FMU to a Linux machine with Ansys Twin Deployer installed.
- Recompile the FMU as a cross-platform FMU and copy it back to the Windows machine.
- Import the recompiled FMU to the Subsheet and replace the Modelica model with this FMU.

For more details on compiling a Modelica model as a cross-platform FMU, see the documentation for Ansys Twin Deployer.

Limitations

Subcircuits that satisfy one or more of these conditions cannot be compiled into an FMU model:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Subsheets with neither parameters nor quantities being exposed.
- Subsheets with conservative terminals.
- Subsheets with bus ports.
- Subsheets with components that are in an inactive/disabled state.
- Subsheets that contain unsupported components.

Recompiling and Updating the Existing FMU component

In the **Compile as Co-Simulation FMU** dialog box, use the previously created co-simulation model name to update the existing FMU component.

Running a Simulation Using the FMU

The Twin Builder solver checks out a license during the co-simulation run. Set the following environment variable before launching the importing tool to specify the host name and the port number of the Ansys license server:

```
ANSYSLMD_LICENSE_FILE=<port>@<host>
```

Step-Size Under Co-Simulation Mode

Under the co-simulation mode for an applied communication step-size, *hc*, at the current communication time tc_i, Twin Builder may take multiple smaller step sizes to reach the next communication step, tc_{i+1} (where $tc_{i+1} = tc_i + hc$), based on the Hmin and Hmax values set at the export time. During the following scenarios, it may be necessary to take a step size, *h*, even below Hmin:

- $hc <$ Hmin: $h = hc$
- $(tc_{i+1} - t) <$ Hmin: $h = (tc_{i+1} - t)$ where t is the current time of the Twin Builder solver.

Note that it will never take a step-size larger than Hmax and it will never go beyond tc_{i+1}.

Simulation under the co-simulation mode will always be slower than the native simulation in the Twin Builder due to the communication overheads and input/output handling between the primary solver and the Twin Builder solver. The finer the communication step-size, the more delays will be incurred. Therefore, the communication step-size by the primary solver must be chosen appropriately to avoid those delays.

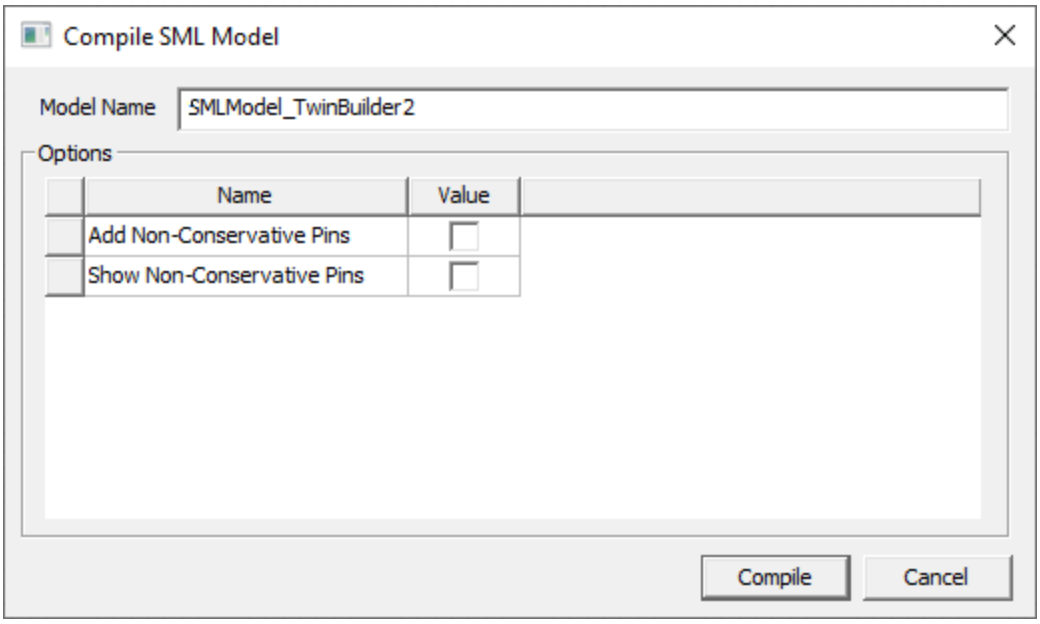
For the detailed mathematical background on FMI co-simulation, see section 4.1 of the FMI 2.0 Standard.

Compile an SML Model from a Schematic

Twin Builder provides you with a method for creating an SML model, component, and symbol from the current schematic. The resulting model, component, and symbol are added to the respective folders in the current project's **Definitions** folder. You can also export the new model to a library.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To create a model, component, and symbol from the current schematic, select **Tools** > **Design Tools** > **Compile As SML Model**, or right-click a subsheet and select **Complile As SML Model**. The **Compile SML Model** dialog box appears.



- **Model Name** – A default name based on the current design name is provided. If you choose to enter another name, it must conform to Twin Builder's naming conventions, and also be unique for the current project. Message reminders appear if the naming criteria are not met.
- **Add Non-conservative Pins** – Select this check box to add pins to the symbol for non-conservative interface ports.
- **Show Non-conservative Pins** – Select this check box to display the pins added above. Clear the check box to make them hidden initially.

> **Note:**
>
> Coupling links and Python components are not supported if they are part of the chosen schematic.

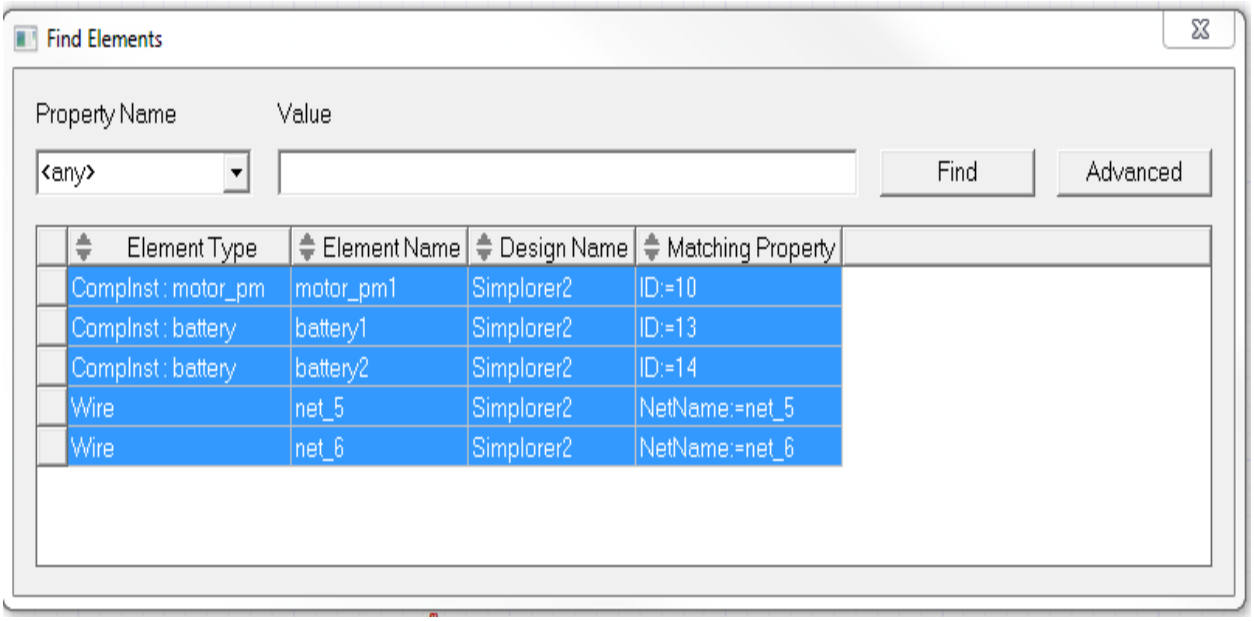
Finding Elements in the Schematic Editor

To find one or more elements in the **Schematic Editor**, you can use the **Find Elements** dialog box. The **Find Elements** dialog box provides a **QuickFind** view and an advanced view. Click the appropriate button to switch views.

To use **QuickFind** to find one or more elements in the **Schematic Editor**:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. On the **Edit** menu, click **Find Elements**, or press F3. This opens the **QuickFind** view of the **Find Elements** dialog box. If you want to access the advanced view, click **Advanced**.



2. Click the **Property Name** drop-down list, and select the element properties for your search. If you want to search all types, select **<any>**. If you want to search for a specific property that is not in the drop-down list, type the property name.
3. Enter a search string for the property in the **Value** field. Wildcards “*” and “?” can be used in searches.
4. Click **Find** to initiate a search.

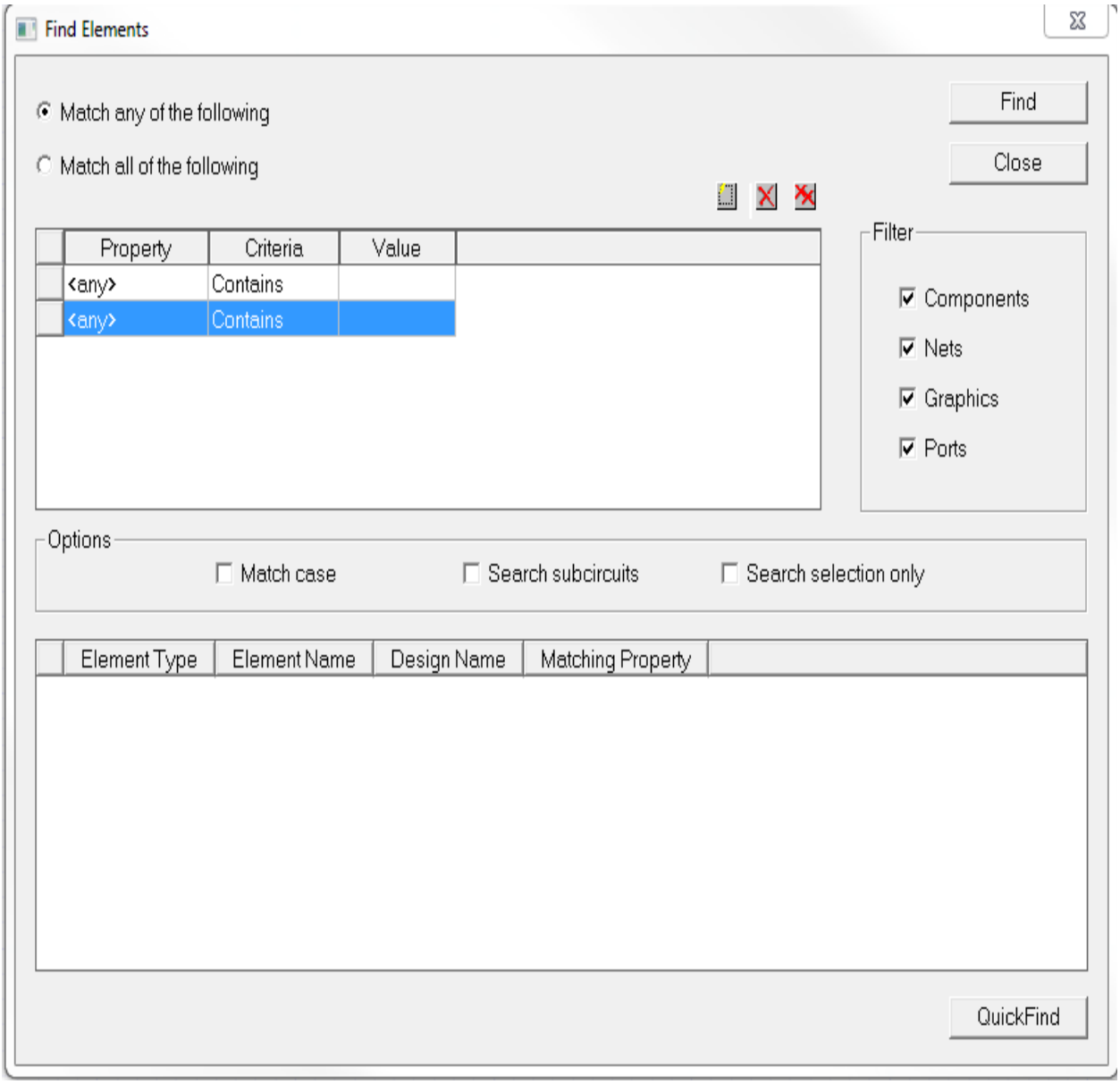
   Search results are displayed in the **Results** window of the **Find Elements** dialog and highlighted on the schematic circuit or subcircuit. Click an individual search result item to highlight that particular circuit or subcircuit element in the **Schematic Editor**.

> **Note:**
>
> If you click **Advanced**, the selections and results are displayed in the advanced view of the dialog.

To use the advanced option to find one or more elements in the **Schematic Editor**:

1. On the **Edit** menu, click **Find Elements**, or press F3. This opens the **QuickFind** view of the **Find Elements** dialog box.
2. Click **Advanced**. This opens the advanced view of the **Find Elements** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



3. Click the **Property** field, and select the element properties you want to search for from the drop-down list. If you want to search all types, select **<any>**. If you want to search for a specific property that is not in the drop-down list, type the property name.

   Use the buttons to add, remove, or remove all search properties in the table.

4. Click the **Criteria** field to select Contains, Does not contain, or Exact as the search criteria.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Enter a search string for the property in the **Value** field. Wildcards "*" and "?" can be used in searches. For example, searching the **Refdes** property for "R?" with **Match any of the following** and **Contains** criterion would find R1 and R2, but not C1.
6. You can further control a search using the following controls:
   - Select a **Filter** check box to include element types (Components, Nets, Graphics, or Ports) in the search.
   - Select an **Option** check box to vary the scope of the search.
7. Choose **Match all of the following** to restrict the results to elements that meet *all* of the specified search requirements. Choose **Match any of the following** to match elements that meet *any* of the search requirements.
8. Click **Find** to initiate a search.

   Search results are displayed in the **Results** window of the **Find Elements** dialog and highlighted on the schematic circuit or subcircuit. Click an individual search result entry to highlight that particular circuit or subcircuit element in the **Schematic Editor**.

> **Note:**
>
> If you click **QuickFind**, the selections and results are displayed in the **QuickFind** view of the dialog box.

Selecting Elements

You can select components, wires, and primitive drawing elements for various operations, such as rotation or deletion, in any of the following ways:

- To select a single element (component or wire segment) in the Schematic Editor, place the cursor over the element and click to select it. Click again anywhere in the schematic window to deselect the object.
- To select multiple elements, hold down the **Ctrl** key and click the elements. To deselect one of the multiple elements, hold down the **Ctrl** key and click the element. To deselect all selected elements, click anywhere in the schematic window.

When you move a selected component, connected wiring normally stays attached to the component.

- To move a component and break its connections, hold down the **Shift** and **Ctrl** keys and drag the component to its new location.
- To select all connected segments of a wire (also called nets), hold down the **Shift** and **Ctrl** keys, then click any segment of the wire. You can also drag the wire segments to a new location, breaking any connections.

If you have difficulty selecting an object, you can select it from The **Select Elements** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Edit** > **Select Elements**. The **Select Elements** dialog box appears..
2. In the **Select Elements** dialog box, click the element(s) you want to select in the **Schematic Editor**. You can select/deselect the **Components** or **Nets** box to vary the list of elements you can choose. You may also click **Select All** or **Deselect All**.

Sorting Components

The sort order of Block components affects how simulations run. To sort designs using block components:

1. Select **Schematic** > **Sort Components**.

   The **Determine Block Sequence** dialog box displays.

   Block sequence numbers also appear next to each block component on the schematic.

2. Sorting can be performed in any of the three following ways:
   - Sorting **Automatically** is done left to right, top to bottom on the block components as they appear on the schematic.

     Click **OK** when finished to close the dialog box.

   - To sort manually, select an item in the list, then use the up and down arrow keys to move the item in the list. The block sequence numbers update accordingly on the schematic.

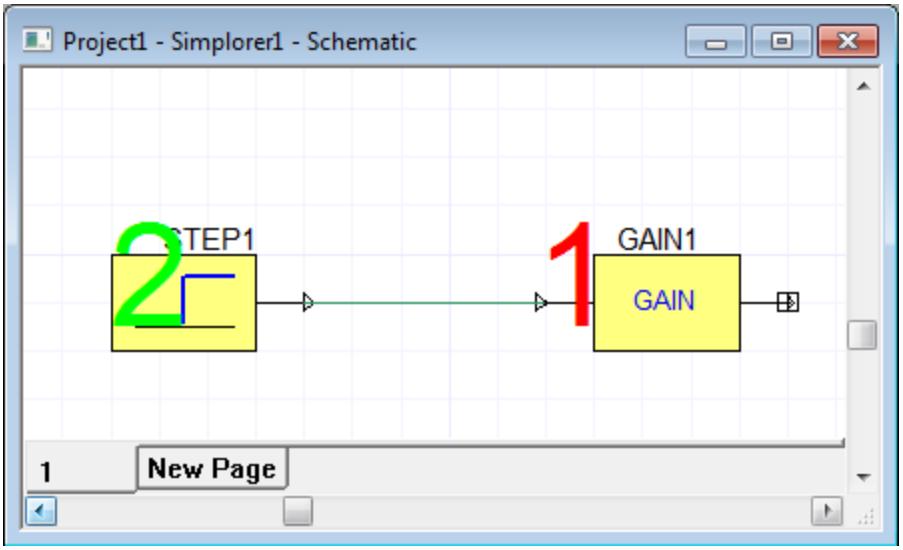
     Click **OK** when finished to close the dialog box.

   - Choosing **Interactive** sorting closes the dialog box while numbers indicating the current sort order appear next to each block component on the schematic.

     Click the block components in the desired sort order. As you make selections, the numbers update accordingly — disappearing when you select the last unsorted component.

> **Note:**
>
> By default, green numbers indicate that the block has already been sorted (picked), red means the component has not yet been selected for sorting.
>
> The colors are arbitrary and can be changed in the dialog box as can the font size.

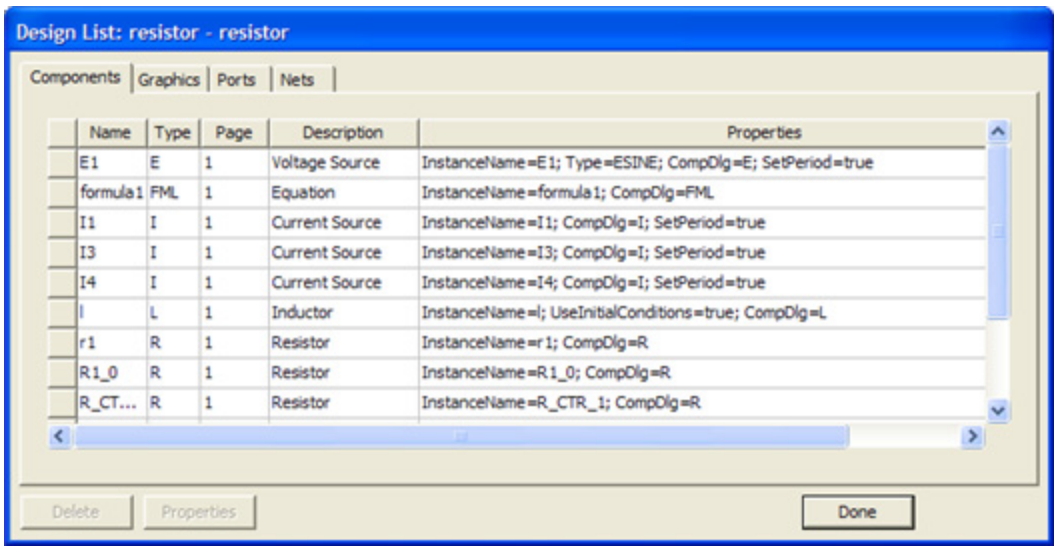
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Listing Design Elements

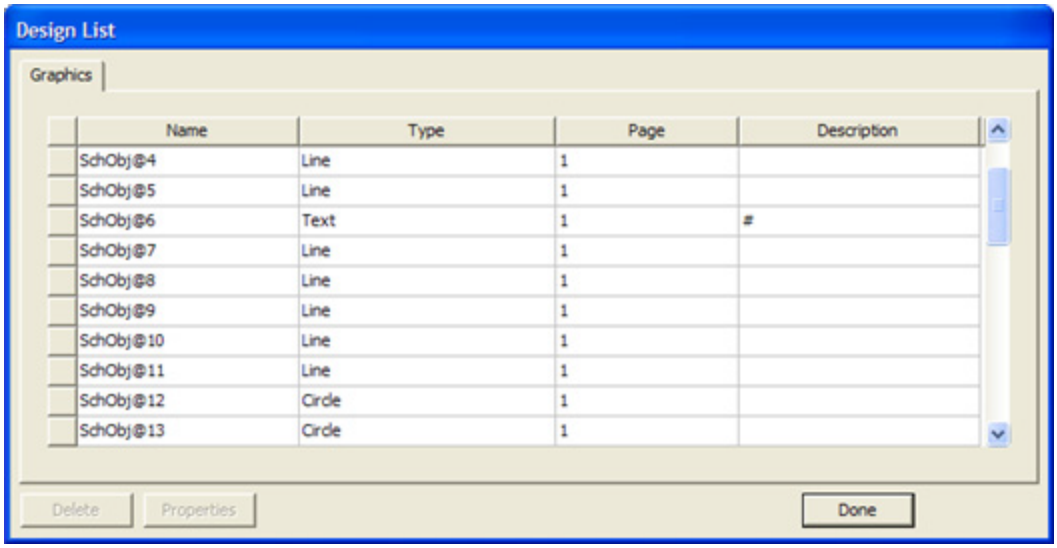
Twin Builder provides a convenient way to list the components, graphics, ports and nets of a schematic, namely the **Design List** dialog box. A **Design List** dialog box is also used in the symbol editor to list a symbol's graphics elements.

- To access the **Design List** dialog box in the schematic editor, select **Schematic** > **List** from the Twin Builder top menu bar:



- To access the **Design List** dialog box in the symbol editor, select **Symbol > List** from the Twin Builder top menu bar.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

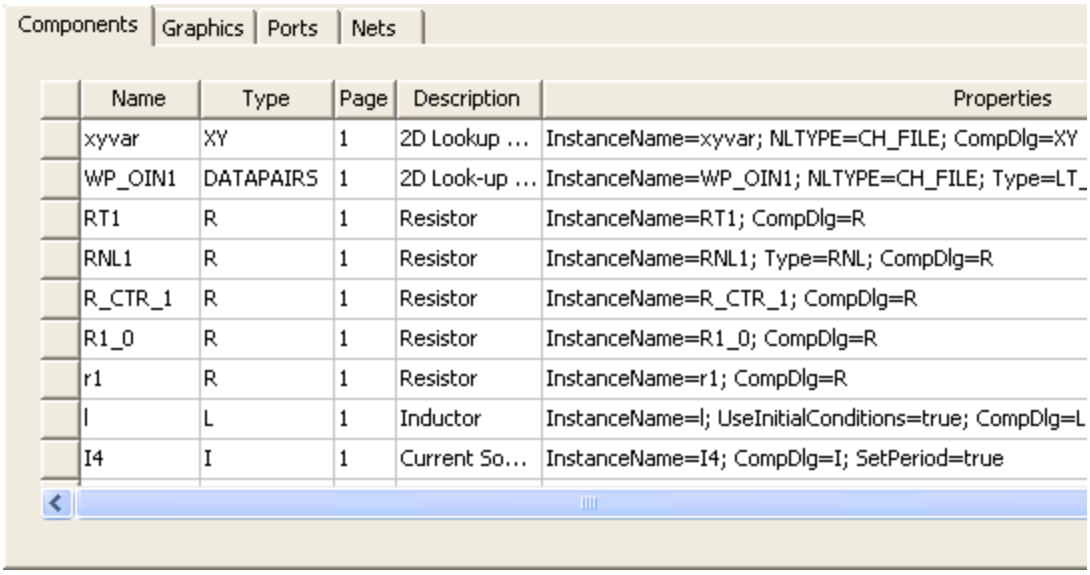


Note that the **Design List** dialog for the schematic editor contains four tabs: **Components**, **Graphics**, **Ports**, and **Nets**. For the symbol editor, only the **Graphics** tab is present.

- Each tab contains a grid that shows the objects that are present in the schematic or symbol editor.
- Each row in the grid contains one item, and each column displays an aspect of that item.
- Click a column header to sort it. Click multiple times to reverse the order of the sort.
- Selecting a row zooms to that item in the editor and selects it.
- Multiple items can be selected at once using the shift or control keys, which will zoom/pan the display so that all selected items are visible.
- The **Delete** and **Properties** buttons apply to the objects in the selected rows.
- To manipulate an object's properties, select the desired object and click **Properties** to open its properties dialog box for editing.
- To delete objects, select them and click **Delete**.
- You can undo property changes and deletions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

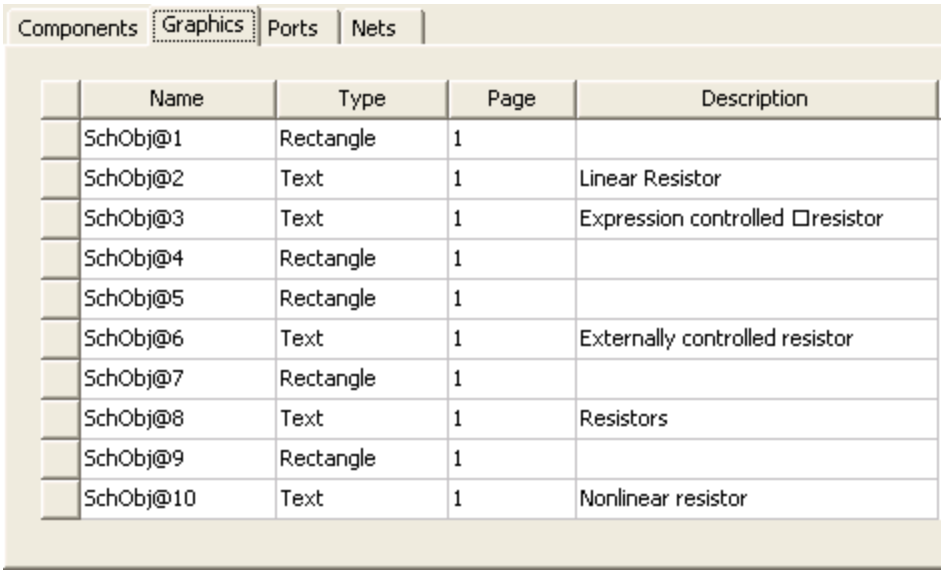
Components Tab



Name	Type	Page	Description	Properties
xyvar	XY	1	2D Lookup ...	InstanceName=xyvar; NLTYPE=CH_FILE; CompDlg=XY
WP_OIN1	DATAPAIRS	1	2D Look-up ...	InstanceName=WP_OIN1; NLTYPE=CH_FILE; Type=LT_
RT1	R	1	Resistor	InstanceName=RT1; CompDlg=R
RNL1	R	1	Resistor	InstanceName=RNL1; Type=RNL; CompDlg=R
R_CTR_1	R	1	Resistor	InstanceName=R_CTR_1; CompDlg=R
R1_0	R	1	Resistor	InstanceName=R1_0; CompDlg=R
r1	R	1	Resistor	InstanceName=r1; CompDlg=R
l	L	1	Inductor	InstanceName=l; UseInitialConditions=true; CompDlg=L
I4	I	1	Current So...	InstanceName=I4; CompDlg=I; SetPeriod=true

- **Name** – Shows the ID of the schematic component.
- **Type** – Shows the component name from the library.
- **Page** – Shows the page number on which that component is located.
- **Description** – Shows the component description.
- **Properties** – Shows the overridden properties of that component.

Graphics Tab

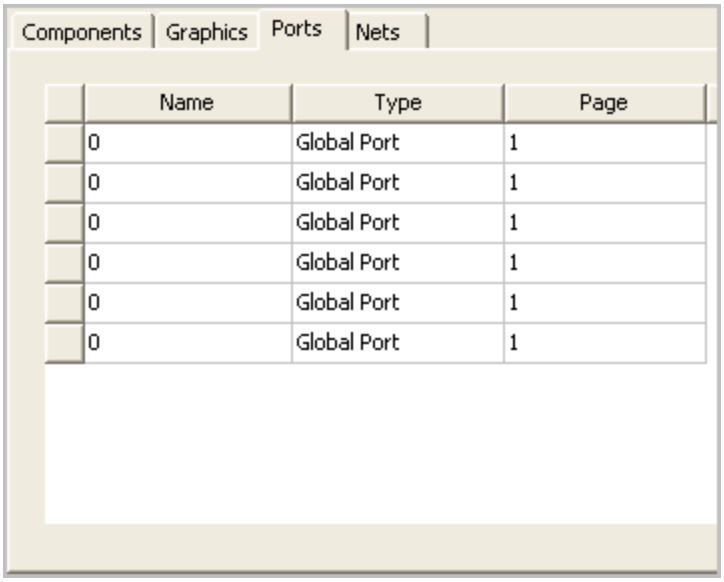


Name	Type	Page	Description
SchObj@1	Rectangle	1	
SchObj@2	Text	1	Linear Resistor
SchObj@3	Text	1	Expression controlled □resistor
SchObj@4	Rectangle	1	
SchObj@5	Rectangle	1	
SchObj@6	Text	1	Externally controlled resistor
SchObj@7	Rectangle	1	
SchObj@8	Text	1	Resistors
SchObj@9	Rectangle	1	
SchObj@10	Text	1	Nonlinear resistor

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Name** – Shows the ID of the graphic.
- **Type** – Shows the graphic type (Circle, Rectangle, Arc, Text, Pin, and so on).
- **Page** – Shows the page number on which that graphic is located.
- **Description** – Shows the descriptive text for text and pin graphic types; and is blank for non-textual graphics.

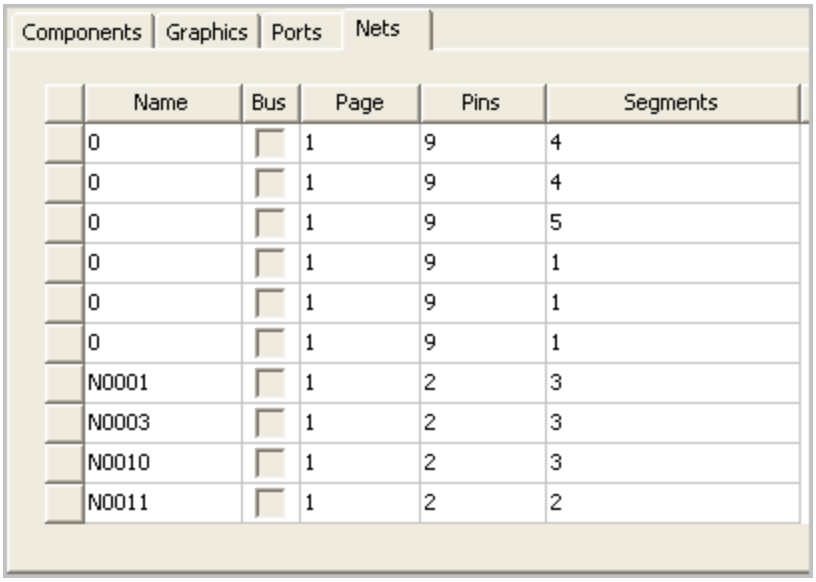
Ports Tab



- **Name** – Shows the name of the port.
- **Type** – Shows the port type (Interface, Page, or Global).
- **Page** – Shows the page number on which that port is located.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Nets Tab



Name	Bus	Page	Pins	Segments
0		1	9	4
0		1	9	4
0		1	9	5
0		1	9	1
0		1	9	1
0		1	9	1
N0001		1	2	3
N0003		1	2	3
N0010		1	2	3
N0011		1	2	2

- **Name** – Shows the name of the net.
- **Bus** – Selected if the item is a bus, and is cleared for normal wires.
- **Page** – Shows the page number on which that net is located.

  Note that a net may appear more than once if it spans multiple pages.
- **Pins** – Number of pins connected to that net.
- **Segments** – Number of individual line segments that comprise that net.

Editing Operations

When the Schematic Editor is the active window, click **Edit** on the top menu bar to access the drop-down list that contains the following commands. The commands that are active depend on what is selected, and on the previous command.

> **Note:**
>
> Commands such as **Copy**, **Paste**, **Delete**, and **Rename** also appear in various context menus. For example, in the Project Manager Project tab to copy and paste Analysis setups and options, and report definitions.

- **Undo** undoes the last operation. The last operation you performed appears in this menu item.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Redo** re-executes the last operation that was undone. The last operation you performed appears in this menu item.
- **Cut** deletes the selected component or wire segment, and retains a copy for pasting into a schematic in the same application.
- **Copy** creates a local copy for pasting into a schematic in the same application.
- **Paste** puts the local object from the previous copy or cut into the schematic.
- **Delete** deletes the component.
- **Rename** activates the text field of the selected schematic, project, plot, or other editable item in the Project tree so that you can modify the object's name.
- **Select All** selects all the components and nets in the schematic.
- **Select Elements** opens a dialog so that you can select a component by name. See ***Selecting Elements***.
- **Find Elements** opens a dialog so that you can find all elements that match the criteria you specify. See ***Finding Elements in the Schematic Editor***.
- **Copy to Clipboard** creates a global copy on the clipboard for pasting into a different application.
- **Properties** opens the Properties dialog for the selected element. See ***Editing Component Properties***.
- **Copy Data** is used for copying properties for components and primitive drawing elements. See ***Copying and Pasting Properties***
- **Paste Data** is used for pasting properties for components and primitive drawing elements. See ***Copying and Pasting Properties***
- **Activate** restores a deactivated component to the circuit.
- **Deactivate (Open)** temporarily converts the component into an open circuit. This is displayed graphically with a red X over the circuit element.
- **Deactivate (Short)** – temporarily converts the component into a short circuit. This is displayed graphically with a circled red X over the circuit element. Deactivating (short) a component with multiple conservative pins will connect all the conservative pins to the same net. Deactivating (short) a block (which contains non-conservative pins) does not work the same as the circuit component, and the output of the shorted block will be 0.
- **Name Wire** opens a dialog so that you can assign a new name to the selected wire node.

Adding Primitive Drawing Elements

In Twin Builder you can add primitive drawing elements such as lines, arcs, circles, rectangles, polygons, and text. You can also import images in bitmap (.bmp) and .gif formats. You can use the included drawing elements or imported images to create symbols for schematic pages or custom components. These drawn elements do not affect the simulation, but can help to document or clarify your schematic.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Copying and Pasting Properties

To Copy and Paste Common Properties for Primitive Drawing Elements

Properties of Drawing Elements

Each primitive drawing element has an associated **Properties** window that you can display when you select the element. In the **Properties** window, you can edit properties such as line width, color, and fill style.

You can select, and edit properties for, and/or delete or copy any drawing primitive.

Related Topics

Copying and Pasting Properties

To Copy and Paste Common Properties for Primitive Drawing Elements

Drawing an Arc

To add an arc to a schematic:

1. Click **Draw** > **Primitive** > **Arc** or click the Arc icon under **Graphics** on the **Schematic** ribbon.

   This enables the Arc drawing mode.

2. Move the cursor to the desired location in the schematic and left click to draw the first end point of the arc.
3. Move the cursor to the desired location for the second endpoint of the arc. and left-click.

   This draws a straight line between the two points and gives the cursor a drag-handle at the mid-point of the line.

4. Drag the mid-point to establish the arc, and left click release the mode.

   This shows the arc selected, with the drag handles on the end points and midpoint of the arc highlighted. If necessary, you can select a handle on the arc to change the location of the endpoints or the arc.

   The arc includes a **Properties** window in which you can edit the arc's properties, including: color (applies to both border and fill), line width, fill style, center coordinates, radius, and starting and ending angles.

5. Click anywhere on the schematic off the arc to deselect it.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Drawing a Circle

To add a circle to a schematic.

1. Click **Draw** > **Primitive** > **Circle** or click the **Circle** icon under **Graphics** on the **Schematic** ribbon.

   This enables the circle drawing mode.

2. Move the cursor to the desired location in the schematic and left click to specify the center point of the circle.
3. Move the cursor to the desired location for the diameter of the circle and left-click.

   This dynamically draws the circle and shows the arc selected, with the drag handles on the end points and midpoint of the arc highlighted. If necessary, you can select a handle on the circle to change the diameter. Notice that the element includes a Properties window.

   The circle includes a **Properties** window in which you can edit the circle's properties, including: fill color, border width, border color, fill style, center coordinates, and radius.

4. Click anywhere on the schematic off the circle to deselect it.

Drawing a Line

To add a line to a schematic:

1. Click **Draw** > **Primitive** > **Line** or click the **Line** icon under **Graphics** on the **Schematic** ribbon.

   This enables the line drawing mode.

2. Move the cursor to the desired location in the schematic and left click to draw the first end point of the line.
3. Move the cursor to the desired location for the second endpoint of the line and left-click.

   This draws a straight line between the two points and shows the line selected, with the drag handles on the end points highlighted. If necessary, you can select a handle on the line to change the location of the endpoints.

   The line includes a **Properties** window in which you can edit the line's properties, including: color, line width, line style, coordinates of the endpoints, and objects such as arrowheads to be placed at line ends.

4. Click anywhere on the schematic off the line to deselect it.

Drawing a Polygon

To add a closed polygon to a schematic:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Click **Draw** > **Primitive** > **Polygon** or click the **Polygon** icon under **Graphics** on the **Schematic** ribbon.

   This enables the Polygon drawing mode.

2. Move the cursor to the desired location in the schematic and left click to draw the first vertex.
3. Move the cursor to the desired location for each subsequent vertex and left-click.

   This draws a straight blue line between the designated vertices, and a straight black line between the last vertex specified and the first.

4. Right-click to complete the polygon.

   The polygon remains selected, with the drag handles visible at the vertices. If necessary, you can select a handle to drag any vertex.

   The polygon includes a **Properties** window in which you can edit the polygon's properties, including: border color, border width, fill style, fill color, and coordinates of the vertices.

5. Click anywhere on the schematic off the polygon to deselect it.

Drawing a Rectangle

To add a rectangle to a schematic:

1. Click **Draw** > **Primitive** > **Rectangle** or click the **Rectangle** icon under **Graphics** on the **Schematic** ribbon.

   This enables the rectangle drawing mode.

2. Move the cursor to the desired location in the schematic and left click to place the first corner of the rectangle.
3. Move the cursor to the desired location for the opposite corner of the rectangle and left click.

   This shows the rectangle selected, with the drag handles on the corners visible. If necessary, you can select a handle resize the rectangle.

   The rectangle includes a **Properties** window in which you can edit the rectangle's properties, including: border color, border width, fill style, fill color, center coordinates, width, height, and angle of rotation.

4. Click anywhere on the schematic off the rectangle to deselect it.

Adding Text to a Schematic

Follow this procedure to add text to a schematic:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Draw** > **Primitive** > **Text** or select **Schematic** > **Graphics** > **Text** on the ribbon. This enables text mode.
2. Move the cursor to the desired location in the schematic and left click. This places a highlighted text box with **Default Text** written and highlighted within it.
3. Type in the text box to replace the default text.
4. Right-click the text box and select **Properties** to open the **Properties** dialog box. You can edit the text's properties, including: color, location (coordinates), angle of rotation, size, font, and justification. Select or clear the check box to toggle display of a bounding rectangle. When displayed, additional rectangle properties can be displayed including: border color and width, fill color and style, the rectangle's center coordinates, width, height, and angle of rotation. Note that the rectangle center and angle can differ from the text center and angle.
5. Click anywhere on the schematic to deselect the text.

Adding an Image to a Schematic

To add an image to a schematic:

1. Select **Draw** > **Primitive** > **Image** or **Schematic** > **Graphics** > **Image** on the ribbon. A file browser dialog box appears.
2. Navigate to the image you want to insert and click **Open**.
3. Click inside the schematic. The image appears at the cursor location. Resize it as needed by dragging the cursor. Resizing maintains the original proportions.
4. Click again to anchor the image and release it from the cursor. Click and drag the image to move it.
5. Click the image to display its properties in the **Properties** pane. In the **Properties** pane, you can edit the image's center coordinates, angle of rotation, width, and height. You can also:
   - Select the **Link To File** check box to establish a link to the image file.
   - Select the **Mirrored** check box to flip the image horizontally.
   - Select the **Display Border** check box to display a border around the image. When you select the **Display Border** check box, additional fields appear in which you can adjust the border width and color.

Drawing Operations

When the Schematic Editor is the active window, use these commands on the **Draw** menu and right-click menu to perform various operations on schematic elements. The commands that are active depend on what is selected, and on the previous command.

- **Rotate** — Rotates a selected object or group of objects 90° counterclockwise. You can also press **Ctrl+R**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Align Horizontal** — Horizontally aligns selected components, property displays, or combination of components and property displays with the first object you selected.

  To align multiple objects horizontally:

1. Press **Ctrl** and click the object with which you want to align the others.
2. Still pressing **Ctrl**, click the additional objects to add them to the selection.

> **Note:**
>
> The first-selected object is highlighted in red, and the subsequently selected objects are highlighted in dark red.

3. On the **Draw** menu, click **Align Horizontal**. The selected objects align horizontally with the first selection.
   - **Align Vertical** — This option vertically aligns selected components, property displays, or combination of components and property displays with the first object you selected.

To align multiple objects vertically:

1. Press **Ctrl** and click the object with which you want to align the others.
2. Still pressing **Ctrl**, click the additional objects to add them to the selection.

> **Note:**
>
> The first-selected object is highlighted in red, and the subsequently selected objects are highlighted in dark red.

3. Select **Draw** > **Align Vertical**. The selected objects align vertically with the first selection.
Flip Vertical — This option flips the selected object or group of objects around the X-axis.

   To flip multiple objects vertically:

1. Press **Ctrl** and click an object to flip vertically.
2. Still pressing **Ctrl**, click the additional objects to add them to the selection.

> **Note:**
>
> The first-selected object is highlighted in red, and the subsequently selected objects are highlighted in dark red.

3. Select **Draw** > **Flip Vertical**.

- **Flip Horizontal** — This option flips the selected object or group of objects around the Y-axis.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To flip multiple objects horizontally:

1. Press **Ctrl** and select an object to flip horizontally.
2. Still pressing **Ctrl**, click the additional objects to add them to the selection.

> **Note:**
>
> The first-selected object is highlighted in red, and the subsequently selected objects are highlighted in dark red.

3. On the **Draw** menu, click **Flip Horizontal**.

- **Bring to Front** — moves the selected object to the front of the drawing.
- **Send to Back** — moves the selected object to the back of the drawing.

Adding Reports to a Schematic

Twin Builder can add several types of reports directly onto a schematic. You can modifiy and update these plots in the same way as those displayed under **Results** in the Project tree.

To place a report on a schematic:

1. Select the schematic so that the **Draw** menu appears in the menu bar.
2. Select **Draw** > **Report** > *report_type*, where report type can be: **Rectangular Plot**, **DataTable**, **Digital Plot**, **Rectangular Stacked Plot**, **Numeric Display**, **Rectangular Contour Plot**, **3D Rectangular Plot**, **Polar Plot**, **Bode Plot**, or **Nyquist Plot** and move the cursor over the schematic.

   Alternatively, click **Standard Report** on the **Results** ribbon and click the icon for the desired report.

   The mouse drags a hollow rectangular box with the cursor at the center.

3. Click the mouse at the location where you want to place the report.

   The report appears on the schematic and the **Report** dialog box displays enabling traces to be added immediately. An entry for the report also appears in the **Results** folder of the **Project Manager**.

> **Note:**
>
> You can also add a report to a schematic by dragging an existing report entry in the **Results** folder of the **Project Manager** onto the sheet.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Modifying an On-sheet Report

Opening an On-sheet Report in a New Window

Generating Reports and Postprocessing

Creating Rectangular Plots

Creating 3D Rectangular Plots

Creating Polar Plots

Creating Rectangular Stacked Plots

Creating a Rectangular Contour Plot

Creating a Bode Plot

Creating a Nyquist Plot

Creating a Digital Plot

Creating a Data Table or Numeric Display

Modifying an On-sheet Report

You can move or resize the report by selecting it, then using the mouse pointer move and resize it. Text elements resize with the plot. Use the **Delete** key to remove the report from the schematic.

1. Right-click the report and select **Edit in Place**.

   The cursor changes from a pointer dragging a coordinate symbol to a normal mouse pointer. When you move the pointer over an editable element of the plot, the pointer changes color.

2. To modify the report, that is, add traces, variables, sweeps, and so on, right-click any elements and select **Modify Report** from the shortcut menu to display the **Report** dialog box.
3. To edit an element's properties do any of the following:
   a. Select the desired element and modify its properties in the **Properties Window**.
   b. Double-click the element to open its **Properties** dialog box in which you can edit the element properties.
   c. Right-click the element and select **Edit** > **Properties** on the shortcut menu to open the element's **Properties** dialog box in which you can edit properties.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Opening an On-sheet Report in a New Window

Generating Reports and Postprocessing

Creating Rectangular Plots

Creating 3D Rectangular Plots

Creating Polar Plots

Creating Rectangular Stacked Plots

Creating a Rectangular Contour Plot

Creating a Bode Plot

Creating a Nyquist Plot

Creating a Digital Plot

Creating a Data Table or Numeric Display

Opening an On-sheet report in a New Window

You can open an on-sheet report in a new editing window. Right-click the report and select **Open in new window** on the shortcut menu. The report opens in a new report editing window in which it can be modified and updated in the same way as reports displayed under **Results** in the **Project** tree.

Related Topics

Modifying an On-sheet Report

Generating Reports and Postprocessing

Creating Rectangular Plots

Creating 3D Rectangular Plots

Creating Polar Plots

Creating Rectangular Stacked Plots

Creating a Rectangular Contour Plot

Creating a Bode Plot

Creating a Nyquist Plot

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Creating a Digital Plot

Creating a Data Table or Numeric Display

Setting Up Multi-Page Schematics

Setting up multi-page schematics involves two activities: adding pages, and adding and naming page connectors to establish electrical connectivity between them as necessary.

To add a page or pages to a schematic:

1. Open the schematic to which you want to add a new page or pages.
2. Click the **New Page** tab; or right-click the tab and select **New Page** from the context menu.

   You are now editing the new page, 2.
3. Repeat as necessary to add additional pages.

To remove a page from a schematic:

> **Warning:**
>
> Removing pages containing components connected to other pages may adversely affect simulation results or render the design inoperable. You can use the **Undo** command to restore removed pages.

1. Open the schematic from which you want to remove a page or pages.
2. Do either of the following:
   a. Right-click the tab of the page you want to remove and select **Remove Page** from the context menu.
   b. Click the tab of the page you want to remove and select **Remove Page** on the **Schematic** menu. In the **Remove Pages** dialog box, select the page or pages you want to remove and click **OK**.

To change a page's properties:

1. Select the page.
2. Do either of the following:
   a. On the **Schematic** menu, click **Page Properties**.
   b. Right-click the tab of the page whose properties you want to change and select **Properties** from the context menu.
3. Edit the properties listed on the Page Properties tab.

To establish a common electrical connection between pages:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

You can place a page connector on each page that will share the connection. To do this on each page:

1. Select the page.
2. Do either of the following:
   - On the **Draw** menu, click**Page Connector**.
   - On the **Schematic Draw** toolbar, click the **Page connector** icon .

The cursor, now associated with a page connector symbol for placement, moves to the center of the schematic window. To place the connector, click at the desired location.

Hint	You can rotate a page connector before placing it by repeatedly pressing **R** on your keyboard. Each press rotates the connector 90º counterclockwise.

If **Multiple Placement** is turned on for page connectors in the **Schematic Options** dialog box, you can click additional locations to place additional connectors.

To stop placing page connectors during multiple placement, do either of the following:

- Press **Enter**, the **SPACEBAR**, **Backspace**, or **Esc** on your keyboard.
- Right-click and select **Place and Finish**, **Finish**, **Cancel**, or **Back**.

> **Note:**
>
> To ensure electrical connectivity among schematic elements, the pins of placed page connectors snap to a 100-mil (2.54-millimeter) grid. This snapping cannot be turned off, and the spacing of the connectivity grid cannot be adjusted.

Within a design and at the same hierarchical level, all page connectors with the same name act as a common electrical connection. Therefore, once you have finished placing page connectors that you want to be electrically common, you must ensure that they all have the same name. To change the name of a page connector, do either of the following:

- Open its **Properties** dialog box - double-click the connector's schematic symbol, type a new name into the **Value** cell for the port's **Name** property, and click **OK**.
- Click the connector.

  In the **Parameters** tab of the **Properties** window, click the **Value** cell for the **NetName** parameter, type the new name, and press **Enter**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Changing the name of any member of a group of same-named page connectors renames all members of the group with the new name.

To remove a page or pages from a schematic:

1. Open the schematic from which you want to remove a page or pages.
2. On the **Schematic Menu**, select **Remove Page** to open the **Remove Pages** dialog box.
3. In the **Remove Pages** dialog box, select the pages you want to remove, and click **OK**.

   To remove a page, right-click the tab of the page you want to remove, and select **Remove Page** from the context menu.

> **Note:**
>
> **Remove Page** removes pages containing components without warning. If you want to restore a page, select **Undo**.

Setting up Hierarchical Designs

A hierarchical design is a schematic design that contains *subdesigns* or *subcircuits*. Any number of hierarchical levels can be created, and any number of subdesigns can be placed in another design. The design at the top of such a structure is commonly called the *top-level design* or *top-level schematic*. A design that contains a subdesign is called the *parent* of that subdesign. Subdesigns can include *subcircuits*, or external models coupled to the parent schematic.

You can set up analyses at any level in a hierarchy, analyze, and view results of the analyses. An analysis performed on a design at a level other than the top level includes that design and all of its subdesigns and subcircuits, if any, but not its parent design or designs at any higher levels.

Analyses you specify for various designs at different levels of a hierarchy are entirely independent of each other. There are two approaches to creating a hierarchical design: bottom-up and top-down.

The Bottom-Up Approach

1. Start Twin Builder, if it is not already running.

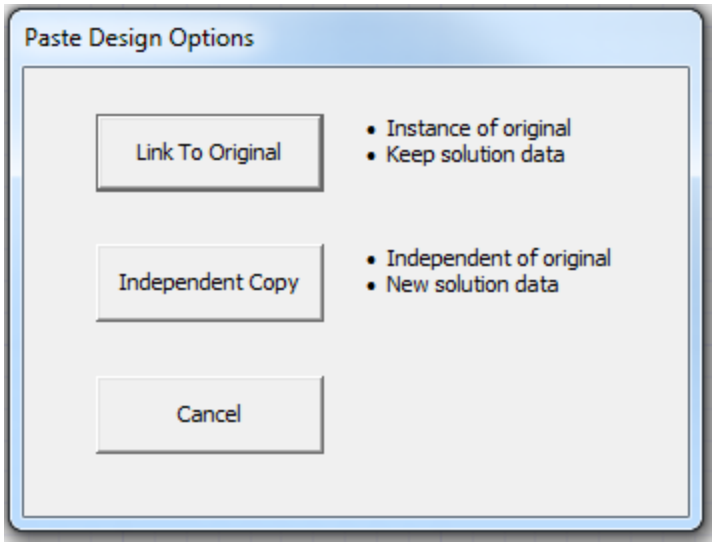
   If Twin Builder is already running, select **File** > **New** to create a new project.

2. Insert a design in the default new project or new project you created by selecting **Insert Twin Builder Design** on the **Project** menu.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. In the **Project Manager** pane, right-click the new Twin Builder design and select **SubCircuit** > **Add SubCircuit**.
4. Repeat step 3 for each subdesign you will need.
5. To build the hierarchy:
   a. In the Project tree, drag the design icon for the design that you want to become a subdesign and drop it on the parent design.

      The **Paste Design Options** dialog box opens.



   b. Click **Link to Original** to keep the design linked to the original and keep its solution data, or click **Independent Copy** to make the design independent of the original and create new solution data for it.

      A symbol representing the subcircuit appears in the parent schematic. You can place and wire it as desired.

The Top-Down

1. Start Twin Builder, if it is not already running.

   If Twin Builder is already running, create a new project; select **File** > **New**.

2. Insert a design in the default new project or new project you created by selecting **Insert Twin Builder Design** on the **Project** menu.

   This will be the top-level circuit.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. For each subcircuit you will need in the top-level circuit:
    a. Make sure the top-level circuit is active in the schematic editor.
    b. On the **Twin Builder** menu, point to **Add SubCircuit**, and click **Add SubCircuit** on the sub-menu.
4. Repeat step 3 at each level of the hierarchy as needed.
5. To reuse subcircuits at multiple places in a design:
    a. In the Project tree, right-click the icon for a design you want to reuse and select **Copy**.
    b. Right-click the icon for the design into which you want to copy the subcircuit, and select **Paste**.

    A symbol representing the subcircuit appears in the parent schematic. You can place and wire the subcircuit as desired.

As you create subcircuits, you will probably want to rename them. To do this:

1. In the Project tree, click the name of the icon for the design that you want to rename.
2. Click the icon name again.
3. The icon name changes to an editable string that you can modify. When you renamed the design name to your liking, press **Enter**.

Twin Builder creates a generic rectangular symbol for each new subcircuit. You can customize the symbol for a subcircuit as follows:

1. Expand the folder in the Project tree labeled **Definitions**, then expand the **Symbols** subfolder.
2. Double-click the symbol icon that corresponds to the subcircuit symbol you want to edit.
3. The symbol opens for editing in Twin Builder's symbol editor.
4. When you're finished editing the symbol, click **Update Project** on the **Symbol** menu to propagate your changes through the project.

Adding a Subcircuit to a Twin Builder Design

Use the **SubCircuit** menu to:

- Add subcircuits to an exiting design.
- Link externally developed subcircuit models from applications such as MATLAB/Simulink, Ansys Maxwell, or Q3D Extractor to a Twin Builder design, then use Twin Builder to interact with these models in a process called co-simulation.

Related Topics

Creating a Subcircuit from a Selection Area

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding a Subcircuit

Follow this procedure to add a new Twin Builder subcircuit to a design:

1. Open the design to which you want to add a subcircuit.
2. On the **Twin Builder** menu, click **Add SubCircuit**.

   This displays a menu of the kinds of subcircuits that you can choose to add.

3. Select **Add SubCircuit** from the menu list.

   Twin Builder adds a subcircuit to the project as a blank schematic sheet and adds a default symbol linked to the new circuit to the parent schematic. The symbol's default name and other properties can be viewed and edited like any other component.

   A design symbol for the new subcircuit also appears in the **Project** tree subordinate to the parent design.

Add interface ports to the subcircuit sheet to provide external connections to the subsheet. The ports are either conservative or quantity of any supported type. You can add these interface ports as pins to the subcircuit symbol on the parent schematic. The pin spacing is controlled by the **SubCircuit Pin Spacing** setting on the Schematic Editor Options: General tab.

> **Note:**
>
> Unconnected interface ports adjust to the domain they are being connected to, in the subcircuit. However if the corresponding top-level pins in the parent circuit are wired, then automatic adjustment does not occur, in which case domain conversion is applied instead.

Related Topics

Setting Up Hierarchical Designs

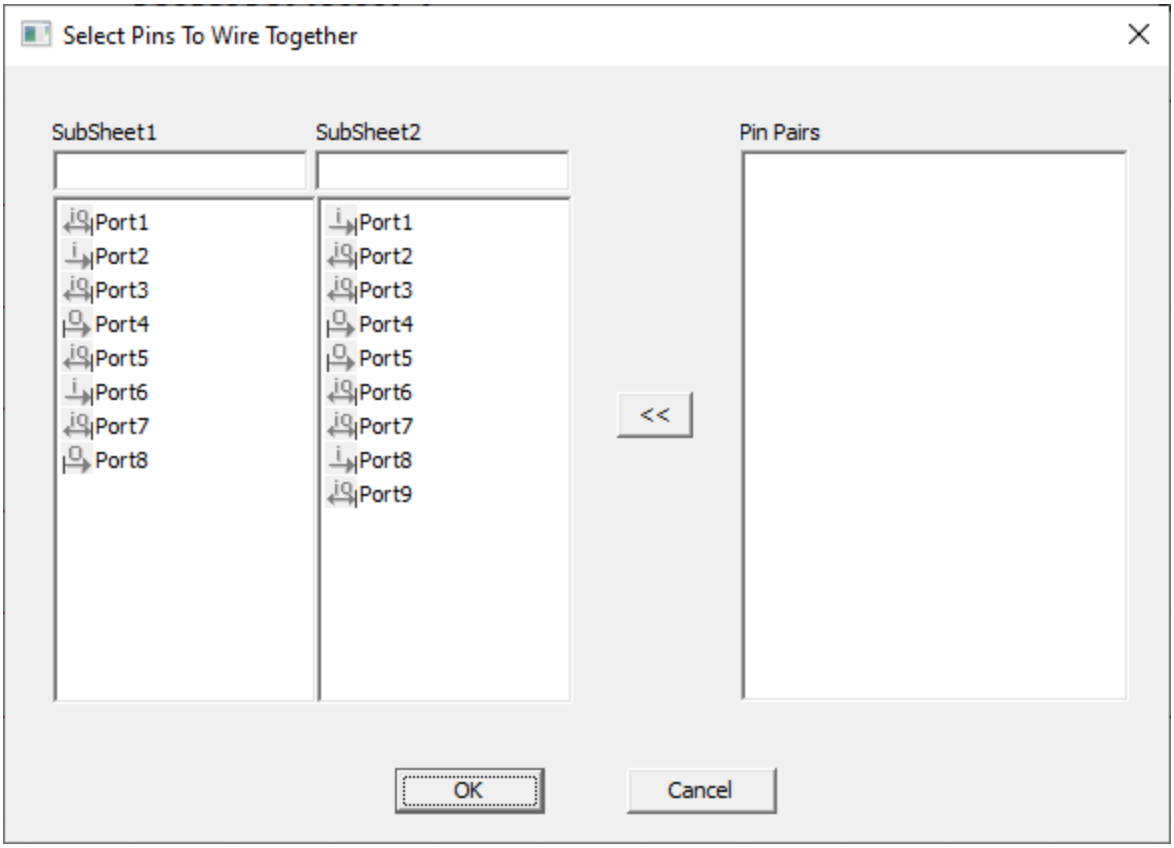
Interface Ports

Schematic Editor Options: General Tab

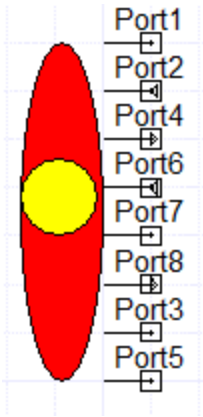
Wiring Window

In the Schematic Editor, hold Ctrl and select two component instances, then right-click and select **Wire Together**. This opens the **Symbol Pin Match** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The two components must be defined by a rectangle shape. The shape is determined by the location of the pins; they must be arranged in a rectangle. See the figure below for an example.



The two fields in the left side of the dialog box display the pins for the selected components; the field label reflects the selected component. The pins of the selected component on the left appear in the component list on the left; the pins of the selected component on the right appear

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

in the component list on the right. Pins appear in alphanumeric order. Select a pin from list to place the pins in the **Pin Pair** list.

To efficiently search through a long list of pins, begin typing in the text field above a component list. The values in the list will filter as you type. You can also use * as a wild card to match zero or more characters.

You can also filter the results by entering one of the following into the text-box above a component list: **<Odd>**, **<Even>**, **<Top Half>**, **<Bottom Half>**.

Input pins can only be connected to output pins, output pins can only be connected to input pins, and in-out pins can only be connected to in-out pins,

Click **<<** to remove selected pins from the **Pin Pair** list.

Click **OK** to confirm your selections. When you click **OK**, the two symbols are moved together and positioned two squares apart (as long as one of them is not pre-wired to other components). For each pair selected, the pin from the left component is moved to the right side of the symbol, if it was not there before. Likewise, the pin from the right component is moved to the left side of the symbol, if necessary. Two pins are aligned horizontally and a wire connection appears between them. Some components overlap and must be moved manually.

Select **Edit** > **Undo Wiring** to undo the wiring operation and restore both symbols to their original states.

Creating a Subcircuit from a Selection Area

Select **Schematic** > **Create SubCircuit from Selection Area** to select a rectangular area of the schematic and send the component instances enclosed within the area to a subsheet attached to the current parent sheet. Net routing and port types are adjusted to make sure that connectivity is maintained.

To create a subcircuit from a selection area:

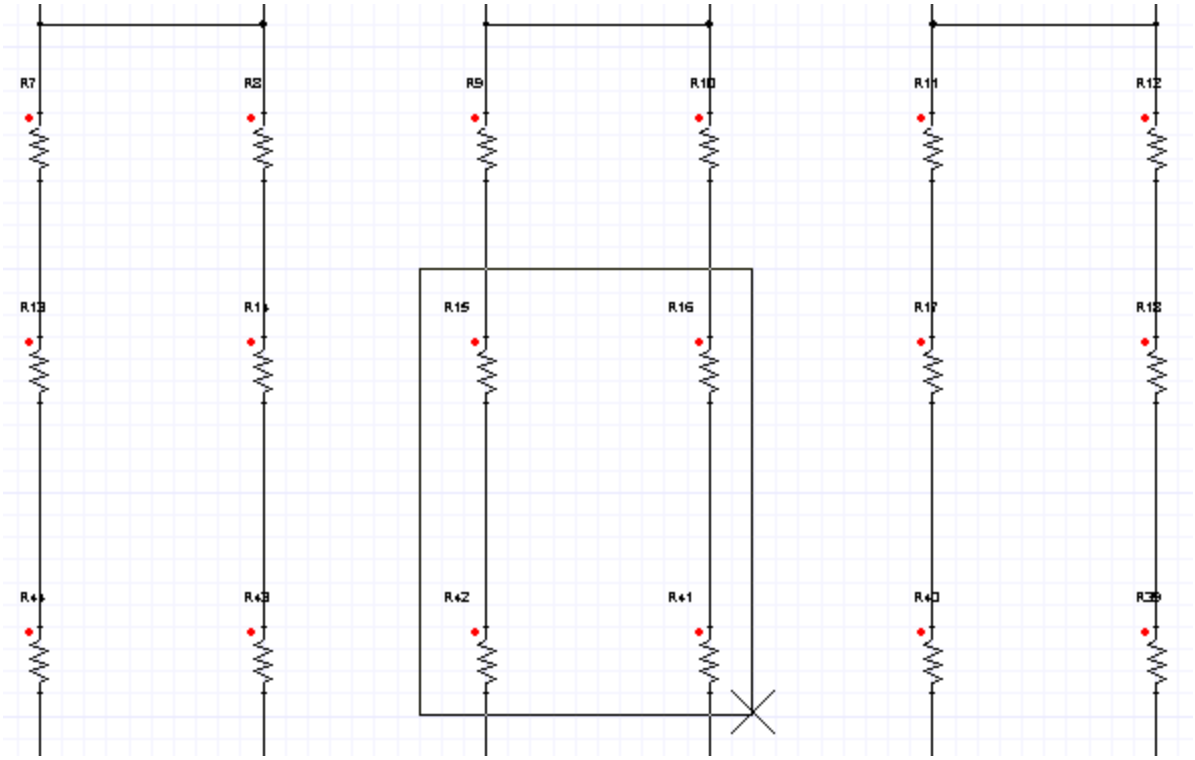
1. Select **Schematic** > **Create SubCircuit from Selection Area**.

   The cursor changes to a large for selecting the area of interest.

2. Click and drag the cursor until the selection rectangle encloses the area containing the components you want to include in the subcircuit, then click again to complete the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

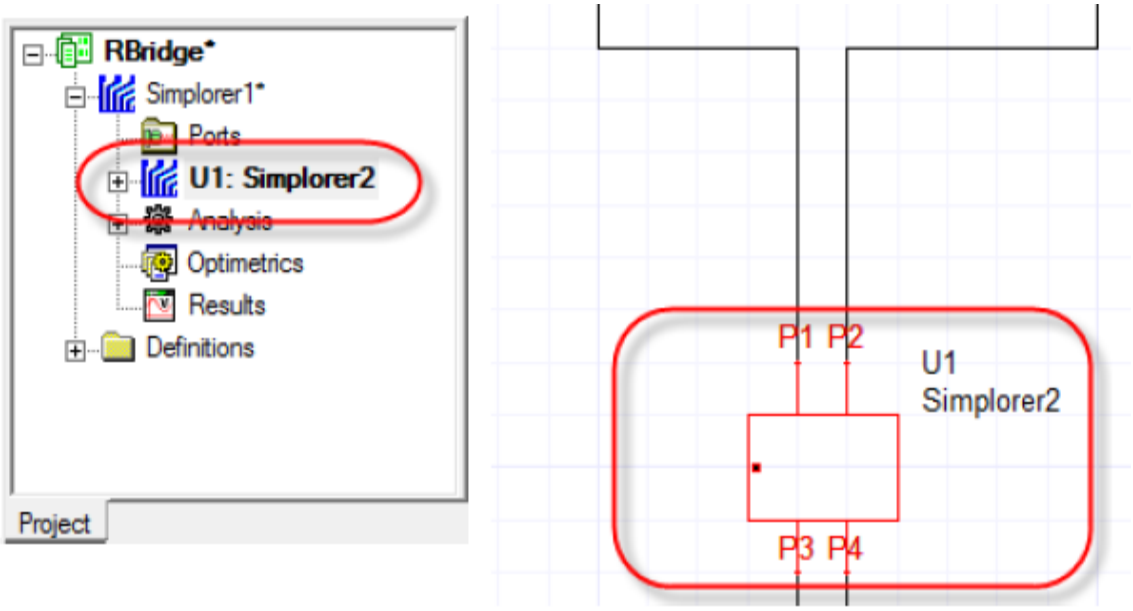
selection.



A subsheet component is created containing the selected components.

The new subsheet's icon also appears in the **Project Manager** Project tree under its parent design.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

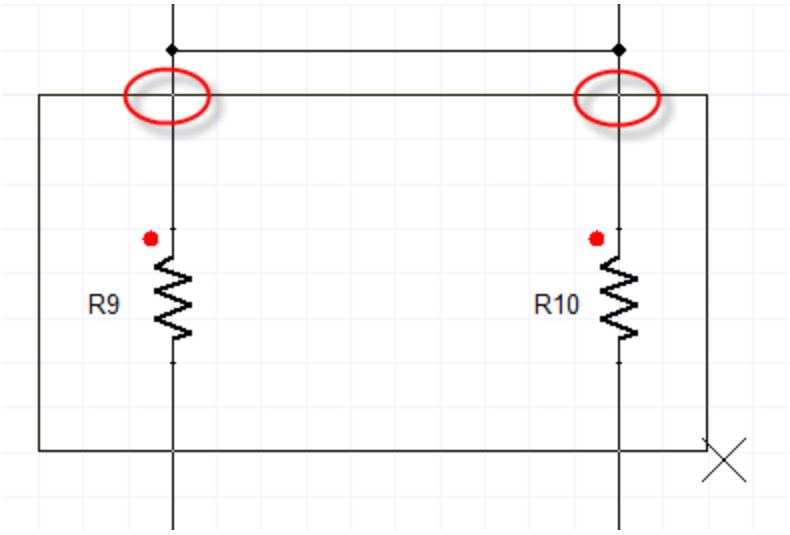


The new subsheet component pin spacing is controlled by the **SubCircuit Pin Spacing** setting on the Schematic Editor Options: General tab. You can right-click the new component and select **Push Down** on the context menu to view the subsheet and confirm the selected components are present.

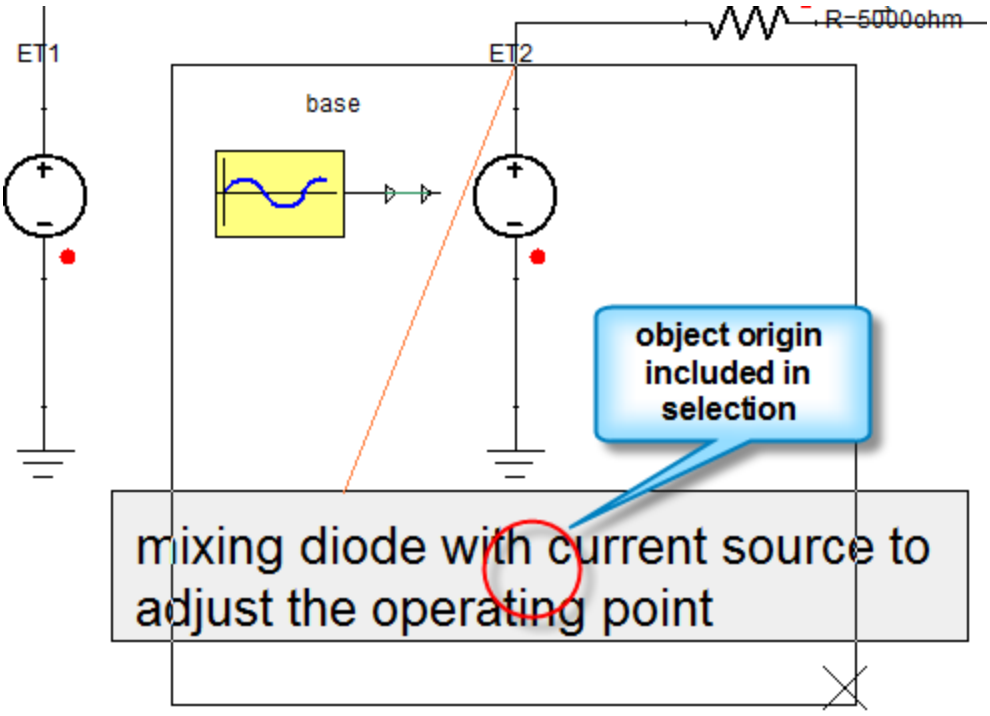
Considerations when selecting an area:

- The selection area boundaries should only intersect nets on the schematics since they form the interface of the subsheet.
- Twin Builder subcircuits do not allow a single net to cross the interface more than once. For example the following selection would fail to create a subcircuit.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- All pins of an enclosed component instance must be inside the selection boundary for that instance to be included.
- For instances that have no pins, the object's origin must be inside the selection boundary for it to be included in the subsheet.



- Interface ports cannot be present inside the selection area.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Page connectors will be inserted to maintain connectivity if Twin Builder is unable to route the interface nets on the parent sheet.

Related Topics

Setting Up Hierarchical Designs

Interface Ports

Page Connectors

Copying an Existing Subcircuit within a Design

There are two methods for copying and pasting an existing subcircuit within a design. One method results in a pasted component that refers to the *same* underlying design. The second method results in a copy that is *independent* of the original source design.

Creating a Copy that refers to the same design

To copy an existing subcircuit within a design:

1. Open the design that contains the subcircuit you want to copy.
2. Do either of the following:
   - In the Project tree, right-click the icon for the subcircuit and select **Copy**.
   - In the design, right-click the subcircuit symbol, and select **Copy**.
3. Do either of the following:
   - On the **Edit** menu, click **Copy**.
   - Press Ctrl+V.

Within the schematic window, the cursor is associated with an *N*-port symbol for subcircuit placement. To place the subcircuit, click at the desired location.

Hint	You can rotate the symbol for a subcircuit before placing it by repeatedly pressing **R** on your keyboard. Each press rotates the component 90º counterclockwise.

If Multiple Placement is turned on for components in Twin Builder's **Schematic Options** dialog box, you can click additional locations to place additional instances of the subcircuit.

To stop placing subcircuits during multiple placement, do either of the following:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Press Enter, the spacebar, Backspace, or Esc on your keyboard.
- Right-click and select **Place and Finish**, **Finish**, **Cancel**, or **Back**.

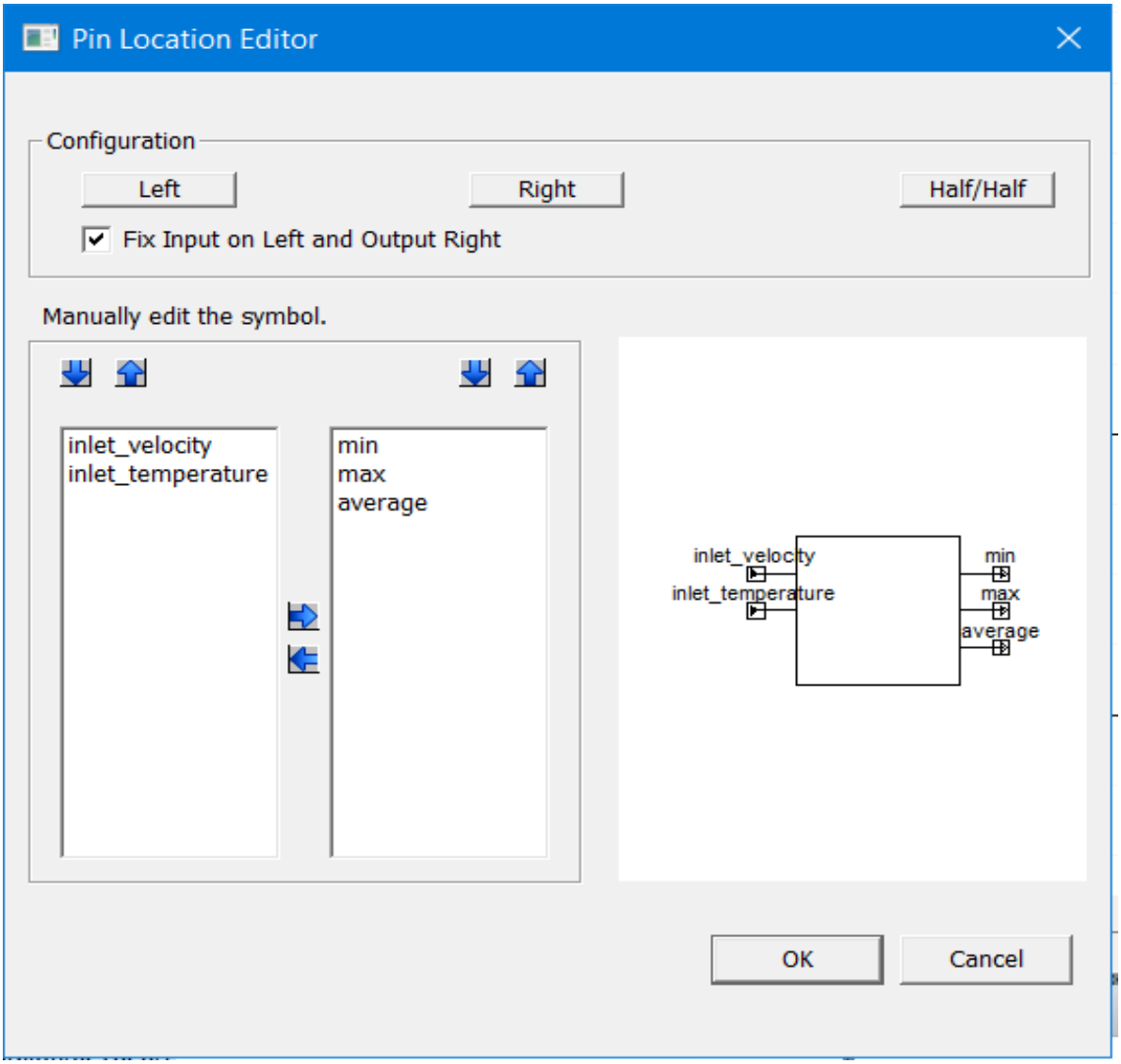
> **Note:**
>
> To ensure electrical connectivity among schematic elements, the pins of placed subcircuit symbols snap to a 100-mil (2.54-millimeter) grid. This snapping cannot be turned off, and the spacing of the connectivity grid cannot be adjusted.

Editing Subcircuit Pin Locations

Use the **Pin Location Editor** dialog box to change the locations of the symbol pins on a subcircuit. Follow this procedure:

1. Right-click the subcircuit and select **Edit Symbol** > **Edit Pin Location**. The **Pin Location Editor** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. In the **Configuration** section:
   a. Click **Left**, **Right**, or **Half/Half** to designate a type of configuration.
   b. Select the **Fix Input on Left and Output Right** check box to specify that input pins are on the left and output pins are on the right.
3. In the **Manually edit the symbol** section, click the arrow buttons to change the position of the pins, as well as move them to either side of the subcircuit.
4. Click **OK** to save your changes and close the **Pin Location Editor** dialog box; click **Cancel** to close the dialog box without making any changes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Moving Between Designs in a Schematic Hierarchy

Double-click a project's different designs and design instances in the Project tree to move between them. Specific menu commands are also available for moving up and down one level.

To move down one level:

1. In the schematic from which you want to move, select the symbol for the design to which you want to move.
2. Select **Schematic** > **PushDown** .

To move up one level:

1. In the schematic from which you want to move, select the symbol for the design to which you want to move.
2. Select **Schematic** > **PopUp** .

Copying a Twin Builder Design into Another Design

To copy an existing Twin Builder design into another design for the first time:

1. In the Project tree, right-click the icon for the design you want to copy.
2. Select **Copy**.
3. Open the design into which you want to copy.
4. Do either of the following:
   - On the **Edit** menu, click **Paste**.
   - On the keyboard, press **CTRL**+**V**.

   The **Synchronize Design** dialog box opens.
5. Specify how the circuit you are copying will be associated with the target design by doing one of the following:
   - Select **Make Copy** to create an independent subdesign, which will not be affected by changes to the original.
   - Select **No copy - use instance of original** to keep the copy linked to the original, so that changes to one will affect the other.
6. Click **OK**.

Within the schematic window, the cursor is associated with a symbol for subcircuit placement. To place the subcircuit, click at the desired location.

Hint	You can rotate the subcircuit symbol before placing it by repeatedly pressing **R** on your keyboard. Each press rotates the component 90º counterclockwise.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If there is already a subdesign in the parent schematic, there is an easy way to create an independent copy. Right-click the subdesign symbol and select **Copy as New Design**. Then paste using the menu command, or CTRL+V. In this case the **Synchronize Design** dialog will not appear.

> **Note:**
>
> • To ensure electrical connectivity among schematic elements, the pins of placed subcircuit symbols snap to a 100-mil (2.54-millimeter) grid. This snapping cannot be turned off, and the spacing of the connectivity grid cannot be adjusted.
>
> - When copying components, you can also use **Copy to Clipboard** in the right-click popup menu to copy and paste components into other applications.

The first time you place a subcircuit in a design, an entry for it is added under the **Project Components** heading in the Project Manager Component tree.

To save time as your work progresses, you can place new instances of the subcircuit on the schematic:

- Double-click its **Project Components** icon.
- Right-click its **Project Components** icon and select **Place Component**.
- Drag its **Project Components** icon.

If **Multiple Placement** is turned on for components in Twin Builder's **Schematic Options** dialog box, click additional locations to place additional subcircuit instances.

To stop placing subcircuits during multiple placement, do any of the following:

- Press Enter, the spacebar, Backspace, or Esc>
- Right-click and select **Place and Finish**, **Finish**, **Cancel**, or **Back**.

Working with Design Configurations

Twin Builder lets you save the model choices for component instances from the current design and its hierarchy to a model configuration file (**.mcfg**). For example, the following shows the contents of a model configuration file for a design containing a voltage source (E1), a resistor (R1), and a capacitor (C1):

```
E1.SimulatorModel=E
R1.SimulatorModel=R
C1.SimulatorModel=C
```

The model used by each component instance is listed following the component instance name.

If a design contains subcircuits, component instances placed on subcircuit "U1" are listed thus:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
U1.E2.SimulatorModel=E
```

Twin Builder also lets you load a saved design configuration, enabling you to reset the component instances to the model types present in the design at the time the .mcfg file was saved.

Because .**mcfg** files are text files, you can edit them to set property values for component instances, as well as set or add design and project variables.

Related Topics

Saving a Design Model Configuration

Loading a Design Model Configuration

Editing a Design Model Configuration

Saving a Design Model Configuration

Follow this procedure to save a model configuration for a design:

1. Ensure that the design whose model configuration you want to save is currently active.
2. Select **Twin Builder** > **More Actions** > **Save Configuration** from Twin Builder's main menu. You can also right-click the design's icon in the Project Manager Project tree and select **Save Configuration**.

   The **Save Model Configuration File** dialog box opens.
3. Browse to the location in which you want to save the **.mcfg** file.
4. Enter a name for the file and click **Save** to save the file and close the dialog box.

Related Topics

Loading a Design Model Configuration

Editing a Design Model Configuration

Loading a Design Model Configuration

To load a model configuration for a design:

1. Ensure that the design whose configuration you want to update is currently active.
2. Select **Twin Builder** > **More Actions** > **Load Configuration** from the **Twin Builder** menu. You can also right-click the design's icon in the **Project Manager** Project tree and select **Load Configuration**.

   The **Select Model Configuration File** dialog box opens.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Browse to the **.mcfg** file you want to load; and click **Open** to load the configuration file.

Related Topics

Saving a Design Model Configuration

Editing a Design Model Configuration

Editing a Design Model Configuration File

Because **.mcfg** files are text files, you can edit them. You can then use the edited file to set design parameters directly by loading the edited configuration file into the design. Use this capability to modify property values for component instances in a design, as well as set or add design and project variables. The use of expressions is also supported. These changes are undoable, except for project variable changes.

In general, any parameter listed in a component's **Properties** dialog box can be set. The general form for entries in the **.mcfg** file is as follows:

<ComponentName>.<parameter_name>=<parameter_value>

For example:

```
R2.R=45kOhm
E1.Type=ESINE
E1.FREQ=400
```

The above configuration lines to set the resistance of component R2 to 45 kOhm, and voltage source E1's type to a sine output having a frequency of 400 Hz. Select **Twin Builder** > **More Actions** > **Load Configuration** to load the configuration lines into the target design, then right-click the design's icon in the **Project Manager** Project tree and select **Load Configuration**,

Project variables can similarly be added. For example:

```
$C1=22.4pF
```

will add project variable $C1 having an initial value of 22.4 pF

Below is an example that adds two local design variables:

```
aa=4
bb=if(aa < 5, 1, 0)
```

These lines add design variable "aa" having a value of 4, and a second design variable "bb" whose value is determined by an expression.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

When you have finished editing a **.mcfg** file, select **Twin Builder** > **More Actions** > **Load Configuration** (or right-click the design's icon in the **Project Manager** Project tree and select **Load Configuration**) to load its contents into the desired design.

Related Topics

Saving a Design Model Configuration

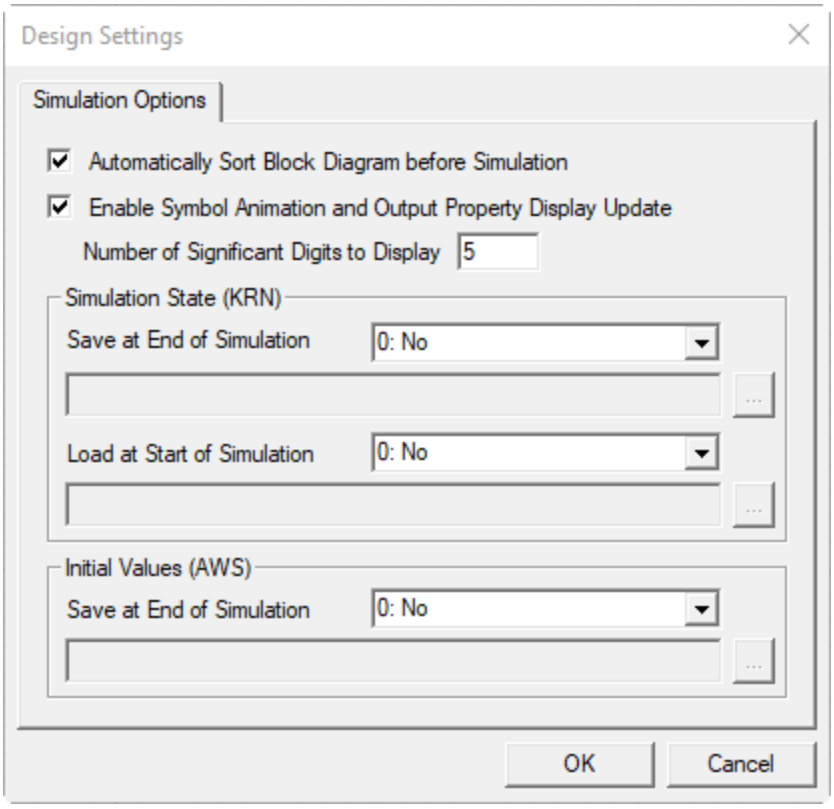
Loading a Design Model Configuration

Twin Builder Design Conventions

Design Settings

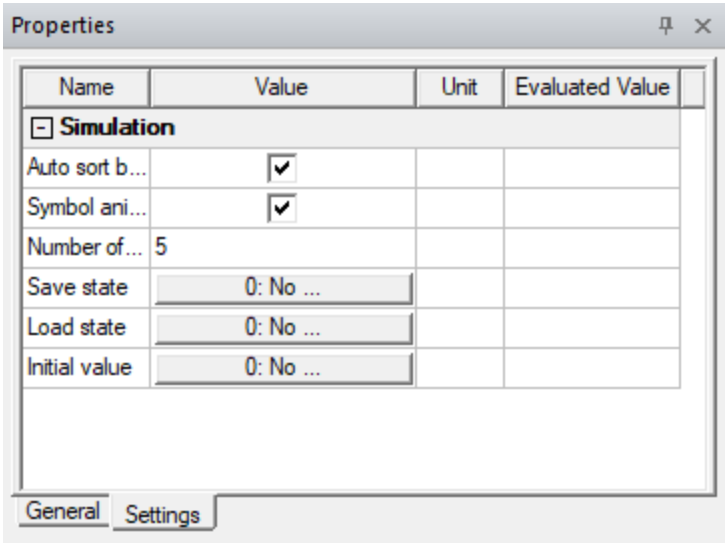
Use the Twin Builder **Design Settings** dialog box to control various simulation options.

1. To make design settings, right-click the desired design icon in the Project tree and select **Design Settings** or select **Twin Builder** > **Design Settings**. The **Design Settings** dialog box opens.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Selecting a current design in the Project tree also displays the Design Settings in **Properties** window under the solver tab.



2. The controls in the **Design Settings** dialog box save state and initial values of a simulation model in separate files at the end of the simulation.
    - **Automatically Sort Block Diagram Before Simulation** – Select this check box to sort block diagrams before running a simulation.
    - **Enable Symbol Animation and Output Property Display Update** – Select this check box to enable symbol animation and enable the **Number of Significant Digits to Display** field. Animations embedded in a symbol change the symbol appearance during the simulation process in response to a change in value of a system quantity. See ***Using the Symbol Editor*** for more details.
    - **Number of Significant Digits to Display** – Enter a value in this field to set the number of significant digits displayed for evaluated properties on schematics. The number set in this field must be between **1** and **14**. The default value is **5**.
    - **Simulation State (KRN): Save at End of Simulation** – Use this drop-down list to determine whether to save the simulation state to a **.krn** file at the conclusion of a simulation. The default location\file name of the state file is \<*projectname*>**.aedtresults**<*projectname*>**_**<*designname*>**.krn**. The file is overwritten an the end of each simulation. You can continue a simulation later with this state file.

    Select **2: Ask for filename before simulation** if you want to specify a path and name for the state file at the start of simulation. When you start a simulation, the **State and Initial Value Files** dialog box appears in which you can specify this information. Click **Cancel** in this dialog box to allow the simulation to proceed without saving a file.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Simulation State (KRN): Load at Start of Simulation** – Use this drop-down list to determine whether an existing simulation state file loads at the start of simulation and is then used for that simulation. The default location\file name of the state file is \<*projectname*>**.aedtresults**<*projectname*>**_**<*designname*>**.krn**.

Select **2: Ask for filename before simulation** if you want to be prompted to choose a state file manually at the start of simulation. When you start a simulation, the **State and Initial Value Files** dialog box appears in which you can specify this information. Click **Cancel** in this dialog box to continue the simulation without loading a file.

- **Initial Values (AWS): Save at End of Simulation** – Use this drop-down list to determine whether to save the simulation model values (inductances and capacitances) to an **.aws** file at the conclusion of a simulation. The default location\file name of this initial values file is \<*projectname*>**.aedtresults**<*projectname*>**_**<*designname*>**.aws**. Saved values can be used as initial values for a subsequent simulation.

Select **2: Ask for filename before simulation** if you want to specify a path and name for the initial values file at the start of simulation. When you start a simulation, the **State and Initial Value Files** dialog box appears in which you can specify this information. Click **Cancel** in this dialog box to continue the simulation without saving a file.

3. When finished, click **OK** to save your choices and close the dialog box.

> **Note:**
>
> When performing simulation by enabling **Simulation State (KRN): Load at Start of Simulation**, the simulation results only correspond to the Twin Builder design for which the KRN was previously saved. Any changes to the design such as replacing components with different input waveforms after the KRN was saved are not reflected in the simulation.
>
> The only exception is that the values of existing input properties can be changed when the Load at Start of Simulation is performed through Optimetrics analysis given that the specific input property value is expressed as a project/design variable, or when the property is enabled with the **Sweep** check box.

Exporting a Schematic

You can export the currently active schematic to a file as follows:

1. On the **Schematic** menu, choose **Export File.**

   A **Save As** file browser dialog box opens.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Choose either **Ansoft Neutral File (V4) (*.anf)** or **Microsoft Enhanced Meatfile (*.emf)** as the **Save as Type**.
3. Browse to the location where the file is to be saved.
4. Enter a **File name**.
5. Click **OK** to save the file.

Printing a Schematic

When you are preparing to print a schematic, you can control page setup, and preview the print with your chosen settings before printing.

Setting Up the Page Layout

To set up the page layout before printing:

1. Select **Page Setup** from the **File** drop-down on the Twin Builder top menu bar.

   The **Page Setup** dialog box opens.

2. Change settings as desired.
   - Use the **Paper**, **Orientation**, **Margins**, and **Print Zoom** controls to specify how you want the print to appear.
   - In the **Schematic Options** panel, you can specify whether to draw the grid and print the design variables.
   - In the **Border type** panel, you can select the type of border to print around the schematic.

   Select the **Ansys** border type to place a single line box around the schematic and add the current date and page number to the print.

   Select the **Schematic-defined** border type to print the border as defined through the **Schematic > Page Borders** menu option at the size specified. Typically, select this option when printing out at a specific scale (that is, 100%) to ensure that the border will be visible. Select **File** > **Print Preview** to preview the effect of your choices.

3. Click **OK** to close the **Page Setup** dialog box with your settings.

Using Print Preview

1. Select **Print Preview** from the **File** menu to see a preview of the schematic print.
2. To navigate through the preview, click **Next Page**, **Prev Page**, and/or **Two Page**.
3. To zoom in or out on the preview, click **Zoom In** or **Zoom Out**. You can also click directly on the preview image to zoom in and out.
4. Click **Print** to close the preview window and open the **Print** dialog box for printing.
5. Click **Close** to dismiss the preview and return to the project.

Printing the Schematic

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To print the schematic that is in the active window:

1. Right-click in the schematic window and select **Print** from the menu (or select **Print** from the **File** menu) to open the **Print** dialog box:
2. Use the controls to specify how you want the print to appear.
   - In the **Schematic Options** panel, you can specify whether to draw the grid and print the design variables.
   - In the **Border type** panel, you can select the type of border to print around the schematic.

     Select the **Ansys** border type to place a single line box around the schematic and add the current date and page number to the print.

     Select the **Schematic-defined** border type to print the border as defined through the **Schematic->Page Borders** menu option at the size specified. Typically, select this option when printing out at a specific scale (that is, 100%) to ensure that the border will be visible.

   - Make any desired selections in the **Print Range**, **Copies**, and **Print Zoom** panels.
3. Click **OK** to print the schematic, or click **Cancel** to cancel the operation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

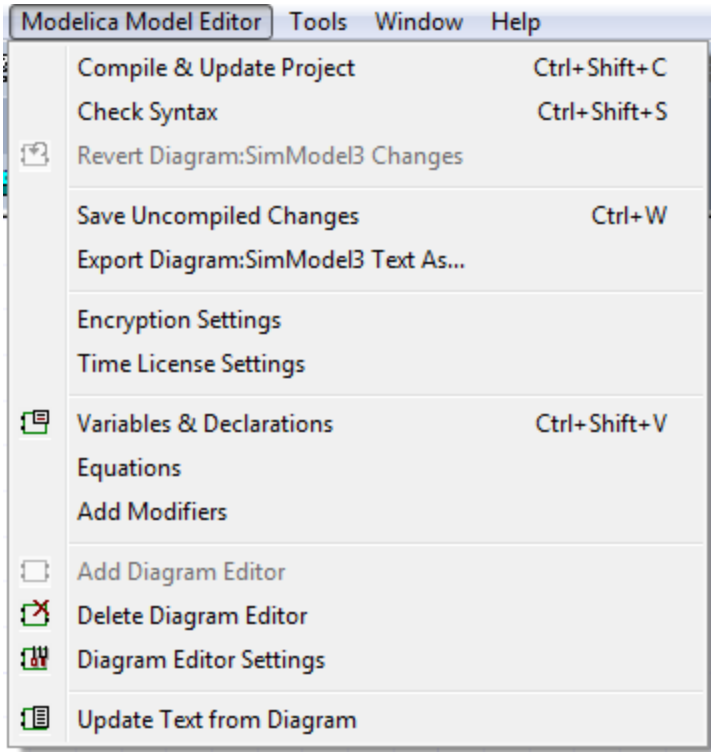
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12 - Diagram Editor

When using the Modelica Language, you can use Twin Builder's Diagram Editor to add and modify models using drag-and-drop actions. You can create and edit the model in a graphical form. As a result, you can

- Construct a model by selecting components from the component libraries and placing them on the Diagram Editor
- Use wires and connectors to connect multiple components
- Modify the properties and parameters of these components

The Diagram Editor is available when you are working with a Modelica model. You can access it through Modelica Model Editor or from the related icons on the **Modelica** ribbon.



Related Topics

Using the Diagram Editor

Operations in the Diagram Editor

Variables & Declarations

Equations

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding or Editing Modifiers

Reverting Diagram Editor Changes

Deleting a Diagram Editor

Modelica Diagram Editor Settings

Updating Text from Diagram

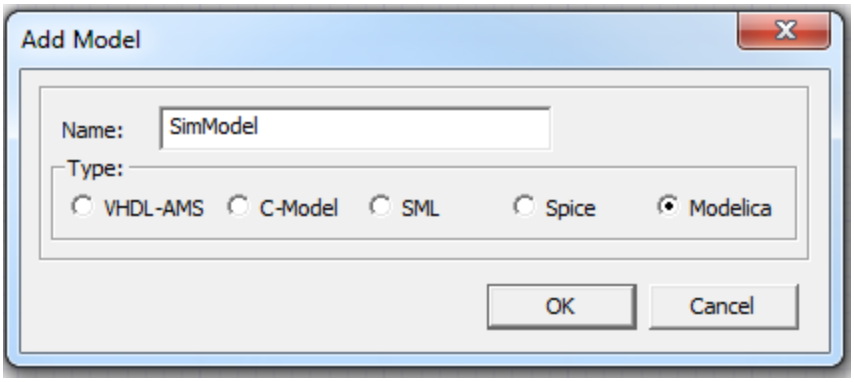
Modelica Model Editor

Using the Diagram Editor to Create a Model

You can use the Diagram Editor to create models.

1. Add a model in one of these ways:
   - Click **Twin Builder** > **Add Model**.
   - Right-click the **Models** folder under **Definitions** in the project manager.
   - Press Ctrl + Shift + M.
   - Click **Tools** > **Edit Libraries** > **Models**. The **Edit Libraries** dialog box appears. Click **Add Model**.

     The **Add Model** dialog box appears with the previous model type selected:



2. Enter the name and click **OK**. The model file opens in the Diagram Editor in a separate tab.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> If an editable model has been imported, you must add a new Diagram Editor. Either click the **Add Diagram Editor** icon in the toolbar, or select **Modelica Model Editor** > **Add Diagram Editor**.

3. From the right side of the Diagram Editor, drag a component to the Diagram Editor. Press **Esc** to complete the action.
   - To move a component after placing it, select and drag it.
   - To rotate a component before placing it, press R before releasing the mouse click.
   - To rotate a component after placing it, select it and press Ctrl+R, which rotates it by 90°, or right-click the component and select **Rotate**.

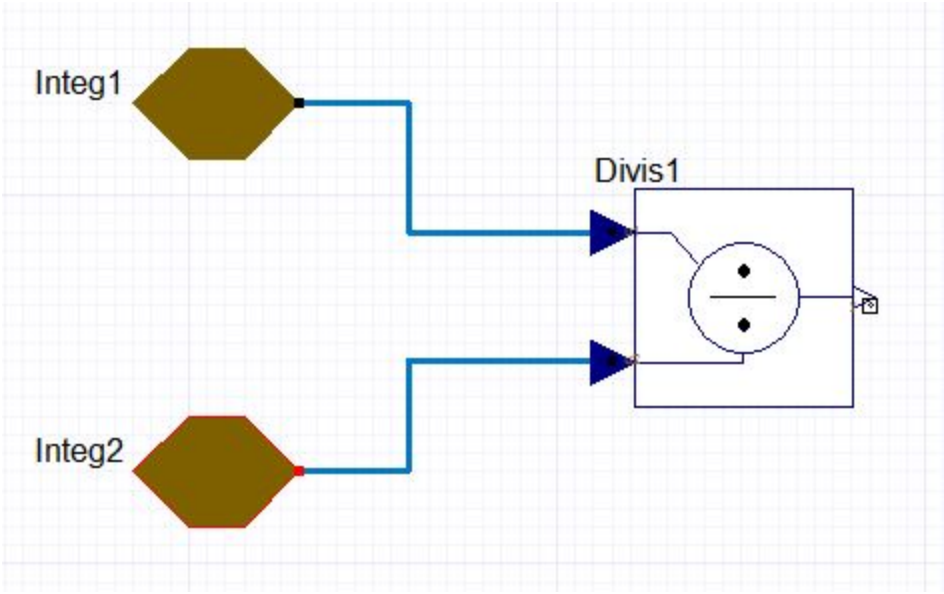
     > **Note:**
     >
     > When you rotate a block or component, the Diagram Editor removes wires connected to unselected components. You must reconnect the blocks and components.

   - To delete a component, select it and press **Delete**, or right-click and select **Delete**.

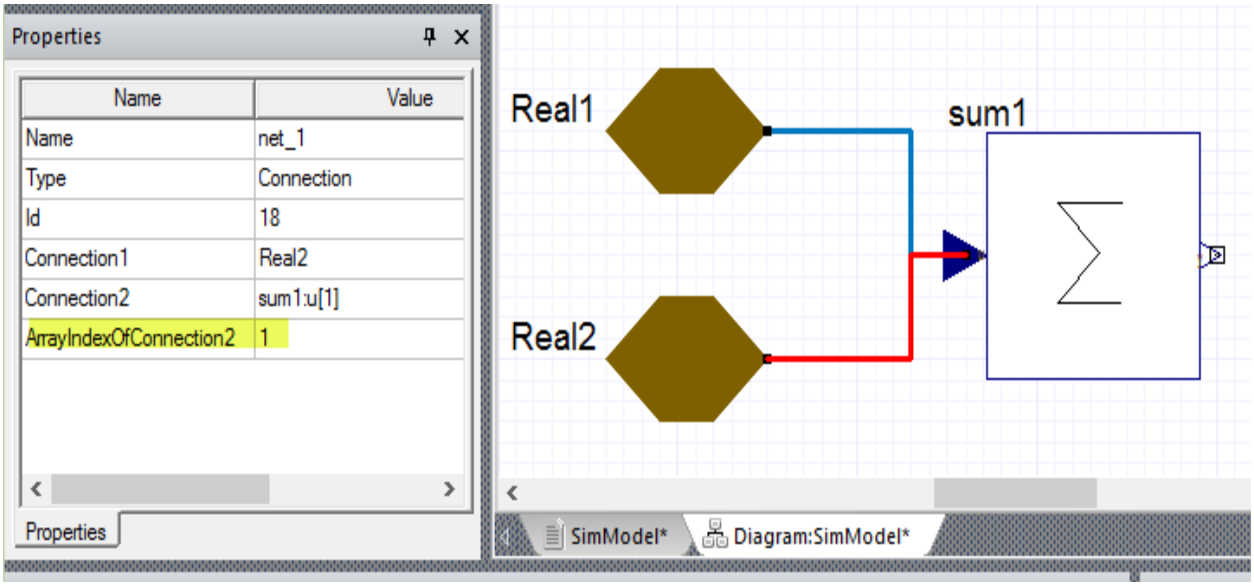
     > **Note:**
     >
     > When you delete a block or component, the Diagram Editor removes all connected wires.

   For more information, see ***Component Libraries***.
4. After arranging the components, use wires and connectors to connect the components. Click the input/output pins of the component and connect them to the desired pins.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



5. For Modelica models, if the component has an array interface port and you make a connection, you can select the connection and change the default value of 1 for "ArrayIndexOfConnectionX" in the **Properties** window.



6. After you complete the diagram, edit the properties and parameters of the components in the **Properties Manager** dialog box.
7. Click **Modelica Model Editor** > **Update Text from Diagram** or click the **Update Text from Diagram** icon on the **Modelica** ribbon. For more information, see Updating Text from Diagram.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Diagram Editor

Zooming and Panning

Operations in the Diagram Editor

Variables and Definitions

Reverting Diagram Editor Changes

Deleting a Diagram Editor

Modelica Diagram Editor Settings

Updating Modelica Text from Diagram

Modelica Model Editor

Zooming and Panning the View

You can magnify (zoom in) or shrink (zoom out) the contents in the view window, or in a rectangular area in the view window. You can pan (scroll) the view in any direction. You can also fit the contents within a window and redraw a window's contents.

Zooming In and Out on the Window Contents

1. Follow this proceduret o magnify or shrink the contents in the view window:
   - Click **Zoom In** or **Zoom Out** on the **View** menu.
   - Right-click in the schematic window and select **Zoom In** or **Zoom Out**.
   - Click the **Zoom In** or the **Zoom Out** icon on the **View** ribbon.
   - Use **Ctrl** + **middle mouse button** and drag to zoom in and out.

     Alternatively, if you are using the **Enable Legacy View Navigation** option in **General** > **User Interface**, you can also use the legacy mouse-button and hotkey combination for zooming. Hold down the **Alt** + **Shift** keys as you click and drag using the left mouse button.

   - Roll the mouse wheel. Make sure to select the **Zoom with mouse wheel** check box in the **Options** dialog box, **Schematic Editor** > **General** section. See ***The Schematic Editor Window*** for more information on interacting with the Schematic Editor.

     Alternatively, if you choose to clear the **Zoom with mouse wheel** option, then hold **Shift** while rolling the mouse wheel to zoom in or out. Rolling the wheel without *Shift* will scroll the view in this case.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. The view zooms to the chosen magnification. The absolute size of the model does not change.
3. Repeat the operation until you achieve the desired magnification.

> **Note:**
>
> Zooming in and out is limited to approximately 20 steps. The **Zoom In** or the **Zoom Out** icon on the **View** ribbon is dimmed when you reach the corresponding zoom limit.

Zooming In on a Rectangular Area

1. To magnify a specific rectangular area in the view window do one of the following:
   - Select **View** > **Zoom Area**.
   - Right-click the schematic window and select **Zoom Area**.
   - Select **View** > **Zoom Area** on the ribbon.

   The cursor changes to a magnifying glass.
2. Draw a rectangle to increase magnification in that area.

Restoring a Previous Zoom View

After zooming in or out, select **View** > **Zoom Previous** to return to the previous magnification level.

Fitting the Drawing Border to the Window

1. When a border is added to a drawing, click **View** > **Fit Border** to scale the bordered contents to fit within the current window. You can also click Ctrl+B to fit the border.
2. The view zooms to fit the bordered contents of the drawing within the bounds of the window. The absolute size of the model does not change.

Panning the View

To pan (scroll) the view in any direction:

- Select **View** > **Pan**. The cursor changes to a hand as you move it over the schematic window. Click and drag to pan the view in any direction.
- **Ctrl + middle-click** and drag to pan the view in any direction.
- If the *Enable Legacy View Navigation* option is selected in the User Interface Options, you can also use the legacy combination (**Shift + click**) and drag the cursor to pan the view in any direction.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Click on the ribbon. The cursor changes to a hand as you move it over the schematic window. Click and drag to pan the view in any direction.
- In the Symbol and Schematic Editors, hold Shift or Ctrl and roll the mouse wheel to pan vertically or horizontally. Make sure to select the **Zoom with mouse wheel** check box in the **Options** dialog box, **Schematic Editor** > **General** section. See ***The Schematic Editor Window*** for more information.

Redrawing a View

Click **View** > **Redraw** to redraw the current window.

Operations in the Diagram Editor

When the Diagram Editor is active, use the commands on the **Edit** menu and right-click menu to perform various operations on elements in the Diagram Editor. Available commands vary based on the current selection and on the previous command.

- **Rotate** – Rotate a selected object or group of objects 90° counterclockwise.
- **Flip Vertical** – Flip the selected object or group of objects around the X-axis. To flip multiple objects vertically:

1. Select (Ctrl+click) the objects you want to flip vertically.

> **Note:**
>
> The first object selected is highlighted in red; each subsequently selected object is highlighted in dark red.

2. Select **Edit** > **Flip Vertical**, or right-click the selected objects and select **Flip Vertical**.

- **Flip Horizontal** – Flip the selected object or group of objects around the Y-axis. To flip multiple objects horizontally:

1. Select (Ctrl+click) the objects you want to flip horizontally.

> **Note:**
>
> The first object selected is highlighted in red; each subsequently selected object is highlighted in dark red.

2. Click **Edit** > **Flip Horizontal**, or right-click the selected objects and select **Flip Horizontal**.

- **Undo** – Cancel the last operation.
- **Redo** – Re-execute the last operation that was undone.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Cut** – Remove the selected objects and retain a copy for pasting into a diagram in the same application.
- **Copy** – Create a local copy of selected objects for pasting into a diagram in the same application.

> **Note:**
>
> **Copy** will not copy variables or equations.

- **Paste** – Place objects from the previous **Copy** or **Cut** operation into the schematic.
- **Delete** – Delete the selected objects.
- **Select All** – Select all the objects on the diagram.
- **Push Down** – Select an object on the diagram, right-click and select **Push Down** to open the hierarchy diagram of that object, if one is available.
- **Pop Up** – Right-click the diagram and select **Pop Up** to go one level up in the hierarchy. This option is available if you are not at the top level of the hierarchy.

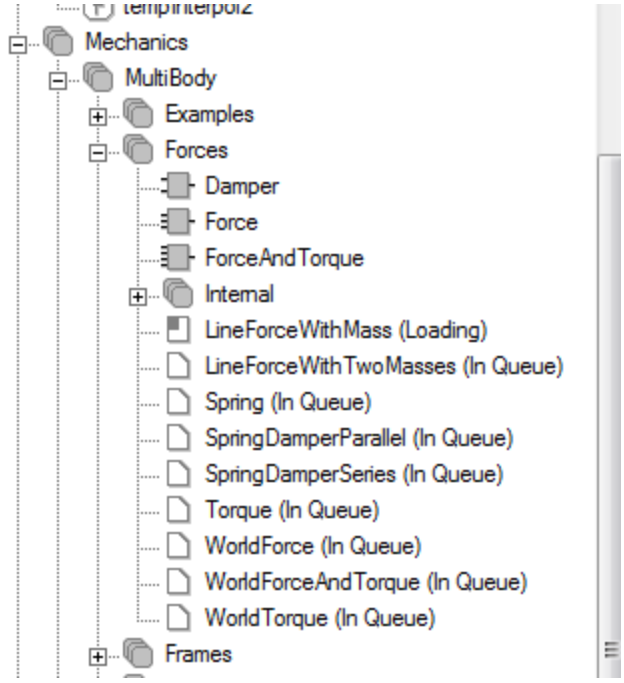
Component Libraries

The Component Libraries contain a list of all blocks and models available in the language-specific packages that have been loaded in the Model Editor.

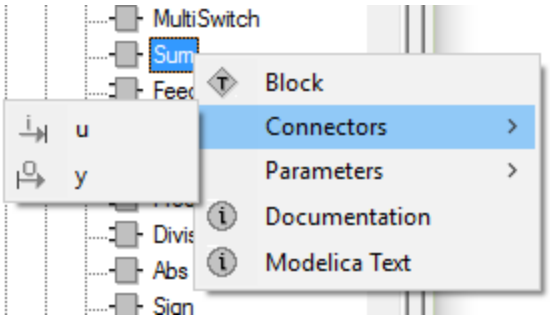
Modelica packages contain models, blocks, types, classes, and other objects. The objects are typically in hierarchical order.

Initially there is no information available for each object, but when you expand a hierarchical level, the compiler begins providing more information, such as interfaces and properties, for each element.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Right-click the model to view the component's type, associated input and output pins, parameters. and documentation.



Project Package

The project package shows all local Modelica models in a project. You can drag and drop models to connect them with other objects.

Select **Add Model** when creating definitions to extend from existing models.

The project package updates when you compile or save an uncompiled model. Remember to update the model text in the diagram before saving.

To export the project package to a single file package, right-click the project definition and select **Export**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:** You cannot instantiate a component into itself.

> **Note:** The project definitions will not show SML, VHDL, and other language models. It will also not show Modelica models that were imported from a package.

Related Topics

Modelica Model Editor Options

Diagram Editor

Using the Diagram Editor

Variables & Declarations

Use the **Variables & Declarations** dialog box under **Modelica Model Editor** to add, edit, or remove language-specific variables.

Related Topics

Adding a Variable

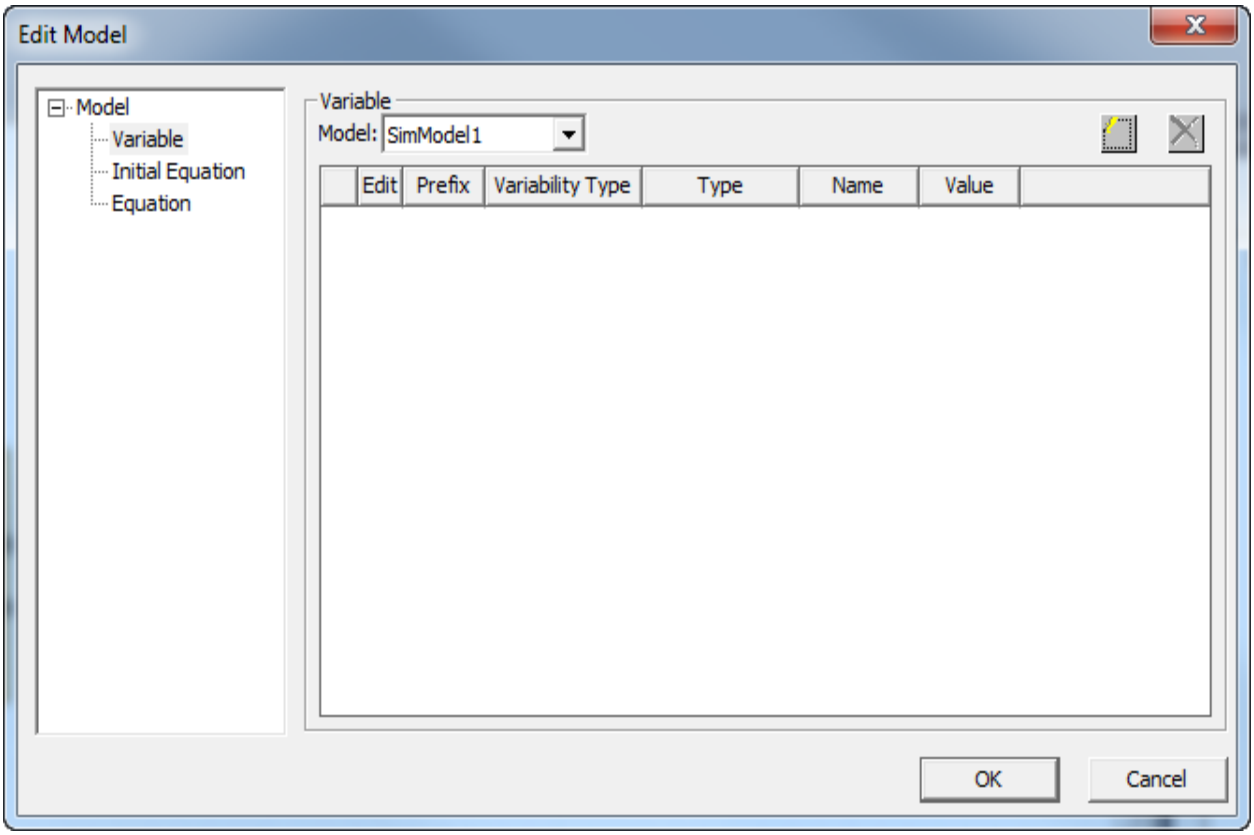
Editing a Variable

Removing a Variable

Adding a Variable

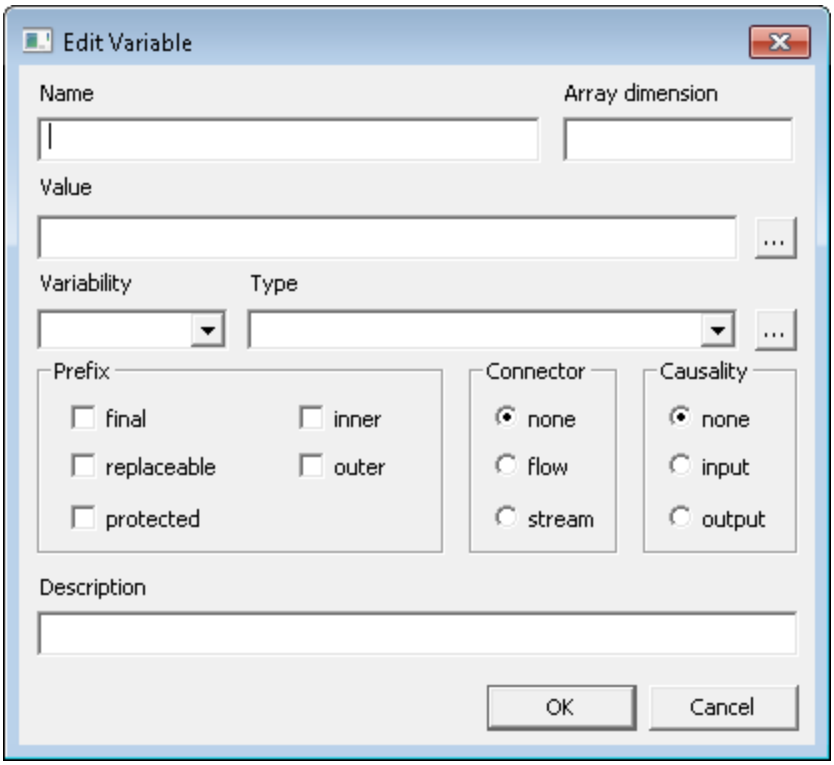
1. Click **Modelica Model Editor** > **Variables & Declarations** to open the **Edit Model** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. Click to add a row. The **Edit Variable** dialog box opens.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- **Name** – Enter a name for the variable.
- **Array dimension** – Enter the dimensions for the array with commas as separators. An example of a one-dimension array is 10. An example of a two-dimension array, like for matrices, is 10,5. An example of a four-dimension array is 10,5,2,6.
- **Value** – Enter a value for the variable or click ... to select a function call from the available packages.
- **Variability** – Select an option from the drop-down list: **parameter**, **constant**, **discrete**.
- **Type** – Select an option from the drop-down list (**Real**, **Integer**, **Boolean**, **String**), or click ... and select a type from the available packages.
- **Prefix** – Add one or more prefixes from the available items: **final**, **replaceable**, **protected**, **inner**, **outer**.
- **Connector** – Select **none**, **flow**, or **stream**.
- **Causality** – Select **none**, **input**, or **output**.
- **Description** – Enter a description of the variable.

4. Click **OK**.
5. To add this variable to the model text, click **Modelica Model Editor** > **Update Text from Diagram**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Editing a Variable

Click in the **Edit** column on the **Equation** panel

to edit a variable. Make your changes in the **Variable** dialog box and click **OK**.

Removing a Variable

Highlight a variable row in the **Equation** panel and click to remove it.

In addition to the above, you can also use variables & declarations to edit equations and initial equations.

Variables & declarations also shows extended base class parameters, if any exist. You can change and modify these parameters.

Related Topics

Creating a Variable by Editing the Component Parameter Properties

Creating a Variable by Dragging a Type Object

Diagram Editor

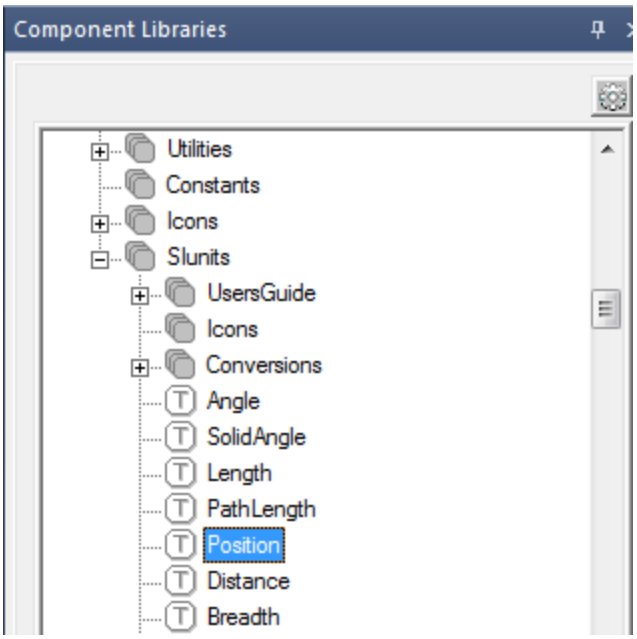
Using the Diagram Editor

Creating a Variable by Dragging a Type Object

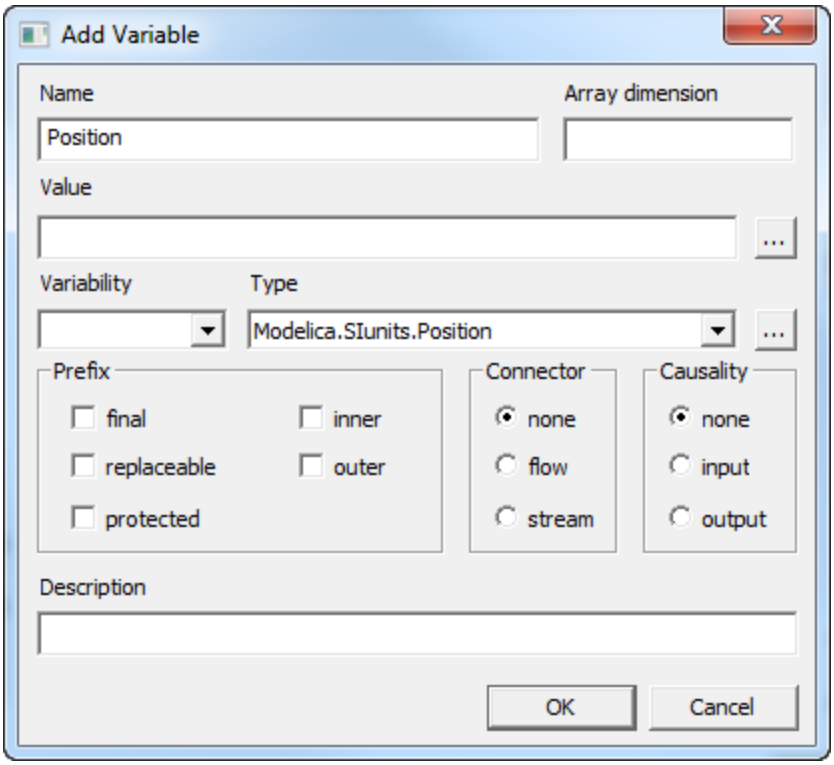
Create a variable in Modelica for *Type* objects by dragging the object onto the Diagram Editor.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select a Modelica *Type* object such as **SIunits** > **Position**.



2. Drag the object to the Diagram Editor. The **Add Variable** dialog box opens with the **Name** and **Type** boxes defined:



3. Edit the options as needed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

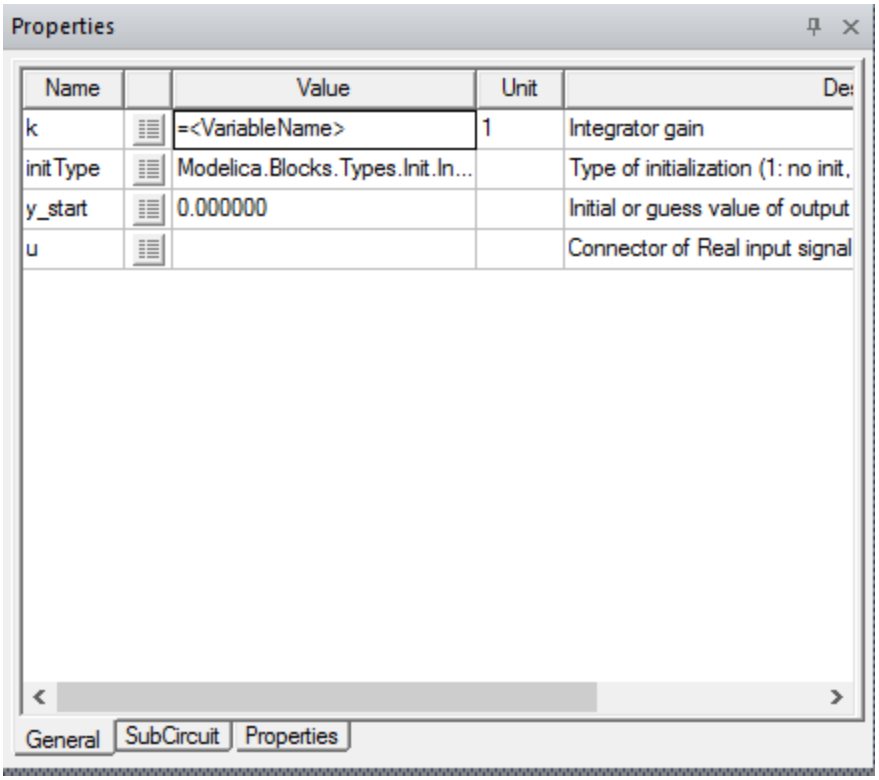
4. Click **OK**.

Creating a Variable by Editing the Component Parameter Properties

Create a variable in Modelica by editing the value in a component's parameters **Properties** dialog box.

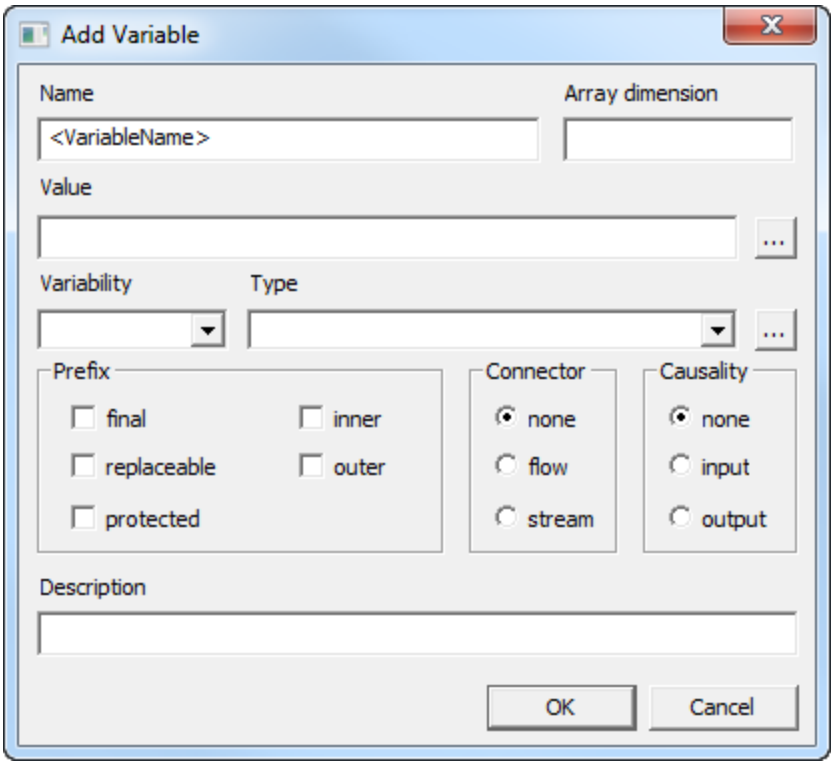
> **Note:**
>
> You can also use this method to select multiple components that have common properties and create a variable to change values in all selected instances.

1. Select a Modelica object and drag it into the Diagram Editor.
2. Enter a variable name in the **Value** field in this format: =<[*VariableName*]>, as in the following example.



3. Press **Enter**. The **Add Variable** dialog box opens.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. Edit the options as needed.
5. Click **OK**.

Initial Equation

Use the **Equations** option under **Modelica Model Editor** to add, edit, or remove language-specific equations used for initialization.

Adding an Initial Equation

Editing an Initial Equation

Removing an Initial Equation

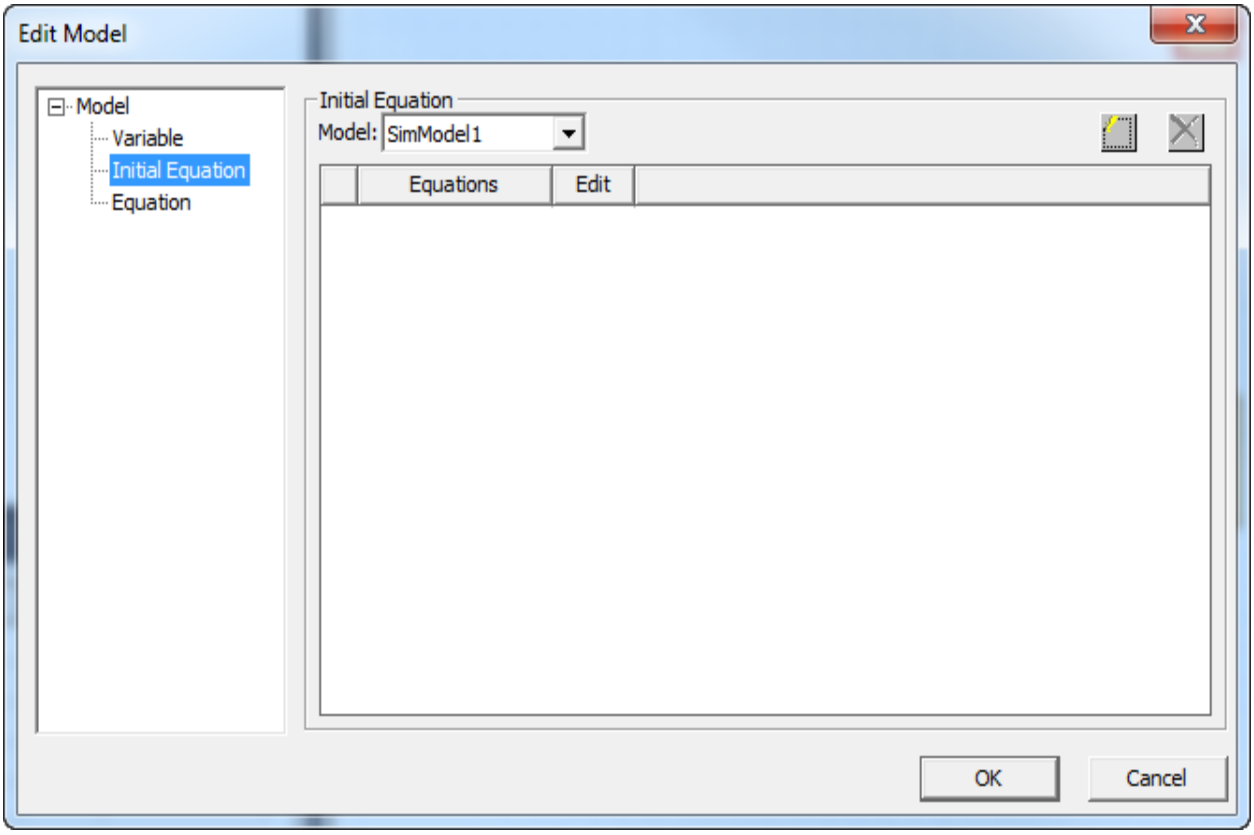
Adding an Initial Equation

1. Click **Modelica Model Editor** > **Equations**. The **Edit Model** dialog box opens.
2. Click **Initial Equation** to open the **Initial Equation** panel.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

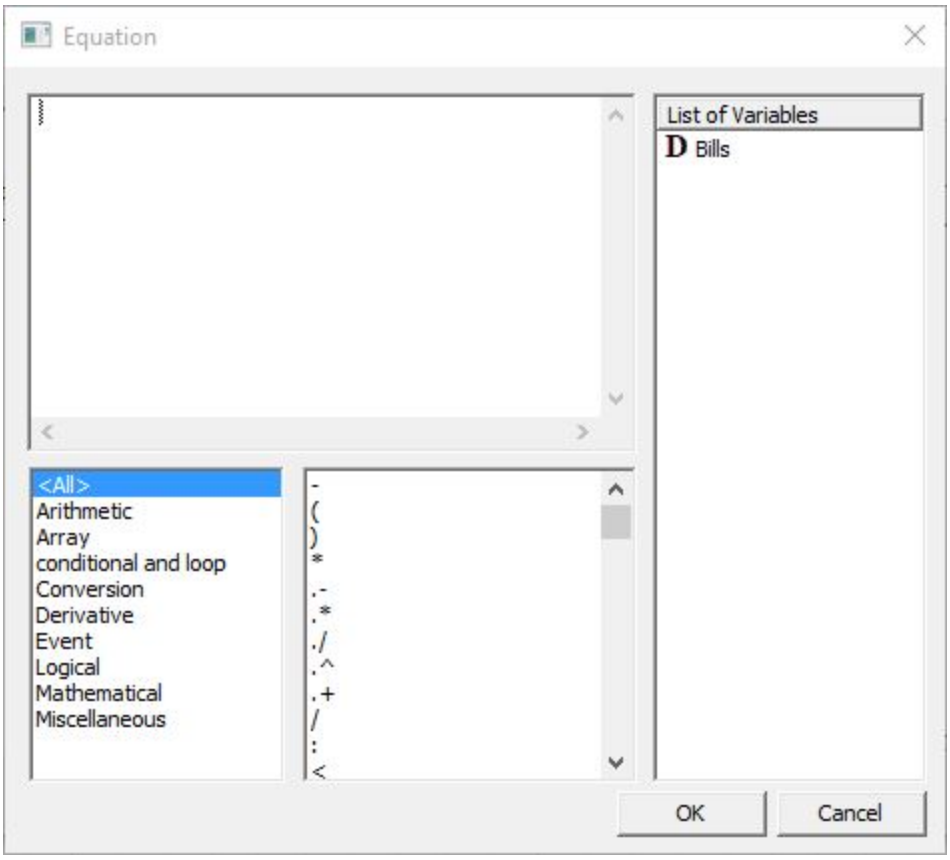
Note:

You can also access this panel from **Modelica Model Editor** > **Variables & Declarations** > **Initial Equation**.



3. Click to add a row. The **Equation** dialog box opens.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. Use the options to create the initial equation. Select from the language constructs and functions. On the right is a list of available local variables and component names.
5. After you create the equation, click **OK**. The equation is added to the **Equations** list.
6. Click **OK**.
7. To add this equation to the model text, click **Modelica Model Editor** > **Update Text from Diagram**.

Editing an Initial Equation

Click ... in the **Edit** column on the **Equation** panel Edit ... to edit an equation. Make your changes in the **Equation** dialog box and click **OK**.

Removing an Initial Equation

Highlight an initial equation row in the **Equation** panel and click to remove it.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Diagram Editor

Using the Diagram Editor

Equations

Use the **Equations** option under **Modelica Model Editor** to add, edit, or remove language-specific equations.

Adding an Equation

Editing an Equation

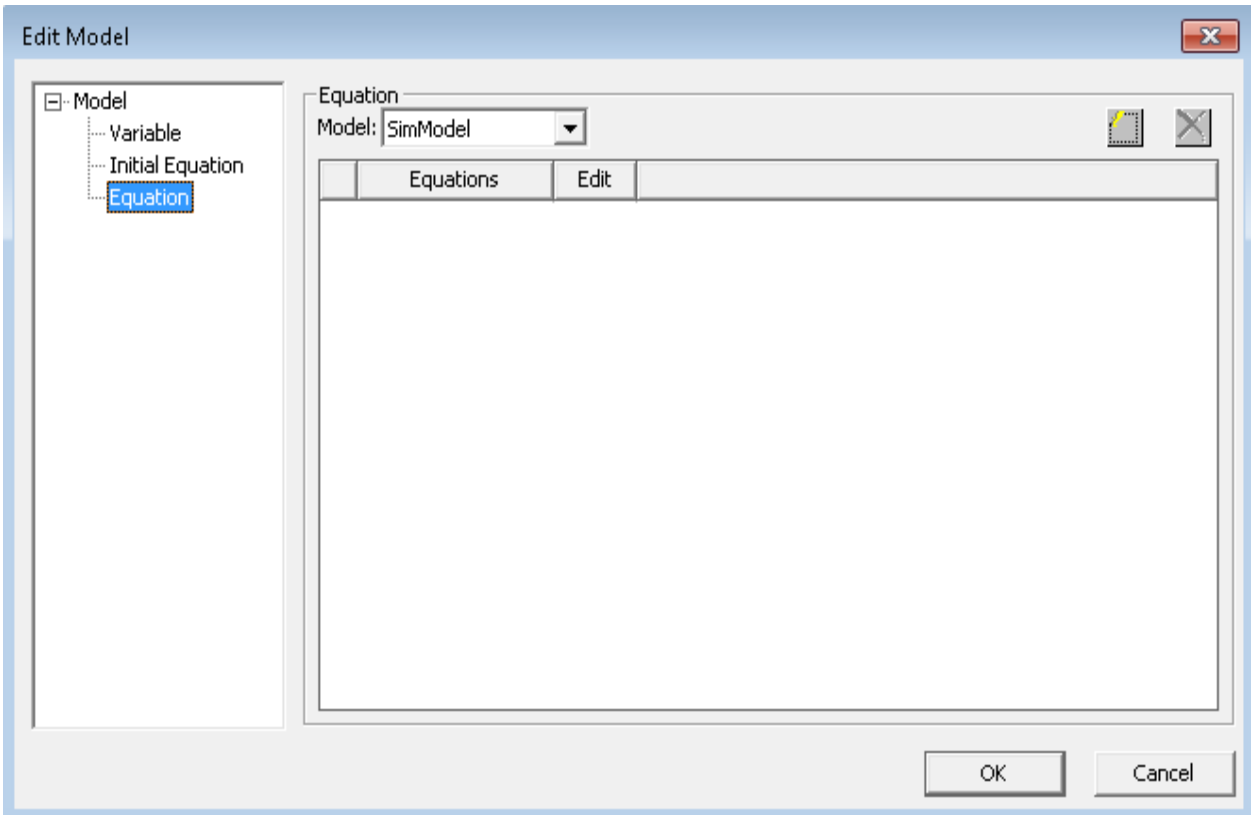
Removing an Equation

Adding an Equation

1. Click **Modelica Model Editor** > **Equations**. The **Edit Model** dialog box opens.

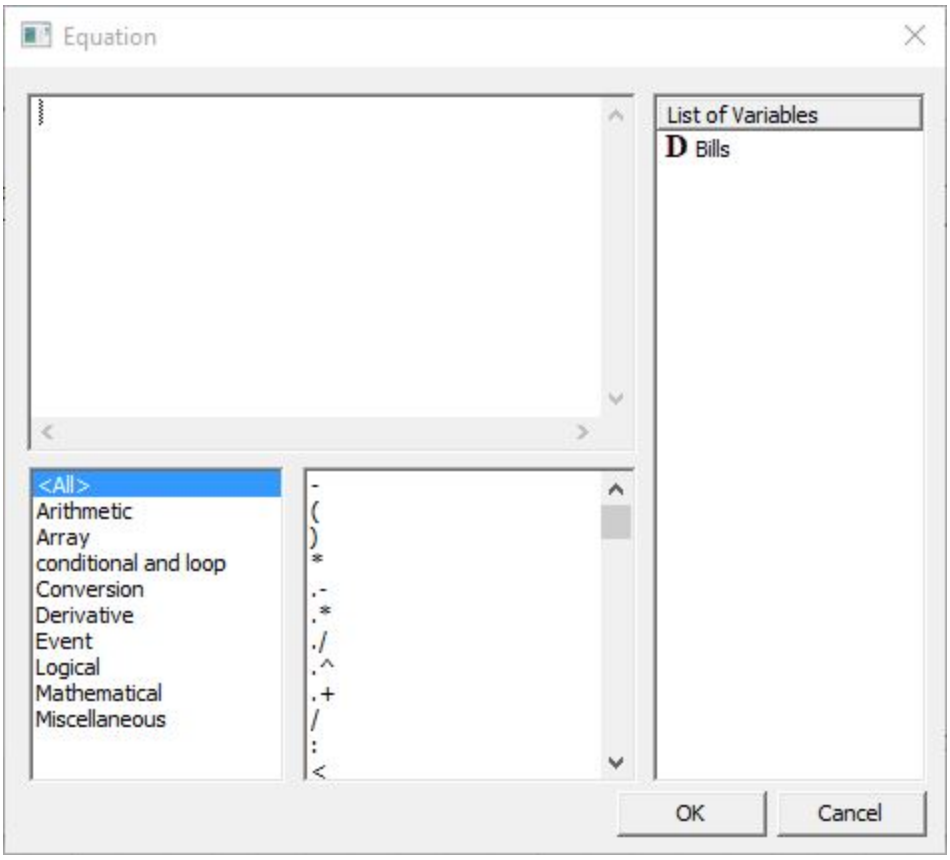
> **Note:**
>
> You can also access this panel from **Modelica Model Editor** > **Variables & Declarations** > **Equation**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. Click to add a row. The **Equation** dialog box opens.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



3. Use the options to create the equation. Select from the language constructs and functions. On the right is a list of available local variables and component names.
4. After you create the equation, click **OK**. The equation is added to the **Equations** list.
5. Click **OK**.
6. To add this equation to the model text, click **Modelica Model Editor** > **Update Text from Diagram**.

Editing an Equation

Click ... in the **Edit** column on the **Equation** panel

to edit an equation. Make your changes in the **Equation** dialog box and click **OK**.

Removing an Equation

Highlight an equation row in the **Equation** panel and click to remove it.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

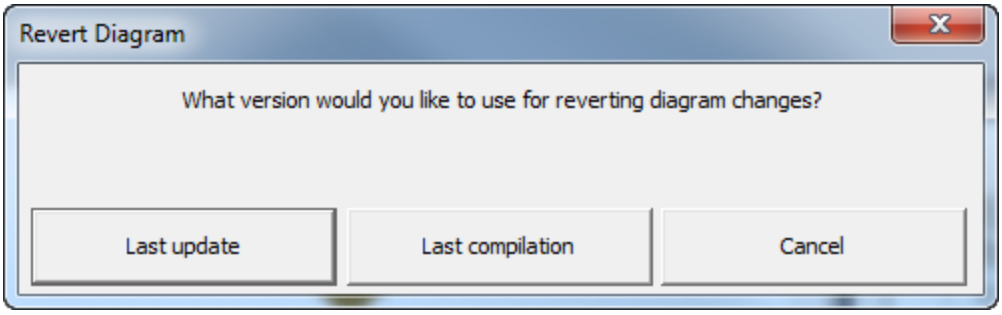
Diagram Editor

Using the Diagram Editor

Reverting Diagram Editor Changes

Follow this procedure to revert current changes and go back to the previous update or last compilation of user-generated model text from the Diagram Editor.

- Click **Revert Changes** on the Modelica ribbon.
- Click **Modelica Model Editor** > **Revert Diagram Changes**.



You have the option to revert to the **Last update** or the **Last compilation** of the model text you generated in the Diagram Editor.

Related Topics

Diagram Editor

Using the Diagram Editor

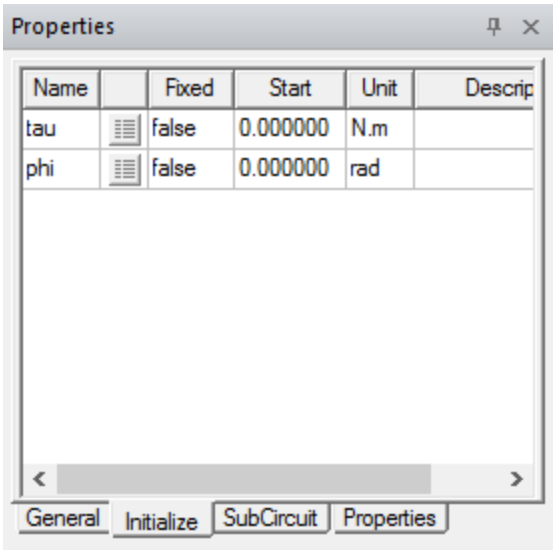
Editing Properties

Use **Edit Properties** in the **Modelica Diagram Editor** to change parameter values and attributes, as well as to parameterize hierarchy.

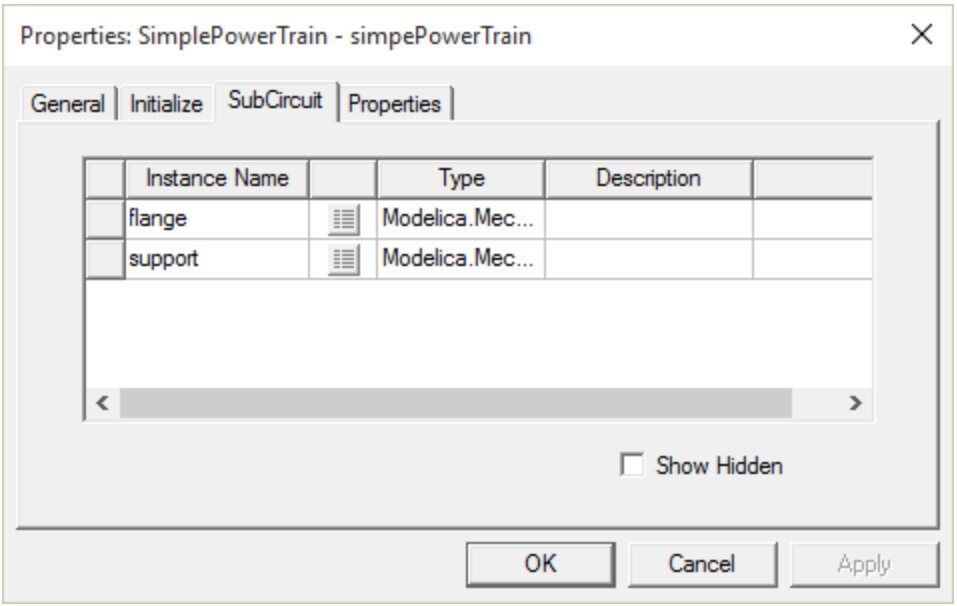
You can change parameters in either the property window or property dialog box. These can be accessed in one of the following ways:

- Select an object in the diagram editor. Its properties appear in the **Properties** dialog box. Click **View** > **Properties** to toggle this dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- Right-click an object and select **Edit Properties** to open the **Properties** dialog box.
- Double-click an object to open the **Properties** dialog box.



Editing Attributes / Hierarchy

Click to open a window for editing a parameter's attributes. For object instances it will bring up hierarchy for editing. Parameters appear on the following tabs:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Initialize** – List variable instances where you can change fixed and start attributes.
- **SubCircuit** – List all object instances where you can modify hierarchy.
- **Properties** – List all standard object properties, including:
  - **Instance Name** – Change the instance name.
  - **Instance prefix** – Provide any prefix required for the given object, such as **Final protected** and so on.
- Instance prefix – Provide any prefix required for the given object, such as **Final protected** and so on.

Add Modifier

Changes made here appear in the **Add Modifier** dialog box as well.

Related Topics

Diagram Editor

Using the Diagram Editor

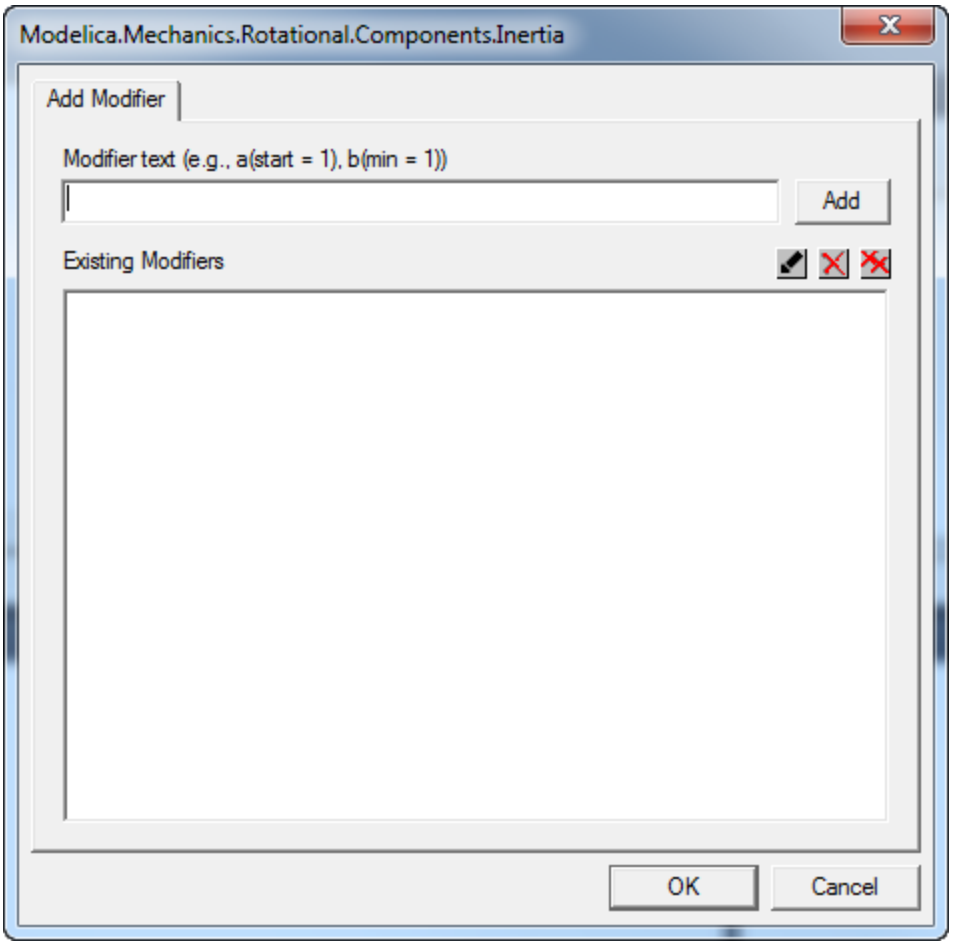
Add Modifiers

Use **Add Modifiers** in the **Modelica Diagram Editor** to add or edit modifiers for components. Modifier text is a comma-separated list of names (attribute/parameter) and associated values, such as *a(start = 100, fixed = true)*. Use this option to change attribute values such as start or fixed. You can also use it to change parameters in a hierarchy.

To add a modifier in Modelica Diagram Editor

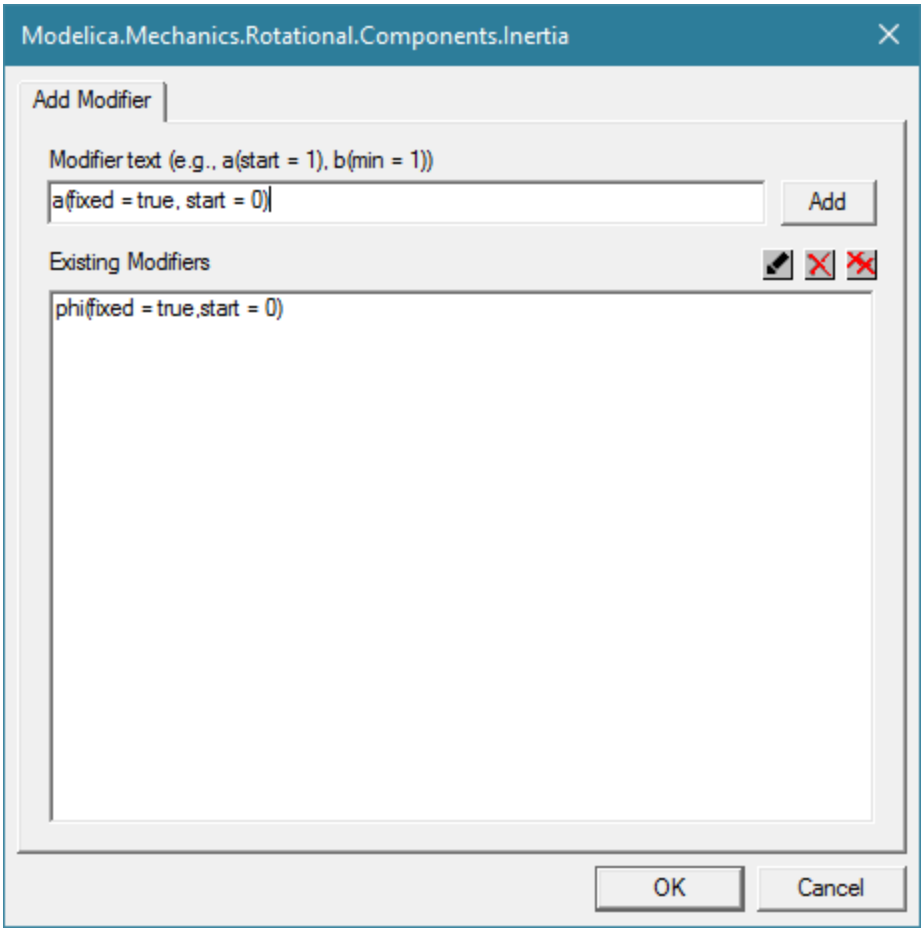
1. In the Diagram Editor, right-click a component and select **Add Modifier**, or select **Modelica Model Editor** > **Add Modifier**. The **Add Modifier** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



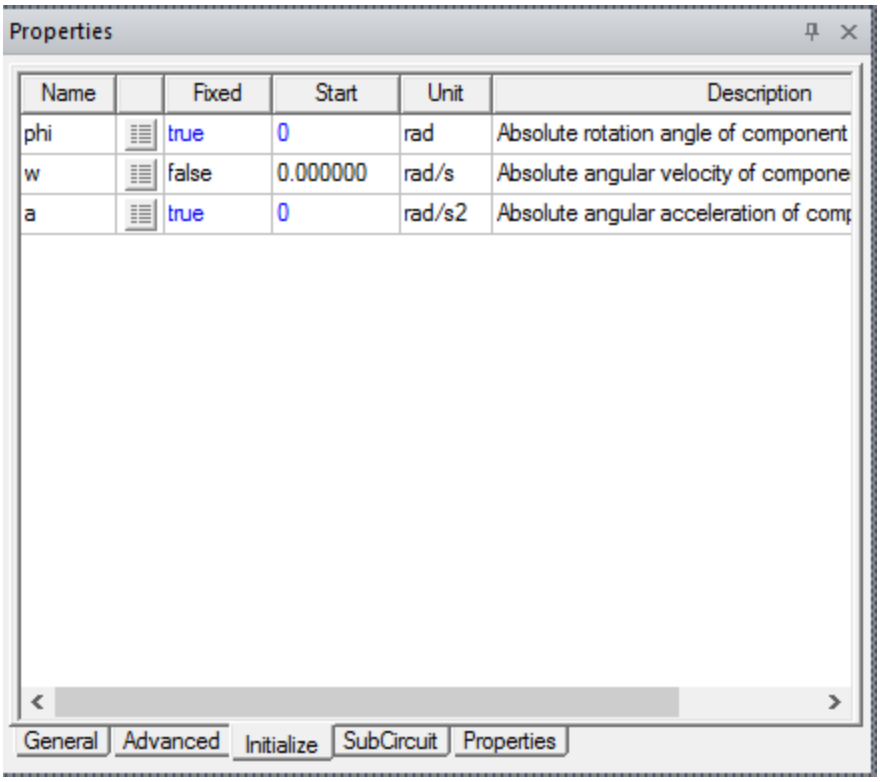
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Enter the modifier text in the **Modifier text** box and click **Add**.



3. Click **OK** to add your changes to the component. The added modifiers also appear in the parameter properties for the component.

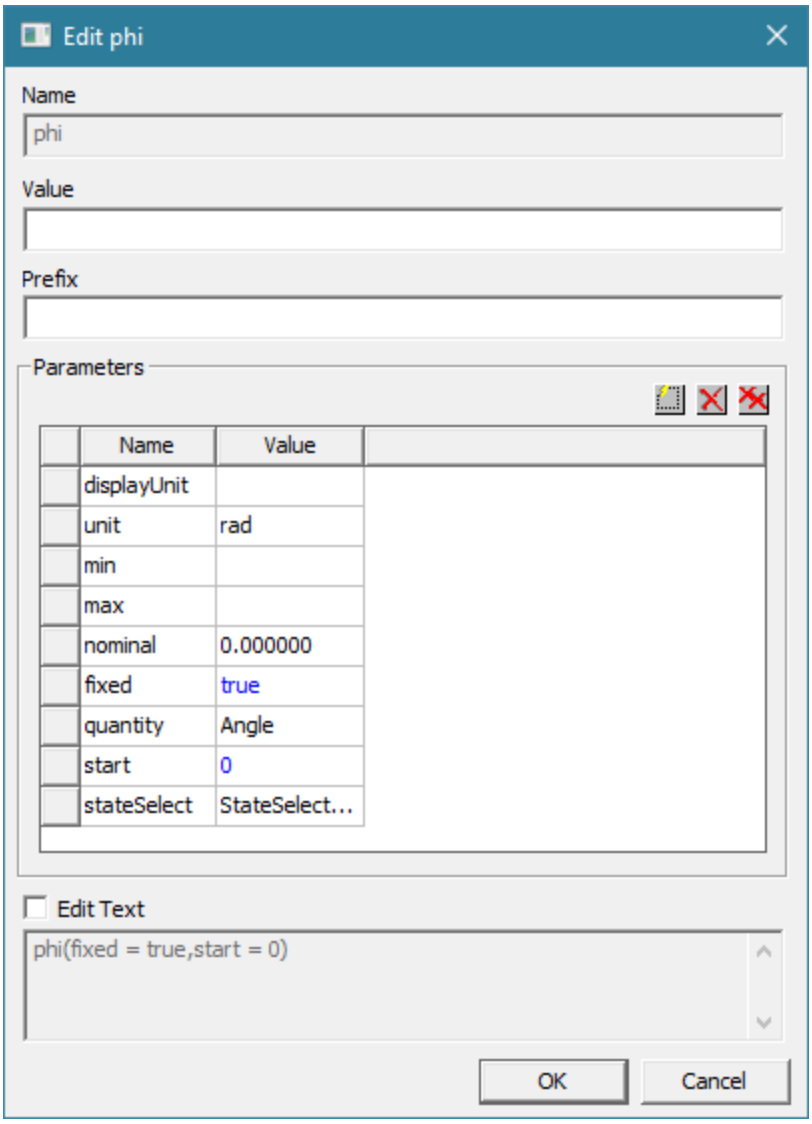
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



To edit a modifier in Modelica Diagram Editor

1. In the Diagram Editor, right-click a component and select **Add Modifier** or click **Modelica Model Editor** > **Add Modifier**. The **Add Modifier** dialog box appears.
2. Double-click the existing modifier you want to edit, or select the existing modifier and click to open the **Edit Modifier** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



3. You can perform the following actions.
   - Edit the name or value directly.
   - Click the **Edit** box and make the edits to the modifier.
   - Click to add a new name and value to the modifier.
   - Click to remove a modifier.
   - Click to remove all modifiers.
4. Click **OK** to accept the changes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Diagram Editor

Using the Diagram Editor

Changing Class

Use this option to change the class of an item on the diagram editor.

1. Right-click an object and select **Change Class** to open the **Select Package Item** dialog box.
2. Select a package item and click **OK** to change the object's class.

> **Note:** The object will keep any changes you made to parameters/modifiers. It will also keep its instance name and prefixes.

Related Topics

Diagram Editor

Using the Diagram Editor

Diagram Object Information

Use these options to show documentation for an object, and to open the Modelica text for an object in a new tab.

Show Documentation

1. Right-click the desired object and select **Show Documentation**.
2. The documentation opens in your browser.

> **Note:**
>
> You can also view object documentation in the package browser. Right-click the desired object and select **Show Documentation**.

Show Modelica Text

1. Right-click the desired object and select **Show Modelica Text**.
2. The Modelica text opens in a new tab (read-only).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You can also view the Modelica object text in the package browser. Right-click the desired object and select **Modelica Text**.

Related Topics

Diagram Editor

Using the Diagram Editor

Deleting a Diagram Editor

When you delete a Diagram Editor, the instance is deleted. However, the model text remains until you add a new Diagram Editor, which overwrites the model text.

To delete a Diagram Editor:

- Click **Delete Diagram Editor** on the **Modelica** ribbon.
- Click **Modelica Model Editor** > **Delete Diagram Editor**.

Related Topics

Diagram Editor

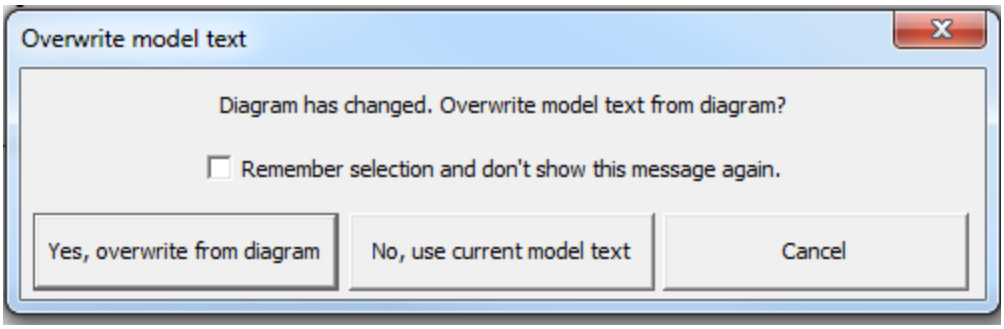
Using the Diagram Editor

Diagram Editor Settings

Follow this procedure to set Modelica Diagram Editor options in Twin Builder:

1. Select **Tools** > **Options** > **General Options**, and expand **Model Editor** > **Modelica**. If the Editor is open, click **Modelica Model Editor** > **Diagram Editor Settings**.
2. Select **General** to display these settings.
   - **Warn user if diagram has changed** – When you select this check box, the **Overwrite model text** dialog box appears after you make changes to a diagram. You can choose to overwrite from the diagram, or to use the current model text and not overwrite from the diagram. You also can cancel. Select the **Remember selection and don't show this message again** check box if you always want the same action when you change a diagram.

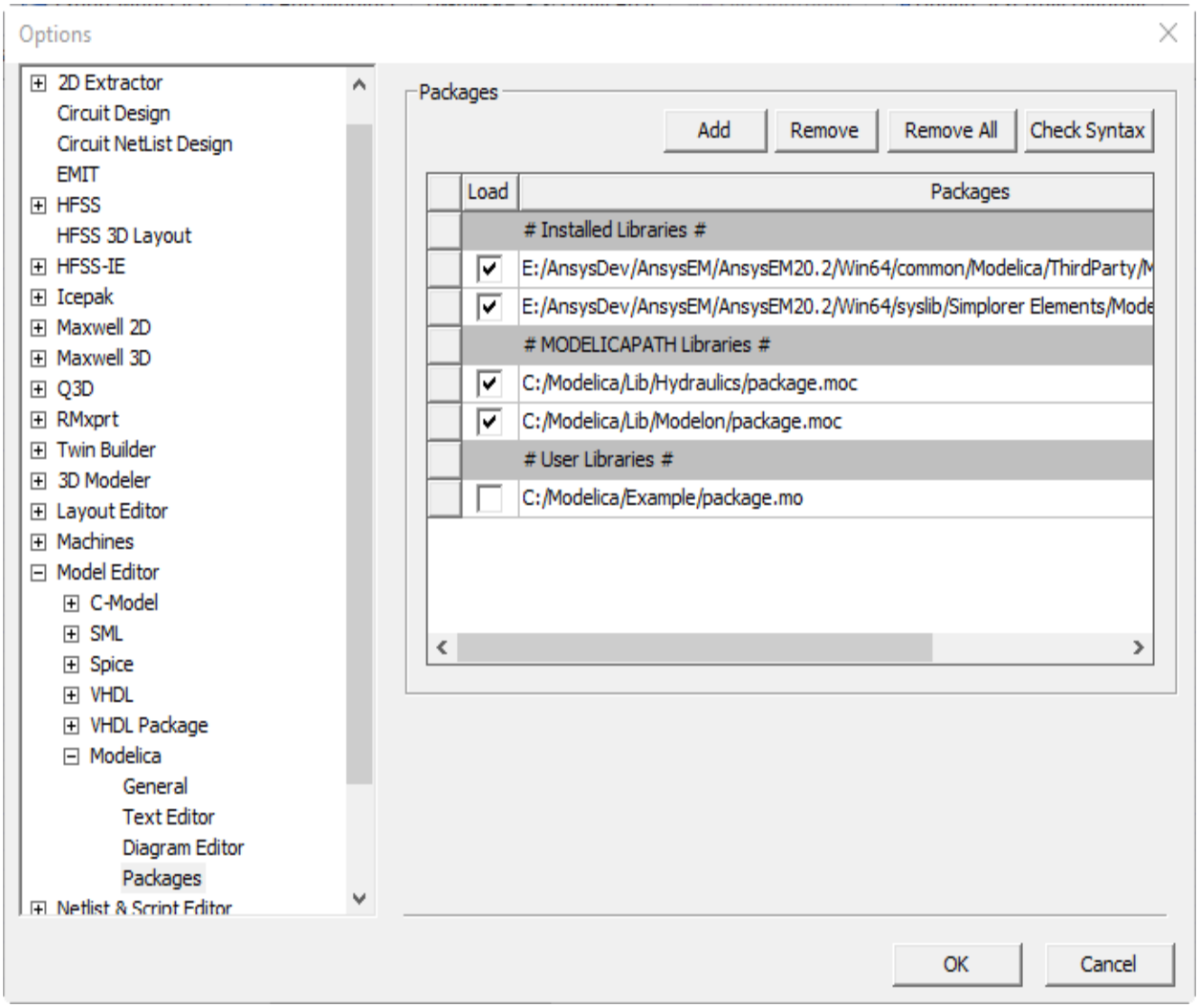
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- **Show diagram tab as default** – Select this check box to display the **Diagram** tab when you start the Editor.
- **Show model interfaces dialog during compilation** – Select this check box to display the **Model Interfaces** dialog box. Use this dialog box to select which interfaces to expose in Twin Builder.
- **Show update component dialog after compilation** – Select this check box to display an update component dialog box after compilation. Use this dialog box to make SDB selections if required.
- **Load packages automatically after editor startup** – Select this check box to start loading all packages when the Editor starts. If you clear this check box, you must expand each library to load individually.

3. Select **Text Editor** to access the standard Twin Builder text editor settings.
4. Select **Diagram Editor** to display these settings.
   - **Show pin names** – Determine whether pin names appear on the diagram.
   - **Rotate property displays on component rotation** – The property display, such as **Instance Name**, rotates when you rotate the component.
   - **Zoom with mouse wheel** – See ***Zooming and Panning the View*** for details on zooming in the Diagram Editor.
   - **Refresh component on parameter change** – If you select this check box, changing a parameter refreshes the component icon (add/remove ports) and properties (enable/disable). If you clear this check box, you can refresh whenever you choose (that is, you can right-click a component and select **Refresh**).
5. Select **Packages** to load installed libraries, MODELICAPATH libraries, and user libraries. Only packages selected to load appear in the package browser.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- The following libraries are provided with installation:
  - **Installed Libraries** – The libraries included in the installation.
  - **MODELICAPATH Libraries** – Set the system environment variable MODELICAPATH and indicate its location. In the example above, the MODELICAPATH environment variable is set to add the PowerSystems library. Select the **Load** check box to load the libraries, or clear the check box if you do not want to load them.
  - **User Libraries** – Click **Add** and find **package.mo** or **package.moc** libraries. Click **Remove** to remove a user library.
- When checking the syntax of a package, keep in mind:
  - The system checks any package being added to the list for syntactical correctness before it is loaded in the package browser. If the syntax check fails, the package is still added to the list, but the check box in the **Load** column is cleared. This way the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

package is registered in the list and can later easily be turned on after the errors have been fixed.

- When the **Load** check box is selected, its associated package is checked for syntactical correctness.
- Click **Check Syntax** to execute a syntax check for a selected package.

> **Note:**
>
> This check is only a syntax check which allows the package to be scanned correctly by the Modelica compiler. This check does not flag semantic errors in the package.

6. When finished configuring settings, click **OK**.

Related Topics

Diagram Editor

Using the Diagram Editor

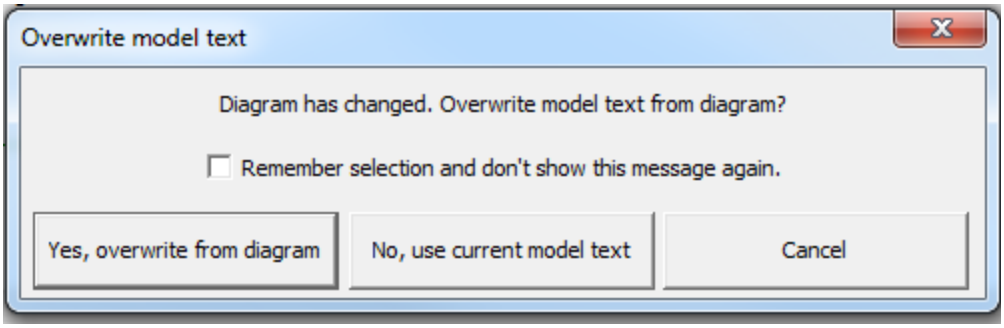
Extend a Modelica Model or Diagram

Updating Modelica Text from Diagram

Changes in the diagram reflect in the model text only after using **Update Text from Diagram**.

- Click **Update Text from Diagram** on the **Modelica** ribbon.
- Click **Modelica Model Editor** > **Update Text from Diagram**.

The model created or edited using the Diagram Editor can be saved and used in the project only after the model text is updated. If you do not use the option before switching tabs to the model text, the **Overwrite model text** dialog box gives you the option to make the diagram changes reflect into the model when you click **Yes, overwrite from diagram**.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

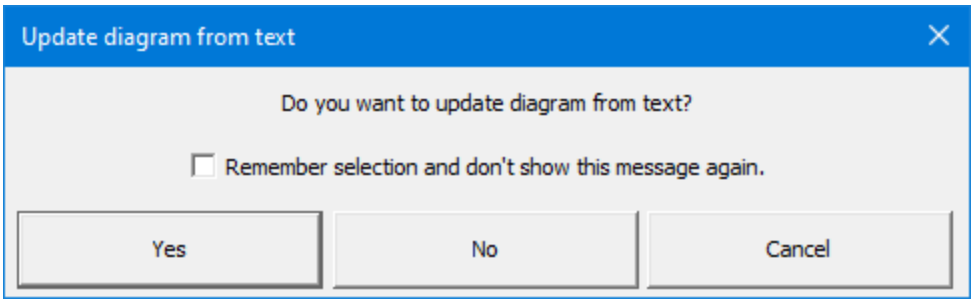
Diagram Editor

Using the Diagram Editor

Updating Modelica Diagram from Text

Changes in the text reflect in the diagram only after using **Update Diagram from Text**.

The system prompts you to **Update Diagram from Text** when there are text changes, and you change the tab to diagram. Select the **Remember selection and don't show this message again** check box to automatically update the diagram when you've made text changes.



Click **Yes** to update the diagram with following information: extend, variable declarations, component modifications, algorithms and equations.

There are a few limitations where it the diagram will not preserve changes made in text editor; you should therefore avoid making these types of changes in the text editor:

- Annotations are not preserved and will be overwritten by the diagram. For example, any graphical annotation changes related to components or connections.
- Conditional components are not supported.
- Local class definitions or nested classes.
- Constrained by clauses.
- Array of components.
- Short class definitions, for example, `type InArgument = input Real;`.

Updating Modelica Library Definitions

To update your Modelica diagram definitions when Modelica libraries or product version changes:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Modelica Model Editor** > **Update Library Definitions**. The **Update Library Definitions** dialog box appears.
2. Select **Update Object Properties** to update changes related to properties of an object while maintaining user-made changes to properties.
3. Select **Update Object Icon** to make changes to model icons if there are changes in a library. If a connector is moved or deleted, the existing connection from that connector is removed.

The **Log** panel displays possible issues with an update. For example:

- If an object is not found in loaded packages in the package browser, it will be listed in the log.
- If a connection is removed, it will be added to log. Selecting it will highlight the connectors where the connection was removed.

Related Topics

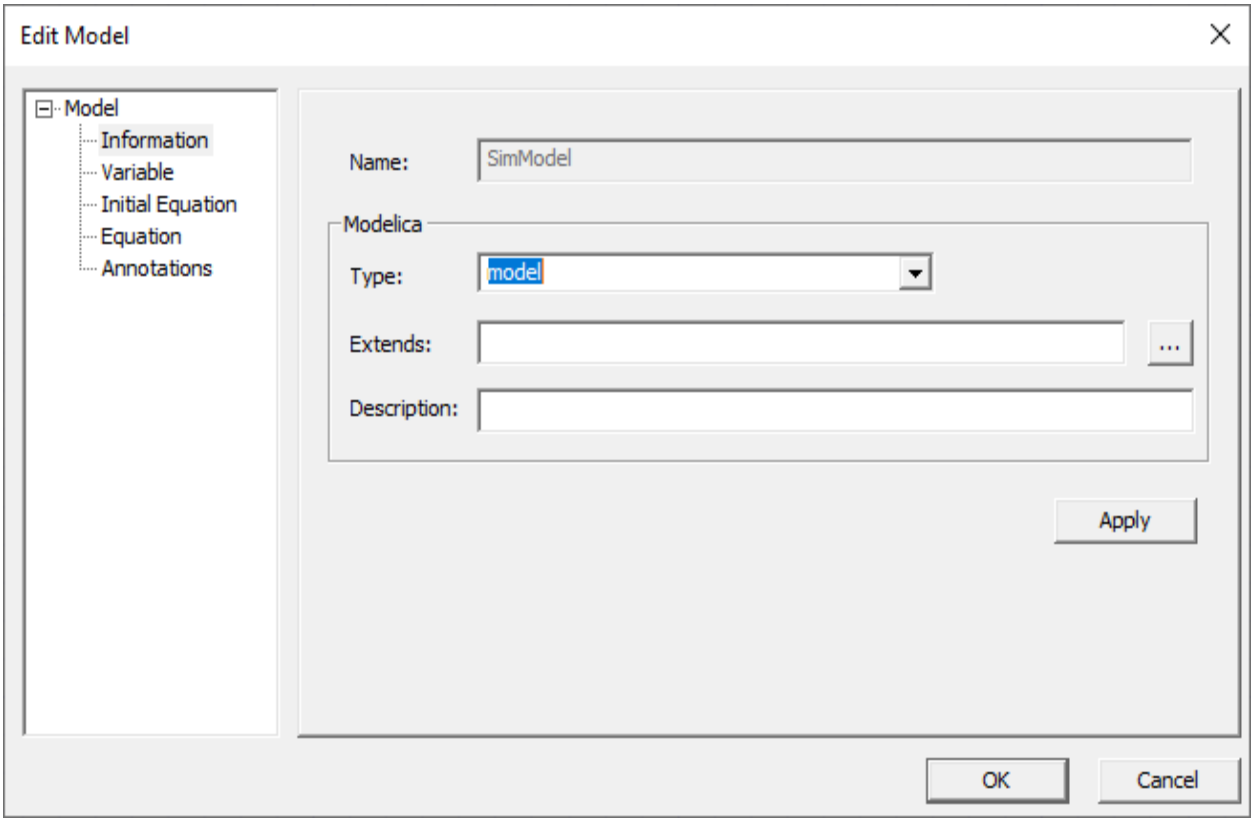
Using the Modelica Model Editor

Edit Model Information

Follow this procedure to edit a model in the Diagram Editor.

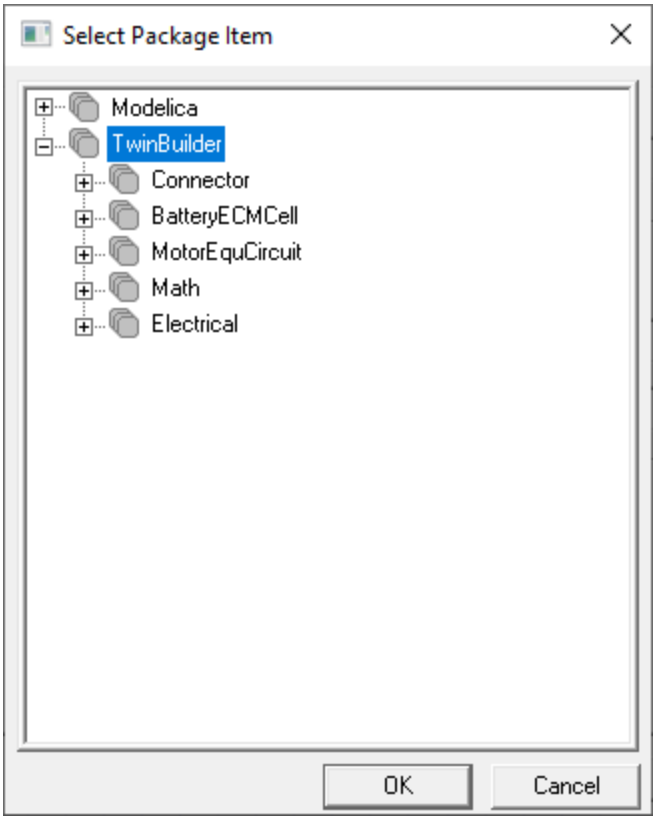
1. Start the Diagram Editor.
2. Select **Modelica Model Editor** > **Model Information**. The **Edit Model** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



3. Select **Information** from the left pane to edit these elements:
   - **Type** – Select a model type from the drop-down list.
   - **Extends** – Click ... to open the **Select Package Item** dialog box and choose the extend classes to include in the model.

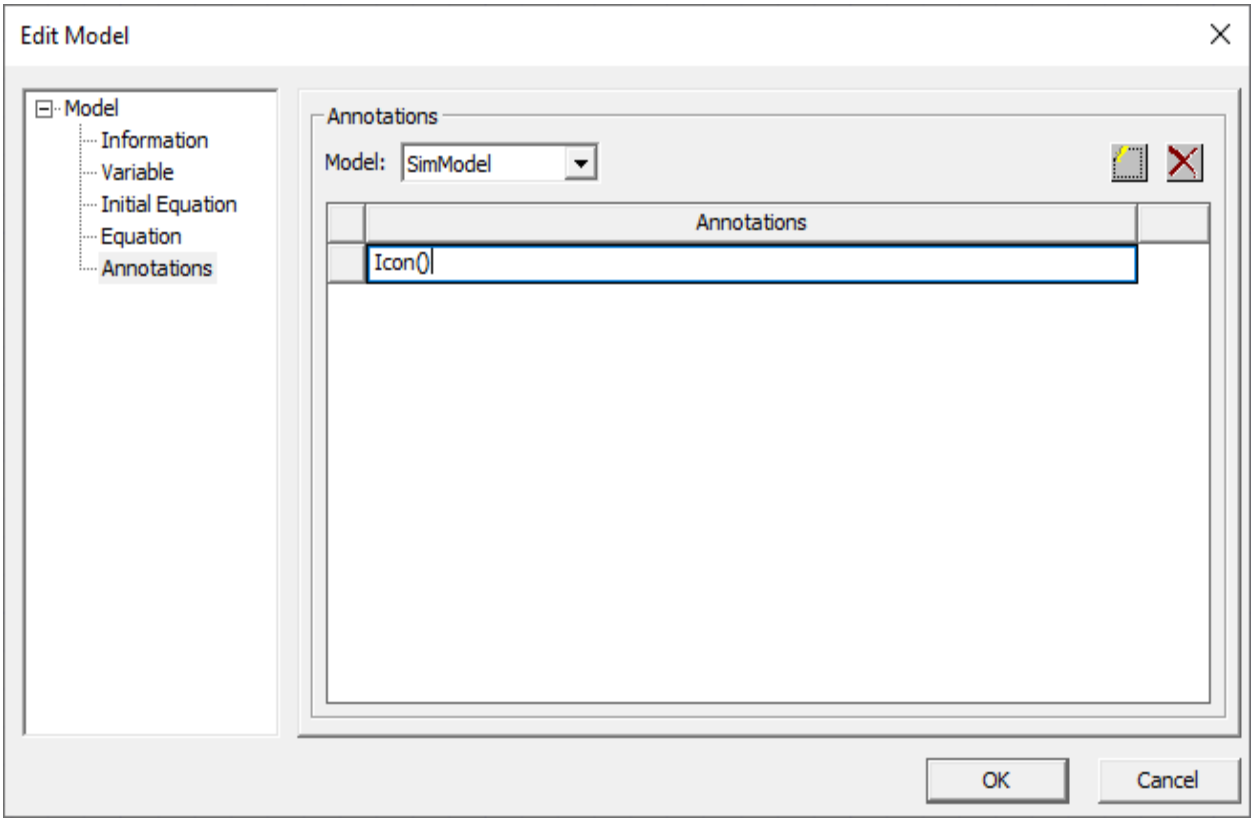
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- **Description** – Edit the model description in the text field.

4. Select **Annotations** from the left pane to add model annotation text such as icon, documentation, protection, licensing, and so on.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



a. Select a model from the **Model** drop-down list. A list of model annotations, if any, appear in the grid below.
b. Click an annotation to edit it.
c. Click to add a new annotation to the grid.
d. Select an annotation and click to delete it.

5. Click **Apply** to commit your changes, then click **OK** to close the **Edit Model** dialog box.

Extend a Modelica Model or Diagram

To extend a Modelica model or diagram, first select a base class to extend from when creating definitions. Use the **Add Model** dialog box to browse for and select a base class in the package browser. After you select a base class, the system generates a base class diagram and adds it to the new definition's diagram. Note that:

- You cannot move or delete base class objects, but you can modify parameterization.
- You can add more objects to this diagram and connect them with base class objects.
- You cannot copy or paste base class objects.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- You can find and modify top-level parameters for a base class object in a variables and declarations dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

13 - Product Coupling

Product coupling allows a product to leverage the capabilities of other products in specialized areas. For example, coupling the Twin Builder System Simulator to Maxwell electromagnetic field simulation provides the accuracy benefit of full field simulation within the circuit simulation environment. Direct integration eliminates the need to translate models, saves time, and allows for greater accuracy.

In addition to the components provided with Twin Builder for use in Schematic Editor designs, Twin Builder schematics can also link to models created in external simulation software through special subcircuit coupling components. These coupling components may use data, extracted from full three-dimensional field data, in a Reduced Order Model (ROM) independent of the original model; or co-simulate with the model in its native product through a transient analysis.

For some ROM coupling components, the Twin Builder model obtained from the external product may be either a dynamic link (default) or static link.

- A dynamic link coupling component always checks the external product for updated ROM data before simulating.
- A static link coupling component only checks the external product if parameters on the coupling component changed since the ROM data was acquired.

> **Note:**
>
> During transient parametric co-simulations with Maxwell, the system adds the **TBOptiVariation** design variable to the Maxwell design in order to facilitate the simulation. This variable is read-only; do not change or delete its value.

***Reduced Order Model* subcircuit coupling components that can be dynamic or static:**

- Maxwell Equivalent Circuit
- Maxwell Dynamic Magnetostatic
- Maxwell Dynamic Electrostatic
- Maxwell Dynamic Eddy Current
- RMxprt Dynamic
- Q3D Dynamic Equivalent Circuit
- Q3D Dynamic Statespace

***Reduced Order Model* Subcircuit coupling components that are only static:**

- PExprt Static
- Mechanical
- Icepak

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Statespace (generic)
- SIwave
- HFSS
- FMU

Cosimulation coupling components:

- Simulink
- Maxwell Transient
- RBD
- Fluent

Twin Builder supports the following subcircuit coupling components:

- Simulink Component
- Maxwell Components
- Rmxprt Dynamic Component
- Q3D Dynamic Components
- PExprt Static Component
- Ansys Mechanical Component
- Icepak Component
- State Space Component
- SIwave Component
- HFSS Component
- RBD Component
- Fluent Component
- Fluent LTI Component
- SCADE Component
- FMU Component
- Data Connector Component

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

You must recreate certain dynamic coupling models created before 2022 R1 in order to utilize the new functionality of using actual pin names in partner designs.

The model types are:

- Q3D state space and Equivalent circuit links.
- Maxwell Dynamic Magnetostatic and Electrostatic links.
- Maxwell Eddy current links.
- HFSS dynamic link.

Simulating designs with these components unchanged results in errors and unpredictable results. You can delete and recreate the component to fix this issue.

Simulink® Subcircuits

MATLAB®/Simulink® is a software application developed by *The MathWorks™ Inc.* to solve numerical problems and visualize data. It is used for a range of applications including manipulations of vectors and matrix arrays, as well as problems in control engineering, system engineering, statistics, and signal processing.

The Simulink subcircuit allows co-simulation between a Twin Builder schematic design and a Simulink model.

To use a Simulink subcircuit with Twin Builder, you need installations of both Twin Builder and MATLAB®/Simulink®. The Simulink subcircuit is available only if the Simulink coupling is licensed in the **license.lic** file in the Twin Builder version you use. Make certain you use the correct versions, **licence.lic** file, and directory settings.

Program Requirements for MATLAB®/Simulink® Subcircuits

Twin Builder and MATLAB

- Twin Builder current version.
- MATLAB R2019b, R2020a, R2020B, R2021a, R2021b, R2022a, R2022b, or R2023a.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

MATLAB must be able to locate the Twin Builder S-Function block (**AnsoftSFunction**) path in order to set up the co-simulation with Simulink. Twin Builder creates the **startup.m** script that contains the necessary paths in <*matlabroot*>**\toolbox\local** during installation; however, MATLAB might not recognize the script if **Toolbox Path Caching** is enabled in MATLAB. If this is the case, follow these steps to update the cache:

1. On MATLAB's **Home** tab in the **Environment** section, click **Preferences**. Select **MATLAB** > **General**.
2. Select **Update Toolbox Path Cache**, then click **OK**.
3. Restart MATLAB.

Note:

The Twin Builder S-Function block path is <*AnsysEMInstallation*>**\Win64\cpl\matlab**<*matlabversion*>. See the MATLAB documentation for instructions on manually adding the path if Twin Builder did not add it during installation. This can happen because of permission issues, if you install MATLAB after Twin Builder, if the **startup.m** script previously existed, and so on.

Note:

Twin Builder checks the latest installed version of MATLAB to establish a link. If that version is not supported, Twin Builder checks for other installed versions. If a supported version is located, Twin Builder registers that version to establish a link.

To manually register MATLAB—which is required to return to the previously registered version—do the following:

1. Open a Windows **cmd** window.
2. Change the current directory to <*matlabroot*>**/bin/win64**.
3. Type **Matlab.exe /regserver** and press Enter.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

- To use an active Simulink window when starting simulation from Twin Builder, and to view simulation results in Simulink, start **Matlab.exe** using **/automation** and launch Simulink *prior* to starting simulation from Twin Builder.
- Select your compiler using **mex -setup** (required to load library) prior to launching or setting up a co-simulation.

Add Simulink Component

Follow this procedure to add a Simulink model to a Twin Builder design.

Warning:

- You can only use one MATLAB/Simulink model subcircuit in a Twin Builder design. An error message appears in the **Message Manager** pane if you attempt to **Browse Netlist** or **Analyze** a design containing more than one Simulink component instance.
- Place the Simulink component on the top level of the design. Co-simulation is not supported when the component is on a subcircuit level.

1. Select **Add Subcircuit** > **Add Simulink Component** from the menu to display the **Simulink Interface** dialog box.
2. Specify a **Name** for the component. See ***Names of Components and Variables*** for naming conventions.
3. Specify the **Matlab File Name**. Click to open a file browser and navigate to your file directories.
4. Select **Read link information from the Simulink file** to enable Twin Builder to launch the Simulink application and load the Simulink **.mdl** file specified in the file name box. If no file is specified, or if the path or file name is invalid, an error appears.
5. If you want to add a copy of the Simulink file to the Twin Builder project, select **Insert Simulink file in the project**.
6. The **Link Assignment** panel contains buttons to add or delete link assignments.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

a. Click to display a row of fields for the link assignment.

Field	Description
Simulink Variable	Enter the identifier for the variable.
Default Value	Specify an initial value for the variable.
Direction	Specify whether the variable supplies data values **To Simulink** or receives data values **From Simulink**.
Add Pin	Select the check box to add a pin for the variable to the component symbol.
Description	Enter a description of the link, if desired.

b. Select existing link assignments from the list and click to delete them..

7. Click **Run Simulink** to launch Simulink.
8. Click **OK** to close the dialog box and apply the changes or **Cancel** to close without changing.
9. Place the component on the schematic sheet.
10. After you place the component, its **Properties** dialog box may open for configuration of the component.

> **Note:**
>
> The Simulink model configuration is saved with the schematic sheet. Twin Builder must establish a link to the MATLAB/Simulink application each time a simulation runs.

Related Topics

Creating and Linking a MATLAB/Simulink Model

Twin Builder/Simulink Co-Simulation

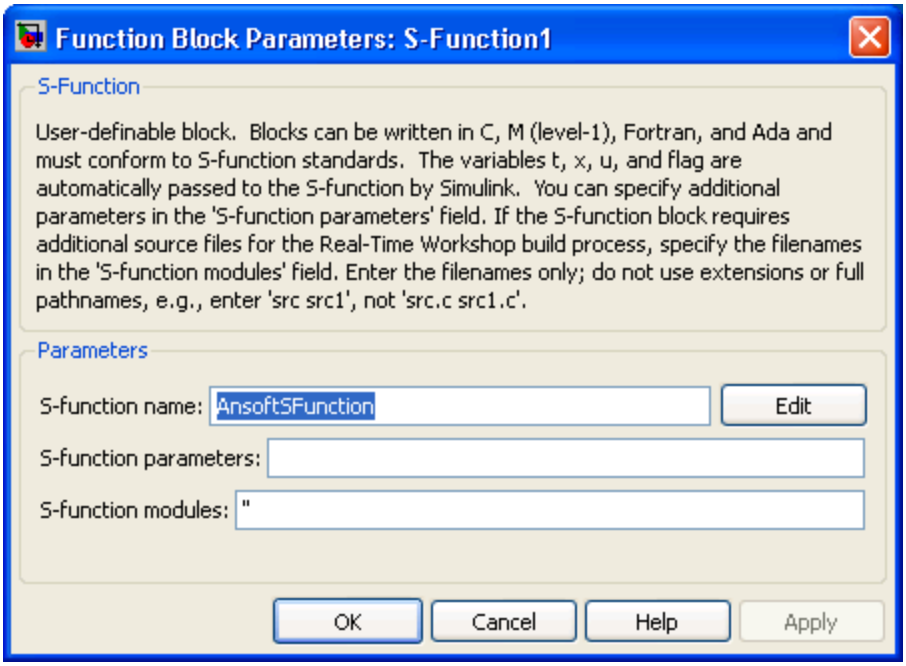
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Creating and Linking a MATLAB/Simulink Model

1. In MATLAB, place an **S-Function** block from the Simulink library on the sheet.

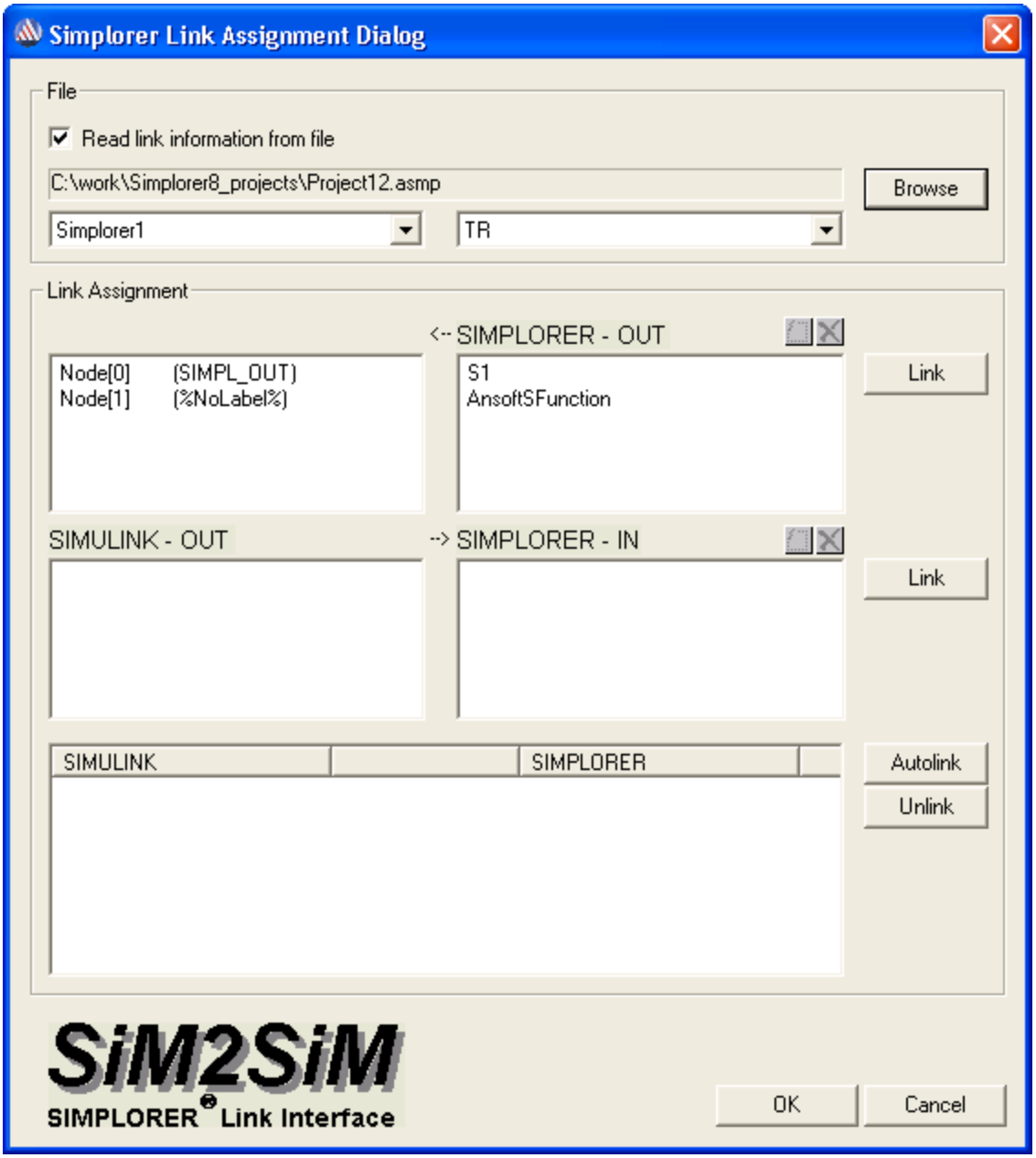
   > **Note:**
   >
   > Only one **S-Function** block (*AnsoftSFunction*) can be used in a model design. An error message appears if you attempt to add a second **S-Function**.

2. Double-click the **S-Function** block to open the **Function Block Parameters** dialog box.



3. In the **S-function name** field, enter **AnsoftSFunction** (case-sensitive) and click **OK**. The **Simplorer Link Assignment Dialog** dialog box opens

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. Select the **Read link information from file** check box and browse to the Twin Builder project **.aedt** file containing the Simulink component.
5. Use the **Link** and **Autolink** buttons to establish the desired signal links between the MATLAB/Simulink model and the Simulink component. **Autolink** links all signals with the same name.
6. Click **OK** when done to close the dialog box.
7. Save the MATLAB/Simulink model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Add Simulink Component

Twin Builder/Simulink Co-Simulation

Twin Builder/Simulink Co-Simulation

1. After you have added a Simulink component in Twin Builder and created a MATLAB/Simulink model, click **Twin Builder** > **Analyze** to start the co-simulation.

   > **Warning:**
   >
   > Only one Simulink subcircuit can be used in a Twin Builder design. An error message appears in the **Message Manager** pane if you attempt to **Browse Netlist** or **Analyze** a design containing more than one Simulink component instance.

2. After initialization, the Twin Builder compiler and simulator start.

Messages generated during the simulation run appear in the **Message Manager** pane. If the Twin Builder subcircuit has specified graphical outputs, they are updated. The MATLAB/Simulink scopes show the outputs defined in MATLAB/Simulink.

> **Note:**
>
> - To achieve a successful co-simulation, at least one output must be defined between the Twin Builder and MATLAB/Simulink models, and the variable step width must be chosen as a simulation parameter in MATLAB/Simulink.
> - **Stop time** in Simulink must be equal to the **End Time (Tend)** in Twin Builder. Extending the simulation end time from Twin Builder (by selecting **Enable continue to solve**, or pausing and extending the end time using the progress bar) will not have any effect on the Simulink side.

Using MathWorks™ Simulink Coder® to Export a Simulink Model to a Twin Builder C-Model DLL

This section describes how to export a Simulink model to a Twin Builder C-Model using MathWorks Simulink Coder. Simulink Coder generates the model based C-code, the Twin Builder C-interface functions, and the necessary makefile for Microsoft Visual C++. After the C-

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

code generation, Simulink Coder calls the compiler and generates a ready-to-use Twin Builder C-interface DLL for a Twin Builder model library.

Twin Builder uses Simulink models directly, without co-simulation. The exported models run standalone directly in a Twin Builder environment. Communication with Simulink is not necessary.

Features and Limitations

There are target entries in the Simulink Coder system target list for exporting to a Twin Builder target.

- You can export the Simulink model to a ***Twin Builder C-Model Target***. That target will work with fixed step size. The maximum integration step size in Twin Builder is fixed by the sample time. You can use more than one instance of the model, exported to that target in one Twin Builder sheet. For the Twin Builder C-Model target, no Simulink license is necessary. The model runs without Simulink because the Simulink integration solver is also exported, and built into the model itself.
- Simulations run completely in the Twin Builder environment. The models have no step delay, and performance is much better than with the Simulink co-simulation interface.
- Because Simulink models don't have separate AC or DC behavior; the exported model works only for transient simulation in Twin Builder.
- All ports must be scalar of `type=double`. Vector or complex ports are not supported.
- Data transfer to the DLL can only be performed through input and output interface ports. Variables defined in the original Simulink model will not appear as quantities on the generated model, and cannot support S-Functions that call MATLAB such as Twin Builder's co-simulation S-Function block (**AnsoftSFunction**).

Requirements for Model Creation

- Microsoft Visual Studio Professional. For supported Visual Studio versions, see *Supported and Compatible Compilers* by Mathworks.
- MATLAB, Simulink, and Simulink Coder.
- Simulink Coder target template files in the **AnsysEM***version***Win64\cpl\matlab\coder** directory.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

MATLAB must be able to locate the Twin Builder target path in order to build a Twin Builder C-Model DLL. Twin Builder creates the **startup.m** script that contains the necessary paths in <*matlabroot*>**\toolbox\local** during installation; however, MATLAB might not recognize the script if **Toolbox Path Caching** is enabled in MATLAB. If this is the case, follow these steps to update the cache:

1. On MATLAB's **Home** tab in the **Environment** section, click **Preferences**. Select **MATLAB** > **General**.
2. Select **Update Toolbox Path Cache**, then click **OK**.
3. Restart MATLAB.

Note:

The Twin Builder target path is <*AnsysEMInstallation*>**\Win64\cpl\matlab\coder**.See the MATLAB documentation for instructions on manually adding the path if Twin Builder did not add it during installation. This can happen because of permission issues, if you install MATLAB after Twin Builder, if the **startup.m** script previously existed, and so on.

Template Files

Template Files	Description
twinbuilder_rt.tlc	This file is the entry point for the C/C++-code generation with Simulink Coder. It controls the target language compiler and adds a new property page with Twin Builder build options to the **Simulink Coder** dialog box.
twinbuilder_vc_rt.tmf	Template for the target makefile. This template collects the created model C/C++ source files, the integration solver files, and the created Twin Builder interface C++ source file to generate a makefile. They set the compiler flags, include and lib directories for the compiler.
twinbuilder_rt_ wrapper.tlc	Template to create the C++ source code for the Twin Builder interface. Depending on the options set in the **Simulink Coder** dialog box and the Simulink model interface, adds inputs

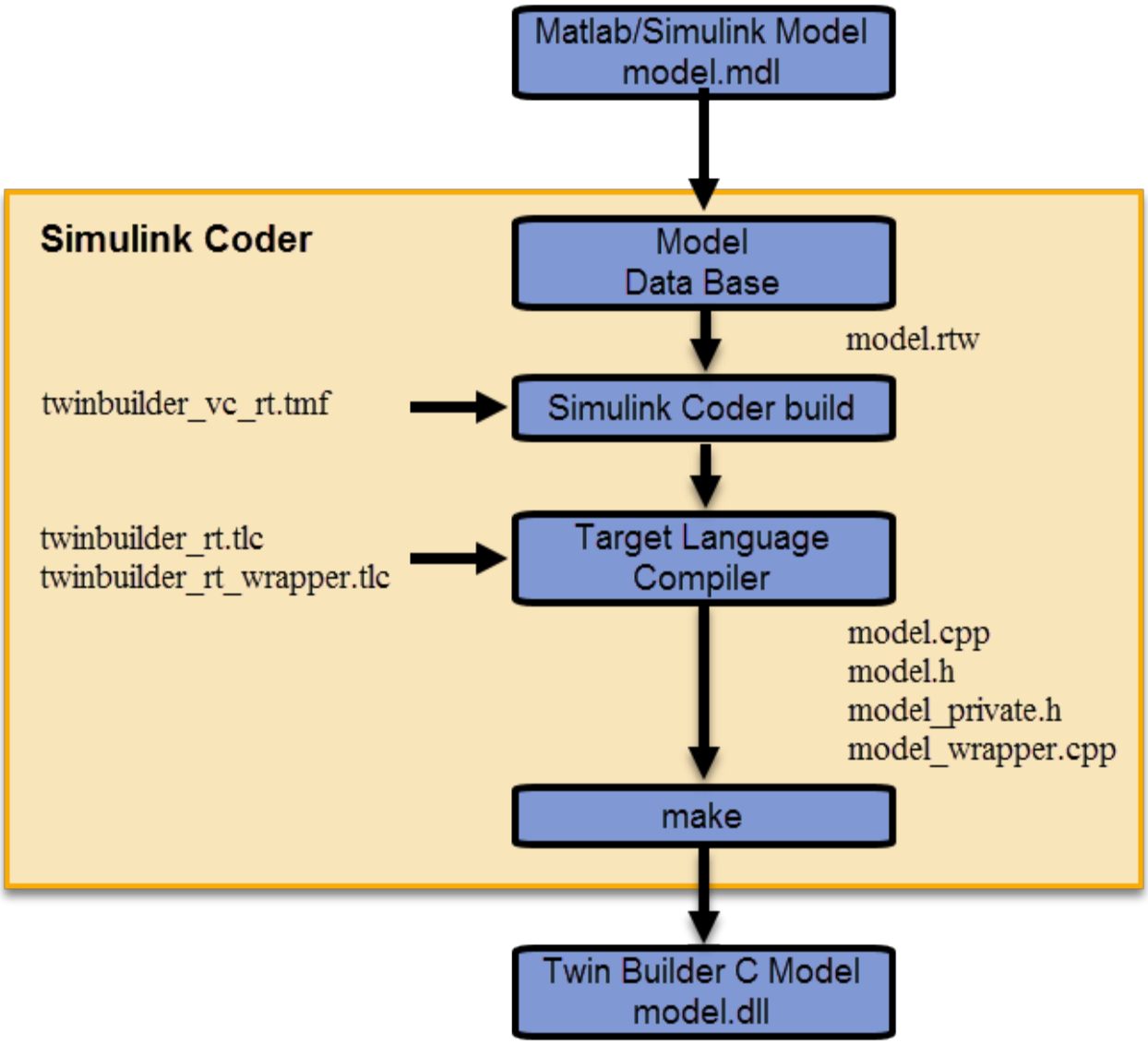
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Template Files	Description
	and outputs, and wraps the Simulink model into a Twin Builder C-Model. Simulink input or outputs labels are base for the Twin Builder C interface names.

Workflow

All models supported by Simulink Coder can be exported to a Twin Builder C-Model DLL file.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

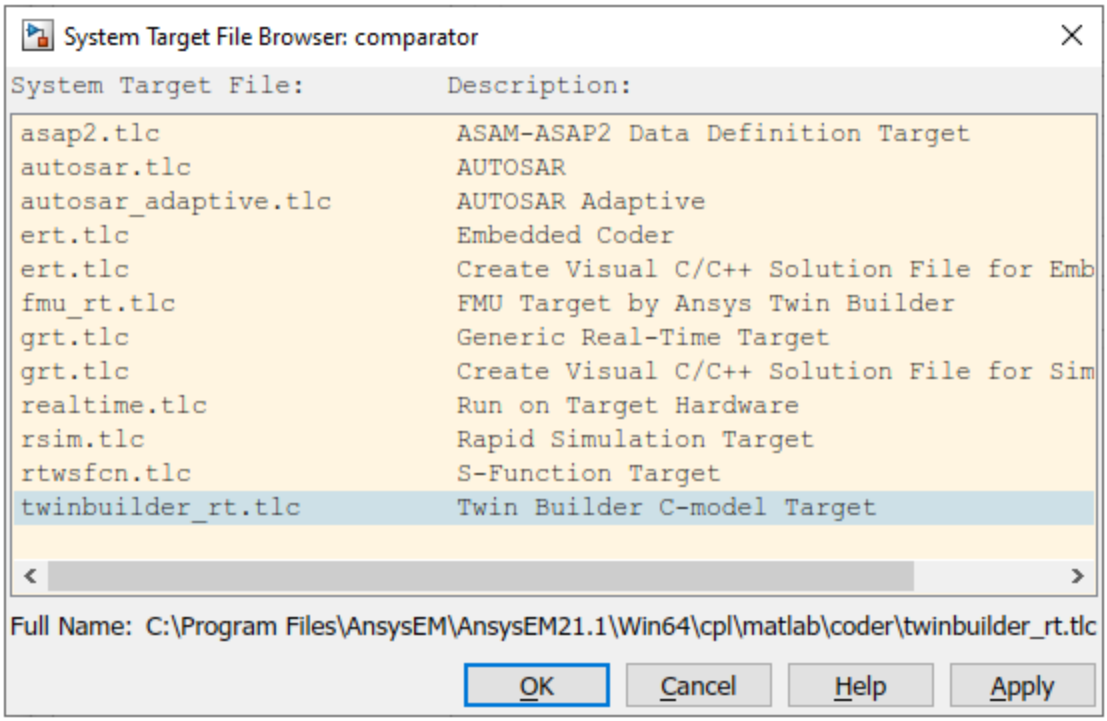


Target

Follow this procedure to build a Twin Builder model.

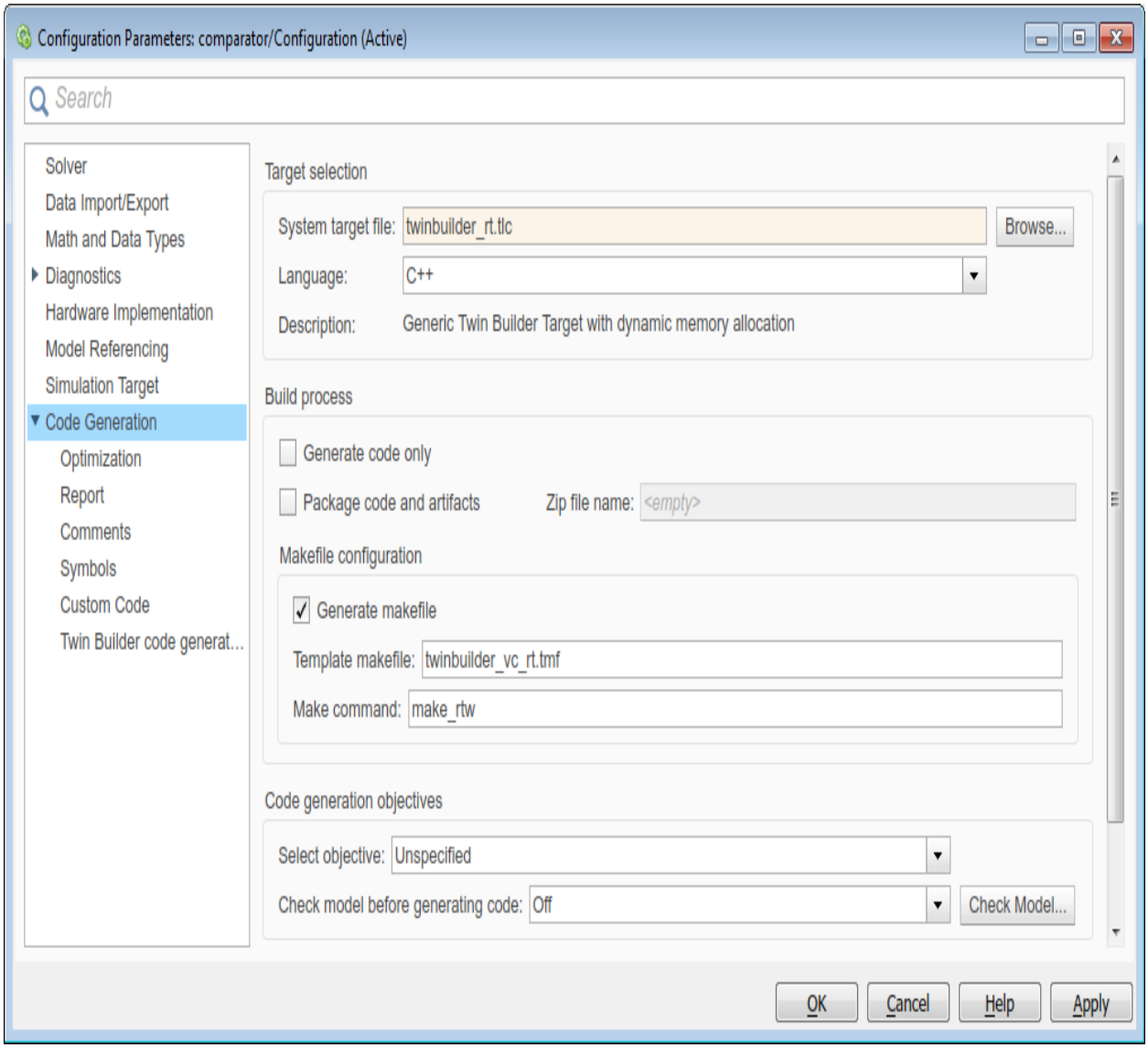
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Open the **System Target File Browser** and select **twinbuilder_rt.tlc** as shown below:



2. The **Configuration Parameters** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

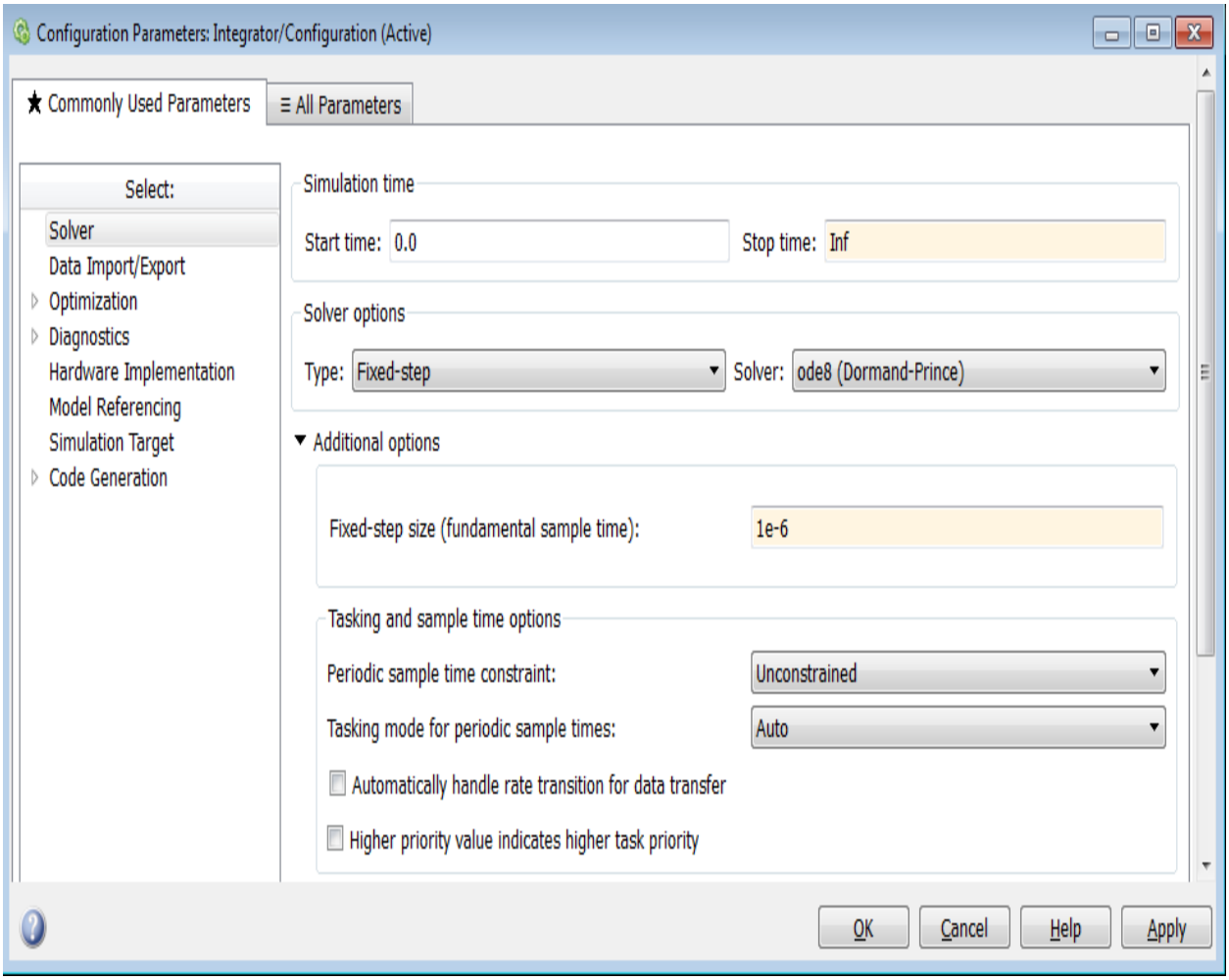


3. After selecting a Twin Builder Target, Simulink Coder updates the parameter dialog box. An additional Twin Builder page called **Twin Builder code generation** appears in the **Code Generation** section in theleft pane. The Twin Builder C-Model DLL is created in the current Simulink working directory.

Twin Builder C-Model Target with Dynamic Memory Allocation

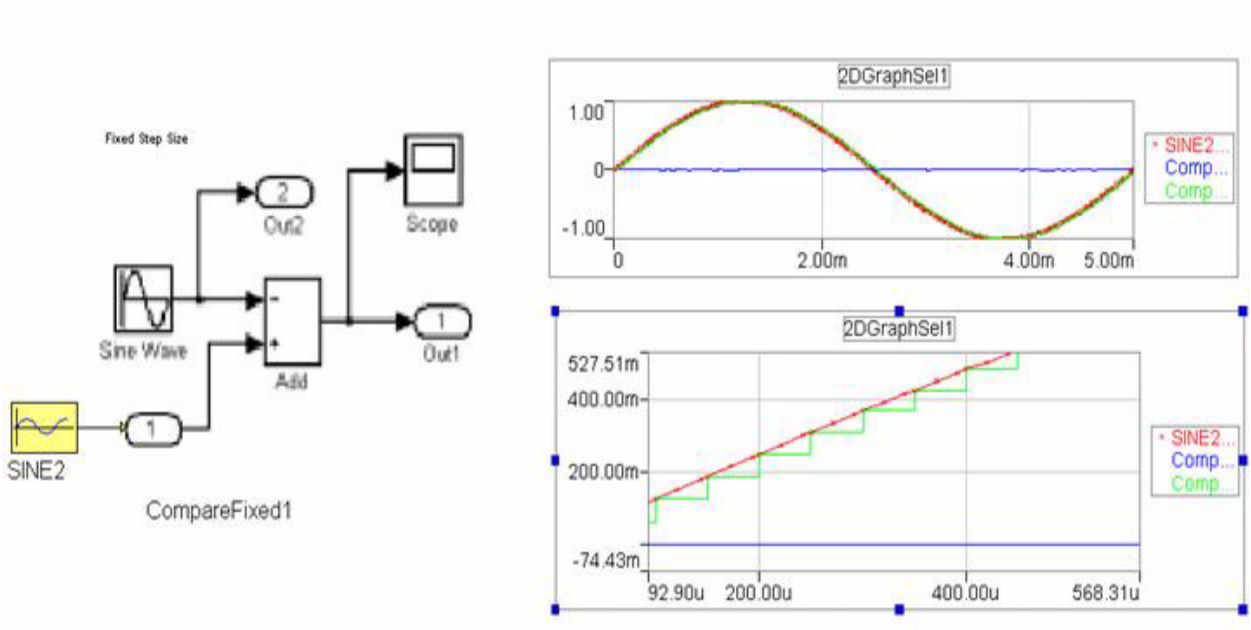
The Twin Builder C-Model target uses dynamic memory allocation. With that feature multiple instances of the Simulink model in one Twin Builder sheet are possible. The exported model includes the Simulink fixed step solver, so you don't need a Simulink license. The model C-code and all temporary files are generated in the *<model>***_grt_twinbuilder** folder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



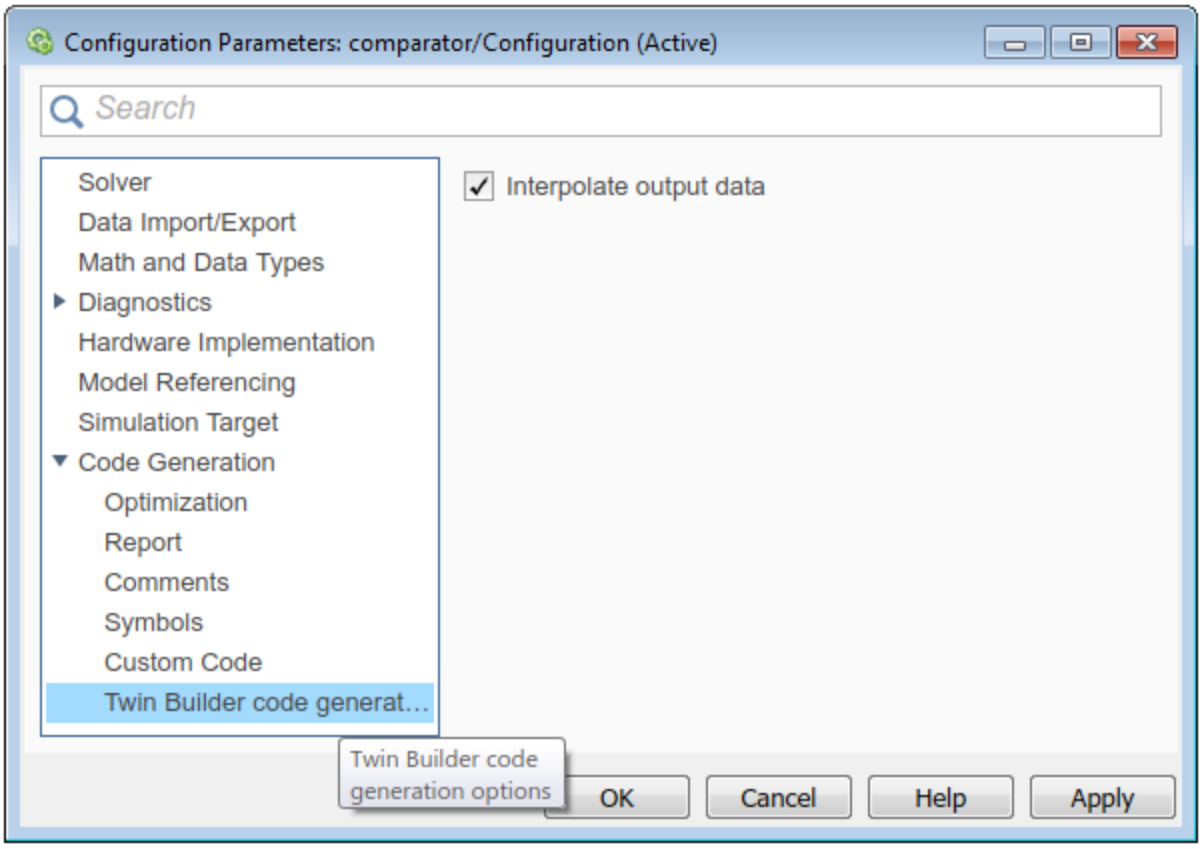
The solver must be set to a fixed type. If the stop (end) time is set to **Inf**, the model uses the Twin Builder end time. This option guarantees the most flexibility and should be used instead of a fixed end time. If you do set a fixed end time, the model output will be constant after that time is passed. The fixed step influences the maximum integration time in Twin Builder. Twin Builder synchronizes to these time points. A small sample time here could slow down the simulation dramatically. This time should not be too large because between two samples, the output of the Simulink model is constant. A value change on an input port is not recognized by the Simulink model between two samples. The step size should be a compromise between accuracy and speed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



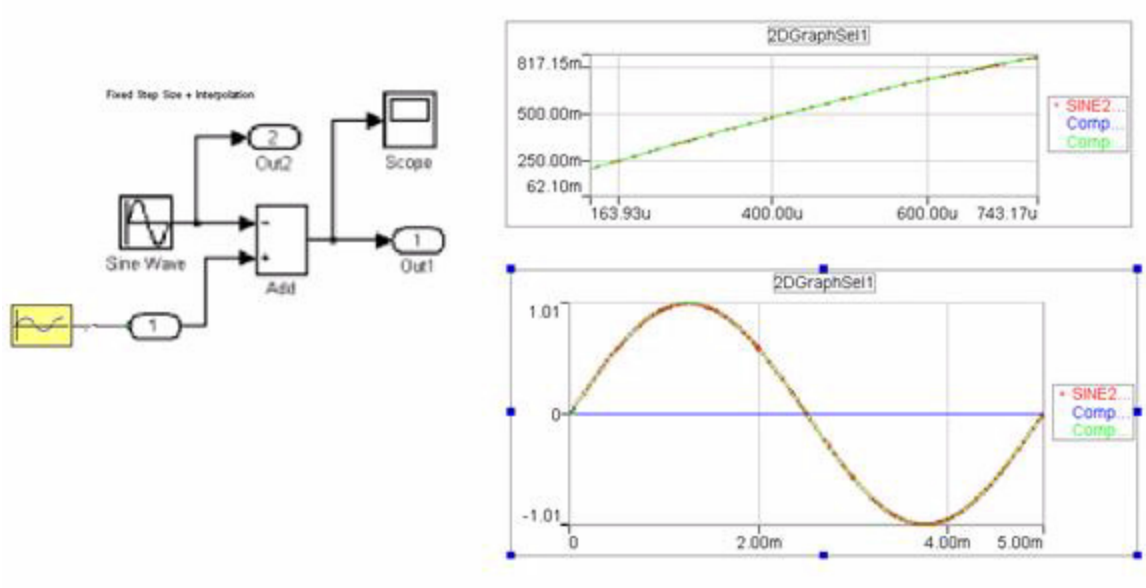
It is possible to interpolate the outputs of the Simulink model. Instead of constant values between two samples, the model uses interpolated values.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



For another output, this option could be used. In some cases, the interpolation increases the number of iterations and reduces the accuracy. This could be the case, for example, if the model switches the output value or if the output is discrete.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Using MathWorks™ Simulink Coder® to Export a Simulink Model as an FMU

This section describes how to export a Simulink model as an FMU using MathWorks Simulink Coder. Simulink Coder generates the model-based C-code, and the necessary Makefile for Microsoft Visual C++. The generated code is compiled into a DLL supporting FMI 2.0 co-simulation interface (see the FMI Web site https://fmi-standard.org/ for detailed information). Along with the DLL and the model description provided as an XML file, an FMU file is created in the current Simulink working directory.

The generated FMU only supports the FMI 2.0 co-simulation interface. Even though the interface is type co-simulation, the model is self-consistent and does not have any interaction with MATLAB or Simulink during the simulation – it is very similar to Simulink Coder-generated Twin Builder C-Model, except for the interface.

The FMU export is provided through ***FMU Target by Ansys Twin Builder***.

Features and Limitations

- Simulink solver type must be Fixed-step for the code generation.
- Target language must be C.
- All ports must be scalar of real type. Vector or complex ports are not supported.
- Data transfer to the FMU can only be performed through input and output interface ports. Variables defined in the original Simulink model will not appear as quantities on the generated model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Cannot support S-Function that call MATLAB such as Twin Builder's co-simulation S-Function block (AnsoftSFunction).
- The exported FMU does not need a Simulink license to run.
- Multiple instances of the FMU can be simulated in a single design.

Requirements for Model Creation

- Microsoft Visual Studio Professional. For supported Visual Studio versions, see *Supported and Compatible Compilers* by Mathworks.
- MATLAB, Simulink, and Simulink Coder.
- Simulink Coder FMU target (**fmu_rt.tlc**) and related files in the <*AnsysEM Installation*>**\Win64\cpl\matlab\coder** directory.

> **Note:**
>
> MATLAB must be able to locate the Twin Builder target path in MATLAB in order to build an FMU. Twin Builder creates the **startup.m** script that contains the necessary paths in <*matlabroot*>**\toolbox\local** during installation; however, MATLAB might not recognize the script **Toolbox Path Caching** is enabled in MATLAB. If this is the case, follow these steps to update the cache:
>
> 1. On MATLAB's **Home** tab in the **Environment** section, click **Preferences**. Select **MATLAB** > **General**.
> 2. Select **Update Toolbox Path Cache**, then click **OK**.
> 3. Restart MATLAB.

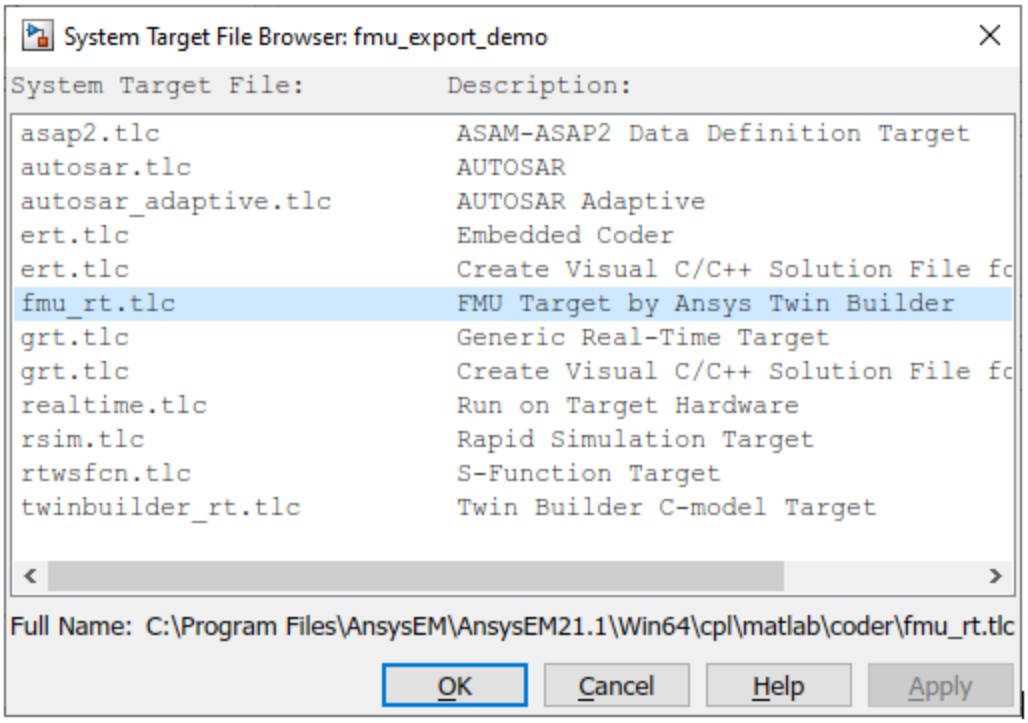
> **Note:**
>
> The Twin Builder target path is <*AnsysEMInstallation*>**\Win64\cpl\matlab\coder**.See the MATLAB documentation for instructions on manually adding the path if Twin Builder did not add it during installation. This can happen because of permission issues, if you install MATLAB after Twin Builder, if the **startup.m** script previously existed, and so on.

FMU Target by Ansys Twin Builder

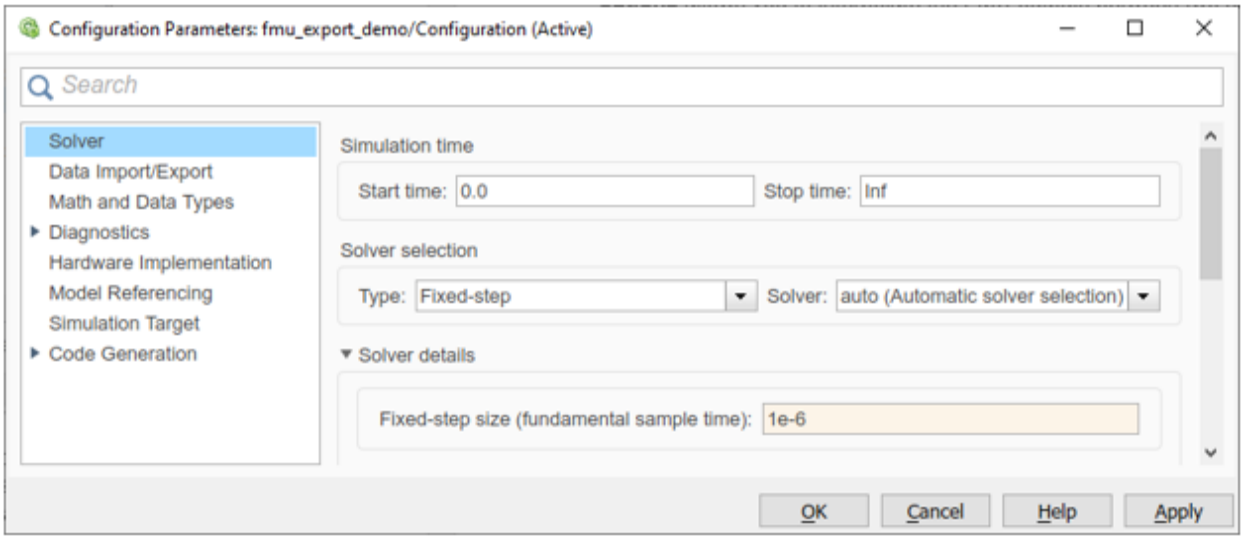
Follow this procedure to build a Twin Builder model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Open the **System Target File Browser** and select **fmu_rt.tlc** as shown below:



2. The **Configuration Parameters** dialog box appears. Select **Solver** from the left pane.

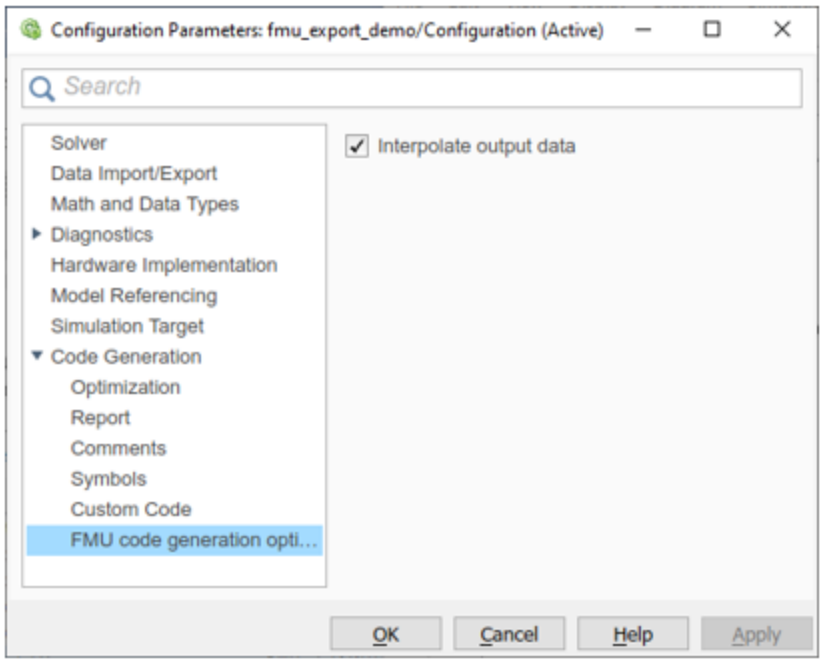


3. Set the **Simulation time** > **Stop time** to **Inf**. This ensures the model uses the end time of the simulation environment, as well as guarantees the most flexibility as opposed to a

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

fixed end time. If you do set a fixed end time, the model output is constant after that time has passed.

4. Set the **Solver selection** > **Type** to **Fixed-step**. The fixed step size influences the simulation's accuracy and speed. A small sample time here could slow down the simulation dramatically, while a large sample time may lead to inaccurate results. A value change on an input port is not recognized by the Simulink model between two samples.
5. Select **Code Generation** > **FMU code generation** from the left pane. Select the **Interpolate output data** check box to interpolate the FMU outputs between two samples. If you clear it, then output between any two samples will be constant.



Adding a Dynamic Coupling Component Subcircuit

The following general procedure provides an overview for adding a dynamic coupling component subcircuit to a Twin Builder design. Detailed information for each dynamic coupling component type is also available following this general procedure.

Related Topics

- Maxwell Coupling Component Subcircuits
- Rmxprt Dynamic Component Subcircuits
- Q3D Dynamic Component Subcircuits

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

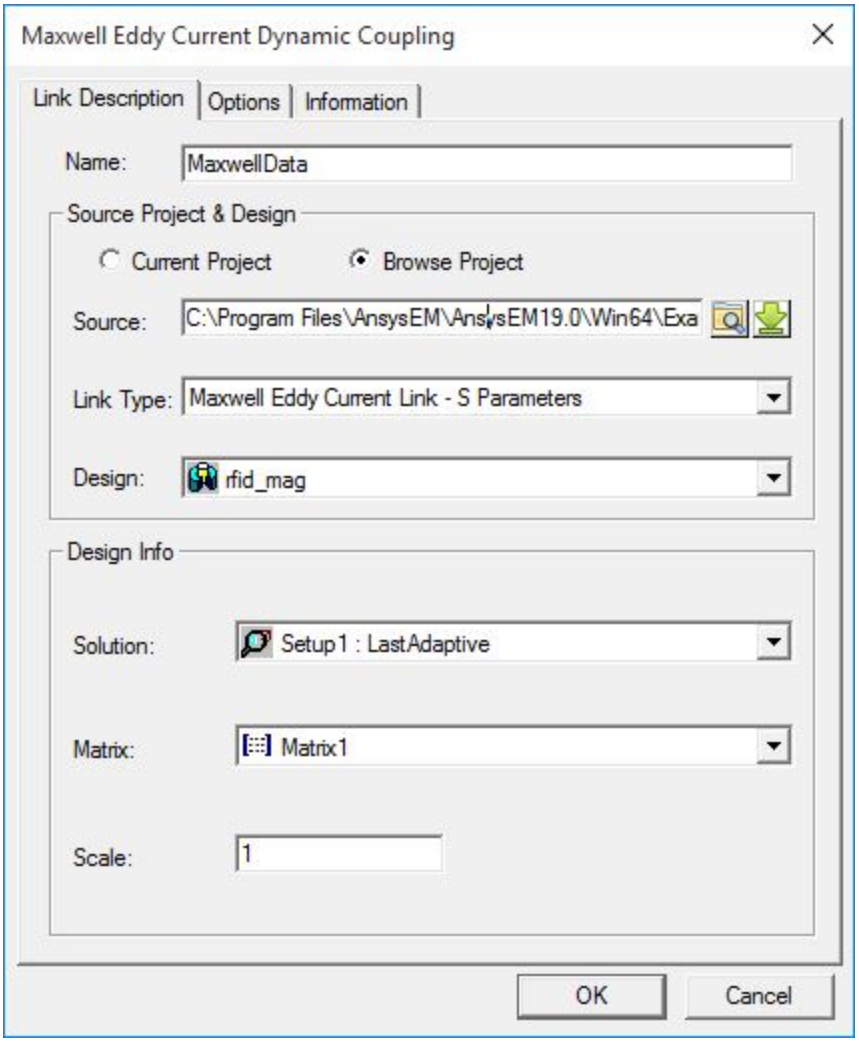
1. Select the appropriate command from the following list to add the corresponding model type:
   - **Subcircuit** > **Add RMxprt Dynamic Component**
   - **Subcircuit** > **Maxwell Component** > **Add Transient Cosimulation**
   - **Subcircuit** > **Maxwell Component** > **Add Dynamic Magnetostatic**
   - **Subcircuit** > **Maxwell Component** > **Add Dynamic Electrostatic**
   - **Subcircuit** > **Maxwell Component** > **Add Equivalent Circuit**
   - **Subcircuit** > **Maxwell Component** > **Add Dynamic Eddy Current**
   - **Subcircuit** > **Q3D Dynamic Component** > **Add Equivalent Circuit**
   - **Subcircuit** > **Q3D Dynamic Component** > **Add State Space**

   The <*Model_Type*> **Dynamic Coupling** dialog box appears, and Twin Builder launches the application in which the model type was created.

Note:

Resize the coupling dialog box to view long field entries in the **Name** and **File** fields, and in all drop-down lists as shown in the example below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. On the **Link Description** tab:
   a. Specify a **Name** for the component. See ***Names of Components and Variables*** for naming conventions.

      > **Note:**
      >
      > The name specified cannot be the same as any existing coupling component and/or model.

   b. In the **Source Project & Design** area, enter the project **Source** file. Select the **Source** project from a current project (if it exists in the current project) or browse to select or enter the path of the source project. The selected project loads into its application.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

c. Select the **Link Type** (S-parameter or RLGC Parameter) for **Maxwell Dynamic Eddy Current** or **Q3D State Space**.

   Twin Builder communicates with the model's source application, retrieves the applicable **Design**, **Solution**, **Matrix** (Maxwell only) or **Reduced Matrix** (Q3D only) information, and **Scale** (Maxwell only) and populates the corresponding fields with the information. The **Information** tab updates to show the **Number of Conservative Pins** and various other **Quantities** present in the selected model.

d. After you select the **Source** file and **Link Type**, select the design you want to use in the coupled model from the list in the **Design** drop-down list.

e. Select the desired solution from the **Solution** drop-down list.

f. For Maxwell **Dynamic Magnetostatic**, **Dynamic Electrostatic**, and **Dynamic Eddy Current** coupling components, select the **Matrix** to be used with the component during simulation.

g. Optionally, for Maxwell **Dynamic Magnetostatic** and **Dynamic Electrostatic** coupling components, set the **Scale** to be passed on to Maxwell during simulation, where it performs the same function as the Maxwell Symmetry Multiplier setting. Maxwell then passes the results back to Twin Builder. (See ***Setting a Symmetry Multiplier*** in the Maxwell help for additional information.)

> **Note:**
>
> Coupling components created in Twin Builder versions prior to version 10 may need to be recreated to enable use of the **Scale** setting.

h. For Q3D **Equivalent Circuit** and **State Space** dynamic coupling components, select the **Reduced Matrix** to be used with the component during simulation.

3. On the **Options** tab:

   a. Select **Save project after use** to have the linked project file saved by its application after co-simulation is completed.

   b. Select **Unload project after use** to have the linked project file closed by its application after co-simulation is completed.

   c. If appropriate for your coupling component type, select **Create static link** to create a static link and enable the **Clear Static Cache** button.

      - If **Create static link** is not selected, Twin Builder obtains sml/netlist information from the coupled product application *every* time a simulation analysis is run – even if no parameter has changed. This is the default behavior.
      - If **Create static link** is selected, Twin Builder obtains sml/netlist information from the coupled product application only if a parameter value has changed since the previous simulation run. Click **Clear Static Cache** to delete the sml/netlist information currently stored in the Twin Builder model. The next simulation run

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

will then obtain fresh sml/netlist information from the other product application and store it in the Twin Builder model.

d. Choose to show either the **Pin Name** or **Pin Description** on the new component. By default, pin names are shown.

> **Note:**
>
> Pin description properties are always added to the component regardless of whether they are selected to be shown with pins.

- If you chose **Pin Description** when adding the component, you can subsequently enable individual pin names. Open the component symbol in the symbol editor, delete a pin's description, double-click the pin and enable **ShowName** in the pin properties dialog box.
- If you chose **Pin Name** when adding the component, you can subsequently hide individual pin names. Open the component symbol in the symbol editor, double-click a pin and disable **ShowName** in the pin properties dialog box. You can then add descriptive text to the pin using the editor's **Draw** > **Text** tool.

e. For Maxwell and Q3D **State Space** coupling components:

- Choose whether to enforce passivity. Enforcing passivity leads to a model that is better for a stable transient simulation.
- Specify **Error Tolerance, Maximum Order**, and **Z(ref)** values.

4. When finished, click **OK** to accept the values and close the dialog box.

> **Note:**
>
> If a model is generated with this component, it is given the same name as the component. Also, the name specified cannot be the same as any existing coupling component and/or model. An informational dialog box displays if the name chosen in *step 2* is already used by another coupling component or model.

5. Place the component on the schematic sheet.

> **Note:**
>
> The dynamic coupling component configuration is saved with the schematic sheet. Twin Builder must establish a link to the associated application each time a simulation is run.

6. After placing the component on a sheet, its **Properties** dialog box may be opened for configuration of the component.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> On the **Properties** dialog box **Parameter Values** tab, click the component **Data** button to open the <*Component_Type*>**Dynamic Coupling** dialog box in read-only mode so you can review the coupling settings. If the component instance contains SML code, the dialog box contains a **Preview** tab on which you can view the SML code.

Python Component Coupling

See the topics in this section to learn how to incorporate Python models in the simulation process.

System Requirements

To use Python language capabilities in Twin Builder, the following must be installed:

- Twin Builder
- Python (64-bit)
- Microsoft® Visual Studio® C++ Compiler (Community or Professional version)

> **Note:**
>
> Make sure to add the Python installation path to your system or user environment variable (PATH).

> **Note:**
>
> Python is also available from the Twin Builder installation to create and simulate Python components.

Overview

A Python model is special C-Model which loads the Python interpreter and runs your Python code during simulation.

Interfaces

You must provide interfaces to interact with other existing Twin Builder components.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Input interfaces are real(float) or string.
- Output interfaces are real(float) only.

Python code:

You must implement one or more of the following functions:

```
def PreProcess(TBInputs) :
  TBOutputs = {}
  return TBOutputs


def Simulate(TBInputs) :
  TBOutputs = {}
  return TBOutputs


def Step(TBInputs) :
  TBOutputs = {}
  return TBOutputs


def PostProcess(TBInputs) :
  TBOutputs = {}
  return TBOutputs
```

The **PreProcess** function is evaluated during initialization (that is, at the beginning of the simulation). In this section, the model can initialize outputs and perform initialization before the simulation process.

The **Simulate** function is evaluated at each time step of the analysis.

The **Step** function is only evaluated after each successful time step of the analysis. In this section, the model can perform tasks unique to every time step.

The **Post Process** function is evaluated at the end of the simulation. In this section, the model can do cleanup operation, post-processing, result file creation, or documentation creation.

Each function has **TBInputs** as argument. **TBInputs** is a dictionary which contains the user-provided interfaces. The interface name is the dictionary key. During simulation, Twin Builder provides all input. The interface's current value is provided by this dictionary.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The **TBInputs** dictionary also contains these simulation parameters for each function:

- **TBInputs[“$Time”]** – Provides the current simulation time.
- **TBInputs[“HMIN”]** – Provides the simulation's minimum time step.
- **TBInputs[“HMAX”]** – Provides the simulation's maximum time step.
- **TBInputs[“Step”]** – Provides the simulation's current step size.

> **Note:**
>
> The global **TBInputs** dictionary does not contain these simulation parameters.

TBOutputs is a dictionary that you can use to write or return the simulation result to Twin Builder. The interface name is again used as the dictionary key. Use the output interfaces to connect them to other Twin builder components, or to create reports.

> **Note:**
>
> The above only works with TR simulation.

> **Note:**
>
> You must update the component if the Python version changes.

Default global input interfaces dictionary

Each definition and instance of a Python component gets a default global dictionary of all inputs. This dictionary lets you access the input interface’s initial values in a global scope. The dictionary name is **TBInputs**.

The global input dictionary contains some of the useful directories you can use to find and use external files. They can be retrieved as shown here:

- **TBInputs[“ProjectPath”]** – Provides the project directory path. It can be useful to import files from same directory as the project.
- **TBInputs[“ProjectName”]** – Provides the current project name.
- **TBInputs[“InstallDir”]** – Provides the Twin Builder installation directory.
- **TBInputs[“Temp”]** – Provides the Twin Builder temporary files directory.
- **TBInputs[“SysLib”]** – Provides the Twin Builder system library directory.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **TBInputs[“UserLib”]** – Provides the Twin Builder user library directory.
- **TBInputs[“PersonalLib”]** – Provides the Twin Builder personal library directory.

> **Note:**
>
> These above paths are only available through the global dictionary.

Add Custom Python Module from Project directory

You can directly import modules present in the project directory with import statements. For any other directory, follow this example:

It can be useful to add directories to **sys.path** to add custom modules as shown below.

```
import sys

sys.path.append(TBInputs['path'])

...
```

where `path` is the input interface (type: string).

In a Twin Builder schematic, you can specify the directory to add by changing the `path` interface value.

Launch Editor

This command launches an editor that you designated for Python (.**py**) file editing. You can designate an editor using open with settings in file properties.

Import & Export

Use **Import** to import any external .**py** file which might contain the code.

Use **Export** to export the code to a .**py** file for external editing and testing.

Add a New Python Component

Follow this procedure to add a new Python component in Twin Builder.

1. Select **Tools** > **General Options** > **Twin Builder** > **Python Model Options**. Use the drop-down lists to choose a Python interpreter and C++ compiler.
2. Select **Twin Builder** > **SubCircuit** > **Add Python Component**. The **Add Python Component** dialog box appears.
3. Add your input and/or output interfaces in the **Interfaces** section.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Select the **Edit** check box and add your Python code in the **Python Code** section.
5. Click **Import**.

The system compiles a Twin Builder C-Model with your provided interfaces and Python code. The C-Model interacts with the Python interpreter you selected in the **Options** dialog box. Upon successful compilation, you can place this component on your schematic.

As with all C-Models, compilation requires that Microsoft® Visual Studio® C++ is installed.

Edit or Update a Python Component

Follow this procedure to edit or update a Python component in Twin Builder.

1. Select a Python component on the schematic.
2. Right-click the component and select **Edit Model**.
3. Make your changes to interfaces/code and click **Update**. The system will update the definition.

> **Note:** You can also use this procedure to edit and make changes to interfaces or python code, as well as recompile.

> **Note:** If the Python version changes, you must recompile/update the component.

Options

Select **Tools** > **General Options** > **Twin Builder** > **Python Model Options**. There are two drop-down lists:

- **Python Location (x64)** – Browse and select the Python installation folder. The installation must be 64-bit and include the **/libs/** and **/include/** folders. Note that by default, Python (x64 bit) is selected from the Twin Builder installation, but you can select it from the local installation as well.

  > **Note:**
  >
  > Make sure to add the Python installation path to your system or user environment variable (PATH).

- **C++ Compiler** – Choose from a list of Microsoft® Visual Studio® C++ compilers installed on your machine that are supported by Twin Builder. Visual Studio C++ Community or

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Professional version are required. For more details, see the topics in ***C-Models in Twin Builder***.

Maxwell Coupling Components

Coupling between Twin Builder and Maxwell can take several forms, depending on your requirements. The following types of coupling components are available to use detailed electromagnetic field simulation models in the Twin Builder environment:

Maxwell Transient Cosimulation Component

Maxwell is an interactive software package that uses Finite Element Analysis (FEA) to solve 2D and 3D electromagnetic problems. To analyze a problem, the component designer specifies the appropriate geometry, material properties, and excitations for a device or a system of devices. When a **Transient Cosimulation Component** subcircuit is used in a Twin Builder design, Maxwell 2D or 3D transient and Twin Builder transient solvers exchange data during each simulation time step.

> **Note:**
>
> During transient parametric co-simulations with Maxwell, the system adds the **TBOptiVariation** design variable to the Maxwell design in order to facilitate the simulation. This variable is read-only; do not change or delete its value.

This section contains information on:

- Program Requirements
- Transient Cosimulation Interface concept
- Maxwell 2D Finite Element Model Prerequisites
- Adding a Maxwell Transient Cosimulation Component
- Preparing a Maxwell 2D Transient Model for a Link to Twin Builder
- How to Use a Maxwell 2D Finite Element Model in Twin Builder
- Continuing a Cosimulation with Maxwell

Transient Cosimulation Component Program Requirements

- Twin Builder current version.
- Maxwell 2D or 3D current version, on a Windows platform only.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

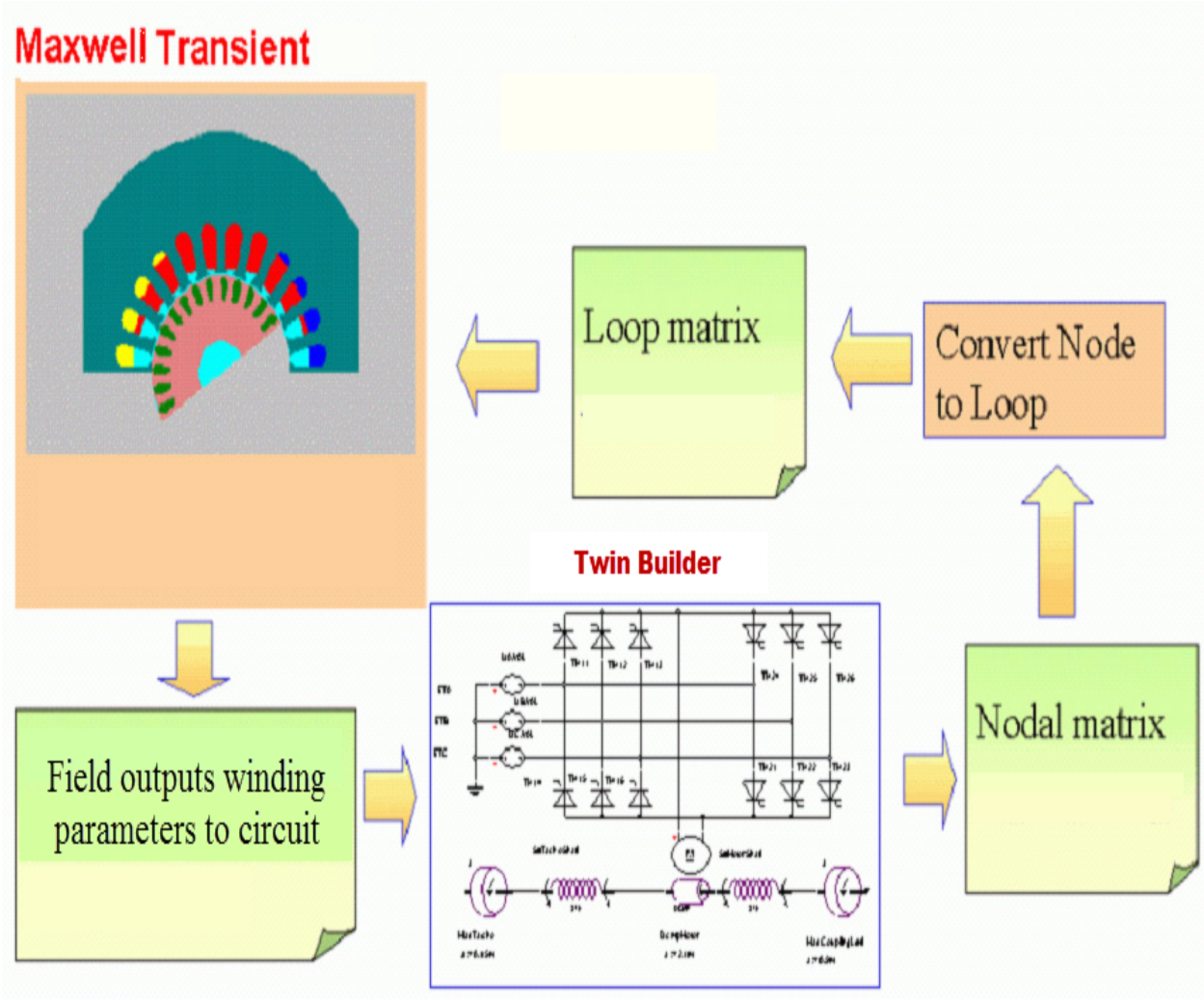
- Maxwell 2D or 3D current version Updates for the Transient Link:

> **Note:**
>
> - For two-way interactive use, Maxwell must be installed and licensed on the same computer as Twin Builder.
> - If the computer is connected to a network and is set to obtain an IP address, and the computer has an IP address in the range 192.168.x.y, disconnecting from the network while co-simulating will cause the co-simulation to stop.
> - Maxwell 2D or 3D may be installed on a different computer than the Twin Builder Installation, provided that a TCP/IP link can be established between the two computers for co-simulation.

Transient Cosimulation Component Interface

The Maxwell 2D and 3D Transient solvers incorporate circuit equations into the finite element system of equations. Maxwell uses a loop form of the circuit equations, while Twin Builder uses a nodal form of the circuit equations. After each time step, Twin Builder forms a Norton-equivalent of the drive circuit at the coupling pins between the Maxwell component and the rest of the system. Maxwell converts this to a loop matrix, solves the finite element equations, then outputs a Thevenin-equivalent for the next Twin Builder time step. This parameter-based coupling enhances solution accuracy and stability.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Twin Builder controls the time step during simulation and also controls the rotational speed or linear velocity of the Maxwell component.

At each time step each solver creates a drive circuit at the coupling pins looking into the other system:

- Twin Builder sees the Maxwell Winding as a Thevenin equivalent circuit with a specified source-voltage, resistance, and inductance at each time step.
- Maxwell sees the Twin Builder circuit as a Norton-equivalent circuit with a specified source-current and admittance at each time step.
- For the Mechanical coupling, The Position is imposed by Twin Builder to Maxwell and the Torque is imposed to Twin Builder by Maxwell.

During each time step, the two solvers exchange coupling matrix data using TCP/IP sockets. Since the data exchange is relatively small, the Maxwell 2D or 3D Transient and the Twin Builder

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

transient solvers can run on different computers, as long as they have access to each other over a network.

Maxwell 2D and 3D Finite Element Model Prerequisites

A Maxwell 2D or 3D Transient project must satisfy the following requirements before Twin Builder can link to it:

- The model must be ready to solve.
- The model must have mechanical motion.
- The model must have at least one external winding.
- It is recommended that design names avoid using illegal characters such as the “space” character. Illegal characters are replaced with an underscore character. See the section on naming conventions for additional information.
- It is recommended, but not required, that the model be solved under some conditions, such as without external windings, or with a very simple TDSLink external circuit.

> **Note:**
>
> See the Maxwell documentation for detailed information on working with Maxwell 2D and 3D projects.

After Twin Builder runs the Maxwell 2D or 3D Transient project, the model, mesh, materials, sources, and boundary conditions will remain unchanged. But the following changes will occur:

- The previous solution is over-written, and most likely is different from a standalone Maxwell 2D or 3D Transient solution.
- Time step and stopping time may change.
- Mechanical transient setup will have been disabled, but the functions are still there.
- User control program will have been replaced by *tdslink_tcp.exe*. This should not be configured manually from m2dfs.

Adding a Maxwell Transient Cosimulation Component

> **Note:**
>
> While Twin Builder lets you add multiple Maxwell Transient Cosimulation components to a schematic, only one can be active for simulation. You must deactivate all others before successful simulation can proceed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> If the Maxwell link includes motion, then one of the two mechanical terminals exposed in Twin Builder should be used as a reference.

Follow this procedure to add a Maxwell Transient Cosimulation component to a Twin Builder design:

1. On the **Twin Builder** main menu, select **Subcircuit** > **Maxwell Component** > **Add Transient Cosimulation**.

   This launches the Maxwell application. The **Maxwell Transient-Transient coupling** dialog box appears after Maxwell has finished loading.

2. On the **Link Description** tab, specify a **Name** for the component. See *Names of Components and Variables* for naming conventions.

3. In the **Source Project & Design** panel, click to locate and choose the Maxwell Finite Element model **.aedt** file containing the design you want to add. The selected project loads into Maxwell.
4. Based on the **Source** file and the **Link Type** (Maxwell Transient-Transient Link), Twin Builder communicates with Maxwell to load the selected file. A list of designs is retrieved from the Maxwell file and displayed in the **Design** drop-down list.

   The dialog box's **Information** tab updates to show the **Number of Conservative Pins** and various other **Quantities** present in the selected model. You can add pins to the component for listed **Quantities**.

5. Select the **Design** and related **Solution** you want to use from the drop-down lists on the **Link Description** tab.
6. On the **Options** tab, choose to show either the **Pin Name** or **Pin Description** on the new component. By default, pin names display.

   > **Note:**
   >
   > Pin description properties are always added to the component, regardless of whether they are selected to be shown with pins.

   - If you chose **Pin Description** when adding the component, you can subsequently enable individual pin names: open the component symbol in the symbol editor, delete a pin's description, double-click the pin, and enable **ShowName** in the pin properties dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- If you chose **Pin Name** when adding the component, you can subsequently hide individual pin names. Open the component symbol in the symbol editor, double-click a pin and disable **ShowName** in the pin properties dialog box. You can then add descriptive text to the pin using the editor's **Draw** > **Text** tool.
- Optionally, choose to **Save** or **Unload** the Maxwell project after use by selecting the appropriate check boxes.

7. When finished, click **OK** to accept the values and close the dialog box, or **Cancel** to close the dialog without the changes.
8. Place the component on the schematic sheet.
9. After placing the component on a sheet, the **Properties** dialog box may be opened for configuration of the component. The component includes the following outputs on the **Quantities** tab, which may be plotted in display elements or saved with simulation results:
   - For each winding: the current, terminal voltage, flux linkage, and internal EMF.
   - Total core loss.
   - Total winding conductor loss.
   - For linear motion: the force, displacement, and velocity.
   - For rotational motion: the torque, angle, and speed.

   > **Note:**
   >
   > The Transient Cosimulation coupling configuration is saved with the schematic sheet. Twin Builder must establish a link to Maxwell 2D each time the simulation is run.

Preparing a Maxwell 2D or 3D Transient model for a link to Twin Builder

The following steps outline the process for preparing and using a Maxwell 2D or 3D Transient component in a Twin Builder Transient simulation.

1. Open a Maxwell project or create a new project using the Maxwell Desktop.
2. Right-click the design name in the **Project Manager** pane and select **Solution Type**. The **Solution Type** dialog box appears.
3. Select the **Transient** radio button in the **Magnetic** section and click **OK**.
4. Right-click the design name in the **Project Manager** pane and select **Design Settings**.
   a. If necessary, select the **Symmetry Multiplier** tab and type the appropriate symmetry value.
   b. Select the **Advanced Product Coupling** tab and select the **Enable transient link with Twin Builder** check box.
   c. If a 2D design is involved, select the **Model Depth** tab and enter a value, and the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

appropriate units, to represent the length of the device.

d. Click **OK** to accept the changes and close the dialog box.

5. Transient-Transient coupling requires a motion setup in Maxwell as follows:
   a. Right-click **Model** in the **Project Manager** pane and select **Motion Setup/Assign Band...**.
   b. Choose the **Type** tab within the **Motion Setup** dialog box and set the desired **Motion Type** and the corresponding **Moving Vector**.
   c. Click **OK**.

> **Note:**
>
> You do not need to define an initial position since the initial condition is established by the Twin Builder simulator.

6. Define the appropriate boundary conditions on the **Maxwell2D/Boundaries** menu.
7. Define the model excitation on the **Maxwell2D/Excitations** menu. The Maxwell model must have at least one external winding as follows:
   a. When assigning the coil excitation, enter the coil **Name**, the **Number of Conductors**, and the appropriate **Polarity** and click **OK**. This step must be performed for each coil definition.
   b. When assigning a **Winding** configuration for a selected **Coil Excitation**, the **Winding Type** should be **External** as either **Solid** or **Stranded**. Define the number of parallel branches and click **OK**.

> **Note:**
>
> You do not need to define an initial current since the initial condition will be established by Twin Builder simulator.

8. Add a solution setup under **Analysis**.

> **Note:**
>
> The Maxwell simulation end time (Tend) must be greater than or equal to that of Twin Builder. All the other options within the solve setup are optional as far the Twin Builder coupling task is concerned.

9. **Save** the Maxwell Project. If you exit the Maxwell simulator, Twin Builder will launch it again in a non-graphical “Extractor” mode to establish the link. If you leave Maxwell open, the link is established with the open Maxwell Desktop and you will be able to monitor the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

solution process in Maxwell as well as any messages that displays during the simulation process.

For more information regarding motion setup, boundaries, excitations, or solution setup in Maxwell, consult the Maxwell documentation.

How to Use a Maxwell 2D or 3D Finite Element Model in Twin Builder

The following steps outline the process for preparing and using a Maxwell 2D or 3D Finite Element model in Twin Builder.

1. Create a model of the component in Maxwell 2D or 3D Transient, including the geometry, a moving band, materials, boundary conditions, external windings, and mesh.
2. Solve the problem in Maxwell to ensure that the Maxwell 2D or 3D Transient project is set up correctly. Errors in the Maxwell project cannot be corrected from Twin Builder.

   Note:

   - See the Maxwell documentation for detailed information on creating and solving the Maxwell 2D or 3D model.
   - This preliminary Maxwell solution need not be for the same conditions or ending time that Twin Builder will establish.

3. Ensure that Twin Builder has run-time access to Maxwell, either on the same computer, or another computer on the network via TCP/IP.
4. Establish a link to the Maxwell 2D or 3D Transient project, and place a Maxwell Transient Cosimulation Component on the Twin Builder schematic.
5. Connect electrical and mechanical pins of the Maxwell Transient Cosimulation component to other models on the Twin Builder schematic.
6. Select a Transient Analysis Setup for the simulation. Attempting to simulate using either AC and DC analysis setups will fail setup with resulting error messages in the **Message Manager**.
7. Run the simulation in Twin Builder.

   Maxwell 2D or 3D Transient will start in a co-simulation mode.

Continuing a Cosimulation with Maxwell

After a successful simulation, either to completion or clean-stopped, you can continue the simulation from the current end time.

To enable this possibility:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- You must first have saved a Simulation State (KRN) file at the end of simulation.

  > **Note:**
  >
  > When using Twin Builder-Maxwell Transient Cosimulation, save / load state only works with the largest time at which Maxwell was solved. This is because Maxwell can currently only save one state. While it is possible to start from an earlier saved time, results may be incorrect.
  >
  > For example, suppose you solve a transient analysis up through one second and save the simulation state to a file named **one_second.krn**, then load **one_second.krn** and solve up through two seconds and save the new state to a file named **two_seconds.krn**. If you then reload **one_second.krn** and solve, Twin Builder may give incorrect results because the internal state of Maxwell is as it was at two seconds. Resuming from **two_seconds.krn** will give correct results.

- You must load the saved KRN file at the beginning of the continuing simulation.

  > **Note:**
  >
  > To specify the saving and loading of Simulation State (KRN) files, see *Design Settings*.

- You must specify an end time greater than the current end time. To reset the end time, double-click the analysis setup in the project window.

Start the analysis. Simulation can be extended this way as many times as desired.

> **Note:**
>
> If Maxwell is used non-interactively—that is, when you select the **Save project after use** check box—it shuts down after each simulation and results data is lost if not saved, so a KRN file cannot be used successfully.
>
> If you select the **Save project after use** check box, the end time in the setup must be made greater before the KRN file is loaded or Twin Builder simulation will fail.
>
> If KRN file save or load is set and the Maxwell project is not set to be saved the following error message displays:
>
> “The Maxwell TR-TR coupling component is not set to save the Maxwell project, but saving or loading a state file has been specified. Maxwell results valid at the state file's end time will be necessary when using a state file.”

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

Unlike usage in a non-cosimulation setting, a specific Simulation State (KRN) file cannot be reused once Maxwell has results at a later time. Also, the **Enable continue to solve** option is ignored for cosimulation with Maxwell.

Maxwell Equivalent Circuit Component

Maxwell is an interactive software package that uses finite element analysis to solve two-dimensional (Maxwell 2D) or three-dimensional (Maxwell 3D) electromagnetic problems. To analyze a problem, the component designer specifies the appropriate geometry, material properties, and excitations for a device or a system of devices. Use the Export Equivalent Circuit function in Maxwell to create a circuit model whose behavior at the input/output pins models the characteristic of the device based on the solution of the electromagnetic fields.

When you use this subcircuit in a Twin Builder design, Maxwell solves the equivalent circuit problem first, then exports a model for Twin Builder. These models are used for AC, DC, and transient analysis in Twin Builder.

This chapter contains information on:

- Maxwell Equivalent Circuit Component Program Requirements
- Maxwell Equivalent Circuit Component Interface Concept
- Adding a Maxwell 2D Finite Element Component
- Preparing a Maxwell Equivalent Circuit Model for Twin Builder
- How to Use a Maxwell Equivalent Circuit Model in Twin Builder

Maxwell Equivalent Circuit Component Program Requirements

- Twin Builder current version.
- Maxwell 2D or Maxwell 3D current version.

Note:

- For two-way interactive use, Maxwell must be installed and licensed on the same computer as Twin Builder. However, Maxwell equivalent circuit models may be used in Twin Builder without a Maxwell license.
- If the computer is connected to a network and is set to obtain an IP address, and the computer has an IP address in the range 192.168.x.y, disconnecting from the network while co-simulating will cause the co-simulation to stop.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Maxwell Equivalent Circuit Component Interface Concept

The Maxwell Equivalent Circuit Component interface allows the offline coupling between a Twin Builder and a Maxwell model. You can also integrate couplings to other simulators such as Simulink. During simulation an automatic step width control in Twin Builder takes place. Equivalent Circuit models are created independently in Maxwell, then imported by Twin Builder and placed on a schematic as components.

Adding a Maxwell Equivalent Circuit Component

To add a Maxwell Equivalent Circuit component to a Twin Builder design:

1. On the **Twin Builder** main menu, select **Subcircuit** > **Maxwell Component** > **Add Equivalent Circuit**.

   This launches the Maxwell application. The **Maxwell Equivalent Circuit Model** dialog box appears after Maxwell has finished loading.

2. Specify a **Name** for the component. See *Names of Components and Variables* for naming conventions.

   > **Note:**
   >
   > The model generated with this component is given the same name as the component. Also, the specified name cannot be the same as any existing coupling component and/or model.

3. In the **Source Project & Design** panel, click to locate and choose the Maxwell EC model **.aedt** file containing the design you want to add.

   > **Note:**
   >
   > - Design names must avoid using illegal characters such as the “space” character. Illegal characters are replaced with an underscore character. See *Names of Components and Variables* for additional information.

   The selected project loads into Maxwell.

4. Based on the **Source** file and the **Link Type** (Maxwell Equivalent Circuit Link), Twin Builder communicates with Maxwell to load the selected file. A list of designs is retrieved from the Maxwell file and displayed in the **Design** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The dialog box's **Information** panel also updates to show the **Number of Conservative Pins** and various other **Quantities** present in the selected model. You can add pins to the model for listed **Quantities**.

5. Select the **Design** you want to use from the drop-down list. You will have all the designs in the drop-down list, with the design name followed by the Maxwell solution type in parentheses.

   The transient solution is typically much faster than the parametric analysis of magnetostatic or electrostatic solutions. The equivalent circuit model of a Maxwell design with transient solution type supports a winding model, 3-phase model, and rotational model.

6. Click **Extract Equivalent Circuit**.

   The **Export Equivalent Circuit** > **From Parametric Solutions** wizard launches in Maxwell. See the Maxwell documentation for help with the wizard.

   The Maxwell Desktop displays with the **Export Equivalent Circuit** wizard running. Select the **Model Type** from the list below for further instructions:

   - Linear Motion
   - Rotational Motion
   - Matrix
   - Transformer
   - Lookup Table

   When finished with the wizard, the **Quantities** listing updates. Select any parameter pins you want to include in the imported model.

7. On the **Options** tab, choose to show either the **Pin Name** or **Pin Description** on the new component. Pin names display by default.

   > **Note:**
   >
   > Pin description properties are always added to the component regardless of whether they are selected to be shown with pins.

   - If you chose **Pin Description** when adding the component, you can subsequently enable individual pin names. Open the component symbol in the symbol editor, delete a pin's description, double-click the pin and enable **ShowName** in the pin properties dialog box.
   - If you chose **Pin Name** when adding the component, you can subsequently hide individual pin names. Open the component symbol in the symbol editor, double-

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

click a pin and disable **ShowName** in the pin properties dialog box. You can then add descriptive text to the pin using the editor's **Draw** > **Text** tool.

- Select the **Unload project after use** check box to cause Maxwell to close the project and shut down after the Twin Builder simulation is complete. This will release the Maxwell license.

8. When finished, click **OK** to accept the values.

   Twin Builder generates the equivalent circuit model component and closes the dialog box.

> **Note:**
>
> The model generated with this component is given the same name as the component. Also, the specified name cannot be the same as any existing coupling component and/or model. An informational dialog box displays if the name chosen in *step 2* is already used by another coupling component or model.

9. Place the component on the schematic sheet.
10. After placing the component on a sheet, its **Properties** dialog box may be opened for configuration of the component. The component includes the following outputs on the **Quantities** tab, which may be plotted in display elements or saved with simulation results:
    - For each winding, the current, terminal voltage, flux linkage, and internal EMF
    - Total core loss
    - Total winding conductor loss
    - For linear motion, the force, displacement, and velocity
    - For rotational motion, the torque, angle, and speed

> **Note:**
>
> Right-click the component and select **Edit Model** for a description of the ports in the component.

The Equivalent Circuit coupling configuration is saved with the schematic sheet. Twin Builder must establish a link to Maxwell each time the simulation runs.

Preparing a Maxwell Equivalent Circuit Model for Twin Builder

The following steps outline the process for preparing and using a Maxwell Equivalent Circuit model in Twin Builder.

1. Create a model of the component in Maxwell, including the geometry and material properties.
2. Define the appropriate boundary conditions under **Boundaries**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Define the model excitation under **Excitations**.
   - When assigning the excitation as **Current**, enter the **Current Excitation Name**, the **Value** as an independent variable, the **Type** as **Solid** or **Stranded** and click **OK**.

     > **Note:**
     >
     > This step must be performed for each terminal or group of terminals are intended to be included on current excitation definitions.

4. Define the appropriate model parameters, assigning them under **Parameters**.
   a. When considering exporting a matrix into the circuit design, you must assign a **Matrix**:
      - Right-click **Parameters** and select **Assign** > **Matrix**.
      - In the **Matrix** dialog box, select the **Include** check box for each of the existing sources to be included into the matrix calculation.
      - Use the **Post Processing** tab to specify turns and group together windings to better represent the physical winding topology.

   b. When considering exporting a specific force calculation into circuit design as a measure of the mechanical through quantity of a linear motion design, a solid object must be selected and a **Force** parameter defined:
      - Select a portion of the geometry on which to calculate the force.
      - Right-click **Parameters** and select **Assign** > **Force**.
      - In the **Force Setup** dialog box, enter a **Name** for the force parameter.
      - Select the **Type** of the force computation, either **Virtual** or **Lorentz**.
      - Use the **Post Processing** tab to set a coordinate system of reference.

   c. When considering exporting a specific torque calculation into circuit design as a measure of the mechanical through quantity of a rotational motion design, a solid object must be selected and a **Torque** parameter defined:
      - Select a portion of the geometry on which to calculate the torque.
      - Right-click **Parameters** and select **Assign** > **Torque**.
      - In the **Torque** dialog box, enter a **Name** for the **Torque** parameter.
      - Select the **Type** of the torque computation, either **Virtual** or **Lorentz**.
      - Select the **Axis** about which to calculate the torque, and the rotational sign.

5. Solve the problem in Maxwell to ensure that the Maxwell project is set up correctly. Errors in the Maxwell project cannot be corrected from Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> See the Maxwell documentation for detailed information on creating and solving the Maxwell model.

6. Add a **Solution Setup** under **Analysis**. The Maxwell simulation end time (Tend) must be greater than or equal to that of Twin Builder.
7. Add a **Parametric Setup** within **Optimetrics**.

   The parametric setup must have the selected independent variables varied to form a complete nested parametric solution. For example, if two variables are entered as Current with the variations 1, 2, 3 and second independent variable Position with the variations 0, 10, 20 then the parametric table will have for first variation value of the Current 1, a sweep of all available variations for Position, and so on. So the very first three rows within the parametric table represent the very first parametric nest solution and it will be sufficient to extract an equivalent circuit to be imported into circuit design.

> **Note:**
>
> In order to extract an equivalent circuit within the **Parametric Setup** there is no need to define any **Calculations** entries unless this is a user choice to bring more field-related quantities as outputs into the circuit design.

8. **Save** the Maxwell Project.
9. Ensure that Twin Builder has runtime access to Maxwell on the same computer or another computer on the network.
10. Establish a link to the Maxwell Equivalent Circuit model project, and place a Maxwell Equivalent Circuit Model component on the Twin Builder schematic.
11. Connect electrical and mechanical pins of the Maxwell Equivalent Circuit model component to other models on the Twin Builder schematic.

> **Note:**
>
> Equivalent Circuit models exported from Maxwell that have mechanical pins must be connected by these pins to a mechanical subsystem having a ROTATIONAL _V nature. For versions of Maxwell lower than Version 10, Service Pack 1, this must be done using a Domain to Domain component. For later versions, the connection can be made directly.

12. Run the simulation in Twin Builder.
13. Switch back to Maxwell if you need to update the model design.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Linear Motion Equivalent Circuits

When exporting a **Linear Motion** model type:

1. Choose the appropriate **Parametric Setup** (defined within Optimetrics).
2. Choose the **Solution Setup**.
3. Choose the **Matrix Setup**.
4. Choose the **Force Setup** and select its coordinate representation as component.
5. Choose the appropriate **Current Variables Represent** value as Ampere-Turns or Amperes.

> **Note:**
>
> If within the Excitation definition the Solid configuration is selected to represent the topology of the winding, then the swept values of the current independent variable in the parametric table has a physical meaning of Amperes; if within the Excitation definition the Stranded configuration is selected to represent the topology of the winding, then the swept values of the current independent variable in the parametric table has a physical meaning of Amperes-Turns.

6. Click **Next**.
7. Under **Circuit Inputs and Outputs**, select the appropriate **Type** for each entry (for example, **Current** for independent variable which stands for current excitation, and **Position** for independent variable which stands for motion definition) and define the Extrapolation method only for Inputs based on specific design considerations choosing the Bezier Interpolation method (for both Inputs and Outputs) or using the default Linear Interpolation method.
8. Click **Next**.
9. In the **Terminals** dialog box, you must define the **Scaling Factor** (the default is one, as the geometry and **Boundary Conditions** are related to the whole device). You may set accordingly the **Coil Terminals** (in terms of Resistance, Turns and Branches) based on the definition of the **Matrix** entry. The **Mechanical Terminals** are selected based on the selections made at the **Circuit Inputs and Outputs** dialog box.

Rotational Motion Equivalent Circuits

When exporting a **Rotational Motion** model type:

1. Choose the appropriate **Parametric Setup** (defined within Optimetrics).
2. Choose the **Solution Setup**.
3. Choose the **Matrix Setup**.
4. Choose the **Torque Setup**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Choose the appropriate **Current Variables Represent** value as Ampere-Turns or Amperes.

> **Note:**
>
> If within the Excitation definition the Solid configuration is selected to represent the topology of the winding, then the swept values of the current independent variable in the parametric table has a physical meaning of Amperes; or if within the Excitation definition the Stranded configuration is selected to represent the topology of the winding, then the swept values of the current independent variable in the parametric table has a physical meaning of Amperes-Turns.

6. Click **Next**.
7. Under **Circuit Inputs and Outputs**, select the appropriate Type for each entry (for example, Current for independent variable which stands for current excitation and Rotation for independent variable which stands for motion definition) and define the Extrapolation method only for Inputs based on specific design considerations choosing the Bezier Interpolation method (for both Inputs and Outputs) or using the default Linear Interpolation method.
8. Click **Next**.
9. In the **Terminals** dialog box, define the Scaling Factor (the default is one, as the geometry and Boundary Conditions are related to the whole device). You may set accordingly the Coil Terminals (in terms of Resistance, Turns and Branches) based on the definition of the Matrix entry. The Mechanical Terminals are selected based on the selections made in the **Circuit Inputs and Outputs** dialog box. Select the box to Use rotational velocity as mechanical terminals conservative definition for the circuit design component (for example, if the box is cleared, the Displacement-Force-Representation as Rotational is used otherwise the Velocity-Force-Representation as Rotational_V is employed).

Matrix Equivalent Circuits

When exporting a **Matrix** model type:

1. Choose the appropriate **Parametric Setup** (defined within Optimetrics).
2. Choose the **Solution Setup**.
3. Choose the **Matrix Setup**.
4. Choose the appropriate **Current Variables Represent** value as Ampere-Turns or Amperes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> If within the Excitation definition the Solid configuration is selected to represent the topology of the winding, then the swept values of the current independent variable in the parametric table has a physical meaning of Amperes; or if within the Excitation definition the Stranded configuration is selected to represent the topology of the winding, then the swept values of the current independent variable in the parametric table has a physical meaning of Amperes-Turns.

5. Click **Next**.
6. Under **Circuit Inputs and Outputs**, select the appropriate Type for each entry (for example, Current for independent variable which stands for current excitation) and define the Extrapolation method only for Inputs based on specific design considerations choosing the Bezier Interpolation method (for both Inputs and Outputs) or using the default Linear Interpolation method.
7. Click **Next**.
8. In the **Terminals** dialog box, define the Scaling Factor (the default is one, as the geometry and Boundary Conditions are related to the whole device). You may set accordingly the Coil Terminals (in terms of Resistance, Turns and Branches) based on the definition of the Matrix entry.

Transformer Equivalent Circuits

When exporting a **Transformer** model type:

1. Choose the appropriate **Parametric Setup** (defined within Optimetrics).
2. Choose the **Solution Setup**.
3. Choose the **Matrix Setup**.
4. Choose the appropriate **Current Variables Represent** value as Ampere-Turns or Amperes.

> **Note:**
>
> If within the Excitation definition the Solid configuration is selected to represent the topology of the winding, then the swept values of the current independent variable in the parametric table has a physical meaning of Amperes; if within the Excitation definition the Stranded configuration is selected to represent the topology of the winding, then the swept values of the current independent variable in the parametric table has a physical meaning of Amperes-Turns.

5. Click **Next**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. Under **Circuit Inputs and Outputs**, select the appropriate Type for each entry (for example, Current for independent variable which stands for current excitation) and define the Extrapolation method only for Inputs based on specific design considerations choosing the Bezier Interpolation method (for both Inputs and Outputs) or using the default Linear Interpolation method.
7. Click **Next**.
8. In the **Terminals** dialog box, define the Scaling Factor (the default is one, as the geometry and Boundary Conditions are related to the whole device). You may set accordingly the Coil Terminals (in terms of Resistance, Turns and Branches) based on the definition of the Matrix entry.

Lookup Table Equivalent Circuits

When exporting a **Lookup Table** model type:

1. Choose the appropriate **Parametric Setup** (defined within Optimetrics).
2. Click **Next**.
3. Under **Circuit Inputs and Outputs**, select the appropriate Type for each entry (for example, Current for independent variable which stands for current excitation and Rotation for independent variable—if any—which stands for motion definition) and define the Extrapolation method only for Inputs based on specific design considerations choosing the Bezier Interpolation method (for both Inputs and Outputs) or using the default Linear Interpolation method.

> **Note:**
>
> - In case of Lookup Table choice you must enter **Calculations** under **Parametric Setup** within Optimetrics to be able to get output entries in the **Table** dialog box within the extraction wizard.
> - Use Twin Builder compiler supported characters when naming these calculations entries within Optimetrics, otherwise you will encounter Twin Builder compiler errors when instantiating the ECE component.

How to Use a Maxwell Equivalent Circuit Model in Twin Builder

The following steps outline the process for preparing and using a Maxwell Equivalent Circuit model in Twin Builder.

1. Create a model of the component in Maxwell, including the geometry, materials, boundary conditions, external windings, and mesh.
2. Solve the problem in Maxwell to ensure that the Maxwell project is set up correctly. Errors in the Maxwell project cannot be corrected from Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> - See the Maxwell documentation for detailed information on creating and solving the Maxwell model.

3. Ensure that Twin Builder has runtime access to Maxwell on the same computer or another computer on the network.
4. Establish a link to the Maxwell Equivalent Circuit model project, and place a Maxwell Equivalent Circuit component on the Twin Builder schematic.
5. Connect electrical and mechanical pins of the Maxwell Equivalent Circuit component to other models on the Twin Builder schematic.

> **Note:**
>
> Equivalent Circuit models exported from Maxwell that have mechanical pins must be connected by these pins to a mechanical subsystem having a ROTATIONAL_V nature. For versions of Maxwell lower than Version 10, Service Pack 1, this must be done using a Domain to Domain component. For later versions, the connection can be made directly.

6. Run the simulation in Twin Builder.
7. Switch back to Maxwell if you need to update the model design.

Exporting a Maxwell Equivalent Circuit Model

You can export an Equivalent Circuit model to an .**sml** file. The exported model can then be imported and used by a Twin Builder user who does not have Maxwell.

To export a Maxwell Equivalent Circuit model to a file:

1. Create the Maxwell 2D or 3D model with the required coupling information.
2. Solve the model to ensure that the EC model is set up correctly.
3. Export the EC model as an .**sml** file using **Maxwell 3D** > **Export Equivalent Circuit** > **From Parametric Solution**. A corresponding .**gif** graphic file is also created. This graphic appears on the schematic.

> **Note:**
>
> If you are sending this model to another user, the exported .**sml** file is required but the .**gif** file is optional.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> See ***Exporting Equivalent Circuit Data*** in the Maxwell help for more information.

Maxwell Dynamic Magnetostatic Coupling Component

Use Ansys Electromagnetics Maxwell Dynamic Magnetostatic coupling components to extract inductance data from a Maxwell inductance matrix solution and use the data in a Twin Builder circuit.

This section contains information on:

- Program Requirement for Maxwell Dynamic Magnetostatic Coupling
- Maxwell Dynamic Magnetostatic Coupling Component Interface Concept
- Overview of Maxwell Dynamic Magnetostatic Coupling
- Adding a Maxwell Dynamic Magnetostatic Coupling Component

Program Requirements for Maxwell Dynamic Magnetostatic Coupling

- Twin Builder current version.
- Ansys Electromagnetics Maxwell current version.

Maxwell Dynamic Magnetostatic Coupling Component Interface Concept

The Maxwell Dynamic Magnetostatic component provides dynamic coupling between Twin Builder and the electromagnetic field simulator. The model is very efficient, and it allows the use of multiple instances, and coupling with other components and simulators. During simulation, the component can limit the Twin Builder time step for accurate results.

Overview of Maxwell Dynamic Magnetostatic Coupling

To prepare and use a Maxwell Dynamic Magnetostatic component, do the following:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Create a Magnetostatic model of the device in Maxwell.

   > **Note:**
   >
   > Avoid using illegal characters such as the “space” character in design names. Illegal characters are replaced with an underscore character. See the section on naming conventions for additional information.

2. Assign a **Matrix** calculation in the **Parameters** folder of Maxwell.
3. Perform a Magnetostatic simulation with an inductance matrix parameter setup.
4. In Twin Builder, add the Maxwell Dynamic Magnetostatic coupling component to the sheet.
5. Create the rest of the simulation design on the sheet.
6. Start the simulations in Twin Builder.

Adding a Maxwell Dynamic Magnetostatic Coupling Component

To add a Maxwell Dynamic Magnetostatic component to a Twin Builder design:

1. On the **Twin Builder** main menu, select **Subcircuit** > **Maxwell Component** > **Add Dynamic Magnetostatic**.

   The **Maxwell Dynamic Magnetostatic Coupling** dialog box appears.

2. Specify a **Name** for the component. See ***Names of Components and Variables*** for naming conventions.

3. In the **Source Project & Design** panel, click to open a file browser window, and locate and choose the Maxwell **.aedt** file containing the design you want to add.

   > **Note:**
   >
   > Avoid using illegal characters such as the “space” character in design names. Illegal characters are replaced with an underscore character. See the section on naming conventions for additional information.

   The selected project is loaded into Maxwell.

4. Based on the **Source** file and the **Link Type** (Maxwell Magnetostatic Link), Twin Builder communicates with Maxwell to load the selected file. A list of designs is retrieved from the Maxwell file and displayed in the **Design** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The dialog box's **Information** tab is also updated to show the **Number of Conservative Pins** and various other **Quantities** present in the selected design. You can choose to add pins to the coupling component for listed **Quantities**.

5. Select the **Design** you want to use from the drop-down list.
6. Select the **Solution** and **Matrix** to use from the selected **Design**.
7. For 2D designs, a **Depth** field is displayed. Enter a positive number for the design length in this field. The default unit of length is "meter". You can change this to any valid unit of length ("ft" or "cm", for example). If you omit the unit of length, the component will use the default unit of length set in the **Default Units** tab of the **General Options** dialog box.

> **Note:**
>
> Depth is controlled based on the **COMPONENT_DEPTH** design quantity of the component instance, so changes will propagate only by changing this property either on the Quantities tab of the component's **Properties** dialog box or in the Depth field in the coupling component dialog box. However, in designs created or saved in Twin Builder, the quantity will be shown as **DEPTH_FROM_ SIMPLORER**.

8. Optionally, set the **Scale** to be passed on to Maxwell during simulation, where it performs the same function as the Maxwell Symmetry Multiplier setting. Maxwell then passes the results back to Twin Builder. (See ***Setting a Symmetry Multiplier*** in the Maxwell help for additional information.)

> **Note:**
>
> Coupling components created in Twin Builder versions prior to version 10 may need to be recreated to enable use of the Scale setting.

9. In the **Options** tab, choose to show either the **Pin Name** or **Pin Description** on the new component. By default, pin names are shown.

> **Note:**
>
> Pin description properties are always added to the component regardless of whether they are selected to be shown with pins.

   - If you chose **Pin Description** when adding the component, you can subsequently enable individual pin names. Open the component symbol in the symbol editor, delete a pin's description, double-click the pin and enable **ShowName** in the pin properties dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- If you chose **Pin Name** when adding the component, you can subsequently hide individual pin names. Open the component symbol in the symbol editor, double-click a pin, and disable **ShowName** in the pin properties dialog box. You can then add descriptive text to the pin using the editor's **Draw** > **Text** tool.

10. You can select the following additional options on the **Options** tab:
    - Select the **Save project after use** check box to cause Maxwell to save any solutions data after the Twin Builder simulation, overwriting any previously saved data.
    - Select the **Unload project after use** check box to cause Maxwell to close the project and shut down after the Twin Builder simulation is complete. This will release the Maxwell license.
    - The **Create static link** option will force Twin Builder to load the data currently available from the project and maintain it internally. Additional calls to Maxwell will not be performed unless the properties of the component are changed. This option can result in faster run times by minimizing the number of calls to Maxwell for data sharing. Click **Clear Static Cache** to force Twin Builder to remove the cached data and acquired new data from Maxwell.
11. When finished, click **OK** to accept the values.

    Twin Builder generates the equivalent circuit coupling component and closes the dialog box.
12. Place the component on the schematic sheet as you would any other component.
13. After placing the component on a sheet, its **Properties** dialog box may be opened for configuration of the component.

Maxwell Dynamic Electrostatic Coupling Components

Use Ansys Electromagnetics Maxwell Dynamic Electrostatic coupling components to extract capacitance data from a Maxwell capacitance matrix solution and use the data in a Twin Builder circuit.

This section contains information on:

- Program Requirement for Maxwell Dynamic Electrostatic Coupling
- Maxwell Dynamic Electrostatic Coupling Component Interface Concept
- Overview of Maxwell Dynamic Electrostatic Coupling
- Adding a Maxwell Dynamic Electrostatic Coupling Component

Program Requirements for Maxwell Dynamic Electrostatic Coupling

- Twin Builder current version.
- Ansys Electromagnetics Maxwell current version.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Maxwell Dynamic Electrostatic Coupling Component Interface Concept

The Maxwell Dynamic Electrostatic coupling component provides dynamic coupling between Twin Builder and the electromagnetic field simulator. The coupling component is very efficient, allowing the use of multiple instances, and coupling with other models and simulators. During simulation, the coupling component can limit the Twin Builder time step for accurate results.

Overview of Maxwell Dynamic Electrostatic Coupling

To prepare and use a Maxwell Dynamic Electrostatic coupling component:

1. Create an electrostatic model of the device in Maxwell.

   > **Note:**
   >
   > Avoid using illegal characters such as the “space” character in design names. Illegal characters are replaced with an underscore character. See *Names of Components and Variables* for additional information.

2. Assign a **Matrix** calculation in the **Parameters** folder of Maxwell.
3. Perform an electrostatic simulation and compute the capacitance matrix.
4. In Twin Builder, add the Maxwell Dynamic Electrostatic component to the sheet.
5. Create the rest of the simulation design on the sheet.
6. Start the simulations in Twin Builder.

Adding a Maxwell Dynamic Electrostatic Coupling Component

To add a Maxwell Dynamic Electrostatic component to a Twin Builder design:

1. On the **Twin Builder** main menu, select **Subcircuit** > **Maxwell Component** > **Add Dynamic Electrostatic**. The **Maxwell Dynamic Electrostatic Coupling** dialog box appears.
2. Specify a **Name** for the component. See ***Names of Components and Variables*** for naming conventions.
3. In the **Source Project & Design** panel, click to locate and choose the Maxwell **.aedt** file containing the design you want to add.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Avoid using illegal characters such as the “space” character in design names. Illegal characters are replaced with an underscore character. See ***Names of Components and Variables*** for additional information.

The selected project loads into Maxwell.

4. Based on the **Source** file and the **Link Type** (Maxwell Electrostatic Link), Twin Builder communicates with Maxwell to load the selected file. A list of designs is retrieved from the Maxwell file and displayed in the **Design** drop-down list.

   The **Information** tab updates to show the **Number of Conservative Pins** and various other **Quantities** present in the selected design.

5. Select the **Design** you want to use from the drop-down list.
6. Select the **Solution** and **Matrix** to use from the selected **Design**.
7. For 2D designs, a **Depth** field displays. Enter a positive number for the design length in this field. The default unit of length is the meter. You can change this to any valid unit of length (“ft” or “cm”, for example). If you omit the unit of length, the component uses the default unit of length set in the **Default Units** tab of the **General Options** dialog box.

> **Note:**
>
> DDepth is controlled based on the **COMPONENT_DEPTH** design quantity of the component instance, so changes will propagate only by changing this property either in the **Quantities** tab in the component's **Properties** dialog box or in the **Depth** field in the coupling component dialog box. However, in designs created or saved in Twin Builder, the quantity is shown as **DEPTH_FROM_SIMPLORER**.

8. Optionally, set the **Scale** to be passed on to Maxwell during simulation, where it performs the same function as the Maxwell Symmetry Multiplier setting. Maxwell then passes the results back to Twin Builder. (See *Setting a Symmetry Multiplier* in the Maxwell help for additional information.)

> **Note:**
>
> Coupling components created in Twin Builder versions prior to version 10 may need to be recreated to enable use of the scale setting.

9. In the **Options** tab, choose to show either the **Pin Name** or **Pin Description** on the new component. Pin names display by default.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Pin description properties are always added to the component regardless of whether they are selected to be shown with pins.

- If you chose **Pin Description** when adding the component, you can subsequently enable individual pin names. Open the component symbol in the symbol editor, delete a pin's description, double-click the pin and enable **ShowName** in the pin properties dialog box.
- If you chose **Pin Name** when adding the component, you can subsequently hide individual pin names. Open the component symbol in the symbol editor, double-click a pin, and disable **Show Name** in the pin properties dialog box. You can then add descriptive text to the pin using the editor's **Draw** > **Text** tool.

10. You can select the following additional options on the **Options** tab:
    - Select the **Save project after use** check box to cause Maxwell to save any solutions data after the Twin Builder simulation, overwriting any previously saved data.
    - Select the **Unload project after use** check box to cause Maxwell to close the project and shut down after the Twin Builder simulation is complete. This will release the Maxwell license.
    - The **Create static link** option forces Twin Builder to load the data currently available from the project and maintain it internally. Additional calls to Maxwell will not be performed unless the properties of the component change. This option results in faster run times by minimizing the number of calls to Maxwell for data sharing. Click **Clear Static Cache** to force Twin Builder to remove the cached data and acquired new data from Maxwell.
11. When finished, click **OK** to accept the values.

    Twin Builder generates the component and closes the dialog box.
12. Place the component on the schematic sheet.
13. After placing the component on a sheet, its **Properties** dialog box may be opened for configuration of the component.

Maxwell Dynamic Eddy Current Coupling Components

When working with Maxwell Dynamic Eddy components, you can insert an S-Parameter model or an RLCG model to create an equivalent model at the last adaptive frequency, providing a dynamic link between Maxwell and Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> A dynamic link coupling component always checks the external product for updated ROM data before simulating.

The simulation end time (tend) of Maxwell must be greater than or equal to that of Twin Builder.

This section contains information on:

- Program Requirement for Maxwell Dynamic Eddy Current Coupling
- Overview of Maxwell Dynamic Eddy Current Coupling
- Adding a Maxwell Dynamic Eddy Current Coupling Component

Program Requirements for Maxwell Dynamic Eddy Current Coupling

- Twin Builder current version.
- Ansys Electromagnetics Maxwell current version.

Overview of Maxwell Dynamic Eddy Current Coupling

To prepare and use a Maxwell Dynamic Eddy Current coupling component:

1. Create a 2D or 3D Eddy Current model of the device in Maxwell.

   > **Note:**
   >
   > Avoid using illegal characters such as the “space” character in design names. Illegal characters are replaced with an underscore character. See *Names of Components and Variables* for additional information.

2. Perform a frequency sweep electromagnetic field solution using the Eddy Current solver.
3. In Twin Builder, add the Maxwell Dynamic Eddy Current component to the sheet.
4. Create the rest of the simulation design on the sheet.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Start the simulations in Twin Builder.

> **Note:**
>
> - To make changes in the Maxwell Design, you must edit the design, recalculate the field solution, and re-export the design.
> - The dynamic link with the Maxwell Dynamic Eddy Current design will not solve if a change is made to the analysis settings. Most importantly, if the sweep type changes and therefore the frequency points themselves change, solving through Twin Builder will not cause Maxwell to solve. The intersection of frequencies in the old and new analyses will be used. You must go into Maxwell and manually solve to use the new frequency sweep.

Adding a Maxwell Eddy Current Dynamic Coupling Component

Follow this procedure to add a Maxwell Eddy Current Dynamic component to a Twin Builder design.

1. On the **Twin Builder** main menu, select **Subcircuit** > **Maxwell Component** > **Add Dynamic Eddy Current**. The **Maxwell Eddy Current Dynamic Coupling** dialog box appears.
2. On the **Link Description** tab, specify a **Name** for the component. See ***Names of Components and Variables*** for naming conventions.
3. In the **Source Project & Design** panel, enter the project **File**. Click to find and select the Maxwell file (**.aedt**) that you want to use. The selected project loads into its application.
4. Based on the **Source** file and the chosen **Link Type** (Maxwell Eddy Current Link – S Parameters / Maxwell Eddy Current Link – RLGC Parameters / Maxwell A-Phi Eddy Current Link – RLGC Parameters (Beta)), Twin Builder communicates with Maxwell, retrieves the applicable **Design**, **Solution**, and **Matrix** information, and populates the corresponding menus with the information.

   The **Information** tab updates to show the **Number of Conservative Pins** and various other **Quantities** present in the selected design.

   Reduced matrices are prefixed with a double colon and the matrix they belong to. For example, **Matrix1::ReducedMatrix1**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

You can create three types of Eddy Current links:

- **Uses S Parameters (Default)** – The Maxwell design produces S Parameters data, which is used to create the equivalent model.
- **Uses RLGC Parameters** – The Maxwell design produces RLGC Parameters data, which is used to create the equivalent model.
- **Uses A-Phi RLGC Parameters (Beta)** – The Maxwell design produces RLGC Parameters data from the Maxwell A-Phi design, which is used to create the equivalent model. In order to see this link type, enable **Maxwell Eddy Current A-Phi Solution** in **Beta Options**.

5. Select the **Design** to use from the drop-down list.
6. Select the **Solution** and **Matrix** to use from the selected **Design**.
7. For 2D designs, a **Depth** field displays. Enter a positive number for the design length in this field. The default unit of length is the meter. You can change this to any valid unit of length (“ft” or “cm”, for example). If you omit the unit of length, the component will use the default unit of length set in the **Default Units** tab of the **General Options** dialog box.

Note:

Depth is controlled based on the **COMPONENT_DEPTH** design quantity of the component instance, so changes will propagate only by changing this property either on the **Quantities** tab in the component's **Properties** dialog box or in the **Depth** field in the coupling component dialog box. However, in designs created or saved in Twin Builder, the quantity will be shown as **DEPTH_FROM_ SIMPLORER**.

8. On the **Options** tab, choose to show either the **Pin Name** or **Pin Description** on the new component. Pin names display by default.

Note:

Pin description properties are always added to the component regardless of whether they are selected to be shown with pins.

- If you chose **Pin Description** when adding the component, you can subsequently enable individual pin names. Open the component symbol in the symbol editor,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

delete a pin's description, double-click the pin and enable **ShowName** in the pin properties dialog box.

- If you chose **Pin Name** when adding the component, you can subsequently hide individual pin names. Open the component symbol in the symbol editor, double-click a pin and disable **ShowName** in the pin properties dialog box. You can then add descriptive text to the pin using the editor's **Draw** > **Text** tool.

9. You can select the following additional options on the **Options** tab:
   - **Save project after use** – Maxwell saves any solutions data after the Twin Builder simulation, overwriting any previously saved data.
   - **Unload project after use** – Maxwell closes the project and shuts down after the Twin Builder simulation is complete. This will release the Maxwell license.
   - **Create static link** – Twin Builder loads the data currently available from the project and maintains it internally. Additional calls to Maxwell will not be performed unless the properties of the component are changed. This option can result in faster run times by minimizing the number of calls to Maxwell for data sharing.
   - **Clear Static Cache** – Twin Builder removes the cached data and acquired new data from Maxwell.
   - **Enforce Passivity** – Choose this option if you want a state space model that is better for a stable transient simulation. **Error Tolerance** sets the error tolerance for S-Matrix fitting. **Maximum Order** value specifies the highest order state space system that can be chosen while fitting to represent the system behavior; a lower order may lead to less accuracy but may result in faster simulation: the default value is 10000. **Z(ref)** sets the value of the Z(ref) in ohms sent from the Twin Builder side for S-parameter creation.
10. When finished, click **OK** to accept the values and close the dialog box.
11. Place the component on the schematic sheet.
12. After placing the component on a sheet, its **Properties** dialog box may be opened for configuration of the component.

    The dynamic coupling component configuration is saved with the schematic sheet. Twin Builder must establish a link to the associated application each time a simulation is run.

Maxwell Excitations Component

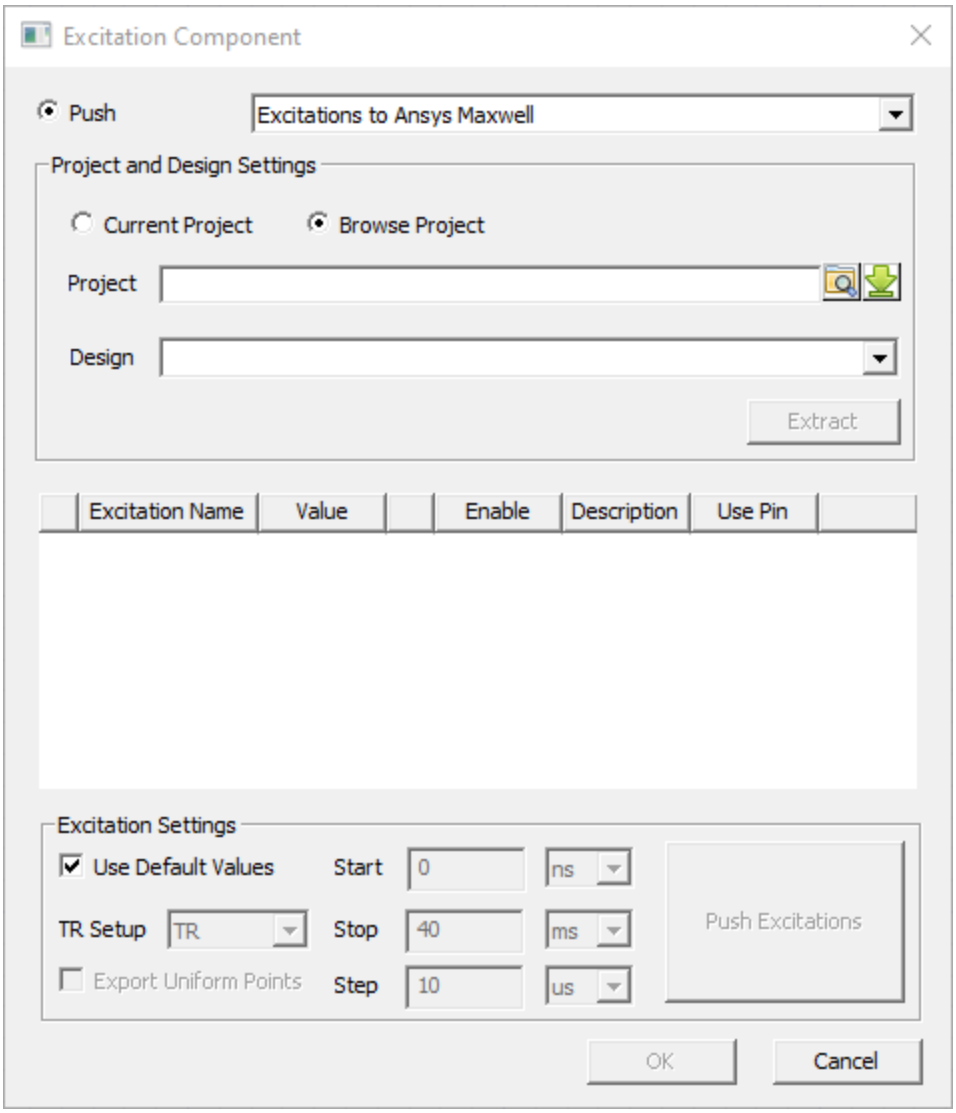
Use the excitations component to assign output quantities in a Twin Builder design to a windings in a Maxwell design. After the simulation completes, you can send the quantity values assigned to these windings to the Maxwell design. See the topics in this section for more information.

Adding an Excitations Component

Follow this procedure to add an excitations component.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **TwinBuilder** > **Add Component** > **Add Excitation Component** or select **Schematic** > **Import** on the ribbon. The **Excitation Component** dialog box appears.



2. In the **Project and Design Settings** section, select **Browse Project**, then browse to your Maxwell project and design.
3. Click **Extract** to populate the grid. The grid contains these columns:
    - **Name** – Displays the name of the winding in Maxwell.
    - **Value** – Enter the values to assign to the windings.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Calculator** – Click the button inside this column to display the quantities inside the Twin Builder design in a separate dialog box.
- **Enable** – Select the check box in this column to send the data for that winding.
- **Description** – Displays the type assigned to the winding in Maxwell: **Voltage** or **Current**.
- **Use Pin** – The design contains pins that represent windings. Select the check box in this column to use one of those pins to make connections on the schematic. If you select the box, the value in the **Value** column is zeroed out, since the expectation is that the value is provided through schematic connections. All pins are non-conservative inputs.

4. Use the controls in the **Excitation Settings** section to further refine your excitation component.
   - **TR Setup** – Choose your TR Setup.
   - **Start and Stop** – Displays the default start and stop time. Edit the times in these fields to choose a custom window of simulation data.
   - **Use Default Values** – Select this check box to populate the default values for the **Start** and **Stop** times for the chosen TR setup.
   - **Export Uniform Points** – Select this check box to have Twin Builder perform linear interpolation to uniformly space out time and data points. The interpolation is based on the step size provided.
5. Click **OK** to create your excitations component.

Sending Data to Maxwell

After the creating an excitations component as described in ***Adding an Excitations Component***, follow this procedure to send your data to Maxwell.

1. Simulate the design.
2. Double-click the component and select **Push Excitations**. The **Excitation Component** dialog box appears. You can edit the settings you entered in ***Adding an Excitations Component*** before sending the data.
3. Click **OK** to send the data to Maxwell. You can also right-click the component and select **Push Excitations**.
4. After you send the excitations, you can view them in the design datasets section in the Maxwell design.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

RMxprt Dynamic Coupling Component

RMxprt is a software package, installed with Maxwell, used to design, analyze, and optimize rotating machines. It combines rapid analytical solutions with optional finite element analysis. Use the RMxprt Link to utilize a motor or generator model for system simulation in Twin Builder, and to use both programs interactively.

Program Requirements for RMxprt Dynamic

- Twin Builder current version.
- Maxwell current version with RMxprt.

> **Note:**
>
> For two-way interactive use, RMxprt must be installed and licensed on the same computer as Twin Builder. However, RMxprt equivalent circuit models may be used in Twin Builder without an RMxprt license.

Overview of RMxprt Dynamic Coupling

To prepare and use an RMxprt Dynamic coupling component:

1. Open a project or create a new project in RMxprt, corresponding to the desired motor type.

   > **Note:**
   >
   > Avoid using illegal characters such as the “space” character in design names. Illegal characters are replaced with an underscore character. See *Names of Components and Variables* for additional information.

2. Enter basic input data, and analyze the design in RMxprt.
3. Create the motor equivalent circuit model in RMxprt.
4. In Twin Builder, add the RMxprt coupling component to the sheet.
5. Create the rest of the simulation design on the sheet.
6. Start the simulations in Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7. Switch back to RMxprt if you need to update the motor design.

> **Note:**
>
> RMxprt motor models are generally used for transient analysis, but they also support AC and DC analysis.

Mechanical Pins for RMxprt Dynamic Coupling Components

When working with RMxprt dynamic coupling components:

- All motor models exported from RMxprt have two mechanical pins that must be connected to a mechanical subsystem having a ROTATIONAL_V nature.
- In RMxprt 4, the port labeled *Torque* is an electrical current source that is grounded internally. The voltage on this pin is sensed as the motor speed. A *Domain to Domain* component must be used to interface with the mechanical subsystem, as shown in the figure. The pin labeled *Position* is for output only, and is an electrical voltage numerically equal to the rotor angle.
- In RMxprt 5 and later versions, the mechanical pins are of ROTATIONAL_V nature, and the *Domain to Domain* component need not be used. The two pins are ROT1 and ROT2, with torque as the through variable and speed as the across variable. The angle is output as a real number.

Electrical Pins for RMxprt Dynamic Coupling Components

- Motor models exported from RMxprt 4 have the neutral pin grounded internally. The neutral pin is not available for external connections, and in this example, the simulation runs even though the sources are not grounded.
- In RMxprt 5 and later versions, the neutral pin is available for external grounding.

Adding an RMxprt Dynamic Coupling Component in Twin Builder

Follow this procedure to add an RMxprt Dynamic coupling component to a Twin Builder design.

1. On the **Twin Builder** main menu, select **Subcircuit** > **Add RMxprt Dynamic Component**. The **RMxprt Dynamic Coupling** dialog box appears.
2. Specify a **Name** for the component. See ***Names of Components and Variables*** for naming conventions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. In the **Source Project & Design** panel, enter the project **File**. Click to find and select the RMxprt file (**.aedt**) that you want to use.

   The selected project loads into its application.

4. Based on the **Source** file and the **Link Type** (RMxprt Link), Twin Builder communicates with RMxprt, retrieves the applicable **Design** and **Solution** information, and populates the corresponding menus with the information.

   The **Information** tab updates to show the **Number of Conservative Pins** and various other **Quantities** present in the selected design.

5. After you select the **Source** file, select the design you want to use in the coupled design from the **Design** drop-down list.
6. Select the desired solution from the **Solution** drop-down list.
7. Optionally, on the **Options** tab:
   a. Select **Save project after use** to have the linked project file saved by its application after co-simulation is completed.
   b. Select **Unload project after use** to have the linked project file closed by its application after co-simulation is completed.
   c. Select **Create static link** to create a static link and enable the **Clear Static Cache** button.
      - If **Create static link** is not selected, Twin Builder obtains sml/netlist information from RMxprt every time a simulation analysis is run, even if no parameter has changed. This is the default behavior.
      - If **Create static link** is selected, Twin Builder obtains sml/netlist information from RMxprt only if a parameter value has changed since the previous simulation run. Click **Clear Static Cache** to delete the sml/netlist information currently stored in the Twin Builder model. The next simulation run will then obtain fresh sml/netlist information from the other product application and store it in Twin Builder.

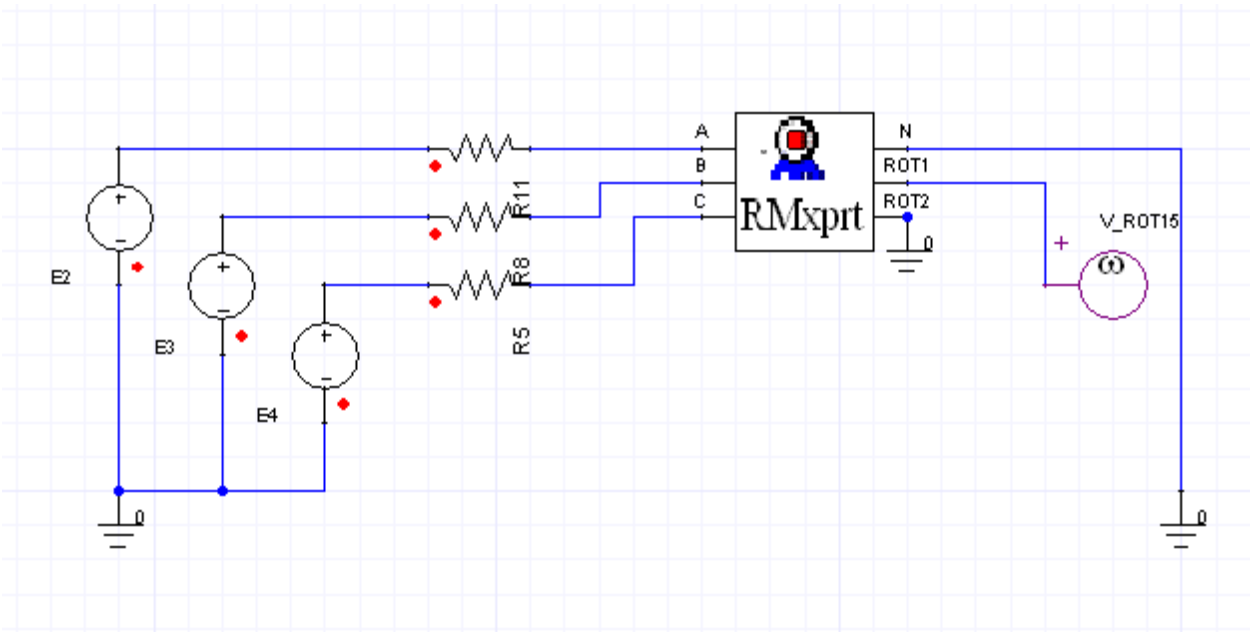
   By default, pin names display. The option to **Show Description** is disabled.

8. When finished click **OK** to accept the values and close the dialog box.
9. Place the component on the schematic sheet.
10. After placing the component on a sheet, its **Properties** dialog box may be opened for configuration of the component.

    The dynamic coupling component configuration is saved with the schematic sheet. Twin Builder must establish a link to the associated application each time a simulation runs.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Example of an RMxprt Dynamic Model in Twin Builder



Q3D Dynamic Coupling Components

This section describes Q3D equivalent circuit and state space components.

Q3D Equivalent Circuit Component

Ansys Electromagnetics Q3D Extractor SML models derive from electromagnetic field solutions for the device. Typical applications include signal integrity, RF design, and wireless system design.

This section contains information on:

- Program Requirement for Q3D Equivalent Circuit Coupling
- SML Equivalent Circuit Coupling Component Interface Concept
- Overview of SML Equivalent Circuit Component Coupling
- Adding an SML Equivalent Circuit Component

Program Requirements for Q3D Equivalent Circuit Component Coupling

- Twin Builder current version.
- Ansys Electromagnetics Q3D Extractor current version.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Q3D Dynamic Equivalent Circuit Coupling Component Interface Concept

The Q3D Dynamic SML Equivalent Circuit coupling component provides dynamic coupling between Twin Builder and the electromagnetic field simulator. The component is very efficient, allowing the use of multiple instances, as well as coupling with other models and simulators. During simulation, the SML model can limit the Twin Builder time step for accurate results.

> **Note:**
>
> The Q3D Equivalent Circuit component creates an equivalent circuit at a single frequency while the Q3D State Space component creates a state-space model that is valid over a range of frequencies.

First create a 2D or 3D model, with material and boundary assignments, in Q3D. Then perform a frequency sweep solution on this model.

Overview of Q3D Dynamic Equivalent Circuit Component Coupling

To prepare and use a Q3D Dynamic Equivalent Circuit coupling component, do the following:

1. Create a 2D or 3D model of the device in Q3D.

   > **Note:**
   >
   > Avoid using illegal characters such as the “space” character in design names. Illegal characters are replaced with an underscore character. See *Names of Components and Variables* for additional information.

2. Perform a solution setup with a single solution frequency.
3. In Twin Builder, add the Q3D Dynamic Equivalent Circuit component to the sheet.
4. Create the rest of the simulation design on the sheet.
5. Start the simulations in Twin Builder.

   > **Note:**
   >
   > To make changes in the Q3D SML model, you must edit the 3D model, recalculate the field solution, and re-export the model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding a Q3D Equivalent Circuit Coupling Component

Follow this procedure to add a Q3D SML Dynamic Equivalent Circuit component to a Twin Builder design.

1. On the **Twin Builder** main menu, select **Subcircuit** > **Q3D Dynamic Component** > **Add Equivalent Circuit**. The **Q3D SML Dynamic Coupling** dialog box appears.
2. Specify a **Name** for the component. See ***Names of Components and Variables*** for naming conventions.
3. In the **Project & Design** panel, enter the project **File**. Click to find and select the Q3D file (***.q3dx** or ***.aedt**) that you want to use.

   The selected project loads into its application.
4. Based on the **Source** file and the **Link Type** (Q3D Equivalent Circuit Link), Twin Builder communicates with Q3D, retrieves the applicable **Design**, **Solution**, and **Reduced Matrix** information, and populates the corresponding menus with the information.

   The dialog's **Information** tab updates to show the **Number of Conservative Pins** and various other **Quantities** present in the selected design. You can choose to add pins for listed **Quantities**.
5. After you select the **Source** file, select the design you want to use in the coupled design from the list in the **Design** drop-down list.
6. Select the desired solution from the **Solution** drop-down list.
7. Select the **Reduced Matrix** to use with the component during simulation.
8. For 2D designs, a **Depth** field displays. Enter a positive number for the design length in this field. The default unit of length is the meter. You can change this to any valid unit of length ("ft" or "cm," for example). If you omit the unit of length, the component uses the default unit of length set in the **Default Units** tab of the **General Options** dialog box.

   > **Note:**
   >
   > Depth is controlled based on the **COMPONENT_DEPTH** design quantity of the component instance, so changes will propagate only by changing this property either on the **Quantities** tab of the component's **Properties** dialog box or in the **Depth** field in the coupling component dialog box. However, in designs created or saved in Twin Builder, the quantity is shown as **DEPTH_FROM_SIMPLORER**.
9. Optionally, on the **Options** tab:
   a. Select **Save project after use** to have the linked project file saved by its application after co-simulation is completed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

b. Select **Unload project after use** to have the linked project file closed by its application after co-simulation is completed.
c. Select **Create static link** to create a static link and enable **Clear Static Cache**.
   - If **Create static link** is not selected, Twin Builder obtains sml/netlist information from Q3D every time a simulation analysis is run, even if no parameter has changed. This is the default behavior.
   - If **Create static link** is selected, Twin Builder obtains sml/netlist information from Q3D only if a parameter value has changed since the previous simulation run. Click **Clear Static Cache** to delete the sml/netlist information currently stored in Twin Builder. The next simulation run will then obtain fresh sml/netlist information from Q3D and store it in Twin Builder.

10. Choose to show either the **Pin Name** or **Pin Description** on the new component. By default, pin names are shown.

> **Note:**
>
> Pin description properties are always added to the component regardless of whether they are selected to be shown with pins.

    - If you chose **Pin Description** when adding the component, you can subsequently enable individual pin names. Open the component symbol in the symbol editor, delete a pin's description, double-click the pin and enable **ShowName** in the pin properties dialog box.
    - If you chose **Pin Name** when adding the component, you can subsequently hide individual pin names. Open the component symbol in the symbol editor, double-click a pin and disable **ShowName** in the pin properties dialog box. You can then add descriptive text to the pin using the editor's **Draw** > **Text** tool.

11. When finished click **OK** to accept the values and close the dialog box.
12. Place the component on the schematic sheet.
13. After placing the component on a sheet, its **Properties** dialog box may be opened for configuration of the component.

    The dynamic coupling component configuration is saved with the schematic sheet. Twin Builder must establish a link to the associated application each time you run a simulation.

Q3D State Space Components

When working with Q3D state space components, you can insert an S-Parameter model or an RLCG model to create a frequency sweep, linking Q3D and Twin Builder with state space.

This section contains information on:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Program Requirement for Q3D State Space Component Coupling
- Overview of Q3D State Space Coupling
- Adding an Q3D State Space Component

> **Note:**
>
> For more information, see the Q3D help.

Program Requirements for Q3D State Space Component Coupling

- Twin Builder current version.
- Ansys Electromagnetics Q3D Extractor current version.

Overview of Q3D State Space Coupling

> **Note:**
>
> The Q3D State Space component creates a state-space model that is valid over a range of frequencies, while the Q3D Equivalent Circuit component creates an equivalent circuit at a single frequency.

To prepare and use a Q3D State Space component, do the following:

1. Create a 2D or 3D model of the device in Q3D.

   > **Note:**
   >
   > Avoid using illegal characters such as the “space” character in design names. Illegal characters are replaced with an underscore character. See *Names of Components and Variables* for additional information.

2. Perform a frequency sweep electromagnetic field solution.
3. In Twin Builder, add the Q3D state space coupling component to the sheet.
4. Create the rest of the simulation design on the sheet.
5. Start the simulations in Twin Builder.

   > **Note:**
   >
   > To make changes in the coupling component, you must edit the Q3D design, recalculate the field solution, and re-export the model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding a Q3D State Space Coupling Component

Follow this procedure to add a Q3D State Space coupling component to a Twin Builder design.

1. On the **Twin Builder** main menu, select **Subcircuit** > **Q3D Dynamic Component** > **Add State Space**. The **Q3D State Space Dynamic Coupling** dialog box appears.
2. On the **Link Description** tab, specify a **Name** for the component. See ***Names of Components and Variables*** for naming conventions.
3. In the **Source Project & Design** panel, enter the project **File**. Click to find and select Q3D file (***.q3dx** or ***.aedt**) . The selected project loads into its application.
4. Based on the **Source** file and the chosen **Link Type** (Q3D State Space Link – S Parameters / Q3D State Space Link – RLGC Parameters), Twin Builder communicates with Q3D, retrieves the applicable **Design**, **Solution**, and **Reduced Matrix** information, and populates the corresponding menus with the information.

   The **Information** tab updates to show the **Number of Conservative Pins** and various other **Quantities** present in the selected design.

> **Note:**
>
> You can create two types of state-space link for both 2D and 3D designs.
>
> - **Uses S Parameters (default)** – A Q3D design produces S Parameters data, which is used to create the equivalent model.
> - **Uses RLGC Parameters** – A Q3D design produces RLGC Parameters data, which is used to create the equivalent model.
>   The RLGC dynamic link is based on a simple lumped circuit model. The RL elements are between the source and the sink terminals; the GC elements are between the sink terminals and ground.

5. After you select the **Source** file, select the design you want to use in the coupled design from the **Design** drop-down list.
6. Select the **Solution** and **Reduced Matrix** to use from the selected **Design**.
7. For 2D designs, a **Depth** field displays. Enter a positive number for the design length in this field. The default unit of length is the meter. You can change this to any valid unit of length ("ft" or "cm," for example). If you omit the unit of length, the component uses the default unit of length set in the **Default Units** tab of the **General Options** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Depth is controlled based on the **COMPONENT_DEPTH** design quantity of the component instance, so changes will propagate only by changing this property either on the **Quantities** tab in the component's **Properties** dialog box or in the **Depth** field in the coupling component dialog box. However, in designs created or saved in Twin Builder, the quantity will be shown as **DEPTH_FROM_ SIMPLORER**.

8. In the **Options** tab, choose to show either the **Pin Name** or **Pin Description** on the new component. Pin names display by default.

> **Note:**
>
> Pin description properties are always added to the component regardless of whether they are selected to be shown with pins.

   - If you chose **Pin Description** when adding the component, you can subsequently enable individual pin names. Open the component symbol in the symbol editor, delete a pin's description, and double-click the pin and enable **ShowName** in the pin properties dialog box.
   - If you chose **Pin Name** when adding the component, you can subsequently hide individual pin names. Open the component symbol in the symbol editor, double-click a pin and disable **ShowName** in the pin properties dialog box. You can then add descriptive text to the pin using the editor's **Draw** > **Text** tool.

9. You can select the following additional options on the **Options** tab:
   - **Save project after use** – Q3D saves any solutions data after the Twin Builder simulation, overwriting any previously saved data.
   - **Unload project after use** – Q3D closes the project and shuts down after the Twin Builder simulation is complete. This will release the Q3D license.
   - **Create static link** – Twin Builder loads the data currently available from the project and maintains it internally. Additional calls to Q3D will not be performed unless the properties of the component change. This option can result in faster run times by minimizing the number of calls to Q3D for data sharing.
   - **Clear Static Cache** – Twin Builder removes the cached data and acquired new data from Q3D.
   - **Enforce Passivity** – Choose this option if you want a state space model that is better for a stable transient simulation. **Error Tolerance** sets the error tolerance for S-Matrix fitting. The **Maximum Order** value specifies the highest order state space system that you can choose while fitting to represent the system behavior; a lower

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

order may lead to less accuracy but faster simulation. **Z(ref)** sets the value of the Z (ref) in ohms sent from the Twin Builder side for S-parameter creation.

10. When finished click **OK** to accept the values and close the dialog box.
11. Place the component on the schematic sheet.
12. After placing the component on a sheet, its **Properties** dialog box may be opened for configuration of the component.

    The dynamic coupling model configuration is saved with the schematic sheet. Twin Builder must establish a link to the associated application each time a simulation is run.

PExprt Static Coupling Components

PExprt Static SML coupling components derive from analytical models of magnetic components such as inductors and transformers. With PExprt, designers can quickly calculate all possible combinations of core size, core material, wire gauge, turns, and gap length that satisfy user-supplied electrical specifications. The resulting “virtual” designs are further refined and optimized using embedded finite element calculations to predict quantities such as magnetizing and leakage inductance, interwinding capacitance, peak flux density, DC winding resistance, eddy current effects, core loss, and temperature rise. PExprt generates an equivalent-circuit model (SML) accounting for magnetic and thermal properties for use in Twin Builder for complete circuit optimization.

Program Requirements for PExprt Static Component Coupling

- Twin Builder current version.
- PExprt current version.

Overview of PExprt Static Component Coupling

To prepare and use a PExprt Static component:

1. Create a model of the device in PExprt.

   > **Note:**
   >
   > Avoid using illegal characters such as the “space” character in design names. Illegal characters are replaced with an underscore character. See ***Names of Components and Variables*** for additional information.

2. Select **Modeler** > **Generate Model** to open **PExprt Modeler(PEmag)**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. In the modeler, select either **Modeler** > **Analystical Modeler** > **Start 1D Model Generation** or **Modeler** > **FEA Based Modeler** > **Start 2D Model Generation** to generate the .**sml** model.
4. On the **Twin Builder** main menu, select **Subcircuit** > **Add PExprt Static Component** to add the PExprt Static Component to the sheet.
5. Create the rest of the simulation design on the sheet.
6. Start the simulations in Twin Builder.

> **Note:**
>
> To make changes in the PExprt SML model, you must edit the PExprt design and re-export the model.

Ansys Mechanical Components

Use Ansys Mechanical component coupling in concert with Twin Builder to import a structural model created in Ansys Workbench/APDL into a schematic for simulation and analysis. See the topics in this section for more information.

Program Requirements for Ansys Mechanical Component Coupling

- Twin Builder current version.
- Ansys current version.

Ansys Mechanical Component Interface Concept

The Twin Builder-Ansys Mechanical component interface is a static link that lets you add the reduced-order model of a structural system to a Twin Builder schematic. Using model order reduction techniques implemented in Ansys Mechanical, (for more information, see the Ansys Workbench Mechanical APDL documentation) you can create a reduced-order equivalent state-space model for Twin Builder.

- Models support Transient, AC, and DC analysis.
- You can specify an initial condition for a model that initializes the state variables (that is, initial positions). You must specify initial condition values as a vector of double values only.
- Based on the flags specified during the reduction procedure, the generated Twin Builder model (.**spm**) file may have conservative or non-conservative terminals.

Conservative terminals

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Conservative terminals use the force-displacement mechanical domain representation and may be rotational or translational in nature (based on the definition of the Degree of Freedom in Ansys Mechanical). For conservative terminals, the system implemented is of the form:

$$\frac{dx}{dt} = Ax + BF$$
$$s = Cx$$

where,

F is the vector of forces (or torques) at each terminal,

s is the vector of displacements at each terminal,

x is the vector of states.

Also in the conservative implementation, references are present in one of the following three forms:

- no reference
- common reference (only when all terminals have the same nature)
- individual references

Non-conservative terminals

In the non-conservative representation, input and output pins are separate and non-conservative in nature. The system implemented by the model is of the form:

$$\frac{dx}{dt} = Ax + B + u$$
$$y = Cx$$

where,

u is the vector of inputs,

y is the vector of outputs.

Overview of Ansys Mechanical Component Coupling Workflow

Ansys Mechanical component coupling lets Twin Builder import a structural model created in Ansys Workbench/APDL into a schematic for simulation and analysis. The workflow is as follows:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

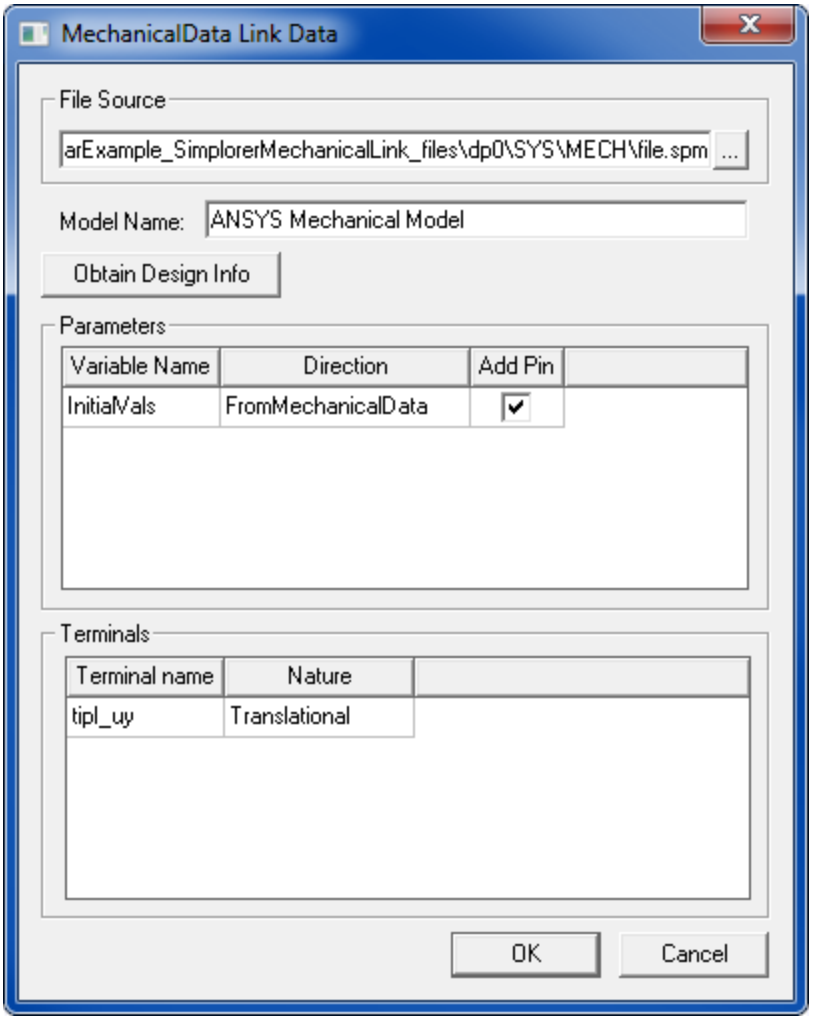
1. Create an Ansys Workbench/APLD structural system.
2. Perform a modal analysis in Ansys Workbench.
3. Export the reduced model to an SML file.
4. Import the SML file into Twin Builder and place the Ansys Mechanical component on the schematic.
5. Connect the Ansys Mechanical component pins to the other schematic components.
6. Run Twin Builder analysis.

Adding an Ansys Mechanical Component

Follow this procedure to create a new Ansys Mechanical component:

1. Follow the steps outlined in the Ansys Mechanical help to generate an **.spm** file in SML format for the Workbench/APDL static structural model with the desired degrees of freedom, type (conservative or non-conservative), and references (single or individual). An image file for the component is also generated. Run **spmwrite** to generate an **.spm** file and an image file.
2. Select **Twin Builder** > **Subcircuit** > **Add Mechanical Component**.
3. In the **MechanicalData Link Data** dialog box, locate and select the appropriate **.spm** file to create a new component in Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The **.spm** file is compiled and parameters and terminals information displays. Toggle pin display with the **Add Pin** check box column in the **Parameters** grid.

4. Optionally, change the **Model Name**.
5. Click **OK** to create a component, symbol (with image from the 3D model), and a model, and add them to the project.
6. Position the new component on the schematic and make the appropriate connections between it and existing schematic components.
7. Run Twin Builder Transient, AC, and/or DC analyses as desired.

Related Topics

Ansys Mechanical Link Example

File Link Dialog Box

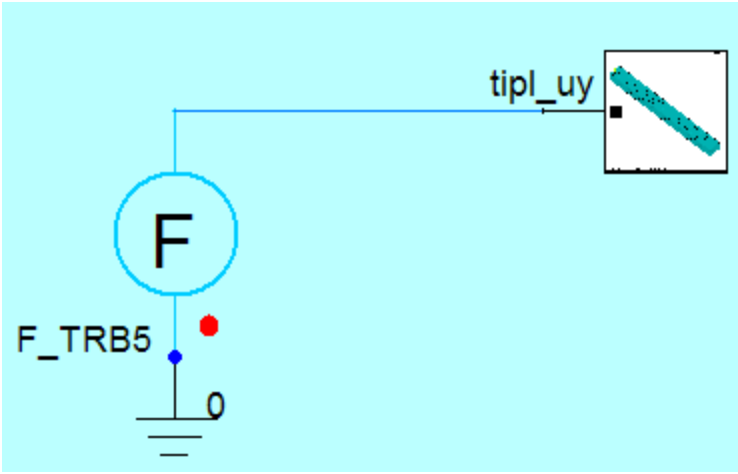
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ansys Mechanical Link Example

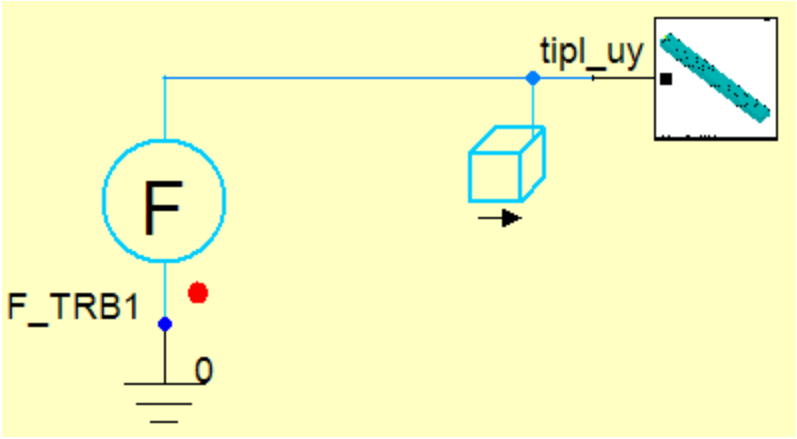
An example Twin Builder project for an Ansys Mechanical Link is provided in the *<installation_directory>***\Examples\Twin Builder\Components\Interfaces\Mechanical** folder.

The example shows three separate reduced order models generated by Ansys Mechanical for Twin Builder:

- Clamped Free Beam



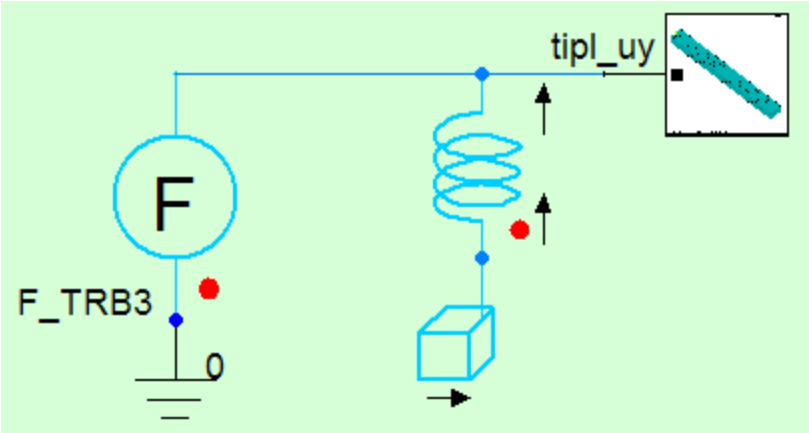
- Clamped Free Beam with a 20 kg mass at the free end



- Clamped Free Beam with a spring (stiffness of 10000 Newton per meter) and 20kg mass

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

at the free end



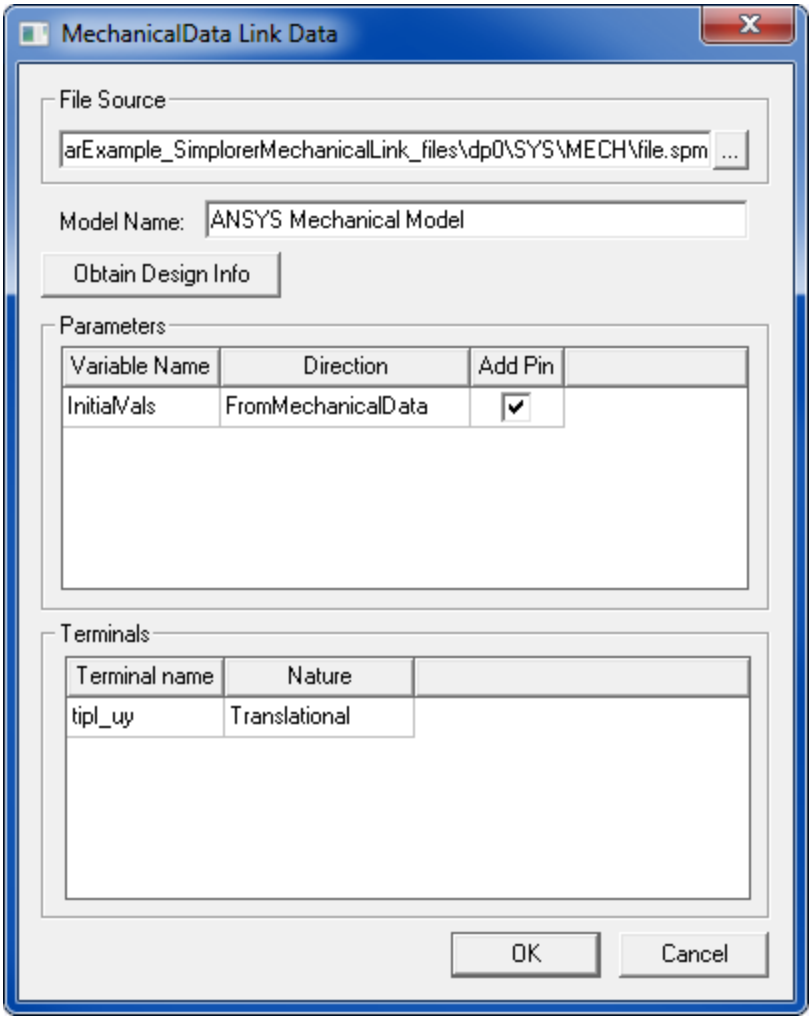
The Ansys Mechanical source project files are also included in the **BarExample_ SimplorerMechanicalLink.zip** file.

The Twin Builder project was created by first bringing the three models into Twin Builder. Three separate instances of the Clamped Free Beam were created and one instance each of the other two models were created on the schematic. A 20kg mass was applied to one instance of the Clamped Free Beam and a 10000 N/m spring and a 20 kg mass were applied to the third instance of the Clamped Free Beam. The results for TR and AC simulation are compared against the corresponding reduced order models.

File Link Dialog Box

Use the file link dialog box is used to link data files for components generated in external applications such as the Ansys Mechanical **.spm** file shown in the following example. The dialog box name shown in its title bar varies to correspond with the coupling source type.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Click ... to open a file selection window for locating and selecting the desired **File Source**. You can assign a unique **Model Name** to the model and component being created. Click **Obtain Design Info** to retrieve design information for the chosen source file. The **Parameters** and **Terminals** panels display information about the parameters and terminals for the model being created. Select **Add Pin** to add a pin for the associated parameter to the new component.

Related Topics

Adding an Ansys Mechanical Component Subcircuit

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Icepak Components

Use the Icepak component to perform thermal analysis using linear superposition in Twin Builder. See the topics in this section for more information.

Program Requirements for Ansys Icepak Component Coupling

- Twin Builder current version.
- Ansys current version.

Ansys Icepak Component Interface Concept

Thermal Analysis using a Foster Network

In this method, the thermal problem is treated like a system. A system is an entity that processes a set of inputs and yields another set of outputs. For the thermal problem at hand, the inputs are power (or heat source), and the outputs are the temperature at user-specified locations. When a system is Linear and Time Invariant (LTI), the system has the following features:

- The system is completely characterized either by its impulse response or step response.
- If two LTI systems have the *same* impulse or step response, the two systems behave identically in that the outputs of the two systems are the same provided that the inputs to the two systems are the same.

This feature allows a Foster network, which is an LTI system, to represent the thermal system, which also can be an LTI system under certain conditions.

Single input and single output system

This section describes the simplest system: one with a single input and a single output. The thermal problem is such that the power is from one component, and the output is the temperature at one user-specified location. A Foster network problem for this kind of system is shown in Figure 1, in which the input is the current to the network, and the output is the voltage at the same location.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

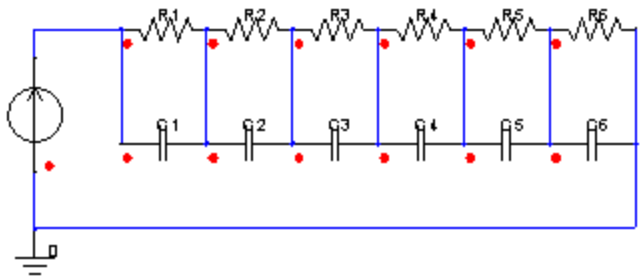


Figure 13-1 Foster Network

As mentioned above, both the thermal and Foster systems are LTI under certain conditions; they will behave identically if they have the same impulse or step response. Since the step response is easier to work with, the goal is to match the step response of the two systems. The step response of a typical thermal system is shown in Figures 2 and 3 for self-heating and cross-heating. Note that the self-heating curve has a positive slope at the start of the curve, while the cross-heating curve typically has close to zero slope at the start of the curve.

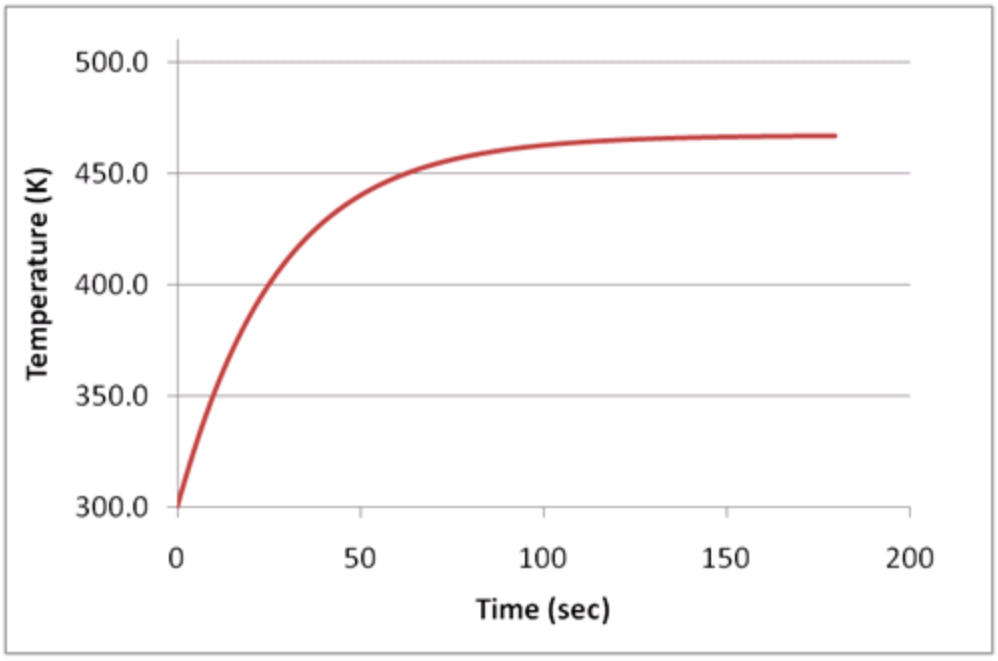


Figure 13-2 Typical step response for self-heating

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

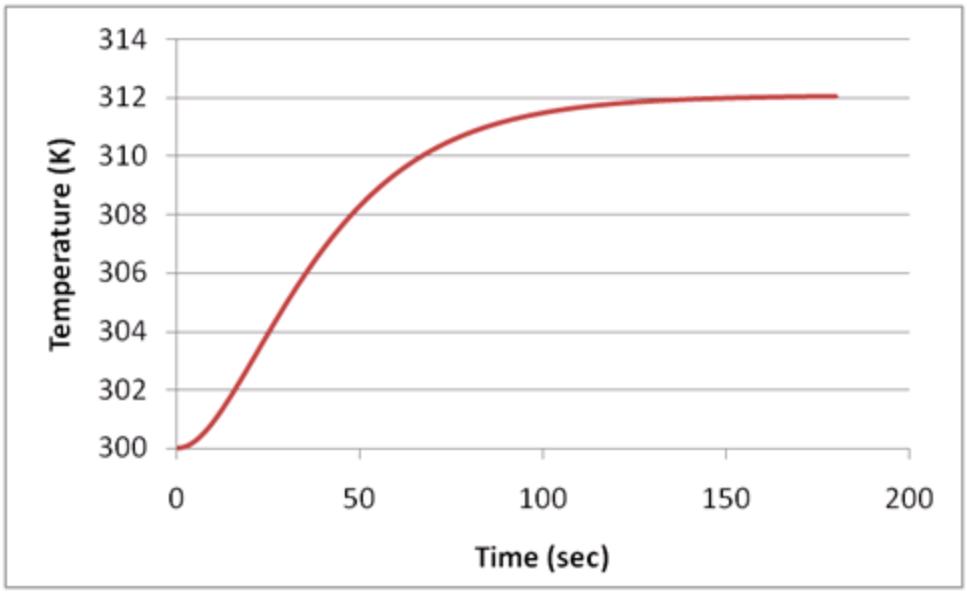


Figure 13-3 Typical step response for cross-heating

The step response of a Foster network is shown in the following equation:

$$V = I \cdot \sum_{i=1}^{m} R_i \cdot (1 - e^{-t/\tau_i})$$

where *m* is the number of rungs or RC pairs, and parameters R_i and τ_i must be extracted through curve fitting the transient step response of the thermal system.

A typical step response from a Foster network will not give close-to-zero slope as required by the cross-heating curves, and thus we cannot match the step response of cross-heating from the thermal system. However, if one allows for negative *R*, the resistance, the curve fit works well. Even though negative *R* poses no difficulty mathematically, it causes problems in circuit simulators. You can overcome this difficulty by using positive *R* but subtracting the *V* contribution from that part of the RC circuit. Figure 4 shows an example. In this example, the voltage (or output) equals VM1 minus VM2 rather than VM1 plus VM2. So, the Foster network itself has no negative *R*s, but rather it has a part with negative voltage contribution.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

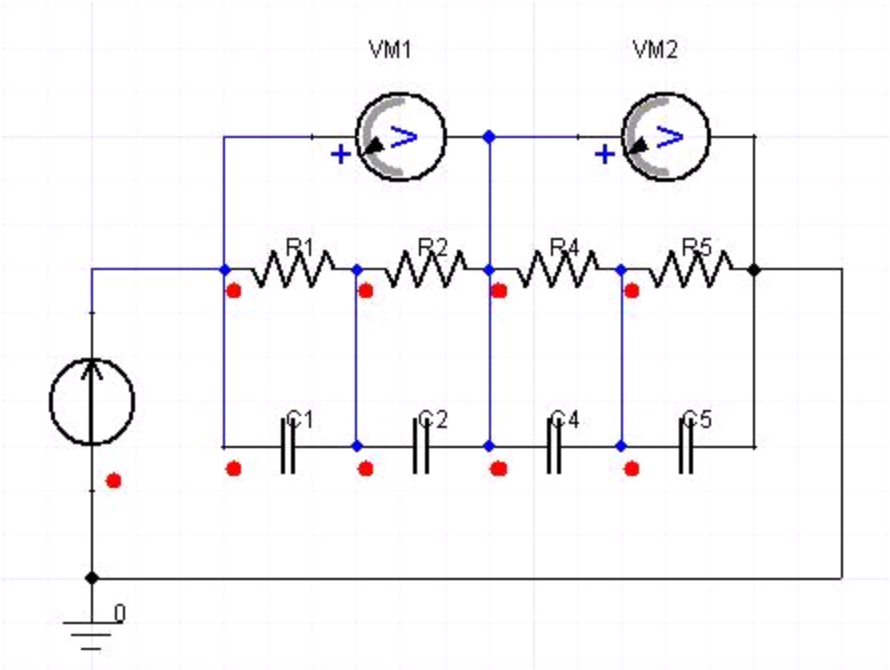


Figure 13-4 A Foster Network accounting for negative *R*

As long as the step response from the Foster network matches that from the thermal system, we are guaranteed that the subsequent response from the thermal system under any transient input can be represented by the Foster network, due to the characteristics of LTI systems mentioned in the introduction. Mathematically, if the step response of a system is known, then the impulse response, designated typically by *h(t),* is known to be the derivative of the step response; and *h (t)* entirely characterizes the system through the following equation:

$$y(t) = h(t) * x(t)$$

where *x(t)* is the input, *y(t)* is the output, *h(t)* is the impulse response, and * designates convolution. Even though we could use the convolution equation to calculate the thermal response, we choose to use the Foster network to represent the thermal system due to its simplicity to implement in circuit simulators. In passing we note that if the step response is too complex for the Foster network to curve fit, one can always rely on the convolution method. Also, convolution notation makes it easier to express the solution for systems with many inputs and outputs as described below.

Systems with many inputs and many outputs

For systems with many inputs and many outputs, the superposition property is used to obtain the solution. Superposition means that the overall output at one location is the sum of the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

contribution from individual inputs. The convolution equations satisfy the superposition property. In terms of the impulse response h_{ij}, the solution for a system with m inputs and n outputs can be calculated using the matrix below:

$$\begin{bmatrix} y_1(t) \\ y_2(t) \\ \dots \\ y_n(t) \end{bmatrix} = \begin{bmatrix} h_{11}(t)\, h_{12}(t) \dots h_{1m}(t) \\ h_{21}(t)\, h_{22}(t) \dots h_{2m}(t) \\ \dots \quad \dots \quad \dots \quad \dots \\ h_{n1}(t)\, h_{n2}(t) \dots h_{nm}(t) \end{bmatrix} * \begin{bmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_m(t) \end{bmatrix}$$

where h_{ij} represents the impulse response of the jth input on the ith output, and again * designates convolution rather than product. As before, instead of using the convolution method, a Foster network represents the thermal system. But instead of curve fitting one step response, we need to curve fit $n \times m$ number of step responses in principle. However, many of the cross-heating responses have very small values, and thus their contribution can be neglected. In passing we note that the Laplace transform of the h_{ij} represents the impedance between the jth input and the ith output.

Also note that the impulse response matrix would, in general, not be symmetrical even though the system is linear. This is because in a convection heat transfer problem, a downstream object will strongly “feel” the heat from an upstream object but not vice versa. However, there are problems with symmetrical impulse responses, such as diffusions problems with temperature boundary condition.

Limitations and assumptions

As mentioned above, this method works provided that both systems are LTI. The linearity part of LTI means that the system must satisfy the superposition property. The time invariant part of LTI mainly concerned with the coefficients of various terms in the governing equations of the system, which are typically ordinary or partial differential equations. For the Foster network, if the resistance and capacitance are constants, the system is linear and time invariant. We will next discuss under what conditions thermal systems are LTI.

Thermal problems, governed by Navier-Stokes equations, are, in general, notoriously non-linear. So thermal systems processing power inputs and yielding temperature outputs are in general *not* linear. The linearity is mainly due to, but not limited to, the momentum equation. However, if density and properties are constant, the momentum equation is decoupled from energy equation. This decoupling makes the energy equation linear and thus the thermal systems linear. But as mentioned above, the thermal system needs to be time invariant as well. To make the thermal system time invariant, the coefficients of all terms in the energy equation need to be constant. Flow velocity, which serves as the coefficients for energy equation in the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

convection term, is the main concern. So, in order for the system to be time invariant, velocity cannot change as a function of time even though no restriction is applied on spatial variation.

The thermal system can also be linearized by choosing the properly operating point. However, this method has limited application for thermal systems as linearization requires that heating has a large DC component and relatively small variation superposed onto it. Most thermal problems do not satisfy this condition. However, if this condition *is* satisfied, the Foster network method is valid.

To summarize, the following conditions must be satisfied in order for the thermal system to be LTI and thus make the method valid:

- **Constant density** – This is a very good assumption for incompressible fluids like water. For compressible fluids like air, constant density implies low Mach number and adiabatic temperature boundary condition. Even though the low Mach number condition does not pose real restrictions on most of problems considered, the adiabatic temperature boundary certainly cannot be satisfied strictly for heat transfer is the main point of interest. However, testing shows that the method still gives very decent results even when the max temperature increase goes up to 140K.
- **Constant property** – This is not a major concern as the properties either do not change appreciably within the range of temperature, or the impact of the property is rather negligible. Density variation will likely show signs of a problem before properties variation.
- **Constant velocity** – The Foster network, characterized at one velocity, cannot be used for other velocities. A new characterization (curve fitting) is needed if velocity changes.
- **No radiation** – This restriction is due to the non-linear relationship between temperature and radiation heat flux.

Foster network and thermal network

Before concluding the discussion, it might be worthwhile to talk about the difference between the current Foster network method and a traditional thermal network. In the traditional thermal network, the thermal problem is represented by thermal resistors and thermal capacitors. To obtain the model parameters, no curve fitting of step (or impulse) response is typically performed; the parameters can be extracted from various methods, ranging from physical argument to testing. The thermal network method is not limited to linear systems, even though it is applied most successfully to such a system. Another important difference is that in a thermal network each node represents a physical location, while in the Foster network it is noted that only the input and output nodes have physical connections; the rest of the nodes in the Foster network have no physical significance. Finally, the Foster network has the advantage of being more accurate provided the conditions are satisfied because the results from the Foster network are as accurate as the method generating the step (or impulse) response.

In summary, the thermal network is more general with perhaps compromised accuracy; while the Foster network is more accurate but with more restrictions attached to it.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Overview of Ansys Icepak Component Coupling Workflow

The following outlines the Icepak component coupling workflow for performing thermal analysis using linear superposition in Twin Builder.

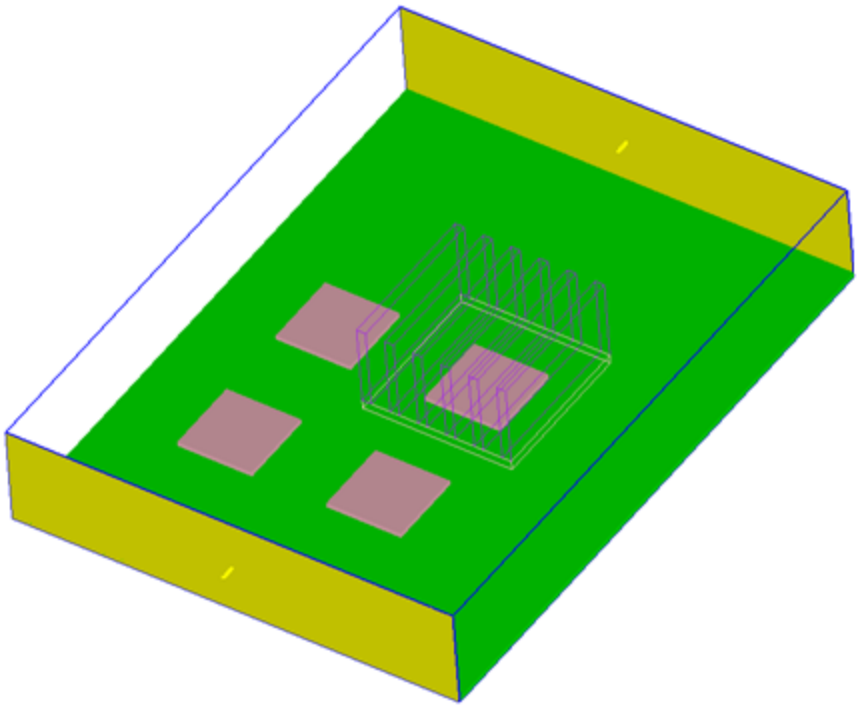
1. Specify the geometry of the multi-heat-source system in Ansys Icepak.
2. In Icepak, apply each heat source individually and measure temperature at nodes of interest (parametric analysis).
3. Run parametric analysis in Icepak.
4. Import the **.simpinfo** file into Twin Builder, process, and place Ansys Icepak component on the schematic.
5. Connect the Ansys Icepak component to the schematic components.
6. Run Twin Builder analysis.

Adding an Ansys Icepak Component Subcircuit

> **Note:**
>
> It is recommended to use the state-space method to generate the step response for Icepak component subcircuits.

The following procedure assumes a system (such as the one shown below) having four potential heat sources has been constructed in Icepak.

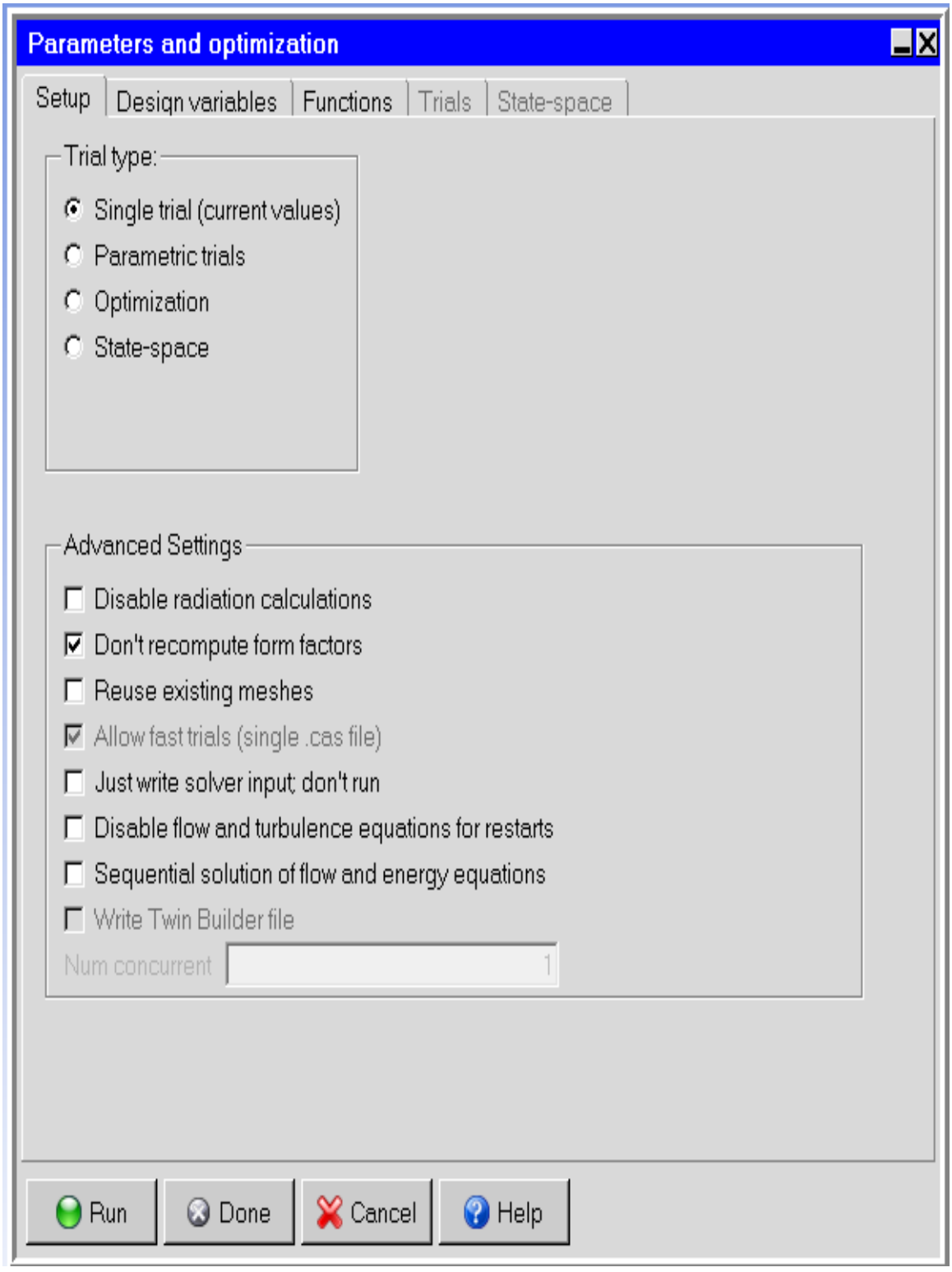
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



To create a new Ansys Icepak component subcircuit:

1. Follow the steps below to generate a **.simpinfo** file, Icepak thermal model, and associated output and symbol files.
   a. Launch Icepak, click **Unpack** in the **Welcome to Icepak** panel, select **Icepak-Twin Builder.tzr**, and click **Open**.
   b. In the **Location for the unpacked project file** selection dialog box, select a directory where you want to place the packed project file, enter **LED** in the **New project** text field, and click **Unpack**.
   c. Select **Solve** > **Run optimization**, and choose the options on the **Setup** tab as shown in the following example. Choose **State-space** in the **Trial type** section.

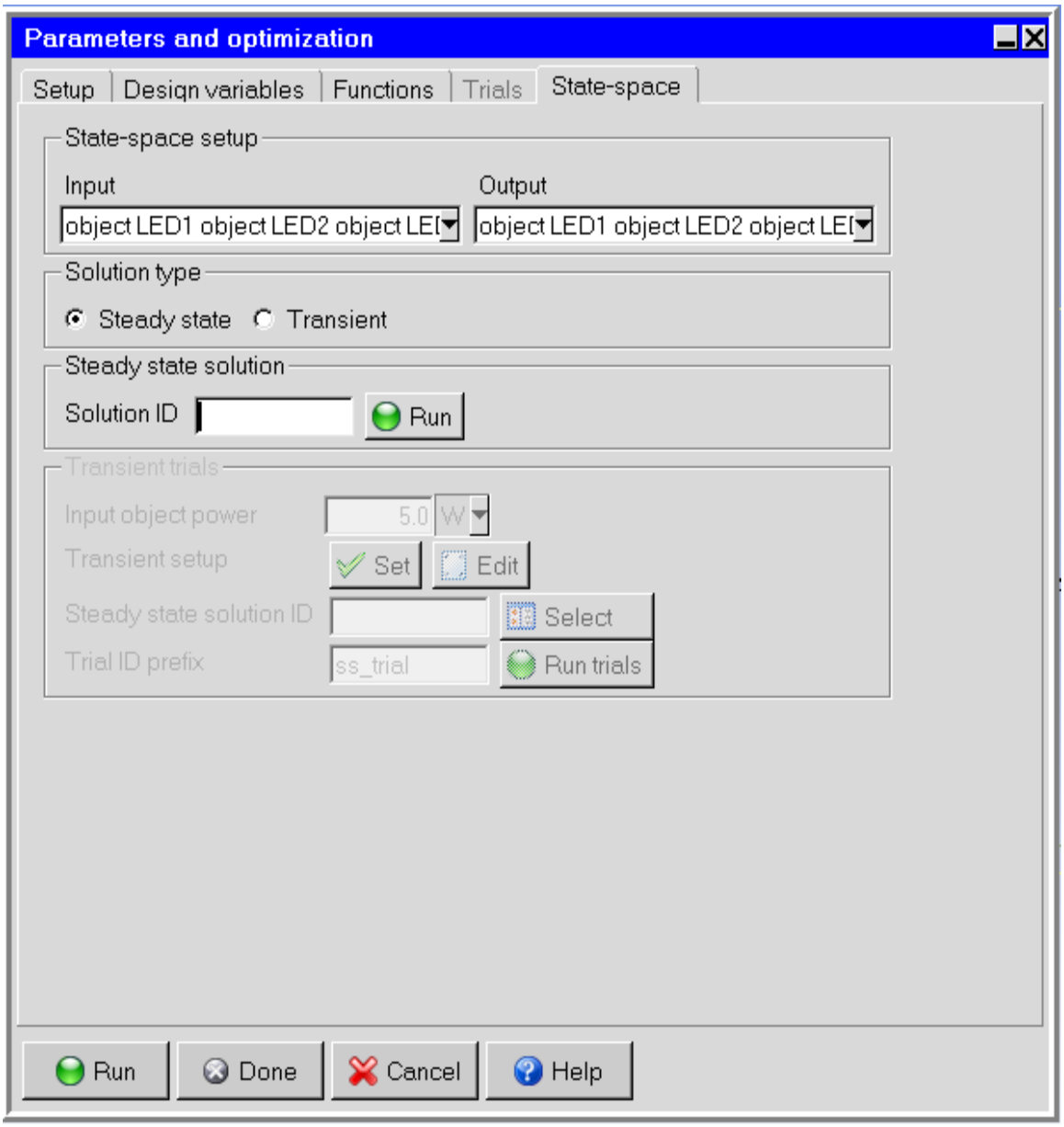
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



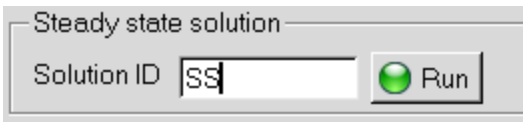
d. Select the **State-space** tab.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

e. Select **LED1, LED2, LED3, LED4** from the **Input** and **Output** drop-down lists and click **Accept**.



f. Enter **SS** in the **Solution ID** box and click **Run**.

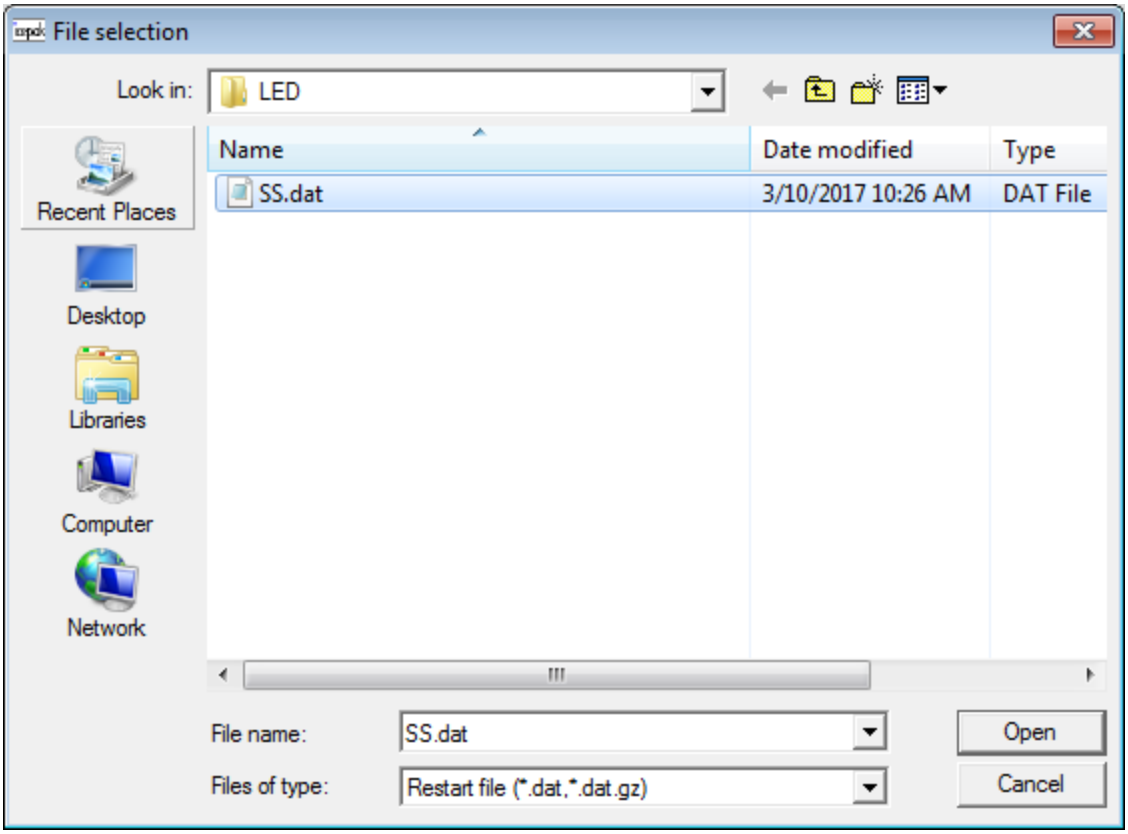


g. After the run is complete, select the **Transient** option and click **Set** to set the transient settings.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

h. Click **Select** to select the Steady state solution ID.

i. Select the **SS.dat** file and click **Open**.



j. Click **Run Trials**.

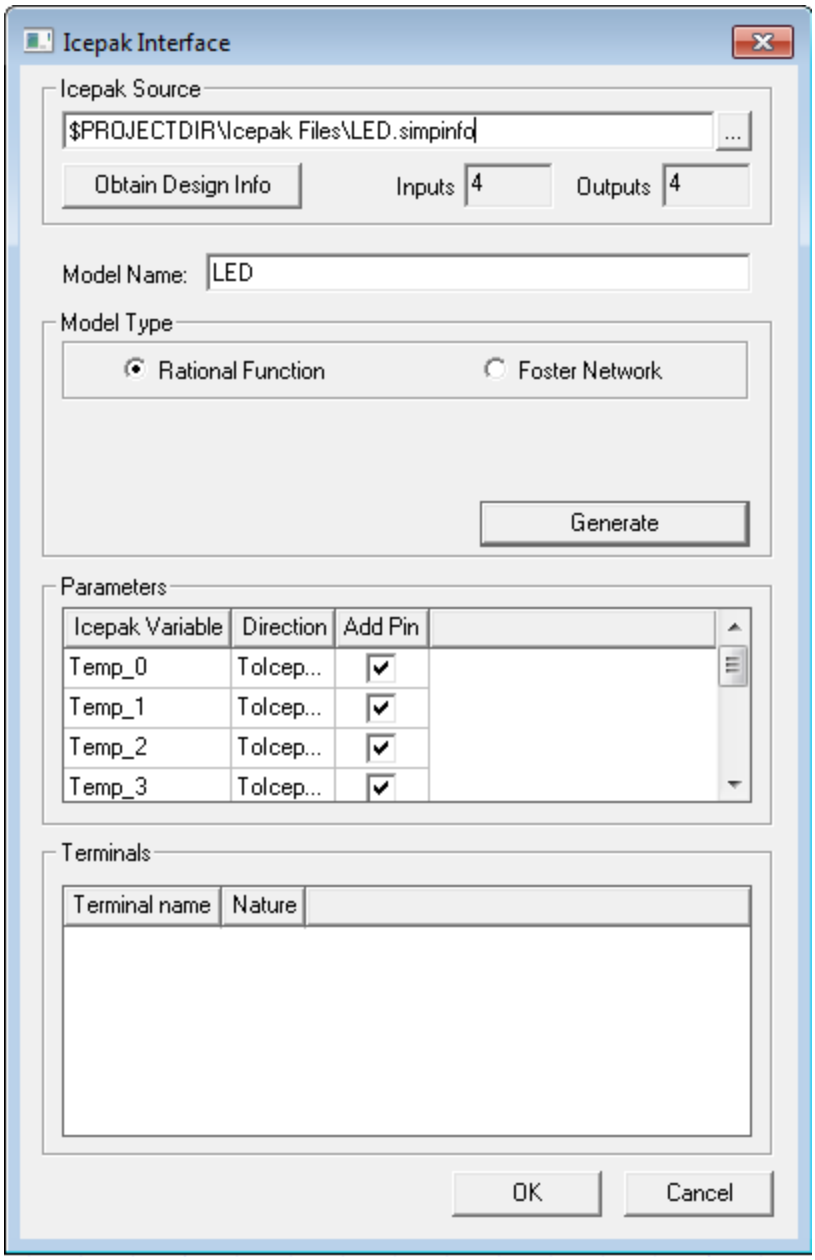
k. Click **Done**.

l. When the analysis is complete, a file with the extension **.simpinfo** is written in the same location as the results **.out** files. A **.gif** image file for the model symbol is also generated and written to this location.

2. On the Twin Builder main menu, select **Twin Builder** > **Subcircuit** > **Add Icepak Component**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The **Icepak Interface** dialog box appears. Browse and select **LED.simpinfo file**. Select other options as shown and press **OK** to add the component.



3. In the **Icepak Source** panel, click ... and select the appropriate **.simpinfo** file to create a new component in Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

- The Icepak output (**.out**) and symbol (**.gif**) files generated by Icepak must be present in the same directory as the **.simpinfo** file. If the Icepak output files are missing, an error dialog box displays, and the component will not be generated. If the symbol file is missing, a generic symbol is used for the component.
- In Icepak-Twin Builder coupling, the model names are used to name ports and terminals. Because model names in Icepak may include characters that are illegal in Twin Builder, name checking is implemented in Twin Builder so that illegal characters such as the **-** and **.** characters are replaced with an **_** character in the generated SML file. For example, the Icepak name "**block.1**" is changed to "**block_1**" in the generated SML file.

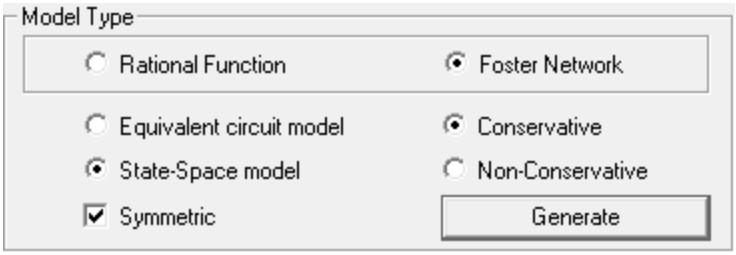
The panel also indicates the number of **Inputs** and trial **Outputs** present in the Icepak output file.

4. Optionally, change the **Model Name**.
5. Choose the desired **Model Type**. The options are **Rational Function** and **Foster Network**. For more information about Foster network, see Thermal Analysis using a Foster Network.

Note:

Rational Fitting provides greater accuracy than Foster network, but it does not support conservative pins and other additional options that are available for Foster network. If you require options not provided in Rational Fitting, use the Foster network approach.

The following example shows the additional options displayed when **Foster Network** is selected:



The Foster network thermal model is represented either an by equivalent thermal network (Equivalent circuit model) or in state-space format (State-Space model). Each representation is either Conservative or Non-Conservative. Select **Conservative** to connect the generated thermal component to an external thermal network through

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

conservative thermal terminals; the **Non-Conservative** thermal component uses input heat source quantities and computes the temperature output quantities.

To enable selection of the Conservative model type, the Icepak thermal system needs to contain at least one heat source and temperature measurement point pair; in other words, at least one thermal model that has heat dissipation needs to have temperature measurement assigned to it. If there is no heat source and temperature measurement point pair in the Icepak system, the Conservative option is disabled, and only the Non-Conservative option will be available.

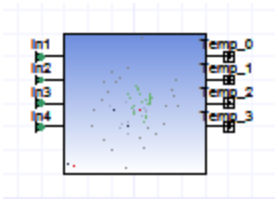
If **State-Space model** representation is selected, the requirement for selecting the Conservative model type is even stricter (due to model restrictions). In this case, all the heat source variables and the temperature measurement points have to be in pairs; in other words, all the thermal models with heat dissipation must have temperature measurement points assigned. If this condition is not satisfied when State-Space model is selected, then the **Conservative** option is disabled, and only the **Non-Conservative** option is available.

6. Click **Generate**.

   Twin Builder generates the new component. Information on the **Parameters** and **Terminals** for the component appear in their respective panels. Use the check boxes in the **Add Pin** column in the **Parameters** grid to toggle pin display .

> **Note:**
>
> Depending on the complexity of the Icepak output, several seconds may elapse before the component is generated and available for placement on the schematic.

7. Click **OK** to create a component and symbol (using the image from the 3D model) and model and add them to the project. The following shows an example:



8. Position the new component on the schematic and make the appropriate connections between it and existing schematic components.
9. Run Twin Builder analyses as desired.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

State-Space Components

State-space components represent a physical system as *n* first-order coupled differential equations. See the topics in this section for more information.

Program Requirements for State-Space Components

- Twin Builder current version.

Overview of State-Space Component Coupling

The state-space component represents a physical system as *n* first-order coupled differential equations. This form is better suited for computer simulation than an n^{th} order input-output differential equation.

The general vector-matrix form of the state-space model is:

$$\frac{dx}{dt} = Ax + Bu$$

$$y = Cx + Du$$

where ***y*** is the output equation, and ***x*** is the state vector.

Detailed description of variables:

- ***x*** – Internal states (dynamics).
- ***u*** – Inputs (function of time).
- ***y*** – Outputs (function of time).
- ***A***, ***B***, ***C***, and ***D*** – State-space matrices.
- ***IC*** – Matrix of initialization values for the state vector.

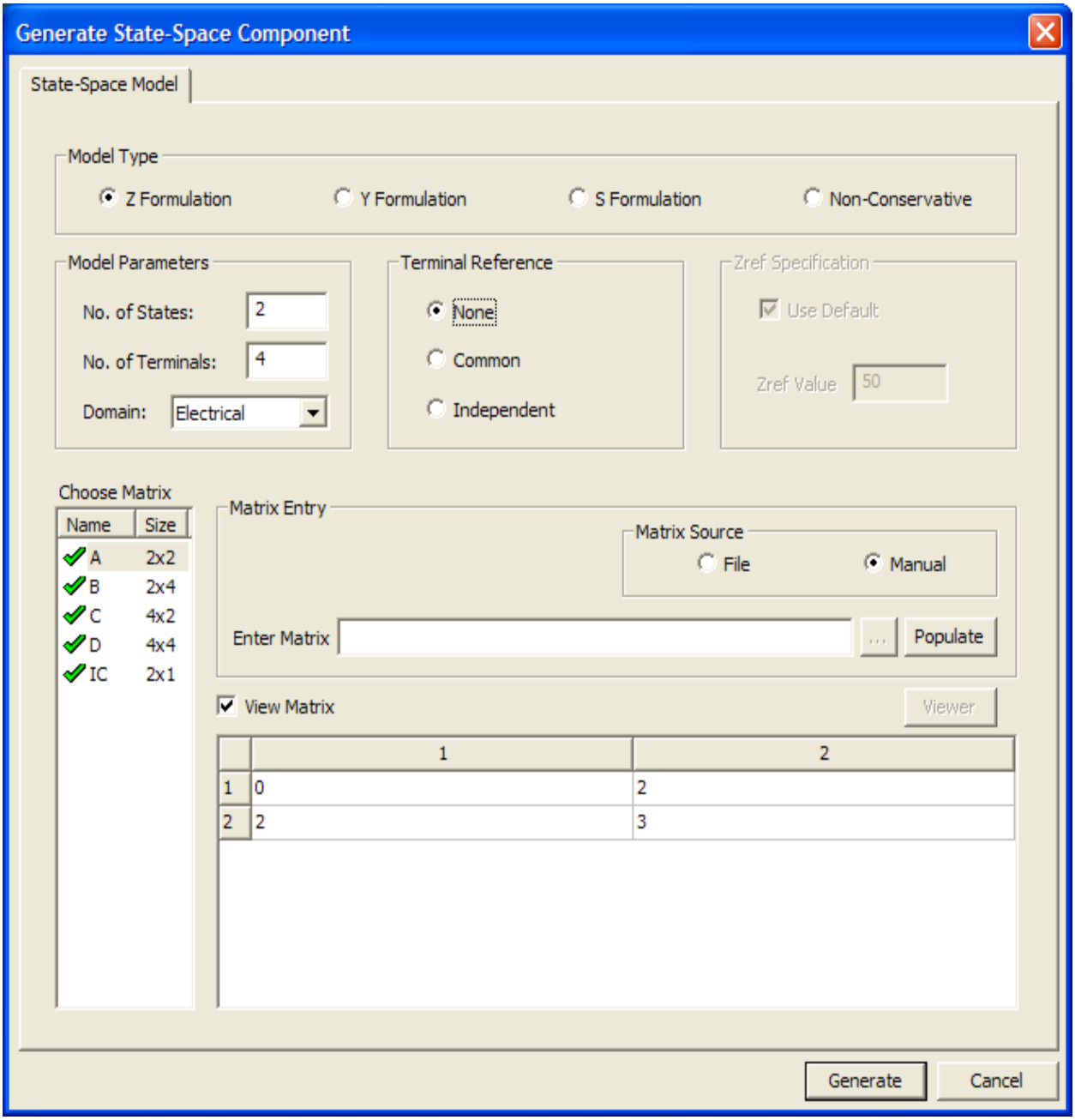
Sizes of matrices are as follows:

- ***A*** = (number of states) x (number of states)
- ***B*** = (number of states) x (number of inputs)
- ***C*** = (number of outputs) x (number of states)
- ***D*** = (number of inputs) x (number of outputs)
- ***IC*** = number of states

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Generate State-Space Component Dialog Box

Use the **Generate State-Space Component** dialog box to define the parameters needed to generate a state-space component.



Model Type Panel

The state-space component supports both **Conservative** and **Non-Conservative** models.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Conservative** models have bi-directional terminals where the number of inputs equals the number of outputs. The state-space component currently supports three matrix forms:
  - Z Formulation
  - Y Formulation
  - S Formulation

    The S Formulation requires specification of a **Zref Value** for each port. A default value of 50 ohms can be used, or you can specify values independently for each port via the matrix list.

- **Non-Conservative** models have separate, non-conservative input and output terminals.

Model Parameters Panel

For Conservative models, you can specify the number of states, number of terminals, and the domain (that is, the nature) of the terminals. For Non-Conservative models, you can specify the number of states, number of inputs, and number of outputs. The values you specify determine the sizes of the state-space matrices in the matrix list.

Terminal Reference Panel

In a physical system each terminal may be assigned a reference port with respect to which the measurement of physical quantities is performed. The state-space model implementation provides the following choices for Conservative models:

- **None** – No reference port is added to the model.
- **Common** – One common reference port is added to the model.
- **Independent** – Each terminal has its own separate reference port added to the model.

Choose Matrix List

This list contains all the matrices whose values need to be specified to generate the model. You must specify at least the A, B, and C matrices to enable the **Generate** button for component generation. Check marks next to each matrix name indicate if the matrix has been populated or accepted. Sizes of matrices are calculated based on the model parameters. After you select a matrix in the list, use the **Matrix Entry** fields to specify the matrix source and to populate it. When you select the **View Matrix** check box, a spreadsheet-style table lets you view and edit the matrix entries.

Matrix Entry Group Panel

Use the settings in the **Matrix Entry** group to specify a correctly-sized matrix as a list of floating point values. All values must be specified. For example, a 3X3 matrix should have nine values in row-major format.

Matrix Source - either of two matrix sources can be chosen:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Manual** – Enter matrix data manually in the **Enter Matrix** field.
- **File** – Import the matrix data with a text file.

Matrix data must be entered in row-major order separated by any of the following delimiters: space, comma, or semicolon. In text files, you can also use newline characters to break the entry at the end of each row.

Click **Populate** or directly edit the matrix in the table to populate the matrices. Attempting to populate the matrix with incorrectly sized or formatted data generates an error message: “Size of input does not match matrix size”.

By default, the **View Matrix** box is selected. The selected matrix data can be viewed and edited directly in the matrix table. When **View Matrix** is cleared, the **Viewer** button is enabled and can be used to view the matrix data in a text editor such as Notepad.

Adding a State-Space Component

To add a State-Space component:

1. On the Twin Builder main menu, select **Twin Builder > Subcircuit > Add State Space Component**.

   The **Generate State-Space Component** dialog box appears.

2. Select the desired **Model Type**: **Z Formulation**, **Y Formulation**, **S Formulation**, or **Non-Conservative**.
3. In the **Model Parameters** panel:
   - For conservative models (that is, **Z**, **Y**, or **S Formulation**), enter the number of **States** and **Terminals**, and select the physical **Domain**.
   - For **Non-Conservative** models, enter the number of **States**, **Inputs**, and **Outputs**.
4. For conservative models, in the **Terminal Reference** panel, choose the reference port assignment for the model’s physical ports.
   - **None** – No reference port is added to the model.
   - **Common** – One common reference port is added to the model.
   - **Independent** – Each terminal has its own reference port added to the model.
5. For **S Formulation** models, the **Zref Specification** panel controls are enabled for use as follows:
   a. Select **Use Default** to use the specified **Zref Value** for all ports in the model.
   b. Clear **Use Default** to specify the Zref value independently for each model port. A Zref matrix entry is added to the **Choose Matrix** list box for this purpose. Select the Zref matrix in the list and use the **Matrix Entry** fields to specify the Zref values. You can also enter the values directly in the matrix table if **View Matrix** is enabled.
6. Specify values for the model matrices. At a minimum, the **A**, **B**, and **C** matrices must be specified.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

a. Select the matrix for which you want to enter values from the entries in the **Choose Matrix** list.

The **Matrix Entry** panel fields are enabled. Also, if **View Matrix** is selected, an editable matrix table displays.

b. Select the **Matrix Source** and specify values.

> **Note:**
>
> Matrix data must be entered in row-major order separated by any of the following delimiters: space, comma, or semicolon. In text files, you can also use newline characters to break the entry at the end of each row.

- **Manual** – Enter the matrix values directly, then enter the desired values in the **Enter Matrix** box or the matrix table.
- **File** – Import the matrix values from a delimited text file; click to open a file selection dialog box. The selected text file path is entered in the **Enter Matrix** box.

c. Click **Populate** to transfer the entered matrix values into the model.

7. Repeat step 6 for each matrix in the list.

> **Note:**
>
> At a minimum, the **A**, **B**, and **C** matrices must be specified.

8. Optionally, you can edit matrix values directly in the matrix table, provided that **View Matrix** is selected.
9. Optionally, clear **View Matrix** to enable the **Viewer** button to view the matrix data in a text editor such as Notepad. Typically this would be used for viewing large matrices. Do not edit the data in the viewer.
10. Click **Generate**. Twin Builder generates the new component.

> **Note:**
>
> Depending on the complexity of the model, several seconds may elapse before the model is generated and available for placement on the schematic.

11. Position the new component on the schematic and make the appropriate connections between it and existing schematic components.
12. Run Twin Builder analyses as desired.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Viewing and Editing State-Space Component Matrix Values

To view matrix values for existing state-space components, double-click the component instance on a schematic to open the component dialog box in read-only mode.

By default, the **View Matrix** box is selected and the selected matrix data appears in the matrix table. When **View Matrix** is cleared, the **Viewer** button is enabled and can be used to view the matrix data in Notepad.

Edit matrix values for an existing state-space component instance with the **Edit Component** dialog box. Follow this procedure:

1. Expand the **Definitions** > **Components** subfolder for the project that contains the instance to edit. Double-click the entry for the component to edit, or right-click the entry and select **Edit Component** to open the **Edit Component** dialog box.
2. Click **Properties** on the **General** tab to open the **Properties** dialog box.
3. On the **Parameter Defaults** tab, click **StateSpaceModel** in the **Value** field to open the component dialog box in which you can change matrix values as needed. When finished, click **OK** to close the component dialog box, click **OK** to close the **Properties** dialog box, and click **OK** to close the **Edit Component** dialog box and complete the change process.

Double-click the component instance on the schematic to re-open the component dialog box in read-only mode and confirm the changes.

Changing or Moving Coupling Model Files

It is important to note that the components placed on a schematic are copies or instantiations of the components in the Twin Builder libraries. When placed on the schematic, changes affect only the local instantiation, not the component definition. This is particularly important in the coupling area since the coupling component instantiated on the schematic is a link to a project file *outside* of the Twin Builder environment.

Changing Coupling Model Files

Follow this procedure to change the coupling model file for a schematic coupling component. Also, if the location (path) for a coupling model file associated with a schematic component changes, as it often does when moving projects and their related coupling files from one machine to another, you can use this procedure to re-establish the link to it.

1. In the Project tree, expand the **Definitions** > **Components** subfolder for the project that contains the coupling model to edit. Double-click the component to edit, or right-click the entry and select **Edit Component** to open the **Edit Component** dialog box.
2. Click **Properties** on the **General** tab to open the **Properties** dialog box.
3. On the **Parameter Defaults** tab, click the button in the **Value** field for the data property (the button’s label is the name of the coupling object).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- If the coupling model file is not in its original location, a message displays informing you that the file does not exist. Click **OK** to continue. The coupling dialog box for the object opens.

4. Click to locate and choose the desired model file to restore the link to it.
5. Click **OK** to close the dialog boxes.

Moving Coupling Model Files

When you save a Twin Builder project containing a coupling component, the current link to the coupling model associated with that component is also saved. If the location (path) for the coupling model file associated with a schematic component changes—for example, when moving projects and their related coupling files from one machine to another—Twin Builder opens a dialog box asking if you want to search for the missing coupling model file.

- Choosing **No** opens the project, but the coupling component retains the original path information to the related coupling model. You can then use the **Changing Coupling Model Files** procedure above to locate and re-establish the link to it.
- Choosing **Yes** opens the coupling dialog box for the component. Click to locate and choose the desired model file to re-establish the link to it.

SIwave Components

SIwave is a tool used by signal and power integrity engineers to study complex electromagnetic phenomena on printed circuit boards (PCBs) and IC packages. It can identify signal delay and excessive cross-talk between interconnects, deficiencies in the power distribution network, and regions in the layout that may cause EMC/EMI problems.

SIwave Full-Wave SPICE models are computed from broadband S-parameter data representing PCB geometry. They are compact macro-models used in circuit tools for time-domain simulations. They optionally incorporate corrections for passivity and causality violations that arise due to issues such as incorrect upstream material parameters.

Exporting an SIwave subcircuit lets you perform an analysis of the structure in Twin Builder. See the SIwave documentation for detailed information.

Program Requirements for SIwave Component Coupling

- Twin Builder current version.
- SIwave current version.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Overview of SIwave Component Coupling

To use an SIwave model:

1. Create and export a model of the component in SIwave.

   > **Note:**
   >
   > See **Getting Results** > **Computing Full-Wave SPICE Subcircuits** in the SIwave help for detailed information on computing and exporting the Twin Builder model.

   - If using SIwave v5, the export process generates three files: <*simplorer_model*>.**sml**, <*simplorer_model*>.**ss**, and <*simplorer_model*>.**jpg**.
   - If using SIwave v6.0 or later, the export process generates two files: <*simplorer_model*>.**sml**, and <*simplorer_model*>.**jpg**
2. On the **Twin Builder** main menu, select **Subcircuit** > **Add SIwave Component** to add the component to the sheet.

   In the **Select SIwave generated SML file** dialog box, navigate to the desired SIwave generated .**sml** file and click **Open** to place the component on the schematic sheet.

> **Note:**
>
> The associated component image (<*simplorer_model*>.**jpg**) file (and state space <*simplorer_model*>.**ss** file if exported using SIwave v5) must be in the same location as the <*simplorer_model*>.**sml** file.

3. Create the rest of the simulation design on the sheet.
4. Start the simulation in Twin Builder.

   > **Note:**
   >
   > To make changes in the SML model, you must edit the SIwave design and re-export the model.

HFSS Components

Exporting an HFSS model lets you perform an analysis of the structure in Twin Builder. See the HFSS documentation for detailed information on exporting an HFSS model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Program Requirements for HFSS Component Coupling

- Twin Builder current version.
- HFSS current version.

Overview of HFSS Component Coupling

To use an HFSS model:

1. Create and export a model of the component in HFSS.

   > **Note:**
   >
   > See the HFSS help for detailed information on computing and exporting the Twin Builder model.

   The export process generates three files: <*simplorer_model*>.**sml**, <*simplorer_model*>.**ss**, and <*simplorer_model*>.**jpg**.

   - If using HFSS v13, the export process generates three files: <*simplorer_model*>.**sml**, <*simplorer_model*>.**ss**, and <*simplorer_model*>.**jpg**.
   - If using HFSS v14.0 or later, the export process generates two files: <*simplorer_model*>.**sml** and <*simplorer_model*>.**jpg**.
2. On the **Twin Builder** main menu, select **Subcircuit** > **Add HFSS Component** to add the component to the sheet.

   In the **Select HFSS generated SML file** dialog box, navigate to the desired HFSS generated .**sml** file and click **Open** to place the component on the schematic sheet.

> **Note:**
>
> The associated component image (<*simplorer_model*>.**jpg**) file (and state space <*simplorer_model*>.**ss** file if exported using HFSS v13) must be in the same location as the <*simplorer_model*>.**sml** file.

3. Create the rest of the simulation design on the sheet.
4. Start the simulation in Twin Builder.

   > **Note:**
   >
   > To make changes in the SML model, you must edit the HFSS design and re-export the model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding an HFSS Dynamic Coupling Component in Twin Builder

Follow this procedure to add an HFSS dynamic coupling component to a Twin Builder design.

1. On the **Twin Builder** main menu, select **Subcircuit** > **Add HFSS Dynamic Component**. The **HFSS Dynamic Coupling** dialog box appears.
2. Specify a **Name** for the component. See ***Names of Components and Variables*** for naming conventions.
3. In the **Source Project & Design** panel, enter the project **File**. Click to find and select the HFSS (**.aedt**) file that you want to use.

   The selected project loads into its application.
4. Based on the **Source** file and the **Link Type** (HFSS Link), Twin Builder communicates with HFSS, retrieves the applicable **Design** and **Solution** information, and populates the corresponding menus with the information.

   The **Information** tab updates to show the **Number of Conservative Pins** and various other **Quantities** present in the selected design.
5. After you select the **Source** file, select the design you want to use in the coupled design from the **Design** drop-down list.
6. Select the desired solution from the **Solution** drop-down list.
7. Optionally, on the **Options** tab:
   a. Select **Save project after use** to have the linked project file saved by its application after co-simulation is completed.
   b. Select **Unload project after use** to have the linked project file closed by its application after co-simulation is completed.
   c. Select **Create static link** to create a static link and enable the **Clear Static Cache** button.
      - If **Create static link** is not selected, Twin Builder obtains sml/netlist information from HFSS every time a simulation analysis is run, even if no parameter has changed. This is the default behavior.
      - If **Create static link** is selected, Twin Builder obtains sml/netlist information from HFSS only if a parameter value has changed since the previous simulation run. Click **Clear Static Cache** to delete the sml/netlist information currently stored in the Twin Builder model. The next simulation run will then obtain fresh sml/netlist information from the other product application and store it in Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- You also have the choice to set passivity, error tolerance, and maximum order. These options are similar to what appears in the Network Data.

By default, pin names display. The option to **Show Description** is disabled.

8. When finished, click **OK** to accept the values and close the dialog box.
9. Place the component on the schematic sheet.
10. After placing the component on a sheet, you can open its **Properties** dialog box to configure the component.

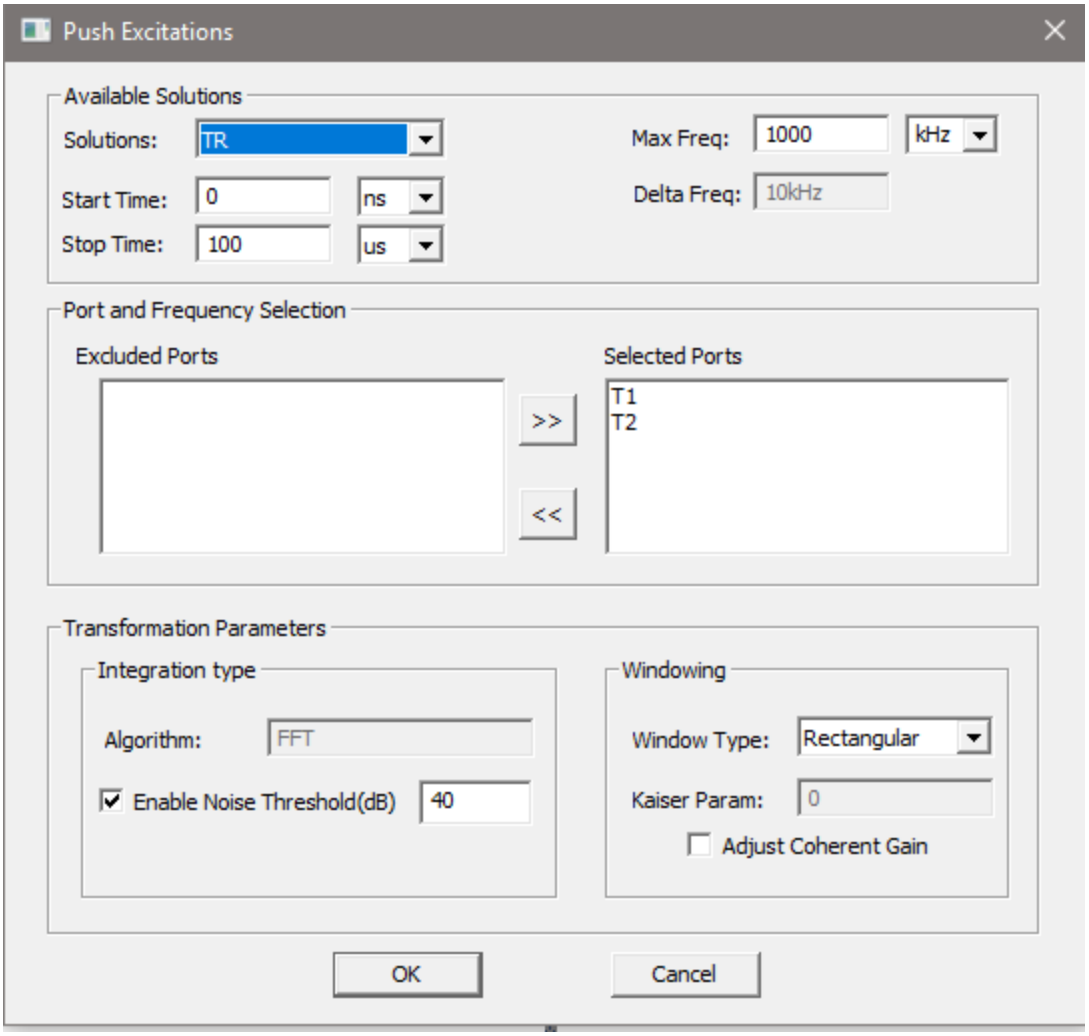
    The dynamic coupling component configuration is saved with the schematic sheet. Twin Builder must establish a link to the associated application each time a simulation runs.

Push Excitations for HFSS and SIwave Components

Twin Builder provides a way to export and push back excitation results of a transient simulation to the current versions of HFSS and SIwave.

1. Create or open a coupling design containing a coupling component created in HFSS or SIwave.
2. Run a transient simulation to generate results.
3. Right-click the HFSS or SIwave coupling component and select **Push Excitations...** to open the **Push Excitations** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. In the **Push Excitations** dialog box, select the various excitation options to control the transformation of signal from transient to spectral domain. The **Push Excitations** dialog box takes the results data from the available solutions. The algorithm defaults to **FFT**. Select the solutions of interest and choose the **Transformation Parameters** as explained in ***Plotting Spectral Domain Data***.
5. Click **OK** to push the excitations.
6. The excitations will appear in the HFSS design datasets and in the **Edit Sources** dialog box for the HFSS design.

ANSYS RBD Components

ANSYS Rigid Dynamics can export a Functional Mockup Unit (FMU). You can directly import This FMU into Twin Builder for co-simulation. Starting with Release 24.1, this is a recommended

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

solution for simulating an RBD component in Twin Builder.

The Transient-Transient Co-simulation link between Twin Builder and ANSYS Rigid Dynamics available in earlier releases is no longer supported after Release 23.2.

Ansys Fluent LTI Components

This option provides the ability to import ***.simpinfo** files from Ansys Fluent, generate a state-space model, and include it in a Twin Builder project for simulation and analysis. See the Ansys Fluent documentation for detailed information on setting up a Fluent project. For more information about Linear and Time Invariant (LTI) systems, see Thermal Analysis using a Foster Network.

Program Requirements for Ansys Fluent LTI Component Coupling

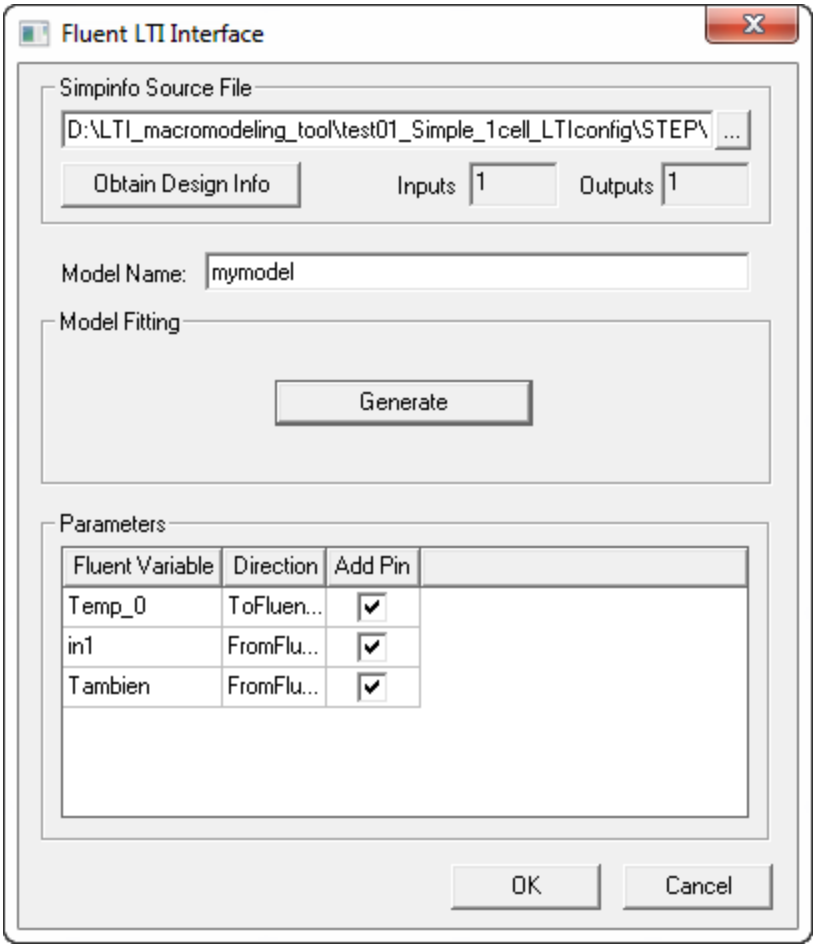
- Twin Builder current version.
- Ansys current version.

Overview of Ansys Fluent LTI Component Coupling

1. On the Twin Builder main menu, select **Twin Builder** > **Subcircuit** > **Fluent Component** > **Add Fluent LTI Component**.

   The **Fluent LTI Interface** dialog box appears. The following example shows the results after the model is generated.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. In the **Simpinfo Source File** panel, click and select the appropriate **.simpinfo** file to create a new component in Twin Builder.

   > **Note:**
   >
   > The output **.out** files generated by Fluent must be present in the same directory as the **.simpinfo** file. If the Fluent output files are missing, an error dialog box displays, and the component will not be generated.

   The panel also indicates the number of **Inputs** and **Outputs** present in the output file.

3. Click **Obtain Design Info** to retrieve design information for the chosen source file.
4. Optionally, change the **Model Name**.

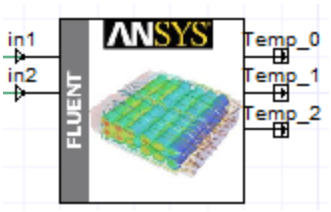
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. In the **Model Fitting** panel, click **Generate**.

   Twin Builder generates the new component. Information on the **Parameters** for the component displays. Toggle pin display with the **Add Pin** check box column in the **Parameters** grid.

   > **Note:**
   >
   > Depending on the complexity of the Fluent output, several seconds may elapse before the component generates and is available for placement on the schematic.

6. Click **OK** to create a component, symbol (using the image from the 3D model), and model, and add them to the project. For example:



7. Position the new component on the schematic, and make the appropriate connections between it and existing schematic components.
8. Run Twin Builder analyses as desired.

Motor-CAD ROM Components

The topics in this section explain the use of Motor-CAD ROM components in Twin Builder.

ProgramRequirementsforMotorCADROMCoupling

Program requirements to use Motor-CAD ROM capabilities and use them in Twin Builder:

- Twin Builder Version 2024.2 and later.
- Ansys Motor-CAD.

Overview of Motor-CAD ROM Components

Use Motor-CAD to evaluate the thermal response of the motor at different operating conditions. Typical results from Motor-CAD include a thermal network representation of the motor defined by a set of interconnected nodes, with capacitances, resistances, temperature and power sources. Based on these results, you can implement a system level model in the form of a

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

thermal network that can be used to evaluate the dynamic response of the model connected with other system level components.

Since the motor operates at a varying rotational speed, it is important to include its dependency on the network parameters, such as capacitances and resistances. Beyond dependency on external inputs like the rotational speed, the network parameters might also have temperature dependencies; for example, when nonlinear effects such as natural convection and/or radiation are included in the model.

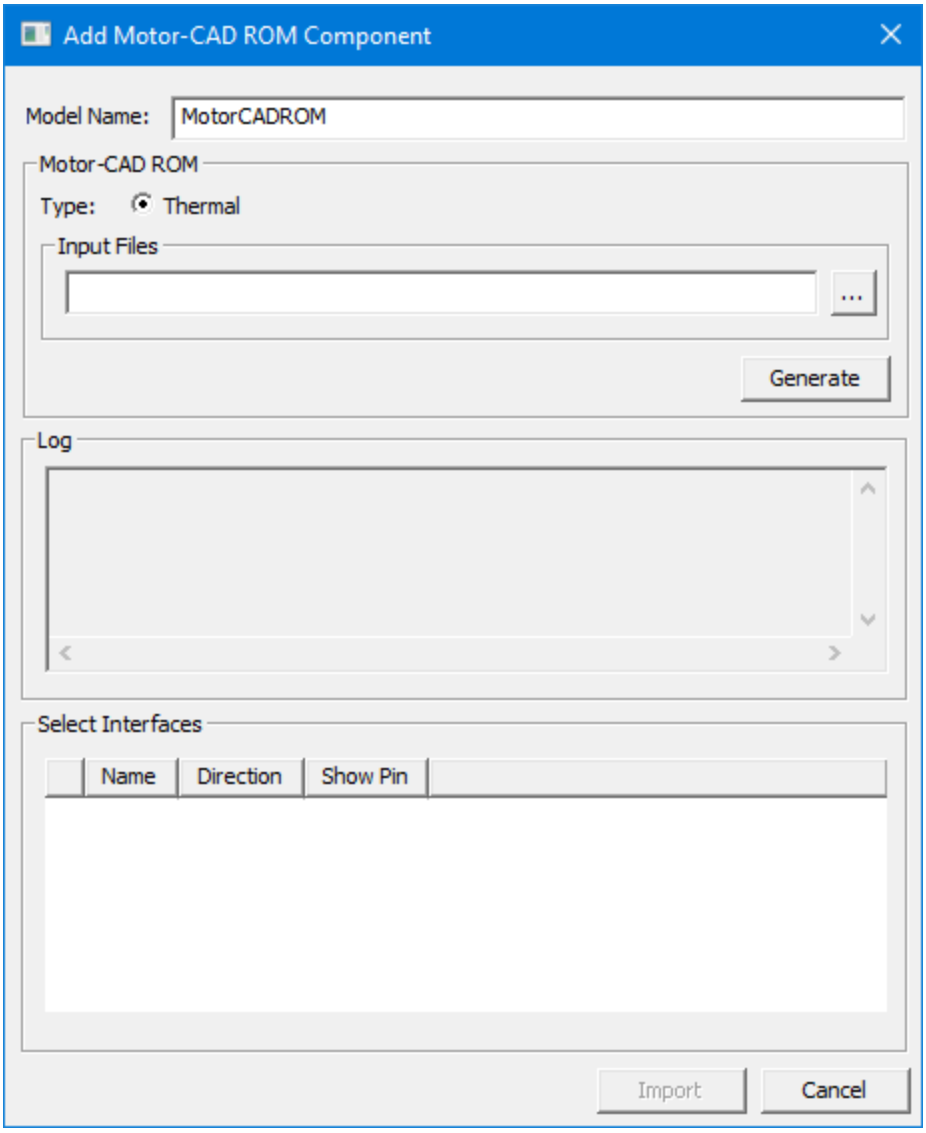
To build a Motor-CAD ROM component in Twin Builder, sweep the Motor-CAD model under different conditions (for example, testing different rotational speed), extract the corresponding results and assemble them in the expected format for Twin Builder (an example Python script will be made available to show the overall workflow with PyMotorCAD). Once the data are available, Twin Builder extracts a nonlinear thermal network and implement it in the form of a new component, taking power losses as inputs, and making predictions of individual nodes temperatures. You can then connect that component to other system components.

Add a New Motor-CAD ROM Component

Follow this procedure to add a Motor-CAD ROM component in Twin Builder.

1. Select **Twin Builder** > **Add Component** > **Add Motor-CAD ROM Component**. The **Add Motor-CAD ROM Component** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. Enter a name in the **Model Name** field.
3. In the **Input Files** field, click ... and browse to the directory containing your Motor-CAD results.
4. Click **Generate**.
5. Click **Import**.

Once the Motor-CAD ROM is generated and compiled into a component, you can then place the component into a schematic.

You can use the **Symbol** menu to select interfaces that you want to expose.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Any issues that occurred while generating the model will appear in the log.

Edit or Update a Motor-CAD ROM Component

Follow this procedure to edit an existing Motor-CAD ROM component.

1. Right-click a component on the schematic and select **Edit model**.

   - In the **Input Files** field, can click to change the directory containing your Motor-CAD results.
   - Click **Generate** to regenerate the Motor-CAD ROM.
   - You can also update the interface selection.

2. Click **Update** to save the Motor-CAD ROM with your changes.

Ansys SCADE Components

Esterel Technologies' SCADE Suite® models can be included in a Twin Builder project for simulation and analysis. See the Esterel SCADE documentation for detailed information on generating and exporting an Esterel SCADE Suite® model.

Access SCADE-related example projects by selecting **File** > **Open Examples**, browsing to **Twin Builder\Applications**, and selecting a project from either the **BLAC Motor Drive System** folder or the **Water Pumping System** folder. Additional documentation is provided in PDF format in these folders.

Program Requirements for SCADE Component Coupling

- Twin Builder current version. A 64-bit Functional Mock-up Unit (FMU) exported from SCADE can be imported through the FMU link.
- Ansys SCADE Suite® current version.

FMU Components

Functional Mock-up Units (FMUs) are component models developed using the Functional Mock-up Interface (FMI) tool independent standard. An FMU model file is a zipped file (***.fmu**) containing the XML description file and the implementation in binary form. The file may also contain other files such as source files, symbol graphics, and documentation files for the component. See the FMI web site (https://fmi-standard.org/) for detailed information.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> When you import an FMU and are shown a choice to select Model Exchange/Co-Simulation, if you select an option not supported by the imported FMU, the simulation terminates with an error during initialization.

Program Requirements for FMU Component Coupling

- Twin Builder current version.
- FMU components developed per the FMI standard.
- Supports FMI Version 1.0 and 2.0.
- Supports only FMI for model exchange and co-simulation.
- Supports only 64-bit FMU.

Overview of FMU Coupling with Twin Builder

The following overview describes the procedure for adding an FMU component in a Twin Builder design. Twin Builder supports import of model exchange and co-simulation. It supports both FMI standard 1.0 and 2.0.

1. Obtain or create an FMU using the FMI standard.

   > **Note:**
   >
   > See the FMI web site (https://www.fmi-standard.org/) for detailed information on creating an FMU model.

2. Select **Twin Builder** > **Add Component** > **Add FMU Component** and use the **Select FMU Model File** dialog box to locate and open the FMU model **.fmu** file to add to your schematic.

   When you select an FMU file, the **Import FMU Model** dialog box lets you choose interfaces to expose in the Twin Builder component.

   - **Select Local Variables** – Include internal or local output properties in the model. Leave it cleared to exclude these properties from selection.
   - **Select Interfaces at Level** – Choose the desired hierarchy level from the drop-down list. The value you choose selects input and output interfaces up to the chosen hierarchy level for inclusion in the model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

An FMU often has thousands of interfaces. For better performance, select only necessary inputs and outputs. Use **Edit Model** to add/remove interfaces in the model.

- Interface names that start with "_" are grouped under **AdvancedSettings**.
- Interface names that indicate derivative (der(<name>)) are grouped under **der()**.

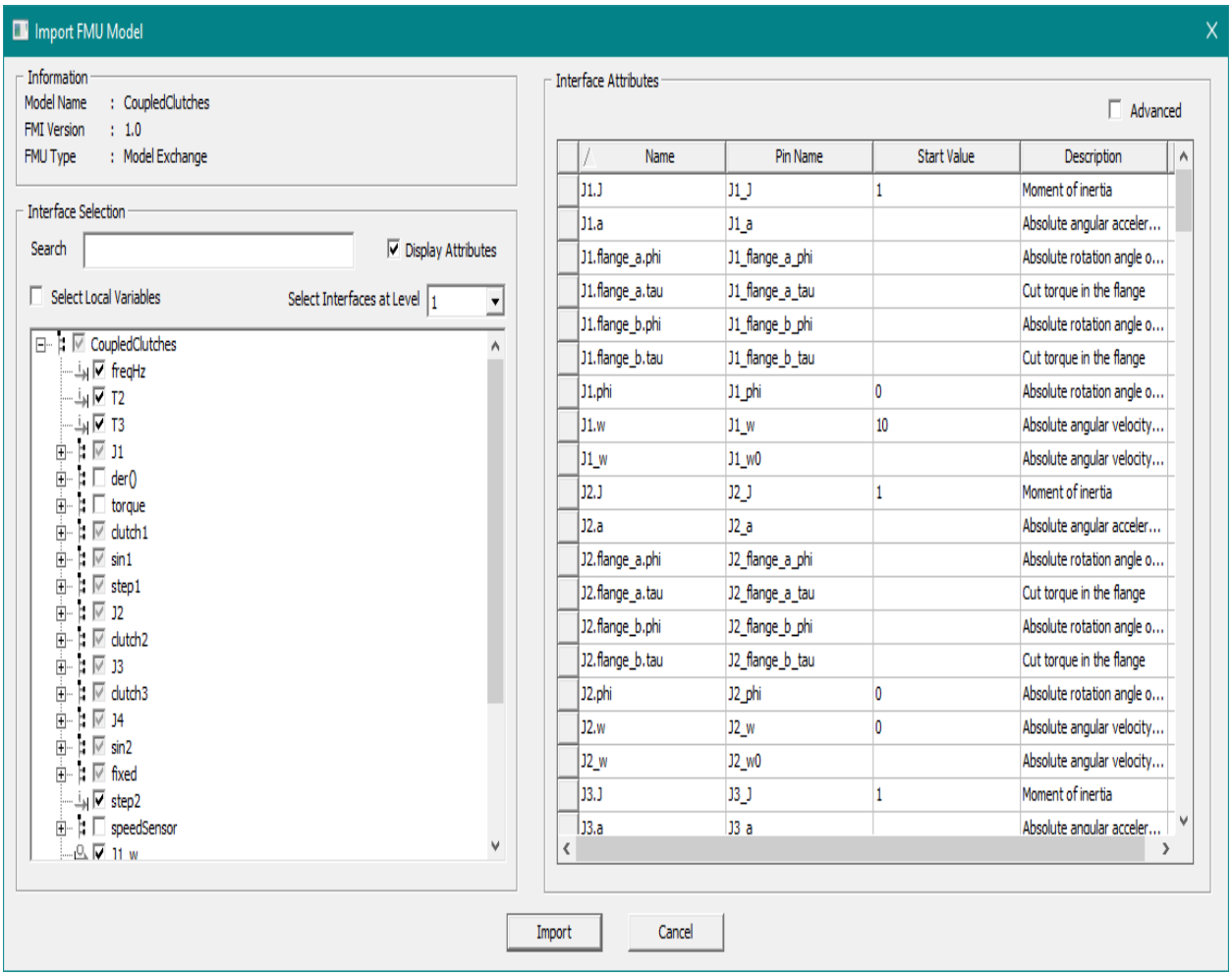


Figure 13-5 FMU Version 1.0 for Model Exchange

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

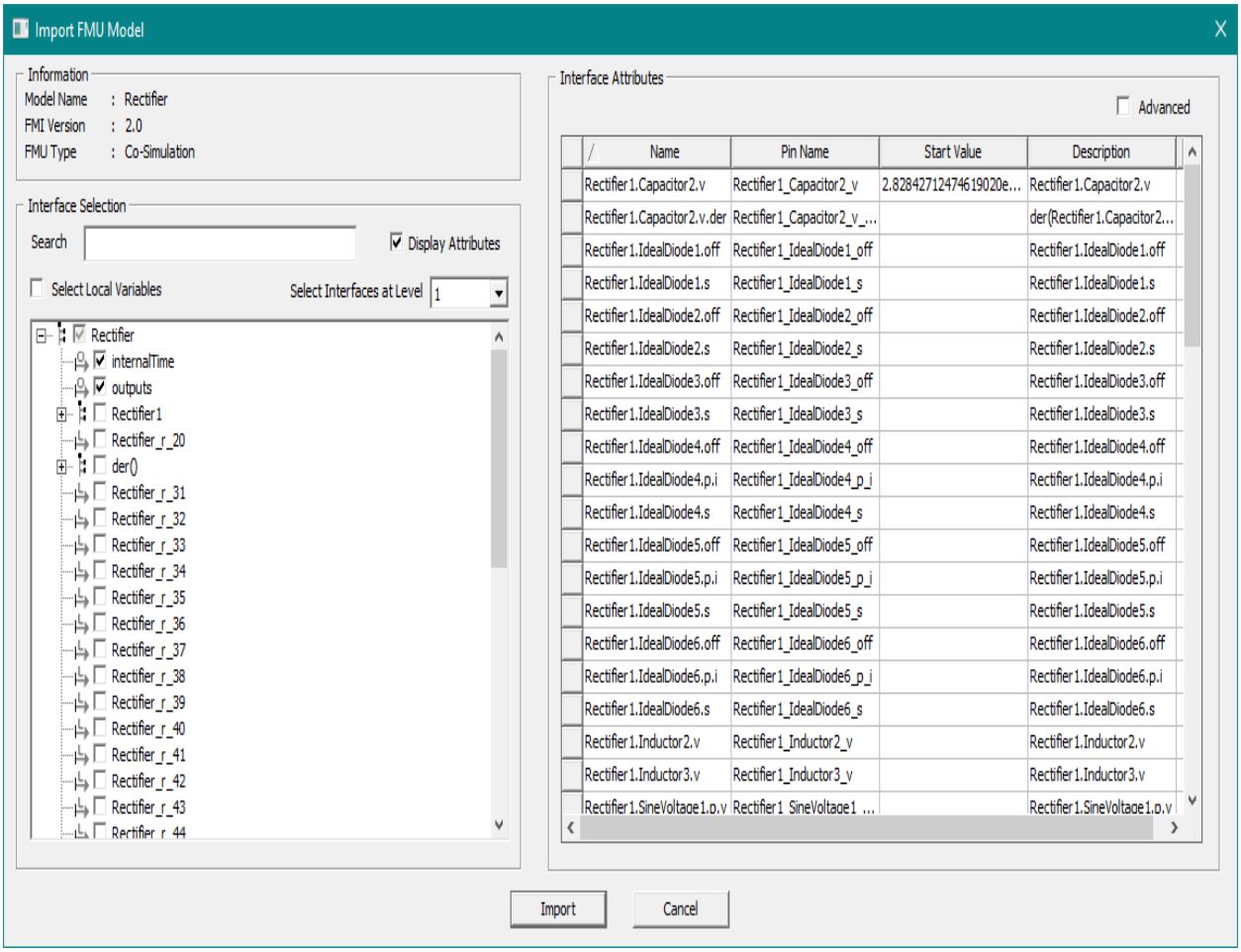
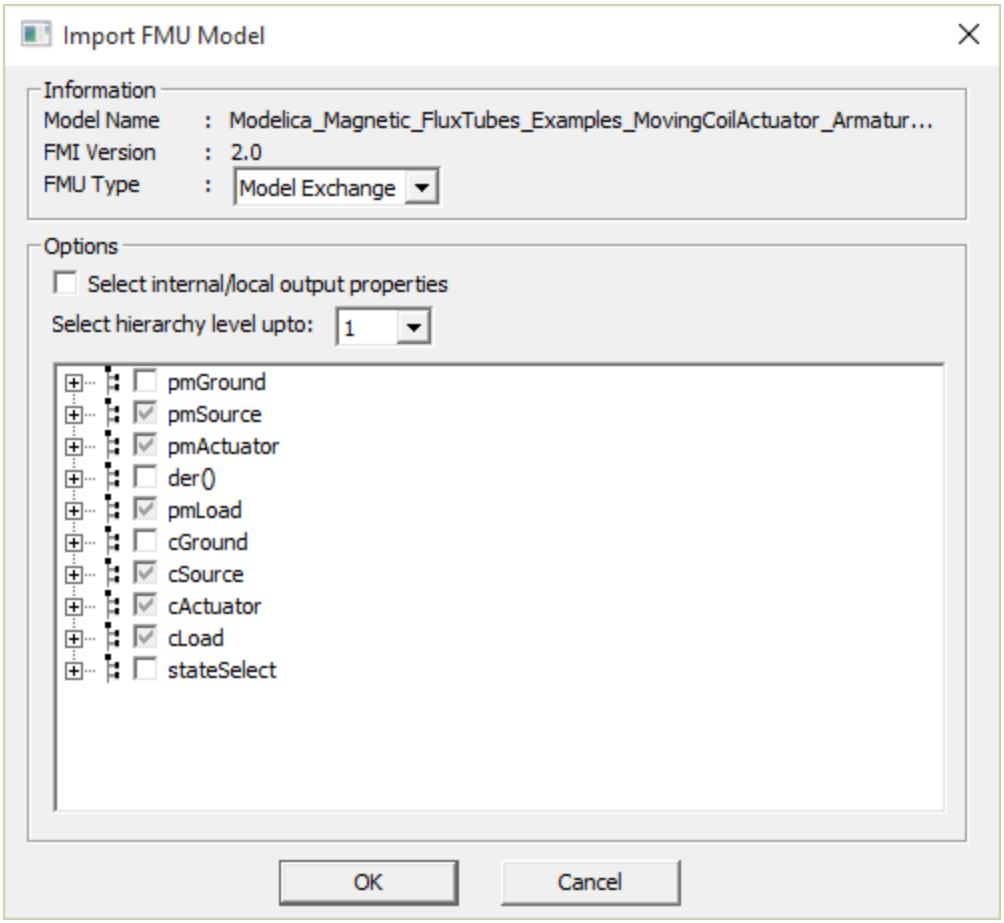


Figure 13-6 FMU Version 2.0 for Co-Simulation

The following figure shows the drop-down list provided when an FMU supports both **Model Exchange** and **Co-Simulation**. Use the drop-down list to change the type of FMU.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

When you import an FMU and are shown a choice to select **Model Exchange/Co-Simulation**, if you select an option that is not supported by the imported FMU, the simulation terminates with the following error: **Unable to load DLL for FMU_Link_**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

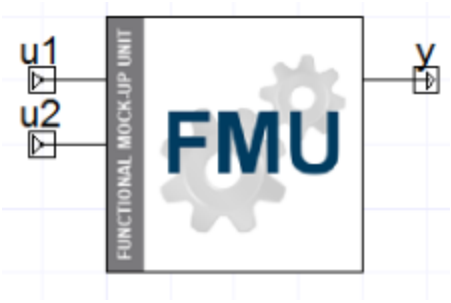
Twin Builder doesn't support periods (.) or brackets ([ ]) characters in interface or property names; they will be replaced by an underscore (_).

For example,

- `assembly.subassembly.part.length` becomes `assembly_subassembly_part_length`
- `assembly.part[0].length` becomes `assembly_part_0_length`

After you click **OK**, Twin Builder brings the FMU into the project file and creates a component ready to place on the schematic. If the FMU model contains a model symbol **.png** file, it is used as the component symbol. Otherwise, Twin Builder uses the default symbol shown below.

3. Place the component on the schematic.



Double-click the FMU component to open its **Properties** dialog box, in which you can view and edit the various quantities and parameters.

4. Create the rest of the simulation design on the schematic.
5. Simulate in Twin Builder and analyze results.

Initialization of Input Values

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Initial values of inputs can be specified using two ways:

- Input pin values at time=0 by setting **UseStartValueInitialization** to “false”.
- Corresponding **_start** parameters by setting **UseStartValueInitialization** to “true”.

The first option is appropriate if the source of the input is from constant blocks, FML_INIT block, timer function components, and so on. However, if the input is connected to a quantity that depends on other components, it could lead to initialization failure due to improper initial values. Specifying the initial conditions for the inputs (the second option) is always the preferred way to initialize the model. The start parameters have the same name of the input plus **_start** suffix.

TS_TwinBuilder: Sample Time Parameter for FMI Co-Simulation

For Co-Simulation FMUs, Twin Builder has an additional parameter called TS_TwinBuilder. If this value is not specified, the FMU is evaluated at the same (variable) time step as the circuit simulator. When it is specified, Twin Builder evaluates this FMU only at every sample point.

> **Note:**
>
> Ansys recommends that you do **not** rely on sample time for a Co-Simulation FMU that only supports **fixed step-size**. There is no absolute guarantee that **all** the sample points will be hit **exactly** at the required intervals when the Twin Builder solver takes a variable time stepping. Therefore, in order to guarantee a successful simulation, we highly recommend that you set a fixed step scheme as follows:
>
> HMIN = HMAX = Required fixed step-size of the FMU

Replacing an FMU with an FMU of a Different Standard

You can replace a model from FMI 1.0 with FMI 2.0, and vice versa.

1. Right-click the FMU in the schematic and select **Edit Model**.
2. In the **Update FMU Model** dialog box, click **Replace** and select the new model. The dialog box shows the new FMI version.
3. Click **OK** to update with the new model.

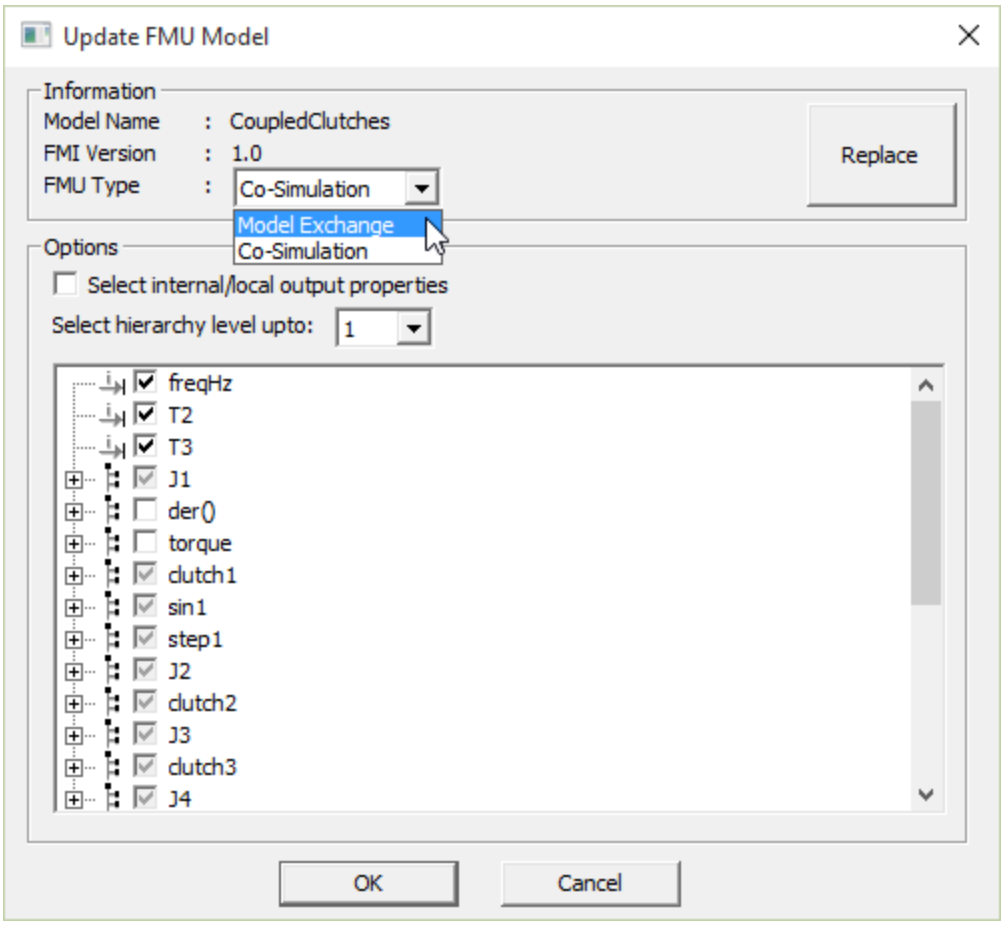
   The FMU component definition and all instances of this component are updated. The FMU component uses the new FMU file.

Replacing an FMU with a Different FMU Type

You can replace an FMU with a different FMU type.

1. Right-click the FMU in the schematic and select **Edit Model**.
2. In the **Update FMU Model** dialog box, click **Replace** and select the new model. The dialog box shows the FMI version and FMU type. You can change the type by using the **FMU Type** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



3. Click **OK** to update with the new model.

   The FMU component definition and all instances of this component are updated. The FMU component uses the new FMU file.

Replacing an FMU with an FMU that has Different Outputs

You can replace an FMU with an FMU that has different outputs.

1. Right-click the FMU in the schematic and select **Edit Model**.
2. In the **Update FMU Model** dialog box, click **Replace** and select the new model. The dialog box shows the new FMI version.
3. Click **OK** to update with the new model.

   The FMU component definition and all instances of this component are updated. The FMU component uses the new FMU file.

Replacing an FMU with an FMU that has Different Identifiers

Follow this procedure to replace an FMU with an FMU that has different Identifiers.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Right-click the FMU in the schematic and select **Edit Model**.
2. In the **Update FMU Model** dialog box, click **Replace** and select the new model. The dialog box shows a message that the new FMU model identifier doesn't match with the current one and asks if you want to continue with the update.
3. Click **Yes** to update the component.
4. Click **OK**

   The FMU component definition and all instances of this component are updated. The FMU component uses the new FMU file.

Storage of large FMU files (>100MB)

Normally, FMU model files that get imported are included in the project file. With large FMU files (>100MB), this can lead to a significant slowdown of saving and loading the project file. In order to prevent a negative impact, large FMU files are stored on disk in a special project data folder (for example, <*projectdir*>*project1*.**aedtdata**). Since those files are part of the project, this folder needs to stay with the project file. We recommend that you use the file commands available in Ansys Electronics Desktop (**Save As**, **Rename**, **Archive**, **Restore Archive**) in order to perform project operations that would change the location or name of the project file. If those operations are done manually using an external file explorer, you must include and adjust the data file folder and its content. Also, in that case the proper functioning of the FMU components is not guaranteed. In most cases, if an external FMU data file got lost, you can recompile it if the model text is available. You can also click **Update Model** on the FMU component to import the FMU file again.

Interface Grid

Select the **Display Attributes** check box to open the interface grid, which displays the variables' attributes. Use the **Advanced** check box to toggle the number of rows displayed.

Selecting a variable in the interface tree will highlight the corresponding attributes in the interface grid.

Variable Search

To search for a variable in the tree, enter a variable name in the **Search** box. The tree display updates, showing all variables matching your search text.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> **Remove Unused Definitions** cannot be undone. If unused FMU models are removed, their external data files will also be removed from the disk. If the project is not saved with the same name before closing, any external FMU data file that was connected to the original project and was removed during **Remove Unused Definitions** cannot be retrieved. In order to save a cleaned-up version of a project under a new name, first save the project under the new name, remove the unused definitions, then save the new project again.

Stand Alone ROM Viewer

This section describes the Stand Alone ROM Viewer and explains how to use it.

Introduction

Use the Ansys Stand Alone ROM Viewer to open and consume twin models. Stand Alone ROM Viewer lets you visualize and export field results for any input parameters of a Reduced Order Model (ROM) parameter set. It is a light-weight and easily portable viewer in which individual data points are visualized as a point cloud. The ensemble of visualized data points of a model provides an optical representation of the model geometry.

Twin files are generated using Ansys Twin Builder and should contain ROMs in order to use the Stand Alone ROM Viewer. Only ROMs built with Static ROM Builder or Dynamic ROM Builder are supported.

Using Stand Alone ROM Viewer

This section contains the main steps for using Stand Alone ROM Viewer.

Input Data

The starting point for the Stand Alone ROM Viewer is a **.twin** file. This file, created with Twin Builder, contains the reduced order model, the corresponding evaluation functions, and the associated visualization data (geometry coordinates and views definition).

After importing a **.twin** file the following data can be loaded:

- A **snapshot** – Solution fields associated either to a design point from the input parameter space (Static ROMs), or to a timestep of a single scenario (Dynamic ROMs). Snapshots are stored in a binary file format. This format consists of a 64-bit integer value denoting the number of values in the field, followed by the corresponding 64-bit decimal values.

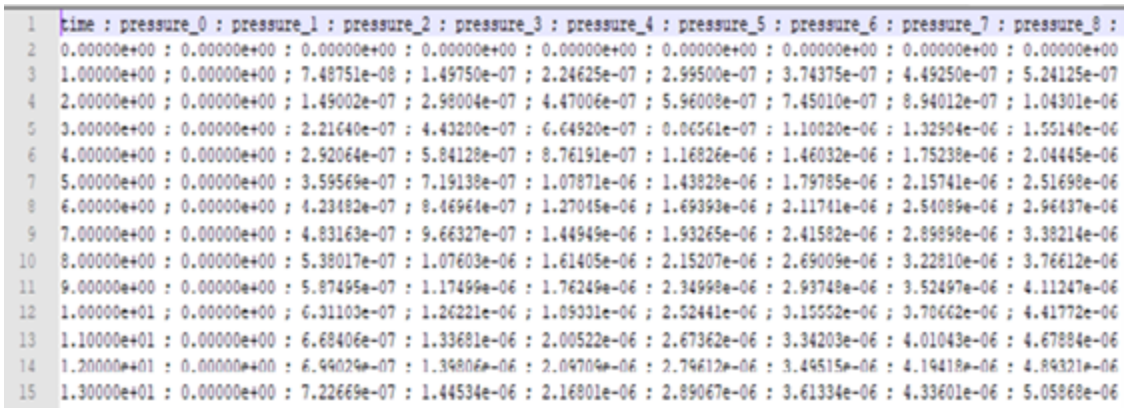
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- A **doe.csv (Static ROMs)** – Data table containing a list of snapshots to import and their corresponding input parameters values. The first column contains the snapshot names; the following columns contain the input parameter values. The file header reflects this as the first column should be called *name* followed by the input parameter names for each column. Acceptable separators are commas and semicolons. A specific folder structure is required for this case as shown below:

  Data used in that case are stored in a specific directory with the following architecture:

```
./directory_name
    ./snapshots
        ./file1.bin
        ./file2.bin
        ./file3.bin
        ./...
    ./doe.csv
```

- An **_exc.csv (Dynamic ROMs)** – File containing a single scenario's excitation data. It contains a time column, followed by a column for each excitation. Each column i represents the time history of the excitation $e_i\ (t)$. The figure below shows an example **_exc.csv** file for a case with applied wall pressure varying in time. Each column reflects a different position along the wall.



	time	pressure_0	pressure_1	pressure_2	pressure_3	pressure_4	pressure_5	pressure_6	pressure_7	pressure_8
2	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
3	1.00000e+00	0.00000e+00	7.48751e-08	1.49750e-07	2.24625e-07	2.99500e-07	3.74375e-07	4.49250e-07	5.24125e-07	
4	2.00000e+00	0.00000e+00	1.49002e-07	2.98004e-07	4.47006e-07	5.96008e-07	7.45010e-07	8.94012e-07	1.04301e-06	
5	3.00000e+00	0.00000e+00	2.21640e-07	4.43280e-07	6.64920e-07	8.86561e-07	1.10820e-06	1.32984e-06	1.55148e-06	
6	4.00000e+00	0.00000e+00	2.92064e-07	5.84128e-07	8.76191e-07	1.16826e-06	1.46032e-06	1.75238e-06	2.04445e-06	
7	5.00000e+00	0.00000e+00	3.59569e-07	7.19138e-07	1.07871e-06	1.43828e-06	1.79785e-06	2.15741e-06	2.51698e-06	
8	6.00000e+00	0.00000e+00	4.23482e-07	8.46964e-07	1.27045e-06	1.69393e-06	2.11741e-06	2.54089e-06	2.96637e-06	
9	7.00000e+00	0.00000e+00	4.83163e-07	9.66327e-07	1.44949e-06	1.93265e-06	2.41582e-06	2.89898e-06	3.38214e-06	
10	8.00000e+00	0.00000e+00	5.38017e-07	1.07603e-06	1.61405e-06	2.15207e-06	2.69009e-06	3.22810e-06	3.76612e-06	
11	9.00000e+00	0.00000e+00	5.87495e-07	1.17499e-06	1.76249e-06	2.34998e-06	2.93748e-06	3.52497e-06	4.11247e-06	
12	1.00000e+01	0.00000e+00	6.31103e-07	1.26221e-06	1.89331e-06	2.52441e-06	3.15552e-06	3.78662e-06	4.41772e-06	
13	1.10000e+01	0.00000e+00	6.68406e-07	1.33681e-06	2.00522e-06	2.67362e-06	3.34203e-06	4.01043e-06	4.67884e-06	
14	1.20000e+01	0.00000e+00	6.99029e-07	1.39806e-06	2.09709e-06	2.79612e-06	3.49515e-06	4.19418e-06	4.89321e-06	
15	1.30000e+01	0.00000e+00	7.22669e-07	1.44534e-06	2.16801e-06	2.89067e-06	3.61334e-06	4.33601e-06	5.05868e-06	

Launching Stand Alone ROM Viewer

The Stand Alone ROM Viewer executable file is named **ViewerStandAlone.exe**. It is located in the <*installation_directory*>**\AnsysEM**<*version*>**\Win64\common\ROMApps\ViewerStandAlone.exe** directory.

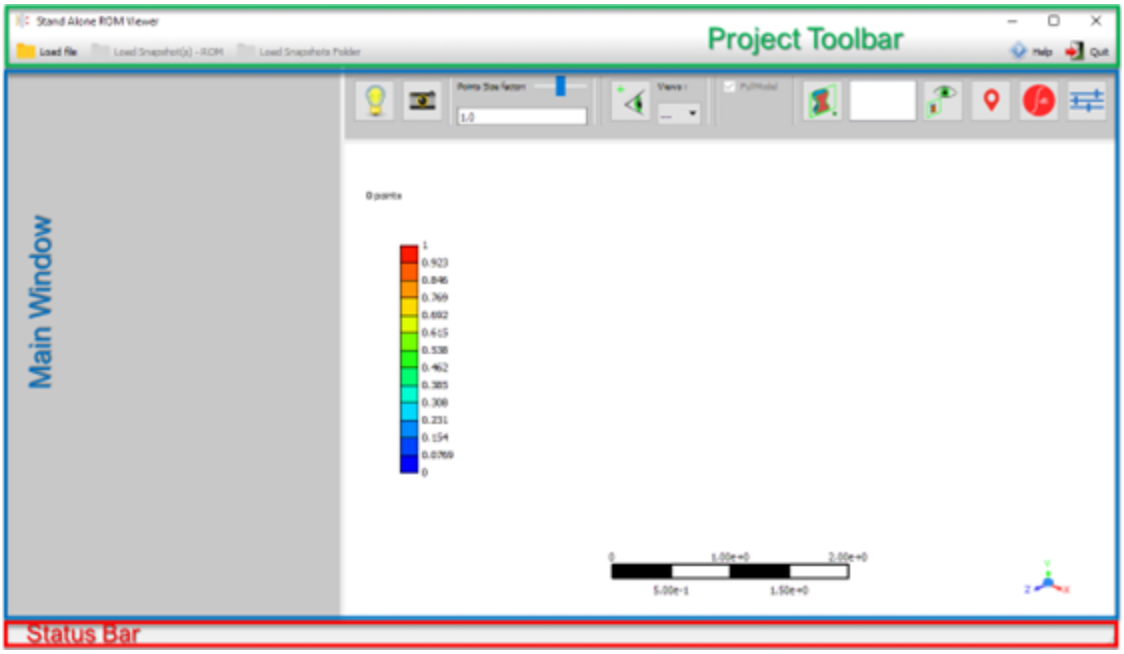
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

installation_directory and *version* must be substituted by appropriate locations for your installation. Launch this executable to open the graphical interface.

Graphic Interface Organization

The Stand Alone ROM Viewer graphical interface is composed of three main parts (see image below):

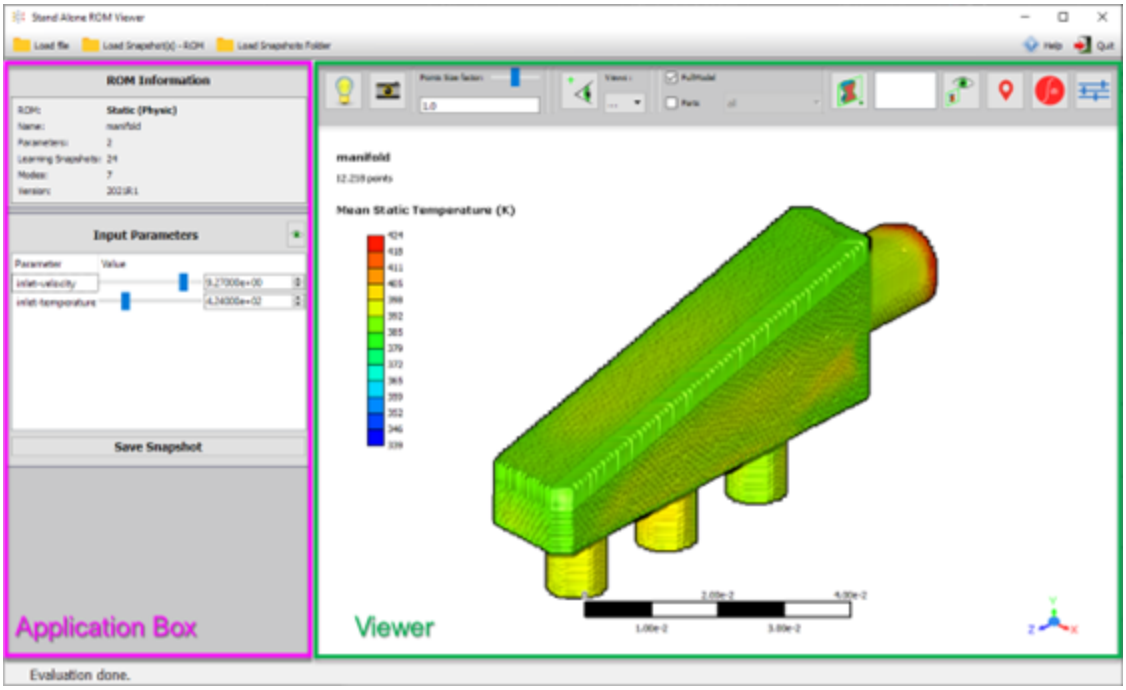
- **Project toolbar** – Used to import a **.twin** file and snapshot files.
- **Status bar** – Displays the current state of the application.
- **Main window** – Used for the ROM consumption process.



The main window is divided into two sections (see image below):

- The application box on the left side exposes information and possible actions related to the imported **.twin** file required for the ROM consumption process.
- The viewer on the right side displays the point cloud visualization of a ROM's output field.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Opening a Case (Project Toolbar)

Click **Load file** in the project toolbar to select the **.twin** file where the ROM is stored. The ROM information appears in the ROM information box and contains:

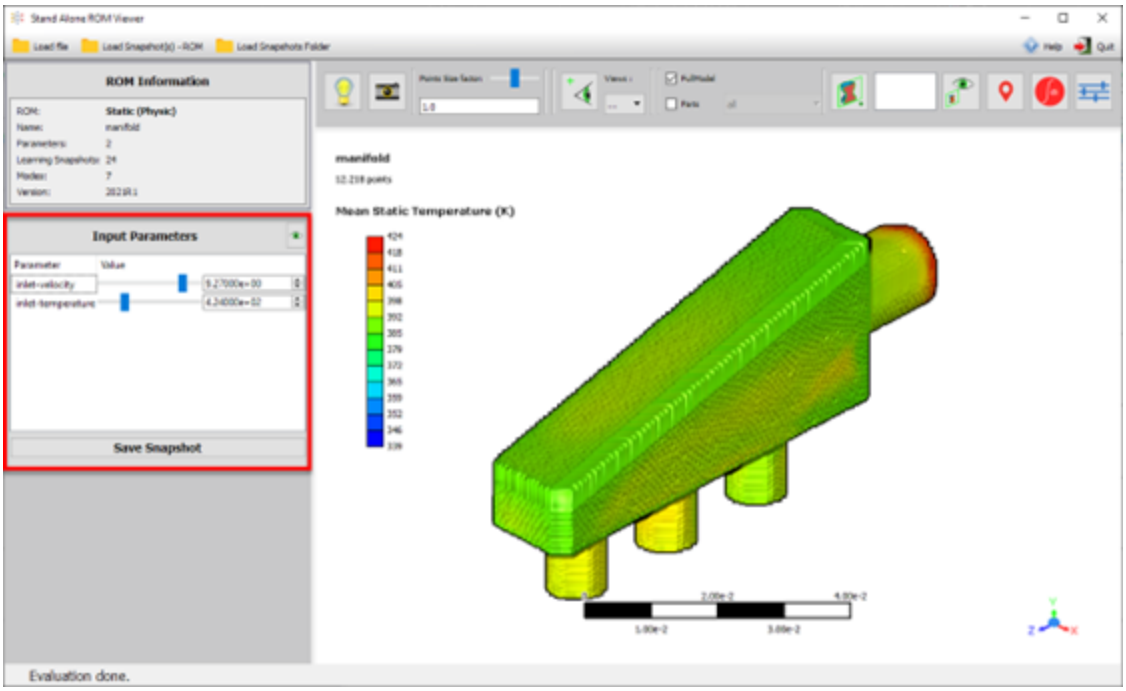
- Type of ROM loaded.
  - Static (Physic)
  - Static (Geometric)
  - Static (Physic with Geometric)
  - Dynamic
  - ModalField
- Name of the ROM.
- Number of input parameters.
- Number of learning snapshots.
- Number of modes used to build the ROM.
- ROM Builder version used.

The viewer displays the geometry after loading.

Consuming a Static ROM

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

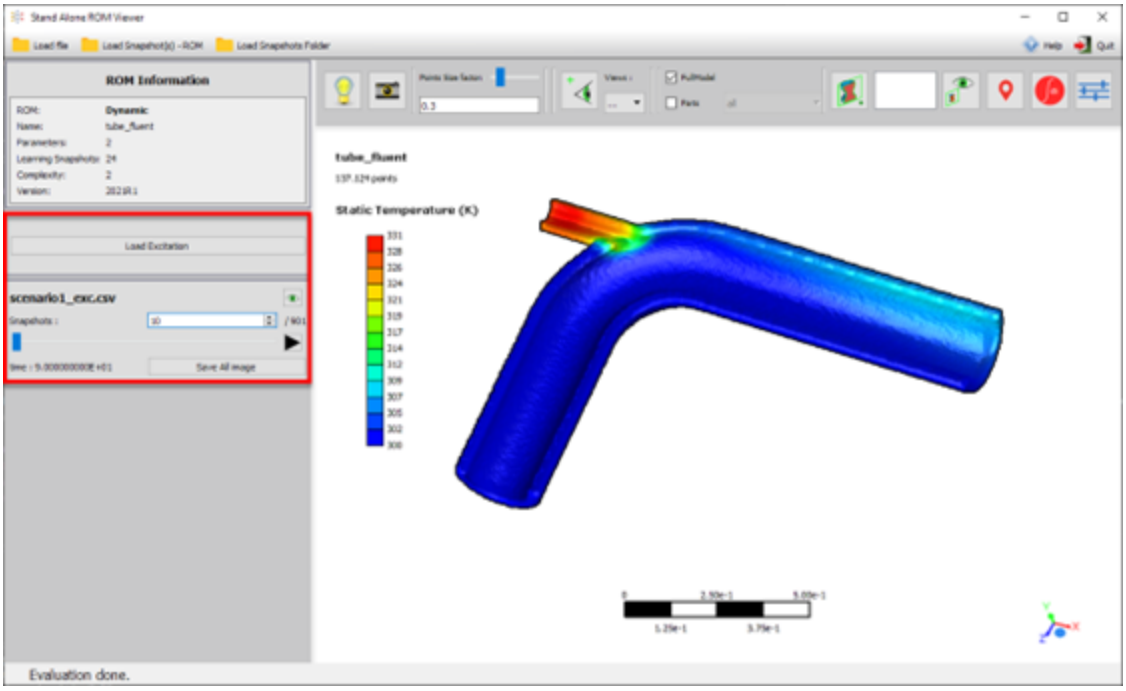
After loading a **.twin** file containing a Static ROM, you can directly use it in consumption mode. Use the sliders in the **Input Parameters** box to select input parameter values and visualize in real time the field approximated by the ROM. For any set of inputs, click **Save Snapshot** to store field results in a binary file.



Consuming a Dynamic ROM

For a **.twin** file containing a Dynamic ROM, a CSV file containing the excitations must be loaded. After loading a **.twin** file containing a dynamic ROM, click **Load Excitation** and import the corresponding ***_exc.csv** file. Use the slider to visualize the output field over time for the loaded excitation file. You can also manually select snapshots to visualize in the viewer by specifying its respective number.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



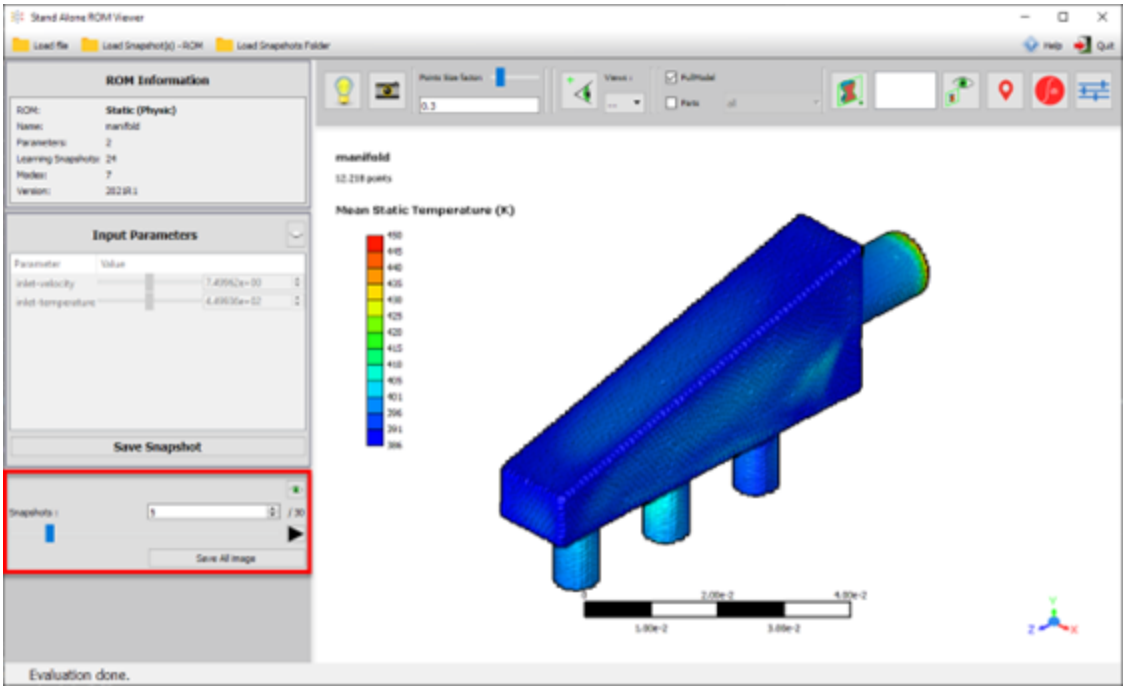
Opening One Snapshot File

Click **Load Snapshot(s)** in the project toolbar to select a binary file containing a snapshot. When importing a **.bin** file, the snapshot field appears in the viewer. This option is enabled for both Static and Dynamic twin model.

Opening a Folder of Snapshots

This option is enabled for both Static and Dynamic twin models. Click **Load Snapshots Folder** in the project toolbar to select a folder containing snapshots. When importing a folder, a box appears below the **Input Parameters** box (Static ROM) or below the **Load Excitation** box (Dynamic ROM). As described in the section **dyn-rom-consumption**, use either the slider or the numerical selection to visualize snapshots (see example figure below for Static ROM). Click ▶ to scroll through the snapshots in the viewer.

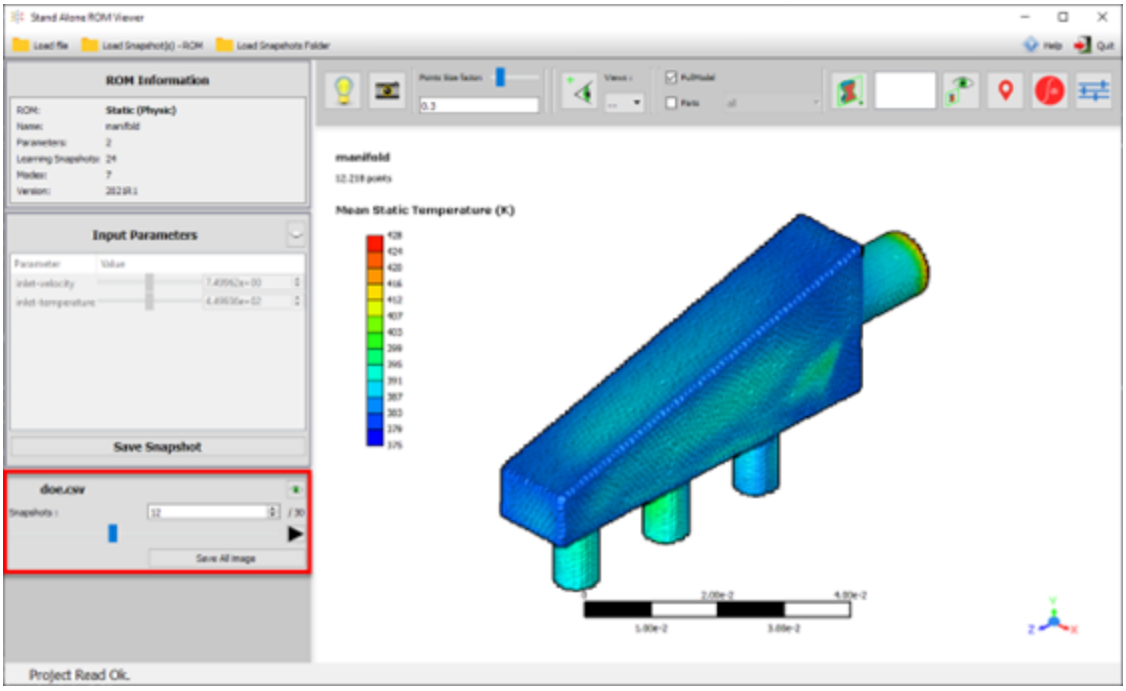
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Opening Snapshots Using a DoE File (Static ROM)

Click **Load Snapshot(s)** in the project toolbar to select the **doe.csv** file containing the list of snapshots to import. A box appears below **ROM information** (see figure below) that lets you select a snapshot to visualize in the viewer. Click ▶ to scroll through the snapshots in the viewer, in the order of the **doe.csv** file. Using a **doe.csv** for snapshot visualization is only available for Static ROMs.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

To return to the ROM consumption mode, click in the top right corner of the **Input Parameters** box.

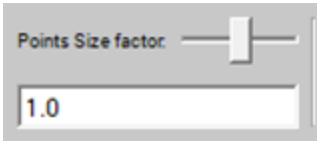
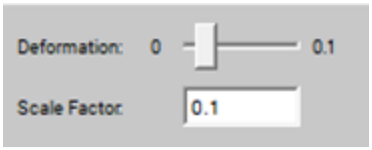
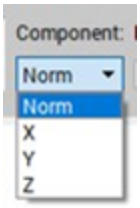
Viewer

The Viewer allows changing the point size, the orientation/zoom you are currently visualizing, and selecting the different parts of the objects to visualize. You can create different views (positions of the geometry in the 3D space) with their associated cut planes.

Presented below is a list of available actions; click the corresponding icon in the task bar located on the top of the 3D viewer window:

or	Toggle shading effects on the 3D geometry.
	Save a picture of the viewer displayed. Available formats are PNG, JPG, and BMP.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	Note: You can also right-click the visualization area to save a picture.
	Adapt the size of your geometry points with the scrollbar.
	Adapt the scaling factor of the deformation. This option is available only if the snapshot field is a deformation field.
 or	Add or remove a view.
	Visualize the selected view in the 3D viewer. Note that view **0** is the default view and cannot be deleted.
	Manage the field components visualized in the Viewer. For vector fields, the vector norms and individual vector components can be displayed. For symmetric tensor fields, the 6 tensor components, the Von Mises criterion and the max principal stress can be displayed. For scalar fields, this option is disabled.
	Select different parts of the geometry to display. This option is available only if the geometry is composed of several parts.
	Add a cut plane. Use Ctrl+scroll wheel to translate the cut plane along its normal vector and Ctrl+drag to rotate it.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	Delete the associated cut plane.
☑	Toggle display of a cut plane in the selected view.
or	Toggle display of cut plane contours.
	Manage (create, rename, or delete) probes scalar results associated with a specific point in the geometry.
	Manage (create, rename, or delete) operations scalar results calculated over a part of the geometry (min, max, average, norm).
	Manage Viewer settings: layout orientation, as well as the maximum and minimum scale values.

Ansys Examples

This section contains two examples:

Example 1: Static Fluent Case (**viewerExample1**) – Steady fluid simulation with two input parameters.

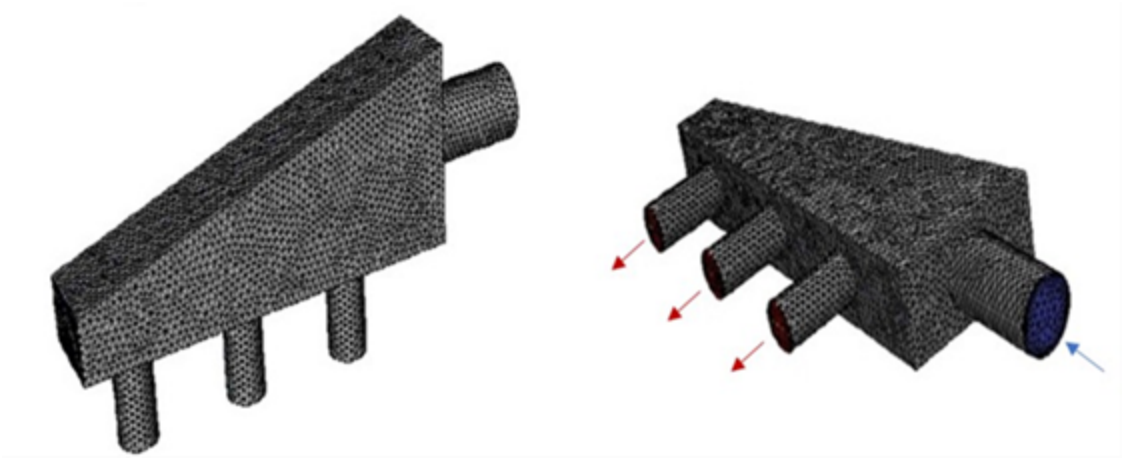
Example 2: Dynamic Fluent Case (**viewerExample2**) – Transient fluid simulation with two input excitations.

Ansys Example 1: Static Fluent Case

This ROM describes a 3D simplified inlet manifold. The geometry has one inlet and three outlets (in blue and red respectively in the figure below). The inlet was parametrized both in velocity and temperature: the velocity varies between 5 m/s to 10 m/s and the temperature also varies within a 400K to 500K range. The pressure at outlet was the same for all outlets and kept at 0Pa. The outside temperature was fixed at 300K with a constant heat-exchange coefficient at the external

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

wall fixed at 30 [W/m2/K]. A $k-\varepsilon$ Realizable model was used in that case to represent with a fair precision the heat transfer inside the manifold.



The inlet temperature and velocity are the ROM's exposed input variables, while the output is the temperature field inside the manifold. A design of experiments consisting of 30 design points was used to cover the parametric space and build the ROM. This example is located in <*installation_directory*>**\AnsysEM**<*version*>**\Win64\Examples\Twin Builder\Applications\StandAloneROMViewer\Ex1_static**. In which *installation_directory* and *version* need to be substituted by appropriate locations for your installation. It contains a ROM built from example1 from the Static ROM Builder documentation. The details of the design of experiments are stored in the **doe.csv** file, inside the example directory (see picture below). The snapshots directory contains all 30 snapshots extracted from each of the design points calculated in Fluent. Lastly, the **Manifold.twin** file created with Ansys Twin Builder contains the ROM built within Static ROM builder.

If not launched already, launch the Stand Alone ROM Viewer as described in ***Launching Stand Alone ROM Viewer***. Click **Load file** near the upper-left corner of the interface and select the **Manifold.twin** file to load the ROM. The **Input Parameters** box lets you consume your ROM directly by selecting any input parameters of the ROM parameter set. Click **Save snapshot** to evaluate a snapshot and store it in a binary file format.

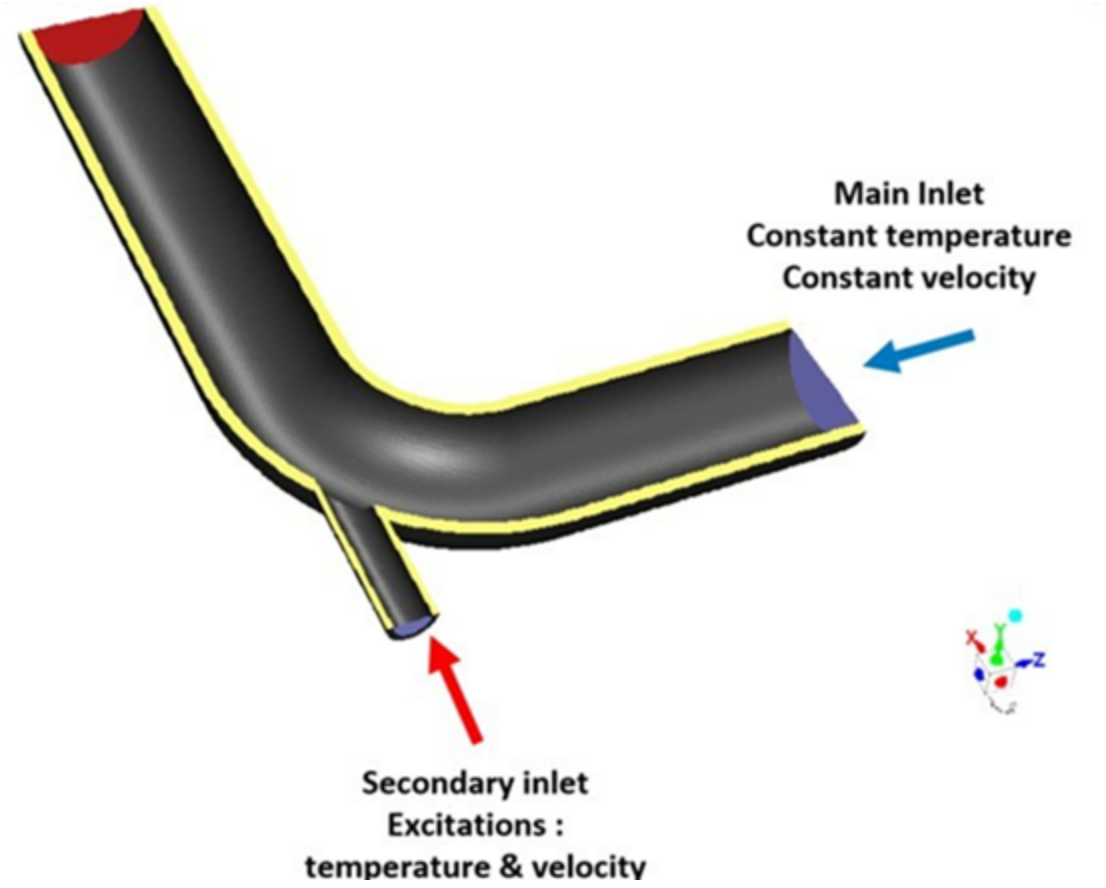
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Click **Load Snapshot(s)** to visualize a snapshot (***.bin**) file located in the **ex1_static\snapshots** folder. Similarly, you can load all the snapshots importing the **doe.csv** file.

Lastly, right-click the image displayed by the viewer to save the image or copy it to the clipboard.

Ansys Example 2: Dynamic Fluent Case

This example considers a 3D transient coupled heat transfer thermal simulation in Ansys Fluent. Both the fluid and solid domains are considered. The fluid is water steam at 20 bars while the solid material is common steel. A $k - \varepsilon$ model with scalable wall functions is used to represent the boundary layer effect on the flow and the heat transfers at the walls.



The velocity and the temperature over time at the secondary inlet are the excitations exposed inside the ROM. The temperature field at the walls is the output field evaluated with the dynamic ROM:

<*installation_directory*>**\AnsysEM**<*version*>**\Win64\Examples\Twin Builder\Applications\StandAloneROMViewer\Ex2_dynamic**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The example directory contains a ROM built from Example 2 from the Dynamic ROM Builder documentation. Inside the example directory you can find an excitation stored in the **scenario1_exc.csv** file, the **tube_model**.twin file created with Ansys Twin Builder and containing the ROM built within Dynamic ROM builder, and the snapshots available in the **ex2_dynamic\snapshots** folder.

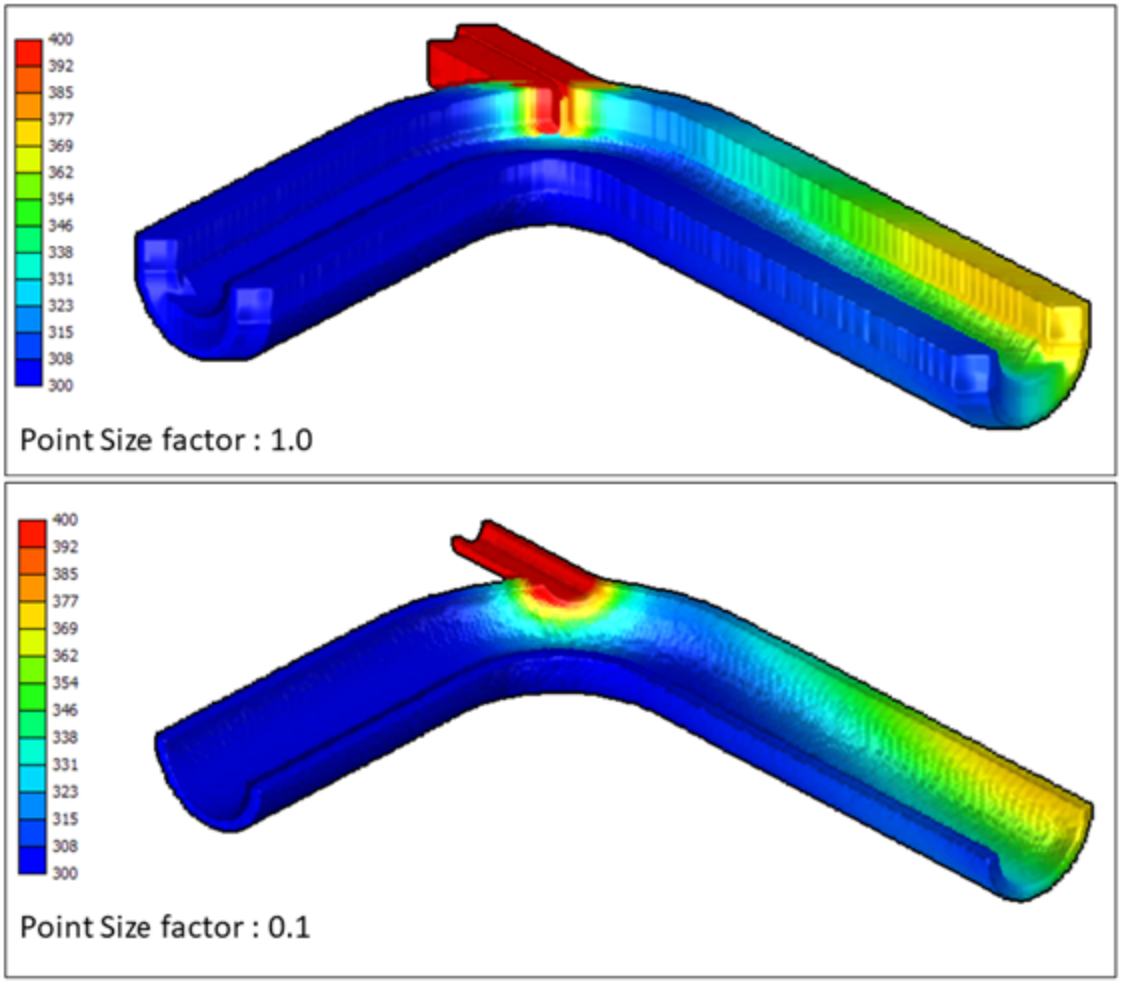
If not launched already, launch the Stand Alone ROM Viewer as described in ***Launching Stand Alone ROM Viewer***. Click **Load file** in the upper-left side of the interface and select the **tube_model.twin** file to load the ROM. Information about the ROM appears in the **ROM Information** box. Click **Load Excitation** and select the **scenario1_exc.csv** file.

For a better visualization of the results in this case, click to fix the color scale at a minimum value of 300K and a maximum of 400K.

Click to visualize the output field over time related to the imported excitations. Individual snapshots can also be selected in the viewer by specifying its number or using the slider. Additionally, you can load stored snapshots (**.bin** files); click **Load Snapshot(s)** and select a **.bin** file located in the **ex2_dynamic\snapshots** folder. You can load several snapshots at once; click the **Load Snapshots** folder to import all snapshots from a folder.

Optionally, change the Point Size factor using the Viewer's toolbar located above the viewer. The point size factor can be used to get a more realistic view of the geometry as shown in the image below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Static ROM Builder

Launching Static ROM Builder

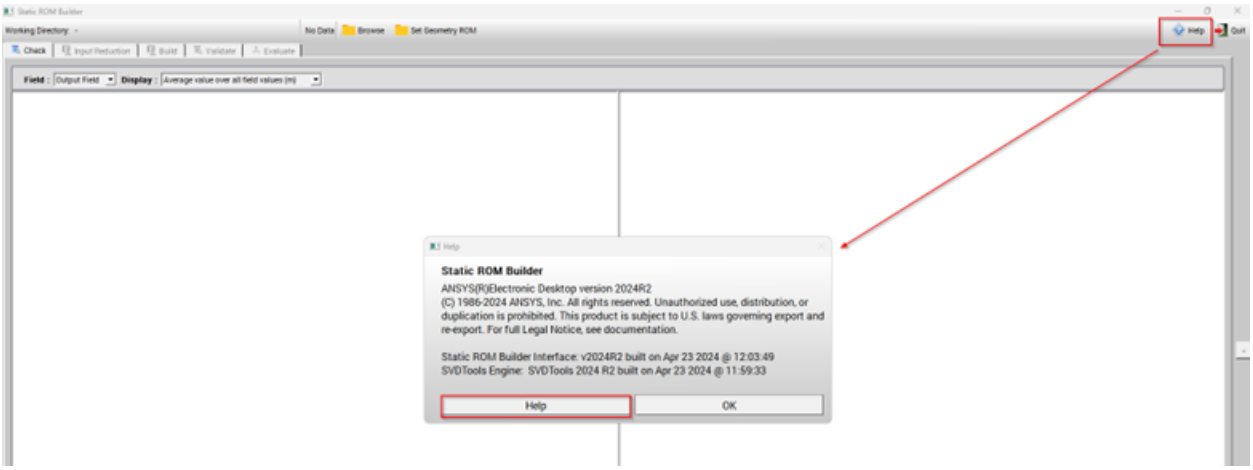
In the Schematic tab, click ROM > Static ROM Builder on the ribbon to open the Static ROM Builder graphic interface.

Shape

Accessing Static ROM Builder documentation

Click on the Help button in the ribbon, a pop-up window displays the version details. Then click on the Help button in this window, a web page will automatically open in your Browser with the expected documentation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



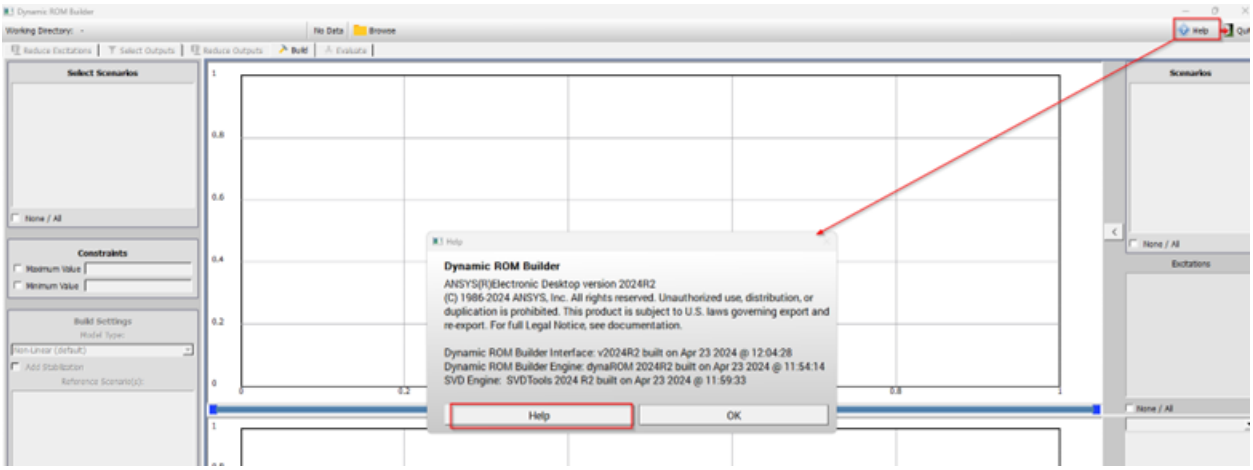
Dynamic ROM Builder

Launching Dynamic ROM Builder

In the Schematic tab, click ROM > Dynamic ROM Builder on the ribbon to open the Dynamic ROM Builder graphic interface.

Accessing Dynamic ROM Builder documentation

Click on the Help button in the ribbon, a pop-up window displays the version details. Then click on the Help button in this window, a web page will automatically open in your Browser with the expected documentation.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Data Connector Component

Use Data Connector to send and receive real values during the transient simulation to a remote application over the network using TCP/IP communication.

This component acts as a server, and the remote application needs to act as a client. It supports creating an arbitrary number of input and output ports. Communication occurs at every sample point. The following two modes of synchronization are supported:

- **Real-time** – Communication at every sample point is wall-clock based with non-blocking mode.
- **Blocked** – Communication at every sample point is blocked until the data is received/sent within a given timeout.

Program Requirements for Data Connector Component

Twin Builder – Host

- Twin Builder current version
- Windows 10
- Proper network connection
- Ability to listen to a user-specified TCP port – no firewall issues

Remote Client

- Any operating system that enables TCP socket communication
- Runtime support for the client application
- Proper network connection
- No firewall issues

Remote Application Development

- Compiler for any programming language that supports TCP socket communication.

Overview of Data Connector

The following overview provides information about Data Connector.

Connection Establishment

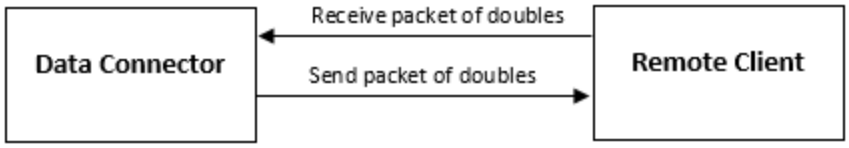
Data Connector starts a server listening for the client connection at the given TCP **Port Number** during the **Initialize** function. Only one client is accepted per component. When the client successfully connects within the **Connection Timeout**, it creates a dedicated TCP socket for the bidirectional communication with the client.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Client application on the remote machine can only connect to the Data Connector server after the transient analysis in Twin Builder is started on the host machine.

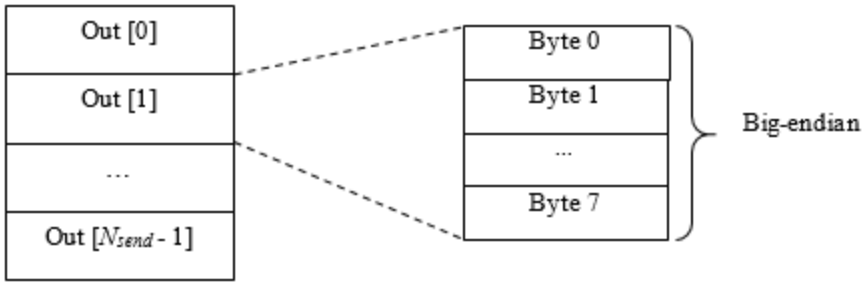
Data Transmission

Data (input and output real values) is communicated in binary format – 64-bit double precision values at every sample point are received/sent using the socket.



Packing the Data from Client Side

If the client has N_{send} output real values for sending to the Data Connector, it should create a byte-stream as follows:



Byte-stream to Twin Builder

As illustrated above, all the output values must be packed in the byte-stream to be sent through the socket. Every byte must be ordered in Big-endian (network) format.

Data is packed in the same way from the Twin Builder side to be sent to the client. Therefore, the client must have a buffer size of N_{recv} × 8-byte, where N_{recv} is the number of inputs from Twin Builder.

Real-time Communication Mode

Enable this mode of communication by selecting the **Sync to real-time** check box. In this mode, simulation synchronizes to the wall-clock at every sample point, and data is received and sent to the client. Receiving/sending calls do not block – that is, Data Connector does not wait until receiving/sending operations are completed. Therefore, if the client does not send any data by

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the time a receiving call occurs, Data Connector just moves on, and keeps the previous time-point values as the output.

For this mode to be successful, the following conditions must be satisfied:

- Simulation complexity must be small enough so that the simulation time never lags behind the real-time – simulation should not suffer due to convergence failures or stuck with smaller step sizes.
- The receiver should sample at an equal or faster rate than the sender – when the Data Connector has both input and output, both Data Connector and the client must use the same sample time.

Blocked Communication Mode

This mode of communication is enabled by clearing the **Sync to real-time** check box. In this mode, sample time is purely simulation based – no relation to the real-time. At every sample point, Data Connector waits for receive and send operations to be completed within the duration specified by I/O Timeout parameter. If data cannot be received by the timeout duration, it keeps the previous time-point values as the output.

Connection Termination

Data Connector shuts down and closes the socket at the end of the simulation. However, if the client disconnects or stops sending the data, simulation stops immediately.

Multiple Data Connectors

You can have multiple Data Connector components in a Twin Builder design. They are completely isolated, and each of them communicates through a separate socket. Therefore, you must create separate sockets on the client application for each of them. For successful simulation, the following conditions must be satisfied:

- Each component should have a distinct TCP port number.
- Twin Builder performs connection establishment one component at a time. Therefore, corresponding client sockets must create the connection in the same order. Typically, the order is determined by the order you placed on the components on the schematic. However, you can manually sort them by selecting **Schematic** > **Sort Components**.

Related Topics

Data Connector Component

Program Requirement for Data Connector Component

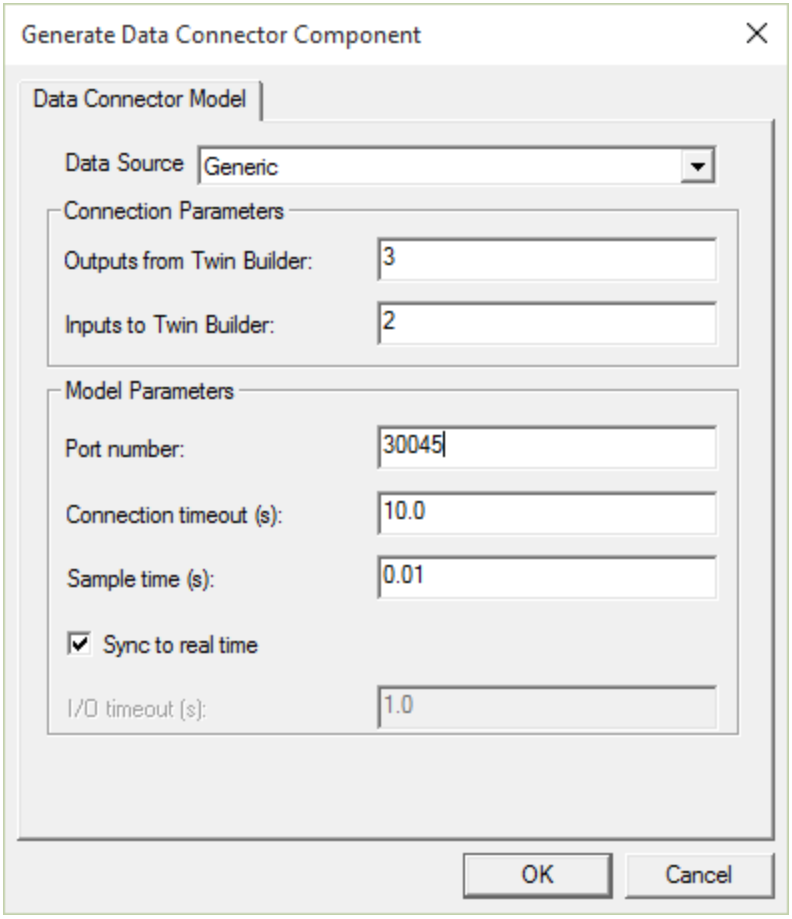
Adding a Data Connector Component in Twin Builder

Adding a Data Connector Component in Twin Builder

Use the following procedure to add a Data Connector component to a Twin Builder design.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

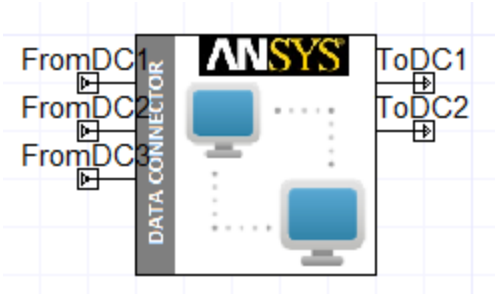
1. On the Twin Builder main menu, select **Twin Builder** > **SubCircuit** > **Add Data Connector Component**. The **Generate Data Connector Component** dialog box appears.



2. Specify the appropriate **Data Source** for the component.
3. In the **Connection Parameters** panel, enter the number of **Outputs from Twin Builder** and the number of **Inputs to Twin Builder**.
4. In the **Model Parameters** panel, enter the following:
   - **Port number** – TCP port number for listening to the client.
   - **Connection timeout (s)** – Timeout for establishing a connection with the client.
   - **Sample time (s)** – Communication sample time.
   - **Sync to real time** – Every sample point is synchronized to real-time clock. Uses non-blocking communication.
   - **I/O timeout (s)** – Read/write timeout in blocking mode. Activated when you clear the **Sync to real time** check box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. When finished, click **OK** to create the Data Connector component and close the dialog box.
6. Place the component on the schematic sheet.



7. After placing the component on a sheet, its **Properties** dialog box may open for configuration of the component.
8. Run Twin Builder analyses as desired by ensuring the client application on the remote machine can run and communicate with this component.

Related Topics

- Program Requirement for Data Connector Component
- Data Connector Component
- Overview of Data Connector

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

14 - Netlist Editor

A netlist is an SML textual representation of a design that describes the design's model instances, connections, subcircuits, ports, as well as instructions for data recording and analysis information. Each component instance in a schematic provides an entry for the netlist based on the model or netlist line chosen for that particular instance (using SML syntax and netlist property expansion rules). Twin Builder creates a netlist from the schematic design and prepares a binary version of the netlist (an .**smt** file). The simulator then uses the binary file to perform the simulation.

> **Note:**
>
> A Twin Builder netlist does not contain information about the physical layout of a circuit.

You can use the Netlist Editor to view the derived netlist without making any changes. The Netlist Editor display is colored to provide easy identification of variables, values, comments, bookmarks, and other elements of the netlist. You can print the netlisting.

Related Topics

Viewing Netlists in Schematic Designs

Netlist Editor Operation

Text Editor Options

Printing

Viewing Netlists in Twin Builder

In Twin Builder designs, the schematic defines the circuit; the derived netlist can be viewed for reference only. Internally, the simulator generates a netlist from the schematic, and updates it from the schematic before performing any simulation.

To view the netlist:

- Select **Twin Builder** > **Browse Netlist**.
- In any simulator design, right-click the name of the design in the **Project** window, and select **Browse Netlist**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Warning:

By default unconnected component pins (terminals) will generate errors (displayed in the **Message Manager** pane) when you generate a netlist. Simulation is not possible until the errors are corrected.

You can change this default component behavior using the component editor so that either no action is taken when unconnected terminals are netlisted, or unconnected terminals are grounded.

Note:

- The netlist in a Twin Builder schematic design is for reference only – so that you can view what will be sent to the simulator.
- The netlist should not be modified within the Netlist Editor view. While the Netlist Editor lets you modify the netlist, and will ask if you want to save the changes, any modifications you make will be overwritten the next time the netlist is generated from the schematic.

Netlist Editor Operation

The Netlist Editor conventions for text entry, selection, modification, and deletion are common to text editors such as MS Notepad[tm].

In addition, use the Netlist Editor commands to employ bookmarks to mark significant locations in the netlist, to perform text searches, to search for text and replace it with new text, and to go to a numbered line in the netlist. You can access Netlist Editor commands from the Netlist tab, from the **Edit** menu, and with keyboard shortcuts.

Netlist Editor Tab

Edit Menu

Using Bookmarks

Searching for Text

Searching for and Replacing Text

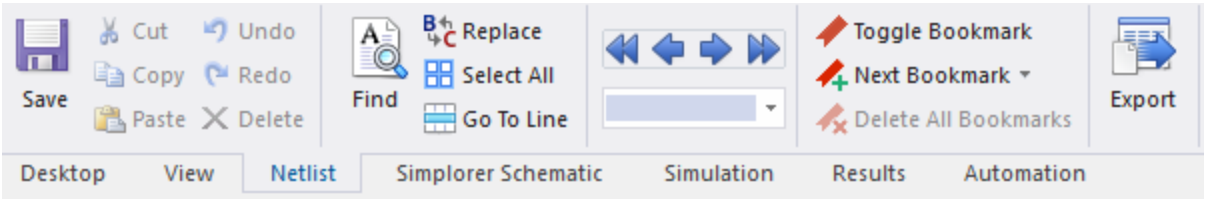
Going to a Numbered Line

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Netlist Editor Tab

> **Note:**
>
> This tab is also used with the C, VHDL, and SML model editors.

When the Netlist Editor is active, the **Netlist** tab provides access to related commands:



The table below summarizes the Netlist Editor commands.

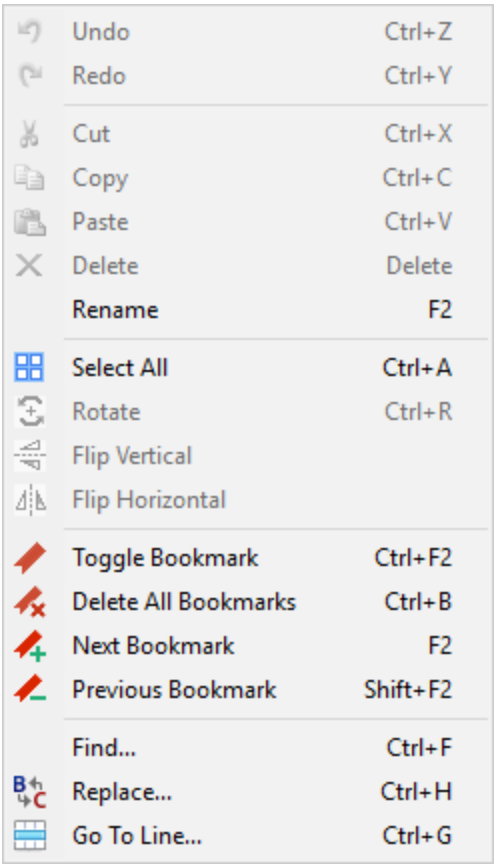
Command Function	Icon	Description
Toggle Bookmark		Insert a bookmark at the active cursor position, or turn off an existing bookmark.
Next Bookmark		Move the cursor to the next bookmark in the text.
Previous Bookmark		Move the cursor to the previous bookmark in the text.
Delete All Bookmarks		Delete all bookmarks in the text.
Search Text		Enter a text string to find in the netlist. Click the drop-down list to display previous text strings.
Search Forward		Search forward in the listing for text entered in the search text field.
Search Backward		Search backward in the listing for text entered in the search text field.
Forward Search, Case-sensitive		Search forward in the listing for text entered in the search text field with case-sensitive control.
Backward Search, Case-		Search backward in the listing for text entered in the search text field with case-sensitive control.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Command Function	Icon	Description
sensitive		

Edit Menu

When the Netlist Editor is the active window, the **Edit** menu includes Netlist Editor commands.



The menu also identifies keystroke shortcuts that may be used in place of the mouse-oriented command methods.

Replace and **Go To Line** are not available on the **Netlist Editor** tab. These commands must be executed from the **Edit** menu or with a keystroke shortcut.

Using Bookmarks

A bookmark is a colored overlay that highlights a line in the netlist that is significant to you. Once created, **Netlist Editor** bookmarks allow quick access to text sections.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Inserting a Bookmark

Deleting a Bookmark

Deleting All Bookmarks

Moving Forward to the Next Bookmark

Moving Backward to the Previous Bookmark

Inserting a Bookmark

To insert a bookmark, click the line to be marked and do one of the following:

- Click the **Toggle Bookmark** icon on the **Netlist** tab.
- Select **Edit** > **Toggle Bookmark**.
- Press Ctrl+F2.

A colored overlay indicates that the line is bookmarked.

The color used for bookmarks can be specified as a **Netlist Editor** option. See ***Netlist & Script Editor Settings*** for details.

Deleting a Bookmark

To delete a single bookmark, click inside a bookmarked line and do one of the following:

- Click the **Toggle Bookmark** icon on the **Netlist** tab.
- Select **Edit** > **Toggle Bookmark**.
- Press Ctrl+F2.

The overlay disappears.

Deleting All Bookmarks

To delete all bookmarks in the netlist, do one of the following:

- Click the **Delete All Bookmarks** icon on the **Netlist** tab.
- Select **Edit** > **Delete All Bookmarks**.
- Press **Ctrl+B**.

Moving Forward to the Next Bookmark

To move the cursor forward to the next bookmark, do one of the following:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Click the **Next Bookmark** icon on the **Netlist** tab.
- Select **Edit** > **Next Bookmark**.
- Press F2.

If no bookmarks have been inserted, **Next Bookmark** does not move the cursor. If the cursor is on or past the last bookmark in the netlist, **Next Bookmark** moves the cursor to the first bookmark in the netlist.

Moving Backward to the Previous Bookmark

To move the cursor backward to the previous bookmark, do one of the following:

- Click the **Previous Bookmark** icon on the **Netlist** tab.
- Select **Edit** > **Previous Bookmark**.
- Press Shift+F2.

If you haven't inserted any bookmarks, does not move the cursor. If the cursor is on or before the first bookmark in the netlist, moves the cursor to the last bookmark in the netlist.

Searching the Netlist

The Netlist Editor can locate an item of interest in the netlist. Specify the search parameters via the **Netlist** tab icons, from the **Edit** menu, or with a keystroke shortcut.

The topics for this section include:

Searching from the Tab

Searching from the Edit Menu or by Keyboard Shortcut

Searching from the tab

To search for an item of text in the netlist, enter the search text in the search text field on the **Netlist** tab:

- To search forward for the text, using case-insensitive matching, click the **Search forward** icon on the **Netlist** tab. If the cursor is past the location of the search text, the search wraps from the beginning of the netlist.
- To search backward for the text, using case-insensitive matching, click the **Search backward** icon on the **Netlist** tab. If the cursor is ahead of the location of the search text, the search wraps from the end of the netlist.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- To search forward for the text, using case-sensitive matching, click the **Forward search case-sensitive** icon on the **Netlist** tab. If the cursor is past the location of the search text, the search wraps from the beginning of the netlist.
- To search backward for the text, using case-sensitive matching, click the **Backward search case-sensitive** icon on the **Netlist** tab. If the cursor is ahead of the location of the search text, the search wraps from the end of the netlist.

If any search fails to find the target text, a message notifies you.

Searching from the Edit Menu or by Keyboard Shortcut

Another way to start a text search is to select **Edit** > **Find** or pressing Ctrl+F. Both operations open the **Find** dialog box.

1. Enter the search text in the **Find what** field.
2. Select the **Direction** for the search. **Up** searches from the cursor toward the front of the netlist, wrapping from the front to the back if the text is not found. **Down** searches from the cursor toward the back of the netlist, wrapping from back to front is the text is not found.
3. Toggle **Match case** to enable case-sensitive text matching.
4. Click **Find Next** to start the search.

If any search fails to find the target text, Twin Builder notifies you with a message.

Searching for and Replacing Text

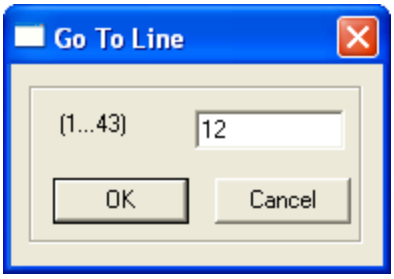
To search for a text item and replace it with new text, select **Edit** > **Replace** or press Ctrl+H. Both operations open the **Replace** dialog box.

1. Enter the search text in the **Find what** field.
2. Enter the replacement text in the **Replace with** field.
3. Select the **Direction** for the search. **Up** searches from the cursor toward the front of the netlist, wrapping from the front to the back if the text is not found. **Down** searches from the cursor toward the back of the netlist, wrapping from back to front is the text is not found.
4. Toggle **Match case** to enable case-sensitive text matching.
5. To find and replace text item-by-item, click **Find Next** to start the search. When the next instance of the search text is highlighted, click **Replace** to replace the search text with the replacement text, or click **Find Next** again to leave the instance unchanged. Repeat this step until all replacements have been made.
6. To replace all instances of the search text with the replacement text globally, click **Replace All**.
7. When all replacements have been made, click **Cancel** to close the **Replace** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Going to a Numbered Line

Errors and warnings from the netlist parser refer to line numbers in the netlist. Line numbers display in the Netlist Editor by default, and you can move the cursor to a line by specifying its number. To move the cursor to a numbered line, select **Edit** > **Go To Line** or press Ctrl+G. Both operations open the **Go To Line** dialog box.



1. The parenthesized display shows the range of available line numbers in the current netlist.
2. Enter the number of the line and click **OK**. The cursor moves to the beginning of that line.

> **Note:**
>
> **Go To Line** is also used by various other Twin Builder text editors such as the VHDL-AMS, Package, Script, C-Model, SPICE, and SML editors.

Exporting a Netlist or Script

To export the netlist representation of a design to a file (to use as the basis of a new model, for example):

1. Select **Netlist** > **Export**. The **Netlist Export** dialog box opens.
2. In the **Save in** field, browse to the directory where the netlist file is to be saved,
3. In the **File name** field, enter the complete name of the file, including the extension if any.
4. Click **Save** to save the netlist file and close the dialog box, or click **Cancel** to cancel the export operation without saving anything.

> **Note:**
>
> The above procedure can also be used to export a script composed in the Script Editor, in which case the menu item and dialog box names will be **Script** and **Script Editor**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Netlist & Script Editor Settings

You can set text options for the Netlist & Script Editor.

1. Click **Tools** > **Options** > **General Options**, and expand **Netlist & Script Editor**.
2. Click **Text Editor** to make changes to the following:
   - Select **Display Line Numbers** to display line numbers in the editing window.
   - Click **Bookmark** to open a **Color** selection dialog box in which you can select a color for bookmarks.
   - The font panel displays the current font name selection and font size. Some non-editable sample text displays the currently selected font and size. To change the font, select the desired font name from the **Name** drop-down list. To change the font point size, edit the **Size** text field.
3. Make the desired selections and click **OK**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15 - Ansys Workbench Integration Overview

Ansys Workbench combines the strength of its core product solvers with the project management tools necessary to manage project workflow. In Ansys Workbench, analyses are built as *systems*, which can then be combined into a *project*. The project is driven by a schematic workflow that manages the connections between the systems.

From the schematic, you can interact with applications (called workspaces) that are native to Ansys Workbench and display within the Ansys Workbench interface. Native workspaces include: Project Schematic, Engineering Data, and Design Exploration (Parameters and Design Points).

You can also launch applications that are data-integrated with Ansys Workbench, meaning the application's interface remains separate, but the data from the application communicates with the native Ansys Workbench data. Thus, data can be passed back and forth between any Ansys Electromagnetics product on a Workbench Project Schematic and any supported Ansys or Ansys Electromagnetics desktop product. Depending on the application, data integration can include basic actions such as saving projects, as well as more complex actions such as the coupling of Ansys Electromagnetics product variables to Workbench Design Exploration parameters.

Data-integrated applications include the following Ansys Electromagnetics products: Ansys Electronics Desktop (HFSS and Q3D Extractor), Maxwell/RMxprt, and Simplorer.

> **Note:**
>
> For detailed information on working with Ansys Workbench, see the Workbench documentation.

Integrating Ansys Electromagnetics Suite Products with Ansys Workbench

Workbench Data Integration Overview

Ansys Electromagnetics - Ansys Multiphysics Coupling

Ansys Electromagnetics CAD Integration Through Workbench

Ansys Electromagnetics to Ansys Geometry Transfer

User Defined Model (UDM) for Ansys WB Integration

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Integrating Ansys Electromagnetics Suite Products with Ansys Workbench

You can integrate Ansys Electromagnetics products with Ansys Workbench either during or after product installation.

Integration with Ansys 2024 R2 during Ansys Electromagnetics Product Installation

Integrate Ansys Electromagnetics products with Ansys 2024 R2 during installation as follows:

1. If you have not already installed Ansys 2024 R2, do so before proceeding. See the Installation Guide for the Ansys Electromagnetics Suite.
2. Proceed through the installation dialog boxes.
3. When the **Integration with Ansys 2024 R2** dialog box appears, click **Yes** to have the installer integrate the Ansys Electromagnetics product with Ansys 2024 R2.

> **Note:**
>
> You can also perform this step after installation.

4. Complete the product installation and exit the installer.

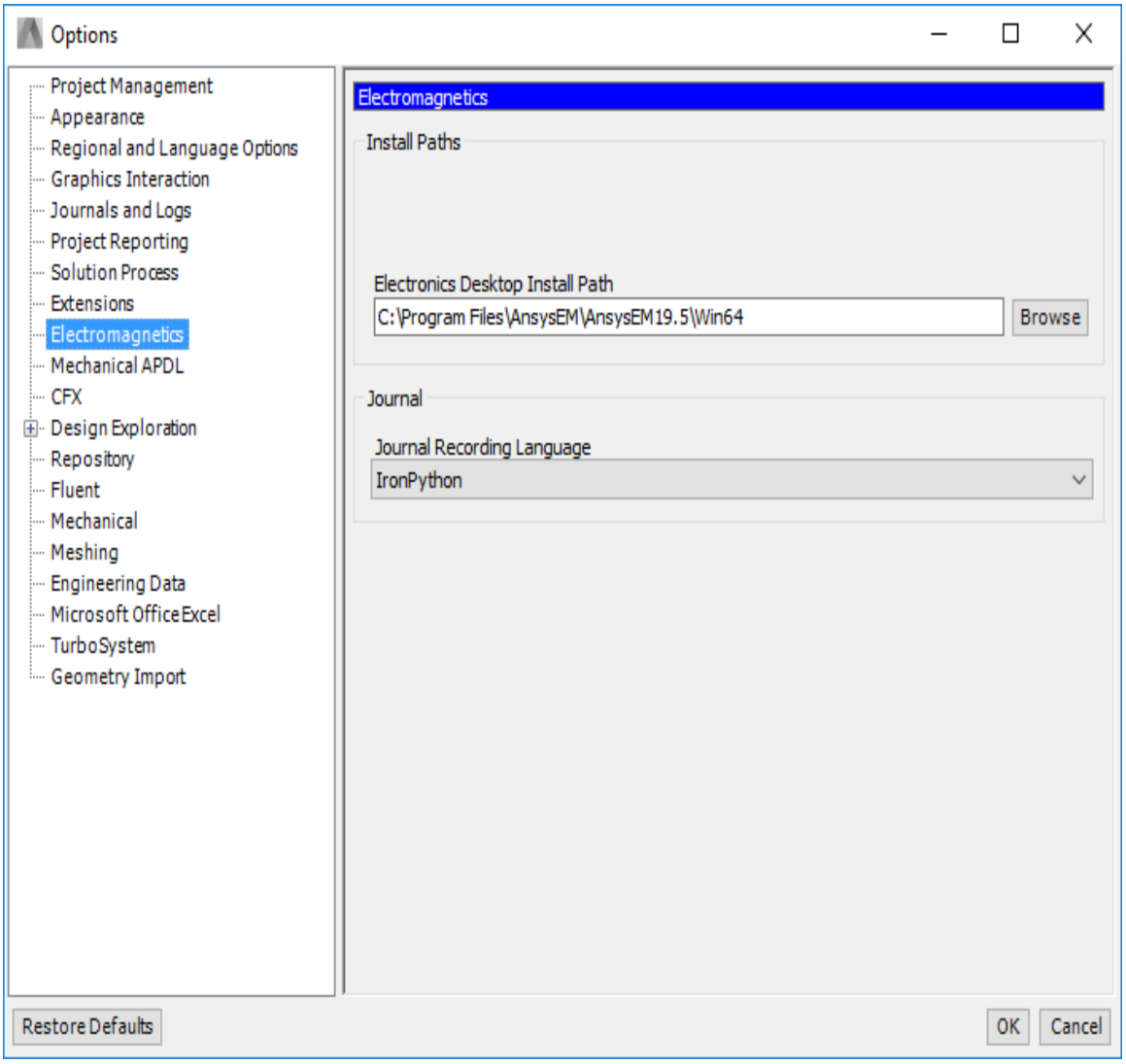
Integration with Ansys 2024 R2 after Ansys Electromagnetics Product Installation

To integrate Ansys Electromagnetics Suite with Ansys 2024 R2 after installation:

1. Select **All Programs** > **Ansys EM Suite 2024 R2** > **Modify Integration with Ansys 2024 R2** from the **Start** menu.
2. In the **Integration with Ansys 2024 R2** dialog box, click **Yes** and click **OK** to complete the integration process.

   You can confirm that Ansys 2024 R2 is “aware” of the Ansys Electromagnetics integrated application via the Ansys Workbench **Options** dialog box, which shows the path to the integrated Electronics Desktop application.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

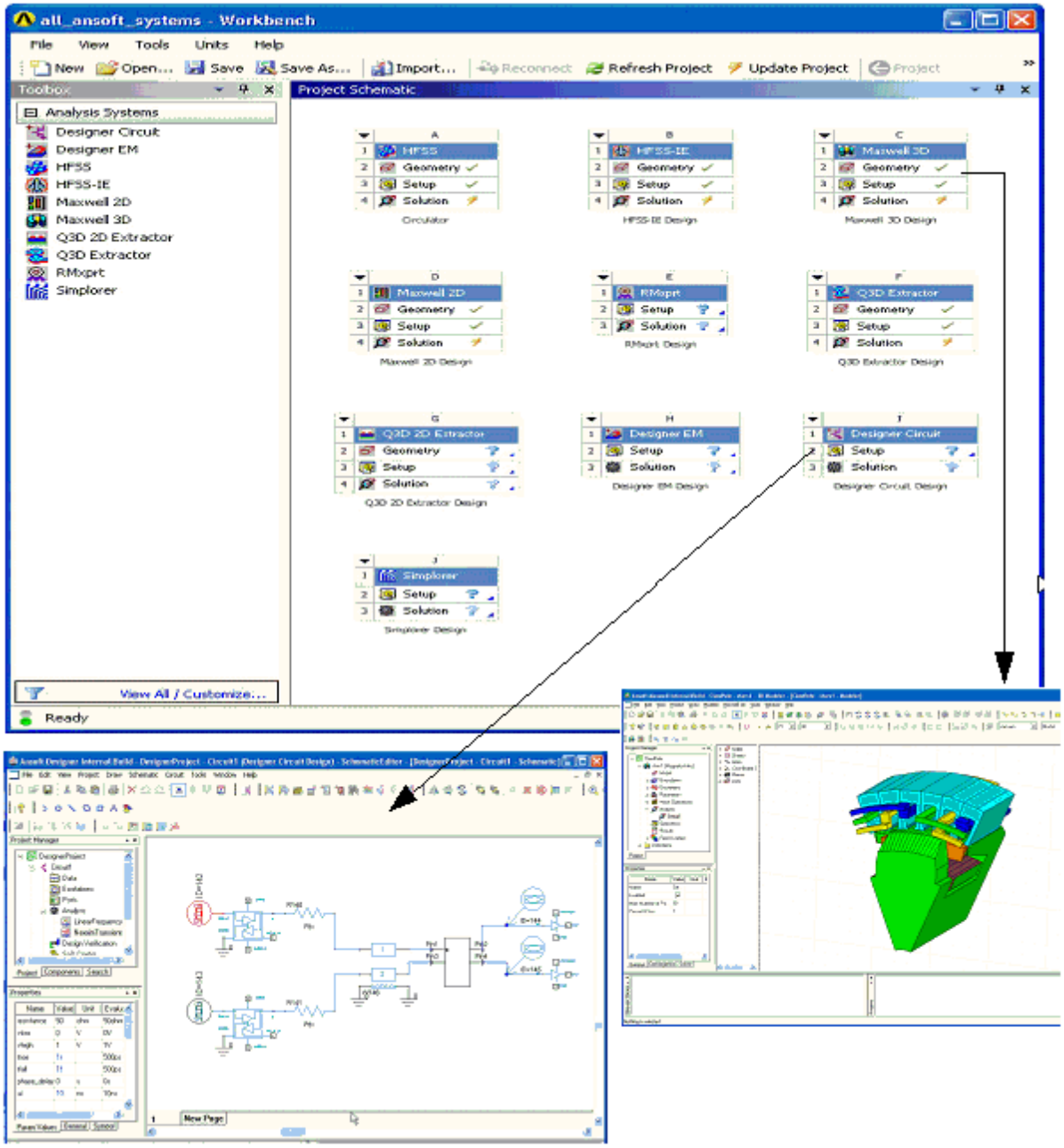


The Journal Recording Language option lets you specify the default journal recording language. The default language is IronPython. You can change the recording language to VB. Changes in the recording language option apply to any Ansys Electromagnetics system launched after the change. Any open Ansys Electromagnetics desktops will continue recording in the language that was in effect when they were launched.

Workbench Data Integration Overview

Ansys Electromagnetics data-integrated applications reside on a Workbench **Project Schematic** as shown below.

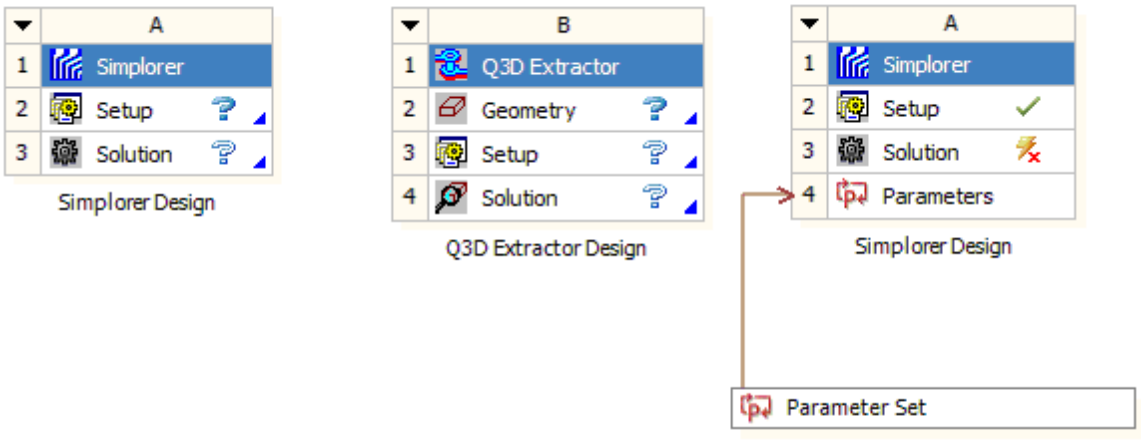
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Objects—such as instances of Ansys Electromagnetics projects—placed on a Workbench Project Schematic are referred to as *systems*. Ansys Electromagnetics circuit/system products Rmxprt, Designer Circuit, and Simplorer appear on Workbench Project Schematics as systems with two “cells” – **Setup** and **Solution**. Ansys Electromagnetics field products HFSS, Maxwell, and Q3D Extractor add an additional **Geometry** cell. If you invoke Ansys **DesignXplorer** to use variables for refining a design, a **Parameters** cell appears with a link to the associated

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Workbench **Parameter Set**. See the Ansys Workbench help for details on working with systems, cells, and parameter sets.



Ansys Electromagnetics desktop products integrate with Workbench commands, services, and DesignXplorer in a similar manner. Here are some of the ways in which Ansys Electromagnetics products integrate with Workbench:

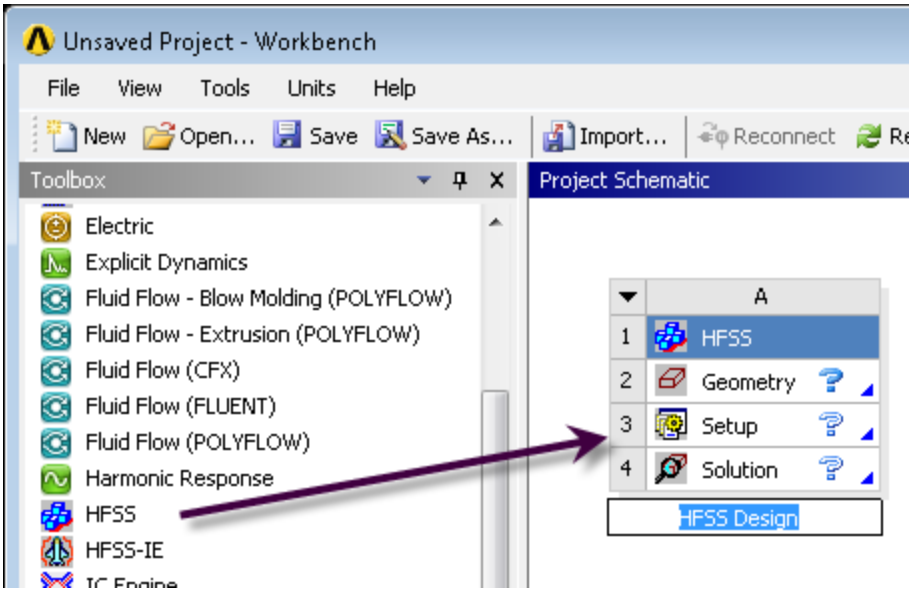
- Adding new analysis systems
- Importing existing desktop projects
- Editing models
- Analyzing models
- Performing parameter studies
- Scripting

In addition to these major features, Workbench also lets you archive, save, back up, duplicate, and delete Ansys Electromagnetics projects used in a Workbench project. Progress information and messages from integrated Ansys Electromagnetics projects also display in Workbench.

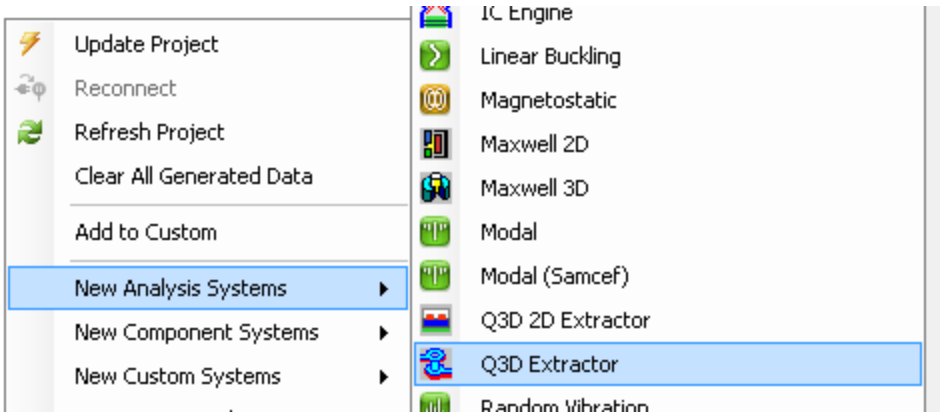
Adding New Ansys Electromagnetics Analysis Systems

Add a new Ansys Electromagnetics Analysis System to a Workbench Project Schematic either by dragging and dropping it from the Toolbox:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



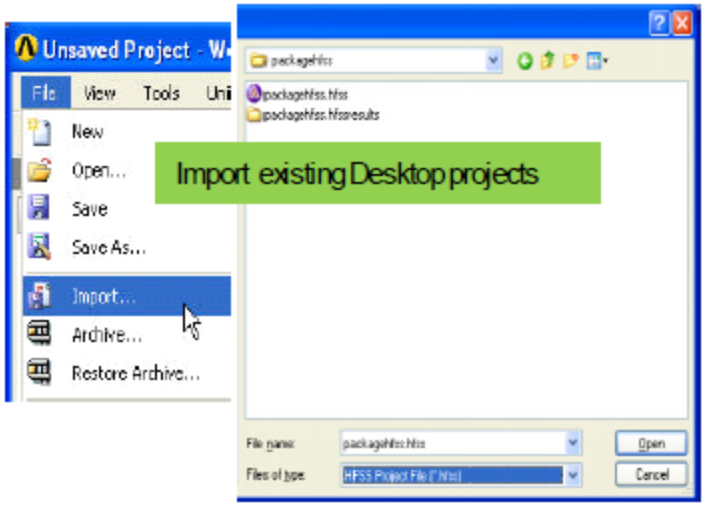
or by selecting it from the context menu in the Workbench Project Schematic window:



Importing Ansys Electromagnetics Projects into Ansys Workbench

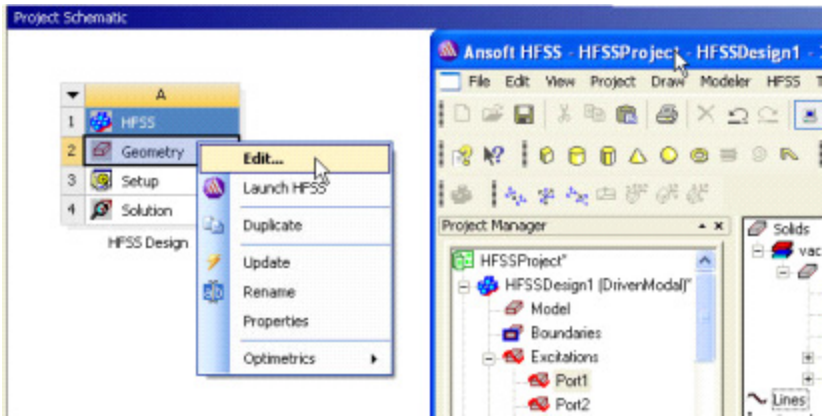
When you import existing Ansys Electromagnetics desktop projects to a Workbench Project Schematic, a copy of the Ansys Electromagnetics project appears in the Workbench Project folder. The original Ansys Electromagnetics project remains intact.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Editing Ansys Electromagnetics Models in Workbench

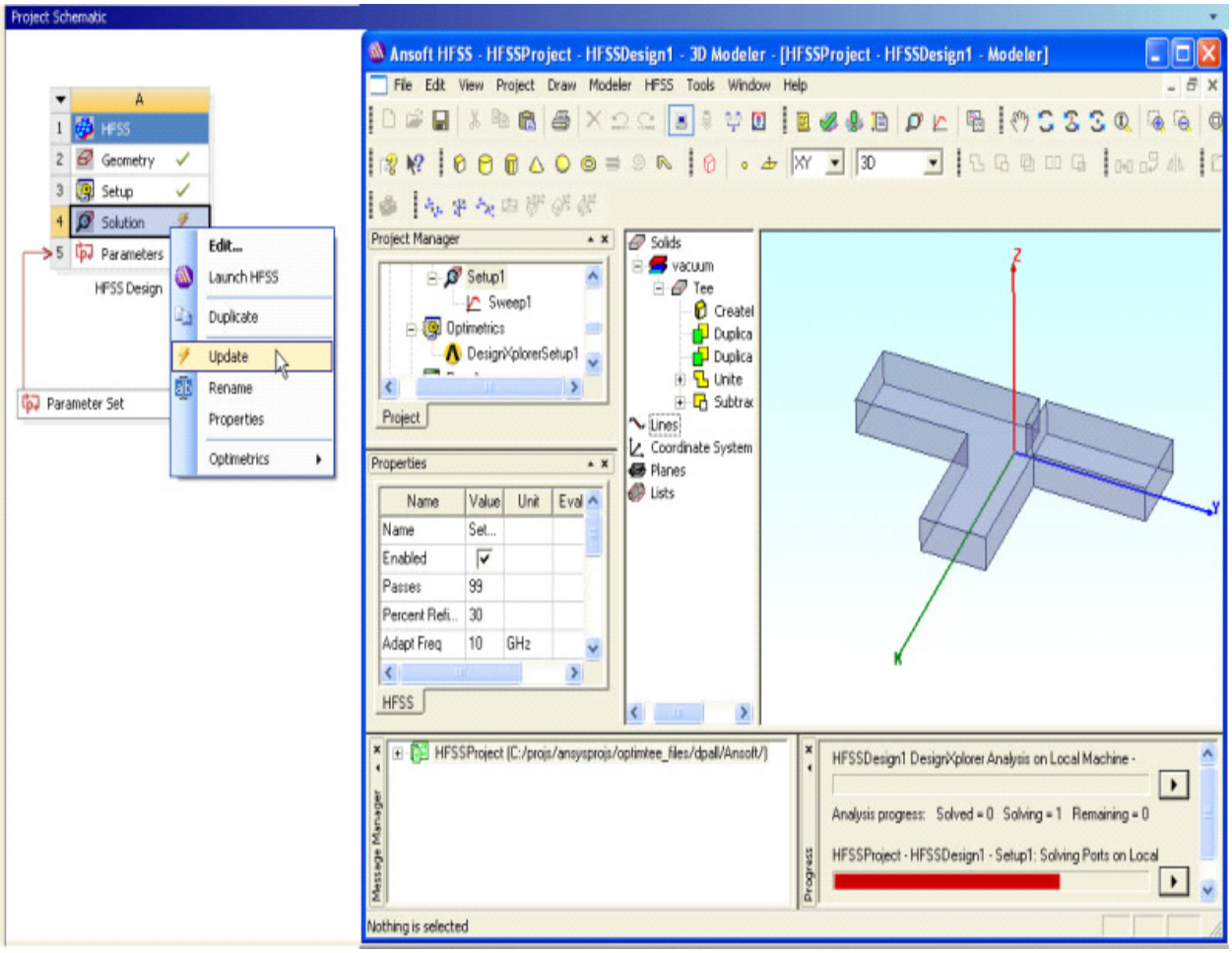
You can edit various properties and parameters (geometry, setup, solution, and so on) of the Ansys Electromagnetics project. Either right-click the project in Workbench and select **Edit**, or double-click the project. Doing so launches the Ansys Electromagnetics desktop application and loads the project so you can set up your project in a familiar desktop environment. Changes made to the Ansys Electromagnetics project are saved to the project instance in the Workbench project folder.



Analyzing Ansys Electromagnetics Models in Workbench

Select **Update** in Workbench to run analyses in an integrated Ansys Electromagnetics project. Progress information also appears in Workbench.

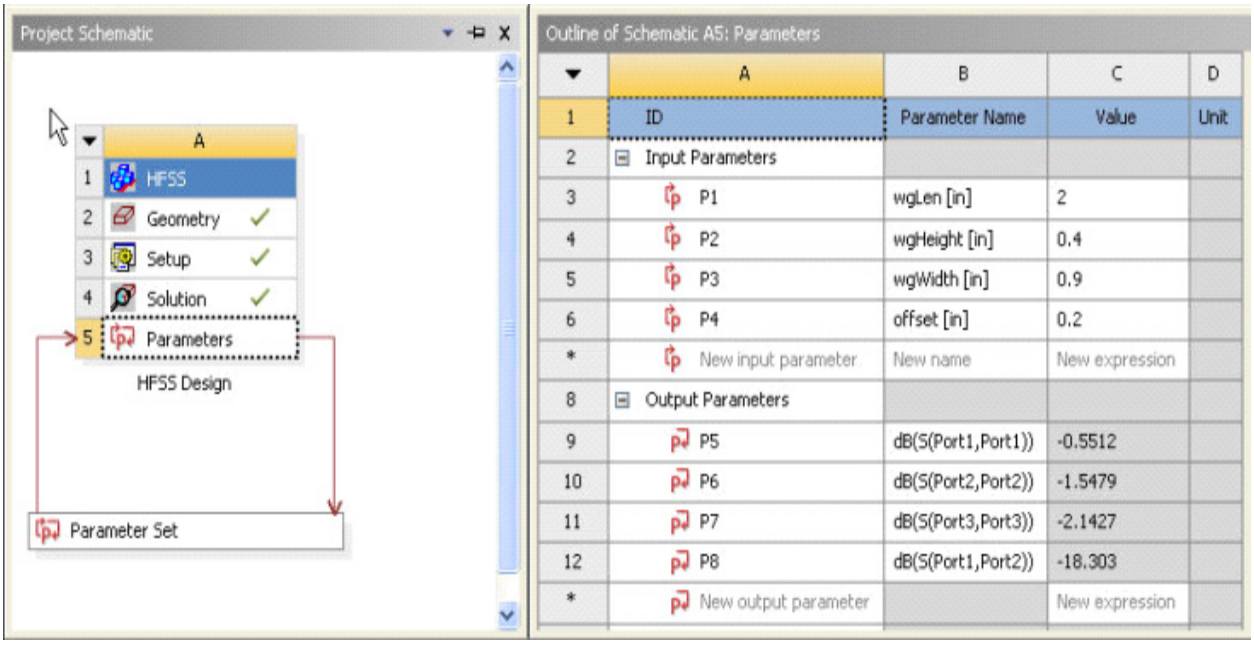
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Performing Parameter Studies in Workbench

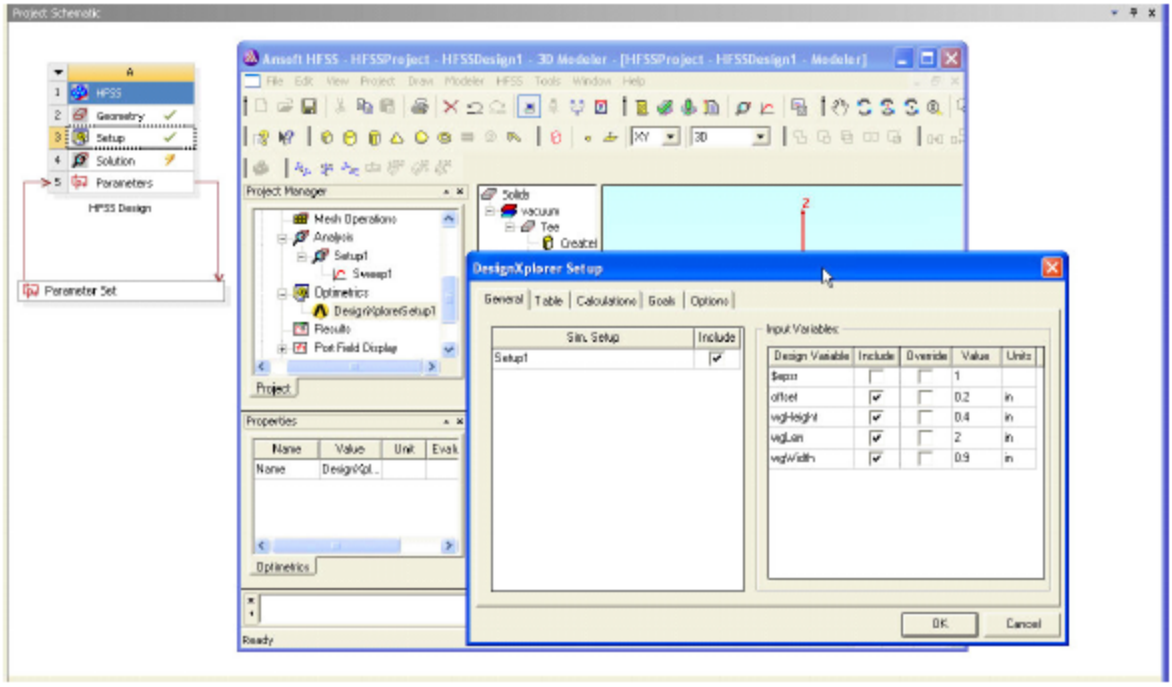
Workbench **Parameter Sets** let you change parameter values and units of measure, or add new parameters. Parameter data is passed back to the Ansys Electromagnetics application for updated analyses.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



	A	B	C	D
1	ID	Parameter Name	Value	Unit
2	Input Parameters			
3	P1	wgLen [in]	2	
4	P2	wgHeight [in]	0.4	
5	P3	wgWidth [in]	0.9	
6	P4	offset [in]	0.2	
*	New input parameter	New name	New expression	
8	Output Parameters			
9	P5	dB(S(Port1,Port1))	-0.5512	
10	P6	dB(S(Port2,Port2))	-1.5479	
11	P7	dB(S(Port3,Port3))	-2.1427	
12	P8	dB(S(Port1,Port2))	-18.303	
*	New output parameter		New expression	

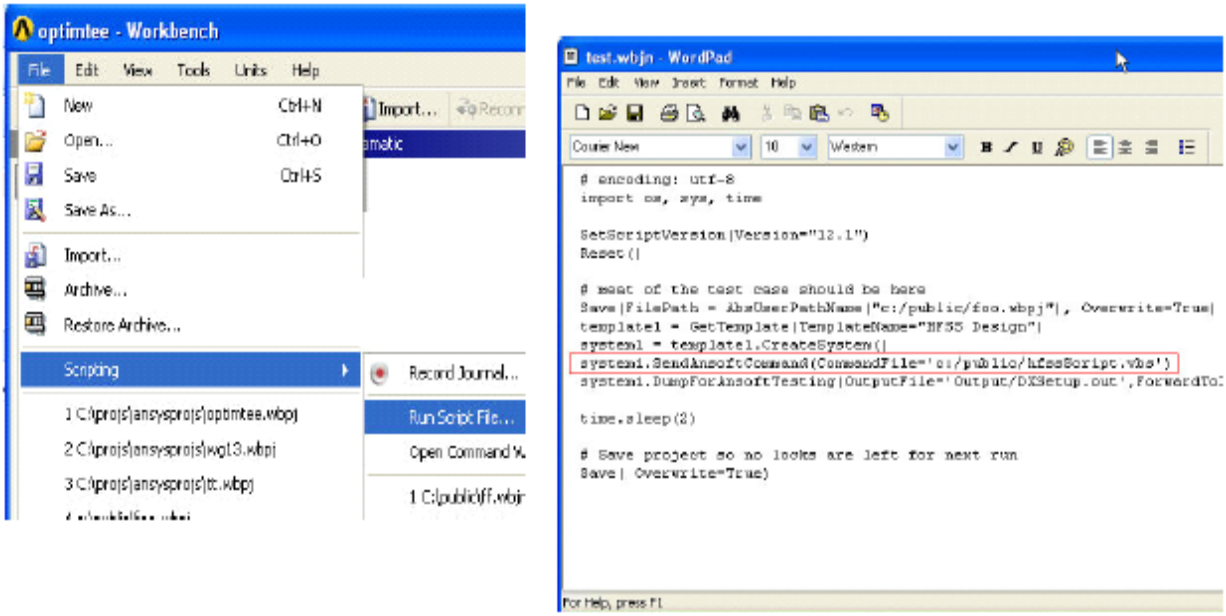
Parameters from the Ansys Electromagnetics project are exposed to Workbench through the **DesignXplorer** setup. The Ansys Electromagnetics system's cell status on the Workbench project updates as you make changes in the Ansys Electromagnetics application desktop.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Scripting in Workbench

You can record or play back scripts that include Ansys Electromagnetics projects via Workbench.



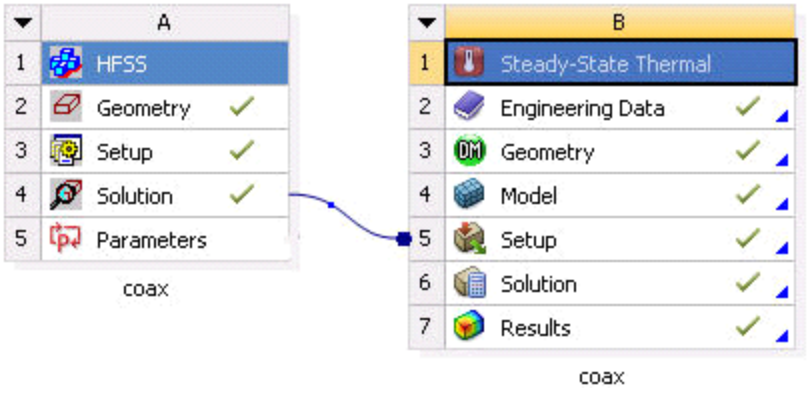
Ansys Electromagnetics - Ansys Multiphysics Coupling

Data integration provides improved multiphysics workflow between Ansys Electromagnetics designs and Ansys applications such as Mechanical and Thermal. Coupling is provided through project schematic links. Heat losses and force data transfer to Ansys Mechanical; there is no need to export/import XML files. Select **Refresh** in Workbench to transfer edits made in Ansys Electromagnetics applications to the Ansys multiphysics application. Workbench commands also enable easier automation of iterative coupling of thermal feedback. The following sections provide some examples of multiphysics coupling.

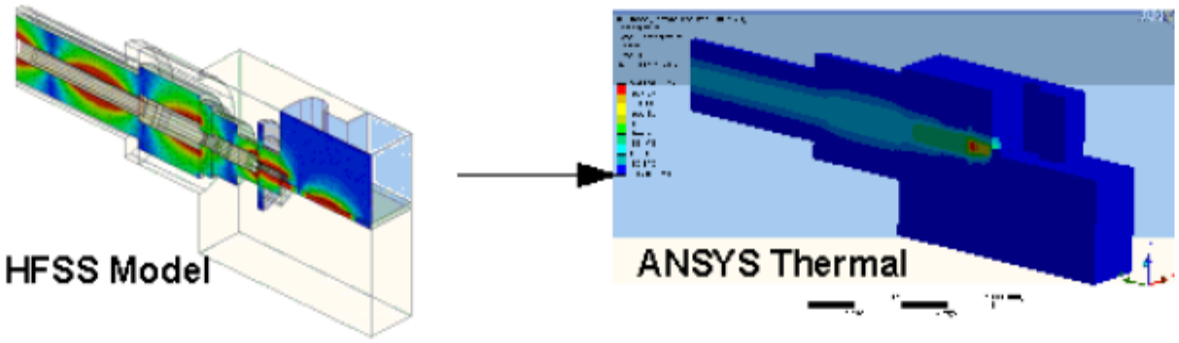
Multiphysics Coupling on Workbench with Ansys Thermal

Using data integration, HFSS, Maxwell, and Q3D Extractor provide heat losses (heat generation and heat flux) to Ansys Thermal. Note how the HFSS design is linked visually to Ansys Thermal on the Workbench project schematic.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



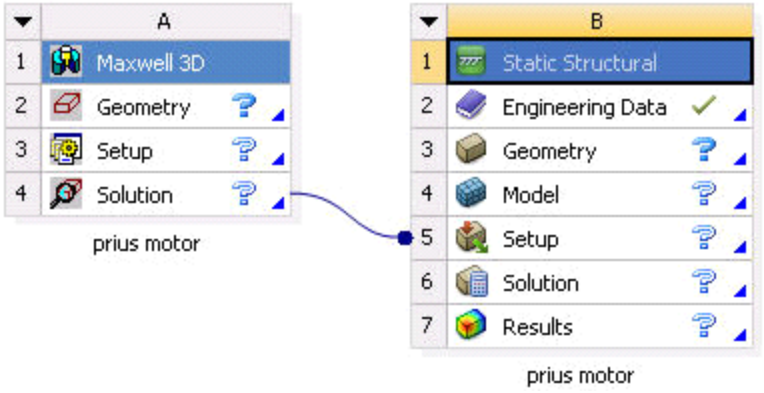
In this example, HFSS coax model olution provides heat loss data as a thermal load to the Ansys Thermal Setup. The resulting analysis shows a thermal hotspot, providing you with the information needed to adjust the design's material to fix the problem.



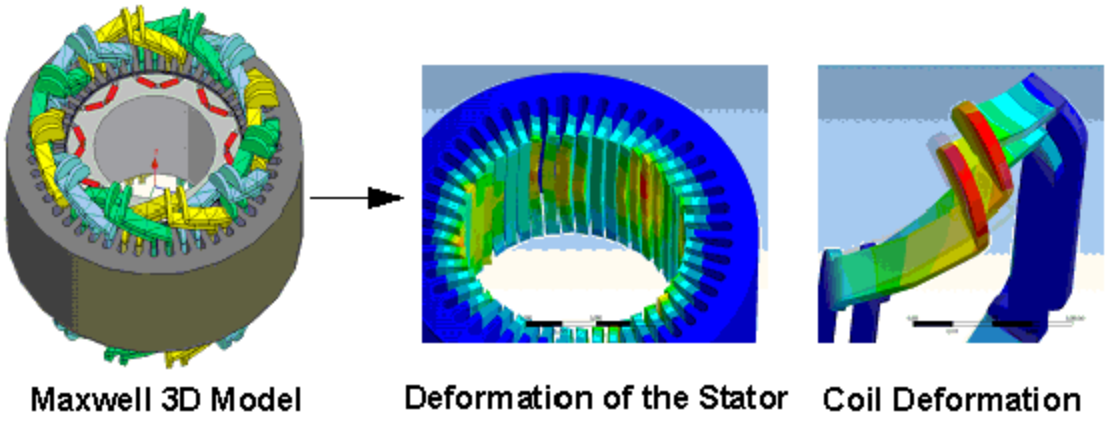
Multiphysics Coupling on Workbench with Ansys Structural

Using data integration, Maxwell 2D and Maxwell 3D provide forces to Ansys Structural.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



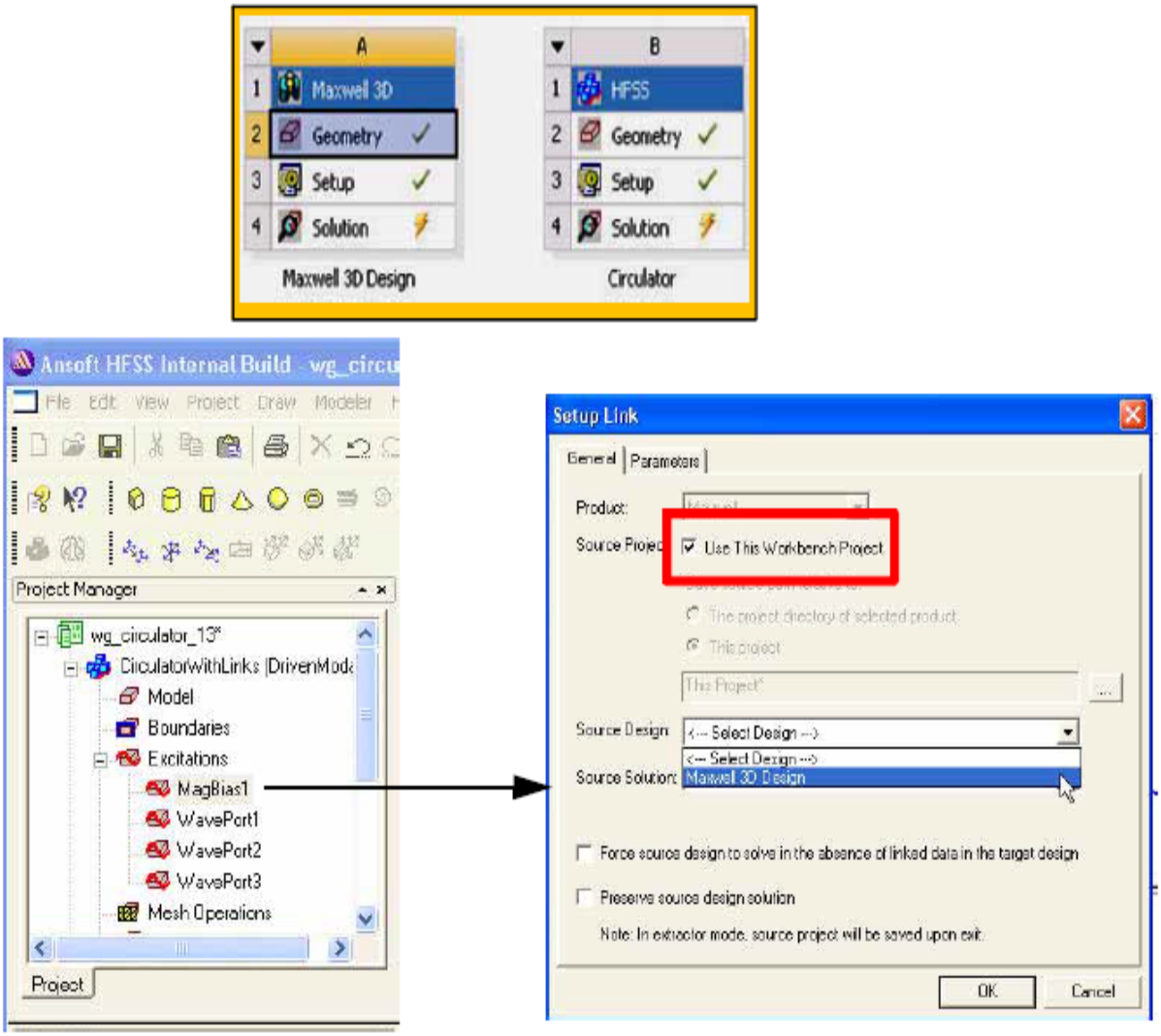
In this example, the Maxwell 3D electromagnetic force density **Solution** is used as the load in Ansys Structural to determine how these forces deform the motor's stator and coils.



Multiphysics Coupling between Ansys Electromagnetics Field Systems on Workbench

You can set up links between Ansys Electromagnetics field systems that reside on a Workbench project schematic. Linking is set up in the Ansys Electromagnetics application as shown in the following example.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



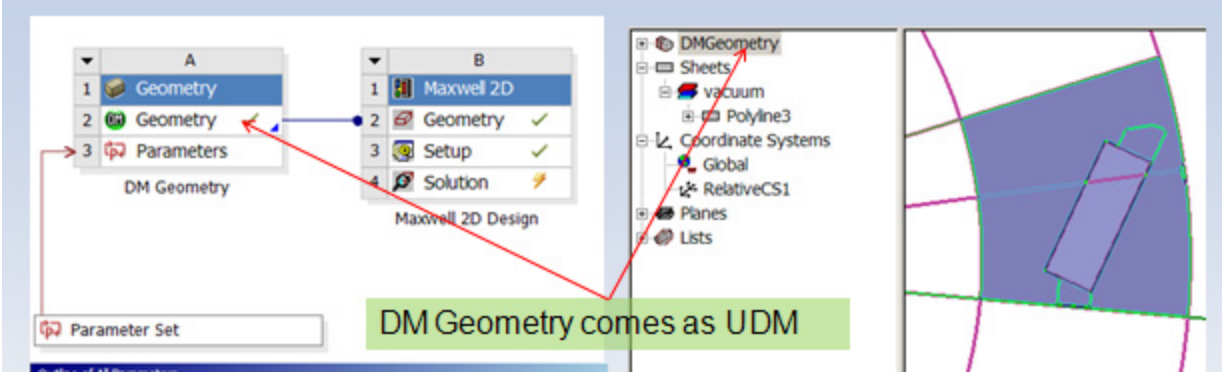
Ansys Electromagnetics CAD Integration through Workbench

Ansys Electromagnetics CAD integration is a Workbench feature available for Ansys Electromagnetics 3D Products HFSS, Maxwell, and Q3D as the Ansys Framework for Electromagnetics package. The feature is available only through Workbench, not with standalone Ansys Electromagnetics products.

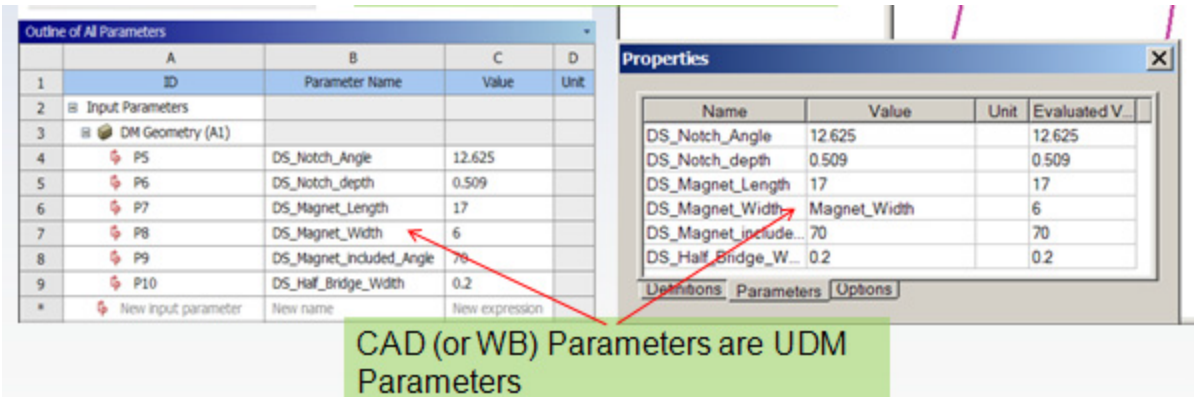
Ansys Electromagnetics CAD integration provides a bi-directional dynamic link through Workbench, which makes it possible to get updated geometry from CAD and to modify CAD parameters in Ansys Electromagnetics products to return updated geometry. It is associative in that IDs persist between the Ansys Electromagnetics model and CAD model during model

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

refresh. The feature is non-associative due to a need to reassign boundaries if modified CAD model is to be used. The process creates a User Defined Model (UDM) for each geometry source.



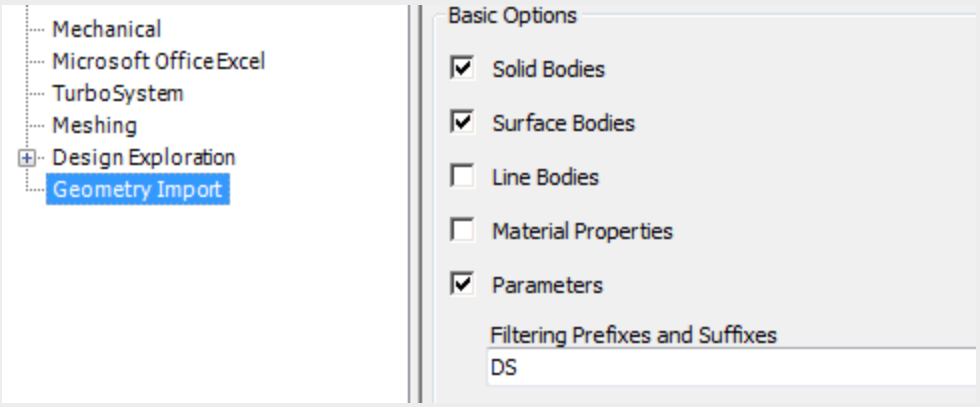
The UDM format makes it possible to exchange parameters.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

The parameters shown in the previous example all have a DS prefix. This is the default for the Workbench **Tools** > **Options** for **Geometry Import.** If you want to import parameters with different prefixes or names, assign an appropriate prefix, or clear the **Filtering Prefixes** field, depending on your needs.



See ***User Defined Model (UDM) for Ansys WB Integration*** for more information.

Ansys Electromagnetics CAD integration makes it possible to consume geometry from multiple upstream source which can be any CAD or Ansys Electromagnetics product. This feature supports direct interfaces with all major CAD systems.

- Creo Parametric
- UG NX
- CATIA V5
- SOLIDWORKS
- Autodesk Inventor
- Ansys Design Modeler (DM)
- Ansys SpaceClaim Direct Modeler (SCDM)

CAD software must be installed on your machine

- Not required on solve nodes

Platform Supported

- Windows 64-bit

See the following sections:

CAD Integration and Geometry Sharing

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Bi-Directional CAD Integration

CAD Integration Model Edits

Multiple Geometry Links for CAD Integration

CAD Integration Functionality

Healing with CAD Integration

Important Geometry Options for CAD Integration

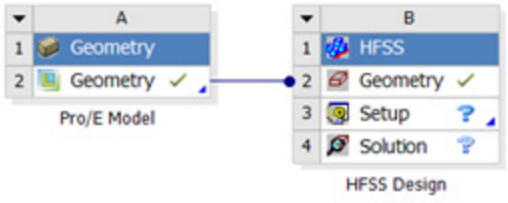
CAD Integration and Geometry Sharing

CAD model comes into the Ansys Electromagnetics design as a UDM.

The input to the Ansys Electromagnetics design from CAD is:

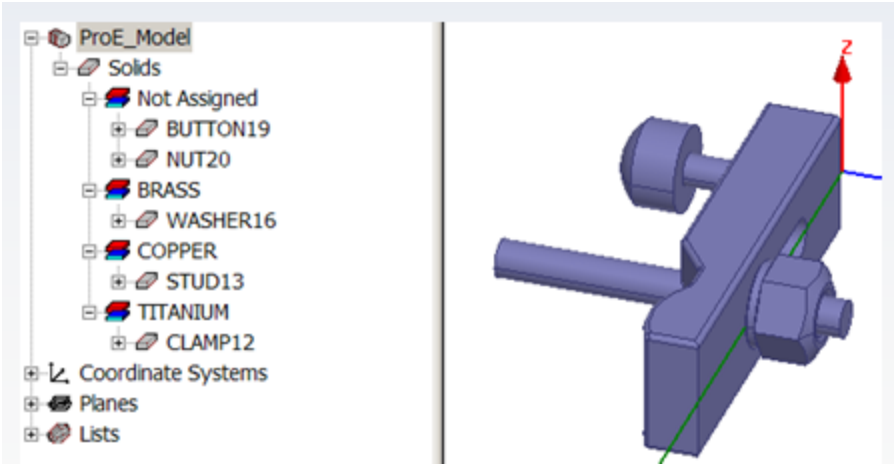
- Geometry/Topology with persistent IDs
- CAD parameters
- Material assignment
- Attributes like name, and color

For example, in Workbench, you can link a Pro/E Model to an HFSS design.



The geometry can then be viewed in HFSS as a UDM.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



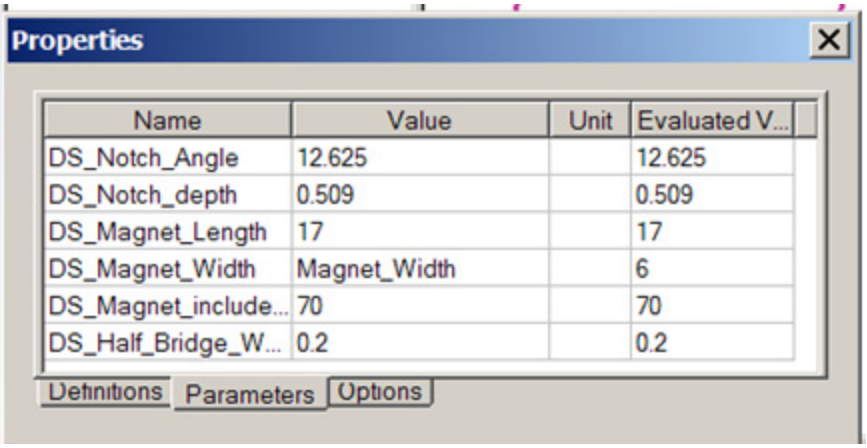
The CAD or WB model parameters appear in the Workbench:

Outline of All Parameters

	A	B	C	D
1	ID	Parameter Name	Value	Unit
2	Input Parameters			
3	DM Geometry (A1)			
4	P5	DS_Notch_Angle	12.625	
5	P6	DS_Notch_depth	0.509	
6	P7	DS_Magnet_Length	17	
7	P8	DS_Magnet_Width	6	
8	P9	DS_Magnet_included_Angle	70	
9	P10	DS_Half_Bridge_Width	0.2	
*	New input parameter	New name	New expression	

Through Ansys Electromagnetics CAD Integration, the linked UDM includes the same parameters.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Related Topics

Ansys Electromagnetics CAD Integration Through Workbench

Bi-Directional CAD Integration

CAD Integration Model Edits

Multiple Geometry Links for CAD Integration

CAD Integration Functionality

Healing with CAD Integration

Important Geometry Options for CAD Integration

CAD Integration Functionality

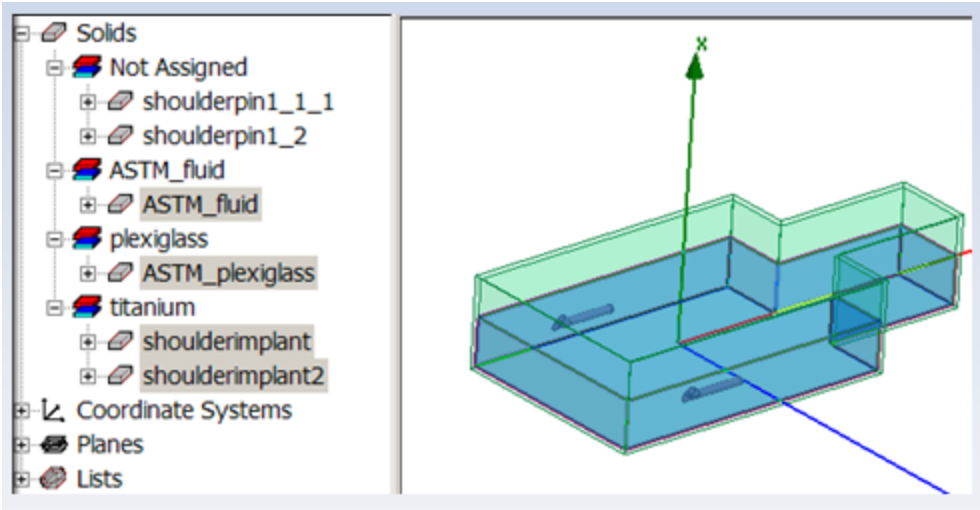
- WB Update Project
- WB Update All Design Points
- DX analysis

For explicit subtraction:

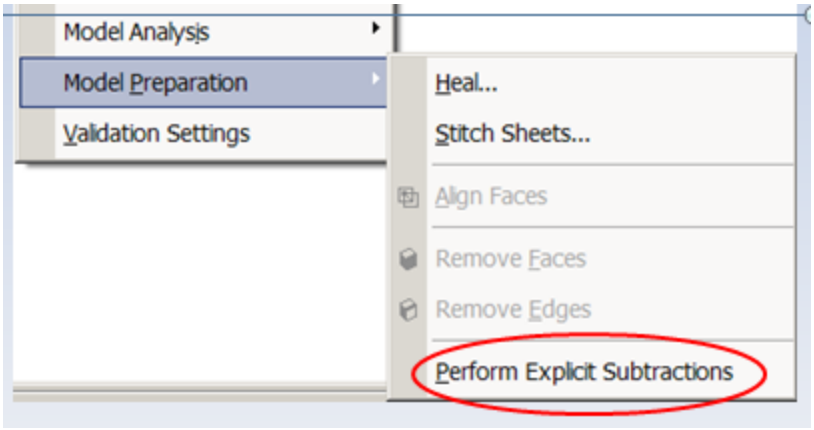
- Ansys Mechanical does not perform implicit subtraction.
- Select **Modeler** > **Model Preparation** > **Perform Explicit Subtraction** to perform explicit subtractions before sending geometry to Ansys.

For example, consider the following model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

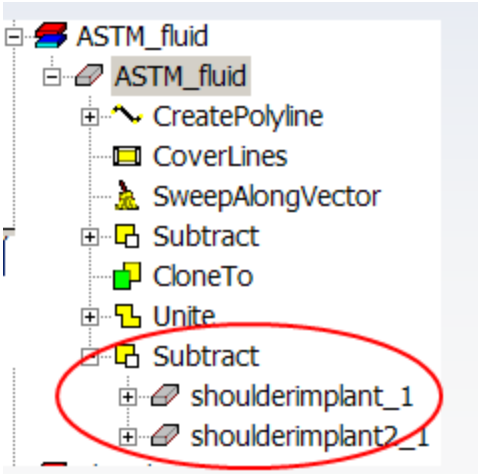


You can perform **Perform Explicit Subtractions**.

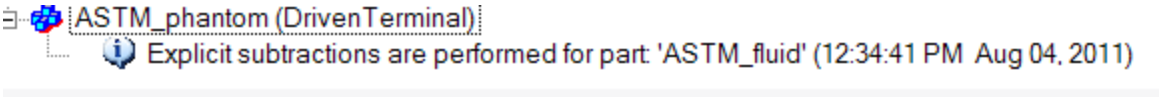


The results appear in the history tree:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The **Message Manager** pane reports this action.



Related Topics

User Defined Model (UDM) for Ansys WB Integration

UDM compared to User Defined Primitives

Insert UDM Command on Draw Menu

UDM Properties

Library of Models for CAD Integration

Ansys Electromagnetics to Ansys Geometry Transfer

CAD Integration Material Assignment Transfer

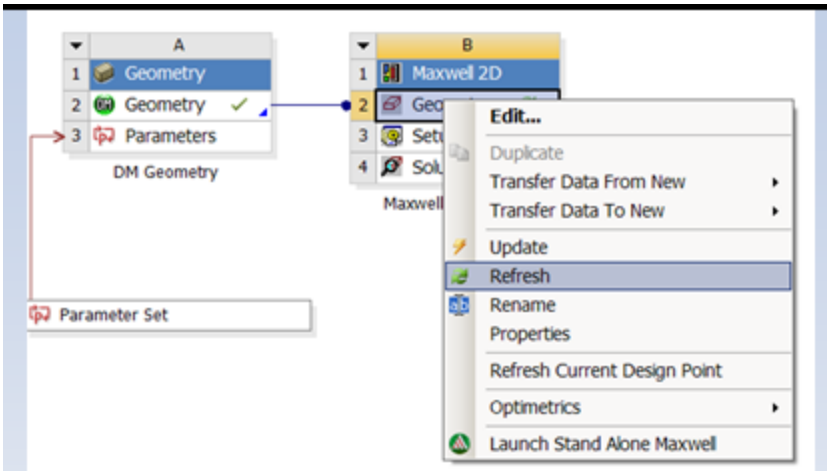
Geometry Transfer through Ansys DesignModeler (DM)

Bi-Directional CAD Integration

Ansys Electromagnetics CAD Integrations uses Refresh (or Generate) CAD Model to pass updates.

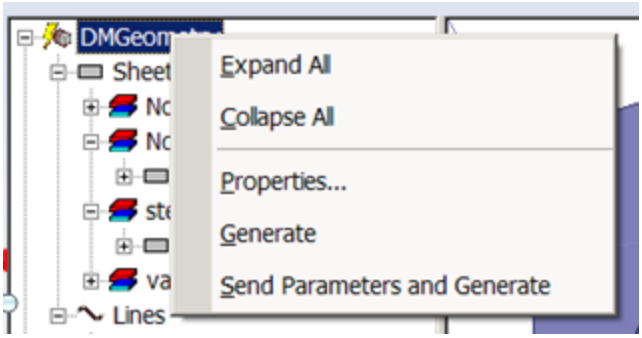
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For example, when you make an edit in a CAD application, you can either run **Refresh** on an Ansys Electromagnetics design Geometry Cell, or **Generate** on the UDM in the Ansys Electromagnetics design window.



Refresh pulls the current state of a CAD model (geometry, parameters, materials etc) and updates the corresponding data in the Ansys Electromagnetics application.

If you edit UDM (CAD) parameters in the Ansys Electromagnetics modeler window, you can run **Send Parameters and Generate** to pass the edited parameters to the linked CAD application and pull corresponding CAD geometry.



Related Topics

Ansys Electromagnetics CAD Integration Through Workbench

CAD Integration and Geometry Sharing

CAD Integration Model Edits

Multiple Geometry Links for CAD Integration

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

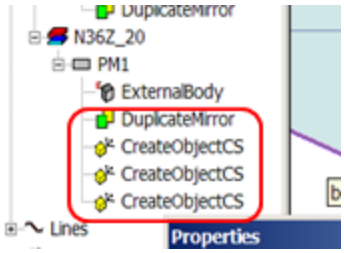
CAD Integration Functionality

Healing with CAD Integration

Important Geometry Options for CAD Integration

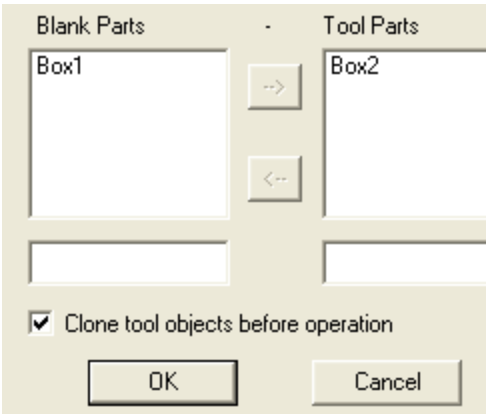
CAD Integration Model Edits

Several modeling operations are allowed on a CAD model in the Ansys Electromagnetics Modeler window. Operations are included in the history tree and retained during model **Refresh**.



The following operations are not allowed:

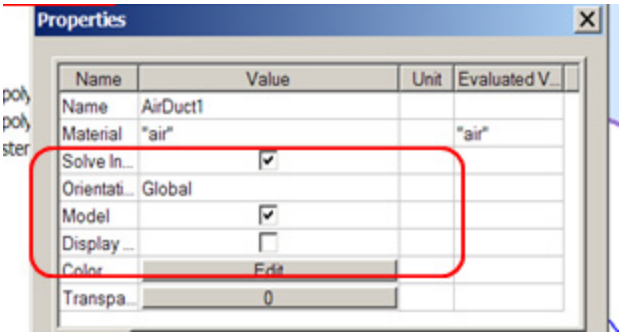
- Non-history tree operations like Heal or defeature.
- Operations which use UDM parts as Tool such as Sweep, or Boolean operations like Split or Unite (but allowed when you select the clone tool option).



The following part attributes can be modified for UDM parts:

- Model/Non-Model flag
- Solve Inside flag
- Part orientation
- Color
- Display Wireframe

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



It is not possible to delete individual parts of a UDM.

Related Topics

Ansys Electromagnetics CAD Integration Through Workbench

CAD Integration and Geometry Sharing

Bi-Directional CAD Integration

Multiple Geometry Links for CAD Integration

CAD Integration Functionality

Healing with CAD Integration

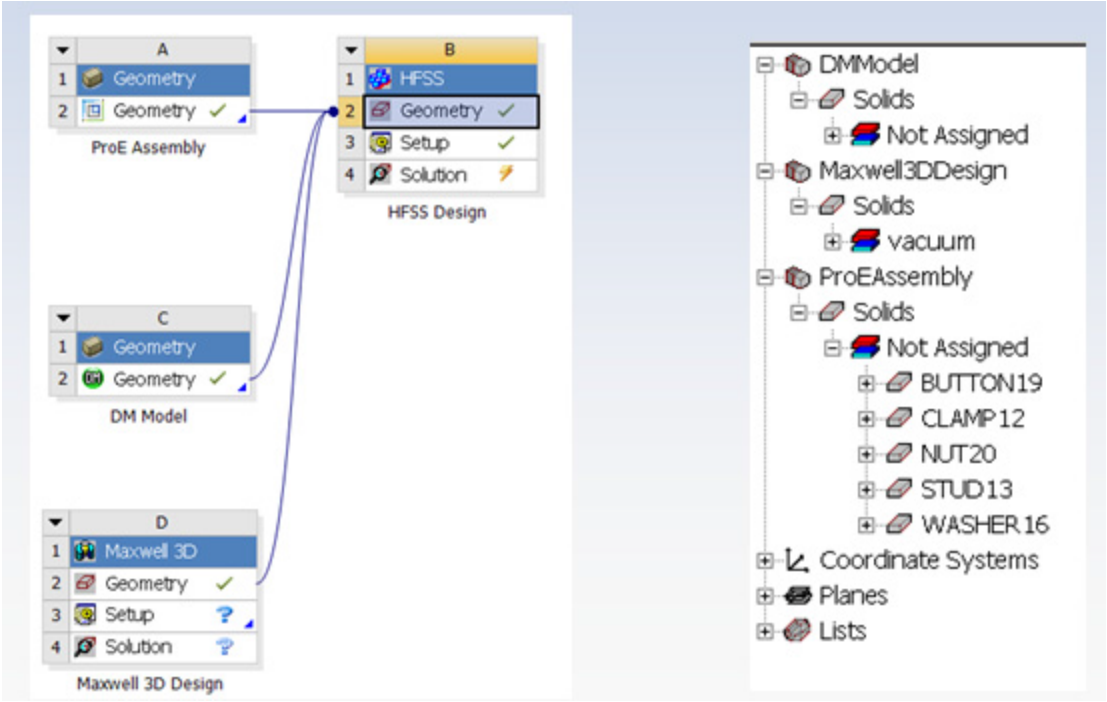
Important Geometry Options for CAD Integration

Multiple Geometry Links for CAD Integration

Use CAD Integration to consume geometry from multiple upstream sources. This creates a UDM for each geometry source. The source can be any CAD or Ansys Electromagnetics product.

For example, the following figure shows DM, Maxwell, and ProE models linked to HFSS in Workbench, and displayed in the HFSS History tree as three UDMs.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Related Topics

Ansys Electromagnetics CAD Integration Through Workbench

CAD Integration and Geometry Sharing

Bi-Directional CAD Integration

CAD Integration Model Edits

CAD Integration Functionality

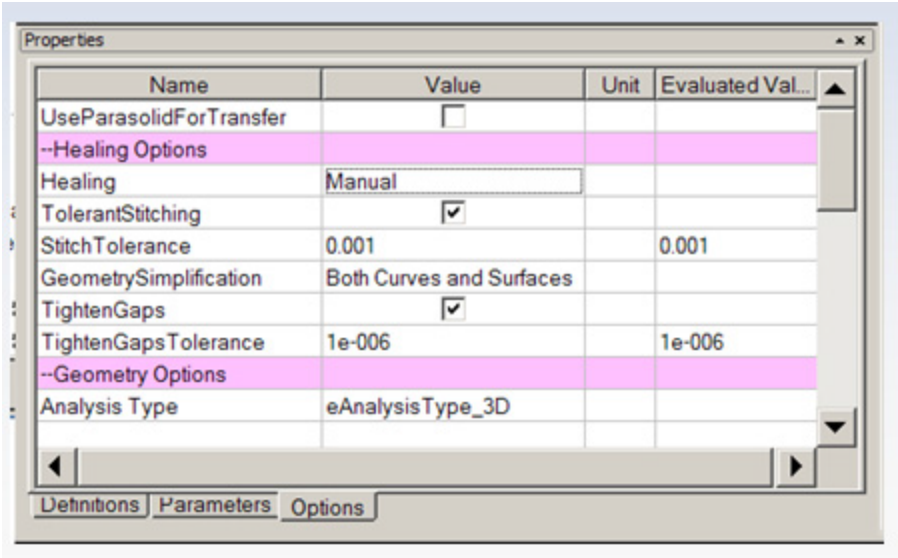
Healing with CAD Integration

Important Geometry Options for CAD Integration

Healing with CAD Integration

It is not possible to use **Heal** in the Ansys Electromagnetics Modeler. Instead similar healing options are available under UDM properties option tab.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The Healing options are **None**, **Auto**, and **Manual**. By default healing is off (**None**) and should be turned on only if required.

Related Topics

Ansys Electromagnetics CAD Integration Through Workbench

CAD Integration and Geometry Sharing

Bi-Directional CAD Integration

CAD Integration Model Edits

Multiple Geometry Links for CAD Integration

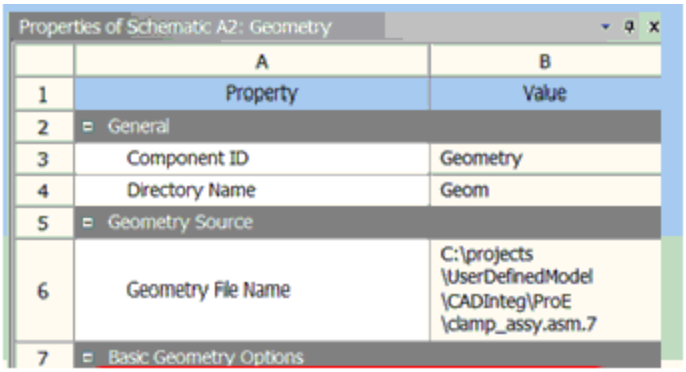
CAD Integration Functionality

Important Geometry Options for CAD Integration

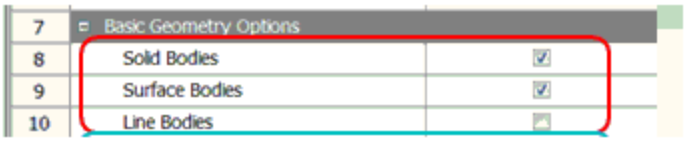
Important Geometry Options for CAD Integration

Select a geometry cell in Workbench to see options in a **Properties** window.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



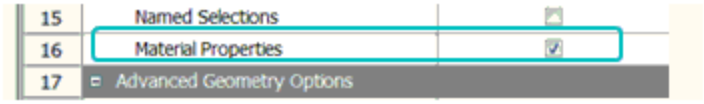
- Control dimension of bodies coming from CAD.



- Make sure **Parameters** is selected and the parameter key (filter) is appropriate to bring CAD parameters.

- Attributes key should be empty or **Color** to bring in CAD colors.

- Material properties must be selected to bring in the material assignment.



- The **Mixed Import Resolution** option resolves parts with mixed dimensions (typically from Pro/E).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

See the Ansys Help for details: Path // CAD Integration // Overview :: 2 // Project Schematic Presence Related to CAD Integration // Geometry Preferences

Related Topics

Ansys Electromagnetics CAD Integration Through Workbench

CAD Integration and Geometry Sharing

Bi-Directional CAD Integration

CAD Integration Model Edits

Multiple Geometry Links for CAD Integration

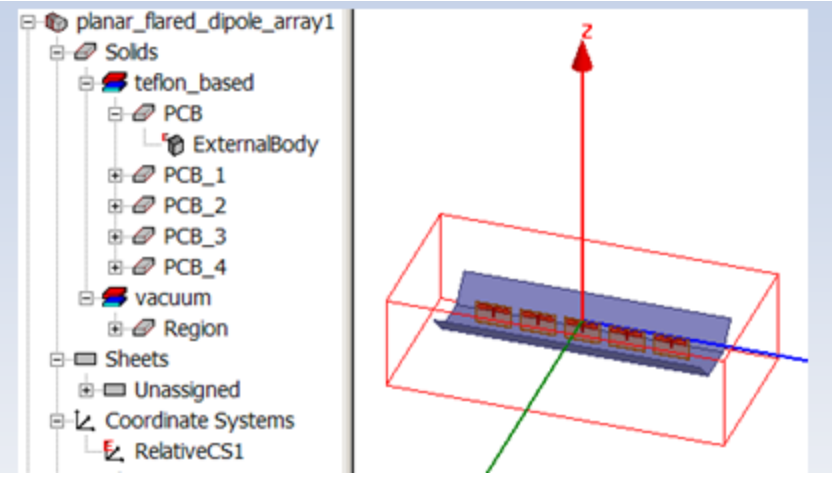
CAD Integration Functionality

Healing with CAD Integration

User Defined Model (UDM) for Ansys WB Integration

A User Defined Model (UDM) is collection of externally defined parts imported into an Ansys Electromagnetics 3D Modeler.

- UDMs include part attributes (like name and color) and material assignment.
- UDMs can have external coordinate systems and corresponding planes.
- UDM parts can be parameterized and manipulated in an Ansys Electromagnetics modeler just like any other part.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

UDMs can represent static geometry models or dynamic links to models of external geometry editors. They can support CAD integration in WorkBench.

UDMs use the same plugin technology as User Defined Parts (UDPs).

See the following sections:

UDM compared to User Defined Primitives

Insert UDM Command on Draw Menu

UDM Properties

Library of Models for CAD Integration

Ansys Electromagnetics to Ansys Geometry Transfer

CAD Integration Material Assignment Transfer

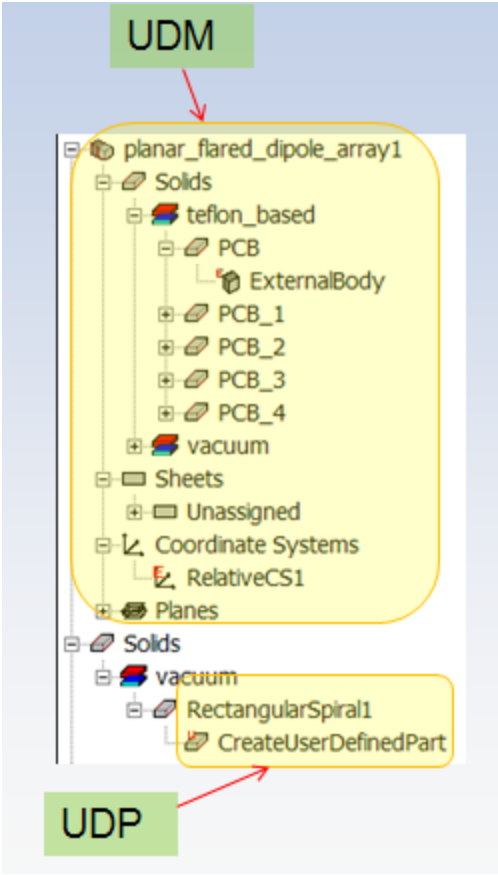
Geometry Transfer through Ansys DesignModeler (DM)

CAD Integration Functionality

UDM Compared to User Defined Primitives

User Defined Models (UDM) resemble User Defined Primitives (UDP).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- Ansys Electromagnetics products can be extended through new UDMs.
- UDM plugins are discovered by searching standard directory paths.
- Plugins for static UDMs can build model using callback interfaces (like create-box, create-cylinder, and subtract) similar to UDPs.
- UDMs run inside the Ansys Electromagnetics application.
- UDMs provide geometry, topology, persistence, and parameters.

In contrast to UDP:

- UDMs provide multiple Parts, CS, and so on.

  UDPs provide primitive operations only for a single part.

- UDMs provide part attributes and material assignment.

  UDPs do not define part attributes or material.

  UDM properties have four tabs: **Definitions**, **Parameters**, **Options**, and **Info**.

Definition tab:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- UDM name.
- Coordinate system used to position UDM.
- External reference to file.

Info tab:

- UDM DLL name, DLL location, version, and so on.

Option tab:

- Lists options, if any.

Related Topics

User Defined Model (UDM) for Ansys WB Integration

Insert UDM Command on Draw Menu

UDM Properties

Library of Models for CAD Integration

Ansys Electromagnetics to Ansys Geometry Transfer

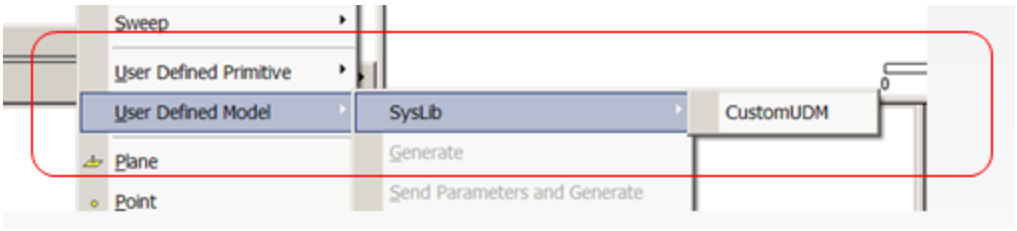
CAD Integration Material Assignment Transfer

Geometry Transfer through Ansys DesignModeler (DM)

CAD Integration Functionality

Insert UDM Command on Draw Menu

To insert a UDM into a design, select **Draw** > **User Defined Model** for the Modeler window.



Related Topics

User Defined Model (UDM) for Ansys WB Integration

UDM compared to User Defined Primitives

UDM Properties

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Library of Models for CAD Integration

Ansys Electromagnetics to Ansys Geometry Transfer

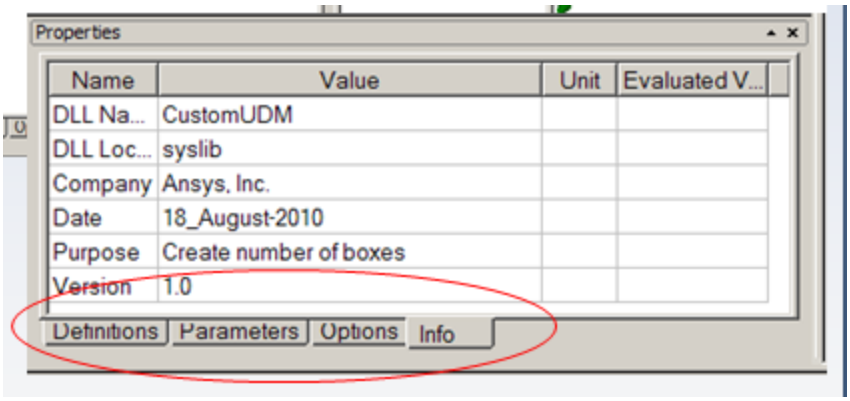
CAD Integration Material Assignment Transfer

Geometry Transfer through Ansys DesignModeler (DM)

CAD Integration Functionality

UDM Properties

UDM properties have four tabs: **Definitions**, **Parameters**, **Options**, and **Info**.



Definition tab:

- UDM name.
- Coordinate system used to position UDM.
- May have external reference to file.

Info tab:

- UDM DLL name, DLL location, version, and so on.

Option tab:

- Lists options, if any.

Related Topics

UDM Parameters

UDM Part Edits

User Defined Model (UDM) for Ansys WB Integration

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

UDM compared to User Defined Primitives

Insert UDM Command on Draw Menu

Library of Models for CAD Integration

Ansys Electromagnetics to Ansys Geometry Transfer

CAD Integration Material Assignment Transfer

Geometry Transfer through Ansys DesignModeler (DM)

CAD Integration Functionality

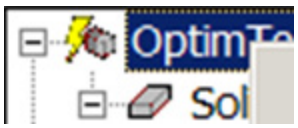
UDM Parameters

UDM geometry can be manipulated through its parameters.

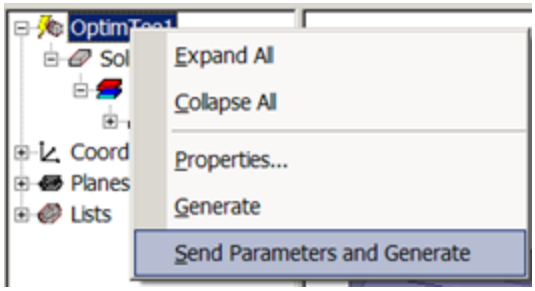
- Can be mapped to design or project variables for animation and parametric analysis.
- IDs persist (allowing to retain boundaries) during parameter edit.

UDM geometry is not dynamically updated upon parameter edits.

- UDM shows a lightning bolt icon by the model name when parameters are edited.



- Run **Send Parameters and Generate** to synchronize parameters with geometry.



Related Topics

UDM Properties

UDM Part Edits

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

UDM Part Edits

Several modeling operations are allowed on UDM parts.

- Operations will be part of the history tree and retained during model refresh.

The following operations are not allowed:

- Non-history tree operations like healing and defeature.
- Operations which use UDM parts as tool, such as sweep or Boolean (but allowed when using the clone tool option).

You can modify these part attributes for UDM parts:

- Model/Non-model flag
- Solve Inside flag
- Part orientation

Related Topics

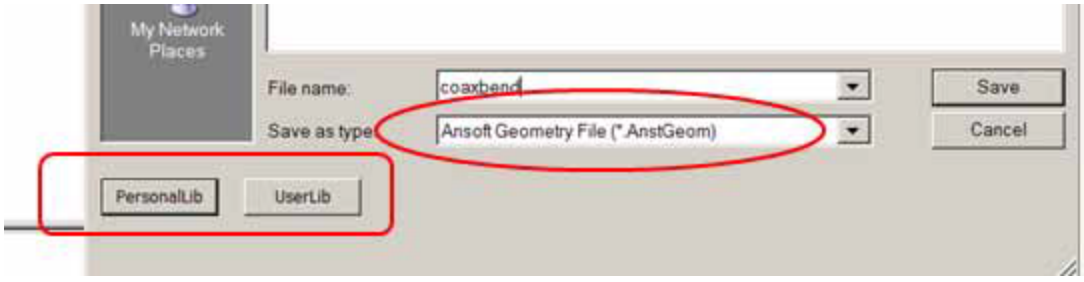
UDM Properties

UDM Parameters

Library of Models for CAD Integration

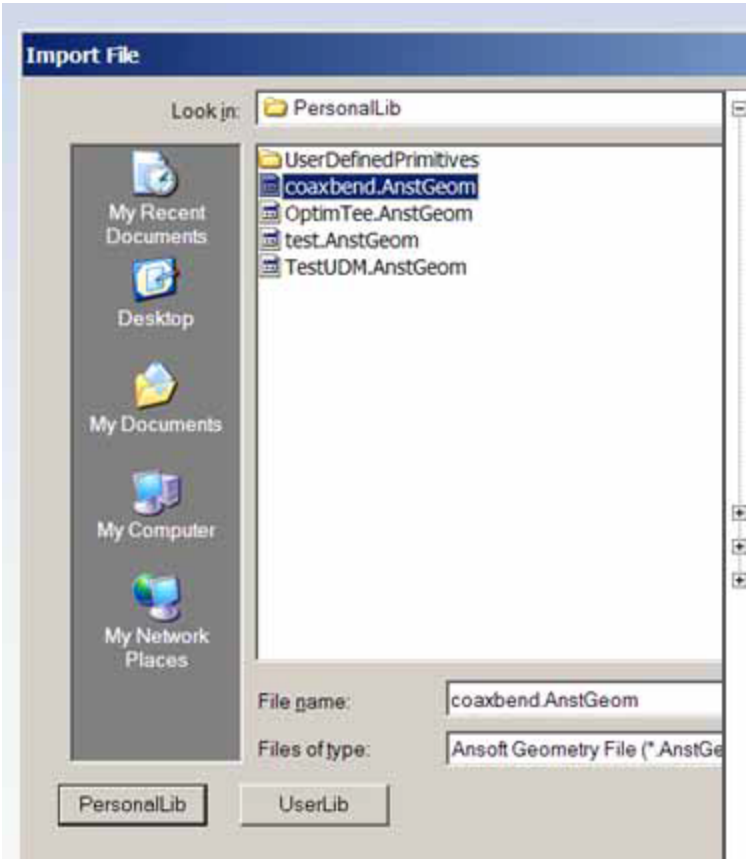
UDM technology allows library of models.

- Any Ansys Electromagnetics model can be exported as **Ansoft Geometry File**.

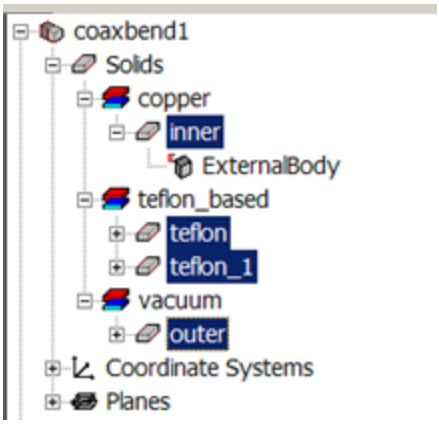


- An Ansoft geometry file can be imported back as a UDM.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- Any design/project variables associated with models are brought in as UDM parameters.

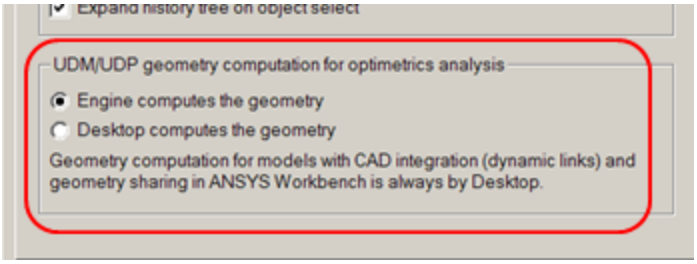


Geometry computation for UDMs (and also UDPs) can be specified in the Modeler options as either on:

- Engine side (default). Requires deployment of UDMs on each node.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Desktop side. UDMs need not be deployed on each engine. The Desktop will be busy during parametric analysis.



Related Topics

User Defined Model (UDM) for Ansys WB Integration

UDM compared to User Defined Primitives

Insert UDM Command on Draw Menu

UDM Properties

Ansys Electromagnetics to Ansys Geometry Transfer

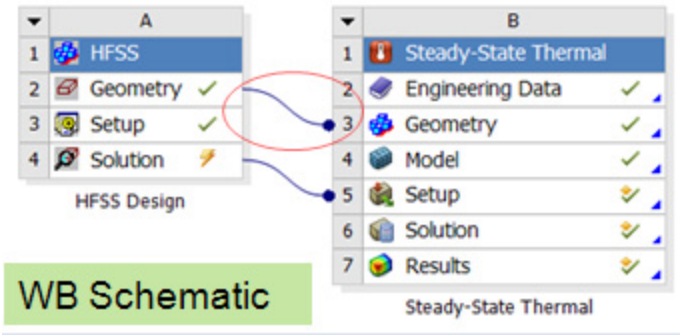
CAD Integration Material Assignment Transfer

Geometry Transfer through Ansys DesignModeler (DM)

CAD Integration Functionality

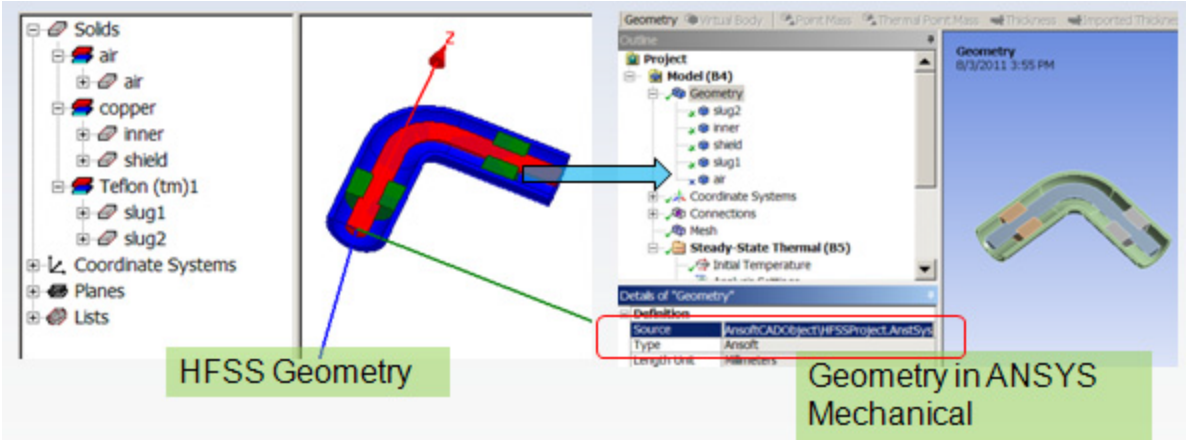
Ansys Electromagnetics to Ansys Geometry Transfer

Ansys Electromagnetics CAD Integration transfers model information based on Workbench links.



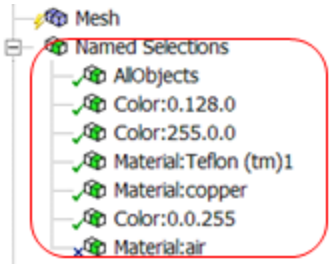
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The following figure shows how the information is transferred between simulators.

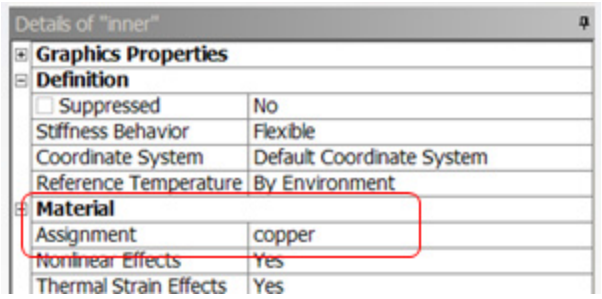


The information transferred includes:

- Geometry.
- Ansys Electromagnetics lists and material assignment as **Named Selections**.



- Material assignment.



The CAD Integration geometry link is:

- Dynamic because you can get updated geometry from Ansys Electromagnetics.
- Associative because IDs persist between the Ansys Electromagnetics model and the Ansys model during model refresh.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Boundary conditions in Ansys are preserved.

Related Topics

User Defined Model (UDM) for Ansys WB Integration

UDM compared to User Defined Primitives

Insert UDM Command on Draw Menu

UDM Properties

Library of Models for CAD Integration

CAD Integration Material Assignment Transfer

Geometry Transfer through Ansys DesignModeler (DM)

CAD Integration Functionality

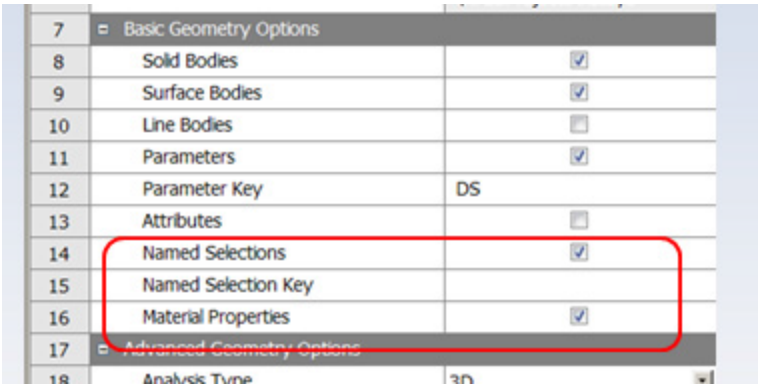
CAD Integration Material Assignment Transfer

Prerequisites for assignment transfer:

- Engineering data should have materials used in the Ansys Electromagnetics model and material names should match with case.
- **Material Properties** in **Basic Geometry Options** must be selected.

Prerequisites for transfer as named selection:

- **Named Selection** in **Basic Geometry Options** must be selected.



- **Named Selection Key** should either be empty or include **Material**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

User Defined Model (UDM) for Ansys WB Integration

UDM compared to User Defined Primitives

Insert UDM Command on Draw Menu

UDM Properties

Library of Models for CAD Integration

Ansys Electromagnetics to Ansys Geometry Transfer

Geometry Transfer through Ansys DesignModeler (DM)

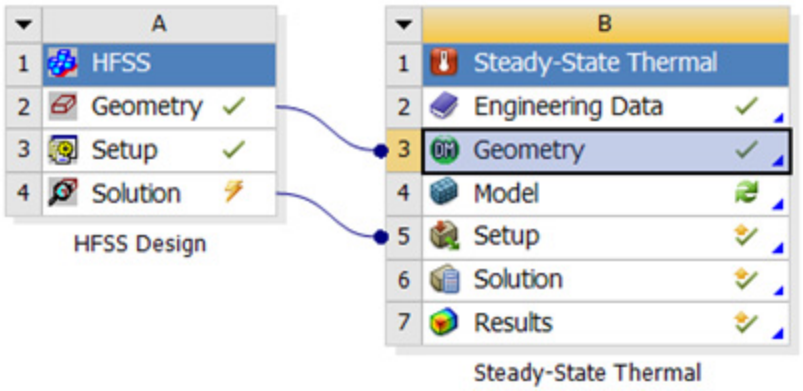
CAD Integration Functionality

Geometry Transfer through Ansys DesignModeler (DM)

You can edit geometry in Ansys DesignModeler (DM) before consuming in Mechanical.

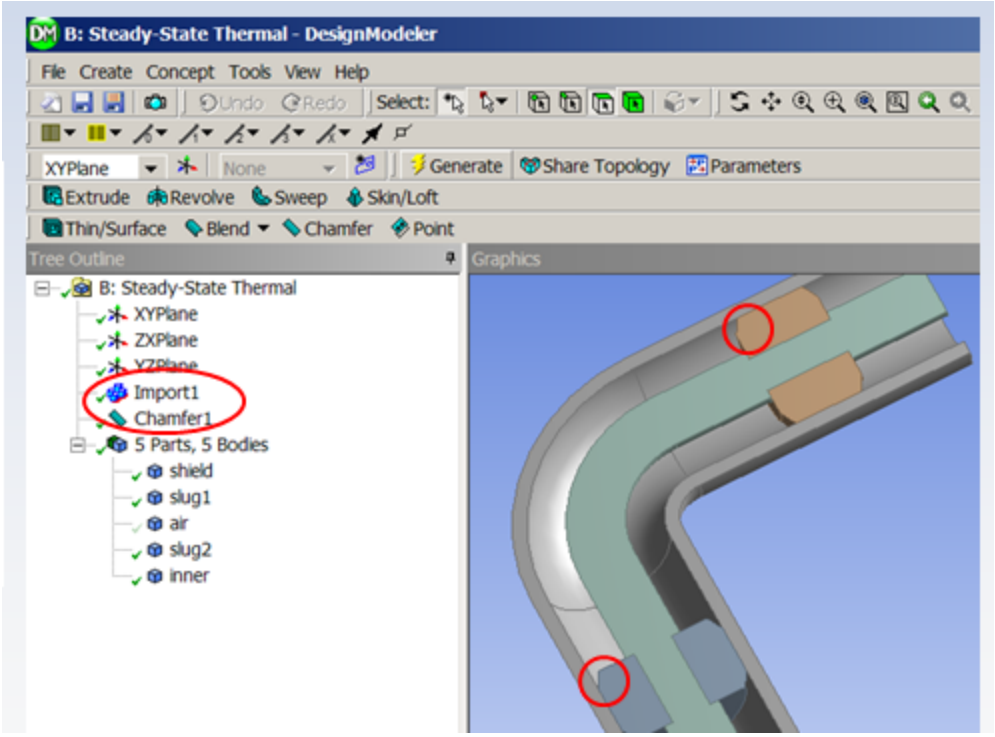
- Useful when geometry needs preprocessing for Ansys simulation.
- Can take advantage of geometry commands available in DM.

For example, consider an HFSS model linked to DM through the Workbench.



In this case, the following figure shows how a chamfer operation on geometries is imported.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Related Topics

User Defined Model (UDM) for Ansys WB Integration

UDM compared to User Defined Primitives

Insert UDM Command on Draw Menu

UDM Properties

Library of Models for CAD Integration

Ansys Electromagnetics to Ansys Geometry Transfer

CAD Integration Material Assignment Transfer

CAD Integration Functionality

CAD Integration Functionality

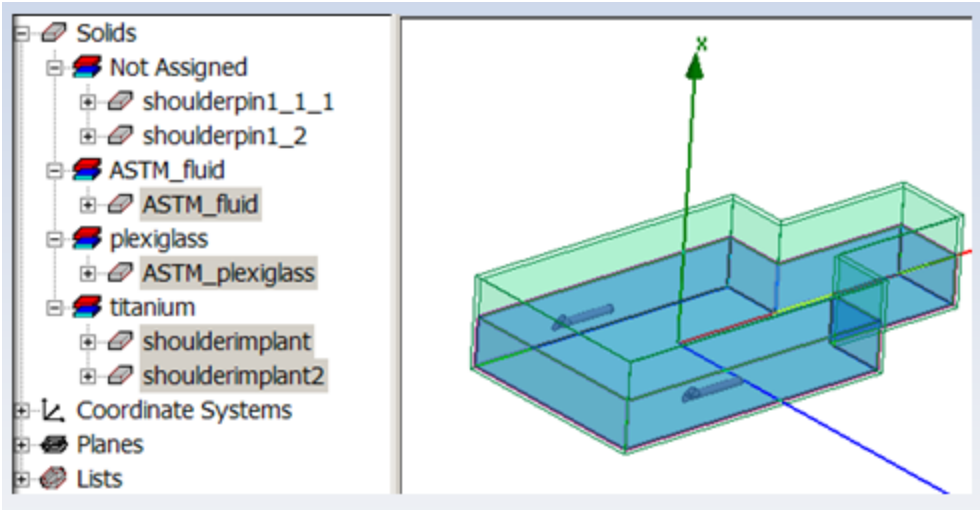
- WB Update Project
- WB Update All Design Points
- DX analysis

For explicit subtraction:

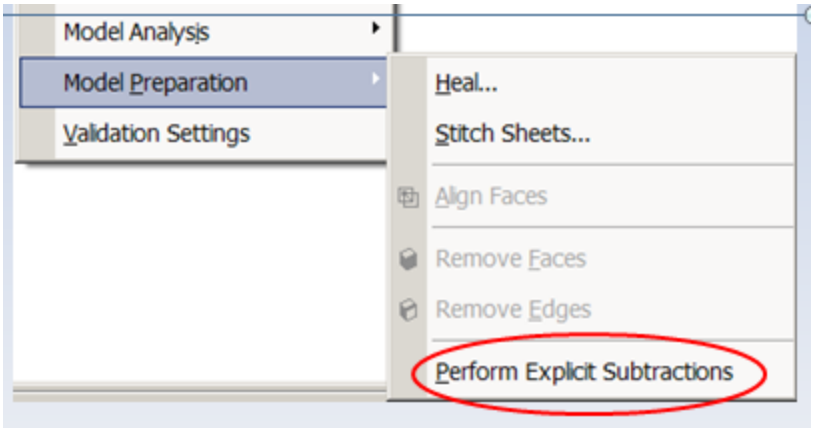
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Ansys Mechanical does not perform implicit subtraction.
- Select **Modeler** > **Model Preparation** > **Perform Explicit Subtraction** to perform explicit subtractions before sending geometry to Ansys.

For example, consider the following model.

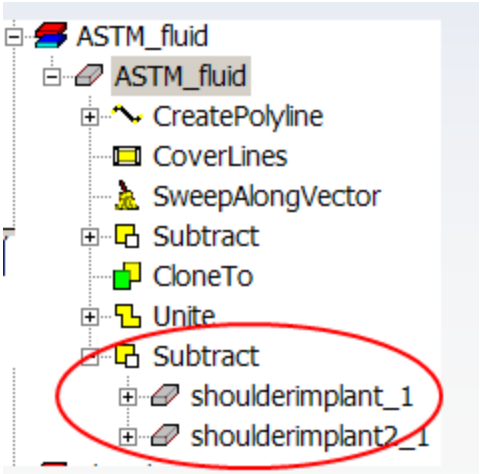


You can perform **Perform Explicit Subtractions**.

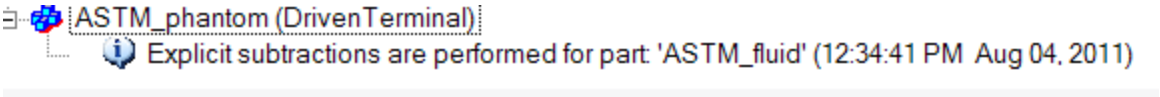


The results appear in the history tree:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The **Message Manager** pane reports this action.



Related Topics

User Defined Model (UDM) for Ansys WB Integration

UDM compared to User Defined Primitives

Insert UDM Command on Draw Menu

UDM Properties

Library of Models for CAD Integration

Ansys Electromagnetics to Ansys Geometry Transfer

CAD Integration Material Assignment Transfer

Geometry Transfer through Ansys DesignModeler (DM)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

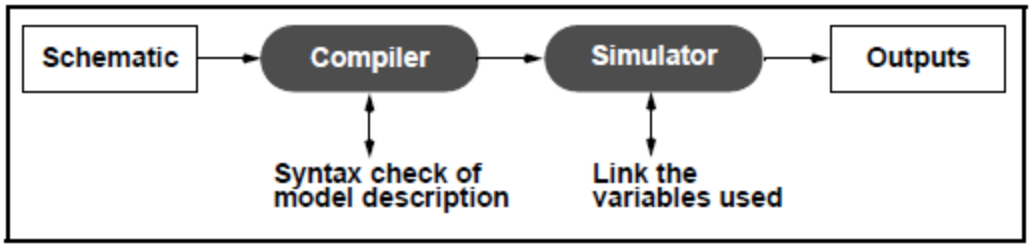
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

16 - Analyzing Twin Builder Designs

Twin Builder lets you set up and run analyses on the designs you construct in the **Schematic Editor** directly from the **Project Manager** using the simulator engine in the background for running the analyses.

The simulator is the heart of the Twin Builder system. Simulation models can be described using SML (Twin Builder's Modeling Language), VHDL-AMS, SPICE, and C. The compiler, which starts when you start the simulator, translates models into code the simulator can read.

At the beginning of each simulation (that is, analysis), the simulator resolves name references of quantities used in different modules and simulators. The compiler recognizes only semantic and syntactic errors in the SML data stream. Except for the output of error messages, you cannot affect the compile process.



Twin Builder uses the principle of simulator coupling. In simulator coupling, different single simulators connect to solve tasks from several technical fields – and to represent their interactions. These simulators exchange data during the simulation process, much as the real system components exchange energy and information in a real-time physical environment.

The simulator provides a variety of features, such as user-defined simulation precision, event-driven modification of simulation parameters, and manual parameter modification during simulation.

Simulator Backplane

Twin Builder achieves simulator coupling through a simulation backplane which is responsible for controlling each sub-simulator. Sub-simulators communicate with each other through a data bus.

Because certain basic knowledge about the simulation methods applied here is important for successful processing of a simulation task, the topics below provide a brief review of the fundamentals of Twin Builder's numerical algorithms and simulator coupling.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Circuit Simulator Processing

The global circuit simulator (also known as the Analog Simulator or Continuous System Simulator) solves a system of non-linear differential and algebraic equations. It employs the modified nodal approach to formulate the system of equations. For the solution of the differential equation system, a numerical integration method (either the Euler or the Trapezoidal algorithm) is applied. The nonlinear equations are linearized by the iterative Newton-Raphson method and the linearized system of equations is solved by LU factorization.

An important feature of the applied solvers is the automatic time step control. This feature ensures an adaptive calculation of the step width depending on the active dynamic situation in the system. This is done in a way that ensures an optimum between precision of calculation and simulation speed. The time step limitations are user-defined by a minimum and maximum time step. Through this specification, the possible time step range for the active simulation task is determined.

- **TEND** – Simulation duration.
- **HMIN** – Minimum time step width permissible.
- **HMAX** – Maximum time step width permissible.

The limit for number of time steps is the real number of time steps between these two boundaries. If the step width for a specific time step is too large, the circuit simulator requests a cancellation of this time step, and it will be repeated with an adapted step width (calculated by the system). The necessary step cancellations are forwarded to a step width manager and processed there.

Block Diagram Simulator Processing

Block diagrams are calculated according to the block transfer characteristics and in the sequence of signal flow. Integrators can be processed with any of the following selectable integration methods:

- Forward Euler
- Backward Euler
- Trapezoid
- Using the current network solver method

You can define a certain sampling time for each block. The blocks are then equidistantly calculated only on these discrete times. The sampling time must be selected very carefully. There may be dead time effects, especially in simulation models with different sampling times.

If no sampling time is specified, the block diagram module runs with the same (variable) time step as the circuit simulator (quasi continuous case). In this case, make sure that the block diagram time constants are larger than those in the electrical circuit model. If no sampling time is

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

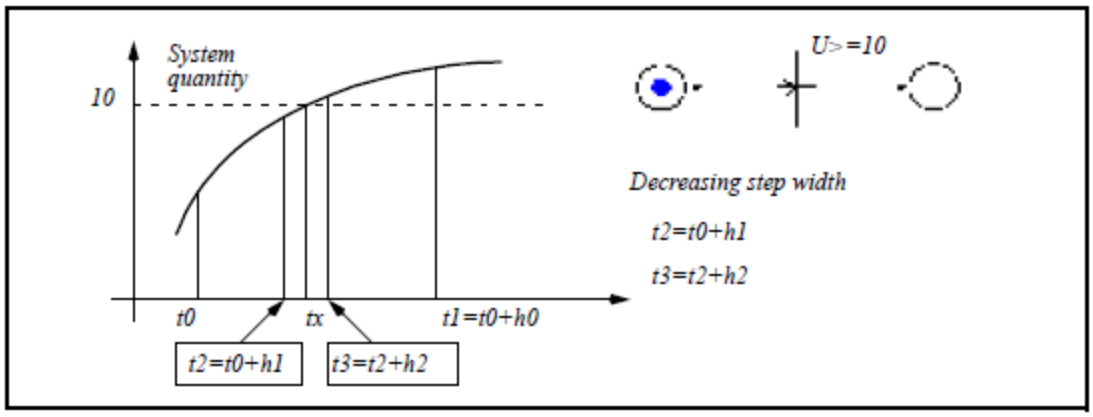
specified and there is also no electrical circuit in the model, the block diagram module runs constantly with HMAX.

The observance of block diagram simulator calculation times is guaranteed by the predictive time step control of the time step manager.

State Graph Simulator Processing

Within state machines the calculation of states, transitions, and actions continue as long as there is no other valid transfer condition. The calculations execute only in active (marked) states. Afterwards the system tests if there have been any events. Only if events have occurred and further states can be marked are new actions calculated.

State graphs do not need their own time step control; they work in discrete time and are actualized at each time step. Otherwise, the state graph simulator itself influences the time step (in the same way as other Twin Builder modules). This method of processing concerns first of all the identification of events. For the time determination of such an event, the simulator searches for a value as precise as possible, canceling and repeating simulation time steps with a decreasing time step until the minimum time step is reached. At this limit the time determination of an event is accepted as valid (with sufficient precision) and the simulation continues as usual. This event synchronization is performed using >= or <= and bypassed on the > and < operators.



VHDL-AMS Simulation

The VHDL-AMS simulator is a sub-simulator of the Twin Builder system. It calculates simulation models described in VHDL-AMS (Very high speed integrated circuit Hardware Description Language - Analog Mixed Signal). The SML compiler starts the VHDL-AMS simulator if VHDL-AMS models are used in the simulation model.

Components for VHDL-AMS Simulation

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Components in the Basic Elements VHDLAMS component library.
- Components in the Digital component library.
- Components implementation using VHDL-AMS.

Twin Builder Analyses

Before running analyses, you must specify the parameters that Twin Builder will use for analyzing the design. Analyses in Twin Builder can be divided into two categories:

- **Standard Analysis Types** – Single-run simulations for DC, Transient, and AC analyses. These are provided by adding a Standard Analysis Setup and Solution Options to the design. When added, these appear in the **Program Manager** Project tree under **Analysis**.
- **Advanced Analysis Types** – Multi-run simulations such as statistical analyses and optimization analyses. These are provided by the various Optimetrics analysis setups.

Standard Analysis Types

A standard analysis represents a single simulation run, in which the design models are evaluated with the specified simulator. Twin Builder supports the following standard analysis types:

- **TR (Transient) Analysis** – A transient analysis computes the time-domain response of a circuit by numerically integrating a system of differential/algebraic equations. The equations derive from the circuit topology and from information provided by the circuit device models. Transient analysis discretizes time and uses numerical integration methods (such as the trapezoidal rule) to solve the circuit equations at each time step.
- **AC Analysis** – AC analysis calculates the simulation model in the frequency domain. It first performs a DC simulation to calculate operating point values, then an AC simulation for a given frequency range.
- **DC Analysis** – DC analysis calculates the operating point for simulation models with nonlinear components in the quiescent domain. The voltage and current information about the operating point is saved to the .**sdb** file of the simulation model. The voltage values for the computed operation point display on the sheet at the components.

DC analysis provides the DC operating point voltages and currents. In turn, the DC operating point provides the initial values for DC sweep analysis, harmonic balance, and transient analyses. It also provides the large signal bias operating point for small signal AC analysis, noise analysis, and linear network analysis.

Related Topics

Standard Analysis Setup Options

Solution Options

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Running Analyses

Advanced Analysis Types - Optimetrics

Twin Builder's advanced analysis types, known collectively as Optimetrics analyses, are used to investigate an existing design in more detail. Optimetrics analyses usually require more than one simulation run. Typically these analyses are used for design optimization, and to determine how parameter variations influence a design's performance.

Related Topics

Optimetrics

Running analyses

Standard Analysis Setup Options

Twin Builder provides three standard analysis setup options:

- Transient Analysis Setup
- AC Analysis Setup
- DC Analysis Setup

Transient Analysis Setup

You can perform transient analysis on:

- All internal components.
- All VHDL-AMS components. See ***VHDL-AMS Simulation*** for additional information.
- C-Models defined for use with transient simulation.
- Macros using any of the components listed above.

Simulation parameters in the **Transient Analysis Setup** and **Solution Options** dialog boxes control the simulation process. Parameter values used by the simulator during a simulation also provide information about the quality of a simulation result. Proper choice of values for simulation parameters is very important for a fast and successful simulation. You can also use simulation parameters in equations and expressions.

The **Transient Analysis Setup** dialog box provides the following options:

- **Analysis Setup Name** – The default name is **TR**. If you specify additional solution setups, the default name increments by 1 (for example, **TR1**).
- **Analysis Control** – Includes values for **End Time - Tend**, **Min Time Step - Hmin**, and **Max Time Step - Hmax**.

Use the check boxes to **Use Initial Values** from a file, and to **Disable This Analysis**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Enable continue to solve causes the simulation to pause at the currently set end time (**Tend**) and to prompt you to set a new end time to continue the simulation. See ***Progress Bar Menu*** for additional information.

> **Note:**
>
> If a simulation ends *before* the specified end time (**Tend**), the simulation does not pause and the Progress bar closes normally.

- **Analysis Options** – Open a **Select Solution Options** dialog box where you can choose the solution option you intend to use for an analysis.

Related Topics

Adding a Transient Analysis

Guidelines for the Proper Choice of Time Step

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Deleting an Analysis Setup

Progress Bar Menu

Guidelines for the Proper Choice of Time Step

The accuracy of simulation results and performance of the simulator depend on the proper choice of minimum and maximum values for the simulation (integration) time steps (**Hmin** and **Hmax**). Smaller time step values yield more accurate results, but require longer processing times.

When specifying the minimum and maximum time step values, you must compromise between accuracy and time. The basic rule of measurement: *"Not as precise as possible, but as precise as required."* is also valid for a simulation. The following guidelines will help you prevent elementary mistakes in choosing the proper integration step width.

For **Hmin**, consider:

- What is the smallest time constant of the electric circuit (for example, R*C or L/R) or of the block diagram?

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Which is the smallest cycle of oscillations that can be expected (natural frequencies of the system or oscillating time functions or smallest digital event)?
- What is the smallest controller sampling time?
- What is the fastest transient occurrence (for example, edge changes of time functions)?
  - For ramp (or trapezoidal) signals with rise/fall time, Hmin should be sufficiently low to sample them.
  - When using models extracted from frequency sweeps (for example, for Maxwell Eddy or Q3D links), Hmin should be sufficiently low to sample the highest frequency of interest.

For **Hmax**, consider:

- What is the largest time constant of the electric circuit (for example, R*C or L/R) or of the block diagram?
- Which is the largest cycle of oscillations that can be expected (natural frequencies of the system or oscillating time functions)?
- What is the fastest transient occurrence (edge changes of time functions)?

For **Tend**, consider:

- What is the time interval to be simulated?

> **Note:**
>
> - Select the smallest of each estimated maximum and minimum time step for your simulation model.
> - All values recommended above are based on numeric requirements and experience and do not guarantee a successful simulation. Consider the algorithm as a guideline.
> - In case of doubt, decrease the maximum and minimum step size by dividing by 10. Repeat the simulation and compare the results. If the second set of results (with the step size decreased) shows conformity with the first results, then the step sizes chosen for the first simulation were appropriate (remember that smaller values increase the simulation time).

Adding a Transient Analysis

1. Right-click **Analysis** in the Project tree and select **Solution Setup** > **Add Transient**, or on the menu bar, select **Twin Builder** > **Solution Setup** > **Add Transient**. The **Transient Analysis Setup** dialog box appears.
2. A default name for the analysis appears in the **Analysis Setup Name** field.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The default name is **TR**. If you specify additional solution setups, the default name increments by 1 (for example, **TR1**). You can change the name by typing in the text field. The name must begin with an alpha character and may contain only alpha, numeric, and underscore characters.

3. In the **Analysis Control** panel:
   a. Set the desired values for:
      - **End Time - Tend** – The default value is 40ms.
      - **Min Time Step - Hmin** – The default value is 10us.
      - **Max Time Step - Hmax** – The default value is 1ms.

Hint	See ***Guidelines for the Proper Choice of Time Step*** for help with these settings.

   b. Select units for each value from the drop-down lists.
   c. Select **Use Initial Values** to load initial values from a file.

      Type the path and file name in the text box, or click for a file **Open** dialog box to locate and load the file.

   d. Select **Enable continue to solve** to pause the simulator after the current end time. When you select **Continue** on the simulator progress bar menu, the **Transient Analysis Setup** dialog box opens, prompting you to enter a new end time (**Tend**) with which to continue the simulation. The new end time must be greater in value than the current end time. This process repeats until you manually stop the simulator. If you set **Simulation Options** to save the simulator state and/or the initial conditions, the corresponding **.krn** and **.aws** files are also updated.
   e. Select **Disable this analysis** to exclude this analysis when analyses are run.

      Right-click an analysis icon in the Project tree and select **Disable** or **Enable** to control the analysis. The icon is gray when the analysis is disabled.

4. To choose solution options other than the default options, click the button to the right of **Analysis Options** to open the **Select Solution Options** dialog box where you can choose the desired solution option for this analysis. The button name indicates the current option.

   The **Select Solution Options** dialog box also lets you add, edit, clone, and remove solution options.

5. Select **OK** to close the **Transient Analysis Setup** dialog box.

   An icon for the transient analysis setup appears in the **Project Manager** window under the **Analysis** icon.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Monitoring and Controlling the Solution Process

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Deleting an Analysis Setup

AC Analysis Setup

You can perform AC analysis on these components:

- Passive
- Electrical cources (except for Fourier source)
- Switches (except for controlled switches)
- Semiconductor system level
- Semiconductor device level
- SPICE compatible models
- Transformers
- Continuous blocks
- Discrete blocks
- Source blocks
- Signal processing blocks (except for MAX, MIN, MAXT, MINT, two-point element with hysteresis)
- Math blocks
- Measurement (electrical domain)
- Time functions
- Characteristics
- Equations (except for DES solver)
- C-Models with definition for DC and AC simulation
- Macros using appropriate models (internal components, C-Models)

The following components are not supported for AC analysis:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Electrical machines
- State graph
- Signal characteristics
- Physical domain
- Digital components
- Time functions
- Simulator parameters

> **Warning:**
>
> If models without DC and AC implementation are used in an AC simulation, an error message appears.

The AC Simulator calculates the simulation model in the frequency domain. The AC Simulator first performs a DC simulation (to calculate operation point values) and an AC simulation (for a given frequency range). The voltage and current information for the DC simulation and the values for the AC simulation are saved in one .**sdb** file

The **AC Analysis Setup** dialog box provides the following options:

- **Analysis Setup Name** – The default name is **AC**. If you specify additional solution setups, the default name increments by 1 (for example, **AC1**).
- **Analysis Control** – Includes values for **Start Frequency - FStart**, **Stop Frequency - FEnd**, **Frequency time step** or **Points per decade - Fstep** (depending on the **AC sweep type** - linear or decadic). Use the check box to disable the analysis.

Enable continue to solve causes the simulation to pause at the currently set stop frequency and to prompt you to set a new stop frequency (**FEnd**) to continue the simulation. See ***Progress Bar Menu*** for additional information.

> **Note:**
>
> If a simulation ends *before* the specified stop frequency (**FEnd**) is reached, the simulation does not pause and the Progress bar closes normally.

- **Analysis Options** – Opens a **Select Solution Options** dialog box where you can choose the solution option to use for an analysis.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

AC simulation is based on small-signal analysis. In small-signal analysis, first a DC analysis is performed to find an operating point. By linearizing at the operating point, small-signal models for all nonlinear devices are analyzed over a user-specified range of frequencies.

It is important to note that the voltage- and current-dependent source nonlinear expressions (that is, when you select the **AC use** check box) are evaluated at the DC operating point and linearized at the operating point for AC analysis. For VHDL models, simultaneous statements must only be expressed purely as time domain equations, and will be linearized at the operating point for AC analysis.

Related Topics

Adding an AC Analysis

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Deleting an Analysis Setup

Progress Bar Menu

Guidelines for Configuring AC Simulation Models

The following configurations cannot be used with AC simulations:

- Parallel connection of voltage sources and inductances.
- Parallel connection of current sources and capacitances without any other branch.

If an AC simulation is started, the values for magnitude and phase, or real and imaginary part, defined for electrical and block sources, are used to generate the sine wave.

The magnitude of AC parameters can be a numerical value, a variable (defined in an initial assignment condition or equation), or an expression. If you use an expression, only quantity types with the attribute **Out** and standard functions can be used, for example *R10.V*ABS(C1.V)*. In contrast to the numerical value and the variable, expressions will be linearized in the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

operating point. Therefore, expressions also contain information about the current phase, whereas numerical values and variables have no phase information.

The parameters: phase, real part, and imaginary part are common parameter types. You can use all numerical values, variables, or expressions.

Adding an AC Analysis

1. Right-click **Analysis** in the Project tree and select **Solution Setup** > **Add AC**, or on the menu bar, select **Twin Builder** > **Solution Setup** > **Add AC**. The **AC Analysis Setup** dialog box appears.
2. A default name for the analysis appears in the **Analysis Setup Name** field.

   The default name is **AC**. If you specify additional solution setups, the default name increments by 1 (for example, **AC1**). You can change the name by typing in the text field. The name must begin with an alpha character and may contain only alpha, numeric, and underscore characters.

3. In the **Analysis Control** panel:
   a. Set the desired values for:
      - **Start Frequency - FStart** – The default value is 1Hz.
      - **Stop Frequency - FEnd** – The default value is 1kHz.
      - **AC sweep type** (**linear**) – The default, or **decadic**.
      - **Frequency time step - Fstep** – The default value is 10Hz. This field displays when **AC sweep type** is **linear**.
      - **Points per decade - Fstep** – The default value is 10. This field displays when **AC sweep type** is **decadic**.

   b. Select units for each value from the drop-down lists.
   c. Select **Enable continue to solve** if you want the simulator to pause after the currently set stop frequency. When you select **Continue** on the simulator progress bar menu, the **AC Analysis Setup** dialog box opens, prompting you to enter a new stop frequency (**FEnd**) with which to continue the simulation. The new stop frequency must be greater in value than the current stop frequency. This process repeats until you manually stop the simulator. If you set **Simulation Options** to save the simulator state and/or the initial conditions, the corresponding **.krn** and **.aws** files are also updated.
   d. Select **Disable this analysis** to exclude this analysis when analyses are run.

      You can also right-click an analysis icon in the Project tree and select **Disable** or **Enable**. The icon is gray when the analysis is disabled.

4. To choose solution options other than the default options, click the button to the right of **Analysis Options** to open the **Select Solution Options** dialog box where you can choose the desired solution option for this analysis. The button name indicates the currently selected option.)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Use the **Select Solution Options** dialog box to add, edit, clone, and remove solution options.

5. Click **OK** to close the **AC Analysis Setup** dialog box.

   The AC analysis setup appears in the **Project Manager** window under the **Analysis** icon.

Related Topics

Monitoring and Controlling the Solution Process

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Deleting an Analysis Setup

DC Analysis Setup Options

You can run a DC analysis on the following components:

- Passive
- Electrical sources (except for Fourier source)
- Switches (except for controlled switches)
- Semiconductor system level
- Semiconductor device level
- SPICE compatible models
- Transformers
- Continuous blocks
- Discrete blocks
- Source blocks
- Signal processing blocks (except for MAX, MIN, MAXT, MINT, two-point element with hysteresis)
- Math blocks
- Measurement (electrical domain)
- Time functions
- Characteristics
- Equations (except for DES solver)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- C-Models with definition for DC and AC simulation.
- Macros using appropriate models (internal components, C-Models)

The following components are not supported for DC analysis:

- Electrical machines
- State graph
- Signal characteristics
- Physical domain
- Digital
- Time functions
- Simulator parameters

> **Warning:**
>
> If models without DC and AC implementation are used in a DC simulation, an error message appears.

The DC Simulator calculates the operating point for simulation models with nonlinear components in the quiescent domain. Voltage and current information about the operating point is saved to an **.sdb** file of the simulation model. Voltage values for the computed operation point display on the sheet at the components.

The **DC Analysis Setup** dialog box provides the following options:

- **Analysis Setup Name** – The default name is **DC**. If you specify additional solution setups, the default name increments by 1 (for example, **DC1**).
- **Analysis Control** – Use this check box to enable or disable the analysis.
- **Analysis Options** – Click **Analysis Options** to open a **Select Solution Options** dialog box where you can choose the solution option to use for an analysis.

Related Topics

Adding a DC Analysis

Viewing DC Bias Values in a Schematic

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Deleting an Analysis Setup

Guidelines for Configuring DC Simulation Models

The following configurations cannot be used with DC simulations:

- Parallel connection of voltage sources and inductances.
- Parallel connection of current sources and capacitances without any other branch.

Adding a DC Analysis

1. Right-click **Analysis** in the Project tree and select **Solution Setup** > **Add DC**, or on the menu bar, click **Twin Builder** > **Solution Setup** > **Add DC**. The **DC Analysis Setup** dialog box appears.
2. A default name for the analysis appears in the **Analysis Setup Name** field.

   The default name is **DC**. If you specify additional solution setups, the default name increments by 1 (for example, **DC1**). You can change the name by typing in the text field. The name must begin with an alpha character and may contain only alpha, numeric, and underscore characters.
3. In the **Analysis Control** panel, select **Disable this analysis** to exclude this analysis when analyses are run.

   You can also right-click an analysis icon in the Project tree and select **Disable** or **Enable**. The icon is gray when the analysis is disabled.
4. To choose solution options other than the default options, click the button to the right of **Analysis Options** to open the **Select Solution Options** dialog box where you can choose the desired solution option for this analysis. The button name indicates the currently selected option.

   The **Select Solution Options** dialog box also lets you add, edit, clone, and remove solution options.
5. Select **OK** to close the **DC Analysis Setup** dialog box.

   The DC analysis setup appears in the **Project Manager** pane under the **Analysis** icon.

Related Topics

Viewing DC Bias Values in a Schematic

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Deleting an Analysis Setup

Viewing DC Bias Values in a Schematic

To run a DC bias analysis on a circuit design in the Schematic Editor, select **Twin Builder** > **View DC Bias Values**. A submenu containing three items appears:

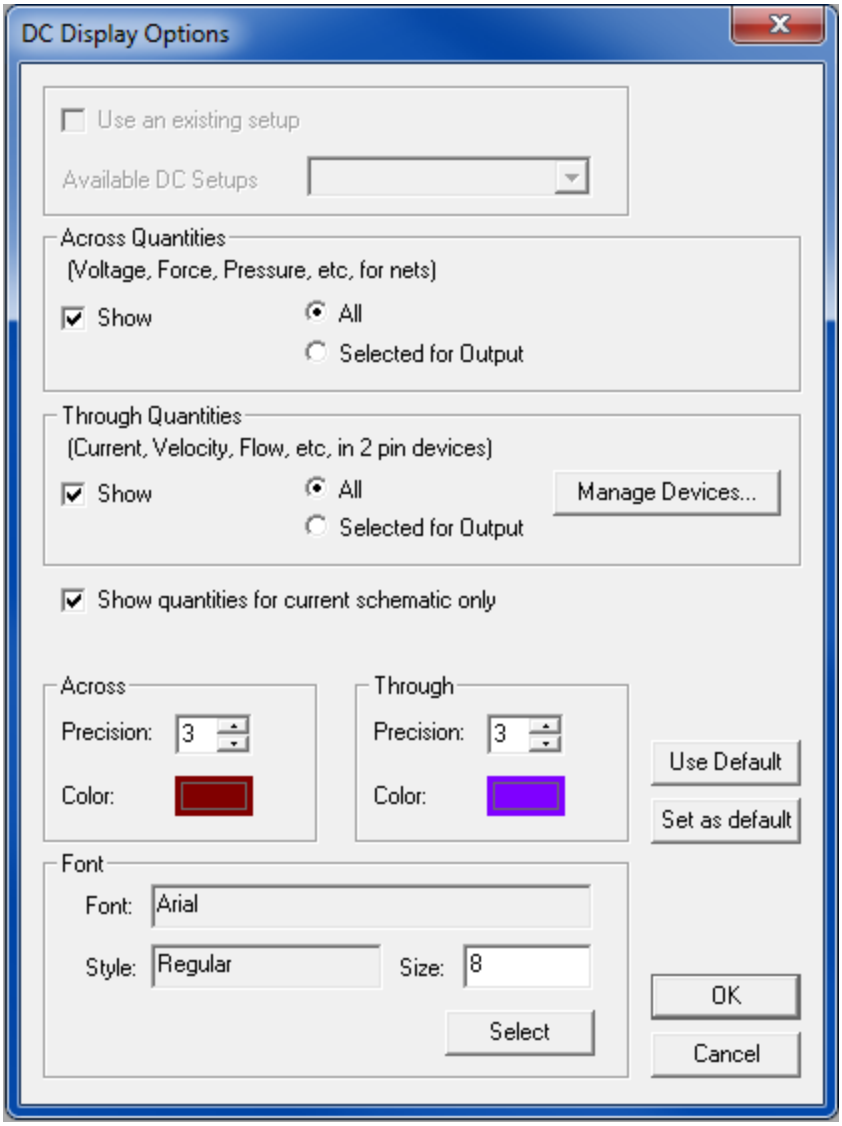
- **Show DC Bias**
- **Update**
- **Display Options**

Show DC Bias – Toggles visibility of existing data on and off. If there is no existing data, a simulation runs to create it. A check mark appears next to the menu command to indicate that visibility is turned on.

Update – Causes a simulation to run so the data reflects the current design topology and settings. **Update** is disabled if DC bias data is not visible.

Display Options – Opens the **DC Display Options** dialog box allowing display customization:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Use the **Use an existing setup** check box to pick an existing DC setup, listed in the **Available DC Setups** drop-down list. The check box and list are disabled if there are no existing DC setups. If an existing setup is not chosen, a temporary setup that uses default values is created when needed.

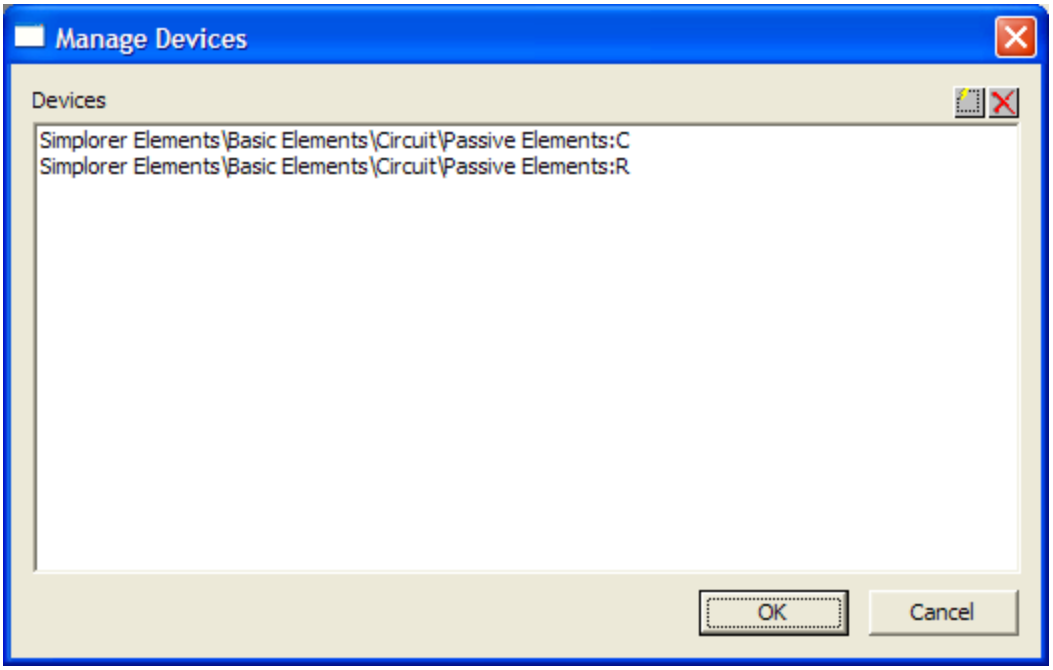
In the **Across Quantities** panel, toggle display of net “across” values with the **Show** check box. Select **All** to show a value for every conservative net. Select **Selected for Output** to display the net “across” quantities selected in the **Output** dialog box.

Similarly, in the **Through Quantities** panel, toggle display of component “through” values with the **Show** check box. Select **All** to show a value for every component instance. Select **Selected**

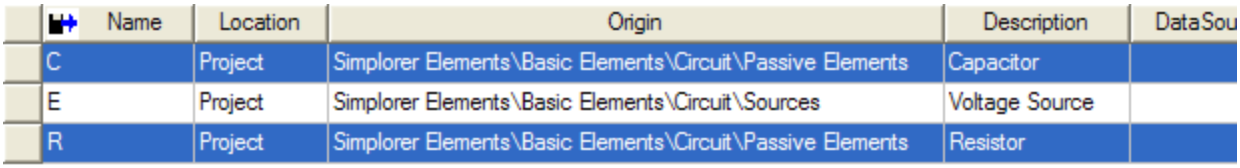
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

for Output to display the component instance “through” quantities selected in the **Output** dialog box.

Click **Manage Devices** to choose appropriate 2-terminal devices through the following dialog box:



Click to add devices to the list.



Name	Location	Origin	Description	DataSou
C	Project	Simplorer Elements\Basic Elements\Circuit\Passive Elements	Capacitor	
E	Project	Simplorer Elements\Basic Elements\Circuit\Sources	Voltage Source	
R	Project	Simplorer Elements\Basic Elements\Circuit\Passive Elements	Resistor	

Select a device and click to remove it from the list.

Select **Show quantities for current schematic only** to limit the bias display to the current schematic. If the box is cleared, bias values display for the current design and all its subdesigns. In a subdesign, display of bias values is based on a parent's simulation if the parent's check box was cleared and that bias display was most recently shown. To see values based on simulation of the current subdesign, invoke **Twin Builder/View DC Bias Values/Update**.

Use the **Across** and **Through** panels to control the precision (number of decimal points) and color used to display these values.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Use the **Font** panel to control the font used for the displayed values. Click **Select** to open the **Font** dialog box. Use the sliders to select the desired font, font style, and size, then click **OK** to return to the **DC Display Options** dialog box.

Click **Use Default** to restore the default option values in the dialog box. Click **Set as default** to make the current options the defaults.

Click **OK** to enable the specified options, or click **Cancel** to close the dialog box without changing any options.

Related Topics

Adding a DC Analysis

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Setting the Outputs for Simulation

Select Solution Options

The **Select Solution Options** dialog box lets you choose the solution options used by your analysis. It also lets you add, edit, clone, and remove solution options.

- The list box displays the solution options currently defined for the project.
- **New** – Opens the **Solution Options** dialog box in which you can name and define a new set of solution options. New options sets are added to the list of available selections.
- **Edit** – Opens the selected solution options set in the **Solution Options** dialog box for editing.
- **Clone** – Generates a copy of the selected solution options.
- **Remove** – Deletes the selected solution options. A warning message displays if you attempt to delete an option set that is in use.

To assign a solution options set for use with an analysis, select the desired options set form the list and click **OK**.

Setting the Active Analysis Setup

If you have more than one analysis setup defined for a project, you can set an active analysis setup as follows:

1. On the Twin Builder main menu, choose **Twin Builder** > **Set Active Setup**. The **Set Active Setup** dialog box appears.
2. Click the setup name you want to make active and click **OK** to confirm the choice and close the dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Deleting an Analysis Setup

Disabling/Enabling an Analysis Setup

To disable or enable an analysis setup, in the Project tree, right-click the icon of the analysis setup, and select **Disable** or **Enable**.

Analysis setup icons are gray to indicate that an analysis is disabled.

Related Topics

Setting the Active Analysis Setup

Editing an Analysis Setup

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Deleting an Analysis Setup

Editing an Analysis Setup

To edit a transient, AC, or DC analysis setup:

1. In the Project tree, double-click the icon of the analysis setup to edit. The appropriate **Analysis Setup** dialog box appears.
2. Make your changes to the analysis setup settings.
3. Click **OK** to save the changes and close the dialog box.

> **Note:**
>
> You can also edit an analysis setup by selecting its icon in the Project tree, then editing the settings in the **Properties** dialog box **Analysis Setup** tab.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Deleting an Analysis Setup

Copying and Pasting an Analysis Setup

1. In the Project tree, right-click the icon of an analysis setup and select **Copy**.
2. Right-click the **Analysis** icon and select **Paste**.

Related Topics

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Renaming an Analysis Setup

Deleting an Analysis Setup

Renaming an Analysis Setup

To rename an analysis setup:

1. In the Project tree, right-click the analysis setup you want to rename. The name next to the icon becomes editable.
2. Make the desired change to the name.
3. Press **Enter** to save the change.

Related Topics

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Copying and Pasting an Analysis Setup

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Deleting an Analysis Setup

Deleting an Analysis Setup

> **Warning:**
>
> Deleting an analysis setup also deletes all plots that depend upon it.

To delete an analysis setup:

1. In the Project tree, right-click the analysis setup you want to delete.
2. Select **Delete** to delete the analysis setup.

Related Topics

Setting the Active Analysis Setup

Disabling/Enabling an Analysis Setup

Editing an Analysis Setup

Copying and Pasting an Analysis Setup

Renaming an Analysis Setup

Solution Options

Use the **Solution Options** dialog box to set various parameters that control the analyses performed on your Twin Builder designs.

The **Solution Options** dialog box includes the following tabs:

- **TR (Transient) Options**
- **AC Options**
- **DC Options**
- **General Options**
- **SML Header Options**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

TR (Transient) Options

- **Integration formula**

Euler	Damping effects may cause false results for energy relationships.
Adaptive Trapezoid-Euler	Default integration formula. Used for oscillating LC systems or analog oscillators. This is especially useful for investigating energy problems.
Trapezoid	Pure trapezoidal integration formula (no switching to Euler like the Adaptive Trapezoid-Euler formula). For purely oscillatory circuits with no discontinuities, this method gives accurate results and may be desirable. The method is less stable particularly in cases with discontinuities or non-zero derivatives at time=0.

- **Local truncation error[%] - LDF**

  Defines the acceptable truncation error value. This value controls the time step depending on the dynamic nature of the circuit. A small error increases the precision but also the calculation time. This number itself does not control the precision of the overall result. The default value is 1.

- **Maximum number of iterations - IteratMax**

  Maximum number of iterations allowed per simulation step. If convergence problems occur, the calculation for the active simulation step stops when the maximum value is reached without consideration of other error limits. The simulation then proceeds to the next step. The default value is 40.

- **Absolute tolerance - RHS**

  Absolute tolerance for RHS vector (residual vector). Used to determine the acceptable error (convergence criteria) for the RHS vector in the Newton-Raphson method. A lower value leads to more accurate results, but convergence problems might be possible in the Newton-Raphson iterations. The default value is 1*m*, where *m* stands for **milli(10^-3)**.

- **Absolute tolerance - LHS**

  Used to determine the acceptable error (convergence criteria) for the solution vector update in the Newton-Raphson method. A lower value leads to more accurate results, but convergence problems may occur using the Newton-Raphson iterations. The default value is 1*m*, where *m* stands for **milli(10^-3)**.

- **Relative tolerance[%]**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If enabled, the simulator uses the specified relative tolerance percentage along with the absolute values (**Maximum Current Error** and **Maximum Voltage Error**) to determine acceptable error, and to determine convergence.

> **Note:**
>
> In case of non-convergence during transient simulation (that is, the maximum number of iterations has been exceeded), see ***Fixing Non-Convergence in Twin Builder***.

- **Apply operating-point convergence scheme**

  Achieving convergence at the transient operating point (at t = 0) can be difficult for circuits with highly nonlinear models. In situations where convergence fails at the operating point, this option applies the continuation methods Gmin-stepping and Source-stepping to improve convergence.

  The *Gmin-stepping* method adds a large conductance value to ground for each node so that the nonlinear behavior is damped and sparse matrix becomes well-conditioned. It then gradually lowers the conductance value until the conductance becomes negligibly small.

  The *Source-stepping* method starts solving with zero value for all independent voltage and current sources, and continues to solve by slowly ramping their values until they reach their full values.

  > **Note:**
  >
  > Gmin-stepping is available only for the default sparse matrix solver (not for new sparse matrix code).
  >
  > Source-stepping is applied only for SML type independent sources.

- **Analog/Digital synchronization**

  Mixed-signal synchronization ensures timely exchange of values between the analog and digital sub-simulators when simulating systems with both digital (VHDL) and analog (VHDL-AMS and SML) models or constructs. This simulation option controls the synchronization strategy used. Although each of the strategies below will yield correct results, correctly selecting the best synchronization strategy for the system to be simulated yields the fastest simulation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Hybrid	Default setting. Employs an optimistic digital and a conservative analog synchronization strategy. Results in faster and more efficient simulations than **Conservative** synchronization by reducing the number of unnecessary analog solution points. Yields the fastest simulations when the digital sub-system is smaller than the analog sub-system. Digital simulation events occur at a higher frequency than analog steps, and not every digital event affects the analog sub-system.
Adaptive	Adaptively controls the amount of optimism in the optimistic digital simulation intelligently depending on the system. Yields the fastest simulations for systems with small fast digital systems. When the digital sub-system is larger than the analog sub-system, digital simulation events occur at a higher frequency than analog steps, and not every digital event affects the analog system.
Conservative	Synchronizes at the minimum time step requested by each sub-simulator. Requires that the analog system simulation be solved for every digital system event, whether the analog system is affected. Yields the fastest simulations for systems with tight feedback loops requiring frequent synchronization (that is, when every digital event affects the analog sub-system).

- **Advanced step mode**

  Switches between the advanced and standard step size algorithms. By default, the improved algorithm is used. Clear the check box to use the standard algorithm. With **Advanced step mode** selected, the step size is more dynamic – increasing and decreasing at a faster rate compared to the standard algorithm. Using **Advanced step mode** leads to faster simulation speed especially when a small **HMin** is necessary for good accuracy, but only at limited points throughout the simulation. For best results with this algorithm, there should be sufficient separation between **HMin** and **HMax**.

  If the system needs a small timestep at a lot of points, the standard algorithm could be the better choice. This is because the advanced step mode causes the simulator to first overrun the required simulation points, then rollback to hit it. This increases the total number of iterations and slows down the simulation speed.

Number of equal steps	When **Advanced step mode** algorithm is selected, this parameter sets the number of consecutive time steps of equal size that must be taken before a larger step size is used. Default value is zero. An integer value greater than zero should be used.
Step acceleration damping [%]	When **Advanced step mode** algorithm is selected, this parameter sets the rate of damping of the step size acceleration for the next time step. Default value is zero. A value between 0-100% should be selected. A lower value for this parameter allows higher simulation

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	speed; use a higher value to increase simulation stability.

- **Samanskii factor**

  The **Update Jacobian after ___ iterations** setting is an optimization method where the simulator only re-evaluates the Jacobian Matrix after the specified number of iterations. Default is one (1) which indicates that optimization is turned off. A value between 1 and 10 is allowed. In case of systems requiring a very large number of iterations per time step for convergence, a value greater than 1 can improve the performance of the simulator by avoiding Jacobian evaluations and matrix factorizations.

  > **Note:**
  >
  > A very high value may lead to non-convergence. Ideally use a value between 2 and 6.

- **Multithreaded Solver**

  Uses multiple threads to perform model evaluation for all the models in the circuit (analog) solver. Model evaluation involves computing model equations with updated solution vector, and Jacobian and residual computations. Each thread processes one model at a time. As a result, if there are *N* threads, *N* models can be evaluated in parallel in a multi-core machine. Given the overhead of processing multiple threads, the real performance benefit can usually be realized when there are large numbers of models involved, or when there are complex models that require significant computational resources.

Enable multithreaded solver	Uses multiple threads for the solver computation.
Auto compute number of threads	Determines how many threads required – based on physical CPU cores available and number of circuit simulator models. If your machine has fewer than four cores, the solver disables multithreading for this option.
Specify number of threads	If the above option is not desired, you can specify how many threads to use. The actual number is limited by circuit simulator model count.

Note	Several models such as VHDL and SPICE are processed in serial in this version due to thread safety issues. However, in future releases, more models will be available for parallel processing.
Note	By default, all user-defined C-Models are processed in serial mode unless an explicit C-Model API function call, SetAsThreadSafe, is made. For more details,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	see ***CModUser Object Methods***.

AC Options

- **Maximum number of iterations - Iteratmax**

  Maximum number of iterations for one simulation step. If convergence problems occur, the calculation for the active simulation step stops when the maximum value is reached without consideration of other error limits. The default value is 40.

- **Maximum Error [A] - EMmaxAC**

  The default value is 1n.

Related Topics

Adding Solution Options

Editing Solution Options

Copying and Pasting Solution Options

Renaming Solution Options

Deleting Solution Options

DC Options

- **Maximum number of iterations - Iteratmax**

  Maximum number of iterations for one simulation step. If convergence problems occur, the calculation for the active simulation step stops when the maximum value is reached without consideration of other error limits. Default value is 50.

- **Maximum error [A] - EMmaxDC**

  The default value is 1m.

- **Maximum number of relaxations - Relaxmax**

  The default value is 10.

- **Use VHDL-AMS quiescent_domain**

  Switch between using VHDL-AMS quiescent-domain equations for DC analyses (VHDL-AMS mode), or time domain equations with 0 transients - that is, steady state (Twin Builder mode). Select this option to enable VHDL-AMS quiescent-domain usage.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Adding Solution Options

Editing Solution Options

Copying and Pasting Solution Options

Renaming Solution Options

Deleting Solution Options

General Options

- **Ambient Temperature (Cel) - Temp**

  Global ambient temperature for temperature dependent components. The default value is 27 degrees Celsius.

- **Enable SPICE Formula Solver**

  Select this setting to use an enhanced formula solver for expression-controlled sources. This solver is particularly beneficial for very large expressions, which typically are the result of resolving a large set of **.FUNC** definitions during the import of SPICE models. The default setting is **ON**. It is also possible to use this solver for textual SML model definitions by adding **SPCFML:=1** in the parameter list of expression-controlled sources

- **Use new sparse matrix code**

  This option uses a different sparse matrix solver for the global circuit simulator. The default matrix solver (when this check box is cleared) is the most appropriate solver for any typical application. However, in certain situations such as the default solver failing due to singularity error or taking a long time to solve, using this solver may be helpful.

Damping Heuristics

Damping heuristics settings provide one possible method for achieving convergence in cases of convergence failure. They are especially useful for highly non-linear behavior – such as that produced by exponential characteristics where the Newton-Raphson algorithm does not converge due to large changes in the flow quantities (such as current) caused by changes in the node potentials. This algorithm prevents divergence by limiting the allowed change in node potentials per Newton-Raphson iteration.

- **Limit node potential change per iteration** – Select to enable the Node Potential Limiting method.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Limit for initial simulation only** – Select to limit node potential change for the first step only (that is, operating point analysis in case of TR analysis, and DC solution in case of AC analysis)
- **Absolute limiting value** and **Relative limiting value** – Used to calculate the acceptable change per iteration. The acceptable change value is calculated as follows - ((**Relative limiting value**)/100) * Current Potential Node Value + **Absolute limiting value**. Any change greater than this in an iteration will be limited to this value. Units for the **Absolute limiting value** are the same as the corresponding potential node units (V, m/sec, rad/sec, Pascals, deg Centigrade, and so on).

Data Reduction

Use these data reduction settings to reduce the quantity of saved solution data.

- **Start saving data at** – The *time* (or *frequency* for AC simulations) at which simulation results data saving begins.
- **End saving data at** – The *time* (or *frequency* for AC simulations) at which simulation results data saving ends.
- **Dynamic Limits** – **Off** by default.

  Output every ... steps lets you sample the solution output every "*n*" steps, where "*n*" is a positive integer. For example, if "*n*" is 1, every solution is saved; if "*n*" is 2, every second solution is saved; if "*n*" is 3, every third solution is saved, and so on.

  Output time step [s] lets you sample the solution output at the set time interval (or the set frequency interval for AC simulations).

  Maximum relative change [%] lets you sample the solution output based on the set percentage change relative to the previous solution value.

Simulator Pivoting Strategy

Select the **Manual** check box to enable the performance slider control. Use this control to influence the pivoting algorithm used by the circuit simulator. The position of the performance slider and the sparseness of the system matrix determine the choice between partial (diagonal) or complete pivoting in the solver. Position the slider to the extreme right to force **Complete pivoting**, or to the left extreme to force **Partial pivoting**. (see ***Using the Simulator Performance Slider*** for additional information.)

Advanced Logging Info

This option enables logging of additional information created by FMUs, Twin models, Modelica models, and Dynamic ROMs. You can select the level of detail and the location of the log file. If you enable logging, a separate log file for each component is written to the design's result folder (<*project location*>\<*projectname*>**.aedtresults**<*designname*>), but you can choose a different

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

location. Generated logs are in the **AdvancedLogs** subdirectory. Log file names have a <*ComponentName*>.**log** format.

> **Note:**
>
> For Optimetrics analysis, logs appear under one more level of subdirectory, corresponding to the sweep's task number.

Related Topics

Adding Solution Options

Editing Solution Options

Copying and Pasting Solution Options

Renaming Solution Options

Deleting Solution Options

Using the Simulator Performance Slider

Use the slider control in the **Simulation Parameters** dialog box to improve transient simulator performance by influencing the pivoting algorithm. The slider only influences the network solver; it has no effect on components that are not using the analog solver, such as state machines and digital systems. The slider is enabled if the **Automatic** check box in the **Performance** sub-section of the dialog box is cleared. Moving the slider to the left or to the right biases the pivoting algorithm toward partial or complete pivoting. At the extreme left slider position, the simulator uses purely diagonal pivoting; while at the extreme right slider position, the simulator uses purely complete pivoting.

> **Note:**
>
> The default position has been determined empirically to be the most effective for the largest variety of systems. In cases where the simulation fails to converge, or the initial solution is incorrect or not found, changing the slider position may be helpful.

Changing the slider position

Use the slider to improve inaccurate or slow simulations. You can also customize simulator performance to best fit the specific application domain and the systems to be simulated.

These guidelines determine the ideal slider position:

Partial Pivoting

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Typically yields a faster simulation.
- May lead to an inaccurate solution (which may lead to poor performance).

Systems with the following characteristics give better simulation results with partial pivoting:

- Small systems (<100 components) with symmetric structure.
- SML models usually contribute to a symmetric structure.

Complete Pivoting

- Typically gives more accurate results.
- Slower performance (as compared to partial pivoting) if the system does not have characteristics specifically suited for complete pivoting.

Systems with the following characteristics give better simulation results with complete pivoting:

- Medium to large systems (>100 components) with highly non-symmetric structure.

  > **Note:**
  >
  > VHDL-AMS models (and other non-SML models) usually contribute a non-symmetric structure.

- Systems with widely varying component values. For example, a system with some zero-valued resistances and some 10M Ohm resistances.

Related Topics

Adding Solution Options

Editing Solution Options

Copying and Pasting Solution Options

Renaming Solution Options

Deleting Solution Options

SML Header Options

Add Twin Builder Modeling Language (SML) text directly to a simulation model using the **SML Header** tab in the **Solution Options** dialog box.

Enter SML statements (such as **#include**) in the text box. The header text is inserted at the beginning of the model's simulation script. You can define the header either for the active model only, or as a default for all models by selecting **Use as default**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Inserted text must comply with the SML syntax rules.

Related Topics

Adding Solution Options

Twin Builder Modeling Language

Adding Solution Options

To add options to a solution setup:

1. Right-click **Analysis** in the Project tree and select **Add Solution Options**. The **Solution Options** dialog box appears with the **TR** tab selected.
2. Enter a name in the **Name** field. The default name is **Options***n*.
3. Enter the following information as appropriate:
   a. On the **TR** (Transient Solution) tab, specify:
      - **Integration formula** (**Euler** or **Adaptive Trapezoid Euler**)
      - **Local truncation error [%] - LDF** (default=1)
      - **Maximum number of iteration - IteratMax** (default=40)
      - **Maximum current error -IEmax** (default=1m)
      - **Maximum voltage error -VEmax** (default=1m)
   b. On the **AC** tab, specify:
      - **Maximum number of iterations - Iteratmax** (default=40)
      - **Maximum Error [A}- EMmaxAC** (default=1n)
   c. On the **DC** tab, specify:
      - **Maximum number of iterations - Iteratmax** (default=50)
      - **Maximum error [A}- EMmaxDC** (default=1m)
      - **Maximum number of relaxations - Relaxmax** (default=10)
   d. On the **General** tab, specify the **Ambient temperature (Cel) - Temp** (default=27).
   e. On the **SML Header** tab, you can provide SML text to be added at the beginning the SML file generated by the schematic.
4. Click **OK** to close the **Solutions Options** dialog box.

   The named solution options appear in the **Project Manager** window under the **Analysis** icon.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Editing Solution Options

Copying and Pasting Solution Options

Renaming Solution Options

Deleting Solution Options

Editing Solution Options

To edit solution options:

1. In the Project tree, double-click the icon of the solution option you want to edit. The **Solution Options** dialog box appears with the **TR** tab selected.
2. Make the desired changes to the solution options settings.
3. Click **OK** to save the changes and close the dialog box.

Related Topics

Adding Solution Options

Copying and Pasting Solution Options

Renaming Solution Options

Deleting Solution Options

Copying and Pasting Solution Options

1. In the Project tree, right-click the icon of the solution option you want to copy.
2. Select **Copy** to copy the solution option.
3. Right-click the **Analysis** icon and select **Paste** to paste the copied solution option.

Related Topics

Adding Solution Options

Editing Solution Options

Renaming Solution Options

Deleting Solution Options

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Renaming Solution Options

To rename solution options:

1. In the Project tree, right-click the icon of the solution option you want to rename. The name next to the icon becomes editable.
2. Make the desired change to the name.
3. Press **Enter** to save the change.

Related Topics

Adding Solution Options

Editing Solution Options

Copying and Pasting Solution Options

Deleting Solution Options

Deleting Solution Options

To delete solution options:

1. In the Project tree, right-click the icon of the solution option you want to delete.
2. Select **Delete** to delete the option.

> **Note:**
>
> A warning message displays if the option is in use. If you choose to delete the in-use option, the analysis that is using it reverts to the default solution option settings.

Related Topics

Adding Solution Options

Editing Solution Options

Copying and Pasting Solution Options

Renaming Solution Options

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Importing Solution Data (.sdb format)

Twin Builder stores simulation solution data in .**sdb** files. You can create reports to import and view the solution data in these files.

To import solution data:

1. Select **Twin Builder** > **Import SDB File**. A file **Open** dialog box appears.
2. Locate the Twin Builder solution **.sdb** file containing the solution data you want to import, and click **OK**.

   > **Note:**
   >
   > Typically, the .**sdb** file is stored in the *yourproject***.aedtresults/***yourdesign***/temp** folder right after your simulation. After you save the project, it is moved to the upper-level folder *yourproject***.aedtresults/***yourdesign*.

3. Expand **Analysis** in the Project tree. Verify the presence of the imported solution.

Related Topics

Plotting Imported Solution Data

Importing a Solution Data File (non-.sdb format)

Importing a Solution Data File (non-.sdb format)

To import solution data in a file format other than .**sdb**:

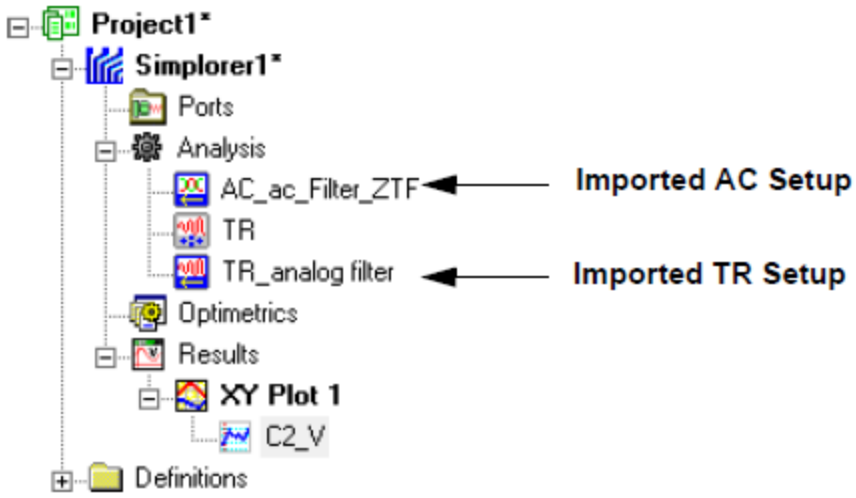
1. Select **Twin Builder** > **Import Data File...**, or right-click the Analysis icon in the Project Manager tree and select **Import Data File...**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. A file browser window appears. Choose to import data from the following file types:

.mdx, .mda	Twin Builder characteristic format.
.xls, .xlsx	Microsoft Excel.
.txt	Text file.
.csv	Comma-separated value.
.out	Maxwell SPICE (read-only – reads data inside the KW_DATA section).
.cfg	Comtrade (IEEE Std C37-111-1999).
.dat	TEK Oscilloscope.
.tab	Tab-delimited data files.

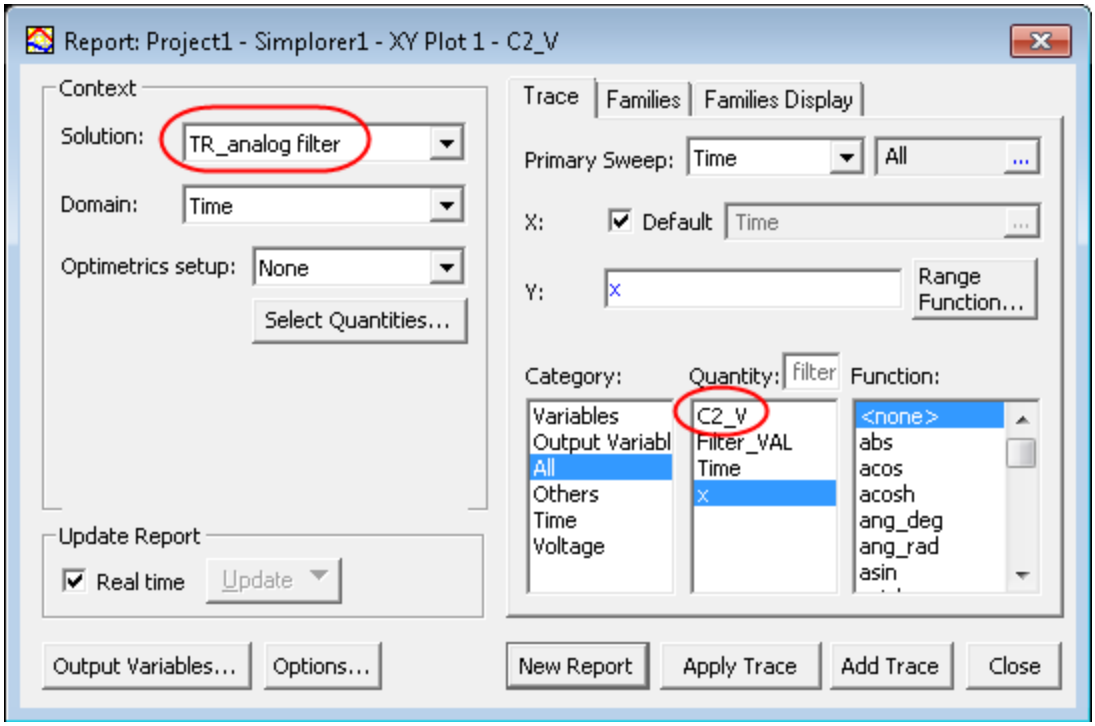
An **Analysis Setup** is created with the appropriate imported setup icon in the Project Manager.



- The imported solution setup name is either **TR_** or **AC_** plus the base file name.
- If the setup type is transient (TR); a Time channel is required. It will be the first channel, if that data is monotonically increasing, or it will be generated from the default transient end-time (40ms) and the number of data elements per channel.
- Data is considered AC if the first channel is “f”; and subsequent pairs are enclosed in either **Abs()** and **Phi()**, or **Re()** and **Im()**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Periods in names are converted to underscores.



- Channels may be labeled with channel names and units.
- Data channels from the imported file are available for plotting.

Related Topics

Plotting Imported Solution Data

Importing Solution Data (.sdb format)

File Formats

File Formats

This section contains descriptions of the file formats used by Twin Builder.

- MDX File Format
- CSV File Format
- XLS and XLSX: Microsoft Excel Format
- COMTRADE File Format

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

MDX File Format

Note:

You can convert most MDX and similar data files to datasets (see Importing Datasets). Project datasets are stored with the project, eliminating an external file reference. Project datasets used in designs represented by hierarchical components are saved to the library with the component and restored when you use the component (see Exporting Hierarchical Components).

The .**mdx** file format is a Twin Builder internal data format used to exchange data between the different programs. Create and edit the files with a text editor. The semicolon at the end of each dataset is essential, including the last one.

Data files in .**mdx** format are structured in a table-like fashion. The datasets are interpreted as a sequence of k-dimensional value tuples (ai,bi,ci,….) with a record length n (that is, data field/table of *k* columns multiplied by *n* rows). The correlated values of a k-tuple are arranged successively in the dataset:

(a1,b1,…,z1, a2,b2,…,z2, … ,an,bn,…,zn)

_____/ _____/ … _____/

1st k-tuple 2nd k-tuple nth k-tuple

In a typical case (time functions, characteristics) the first value in the n-tuple (ai) represents the common abscissa (“x”, time “t”…) where the datasets are in ascending order of ai ("x", "t", …).

Each file starts with header information for the data sets. The header always terminates with the “END:=” statement. All data within a line is separated by a semicolon; a line must be closed by a semicolon and CR/LF.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
TYP:=A
DIM:=3
LEN:=10
N1:=C2.V
U1:=V
N2:=Filter
U2:=
N3:=x
U3:=
END:=
0; 0.00001600; 0.00000400;
0.00001587; 0.00004787; 0.00001998;
0.00003168; 0.00009548; 0.00005188;
0.00006323; 0.00015872; 0.00009959;
0.00011041; 0.00023744; 0.00016299;
0.00019170; 0.00033153; 0.00024193;
0.00026967; 0.00044087; 0.00033631;
0.00036288; 0.00056532; 0.00044599;
0.00047123; 0.00070470; 0.00057084;
0.00061599; 0.00085909; 0.00071075;
```

The specifications N*xx* and U*xx* for names and units of data channels are optional and can be omitted.

Data Set Parameters

The required parameters are (in this order):

TYP:=	The data file type. • 4 4-Byte-Real data The datasets are read-only via the separate conversion program. • A ASCII data
DIM:=	Dimension *n* of the file (number of columns of the table). Usually in a simulation data file this corresponds to the number of data channels.
LEN:=	The number of data records (for ASCII files = number of rows). In a simulation data file, this is equal to the number of time steps calculated.
END:=	Close of data identification set.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The optional parameters (in arbitrary sequence and selection) are:

SEP:=	A separator between the single data values of ASCII sets. This specification is necessary only when selecting a separator different from the default semicolon character (";"). The separator may be enclosed by quotation marks (SEP:=/ or SEP:="/").
COD:=	The kind of data (3 Characters).
DAT	Time domain data.
NAM:=	Data file name (max. 64 characters), for example, used for titles in graphics.
USR:=	User name (max. 64 characters allowed).
NEW:=	Date of creation (DD.MM.YY).
MOD:=	Date of latest modification.
TIM:=	Time of creation (HH:MM:SS).
Nxx:=	Name of the *XX*th quantity in the data record (the first 14 characters are valid in DAY); can be used, for example, for automatic axes labeling; XX = '01' … n (n - data set dimension); leading zeros must be specified for XX.
Uxx:=	Unit of the *XX*th quantity in the data record - analogous to 'nXX' (max. 6 characters valid); may be any valid unit string in Twin Builder. Values are internally converted to SI and scaled appropriately.
Optional supplementary parameters, especially for measured transient recorder data:	
DX2:=	2nd sampling rate when dual time base is used.
The header is always terminated with “END:=”.	

Data Set Format

- *n*-dimensional value sets are possible. Each record containing *n* values must be positioned on one line (limited to 255 ASCII characters per record).
- The k values are separated by a separator character (also after the last value per line) usually the semicolon (;) - space characters are then ignored.
- All other special ASCII characters are also possible separators, except for the decimal point (.) and the space.
- If the separator is not the semicolon, the separator must be specified in the data set identification file as the SEP:= parameter, for example, SEP:="/".
- You can write the data in any standard number format used in data processing, such as: 10/ 10.0/ 1.0E1.

.mdx Files used for Components using 2D Characteristics

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
typ:=A
dim:=2
len:=6
end:=
0;0;
0.2;0.01;
0.5;0.02;
0.8;0.03;
1;2;
1.5;200;
```

.mdx Files used for Fourier Sources

```
typ:=A
dim:=3
len:=6
n01:=f
u01:=Hz
n02:=Amplitude
n03:=Phase
u03:=rad
end:=
50;0;0;
50;1;0;
100;2;0;
150;3;0;
200;4;0;
250;5;0;
```

.mdx Files used for 3D Lookup Tables

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
typ:=A
dim:=3
len:=12
n01:=V
n02:=T
n03:=A
end:=
0;0;0;
1;1;0;
2;2;0;
0;0;0.2;
1;2;0.2;
2;4;0.2;
0;0;1;
1;4;1;
2;6;1;
0;0;1.2;
1;6;1.2;
2;8;1.2;
```

.mdx Files used for DES Models

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
Ord=2
Dim=2

Name=DES1

M_0_0_0=C1
M_0_0_1=-C1
M_0_1_0=-C1
M_0_1_1=C1

M_1_0_0=Ki1
M_1_0_1=-Ki1
M_1_1_0=-Ki1
M_1_1_1=Ki1

M_2_0_0=J1
M_2_1_0=J3
M_2_1_1=J2

RS_0=Jump1
RS_1=0

IC_0_0=5
IC_0_1=4
IC_1_0=-4
IC_1_1=-5
```

CSV File Format

ASCII text file with real-valued channel data. For *n* channels, there are *n* values in a line, separated by a separator character, and terminated with a newline (or carriage return and newline). The first line may contain channel names and units, separated in the same manner as the numeric data. If units are present, they may be any valid unit string in Twin Builder, and they must follow the channel name and be enclosed in square brackets. The channel name and unit must together be enclosed in double quotation marks.

Example:

"Time [ms]", "R1.V [V]", "C1.I [uA]"

0,4.2,2.0

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1.33,4.1,1.95

2.0,3.9,1.8

Complex data is handled as follows:

- The first channel name must be "**f**", and may have units.
- Following channels are in pairs, either
  - "**Re***(name) [units]*", "**Im***(name) [units]*" where **Re()** and **Im()** are for complex data in real and imaginary form, or
  - "**Abs***(name) [units]*", "**Phi***(name) [units]*", where **Abs()** and **Phi()** specify complex data in magnitude and phase.

Name is the channel name, the same for each pair of channels. *Units* are optional, *SI* if none specified.

Example (comma-separated):

"f [kHz]", "Abs(x)[mV]","Phi(x)[rad]"

XLS, XLSX: Microsoft Excel Format

As with CSV files, the first row in a sheet may contain channel names and units. Unlike CSV files, the name and unit string are not enclosed in double quotation marks. For non-frequency based data, units may be enclosed in parentheses. For example:

Time (ms)	**R1.V (V)**
0	4.2
1.33	4.1
2.0	3.9

Complex data is handled as follows:

- The first channel name must be "**f**", and may have units.
- Following channels are in pairs, either
  - **Re***(name) [units]*, **Im***(name) [units]* where **Re()** and **Im()** are for complex data in real and imaginary form, or
  - **Abs***(name) [units]*, **Phi***(name) [units]*, where **Abs()** and **Phi()** specify complex data in magnitude and phase.

Name is the channel name, the same for each pair of channels. *Units* are optional, *SI* if none specified. Square brackets must be used if units are desired for complex data.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Example:

f [kHz] Abs(a) [mV] Phi(a) [deg]

COMTRADE File Format

IEEE Standard C37.111-1999, Common Format for Transient Data Exchange for Power Systems. This is a set of two or more files (***.cfg**, ***.dat**, and optionally ***.hdr**, ***.inf**) designed to contain oscilloscope data.

Setting the Outputs for Simulation

For a given simulation, you typically have an interest in looking at specific conditions in the schematic. You specify the outputs for a simulation with the **Output**dialog box.

After running a simulation, your specified outputs appear in the **Report** dialog box **Trace** tab for the corresponding Category, such as Voltage or Current; they also appear when you create a Plot-On-Schematic. The first item in the list is also used by default when you add an XY Report to a schematic. Note that outputs that are Boolean (enum-type) can only be plotted in digital plots or in data tables.

The **Select Quantities** dialog box, used to make additional quantities available for plotting, is functionally similar to the **Output** dialog box.

1. Select **Twin Builder** > **Output Dialog**. The **Select Quantities** dialog box appears.
2. The dialog box has three fields.
   - **Defined Output** – This panel contains a list of the outputs defined for the current simulation. Each item listed includes the name of the item in the schematic and a period-delimited abbreviation for the output property defined as an output. For example, **CD.V** refers to a component named **CD** and the **V** (for voltage) property for that component. To remove items from the list, select them and click . Make multiple selections by dragging the cursor. If the list exceeds the size of the window, a scroll bar appears.
   - **Search** – For large schematics, it may be easier to search for defined outputs by name. **Search** highlights items matching each successive number or letter you add in the text box.

     > **Note:**
     >
     > Boolean (enum-type) outputs can only be plotted in digital plots or in data tables.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Add/Remove** – This panel contains hierarchical lists of the nets and elements of the design, including submodel properties for VHDL-AMS, SPICE, and SML submodels. You can specify the kinds of properties that appear in the list using the check boxes to show: **Outputs** (blue), **Inputs** (red), **InOuts** (brown), **Subcircuits**, and **Derivatives**. Displayed properties include parameters, quantities (including those defined in VHDL-AMS component architectures), and signals.

  Click **+** to open the lists of nets or elements. Click the lower-level **+** to open lists of potential outputs. Click **Expand All** and **Collapse All** to expand or collapse all of the elements in the list tree. Each potential output has a check box that you use to select it. Each potential output corresponds to a property available for that component or net.

  Click **Select All** and **Select None** to select and add (or remove) all defined outputs for the element or net currently selected in the list.

  Select the check box next to an element in the list tree to add it to the **Defined Output** list.

3. After making your selections, click **OK** to accept the defined outputs or click **Cancel** to close the dialog box without accepting the changes.

> **Note:**
>
> The maximum array expansion limit for outputs of the array type is set on the **General** tab for Twin Builder on the **Options** dialog box.

Creating Reports

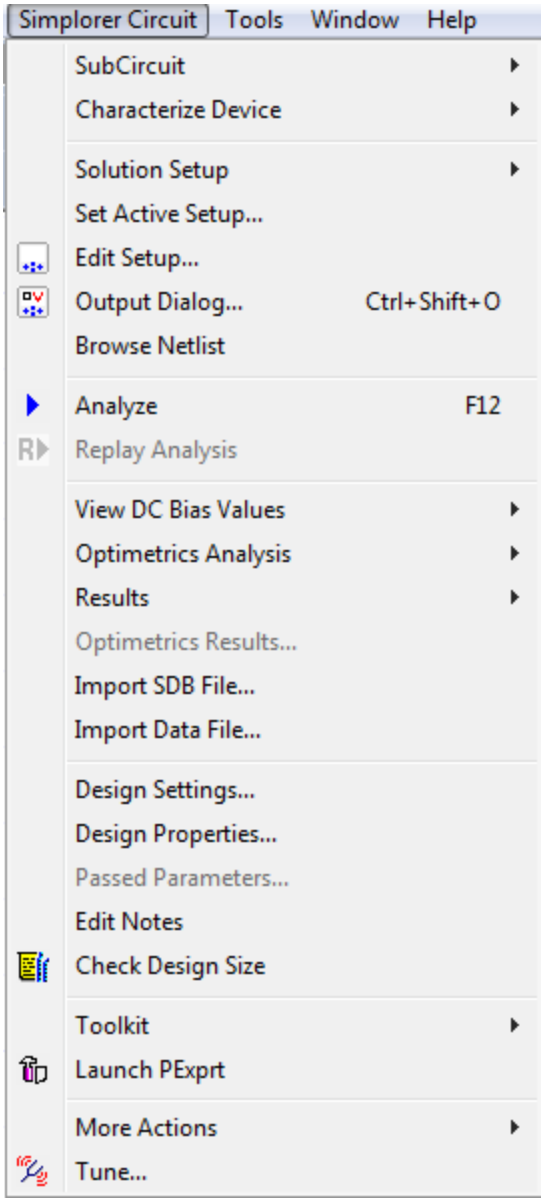
In addition to setting the outputs, you can create and modify simple reports using these buttons in the **Selected Traces** section:

- **Apply Trace** – Adds selected traces to a report and deletes traces that are used in an expression. For example, if you add a trace with the expression **E1.V+E2.V**, **E1.V** and **E2.V** are deleted.
- **Add Trace** – Adds selected traces to a report. Select one or more traces from the **Defined Output** section, or manually add them in the **Selected Traces** section. Select the target report from the report tree and click **Add Trace**.
- **Delete Trace** – Removes selected traces from a report. Select a trace in the **Selected Traces** section and click **Delete Trace**.
- **New Report** – Creates a new report. Click **New Report** and select a report type from the drop-down list. Click **Add Trace**, **Apply Trace**, and **Delete Trace** to add or remove traces to the report.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Checking the Design Size

Determine the number of models in a simulation by selecting **Twin Builder** > **Check Design Size**.



The following models are counted:

- INTERN – Electric circuit models, block diagram models, characteristic components, time functions, Fourier source.
- UMODEL – C-Models.

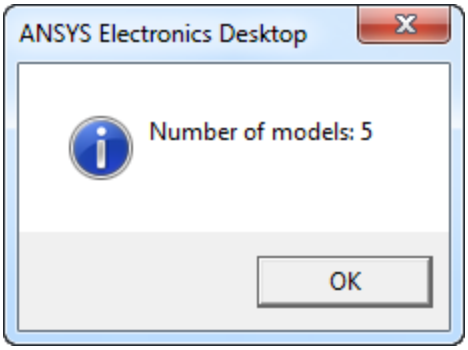
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- MODEL – Structural model definition (as 1) and its children.
- COUPL – VHDL model definition (as 1) and its children.
- Graphical subsheets (as 1).
- External coupling components (as 1).
- Modelica models (as provided by the Modelica compiler).

The following models are not counted: FML, FML_init, state graph states, and state graph transitions.

To check the number of models in the design:

1. Select **Twin Builder** > **Check Design Size** or click on the toolbar. The number of models is displayed.



2. Click **OK**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

17 - Running Simulations

After you specify how Twin Builder is to compute solutions, you can begin the solution process. In general, **Analyze** applies to the selected setup and associated sweeps, if any, or to a select sweep. To use this command, right-click a setup or sweep in the Project tree, and select **Analyze**. **Analyze All** applies to all enabled setups at or below the level invoked in the Project tree.

Related Topics

Analyzing Twin Builder Designs

Solving a Single Setup

Running More Than One Simulation

Specifying the Analysis Options

Using Replay Analysis

Remote Analysis

Monitoring and Controlling the Solution Process

Large-Scale DSO for Parametric Analysis

Aborting Analyses

Solving a Single Setup

To solve *a single specific solution setup*:

1. Right-click a solution setup in the Project tree. Standard Analysis setups can be either transient, AC, or DC. Optimetrics setups can be either parametric, optimization, sensitivity, or statistical.
2. Click **Analyze**, or press **F12** to start an analysis.

   Twin Builder compiles a netlist for the project, then computes the solution.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Warning:**
>
> By default, unconnected component pins (terminals) generate errors (displayed in the **Message Manager** pane) when a netlist is generated. Simulation is not possible until the errors are corrected.
>
> You can change this default component behavior Using the Component Editor so that either no action is taken when unconnected terminals are netlisted, or unconnected terminals are grounded.

To solve *two or more sweeps or two or more parametric analyses under a setup*:

1. Configure two or more machines for a distributed analysis. See Solving Remotely for configuration issues, and Configuring Distributed Analysis for setting up the Twin Builder general options.
2. In the Project tree, under the design you want to solve, right-click **the setup** icon that includes the sweeps of interest, then click **Analyze**.

Each solution sweep under that setup is solved in the order it appears in the Project tree, using the available machines.

Related Topics

Analyzing Twin Builder Designs

Running More Than One Simulation

Specifying the Analysis Options

Twin Builder Options: General Options Tab

Using Replay Analysis

Remote Analysis

Monitoring and Controlling the Solution Process

Aborting Analyses

Running More Than One Simulation

To run more than one analysis at a time, follow the same procedure while a simulation is running. The next solution setup will be solved when the previous solution is complete.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To solve every solution setup in a *design*:

1. In the Project tree, right-click the design you want to solve.
2. Select **Analyze**.

   Each solution setup is solved in the order it appears in the Project tree.

To solve every solution setup in a *project*:

1. Either click **Project** > **Analyze All**, or right-click the project icon in the Project tree and select **Analyze All**.

   Each solution setup is solved in the order it appears in the Project tree.

Related Topics

Analyzing Twin Builder Designs

Solving a Single Setup

Specifying the Analysis Options

Twin Builder Options: General Options Tab

Using Replay Analysis

Remote Analysis

Monitoring and Controlling the Solution Process

Aborting Analyses

Parametric Analysis of Simulation Models

When a component contains multiple models, set up a parametric analysis using the component's **SimulatorModel** property. Twin Builder references the index of model choices for this type of analysis.

Follow this procedure to set up a parametric analysis on a component with multiple models.

1. Load a Twin Builder project.
2. Right-click a component and select **Edit Component**. The **Edit Component** dialog box appears.
3. Select the **Simulation Models** tab, then click **Add Model** to select models to add to the component. See ***Simulation Models Tab: Edit Component*** for more information.
   - The multiple models must have the same conservative (terminal) interfaces. They can have different non-conservative interfaces.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Netlist line-based components are not supported at this time.
- You could also use the **Import Components** dialog box to import your models. See ***Importing Twin Builder Models*** for more information.

4. To run the parametric analysis in Optimetrics, open the Twin Builder project, then in the schematic area, right-click a component with multiple models and select **Properties**. The component **Properties** dialog box appears.
5. Select the **Parameter Values** tab.
6. Click the **SimulatorModel** row, then select the **Sweep** check box. Click **OK** to save your changes and close the dialog box.
7. Right-click **Optimetrics** in the Project tree, then select **Add** > **Parametric**. The **Setup Sweep Analysis** dialog box appears.
8. Click **Add**. The **Add/Edit Sweep** dialog box appears.
9. Select **SimulatorModel** from the **Variable** drop-down list. See ***Adding a Variable Sweep Definition*** for information on adding sweep (variation). Click **OK** to save your changes and close the **Add/Edit Sweep** dialog box.
10. Click **OK** to close the **Setup Sweep Analysis** dialog box and return to the Twin Builder schematic.

> **Note:**
>
> You can add multiple simulation models to perform a parametric sweep. These models can have different non-conservative interfaces, but they must have same conservative interfaces (terminals). In there are different non-conservative interfaces, use local variables to create a sweep for these interfaces. If a sweep flag of interfaces was used to create a parametric sweep, then it may result in errors. The solver will not be able to set variation when interfaces are not available in other models of same component.

Using Replay Analysis

Replay analysis provides the ability to replay plots and symbol animation at different speeds without the need for re-simulation. It also provides the ability to drag the slider to any intermediate point on the simulation time line with dynamic plot update to that point. You can also use replay analysis to debug state graphs and other intermediate signal changes.

Simulation data must be present, or the design must have been analyzed before running replay analysis.

The following schematic objects are updated to their current simulation state during analysis replay.

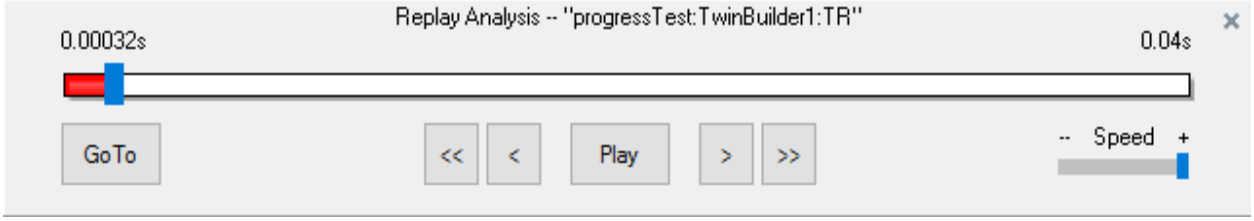
- Dynamic plots
- Symbol animation

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Output property displays

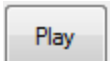
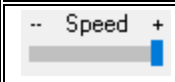
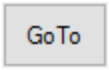
To replay an analysis:

1. Select **Twin Builder** > **Replay Analysis**, click , or in the Project tree, right-click the analysis and select **Replay Analysis**. The Replay Analysis time line appears in the progress window.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Perform the action you want using the following controls.

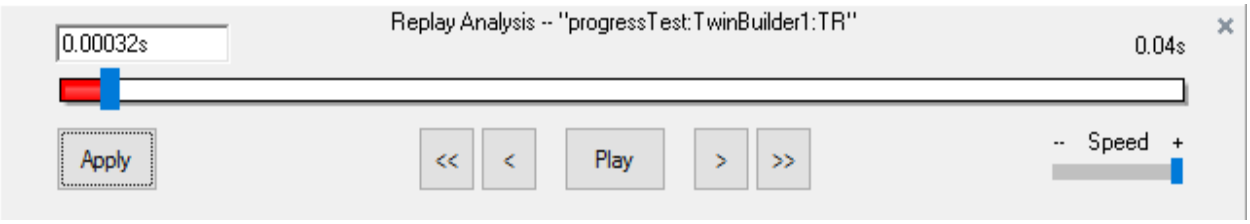
Control	Action
	Click to move 10 frames backward. You can also press **Page Up**.
	Click to move one frame backward.
	Click to play the simulation. If you have moved the slider, the simulation starts at that position. After you click this button, it changes to **Pause**. Click **Pause** to pause the simulation. If you click the slider after the replay has finished, the button displays **Replay**. Click **Replay** to replay the simulation.
	Click to move one frame forward.
	Click to move 10 frames forward. You can also press **Page Down**.
	Drag the slider to adjust the speed of the replay.
	Click to manually enter a replay start time.
	Click to go to the manually entered replay start time.
	Enter a replay start time.

To manually select replay start time:

1. Click **GoTo**.
2. Enter the desired replay start time.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Click **Apply**.



You can also drag the slider to a position and click **Play**.

In addition to using the buttons, click the slider and use defined keys for the following actions.

Key	Action
left arrow	Press to move one data point backward.
right arrow	Press to move one data point forward.
Home	Press to move to the start point.
End	Press to move to the end point.
PgUp	Press to move 10 frames backward.
PgDn	Press to move 10 frames forward.

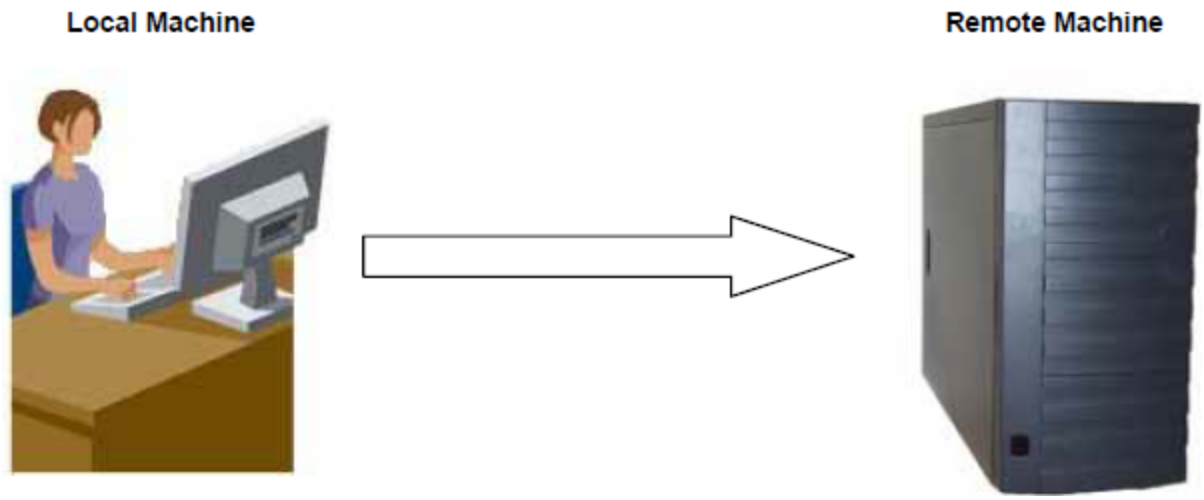
Click to close the **Replay Analysis** dialog box.

> **Note:**
>
> You can simultaneously replay more than one analysis as long as the analyses are not part of the same design. If you replay another analysis of the same design, the active panel closes before the new one starts.

Remote Analysis

It is possible to solve a project on a different machine from the one on which you set up your designs. This is particularly useful when you want to take advantage of a more powerful machine but it is not convenient to access that machine. This process involves configuring the machine that is to perform the solving (the remote machine), as well as the machine from which the simulation is to be launched (the local machine). This can also be extended into distributed analysis, where a specified analysis, if supported, is concurrently solved on multiple machines.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

Communication between machines in remote analysis and distributed analysis can drastically affect performance. Use a high-speed network system, like Gigabit or Infiniband, for optimal performance.

- Prerequisites for Remote and Distributed Analysis
- Configuring the Local Machine to Solve Remotely
- Remote Analysis Options
- Running Remote Analysis

Select **Tools** > **Options** > **Export Options Files** to write XML files containing the Options settings at all levels to the specified directory (default, **%UserProfile%\Documents\Ansoft**). **Tools** > **Options** > **Export Options** makes it easier for different users to use Ansys Electromagnetics tools installed on shared directories or network drives. The Example Uses for Export Options Features section outlines some use cases enabled by this feature.

Prerequisites for Remote and Distributed Analysis

1. You must have Ansys Electromagnetics' Remote Simulation Manager (RSM) or a supported High Performance Computing (HPC) management software program. (See ***High Performance Computing (HPC) Integration***.) The list of currently-supported HPC software includes:
   - Altair's PBS
   - Sun GridEngine
   - Microsoft® Windows® Compute Cluster Server 2003
   - Microsoft® Windows® HPC Server 2008 R2 and HPC Server 2012

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Twin Builder must be accessible from all remote machines as well as accessible on the local machine.
3. If you use RSM, it must be accessible from all remote machines. In addition, the engines must be registered with each initialization of RSM. To do this, on each remote machine:
   - On Windows on the local and remote machines, click **Start** > **Programs** > **Ansys EM Suite** *[version]* > **Register with RSM**. You can also run **RegisterEnginesWith RSM.exe**, located in the product installation folder (for example, **C:\Program Files\AnsysEM\\<*version*>\Win64\RegisterEnginesWithRSM.exe**).

   In each case, you see a dialog box confirming the registration. Click **OK**.

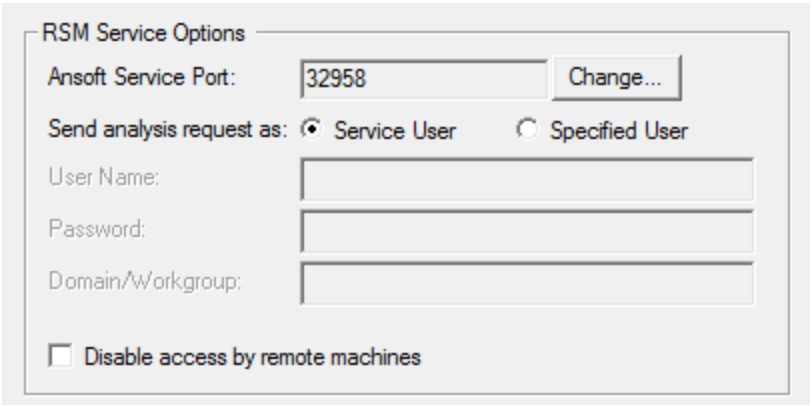
If the RSM service cannot run due to permission issues for the configuration file, it issues an error message and closes. If your product is not registered with RSM, the analysis will run locally.

Configuring the Local Machine to Solve Remotely

To set the analysis options in Twin Builder, see ***Configuring Distributed Analysis***.

Remote Analysis Options

Open the **Options** dialog box, then select **General** > **Remote Analysis**.



Select whether to run simulation processes as the user running RSM **Service User**, or a **Specified User**. If you select **Specified User**, you must provide the **User Name**, **Password**, and any **Domain/Workgroup** on which this user is defined. If the name or password is incorrect, the **Message Manager** pane displays a warning message, and the solver attempts to perform the analysis as the Service User.

Running Remote Analysis

When you run a simulation remotely, you should see a message in the Progress window identifying the design name, and the specified remote machine. You will see progress messages

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

as the simulations continues. When the simulation is complete, you will see a message in the **Message Manager** pane.

Related Topics

Distributed Analysis

Troubleshooting

Troubleshooting

Problem: When you try to solve from local to remote machine, a SIMPLORERLCOMENGINE process starts on the remote machine, but the Twin Builder user interface freezes.

This occurs when you enable the remote solve option after starting the COM daemon, or when you select **Don't allow exceptions** for the Windows firewall.

Resolution: Remote solve needs either firewall exceptions to be ON or firewall to be completely turned off.

Problem: When you try to solve from a local to a remote machine, you receive the following error message:

[error] Unable to locate or start COM engine on 'nomachine' : Unable to reach AnsoftRSMService. Check if the service is running and if the firewall allows communication. (10:57:13 PM Aug 13, 2019)

Resolution: This message can happen if the machine is not present, the network connection is down, if there are firewall issues or if the service is not running.

Remote Solve Node = Windows

Error: "Unable to locate or start COM engine on <remote node> : Unable to reach AnsoftRSMService. Check if the service is running and if the firewall allows communication."

1. Disable the firewall.
2. Confirm that you have not changed the Ansoft Service Port in **Tools** > **Options** > **General Options** > **Remote Analysis Options** from the default 32958. If you have, change it back to 32958, restart Twin Builder, and try to solve again.
3. Make sure that the local machine can contact the RSM port on the remote node. Open a command prompt on the local machine and type **telnet** *<remote node name>* **32958**. If the terminal appears to be hanging, then the connection was successful.
4. Check to make sure the Ansoft Communication Service is running. To do this, go to the **Windows Control Panel** and choose **Administrative Tools** > **Services**. Find the RSM service and make sure its status is **Started**. If it is not running, try to start it; right-click the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

service and choose **Start**. If it still does not start, check the user name/password combination listed in the **Log On** tab of the service properties.

5. Make sure the user listed in the service is an administrator.
6. Make sure the COM engine is registered with the RSM service. From the Windows menu, choose **Start** > **All Programs** > **Ansys EM Suite 2024 R2** > **Register with RSM** to register the engines.

Error: "Unable to locate or start COM engine on <remote node>: Engine is not registered with the RSM service which is running on this machine."

- To register the engine, from the Windows menu, select **Start** > **All Programs** > **Ansys EM Suite 2024 R2** > **Register with RSM**.

Distributed Analysis

Use distributed analysis to split certain types of analyses and solve each portion of an analysis simultaneously on multiple machines. Simulation times can be greatly decreased by using this feature.

Twin Builder supports one form of distributed analysis:

- Distributing rows of a parametric table.

> **Note:**
>
> Communication between machines in remote analysis and distributed analysis can drastically affect performance. Use a high-speed network system, like Gigabit or Infiniband, for optimal performance.

Related Topics

Configuring Distributed Analysis

Editing Distributed Machine Configurations

Selecting an Optimal Configuration for Distributed Analysis

Configuring Distributed Analysis

To configure distributed analysis, select a distributed machine configuration. This is a list of machines to use for a simulation, based on considerations such as whether the simulation is more memory intensive or more CPU intensive, relative to the resources available on your network. See ***Selecting an Optimal Configuration for Distributed Analysis*** for a discussion of issues. To create a new distributed machine configuration. or to edit an existing one, see ***Editing Distributed Machine Configurations***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

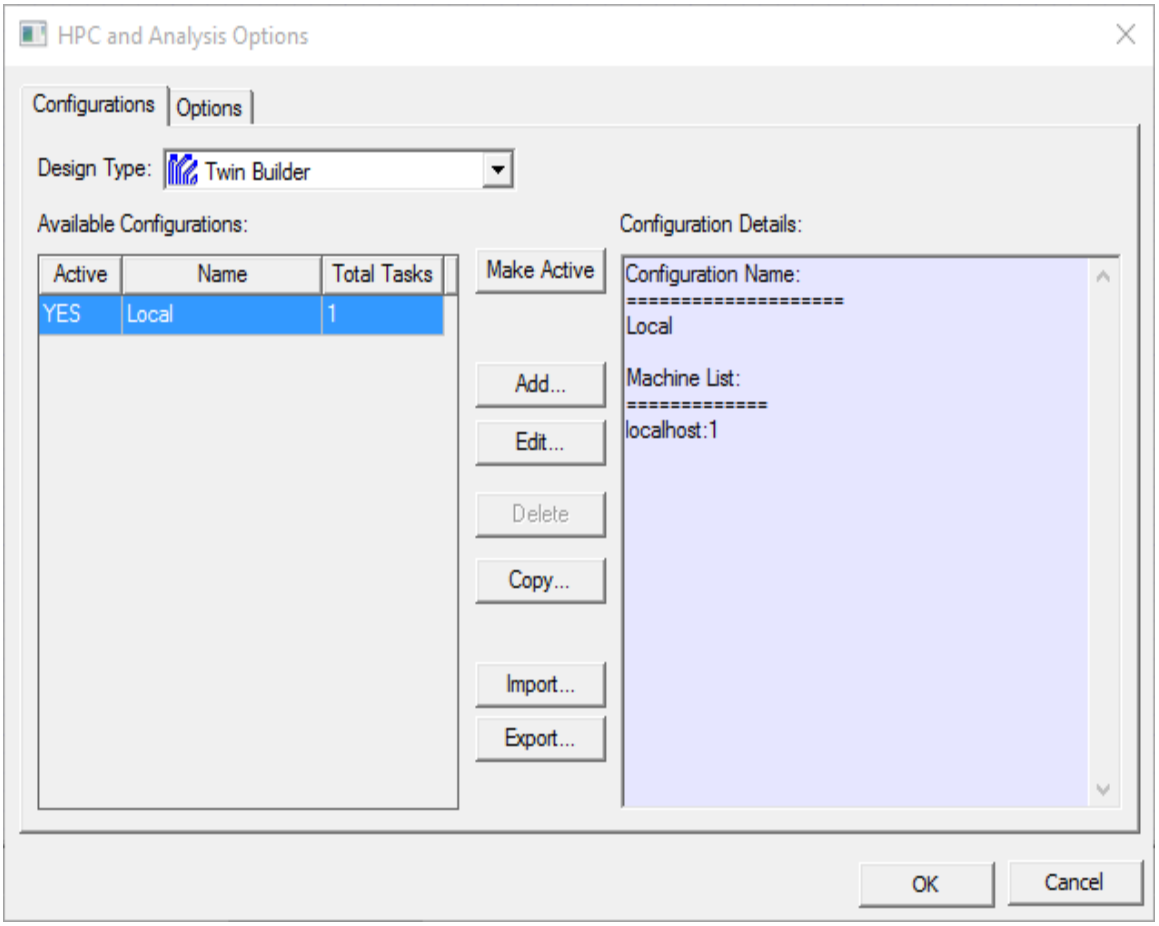
To select from an existing configuration:

1. Click the **HPC Options** settings icon on the **Simulation** ribbon or click **Tools** > **Options** > **HPC and Analysis Options**.

   This opens the **HPC and Analysis Options** dialog box. You can view a list of available configurations, as well as a report of the configuration details. From the current list, you can select a configuration to **Make Active**. You can also **Add** a new configuration, **Edit** an existing one, or **Export** as a Ansys Configuration file (*.**acf**). You can also **Import** a configuration file. This lists existing configurations, and shows all machines in the selected configuration, enabled or not. You can **Copy** an existing configuration, typically to edit the name and contents for other purposes.

   The **Options** tab lets you choose to queue all simulations.

   For a more detailed discussion of this dialog box, see ***Setting HPC and Analysis Options***.



To define a new configuration, click **Add** to open the **Analysis Configuration** dialog box. See ***Editing Distributed Machine Configurations***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

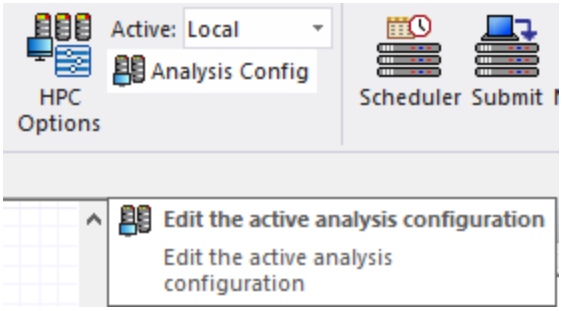
Editing Distributed Machine Configurations

Selecting an Optimal Configuration for Distributed Analysis

Editing Distributed Machine Configurations

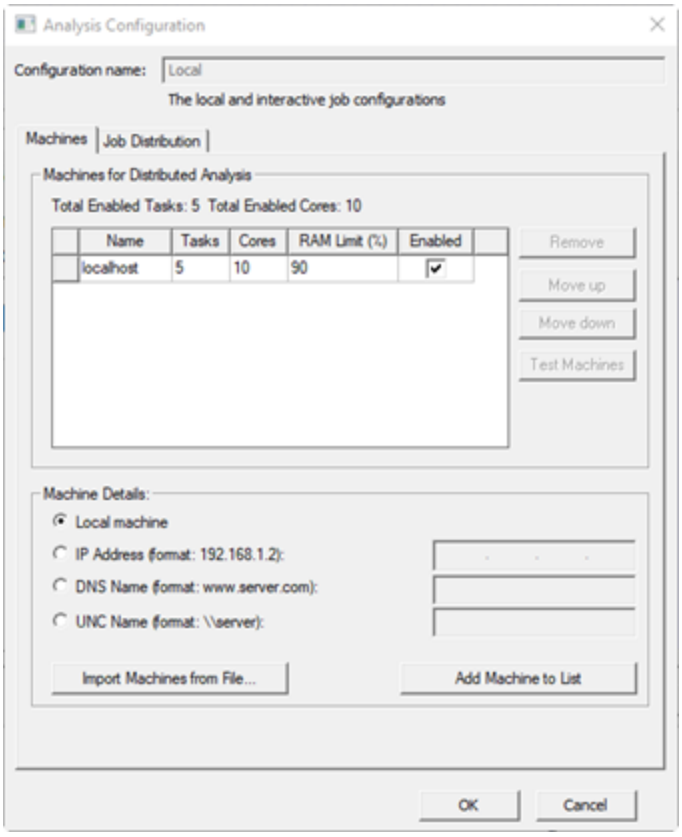
Follow this procedure to edit an existing machine configuration:

1. Select **Tools** > **Edit Active Analysis Configuration** to open the **Analysis Configuration** dialog box directly or select **Simulation** > **Analysis Config** on the ribbon. You can also open the **HPC and Analysis Options** dialog box and click **Add**, **Edit**, or **Copy**.



   If you select **Add** from the **HPC and Analysis Options** dialog box, the fields are empty. If you select **Edit** or use **Tools** > **Edit Active Analysis Configuration** or click **Copy** from the **HPC and Analysis Options** dialog box, the fields show the selected configuration.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. Specify the name of the new or edited configuration. The **Configuration name** field cannot be empty or contain a previously used name or a reserved word.
   - The **Machines** tab contains the machine list for the analysis configuration. Here you can provide machine information: you can either specify machine details and click **Add machine to list**, or import a list of machines from a file. You can remove, order, test, and enable machines on the list.
   - To add machines manually to the list, under **Machine Details**, specify an IP Address, DNS Name, or a UNC Name.

     > **Note:**
     >
     > The remote machines must have the same Ansys Electromagnetics Suite version installed in the same OS and version, as well as have the RSM service active.

     - You can choose to **Import Machines from File**. Each line of the file can contain a machine specifier of the form:

       `<MachineName>:<NumTasks>:<NumCores>.`

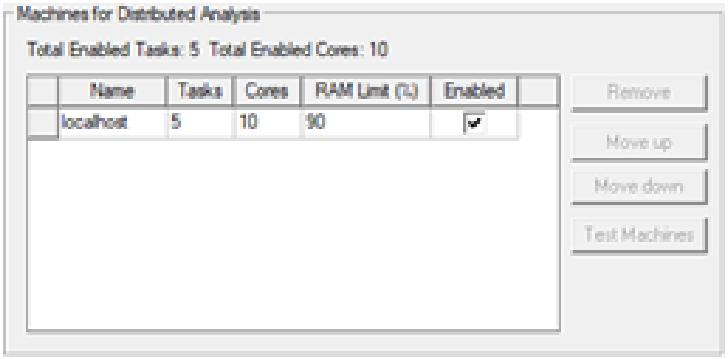
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> This same format is used with the **-machinelist file=** command line option.

- To streamline the common case of running jobs on the current machine, use the dedicated **Local Machine** option to specify the local machine for the list.

3. Machine List Details

   The **Machines for Distributed Analysis** area lists machines in the order in which you entered them, irrespective of the load on the machines. To control the list order, select one or more machines, and click **Move up** and **Move down**. To remove one or more machines, select the machines and click **Remove**.



- **Name** – The machine name.
- **Tasks** – This column allows you to specify the number of tasks a given machine will perform simultaneously. Each separate solver or instance counts as one task.
- **Cores** – Specifies the total number of cores that will be used on the given machine. The total number of tasks and cores are described just above the machine list. For distributed tasks, the software will allocate the total cores on a given machine to that machine's tasks. If a machine with 8 cores is running 2 distributed tasks, the software allocates 4 cores to each task. If it is running 4 distributed tasks, each gets 2 cores. And if it is running 3 distributed tasks, the first two tasks get 3 cores, and the last task gets 2 cores. The number of cores must always be greater than or equal to the number of tasks.
- **RAM Limit (%)** – Specifies the maximum percentage of each machine's RAM you would like to be in use by the solver.

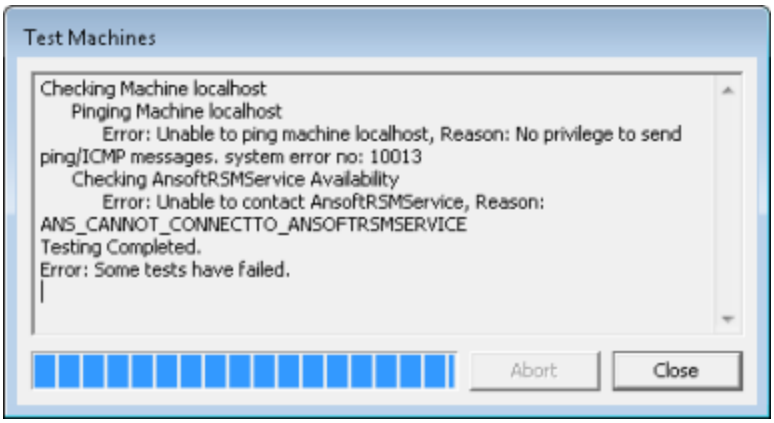
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> The settings for **Cores** and **RAM Limit** are used only for distributed solves that involve a Co-simulation. The Twin Builder solver will otherwise ignore those settings.

- Each machine on the current list has an **Enabled** check box. Here you can enable or disable the listed machines. Above the table, the dialog box displays a count of the total enabled tasks.

  In general, Twin Builder machines in the distributed analysis machines list are used in the order in which they appear. If you select a distributed configuration (rather than Local) from the ribbon and you launch multiple analyses from the same UI, Twin Builder selects the machines that are running the fewest number of engines in the order in which the machines appear in the list. For example, if the list contains four machines, and you launch a simulation that requires one machine, Twin Builder chooses the first machine in the list. If another simulation is launched while the previous one is running, and this simulation requires two machines, Twin Builder chooses the second and third machines from the list. If the first simulation terminates and you launch another simulation requiring three machines, Twin Builder chooses 1, 4, and 2 (in that order).

- **Test Machines** – When multiple users on a network are using distributed solve or remote solve, they should check the status of their machines before launching a simulation to ensure no other Ansys EM processes are running on the machine. To do this, you can select one or more machines and click **Test Machine**. The **Test Machines** dialog box opens.



  The test goes through the current machine list and gives a report on the status of each machine. A progress bar shows how far testing has gone. Click **Abort** to cancel a test. When the test is complete, click **OK** to close the dialog box. If you need to disable or

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

enable machines from the list based on the report, you can do so in the **Distributed Analysis Machines** dialog box.

Related Topics

Configuring Distributed Analysis

Editing Distributed Machine Configurations

Selecting an Optimal Configuration for Distributed Analysis

General Options: Analysis Options Tab

Selecting an Optimal Configuration for Distributed Analysis

For multiple DSO simulations on a single machine, the total memory needed is the sum of the memory used by each simulation. For example, assume you are running a parametric sweep and each run needs 3.5GB. Your computer has only 8GB of RAM but is a quad core system. To keep the computer from going heavily into swap, which is highly inefficient, you only want to assign this computer twice to the list of machines in the DSO setup. In addition, you would need to be sure that the amount of disk space available is also sufficient to fit all the requested simulations - typically hard drive space is not the limiting factor.

Related Topics

Configuring Distributed Analysis

General Options: Analysis Options Tab

Monitoring and Controlling the Solution Process

While a simulation is running, you can monitor the solution's progress in the Progress window. Above the green progress bar, messages describe the setup and step. The progress bar shows the relative progress of each step. Under the bar, messages note the part of the design being solved, and give memory estimates during the factoring process.

To view general information about standard analysis runs:

1. Right-click an **Analysis** solution in the Project tree and select **Profile**.
   - You can also right-click **Results** in the Project tree and select **Solution Profile**, or click **Schematic** > **Profile & Monitor** on the ribbon.

   A **Solutions** dialog box appears with the **Profile** tab selected.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Use the **Simulation** drop-down list to filter the profiles shown by analysis type.
- **Profile** data shown includes the starting time and date of the simulation run, the machine name on which the simulation was run, the Twin Builder version used for the simulation, the elapsed time for the simulation, the amount of memory used by the SimplorerServer process, the simulation completion time and date, and the status of the run (for example, Normal Completion). The profile is cleared when the solution is invalidated.

2. Optionally, click **Export** to export the profile data to a **.prof** file.
3. Select the **Runtime Simulation Monitor** tab to monitor simulation progress in real time. This panel can remain open while you run an analysis. It displays various simulation parameters depending on the analysis type.
   - For transient simulations, current time, end time, and min, max, and current time step values display.
   - For AC simulations, current and end frequency display.

   A graphical representation of the simulation iterations and time step (transient only) also appears.

To monitor the status of the solutions:

1. Right-click the **Results** icon in the Project tree and select **Browse Solutions**.

   The **Solutions** dialog box appears with two tabs: **Statistics** and **Browse**.

2. Click the **Browse** tab to display data about the passes completed.

   The left panel contains a tree structure showing the solutions listed according to setup, solution, and variation (if any). A table in the right panel lists the **Setup**, the **Solution**, the swept variable (if any), and the **State** of the solution.

   a. Use the **Optimetrics Setup** drop-down list to select the Optimetrics setup whose solution data displays. Choose **None** to display data for standard analyses (transient, AC, or DC).
   b. Click **Properties** to display a dialog box that lets you change the way the setup, solution, and variation appear in the tree structure.
   c. Select a solution from the table entries and click **Delete** to remove it. Use **Ctrl-click** to select multiple solutions, or **Shift-click** to select a range of solutions. Click **Select All** to select all solutions for deletion. When you delete a variation belonging to **None**, only that variation's solution file on disk is deleted. When a variation that is part of an Optimetrics analysis is deleted, the **.sdb** file containing all the variations corresponding to that Optimetrics solution will be deleted.
3. Click the **Statistics** tab of the **Solutions** dialog box to display file and path information, as well as file format, number of files, and file size.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Twin Builder Simulation Controls

Progress Bar Menu

Monitoring Solution Progress

Modifying Parameters During a Simulation

Twin Builder Simulation Icons

The Twin Builder simulation icons on the **Schematic** ribbon provide these simulation controls:

- **Edit Active setup** – Opens the setup dialog box for the currently selected standard setup.
- **Analyze** – Starts a simulation for the chosen active setup.
- **Profile & Monitor** – Opens the **Solutions** dialog box with the **Profile** tab selected and the current/appropriate **Simulation** Analysis Setup chosen.

> **Note:**
>
> You can also right-click **Results** in the Project tree and select **Solution Profile** to open the **Solutions** dialog box.

Progress Bar Menu

The progress bar context menu provides options for aborting, stopping, pausing, continuing, and resuming simulations, as well as for modifying various parameters during a simulation.

- To abort a simulation, right-click the **Progress** bar and select **Abort**.

  Twin Builder ends the analysis immediately.

> **Note:**
>
> - If you abort the solution in the middle of an adaptive pass, the data for that pass or current frequency point is deleted. Any solutions that were completed prior to the one that was aborted are still available.
> - The solutions that are available depend on when you abort the analysis. For example, if you stopped the solution while a post-processing macro was executing, the field solution computed for that setup is still available.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- To stop a simulation cleanly between time steps, right-click the **Progress** bar and select **Clean Stop**.

  Twin Builder ends the analysis after the next solved pass or frequency point.

> **Note:**
>
> Upon license failure, the simulation is clean stopped after the product has spent enough time waiting for the license to be available. Follow the instructions to save all data and exit the application, which results in simulation results being saved until the point the simulation was run.

> **Note:**
>
> If you request a clean stop between adaptive passes, the solutions for those passes will still be available.

- To pause a simulation, right-click the **Progress** bar, and select **Pause**.

  > **Note:**
  >
  > While the simulation is paused, you can update reports with the SDB data current at the time the simulation was paused. You can also selectively modify parameters if desired before resuming the simulation.

- To resume a simulation, right-click the **Progress** bar and select **Resume**.
- If you selected **Enable continue to solve** in either the Transient Analysis Setup or AC Analysis Setup, the simulation pauses at the currently set end time or stop frequency.

  > **Note:**
  >
  > If a simulation ends *before* the currently set end time (transient analysis) or stop frequency (AC analysis) is reached, the simulation won't pause and the Progress bar will close.

  To continue the simulation, right-click the Progress bar and select **Continue**. The transient or AC analysis setup dialog box opens in which you can set a new, larger-value end time or stop frequency to continue the simulation. Otherwise, choose **Stop** to end the simulation and close the Progress bar.

- If you delete either the Transient Analysis Setup or AC Analysis Setup and attempt to **Continue** simulation, a dialog box informs you of this fact and asks if you want to stop the simulation. Choose **Yes** to stop the simulation and close the Progress bar. Choose **No** to

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

undo the deletion of the setup and **Continue** the simulation.

- To modify various simulation and model parameters during a simulation, right-click the Progress bar and select **Change quantity values**.

  Twin Builder pauses the simulator and opens the **Set Assignments** dialog box in which you can modify parameters.

- To open the **Solutions** dialog box with the **Profile** tab selected and the current/appropriate **Simulation** Analysis Setup chosen, select **Profile...** on the context menu.

Monitoring Solution Progress

While a simulation is running, monitor the solution's progress in the **Progress** dialog box. Above the green progress bar, messages describe the setup and step. The progress bar shows the relative progress of each step. Under the bar, messages note the part of the design being solved, and give memory estimates during the factoring process.

Modifying Parameters During a Simulation

Twin Builder lets you pause an in-progress transient or AC simulation, modify the values of *unconnectedrealinput* properties, then resume the simulation using the new values. Modified parameter values affect only the current simulation run. Corresponding parameter (property) values in the *project* are not modified.

> **Note:**
>
> Output and InOut properties, as well as properties whose value is a variable or expression cannot be modified.

To modify simulation and model parameters while a transient or AC simulation is running:

1. Right-click the Progress bar and select **Change quantity values**. You can also click the Progress bar arrow to access the menu.

   Twin Builder pauses the simulator and opens the **Set Assignments** dialog box. The main panel displays a simulator tree containing a **Solution parameters** folder, a **Solution options** folder, a **Design variables** folder, and icons for each model instance used in the simulation. It may also contain a **Project Variables** folder if project variables exist in the project.

   Click **+** next to an item in the tree to expand the item to show its parameters for which new values can be set. Click **–** to collapse the item. You can also use the keyboard's right-arrow and left-arrow keys to expand and collapse items in the tree. Similarly, you can use

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the up-arrow and down-arrow keys to move up and down the tree to select items and parameters.

2. Select the parameter you want to modify.

   The parameter name you selected appears directly below the tree panel along with a text entry field showing the current value for the selected parameter.

   > **Note:**
   >
   > Parameter changes made in the **Set Assignments** dialog box will affect only the Twin Builder circuit and not the external model when performing transient cosimulation with other products such as Maxwell, Simulink, or Ansys RBD.

   a. You can modify the following parameters in the **Solution parameters** folder:
      - For Transient analyses – **Tend**, **Hmin**, and **Hmax**.
      - For AC analyses – **StartFrequency**, **StopFrequency**, and **NumberOfSteps**.

      For additional information on these parameters, see ***Transient Analysis Setup*** and ***AC Analysis Setup***.

   b. You can modify the following parameters in the **Solution options** folder:

      > **Note:**
      >
      > Parameters in the **Solution options** folder are listed using their netlist names. Each of these parameters corresponds to a setting found in the **Solution Options** dialog box. The following tables give the parameter name, the corresponding **Solution Options** dialog box setting name, and the range of values that can be entered for the parameter.

      Transient solution options:

Solution options parameter name (netlist name)	**Solution Options dialog box name**	**Acceptable Values**
ADSync	Analog/Digital synchronization	0=Hybrid (default) 1=Adaptive 2=Conservative
EqualSteps	Number of equal steps	positive integers
IEmax	Maximum current	positive real values >0

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Solution options parameter name (netlist name)	Solution Options dialog box name	Acceptable Values
	error	
Iteratmax	Maximum number of iterations	positive integers >=1
LDF	Local truncation error [%]	positive real values >0
PerformanceFactor	Simulator pivoting strategy	positive real values >=0, <=1 (0=Partial, 1=Complete)
RelTol	Relative tolerance [%]	positive real values >0
SamanskiiFactor	Samanskii factor	positive integers >= 1, <=100
Solver	Integration formula	0=Euler 1=Adaptive Trapezoid-Euler (default) 3=Trapezoid
StepAccelDamping	Step acceleration damping [%]	positive real values >0
Temp	Ambient temperature (Cel)	real values >0
VEmax	Maximum voltage error	positive real values >0

AC solution options:

Solution options parameter name (Netlist name)	Solution Options Dialog Name	Acceptable Values
EmaxAC	Maximum error [A]	real values >0
Iteratmax	Maximum number of iterations	positive integers >=1
Temp	Ambient temperature (Cel)	real values >0

c. You can modify properties for design and project variables, and for component properties as defined in their respective **Properties** dialog boxes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

d. Enter the new value for the selected parameter in the text field and click **Set** to send the new value to the simulation engine.

   The **Assignments** panel echoes the newly assigned values. Modify the parameter values listed in the **Assignments** panel. Reselect the parameter in the tree, enter the desired value, and click **Set** to update the listing.

3. Repeat steps **2** and **3** for any additional values you want to modify.
4. When finished, click **Continue** to close the **Set Assignments** dialog box. The simulation run resumes using the modified values.

> **Note:**
>
> The modified values will be used only in the current simulation run. Values must be re-entered for each new simulation.

Monitoring Queued Simulations

If you have multiple setups for a design, and have selected **Analyze All**, the simulations are queued until there is a machine available. Enable queuing in the **HPC and Analysis Options**: **Options** tab. If queuing is enabled and you run multiple setups, they are solved in the order that they appear in the Project tree. You can prioritize setups by changing the order in the queue.

1. To view the solution queue, click **Tools** > **Show Queued Simulations**.

   This displays a dialog box that displays each simulation and its current status.

2. To change the order of a simulation in the queue, select the simulation, and click **Move up** and **Move down** to effect the change.
3. To remove a simulation from the queue, select the simulation and click **Remove from Queue**.

Large Scale DSO for Parametric Analysis

Large Scale DSO for parametric analysis operates through a non-graphical batch application called **desktopjob**. You can run **desktopjob** to perform parametric analysis DSO. The command-line interface supported by this batch program is consistent with the command line used for regular DSO jobs.

Large Scale DSO is used for “large scale parallel” jobs, which either fail or scale poorly as regular DSO jobs. A Large Scale DSO job does not support the output of full parametric results, but produces “reduced” datasets corresponding to predefined Rectangular plots. The extracted columns of data are saved as .**csv** files. Typically, there is one .**csv** file per trace, per variation. These .**csv** outputs can be used directly in downstream applications (for example, Excel, or custom programs that parse .**csv** files). They can also be imported as dataset solutions for post

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

processing. Non-rectangular plots of the design (such as digital plots) are not extracted. In order to produce a new output you must rerun the analysis.

The basic process involves:

1. Prepare the project for Large Scale DSO Analysis.
2. Submit the Large Scale DSO Job through the Tools > Job Management menu, or via a command line.
3. Monitor the job's progress.
4. Post process the results.

For details, see:

- Prerequisites for Large Scale DSO
- Job Management Interface for Large Scale DSO
- Command Line Syntax
- Deployment/Configuration
- Results Database Organization
- Job outputs
- Job Monitoring
- Known Issues for Large Scale DSO
- Troubleshooting for Large Scale DSO

Related Topics

Running Twin Builder from a Command Line

Distributed Analysis

High Performance Computing (HPC) Integration

Large Scale DSO Theory

Prerequisites for Large Scale DSO

General Prerequisites

- Twin Builder must be installed on the cluster which runs either a supported scheduler or RSM.
- The cluster is compatible with large scale DSO requirements.
- There is a folder on a shared drive where the input projects are located.
- Every node of the cluster supports the disk space (in a temp directory) and memory requirements of multiple engines that run in parallel.

RSM Environment

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

On Windows, RSM is started as an admin account, rather than as a system account.

> **Note:**
>
> Large scale DSO does not support RSM service running with system login credentials.

On each machine of the cluster, **desktopjob** is registered with the RSM service using this command:

- Windows: *<installation-directory>/<platform>* **desktopjob.exe -regserver**

> **Note:**
>
> The use of 64-bit versions of Twin Builder must be consistent on each machine of the cluster.

Scheduler Environment

No extra configuration is needed.

Related Topics

Large Scale DSO for Parametric Analysis

Distributed Analysis

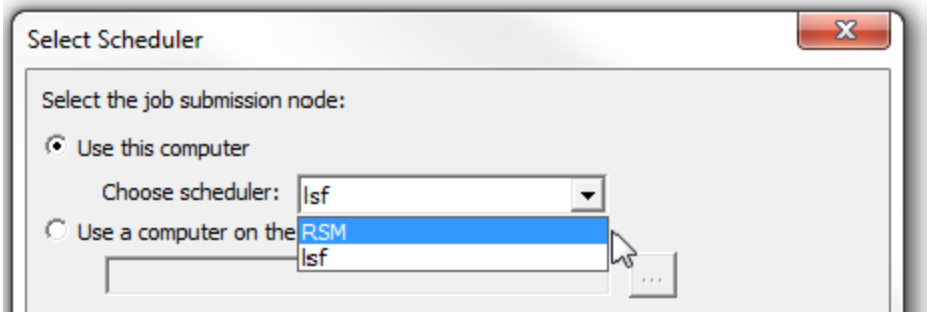
High Performance Computing (HPC) Integration

Job Management Interface for Large Scale DSO

Large scale DSO jobs run only in non-graphical batch mode, irrespective of the scheduler environment. This is in contrast to a regular DSO job, which, in an RSM environment, can be launched from a graphical desktop. This consideration implies that an input project corresponding to large scale DSO job must be saved and closed, prior to job submission. Secondly, the command to submit a large scale DSO job is only available through the **Tools** > **Job Management** menu or a window. Right-click the parametric setup to run a regular DSO job in the RSM environment. The Job Management UI is accessed by running Ansys Electromagnetics Desktop on the designated postprocessing node of the cluster. The Desktop provides UI commands for scheduler selection, job submission and job monitoring/control. Select **Tools** > **Job Management** > **Select Scheduler** to access the Scheduler User Interface.

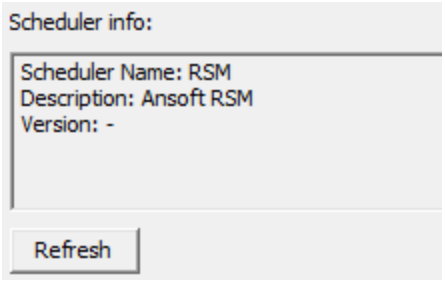
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Click **Select Scheduler** to display the selection dialog box. A drop-down list displays potential schedulers, which can include RSM, or lsf, depending on the environment.



If you select a scheduler that is not supported in your environment, you will receive a warning message.

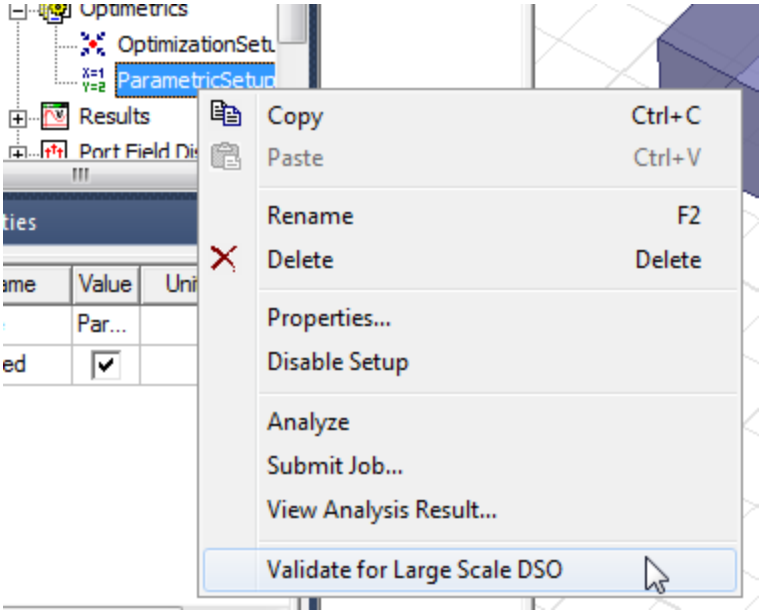
After selecting a scheduler, click **Refresh** to display information for that scheduler.



Once you have selected a scheduler supported in your environment, follow these steps to submit a large scale DSO job:

1. Set up and prepare the model on a local workstation
   - Launch Desktop. Open project, for example, **/home/projects/spool/test.aedt**.
   - Suppose the variations to solve come from **ParametricSetup1** setup. Right-click **ParametricSetup1** and run **Validate for Large Scale DSO**. Fix any validation errors.

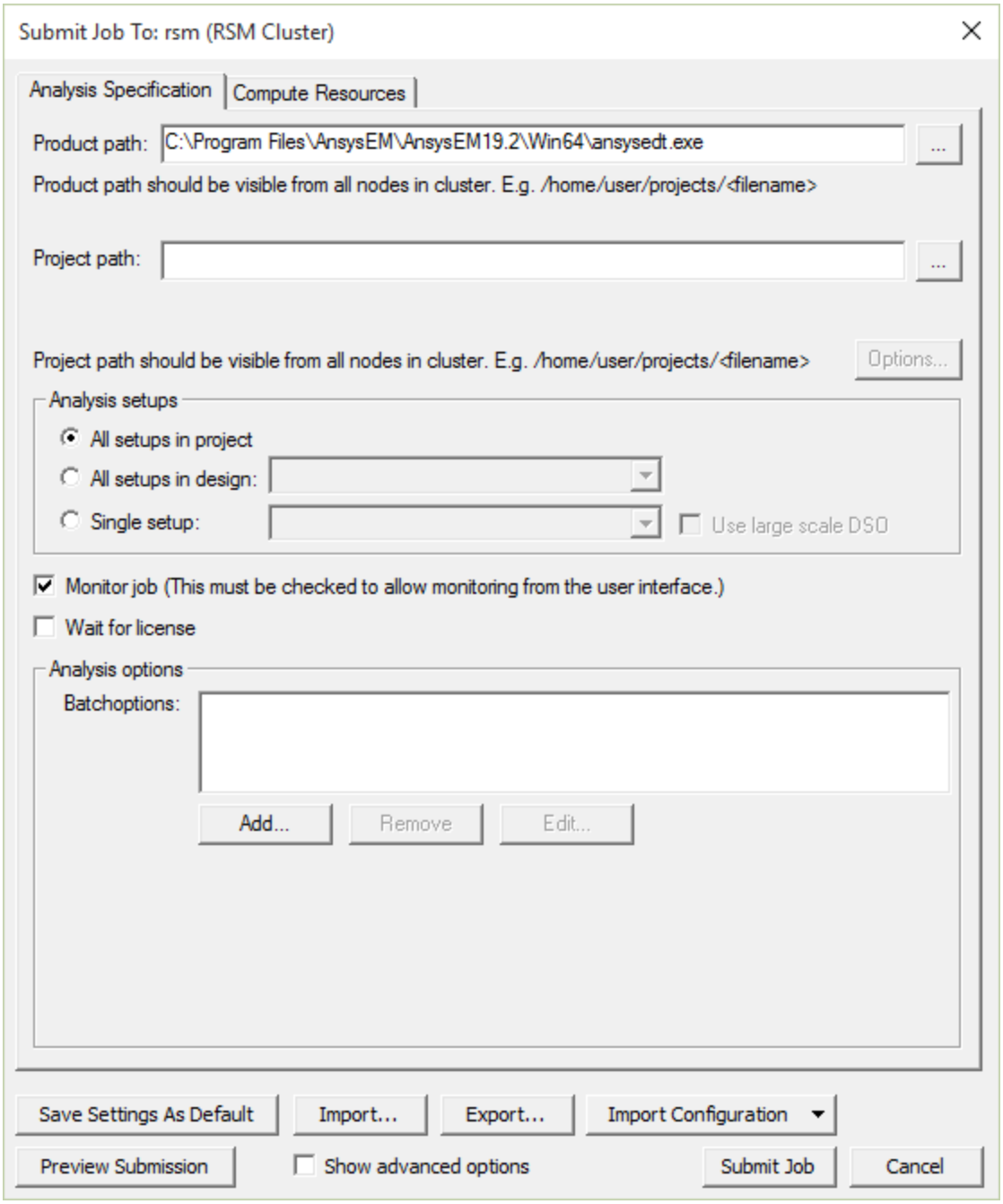
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- Save the project, in case of edits.
- Close the project.

2. Copy the input project (or folder, if the project references external files) from a personal workstation to a shared drive on cluster (say project is copied to **/home/projects/spool/test.adsn**).
   - In the RSM environment, you must specify a machine list. See ***Setting HPC and Analysis Options***. For example, if the machine list is: 3 cores from 'm1' and 3 cores from 'm2', for a total of 6 engines. Select the list on the **Compute Resources** tab described below.
3. Open a remote desktop session (or equivalent such as vnc session) on the node corresponding to the first machine of job's machine-list, 'm1' in this case. Launch Desktop graphically on 'm1'.
4. Optionally, double-check that the model is prepared correctly.
   - Open project **/home/projects/spool/test.adsn**.
   - Right-click **ParametricSetup1** and run **Validate for Large Scale DSO**. Fix any validation errors.
   - Save the project, in case of edits. Close the project
5. Run **Tools** > **Job Management** > **Submit Job**. The standard job submission panel displays.

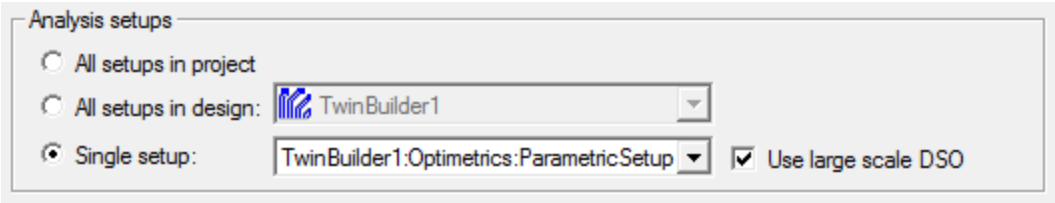
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



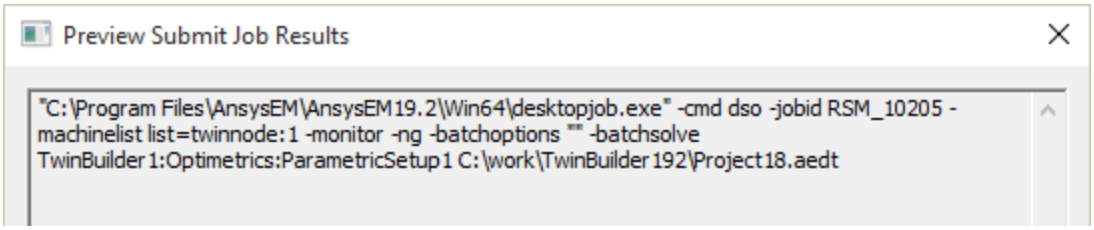
The panels for LSF and other schedulers have some differences. See Job Management User Interface for LSF.

- Enter all fields. Make sure to select **ParametricSetup***n* for analysis and select the **Use large scale DSO** check box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- To see the command line submitted to the scheduler, click **Preview Submission.** This opens a dialog box showing the command to send to the scheduler.

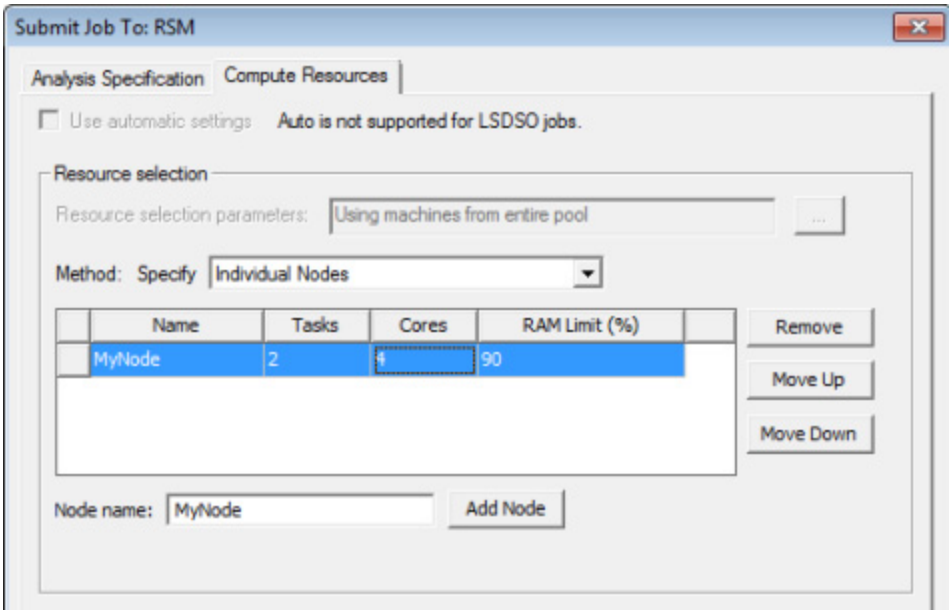


  The text can be copied to the clipboard.

- The **Batchoptions** field lets you add additional **-batchoptions** parameters.
- If you intend to monitor the job through a user interface, select Monitor job. You can then monitor this job by selecting **Tools** > **Job Management** > **Monitor Jobs**, or by checking the dialog box that opens when you submit the job.

6. The **Compute Resources** tab displays other parameters. Depending on the resources available for a scheduler environment, some of the fields may be disabled.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



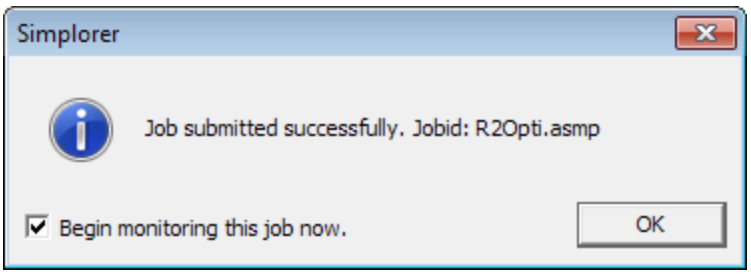
- Specify nodes.
- Total number of Tasks.
- Whether nodes are for exclusive usage by this job.
- Whether to limit number of tasks per node to a value.

  This can be useful in situations where the amount of memory available for a node is limited relative to the requirements for the project, and you want to ensure sufficient memory per process.

7. To submit the command with the specified parameters, click **Submit Job**.

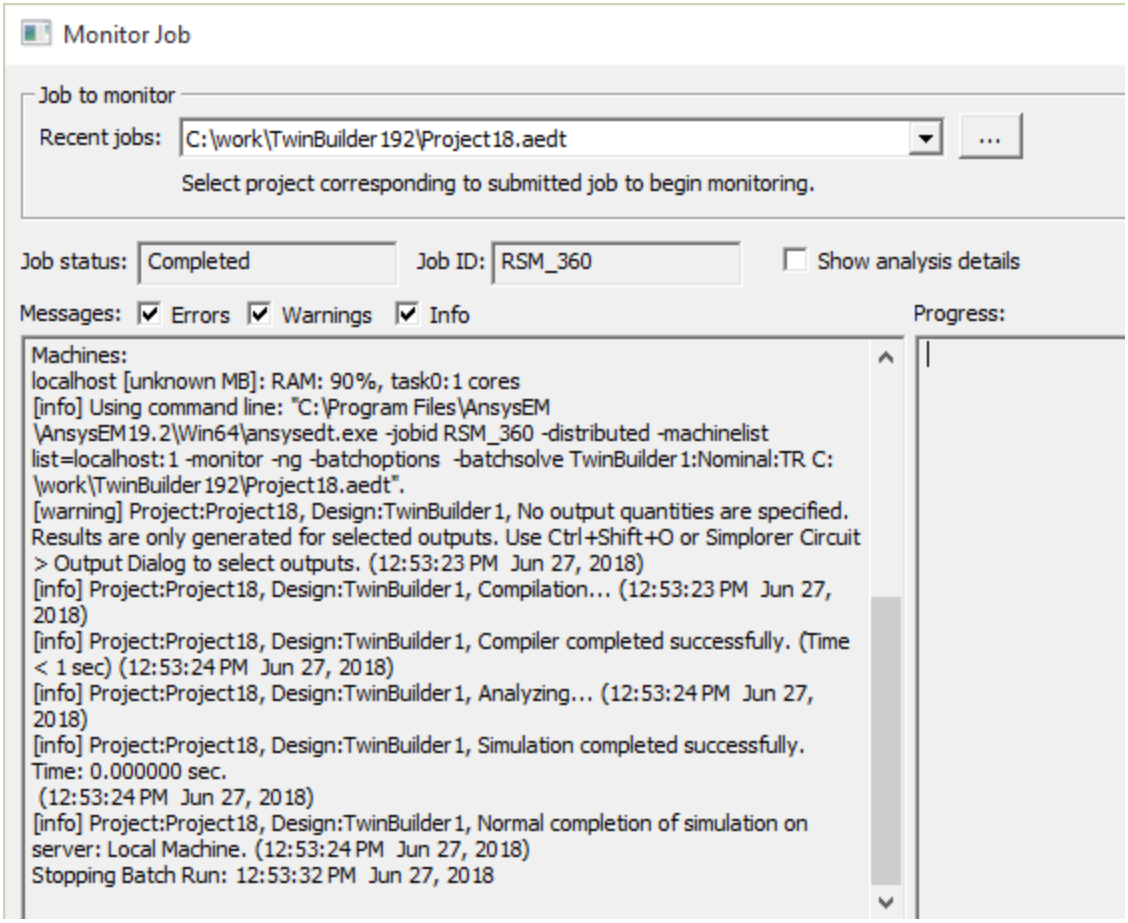
> **Note:**
>
> The RSM environment does not support queuing, so the job will immediately start running.

A dialog box displays in which you can **Begin monitoring this job now**.

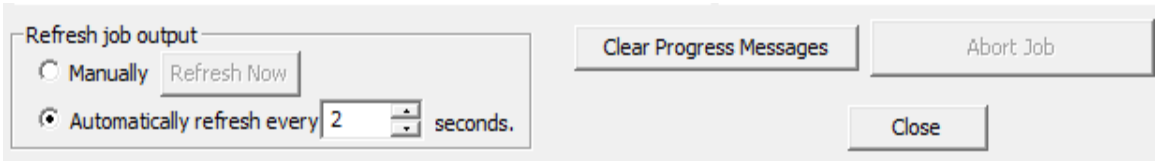


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

8. You can monitor this job by checking the option, or with the **Tools** > **Job Management** > **Monitor Jobs** command.



The dialog box contains fields reporting the job status, job ID, messages issues, and progress. You can filter the messages for errors, warnings, and info. You can **Clear Progress Messages**. You can refresh the job output manually or at specified intervals.



You can also perform a **Clean Stop** for a simulation between time steps, or **Abort Job**.

The process for submitting and monitoring large scale DSO jobs in the LSF environment is similar.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Set up and prepare model on your local workstation:
   a. Launch Desktop. Open project **/home/projects/spool/test.adsn**.
   b. Suppose the variations to solve come from **ParametricSetup1** setup. Right-click **ParametricSetup1** and run Large Scale DSO/Validate. Fix any validation errors.
2. Copy the input project (or folder, if the project references external files) from your personal workstation to a shared drive on cluster (say project is copied to **/home/projects/spool/test.adsn**).

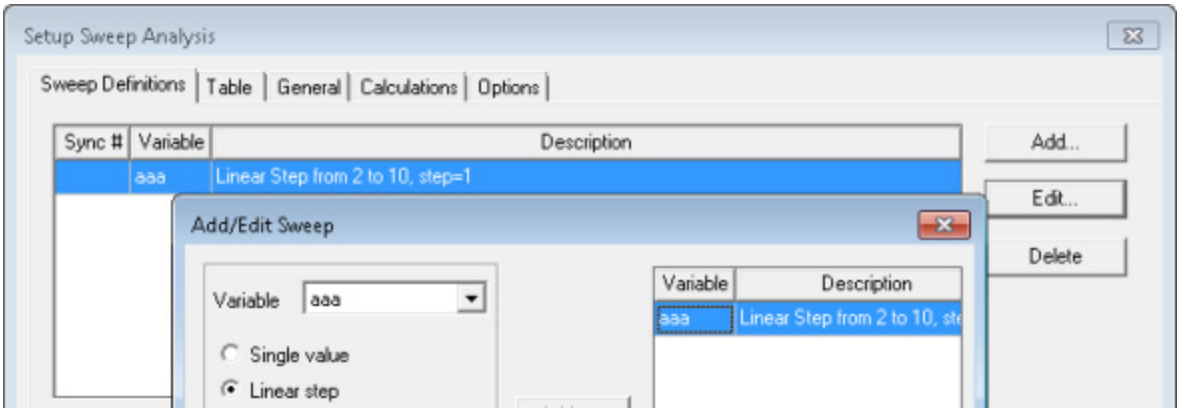
   In the scheduler environment, a cluster must have a designated postprocessing node. Open a remote desktop session (or equivalent such as vnc session) on the designated post-processing node. Say the name of this node is 'm1'. Launch Desktop graphically on 'm1'.

   Optionally, double-check that the model is prepared correctly.
   a. Open project **/home/projects/spool/test.adsn**.
   b. Right-click **ParametricSetup1** and run **Large Scale DSO/Validate**. Fix any validation errors.
   c. Save project, in case of edits. Close the project.
3. Run **Tools** > **Job Management** > **Submit Job**. The standard Job Submission panel pops up, which is documented as part of the scheduler-gui-integration feature.
4. Enter all fields. Make sure to select **ParametricSetup1** analysis and select the **Use large scale DSO** check box.
5. Click **Preview** to check (and/or copy to clipboard) the job's command-line. Click **Submit Job**.
6. Monitor this job through the **Tools** > **Job Management** > **Monitor Jobs** command. See Scheduler GUI Integration for details regarding the job submission and monitoring panels.

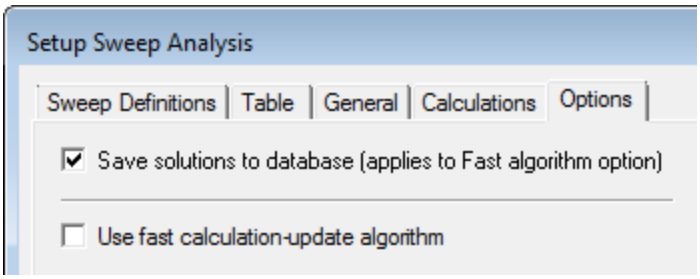
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Preparing a Project for Large Scale DSO Analysis

1. Ensure that the Twin Builder project is on the shared drive.
2. Set up a parametric table to use.

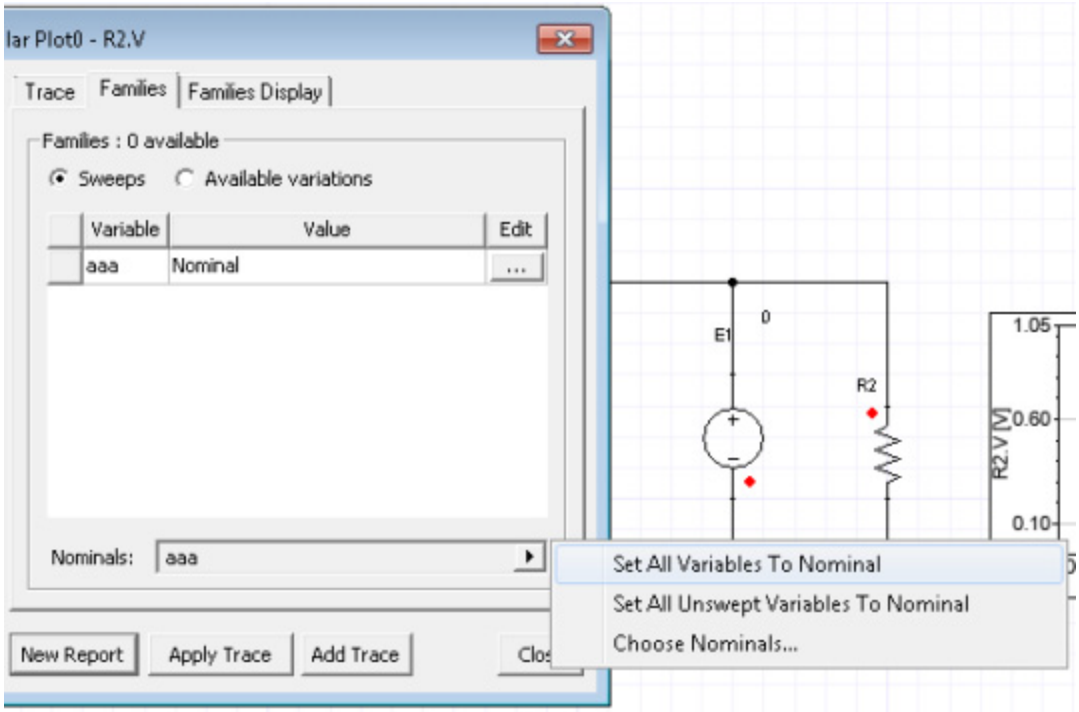


3. On the **Options** tab, select **Save solutions to database**, as shown below.



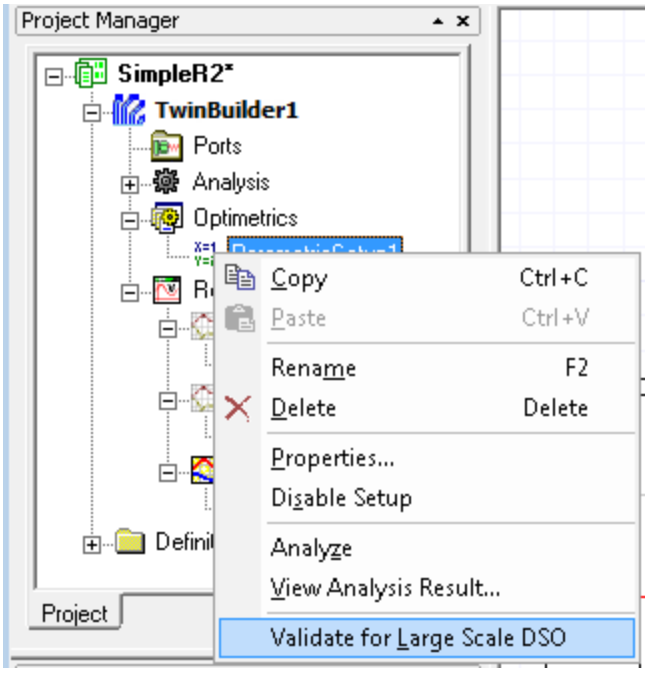
4. Outputs from large scale DSO come from predefined rectangular plots that are created before the Analysis command is issued. Non-rectangular plots of the design (digital plots, for example) are not extracted. Follow the steps below:
   a. Use the **Report** window to define outputs for the desired rectangular plots.
   b. Click the **Families** tab and ensure that all variables are set to **Nominal**, as shown below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



c. Click **New Report** to create the report.

4. Right-click the parametric sweep, and select **Validate for Large Scale DSO**.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. A dialog box reports any errors. The following are some typical messages:

   [error] Please remove all the calculations in Parametric Setup/Calculation page.

   [error] No rectangular plot exist in the design. Please create a rectangular report.

   [error] Report 'XY Plot 1' trace 'E1.I': please set all variables to nominal in Families tab.

   [error] Please turn on "Save solutions to database" in Parametric Setup/Options page.

6. Correct any errors, if necessary to pass validation.
7. Save and close the project.

Related Topics

Large Scale DSO for Parametric Analysis

Large Scale DSO Command Line Syntax

Large scale DSO operates through a non-graphical batch application called **desktopjob**. You can run the **desktopjob** command line to perform parametric analysis DSO. The command line interface supported by this batch program is consistent with the command line used for current DSO jobs. **desktopjob -help** lists all available command line options as shown below:

Command Line Syntax

desktopjob.exe <options> <project-path-on-shared-drive>

Options

-help: Print this help text.

-cmd: Specify command to run. Available choices: **dso**.

-ng: Run analysis in non-graphical mode.

-monitor: Output progress and messages to standard output/error.

-waitforlicense: Queue the job until the availability of licenses.

-preserve: Preserve local storage space of the distributed job for investigation into job's run. If local storage directory (such as the temp directory) is provisioned by the scheduler, ensure it is also configured to preserve the job's local storage. This storage should be deleted manually.

-batchoptions: Override the Tools/Option entries through either a **batchoptions** file or **batchoptions** string.

Example:

```
-batchoptions <config-file-on-shared-drive>
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
-batchoptions "'name1'='val1' 'n2'='v2'"
```

-machinelist:

- In the context of RSM:

  Specify machines for distributed analysis. Machine list is specified either inline (as a comma separated list of machine names) or through a file. Multiple cores are specified by repeating the machine name or by embedding number of cores in the machine name, using a colon separator.

  Example 1:

```
-machinelist "list=m1,m1,m1,m2,m2,m3"
```

  Example 2:

```
-machinelist "list=m1:3,m2:2,m3"
```

  Example 2:

```
-machinelist "file=machines.txt"
```

- In the context of **a scheduler such as LSF**:

  Specify the portion of total machines for distributed analysis. Use remaining for overhead or shared memory multiprocessing.

  Example:

```
-machinelist "Num=10"
```

-usefolderasinput: Choose this option if the job's input represents the entire folder rather than just the project file.

-maxfolderInMB: Specify the maximum size (MBytes) of input folder that is allowed for a valid job. By default, the maximum size allowed for input is 10MB. Specify a value of 0 to remove this size restriction and enable inputs of any size. This option applies when **-usefolderasinput** is used.

-workdir: Specify the shared drive folder for status and result files generated by analysis. By default, the results folder of input project is the work directory.

-batchsolve: Solve the specified parametric setup. Syntax for the setup:

<design-name>`:Optimetrics:`*<parametric-setup>*`.`

Related Topics

Large Scale DSO for Parametric Analysis

Running Twin Builder from a Command Line

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Large Scale DSO Job outputs

A large scale DSO analysis does not support the output of full parametric results. Instead, it extracts “subset” results using predefined rectangular plots, which you must create before the job is run. The extracted columns of data are saved as CSV files. Typically, there is one CSV file per trace, per variation. Non-rectangular plots of the design (digital plots, for example) are not extracted. The outputs can be either imported as datasets for post-processing in the desktop also as function of parametric variations, or used directly in downstream applications (for example, Excel, or custom programs that parse .**csv** files).

CSV File contents

The initial header rows of CSV files define the solved variation. For each such row, the first column has the variable name and the second column has the variable value. The row following variation rows has the name of primary sweep and the names of extracted quantities. Subsequent rows contain 'data' - quantity values as a function of primary sweep.

Related Topics

Large Scale DSO for Parametric Analysis

Import Large Scale DSO Dataset Solution

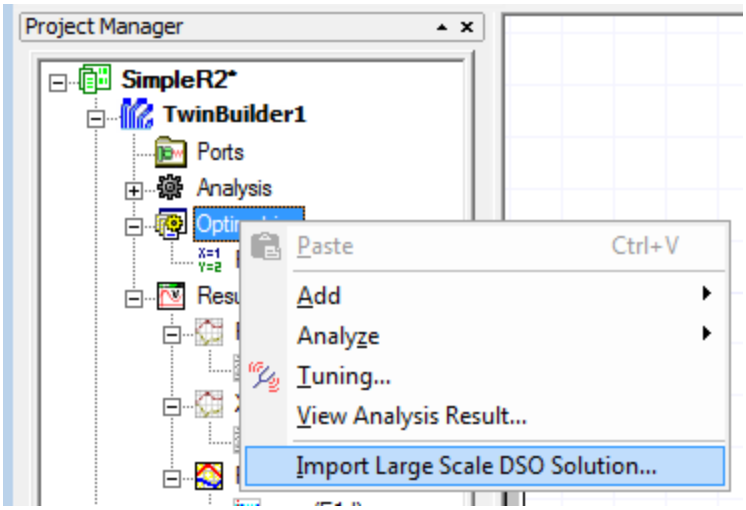
Import Large Scale DSO Dataset Solution

For post-processing large scale DSO dataset solutions in the desktop, create a dataset solution with **Import Large Scale DSO Solution** and point to large scale DSO job’s top level results folder.

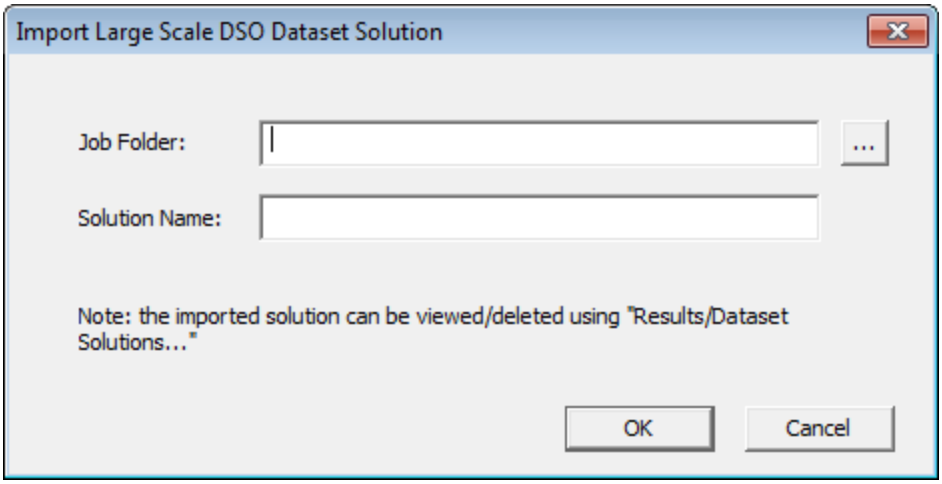
1. Import solved large scale DSO solution. You can do this in two ways.

   In the Project tree, right-click **Optimetrics** and select **Import Large Scale DSO Solution**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



This opens the **Import Large Scale DSO Dataset Solution** dialog box.



From here you can browse to the results of a large scale DSO job and select a job folder. See Large Scale DSO Results Database Organization.

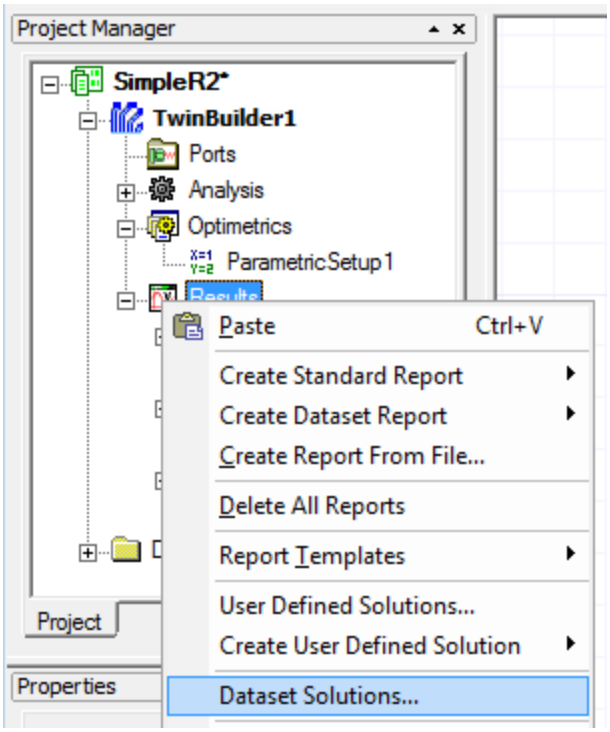
You can then specify the job folder. A dataset solution will be created and appear in the **Dataset Solutions** dialog box.

You can also import a large scale DSO dataset with the **Dataset Solutions** dialog box.

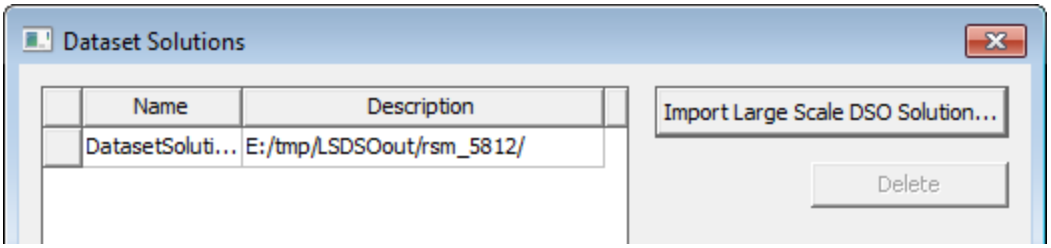
Viewing and Deleting Created Dataset Solutions

To view or delete created dataset solutions, in the Project tree, right-click **Results** and select **Dataset Solutions...**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



This **Dataset Solutions** dialog box appears.

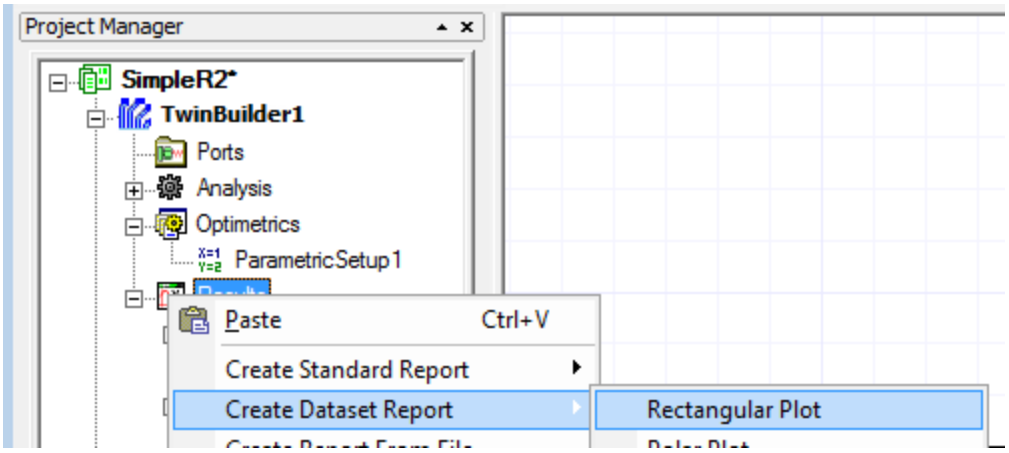


The dialog box lists any existing datasets. Select one or more from the list and click **Delete** to delete them. Click **Import Large Scale DSO Solution** to open the dialog box for importing a dataset solution.

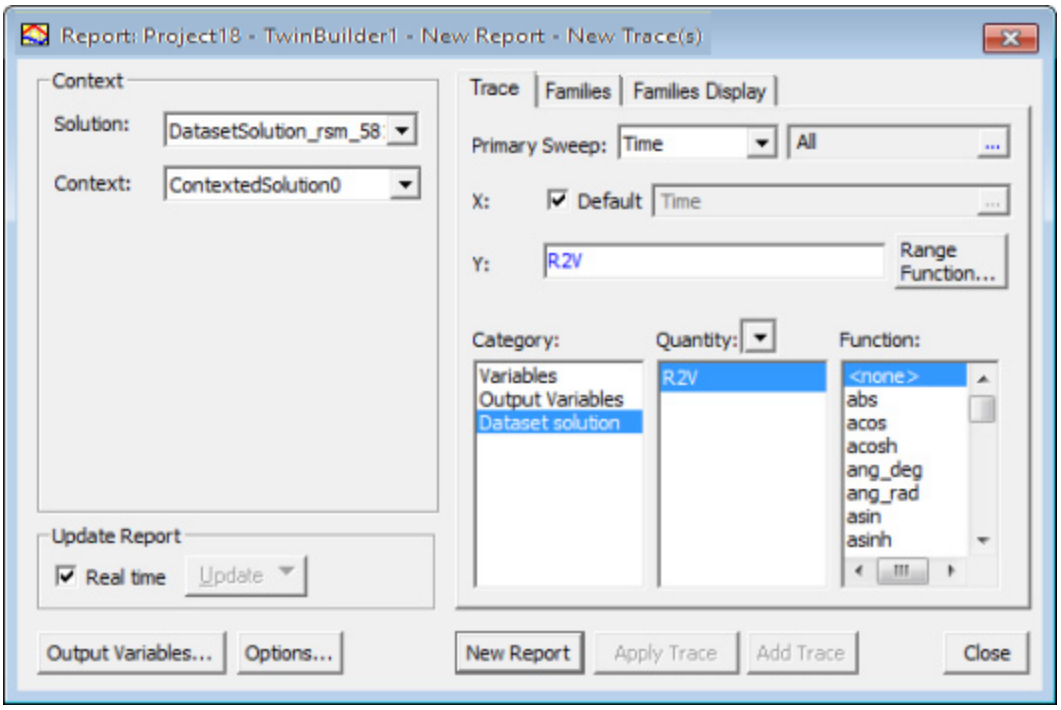
Creating a Dataset Report from Imported DSO Solutions

After you have imported one or more DSO solutions, you can create a **Dataset Report**. Right-click **Results** and select **Create Dataset Report**, then the type of report.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



A **Report** dialog box displays in which you can select quantities to plot, apply functions, and so on.



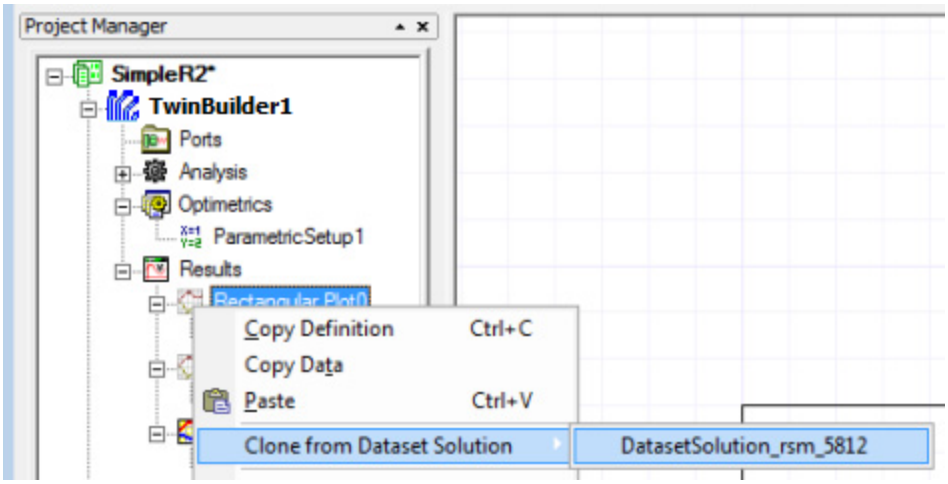
Related Topics

Large Scale DSO for Parametric Analysis

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Cloning a Report from a Dataset Solution

If you re-open a project that was solved using large scale DSO, you can quickly clone a report for a solved large scale DSO solution. Right-click the Project tree on the report and choose **Clone from Dataset Solution** > *Dataset_SolutionName,* provided this report is qualified for large scale DSO data extraction. This provides a way for you to reuse the existing report definition, thus saving the work of creating a new report.



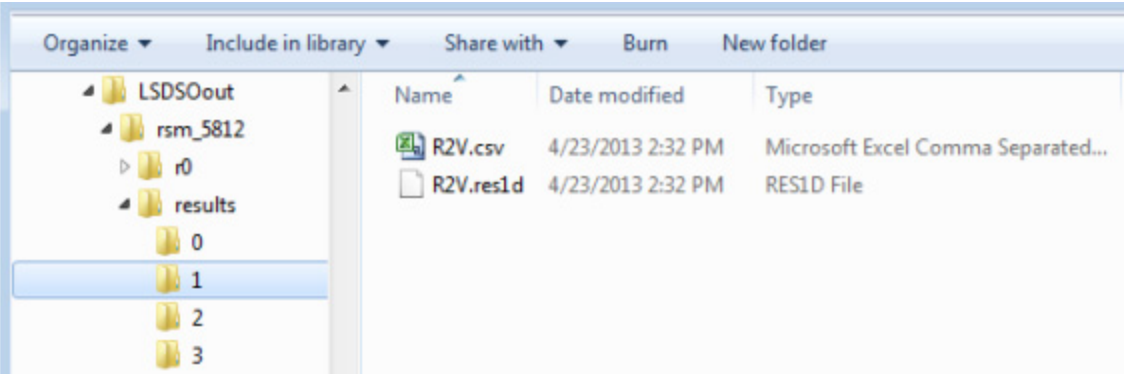
Large Scale DSO Results Database Organization

A large scale DSO analysis does not support the output of full parametric results. Instead, it extracts “subset” results using predefined rectangular plots, which you must create before running a job. The extracted columns of data are saved as CSV files. Typically, there is one CSV file per trace, per variation. Non-rectangular plots of the design are not extracted.

The results of a large scale DSO job are located in the <*workdir*>**/<jobid>/results** folder. If <*workdir*> is not specified on the job command line, it is same as the input project's results folder. For example, the default <*workdir*> corresponding to **\\shared\projects\SimpleR2.aedt** is **\\shared\projects\SimpleR2.aedtresults**. Within this results folder, there is one folder per variation. The name of the variation's folder is an integer number corresponding to the variation's index in the parametric table. For example, a variation folder named **4** has results for the fifth row of parametric table, while a variation folder named **0** has results for the first row of the table.

Below is a sample results folder showing the contents corresponding to results of the second variation. There is one CSV file corresponding to the project’s predefined trace.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Related Topics

Large Scale DSO for Parametric Analysis

Large Scale DSO Job Monitoring

Large scale DSO avoids detailed intra-variation monitoring as it increases network traffic for large scale jobs. Large scale DSO jobs are monitored as below:

- **Cluster monitoring tools** – The resource usage (CPU, memory, network) of large scale DSO jobs is monitored using standard cluster monitoring tools. Such job-neutral resource monitoring is ideal as it uses negligible network bandwidth, CPU/memory.
- **Detailed monitoring of analysis of a variation** – For any detailed monitoring you must examine the information provided in the job's log files. Specifically, the large-scale DSO job writes detailed logs conveying information regarding the machines where engines are running and the local storage location of per-engine distributed database. With such information, you can log in to individual machines for deeper probing of each distributed engine. The following logs are available:
  - **Per-node logs** – There is one **desktopjob.log** file per node assigned to the job. This log contains information regarding the node such as name, local storage folder, and number of engines started on this node. It is located in *<workdir>***/***<jobid>***/r***<nodeIndex>*. For example, *<workdir>***/***<jobid>***/r0** has **desktopjob.log** corresponding to the engines running on the first node of job, while *<workdir>***/***<jobid>***/r2** has logs corresponding to engines running on the third node.
  - **Per-engine logs** – There is one **desktopjob.log** file per distributed engine. It is located in *<workdir>***/***<jobid>***/r***<nodeIndex>***/r***<coreIndex>*. For example, *<workdir>***/***<jobid>***/r0/r0** has logs corresponding to the first engine running on the first node, while *<workdir>***/***<jobid>***/r1/r2** has logs corresponding to the third engine running on the second node. Engine-unique information (such as local storage of this engine) is logged here.
  - **Parametric analysis log** – This log file is located in '*<workdir>***/***<jobid>***/r***<nodeIndex>***/r***<coreIndex>* folder and corresponds to Desktop's

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

local-machine parametric **-batchsolve**. It is available only at the end of analysis and contains information regarding the variations solved by this engine and any info/warning/error messages.

- **Root desktopjob.log** – This is the top-level log that logs job distribution information such as hierarchical activation and the list of nodes assigned to this job.

Related Topics

Large Scale DSO for Parametric Analysis

Large Scale DSO Deployment/Configuration

Major Limitation

In the RSM environment, large scale DSO can only be enabled for one product.

Troubleshooting hints (RSM environment only)

The “shared drive read/write” requirement is a new constraint introduced in large scale DSO. If you experience a situation where regular DSO jobs run and large scale DSO jobs fail, one possible cause for the failure would be that the RSM service does not have privileges to read and write to the project folder located on the shared drive.

Windows Cluster Configuration

- **Shared drive for projects** – The cluster must provide a shared drive that hosts job inputs – the submitted project must be located on a shared drive. The shared drive must be accessible using the same path on every node of the cluster.
- **The temp directory configuration**

  Temp directory is either on "local storage" or on storage that has equivalent speed characteristics; that is, the I/O rates of the storage should be invariant to network traffic.

  The temp directory on a host has sufficient space to hold results database for the variations that are solved on it. This storage is freed at the end of the analysis.

  The amount of required space depends on the number of engines per node and the cumulative variations solved on this node.

- **RSM environment** – In the case of supported scheduler environments, there is no extra configuration needed. In the case of RSM environment, the following additional steps are needed:

  RSM must be running on all the nodes of the cluster. The credentials of "RSM service" allow read/write to the shared drive. This is because the remote engine processes launch using the credentials of RSM service.

  When registering **desktopjob.exe** with RSM service, the **desktopjob** program must be registered with Ansys Electromagnetics RSM using **desktopjob -regserver**. To ensure

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

that the registration is successful, check that the **desktopjob** entry in *<RSM-installation-folder>*/**AnsoftRSMService.cfg** file is valid.

- RSM and Ansys Electromagnetics products either are installed locally on each node of the cluster (local installation), or installed on a single shared drive available to all nodes of the cluster (network installation).
- Registration of **desktopjob.exe** with RSM service is either network or local.
  - **Network installation:desktopjob.exe** is registered with RSM service once, on any of the nodes of the cluster.
  - **Local installation** – Since each node has its own RSM installation, **desktopjob.exe** must be registered with RSM on each node.

> **Note:**
>
> IMPORTANT! The RSM service must be started using the credentials of a non-system admin account, which has read/write permissions to the project's shared drive. If the RSM service runs as a system user, large scale DSO jobs will fail.

Related Topics

Large Scale DSO for Parametric Analysis

Known Issues for Large Scale DSO

In a cluster configuration (shared drive requirement), the input files (project, and so on) must be present on a shared drive that is accessible from every node of the cluster.

Related Topic

Large Scale DSO for Parametric Analysis

Troubleshooting for Large Scale DSO

Deployment/installation errors (such as related to deployment configuration) are not captured. If there is such an issue, the large scale DSO job will fail without useful messages in the logs.

Report-based extraction fails mysteriously if traces and parametric-setup are not 'prepared' as per the Getting Started guides.

Job monitoring and control

There is no provision for stopping and restarting a job. A new job does not reuse solved results; it always solves all rows in the table. Therefore, an abort or failure of a job restarts from the beginning unless a new parametric table with the unsolved rows is created.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Job outcome

Job status: The exit code of a job doesn't indicate success or failure correctly. The error messages from multiple log files need to be combined to determine the reason for failure. In many situations, the reason for a failure is apparent only after re-running the job after turning debug logging on.

In some scenarios, the analysis appears to finish successfully with valid results, except that the exit code is **134**. In this case, although the exit is abnormal, the failed exit code can be ignored.

Load Balancing: For models with "unbalanced variations table" (that is, variations that take a considerably different amount of time to solve are clustered in a few regions of the table), the job will take a longer time to solve than a regular DSO as the job's overall completion time is determined by the slowest solving region.

One workaround is to rearrange the rows in the parametric table so that each region takes a similar time to solve.

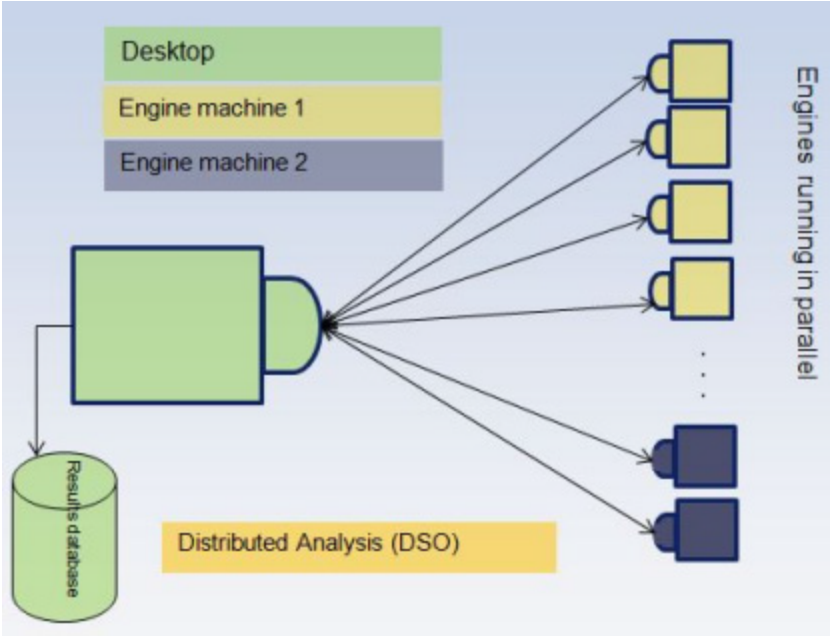
Related Topic

Large Scale DSO for Parametric Analysis

Large Scale DSO Theory

The parametric analysis command in Ansys Electronics Desktop computes simulation results as a function of model parameters, such as geometry dimensions, material properties and excitations. The parametric analysis is either performed on a local machine, where each variation is analyzed serially by a single engine, or distributed across machines through a DSO license. Desktop's DSO analysis runs multiple engines in parallel, thus generating results in a shorter time. In the regular DSO algorithm, the parametric analysis job (Desktop) runs on the primary node, which in turn launches one or more distributed-parallel engines on each machine allocated to the job. Desktop distributes parametric variations among these engines running in parallel. As variations are solved, the progress/messages and variation results are sent back to Desktop, where they are added into the common results database on primary node, as illustrated below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Regular DSO Bottleneck

As illustrated above, regular DSO's speedup is limited by the resources of the centralized “Desktop” bottleneck. It's been observed that DSO becomes unreliable at a certain point, as the number of engines and number of variations increases. The term “large scale parallel” can be used to define this tipping point. For a given model, a “large-scale parallel” job denotes scenarios, in terms of the number of distributed-parallel engines and the number of parametric variations, where the regular DSO runs into centralized bottlenecks that result in progressively smaller speedups and/or unreliability.

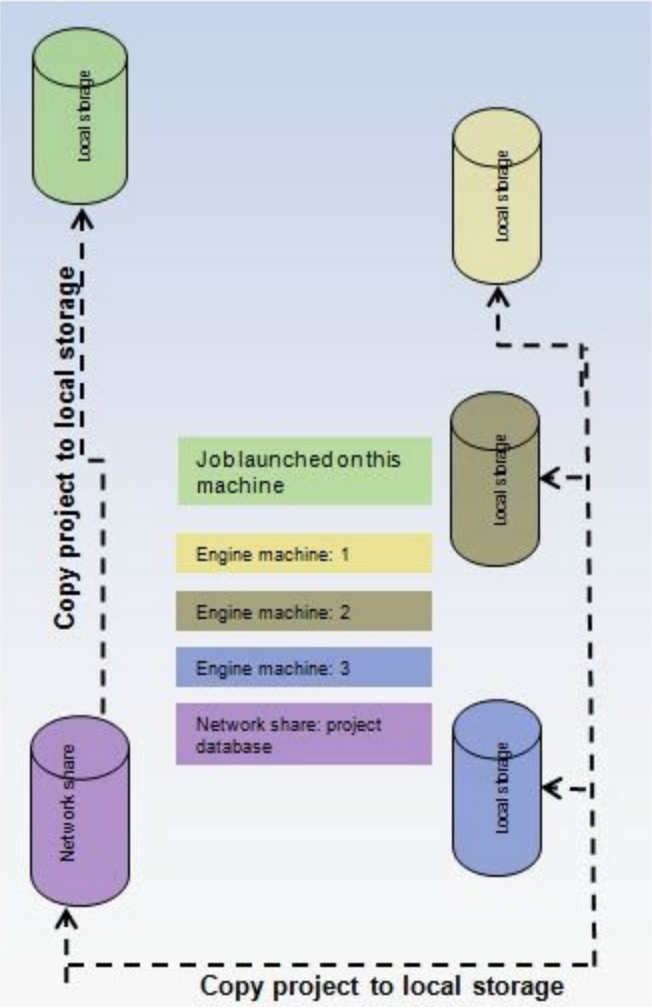
With the advent of economical availability and timely provisioning of compute resources, product designers have access to large compute clusters to run their simulations. They are using more compute resources at simulation jobs to obtain results faster. The parametric DSO needs to meet this challenge and target linear speedup for “large-scale parallel” jobs. The large scale DSO feature is targeted toward 100% reliability and linear speedup of large-scale parallel DSO jobs.

Key Algorithms/Concepts for Large Scale DSO

- **Embarrassingly Parallel algorithm** – Large scale DSO exploits the embarrassingly parallel nature of parametric runs. The solve of each variation is made fully independent of another variation by replacing regular DSO's centralized database with per-engine distributed databases.
- **Distributed databases** – For each distributed-parallel engine, the database of the input model is cloned to local storage of the engine's compute node (as illustrated in the following picture). Parallel analysis is performed on these cloned models and the analysis results also go to the corresponding location in local storage. As each engine has its own

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

database, there is no “shared database” contention between any two engines. There is negligible network traffic as analysis/computations are contained in a single machine and use just the “local resources.”



- **Hierarchical activation of engines** – In the large scale DSO algorithm, engine activation is done hierarchically, where the overhead of launching of engines is also shared across the nodes. Such hierarchical activation improves reliability of the activation phase of large-scale parallel jobs. A new **desktopjob** program encapsulates hierarchical activation, as illustrated in the pictures below. In this approach, the “root” **desktopjob** (Root DJ) activates a “level-one” child **desktopjob** (DJ(L1)) for each unique node allocated to the DSO job. Each level-one **desktopjob** activates one or more “leaf” **desktopjobs** (DJ(L2)) equal to the number of distributed engines per node. A leaf **desktopjob** in turn runs Desktop in batch mode, to perform local-machine parametric analysis to solve the variations assigned to this engine.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Decentralized Load balancing** – A parametric table is divided into regions of equal number of variations, with the number of regions equal to the number of engines. In the illustration below, the analysis of 30 parametric variations is distributed among five engines. Each engine solves its assigned region as a “local machine” parametric analysis. The large scale DSO job is finished when all engines are done with their assigned variations.

Distributed results postprocessing – As an engine is done with the solve of a variation, it extracts results for the solved variation before progressing to the analysis of next variation. The extracted results are saved to the local storage. When the engine is done with analysis of all variations, the extracted results are transferred from local storage to the results folder of the input project.

High Performance Computing (HPC) Integration

Ansys Electromagnetics products offer direct integration with a number of High Performance Computing (HPC) software programs (job schedulers). This direct integration does not require the RSM service. The list of currently-supported HPC software includes:

- Microsoft Windows® HPC Server 2008

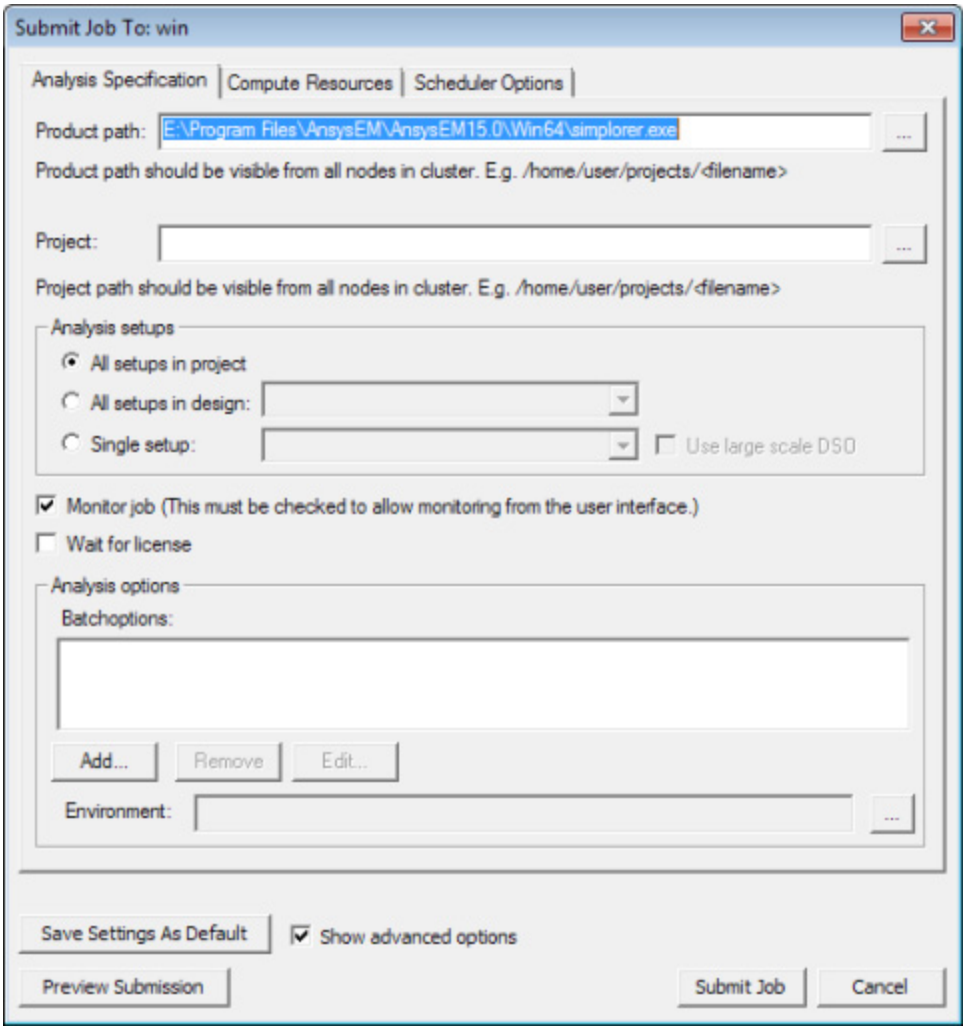
You can also do custom integration.

> **Note:**
>
> For additional information about high performance computing not in this guide, see the *Ansys Electromagnetics HPC Administrator's Guide* (**HPC_Admin.pdf**) in *<installdir>*/**AnsysEM**/*<release_number>*/**Win64**/**Help**.

A job scheduler may also be described as a batch system, a Distributed Resource Management System (DRMS) or Distributed Resource Manager (DRM). The features supported on each scheduler are included in the documentation for each. For each job scheduler, the versions or revisions that have been tested are included.

You can submit jobs using the command line tools or other tools provided by the scheduler. Ansys Electromagnetics products include a **Submit Job To** dialog box such as the example shown below to help you submit jobs to a job scheduler.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The general procedure is to specify the scheduler and head node, describe and submit the job, and monitor the results.

The **Submit Job To** dialog box contains three tabs:

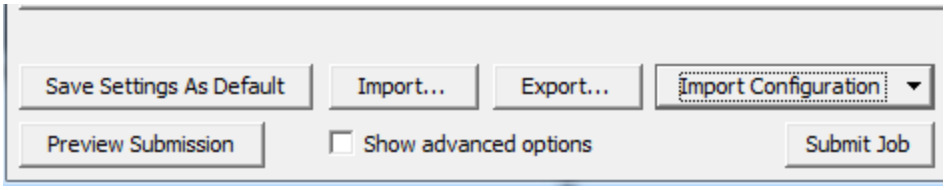
- **Analysis Specification** – Specify the project name, the setups, and analysis options such as batchoptions. If **Show advanced options** is selected, environment variables can also be added. The project file path name must be a UNC path that is accessible from each compute host used for Ansys Electromagnetics jobs.
- **Compute Resources** – Specify either a predefined analysis configuration (see ***Configuring Distributed Analysis***) or specify parameters in the Resource selection fields. Job parallelization settings are enabled if supported by the chosen scheduler.
- **Scheduler Options** (not present for RSM jobs) – Specify the job name and priority. If **Show advanced options** is selected, additional job submission options can be set.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

When you use the GUI to submit jobs, the Desktop (UI) process must run on a host which is also a submission host for the job scheduler. This mode is called local mode or working mode.

Import and Export Configurations

The bottom of the job submission interface has buttons for **Import**, **Export**, and **Import Configuration** let you save a configuration for each solver type.



- Click **Export** to export most of the settings of this dialog box (all tabs) to a file.
- Click **Import** to update most of the settings in this dialog box (all tabs) from a file.
- Click **Import Configuration** to update the DSO settings in this dialog box from any DSO configuration as shown in the **Configurations** tab of the **HPC and Analysis Options** dialog box. The Design Type of the DSO configuration must match the design type of one of the designs in the project, so the project must be specified before using the **Import Configuration** button. The batchoptions are also set from the specified DSO configuration or from the Design Type options settings, which are shown in the **Options** tab of the **HPC and Analysis Options** dialog box.

Click **Export** and **Import** to save and restore a frequently used collection of job submission settings. You can also click **Save Settings as Default** to save the current settings, but it always overwrites any previously saved settings. When you click **Export**, you can save multiple sets of settings, or may transfer the settings to another machine.

The **Select Scheduler** dialog box also has **Export** and **Import** buttons. Use these buttons to save the settings in this dialog box to a file or restore them from a file.

An exported configuration is named **Submit_Job_Settings** by default and has an **.areg** extension. A file browser window opens in the project folder and lets you name an exported file and location, and select **.areg** files to import. The **SubmitJob** scripting command uses job submission settings that have been exported from the **Submit Job** dialog box to an **.areg** file. The path to this **.areg** file is thus the first argument to the **SubmitJob** scripting command. For further information, see ***Job Submission Scripting***.

Related Topics

Scheduler Terminology

What a Scheduler Does

Installation of Ansys Electromagnetics Tools

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ansys Electromagnetics Jobs

Submitting and Monitoring Ansys Electromagnetics HPC Jobs

Scheduler Terminology

- **Core** – Unit of processing.
- **Processor** – Consists of one or more cores.
- **Machine/Host/Node** – Consists of one or more processors, memory, disk, and so on.
- **Resource** – Machines, licenses, and so on that are used by a job.
- **Job** – Application (or program, executable) with command line options that uses resources to produce useful results. For example, **ansysedt.exe –ng –BatchSolve**.
- **Serial Job** – Job that runs on a single core.
- **Parallel Job** – Job that runs on multiple cores (belonging to same or different machines).
- **Compute Cluster** – Network of machines on which jobs run. Typically consists of head nodes and many compute nodes.
- **Service** – Program that runs in the background (like an RSM service). Listens on a port. OS provides programming interface by which applications communicate with service, once machine and port number are known. Launching an executable on a remote machine requires a service to run on the remote machine.

Ansys Electromagnetics Terminology

- **Desktop** – The main application used to accomplish a task, such as Twin Builder. The desktop may run as a GUI or as a batch command.
- **Engine** – Application (or executable) that launches during analysis commands, to generate analysis results.
- **Distributed-processing** – Multiple engines launched simultaneously on one or several machines. Uses **ansoft_distrib** (and related) license.

Related Topics

What a Scheduler Does

Command Line Information for Ansys Electromagnetics Desktop Products

High Performance Computing (HPC) Integration

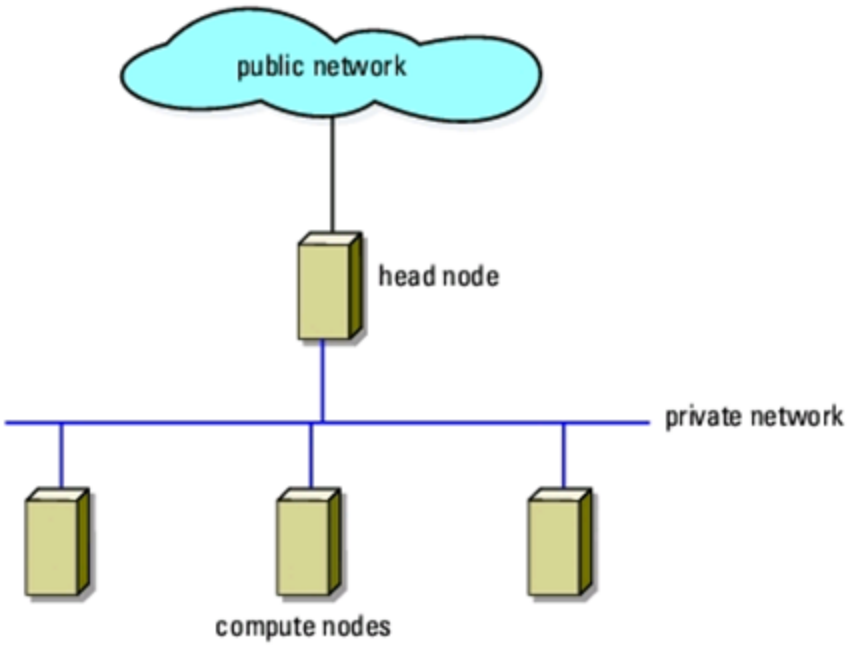
What a Scheduler Does

- Enables effective/efficient utilization of a cluster's resources consistent with your organization's goals.
- Maintains job queues.
- Maximizes throughput of the jobs by processing all jobs as fast as possible.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Typically, one job per CPU policy.
- Allows choice of various scheduling policies (for example, first come first serve, priority based, preemption).
- Provides a suite of tools or utilities (graphical or command line) for you to submit jobs, monitor jobs, abort jobs, and suspend jobs.
- Manages a compute cluster by running various interacting services on head nodes and compute nodes.
- Provides a programming interface to access services.

Scheduler Managed Compute Cluster



Head nodes maintain queues. Compute nodes are typically on a high speed network, improving the scalability of parallel jobs. Services running on nodes interact with each other to manage resources. End user tools communicate with services to submit, abort, and suspend jobs.

Related Topics

High Performance Computing (HPC) Integration

Command Line Information for Ansys Electromagnetics Desktop Products

Installation of Ansys Electromagnetics Suite

Ansys Electromagnetics Suite must be available on each cluster host where Ansys Electromagnetics jobs may be run.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- On a Windows platform, Ansys Electromagnetics Suite must be installed separately on each host of the cluster.

The Ansys Electromagnetics Suite must be accessible using the same path on each host. All cluster users running Ansys Electromagnetics jobs must have permission to read and execute the files in the Ansys Electromagnetics installation directory and its subdirectories.

The Temp directory selected during installation must be readable and writable by all user accounts used to run the Ansys Electromagnetics Suite. This temp directory path should be the same on all machines of the cluster and should be local to every machine. For example, **c:\temp** on Windows.

Because HPC is offered as a direct integration, you need only install the Ansys Electromagnetics Suite software. No additional configuration is required.

Example

Install the Ansys Electromagnetics Suite in the **C:\Program Files\AnsysEM\v**<*version number*>**\Win64** directory on each node of the cluster. The same directory path name must be used on all hosts.

Related Topics

High Performance Computing (HPC) Integration

Firewall Configuration

Installation Directory Examples

Firewall Configuration

If the firewall is turned off between the machines of the cluster, there is no need for any configuration. If the firewall is turned on, you, or a system administrator, should perform the steps below.

- **Windows cluster** – Configure firewall by adding exceptions that allow Ansys Electromagnetics programs and services to communicate with each other. If you are using standard Windows Firewall, this is done by the Ansys Electromagnetics installation program. On the other hand, if you are using a 3rd-party firewall software, it needs to be configured in a similar manner.

Related Topics

High Performance Computing (HPC) Integration

Installation Directory Examples

Installation Directory Examples

Microsoft Windows Example

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Install the Ansys Electromagnetics tools in directory **C:\Program Files\AnsysEM\v**<*version number*>**\Win64** on each node of the cluster. Use the same directory path name on all hosts.

Related Topics

High Performance Computing (HPC) Integration

Firewall Configuration

Ansys Electromagnetics Jobs

Ansys Electromagnetics Jobs

For most cluster environments, an Ansys Electromagnetics job consists of an Ansys Electromagnetics Desktop application running in non-graphical mode, performing a batch solve. Submit the job to the scheduler, specifying an Ansys Electromagnetics Desktop command line to be executed on the cluster. For some schedulers, you may, or must, specify a script to run instead of specifying the Ansys Electromagnetics Desktop application command line. In these cases, the script will contain the corresponding Ansys Electromagnetics Desktop application command line. When the resources requested for the job are available, the scheduler will start the job. In many cases, you will not know which hosts are allocated to the job. With direct integration, if the Ansys Electromagnetics job is a distributed job, the Ansys Electromagnetics Desktop application will query the scheduler for the hosts allocated to the job, and it will use the scheduler facilities to launch the distributed engines.

Related Topics

High Performance Computing (HPC) Integration

Running Twin Builder from a Command Line

Job Submission Scripting

To help with automation, you can submit batch jobs through script commands of the **oDesktop** object. The **SubmitJob** script command uses job submission settings that have been exported from the **Submit Job** dialog box to an **.areg** file. The path to this **.areg** file is the first argument to the **SubmitJob** command. Additional arguments include the path to the project file, the design name (if restricting the solve to a particular design), and the setup name (if further restricting the solve to a single setup within a design).

For further automation, use the **SelectScheduler** scripting command to determine what scheduler to use for submission, to include options for head node, user name, and whether to require password entry. If the user name differs from the cached user name, or the force password flag is set, then the **Select Scheduler** dialog box appears. If there are any issues with the scheduler selection (for example, a password is required or the requested host wasn't found), then the **Select Scheduler** dialog box appears. This is the only part of job submission scripting that may require user intervention. This same mechanism is used if, from within the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SubmitJob command, there is failure to connect to the scheduler. Even though there are allowances for graphical user intervention if something goes wrong, if the password (if required) is cached and all settings are correct, the entire submission process can run non-graphically and fully automated.

Limitations

All settings besides the arguments passed to the **SubmitJob** command must be stored in the **.areg** file containing settings exported from the **Submit Job** dialog box. These include (but aren't limited to) batch options, environment variables, batch extract settings, and compute resource selections. To run many job submission scripts with variation of these settings, there must be multiple **.areg** files available.

You can select the same project multiple times with a single script. Take care in this situation because each time a project is submitted, the state-keeping files used for monitoring are removed so that the job can create them from scratch to ensure consistency. While this ensures proper monitoring for a job that is just being submitted, it could interfere with monitoring (or even correct solving if a lock file is deleted) of a job that is already in progress. Because of this, if the same project is to be re-submitted from within a single script, the job should be monitored (waiting for completion) before trying to submit it again. Perform this monitoring/waiting with a combination of a single **LaunchJobMonitor** command followed by a loop that checks the result of a **RefreshJobMonitor** command.

How to do Job Submission Scripting

The typical scenario for job submission scripting is:

1. Manually select the scheduler. Use the **Select Scheduler** dialog box to open the **Submit Job** dialog box.
2. Choose a representative project (with the desired design type), and select the appropriate analysis settings.
3. Make the required compute resource selections and try to preview the job.
4. If preview is successful, export the dialog box settings and record the path to the new **.areg** file.
5. Create a script containing at a minimum a **SubmitJob** command with the path to the **.areg** file, and the path to the project file. Note that there must be double backslashes for each backslash of a path, since the backslash is an escape character. When the script successfully runs there should be a message in the message windows stating that the job was submitted, including the job IDs. There could be multiple job IDs if multi-step submission is used.

See the Twin Builder Scripting help (select **Help** > **Twin Builder Scripting Help**) for details on the **SelectScheduler**, **SubmitJob**, **LaunchJobMonitor** and **RefreshJobMonitor** commands.

Related Topics

High Performance Computing (HPC) Integration

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Running Twin Builder from a Command Line

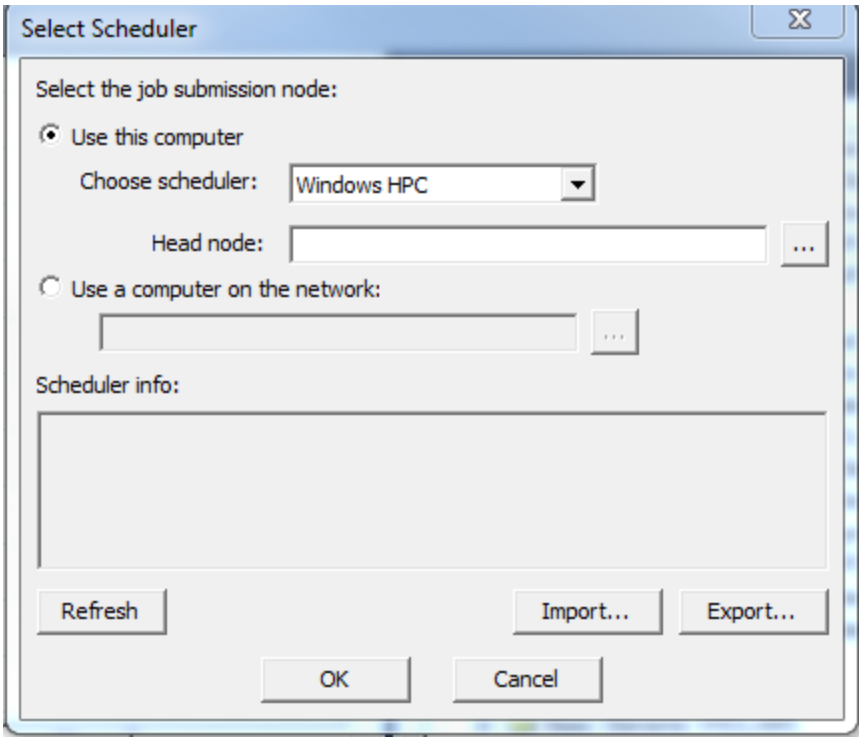
Integration with Microsoft Windows® HPC Scheduler

The Windows HPC scheduler is only supported on Windows. Jobs may be submitted by:

- Using the Windows HPC GUIs from Microsoft: Job Manager or Cluster Manager.
- Using the Windows HPC command line tools (job, and so on).
- Using Windows HPC specific settings from the Ansys Electromagnetics Desktop. The Desktop provides commands for Scheduler selection, Job submission and Job monitoring/control.

Before you can use **Submit Job**, select **Tools** > **Job Management** > **Select Scheduler** to select a scheduler.

This opens the **Select Scheduler** dialog box.



Specify the following parameter:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Use this computer** – Choose this option if scheduler commands are enabled on the post-processing node.

  Select the scheduler from the **Choose scheduler** drop-down list. If Windows HPC Integration with the generic job submission GUI is not enabled, Windows HPC will not appear in the drop-down list.

  Head node – This is the node on the cluster where scheduler commands are allowed to run.

- **Use a computer on network** – Choose this option of if the cluster is configured in a manner that disallows job submission from the post-processing node. Specify the desired node name. Click to open a file browser dialog box.

After specifying the job submission node, click **Refresh**. This verifies that the head node can be contacted, and displays the scheduler name, a brief description (including the head node name), and the version of the Windows HPC head node.

Press **Cancel** to discard changes made in this dialog box. Press **OK** to verify that the head node can be contacted before accepting the changes. If no problem occurs, the dialog box closes. If there is a problem contacting the head node, the dialog box stays open and the changes are not accepted.

Once you select a scheduler, you can use the **Tools** > **Job Management** > **Submit Job** and **Tools** > **Job Management** > **Monitor Jobs** commands to open dialog boxes for job submission, monitoring and control.

The Ansys Electromagnetics Suite 2024 R2 release has been tested with the following versions of Windows HPC:

- Windows HPC Server 2008 R2
- Windows HPC Server 2012

General Guidelines for Submitting Ansys Electromagnetics Jobs

A job submitted to a Windows HPC Cluster is defined by its properties, as well as the task list and task properties. Priority, resource requirements, node preferences, and so on, come from job properties. In the case of Ansys Electromagnetics jobs, the task list consists of a single task. Properties of this task specify command line that runs Ansys Electromagnetics Desktop in non-graphical mode to perform analysis of an Ansys Electromagnetics project.

Specifying the Number of Compute Resource Units for HPC Jobs

Select **Use automatic settings** on the **Compute Resources** tab, or enter the number of tasks and total cores per machine, or individual nodes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ansys Electromagnetics Project File and Project Directory for use with Windows HPC Scheduler

Ansys Electromagnetics tools write their results to a subdirectory of the directory containing the Ansys Electromagnetics project file. The project directory must be accessible to all of the cluster hosts that may run Ansys Electromagnetics jobs. The user account for the job must have permission to read the project directory, and to create and modify files and subdirectories of this directory. The path name of the project file must be accessible to all cluster hosts using the same path name, which is generally expressed as a UNC path name.

Example:

The project file is on your workstation (with hostname **user1_PC**) in directory **C:\user1\projects\new\project1.aedt**, and the directory **C:\user1\projects** is shared with sharename projects.

Correct

When submitting the job, use the following path name to specify the project file:

```
\\user1_PC\projects\new\project1.aedt
```

Incorrect

If a local path name is used, the cluster hosts will not be able to find your project on the workstation:

```
user1_PC: ' C:\user1\projects\new\project1.aedt '
```

Related Topics

Integration With Microsoft Windows® HPC Scheduler

Windows® HPC Job Templates

Selecting Computation Resource Units (Job Unit Type)

Windows® HPC Job Credentials

High Performance Computing (HPC) Integration

Submitting and Monitoring HPC Jobs

Submit jobs to the Windows HPC Scheduler by:

- Using the **Submit HPC Job** dialog box (see ***Submitting and Monitoring Jobs for Windows HPC***).
- Using the Windows HPC Job Manager GUI.
- Using the Windows HPC Command Line Tools.
- Using the Windows PowerShell.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

See Microsoft's documentation for information on the latter three methods.

Install Client Utilities from the Microsoft HPC Pack on the submit host to use any of these methods to submit a job to a cluster. The **Submit HPC Job** dialog box cannot contact the cluster head node if the client utilities are not installed.

Submit jobs from any Microsoft Windows host meeting the following requirements:

- For submitting jobs to the Windows HPC scheduler, the Desktop process must run on a node that is configured for submission of jobs to the Windows HPC cluster. That is, the Windows HPC Client Utilities must be installed on the node, and network communication from the Desktop node to the head node of the cluster must be allowed. For R15, Windows HPC Server 2008 R2 (or later) client utilities are required. Using a computer on the network is not supported for submission of jobs to the Windows HPC cluster.
- When submitting jobs to a Windows HPC cluster, you must also specify the head node of the cluster to which the jobs will be submitted. When you select the "Windows HPC" scheduler in the "Choose scheduler" list, the Head Node edit control is enabled. You may enter the Windows HPC cluster head node name into the edit box. Alternatively, press to browse for and select the head node.
- The Windows HPC Pack client utilities are installed on the submission host.
- Network communication between the submission host and the Windows HPC Cluster head node is permitted; there is a network connection between these hosts that is not blocked by a firewall.
- The submission user can submit jobs to the Windows HPC Cluster.

Job Monitoring

Monitor Windows HPC Jobs using the **Monitor Job** dialog box; select **Tools** > **Job Management** > **Monitor Jobs**. This dialog box can also be opened by selecting the **Begin monitoring this job now** check box when a job is successfully submitted using the job submission dialog box.

In addition to the above requirements to allow job monitoring the following is also necessary:

- Network communication between the submission host and all Windows HPC Cluster nodes where the job may run must be permitted.
- There must be a network connection between these hosts that is not blocked by a firewall.

Cluster Configuration

Any job running on a Windows HPC Cluster that is distributed over multiple compute hosts requires network communication between processes running on these hosts. The cluster must be configured to allow this communication. Any firewall or other security software must be

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

disabled or configured to allow communication between any of the compute hosts where a job could run.

Job Submission User Profile on Cluster Compute Nodes

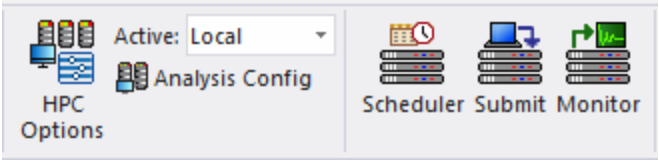
For a job to run correctly, the submission user's profile must be accessible and properly initialized on the cluster compute nodes where the job runs. If the Ansoft/temp subdirectory of your **My Documents** directory does not exist or is not accessible on the compute cluster nodes where a job runs, the batchoptions for the job will not be processed correctly, resulting in job failure. One way to ensure that this directory is created on each compute host is for the submission user to log in to each compute host and run the Ansys Electromagnetics product application one time.

Related Topic

Specifying the Number of Compute Resource Units for HPC Jobs

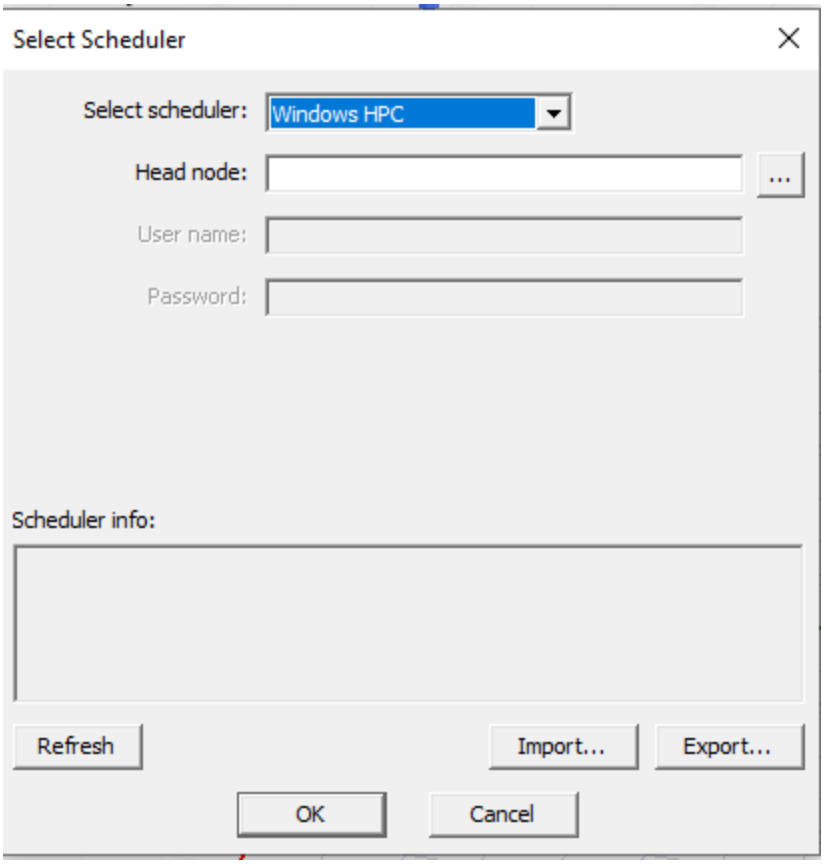
Submitting and Monitoring Jobs for Windows HPC

In order to submit jobs using **Windows HPC**, select **Tools** > **Job Management** > **Select Scheduler** or **Simulation** > **Scheduler** on the ribbon.



The **Select Scheduler** dialog box appears. Specify **Windows HPC** as the scheduler.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



For Windows HPC, specify the head node of the cluster.

Specify the cluster head node and click **Refresh**. This verifies that the head node may be contacted, and displays the scheduler name, a brief description (including the head node name), and the version of the Windows HPC head node.

Click **Cancel** to discard your changes. Click **OK** to verify that the head node can be contacted before accepting the changes. If no problem occurs, the dialog box closes. If there is a problem contacting the head node, the dialog will not be closed and the changes are not accepted.

After setting the job submission node, select **Tools** > **Job Management** > **Submit Job**, **Project** > **Submit Job**, or *ProductName* > **Submit Job** to open the **Submit Job To:** dialog box. You can also select the **Simulation** tab of the ribbon and click the **Submit** icon. You can also access **Submit Job** from the shortcut menus for the Project Name, Design name, Analysis Setup, or Optimetrics Setup.

The **Submit Job To** dialog box contains three tabs:

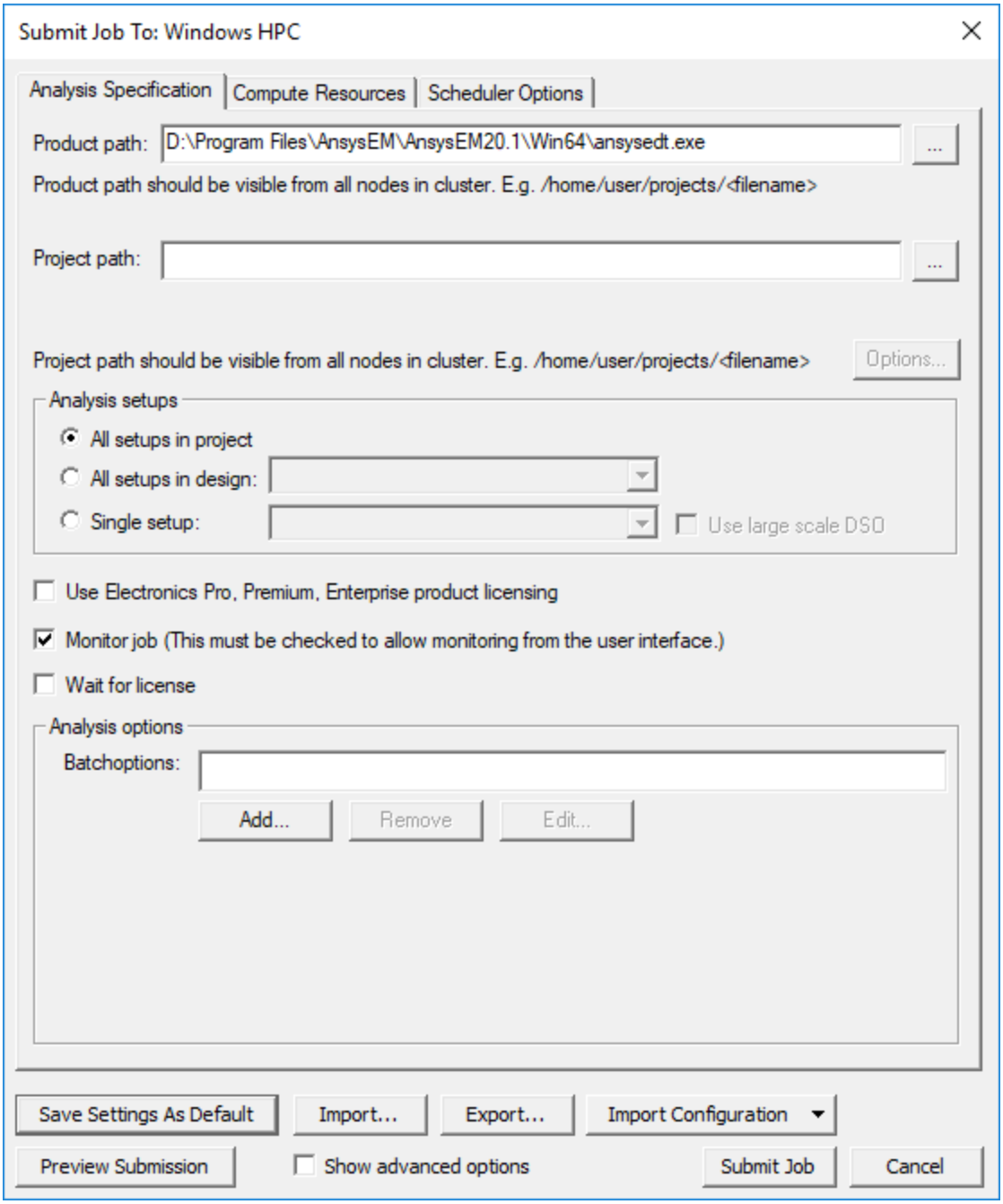
- **Analysis Specification** – Specify the Product path, Project name, the setups, and analysis options such as batchoptions, or, for advanced users, Environment variables. If you select the Analysis or Optimetrics setup, the Analysis Specification is pre-populated.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Compute Resources** – This tab can be populated either by automatic settings, by predefined Analysis Configuration, or specifying parameters in the fields for resource selection, for job parallelization and enabled forms of parallelization.
- **Scheduler Options** – Contains fields for Job name and priority. The customization options shown by selecting advanced are not used for Windows HPC.

In the **Analysis Specification** tab, enter the path names of the product path and of the project file in the “Project” edit box. These must be UNC paths that are accessible from each compute host used for Ansys Electromagnetics jobs. The Project can be an archive. The submission user must have permission to write to the directory containing the project file.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



You can select which setups are analyzed in the Analyze Setups section of this dialog box. There are radio buttons to select:

- All setups in the project
- All setups in a specified design: you select the design from the drop-down menu

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

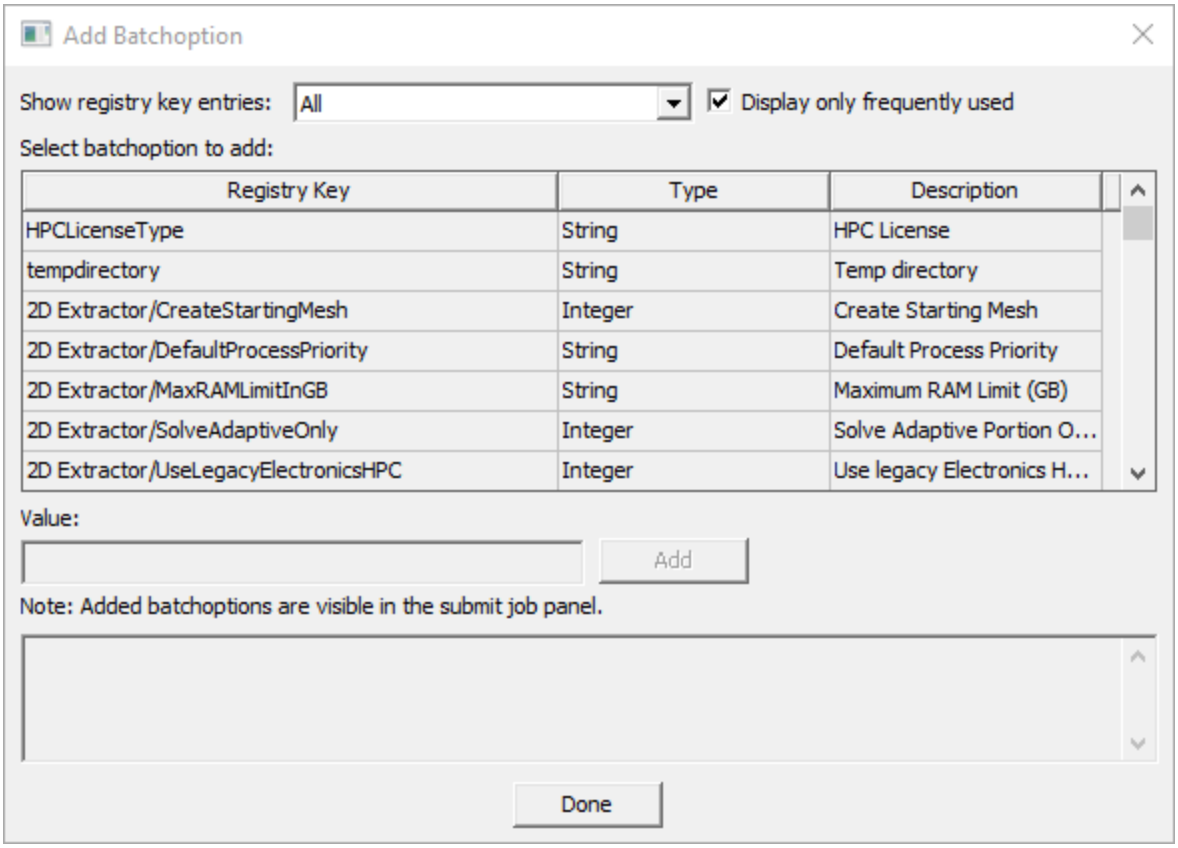
- Single setup. If you select the Submit Job command from the shortcut menu, the setup name populates the field.

If you specify multiple setups, they will be processed sequentially in the order displayed in the edit box.

The Analysis options include:

- Monitor job. You must enable this option to monitor the job from the user interface.
- Wait for license. Whether to wait until a license if available before starting a simulation.
- Batch options. You can optionally specify -batchoptions in the text field. See detailed discussion of -batchoptions beginning under ***Running Twin Builder from a Command Line***.

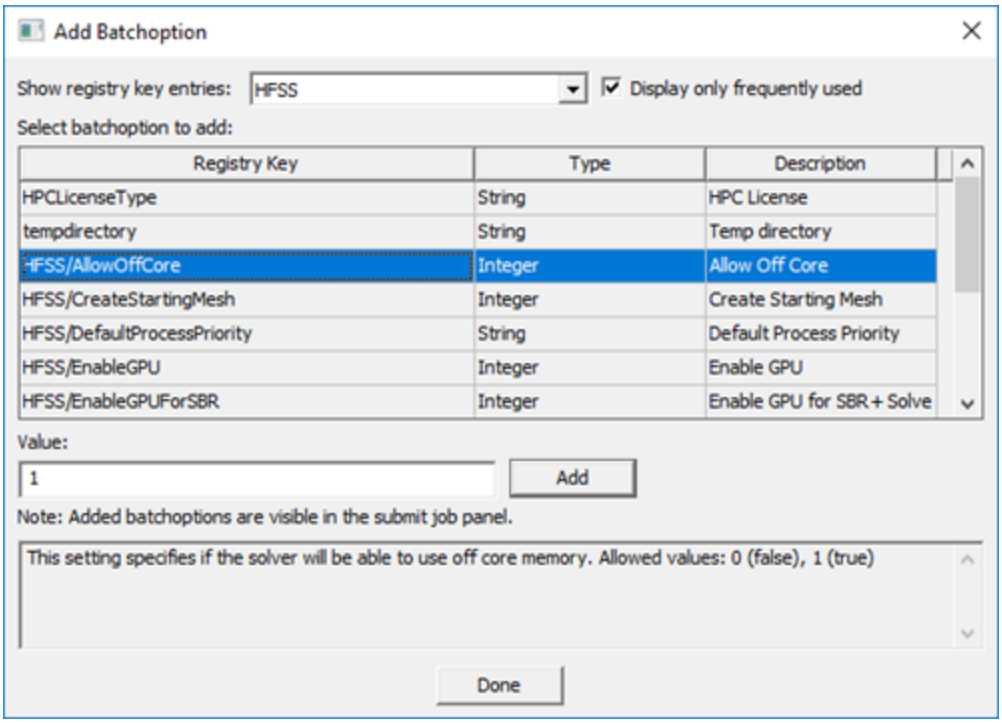
Click **Add...** to open the **Add Batchoption** dialog box.



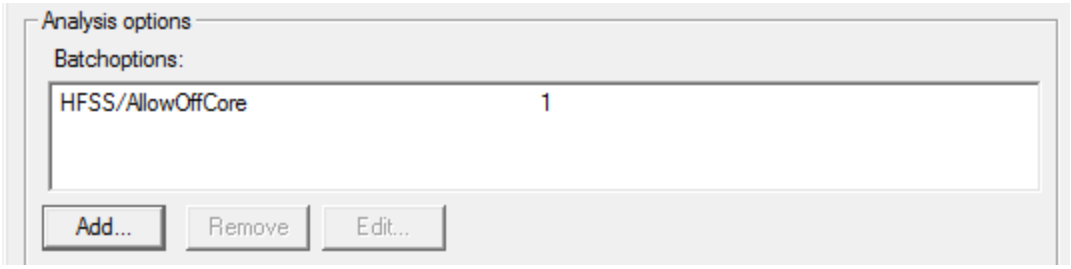
This dialog box provides access to all -batchoption commands. The drop-down menu lets you select specific categories, and you can choose to display only frequently used commands. You can edit and remove any batch options you specify.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Select a Registry Key in order to show the current Value for the type. The lower field explains the meaning of the Type Value.

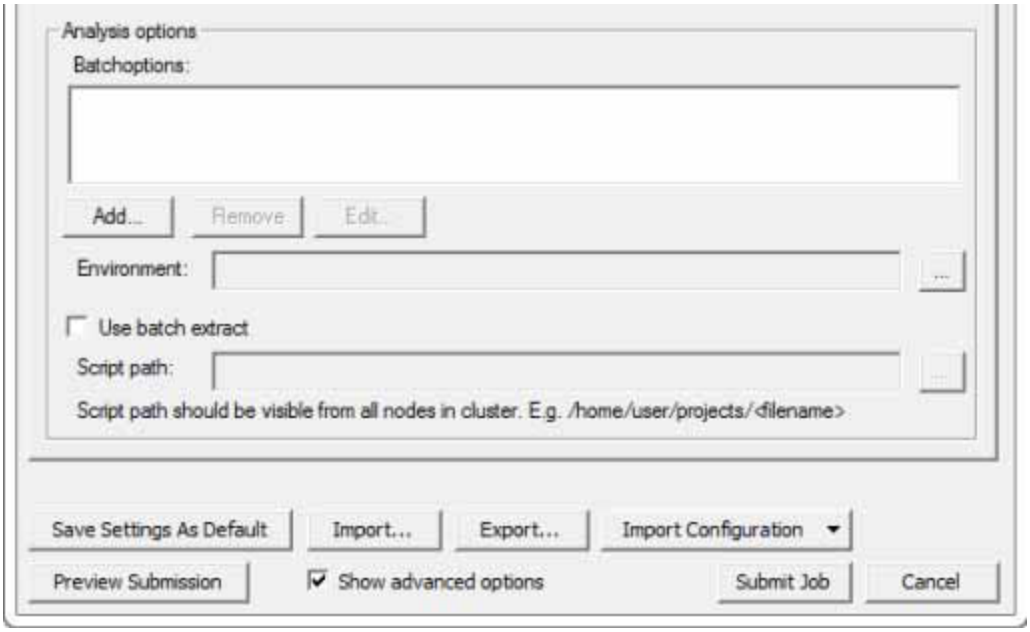


Any batchoptions for which you select **Add** will be visible in the *Submit Job* dialog box.

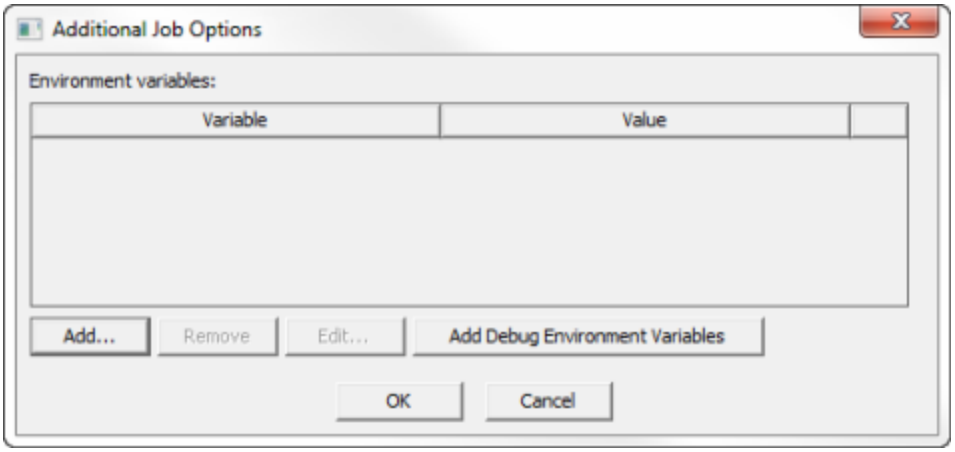


If you select the **Show advanced options** check box in the *Submit Job* dialog box, the Environment field and the Use batch extract fields display.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

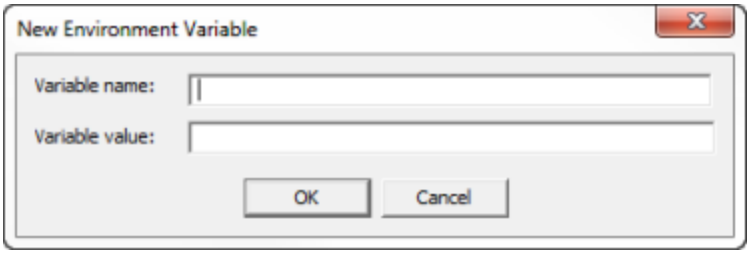


The Environment field lets you specify any Environment variables. Click ... to display the **Additional Job Options** dialog box.

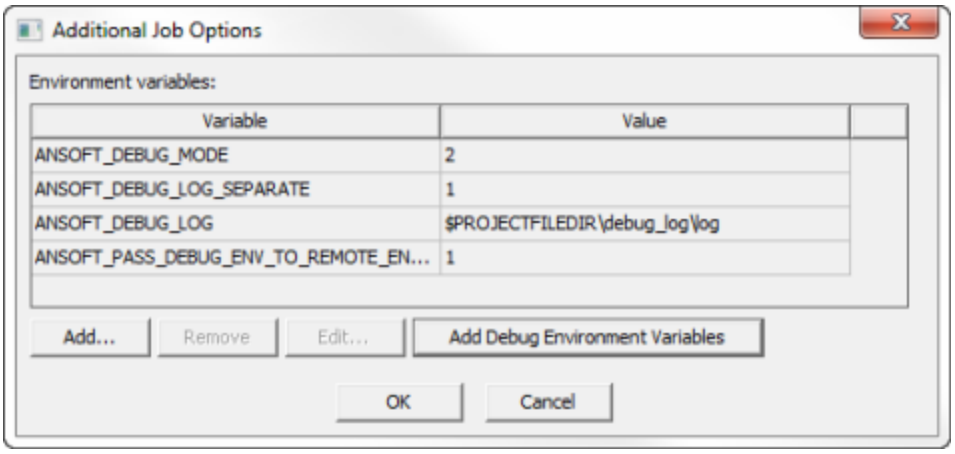


Click **Add...** to open the **New Environment Variable** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Here you can provide a Variable name and Variable value. Click **OK** to display the variable in the **Additional Job Options** dialog box. Select a variable to enables the **Remove** and **Edit...** buttons. You can also click **Add Debug Environment Variables**.

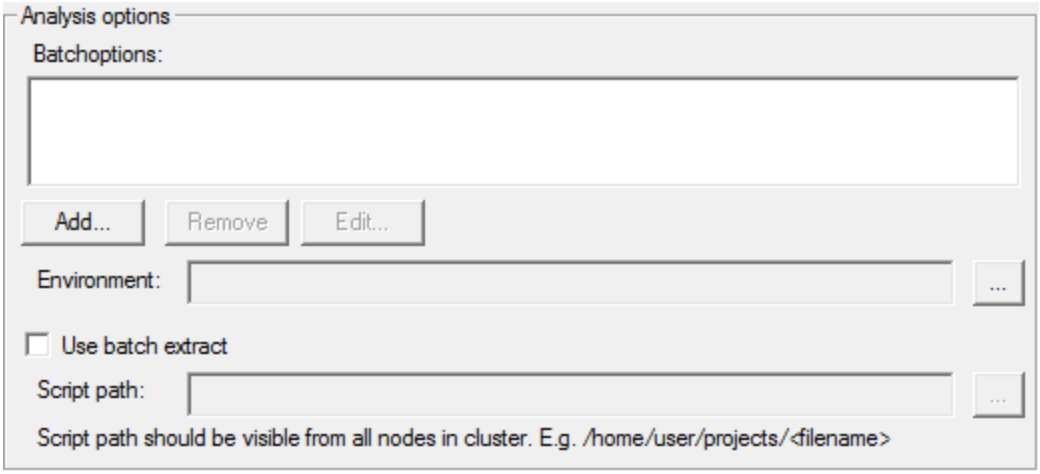


Any Variables that you add will be displayed in the Environment field of the **Submit Job** dialog box, if you have also enabled Show Advanced options.

Use Batch Extract for Windows HPC

Select **Show advanced options for Windows HPC** to display the **Use batch extract** fields.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

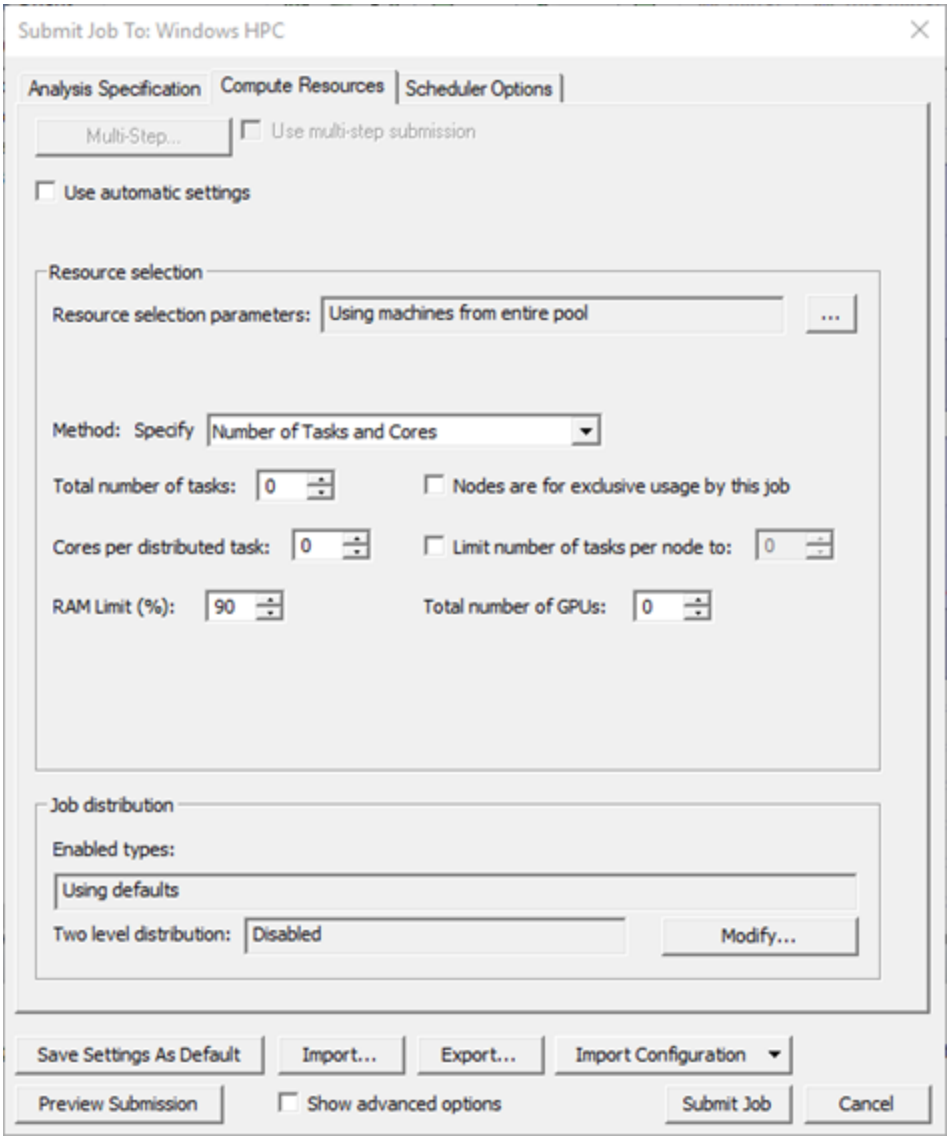


See ***Running Twin Builder from a Command line*** for a discussion of the solve information available through batch extract.

Click **Preview Submission** to open a window that shows the text commands that will be sent to the scheduler.

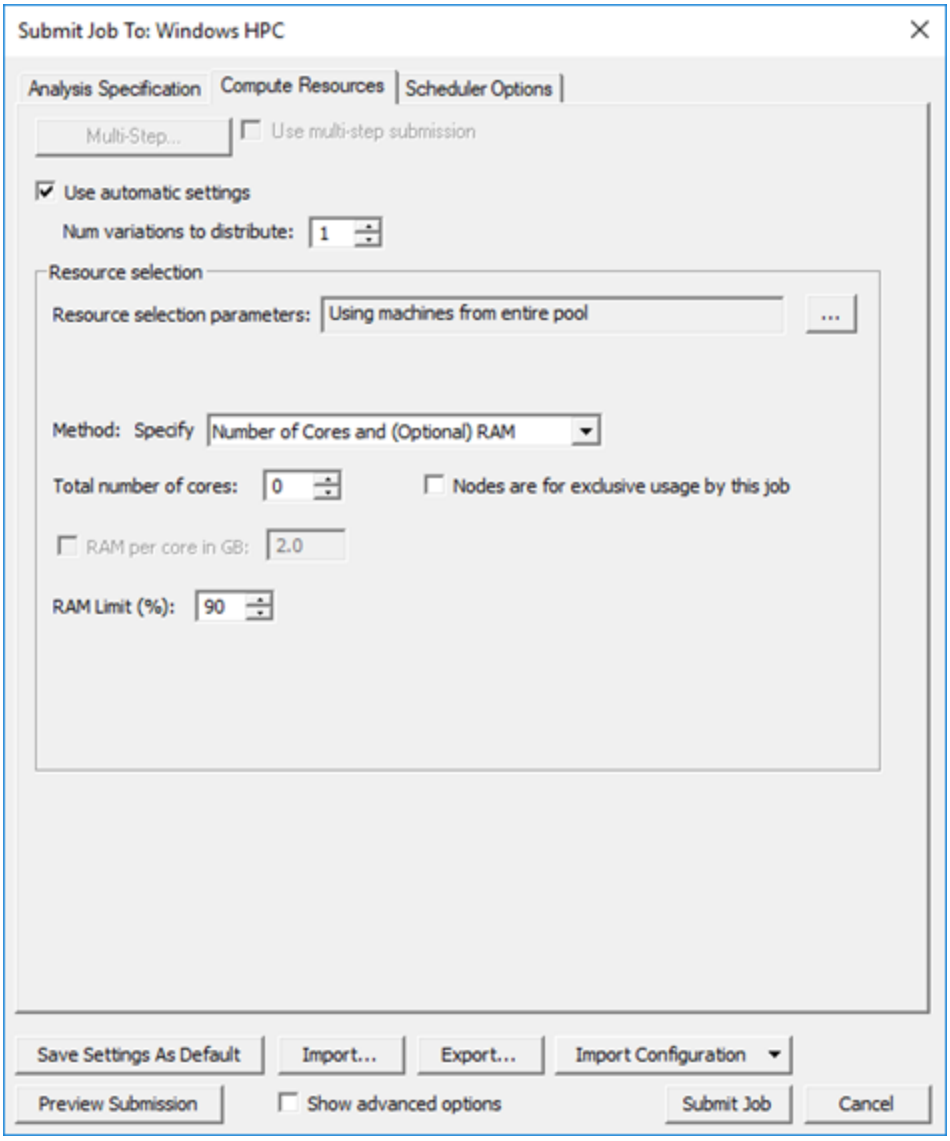
The following figure shows the **Compute Resources** tab of the **Submit Job To** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



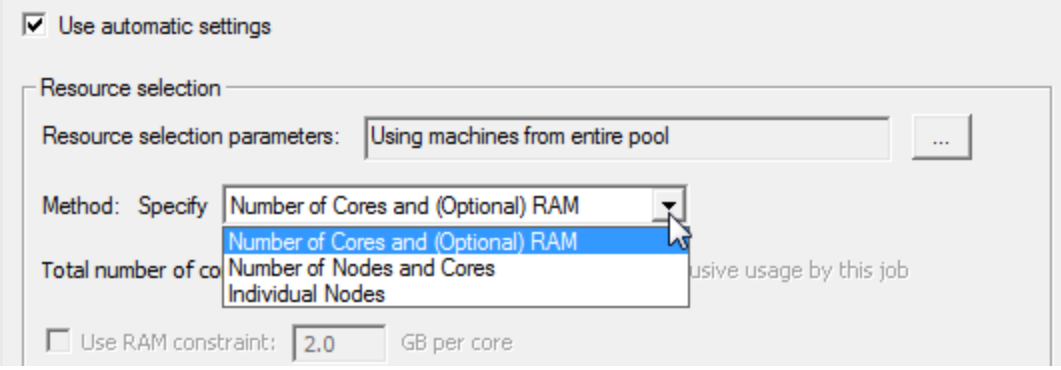
For Ansys Electronics Desktop configurations, the **Submit Job** dialog box includes a **Use automatic settings** check box that simplifies the **Compute Resources** tab.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The **Method** field of the **Submit Job To** dialog box has a drop-down menu with two or three selections, depending on whether you select **Use automatic settings**.

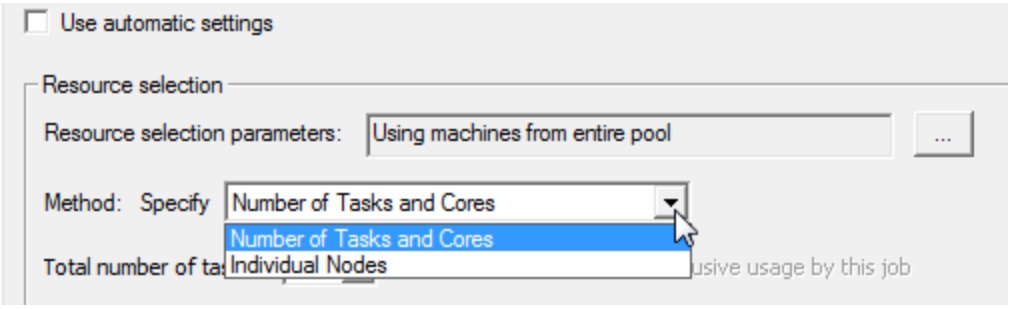
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

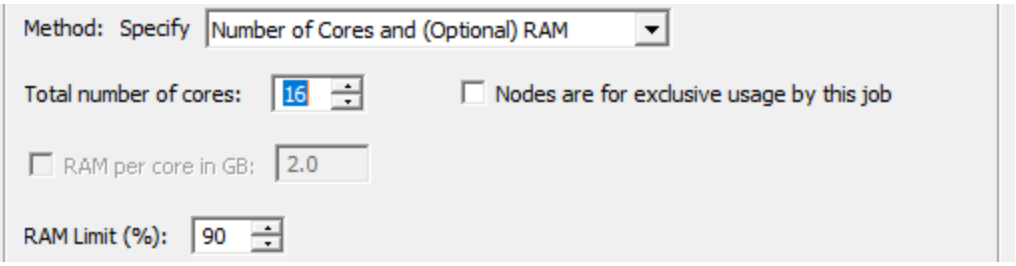
If you select Use automatic settings, Optimetrics variations will be solved sequentially. Other distribution types will be distributed automatically. It does distribute frequencies, domains, and use of multiple level domains.

If you clear or cannot access the **Use automatic settings** check box, these two methods are listed:



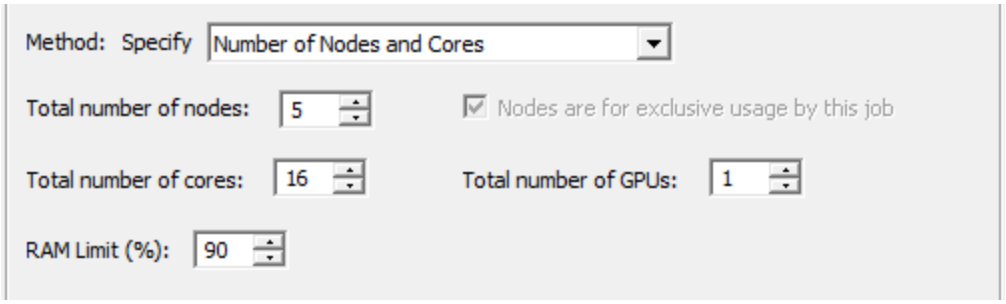
Each method selection changes the available options listed:

- Specify Number of Cores and (Optional) RAM

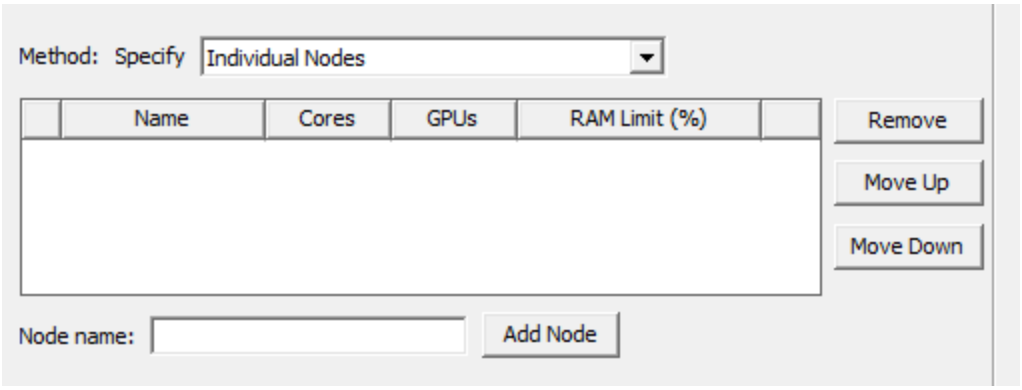


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Number of Nodes and Cores

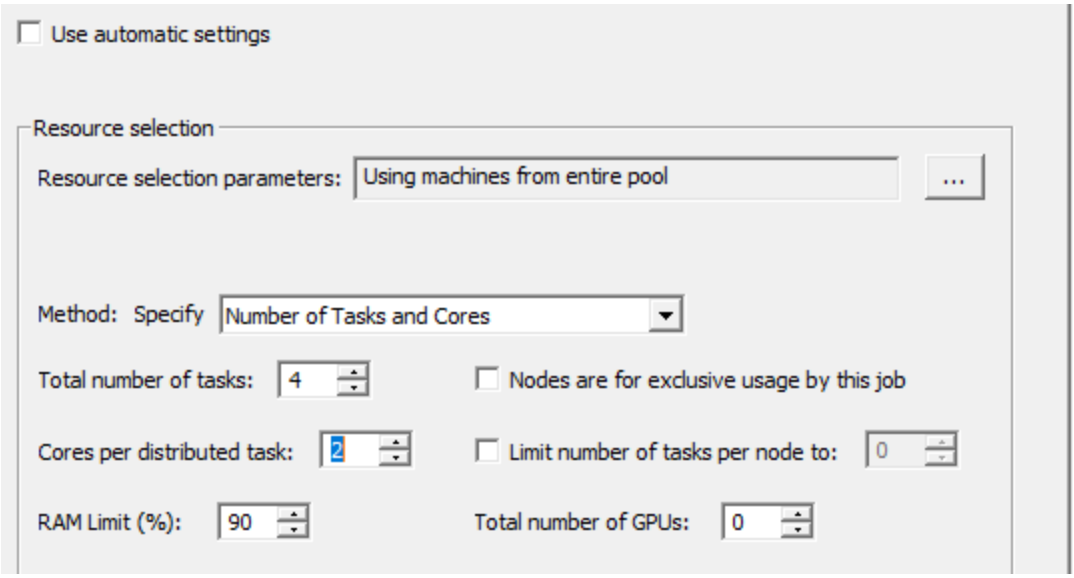


- Individual Nodes



- Number of Tasks and Cores (the **Use automatic settings** check box is cleared for this option. Selecting **Use automatic settings** means that you do not have to specify tasks or core parameters):

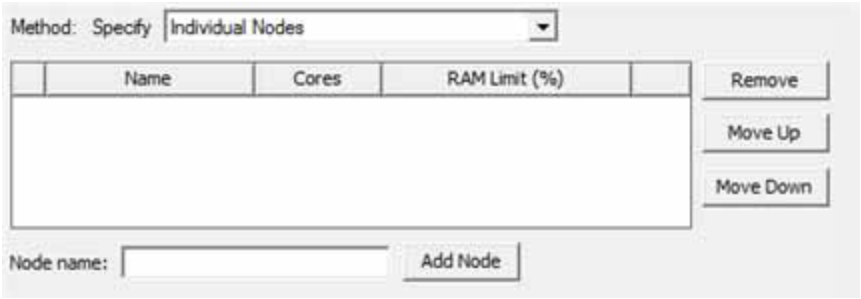
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Individual Node List

For Windows HPC jobs, you may either specify a node list, or specify job parallelization parameters, but not both.

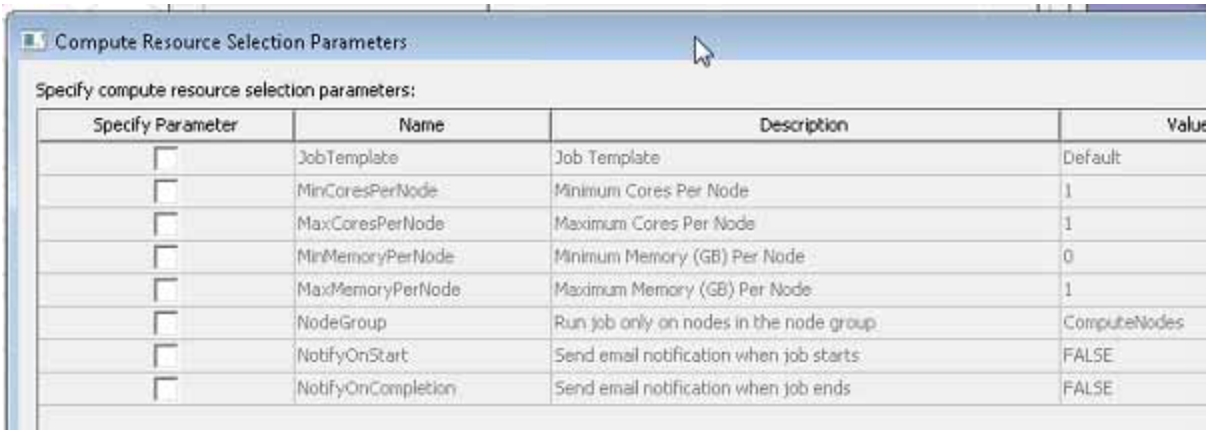
If you select the Individual Nodes Method, you may specify a node list, and the Job parallelization controls are disabled. In this case, the node list should only include cluster nodes that are valid for the job. For each node, you enter the node name and add the node. In the table, you can specify the number of cores and the RAM limit as a percentage. Click **Remove**, **Move Up**, and **Move Down** to edit and order the list.



Compute Resource Selection Dialog

By default, you can draw from the entire pool. You can also click ... to open a **Compute Resource Selection** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The resource selection parameters for Windows HPC jobs are:

- JobTemplate: Job Template - The JobTemplate may limit the job parameters or specify defaults values for job parameters
- MinCoresPerNode: Minimum Cores Per Node
- MaxCoresPerNode: Maximum Cores Per Node
- MinMemoryPerNode: Minimum Memory (GB) Per Node
- MaxMemoryPerNode: Minimum Memory (GB) Per Node
- NodeGroup: Run job only on nodes in the node group
- NotifyOnStart: If True, send email notification when job starts. Email notifications must be configured and enabled for the cluster by the administrator. (The cluster head node must run Windows HPC Server 2008 or above.)
- NotifyOnCompletion: If True, send email notification when job ends. Email notifications must be configured and enabled for the cluster by the administrator. (The cluster head node must run Windows HPC Server 2008 or above.)

Job Parallelization

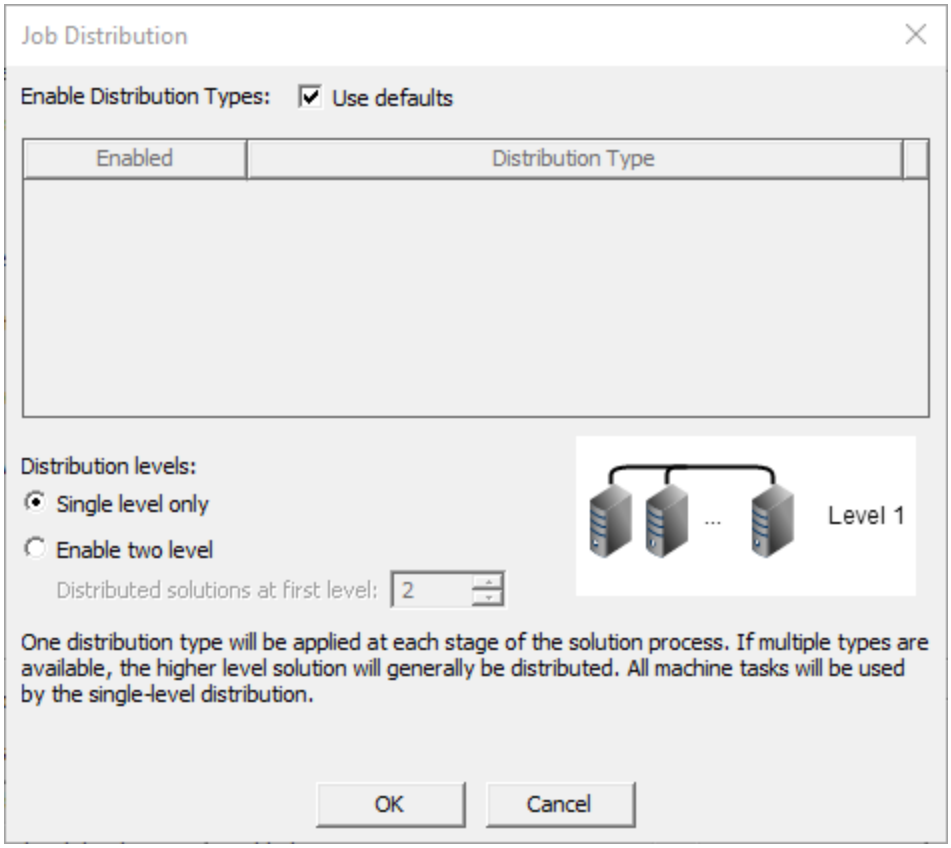
For Windows HPC jobs, you may either specify a node list, or specify the job parallelization parameters, but not both. The Job parallelization fields let you specify

- Total number of tasks: The number of nodes requested for the job is the total number of tasks divided by limit on the number of tasks per node, rounded up if it is not an integer.
- Cores per distributed task. This determines the amount of multiprocessing per task.
- Whether nodes are for exclusive usage by this job
- Whether to limit the number of tasks per node to a value. If the **Limit number of tasks per node** check box is not selected, then the job is submitted with a job unit type of "Core".

Job Distribution

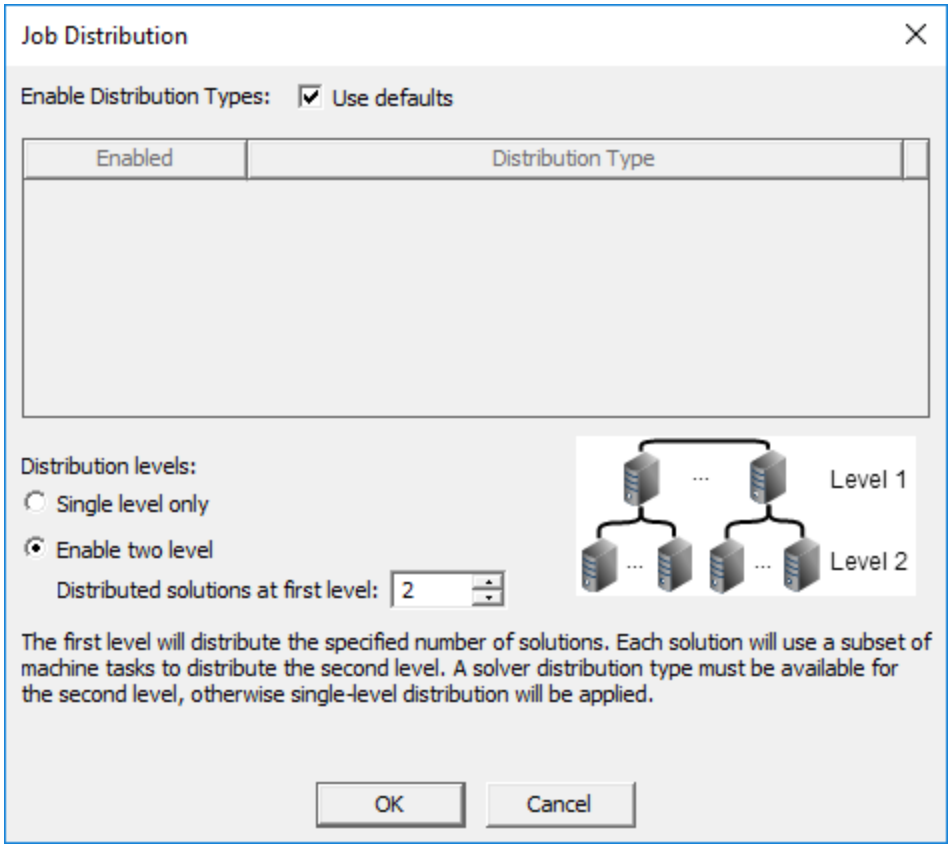
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Single level or two level distribution (*single level* is the default). Click **Modify** to display the **Job Distribution** dialog box and select the **Enable two level** option if applicable and desired.



- Second level distribution operates within DSO. If available and enabled you can specify the number distributed solutions for level 1.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

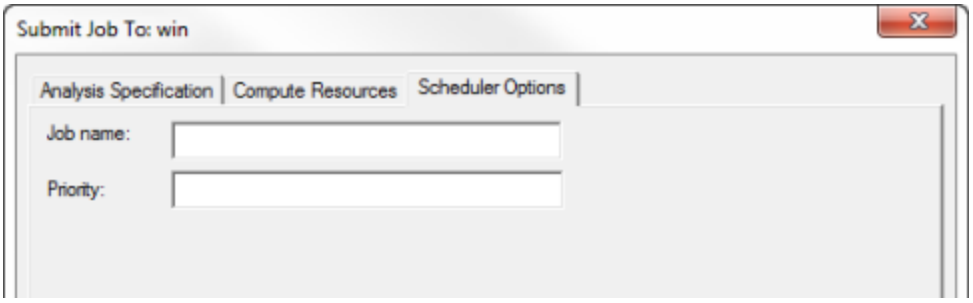


In response to a set of minimal constraints, the Scheduler may increase the resources assigned beyond the minimal values in order to meet the full set of requirements. For example, if you specify 7 distributed engines, with two processors per engine, and also limit the number of engines per node to 4, the scheduler may increase the number of cores used in order to meet the limit specified for engines per node. Notice that a preview of the Submit Job Results shows the number of resources assigned, and that the scheduler generated code includes an MPI specification.

Scheduler Options

The **Scheduler Options** tab provides for specifying the job name and/or the job priority. While the **Show advanced options** check box enables the display of job submission options, no job submission options should be specified for Windows HPC.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Preview Submission

Click **Preview Submission** to open a window that shows a text description of the job to be submitted and the task used to start the product on one of the nodes.

The JOB PARAMETERS section contains information on parameter that apply to the job as a whole.

- The "Job resource parameters" section indicates whether the job has exclusive use of nodes, the job unit type, and the minimum and maximum number of units requested for the job, node group, and email notifications.
- The "Job attributes" section displays the job name and job priority.
- The "User Specified Compute Resource Attributes" displays the Resource selection settings.

The TASK PARAMETERS section contains information on parameters that apply to the Desktop task, which is the main task of the job.

- The "Desktop task resource parameters" section indicates the job unit type (which is the same as in the JOB PARAMETERS), and the minimum and maximum number of units requested for the Desktop task.
- The "Command Line section" displays the desktop task command line, including all arguments.
- The "Environment variables" section displays the environment variables that are set for the Desktop task; the same environment variables will also apply to all other tasks of the job.
- The "Working directory" section indicates the working directory in which the Desktop task will run.

Monitor Job

If you selected the Monitor Job option on the **Submit Job To** dialog box, **AnalysisSpecification** tab, you can invoke the **Monitor Job** window by selecting **Tools** > **Job Management** > **Monitor Jobs**. This dialog box may also be brought up by selecting the **Begin monitoring this job now** check box when a job is successfully submitting using the job submission dialog box. .

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Windows® HPC Job Templates

Job templates are managed by the Windows HPC cluster administrator. Every cluster has at least one job template: the default job template. Every job has an associated template. If no job template is specified, then the default template is used. The job template controls two related aspects of the job submission process. When a job is submitted, there are a number of job parameters which may be specified. Each parameter has a set of valid values. For example, the Priority parameter has five valid values: Highest, AboveNormal, Normal, BelowNormal, and Lowest. The job template controls the default value of each parameter; this is the value that the parameter has if it is not specifically overridden by the submitter. For example, in the default job template, the default value of the Priority parameter is Normal. The job template may also limit the allowed values of each parameter to a subset of the valid values. For example, a job template for privileged users could allow all five Priority values, which a job template for unprivileged users could limit the allowed Priority values to Normal, BelowNormal, and Lowest.

Each job template is a Windows object with access controlled by an ACL (access control list). Instead of the usual "Read" or "Read & Execute" permissions, there is a "Submit Job" permission which corresponds to the right to submit a job with this job template. The cluster administrator may create job templates to limit or control access to cluster resources. For example, a job template with limited allowed job run times, or access to a limited set of compute nodes could be created by the cluster administrator. Specific users or user groups could be forced to use this limited job template by omitting access to the other job templates or by adding a deny access entry for the specified user or group to the other job templates. See the *Windows HPC Server 2008 Job Templates* white paper from Microsoft for additional details:

http://www.microsoft.com/en-us/download/confirmation.aspx?id=5659

You can create job templates to run jobs with limited knowledge of the appropriate job parameters. The cluster administrator creates a job template which has reasonable default values for the type of job to be run, and informs you which job template to use for each type of job. The template could also limit some parameters to only the subset of all values that are useful for the type of job associated with the template.

Related Topics

Integration With Microsoft Windows® HPC Scheduler

Selecting Computation Resource Units (Job Unit Type)

Windows® HPC Job Credentials

High Performance Computing (HPC) Integration

Selecting Computation Resource Units (Job Unit Type)

The job unit type is the smallest unit of processing resources used to schedule the job. This is one of the most important job properties. There are three options for the job unit type: cores,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

nodes, or sockets.

- **Cores** – Jobs are scheduled in units of cores, which may be also described as a CPU cores, logical processors, or CPUs. This is the smallest unit of granularity available. This selection allows the scheduler to start multiple tasks on a processor, if the total number of cores needed by the tasks is less than or equal to the number of cores on the processor. This selection lets the scheduler distribute more of the computational load to processors with more cores than to processors with fewer cores.
- **Nodes** – Jobs are scheduled in units of nodes, hosts, or machines. This is the coarsest level of granularity. When this option is selected, only one task will run on a node. This is useful in cases where it is not desirable to run multiple tasks on a single host. For example, if each task is multi-threaded, running multiple tasks on the same node may not be needed to fully utilize the computing resources on the node. This may also be preferred if the tasks are memory intensive, and multiple tasks would be competing for the limited memory resources.
- **Sockets** – A socket (which may also be called a NUMA node) is a collection of cores sharing a direct connection to memory. A socket contains at least one core. The socket concept may not necessarily correspond to a physical socket. Scheduling at the socket level may be useful in cases in which each task requires extensive use of the memory bus, and scheduling multiple tasks on the same socket would result in excessive bus contention.

Related Topics

Integration With Microsoft Windows® HPC Scheduler

Windows® HPC Job Templates

Windows® HPC Job Credentials

High Performance Computing (HPC) Integration

Windows® HPC Job Credentials

Normally, you are prompted for the credentials used to submit a job. One way to simplify this process is to use the **cluscfg setcreds** command to set your credentials in the credentials cache. If this is done, then no password needs to be supplied for a job submitted for the specified user. Here is a **cluscfg** command that sets the user credentials in the credentials cache:

```
cluscfg setcreds /password:* /scheduler:cluster_name _
/user:domain\user_name
```

Here:

- **cluster_name** – The name of the cluster (hostname of the head node).
- **domain** – An optional domain name; if omitted, the following **** should also be omitted.
- **user** – The user name.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

When this form of the command is used, you are prompted for the password and also asked if the password should be remembered (cached).

See the following Web page for more information on the **cluscfg setcreds** command:

http://technet.microsoft.com/en-us/library/cc947669(WS.10).aspx

Related Topics

Integration With Microsoft Windows® HPC Scheduler

Windows® HPC Job Templates

Selecting Computation Resource Units (Job Unit Type)

High Performance Computing (HPC) Integration

Command Line Information for Ansys Electromagnetics Desktop Products

You can specify any **Tools** > **Options** setting on the command line using corresponding registry keys.

Examples

```
ansysedt.exe -batchsolve
-batchoptions
"'Twin Builder/Preferences/TempDirectory'='G:/tmp'"
projectname.aedt
```

This example demonstrates how to set the same options as the previous example, but here using a **registry.txt** file:

```
ansysedt.exe -batchsolve -batchoptions registry.txt projectname.aedt
```

registry.txt contains:

```
$begin 'Config'
'Twin Builder/Preferences/TempDirectory'='G:/tmp'
$end 'Config'
```

Distributed Jobs

An Ansys Electromagnetics batch job which distributes the analysis over several hosts may also be called a distributed job. To submit a distributed job, the following Ansys Electromagnetics desktop command line options should be used:

- The **-Distributed** option is present, and the **-Local** option is absent. When running as a batch job under one of the schedulers with direct integration, this option is a directive to

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the job to:

- Obtain the list of hosts allocated to the job directly from the scheduler.
- Use the scheduler to launch the analysis engines on the hosts allocated to the job.

- Include the **-Machinelist num=***num_distributed_engines* option, where *num_distributed_engines* is the total number of analysis engines to be started on the hosts assigned to the job.

Other examples:

- Serial Job on a Single Processor
- Distributed Job using Four Processors

Serial Job on a Single Processor

Suppose Twin Builder is installed at **C:\Program Files\AnsysEM\v242\Win64** and you are using RSM for remote-analysis/DSO:

```
C:\Program Files\AnsysEM\v242\Win64\ansysedt.exe -ng -BatchSolve -
remote 10.1.1.221
-monitor \\shared_drive\projs\capacitor.aedt
```

Using LSF for remote-analysis/DSO:

```
bsub -n 1 C:\Program Files\AnsysEM\v242\Win64\ansysedt.exe -ng -
BatchSolve -monitor -local \\shared_drive\projs\capacitor.aedt
```

Distributed Job using Four Processors

RSM

```
C:\Program Files\AnsysEM\v242\Win64\ansysedt.exe-ng -Batchsolve -
monitor -Distributed
-machinelist list="10.1.1.221, 10.1.1.222, 10.1.1.223,_ 10.1.1.224"
\\shared_drive\projs\capacitor.aedt
```

LSF

```
bsub -n 4 C:\Program Files\AnsysEM\v242\Win64\ansysedt.exe -ng -
Batchsolve -monitor
-Distributed -machinelist num=4
\\shared_drive\projs\capacitor.aedt
```

Related Topics

Running Twin Builder from a Command Line

Scheduler Terminology

What a Scheduler Does

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Aborting an Analysis

Integrating Ansys Electromagnetics Suite with Third Party Schedulers

This section describes how to create a dynamically linked library to allow integration of Ansys Electromagnetics Suite with an arbitrary scheduler environment. Each scheduler proxy library is used for a single specific scheduler environment. If the library is installed with a valid name and in the correct location, then it loads and is used by Ansys Electromagnetics Suite.

Installation Details

The scheduler proxy library must be installed in the **schedulers** subdirectory of the Ansys Electromagnetics installation directory. For example, if the Ansys Electromagnetics installation directory is **C:\Program Files\AnsysEM\v**<*version number*>**\Win64**, then the scheduler proxy library must be installed in **C:\Program Files\AnsysEM\v**<*version number*>**\Win64\schedulers**.

The scheduler proxy library base name must match *libprefix_scheduler* on Windows. The extension must be a valid extension for a dynamically loaded library on the platform where it is used. The scheduler proxy library name prefix *libprefix* must be unique so it does not conflict with other scheduler proxy libraries in the same directory. To avoid confusion, the scheduler proxy library name must be all lower case on an OS where file names are case sensitive.

Related Topics

Build Information for Scheduler Proxy Library

Implementation Details for Custom Scheduler Integration

Testing Your Scheduler Integration

Troubleshooting Custom Scheduler Integration

Build Information for Scheduler Proxy Library

This section contains the recommended compiler and linker settings for building a scheduler proxy library.

Microsoft Windows

The proxy library must be compiled and linked as a 32-bit DLL using the following recommended compiler and linker options:

Compiler Options

- Use of MFC: Use Standard Windows Libraries
- Character Set: Use Multi-Byte Character Set [/D "_MBCS"]

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Runtime Library: Multi-threaded DLL [/MD]
- Calling Convention: __cdecl [/Gd (default)]

Linker Options

- Create a DLL [/DLL]
- 32 bit code [MACHINE:X86]

Implementation Details for Custom Scheduler Integration

Function Name Prefix

Each exported function will have a scheduler specific function name prefix. The function name prefix is the same as the library name prefix, except that it is converted to upper case. For example, if the library name prefix is **pbs**, then the function name prefix is **PBS**. In the examples below, **FN_PREFIX** indicates the function name prefix.

The scheduler proxy library must provide implementations of the following extern "C" functions:

- IsProductLaunchedInYourEnvironment
- GetTempDirectory
- GetMachineListAvailableForDistribution
- LaunchProcess
- GetUseRsmForEngineLaunch
- GetThisJobID
- GetSchedulerDisplayName

IsProductLaunchedInYourEnvironment

Purpose

Determine if the program is running in the context of the scheduler for which this library was written.

Signature

```
extern "C" bool FN_PREFIX_IsProductLaunchedInYourEnvironment();
```

Arguments

None.

Return Value

Returns TRUE if the current process is running as a job of the scheduler. Otherwise returns FALSE.

Notes

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For many schedulers, the presence of certain environment variables or their values are checked to determine if the current process is running as a job of the scheduler.

GetTempDirectory

Purpose

Get the path name of the temporary directory provided by the scheduler for the current job. The path name is an empty string if the scheduler does not provide a temporary directory for the current job.

Signature

```
extern "C" bool FN_PREFIX_GetTempDirectory(char * buffer, unsigned
int* length);
```

Arguments

buffer – Pointer to a character buffer to contain the temporary directory path name or NULL.

length – Pointer to a location to contain the length of the buffer. Must be a valid pointer to an unsigned int.

Return Value

If argument buffer is NULL, then the required length of the buffer is stored in the location to which the **length** argument points, and TRUE is returned.

If argument buffer is not NULL, then the value to which the **length** argument points (the buffer length) is checked. If it is large enough to contain the path name of the temporary directory, including the terminal null byte, then the path name is copied to the buffer and TRUE is returned. If the buffer length is insufficient for the path name of the temporary directory, then the buffer is unchanged, and FALSE is returned.

Notes

To get the path name of the temporary directory, the infrastructure calls this function with a NULL buffer, and obtains the required length of the buffer for the path name. After creating a buffer of the appropriate size, the infrastructure calls this function again, passing the pointer to the buffer in the **buffer** argument and a pointer to the size of the buffer in the **length** argument.

GetMachineListAvailableForDistribution

Purpose

Get the list of hosts allocated to the current job. A host will appear in the list multiple times if the scheduler has allocated multiple processors or cores on the host to the job. The number of times the host appears in the list is equal to the number of processors or cores of the host that are

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

allocated to the current job. The list is a text string containing a space separated list of host names.

Signature

```
extern "C" bool FN_PREFIX_GetMachineListAvailableForDistribution(char
 * buffer, unsigned int* length);
```

Arguments

buffer – Pointer to a character buffer to contain the list of machines available for distribution or NULL.

length – Pointer to a location to contain the length of the buffer. Must be a valid pointer to an unsigned int.

Return Value

If the **buffer** argument is NULL, then the required length of the buffer is stored in the location to which argument length points, and TRUE is returned.

If the **buffer** argument is not NULL, then the value to which the **length** argument points (the buffer length) is checked. If it is large enough to contain the lists of hosts, including the terminal null byte, then the list is copied to the buffer and TRUE is returned. If the buffer length is insufficient for the list of hosts, then the buffer is unchanged, and FALSE is returned.

Notes

To get the list of hosts for distribution, the infrastructure first calls this function with a NULL buffer, and obtains the required length of the buffer for the list. After creating a buffer of the appropriate size, the infrastructure calls this function again, passing the pointer to the buffer in the buffer argument and a pointer to the size of the buffer in the length argument.

The host names in the list provided by this function are used in calls to LaunchProcess(). These host names must be in a format that is accepted by that function. See ***LaunchProcess***.

LaunchProcess

Purpose

Launch a local or remote process to run an analysis engine. This function is called by the Ansys Electromagnetics Desktop to launch an engine process on a specified host. The host name is one of the names in the list provided by the ***GetMachineListAvailableForDistribution*** function. If the host name does not refer to the local host, then this function shall use the scheduler to launch the engine on the specified host. If the host name refers to the local host, then the engine may be started as a child process, or it may be started using the scheduler.

Signature

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
extern "C" int FN_PREFIX_LaunchProcess(const char* hostName, const
char* exePathName, const char* arg1, const char* arg2);
```

Arguments

hostName – The name of the host where the process is to be launched.

exePathName – The path name of the analysis engine executable to be started.

arg1 – The first argument of the analysis engine command line.

arg2 – The second argument of the analysis engine command line.

Return Value

Returns 0 on success. Returns a non-zero value if an error occurs.

Notes

The **hostName** argument will be one of the host names provided by the **GetMachineListAvailableForDistribution() function**.

If the **hostName** argument is the same as the current host, then the analysis engine process may be started as a child process. If the **hostName** argument is not the same as the current host, then the analysis engine process will be started on the remote host using the facilities available in the scheduler environment. The command line of the analysis engine process is **exePathName arg1 arg2**. The command line arguments **arg1** and **arg2** may contain newlines, tabs, spaces or other characters that are interpreted by the command processor, such as single quote (') or double quote (") characters, or dollar signs ($). Newlines or tabs may be replaced by spaces, if the newline or tab characters cannot be easily handled. If the analysis engine command is processed by a shell, then it may be necessary to quote any special characters in the **exePathName** or in the arguments so that the special meaning is removed. If you use a scheduler command to request the scheduler to launch the command to start the engine process, the analysis engine command may be processed by the shell twice: once when the scheduler command is processed, and a second time when the analysis engine process is started. If this is the case, then the quoting of special characters needs to account for two passes through the command processor.

GetUseRsmForEngineLaunch

Purpose

This function is optional. If this feature is not needed, then the function need not be implemented. Most schedulers should not need this feature.

For some schedulers, it may be desirable for the RSM service to launch the engine processes instead of using the scheduler proxy library. For example, if the scheduler proxy library is limited to launching one process per host, then the scheduler proxy library may be

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

used to launch one RSM service executable per host, and the RSM executable will launch all of the engine processes.

If the RSM service is used to launch engine processes for this scheduler, then this function shall be implemented and it returns TRUE.

If the RSM service should not be used to launch engine processes for this scheduler, then this function is not required. If it is implemented, it should return FALSE If it is not implemented, it will be treated the same as if it was implemented and returns FALSE.

Signature

```
extern "C" bool FN_PREFIX_GetUseRsmForEngineLaunch(void)
```

Arguments

None.

Return Value

Returns TRUE if the RSM service should be used to launch engine processes for this scheduler. Returns FALSE if the RSM service should not be used to launch engine processes for this scheduler.

Notes

This function is optional. If not implemented, then it is treated the same as if it was implemented and returns FALSE.

GetThisJobID

Purpose

Get a string identifying the job currently running in the scheduler environment. This string appears to the end user to identify the job.

Signature

```
extern "C" bool FN_PREFIX_GetThisJobID(char * buffer, unsigned int*
length);
```

Arguments

buffer – Pointer to a character buffer to contain the Job ID or NULL.

length – Pointer to a location to contain the length of the buffer. Must be a valid pointer to an unsigned int.

Return Value

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If argument buffer is NULL, then the required length of the buffer is stored in the location to which argument length points, and TRUE is returned.

If argument buffer is not NULL, then the value to which the **length** argument points (the buffer length) is checked. If it is large enough to contain the string identifying the current job, including the terminal null byte, then the job ID is copied to the buffer and TRUE is returned. If the buffer length is insufficient for the job ID, then the buffer is unchanged, and FALSE is returned.

Notes

To get the job ID, the infrastructure first calls this function with a NULL buffer, and obtains the required length of the buffer for the job ID. After creating a buffer of the appropriate size, the infrastructure calls this function again, passing the pointer to the buffer in the buffer argument and a pointer to the size of the buffer in the length argument.

For many schedulers, the job ID may be obtained from the value of an environment variable.

GetSchedulerDisplayName

Purpose

Get a string identifying the scheduler associated with the current scheduler proxy library. This string appears to the end user to identify the scheduler.

Signature

```
extern "C" bool FN_PREFIX_GetSchedulerDisplayName(char * buffer,
unsigned int* length);
```

Arguments

buffer – Pointer to a character buffer to contain the scheduler display name or NULL.

length – Pointer to a location to contain the length of the buffer. Must be a valid pointer to an unsigned int.

Return Value

If argument buffer is NULL, then the required length of the buffer is stored in the location to which argument length points, and TRUE is returned.

If argument buffer is not NULL, then the value to which argument length points (the buffer length) is checked. If it is large enough to contain the scheduler display name, including the terminal null byte, then the scheduler display name is copied to the buffer and TRUE is returned. If the buffer length is insufficient for the scheduler display name, then the buffer is unchanged, and FALSE is returned.

Notes

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To get the scheduler display name, the infrastructure first calls this function with a NULL buffer, and obtains the required length of the buffer for the scheduler display name. After creating a buffer of the appropriate size, the infrastructure calls this function again, passing the pointer to the buffer in the buffer argument and a pointer to the size of the buffer in the length argument.

The scheduler display name is generally a fixed string.

Scheduler Proxy Interfaces

Scheduler proxy supports the following graphical interface functions. The scheduler-specific prefix of each function is not shown in this listing.

void Initialize(const std::string& config):

Initialize the proxy library for scheduler interaction. The **config** argument contains scheduler specific initialization information.

int CheckEnvironment(std::string& msg):

Check the environment in which the proxy library is running.

- Returns zero (success) if the environment is appropriate for submitting jobs to the scheduler.
- Returns a non-zero error code if the environment is incorrect. If a non-zero error code is returned, the system writes an error message to the **msg** argument.

int GetSchedulerInfo(std::string& msg, std::string& schedulerName, std::string& schedulerDescription, std::string& schedulerVersion):

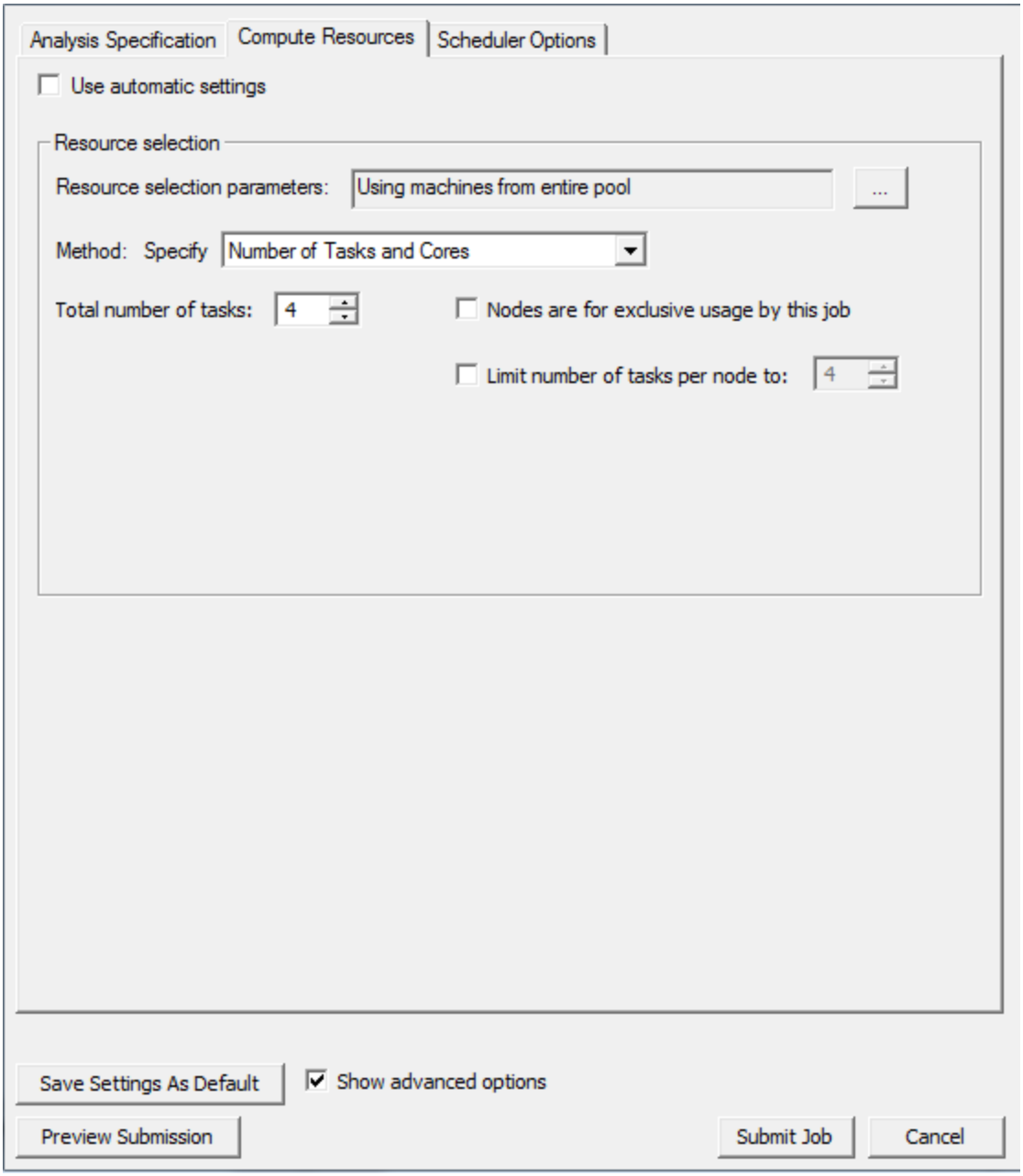
This function returns some basic information about the scheduler with which the scheduler proxy library interacts.

- On success, zero returns, and the system writes the scheduler name, scheduler description, and scheduler version to the **schedulerName**, **schedulerDescription** and **schedulerVersion** arguments.
- On failure, a non-zero error code is returned, and the system writes an error message to the **msg** argument.

int GetComputeResourceAttributes(std::string& msg, AttributeDefinitionsStruct& attributeDefs):

Use the **Compute Resource Selection Parameters** dialog box to specify scheduler specific resources. This function returns the information used to create and populate the **Compute Resource Selection Parameters** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Each line in the dialog box is defined by a single attribute definition in the **attributeDefs** argument. An attribute definition defines the name and description of an attribute, as well as information about the allowed values and the default value. In general, only the most commonly specified job attributes are included in the **attributeDefs** argument.

- On success, zero returns, and the system writes the attribute definitions to the **attributeDefs** argument.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- On failure, a non-zero error code returns, and the system writes an error message to the **msg** argument.
- If the scheduler proxy library does not support any attributes using this approach, the **attributeDefs** argument will contain no attribute definitions, and zero returns.

int AbortJob(std::string& msg, const std::string& jobID, bool force, const SubmissionUserStruct& submissionUser):

This function requests the scheduler to abort a job identified by the **jobID** argument. If the force argument is true, then errors should be ignored (the exact behavior is scheduler specific). The **submissionUser** argument contains information about the client user (the user running the Desktop process). The request to abort the job should run in the context of this user. If no user is specified, then the request to abort the job runs as the user of the process or thread running the function.

- If the request is successfully submitted, then zero returns.
- If there is an error, then a non-zero error code returns, and the system writes an error message to the **msg** argument.

int SubmitUniformJob(std::string& msg, std::string& jobID, const CmdLineStruct& cmdLineInfo, const JobParallelizationStruct& jobParallelization, const UniformComputeResourcesStruct& computeResources, const JobOptionsStruct& jobOptions, const JobAtributesStruct& jobAttributes, const SubmissionUserStruct& submissionUser,const IJobParameters* jobParametersCB):

This function submits a job to the scheduler.

- On success, zero returns, and the system writes a job identifier of the newly submitted job to the **jobID** argument.
- On failure, a non-zero error code returns, and the system writes an error message to the **msg** argument.

This function is used to submit jobs to the scheduler in which the resources allocated to the job are uniformly distributed across the nodes allocated to the job. All other arguments are input arguments, and they are described below:

The **cmdLineInfo** argument contains the command line arguments. The first argument is the command name.

The **jobParallelization** argument contains information on how the job should be parallelized. It contains these integral parameters:

- Total number of distributed engines.
- Number of cores to allocate for each distributed engine.
- Maximum number of engines to allocate to a single node (optional).
- Number of cores to allocated for the non-distributed portion of the analysis.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

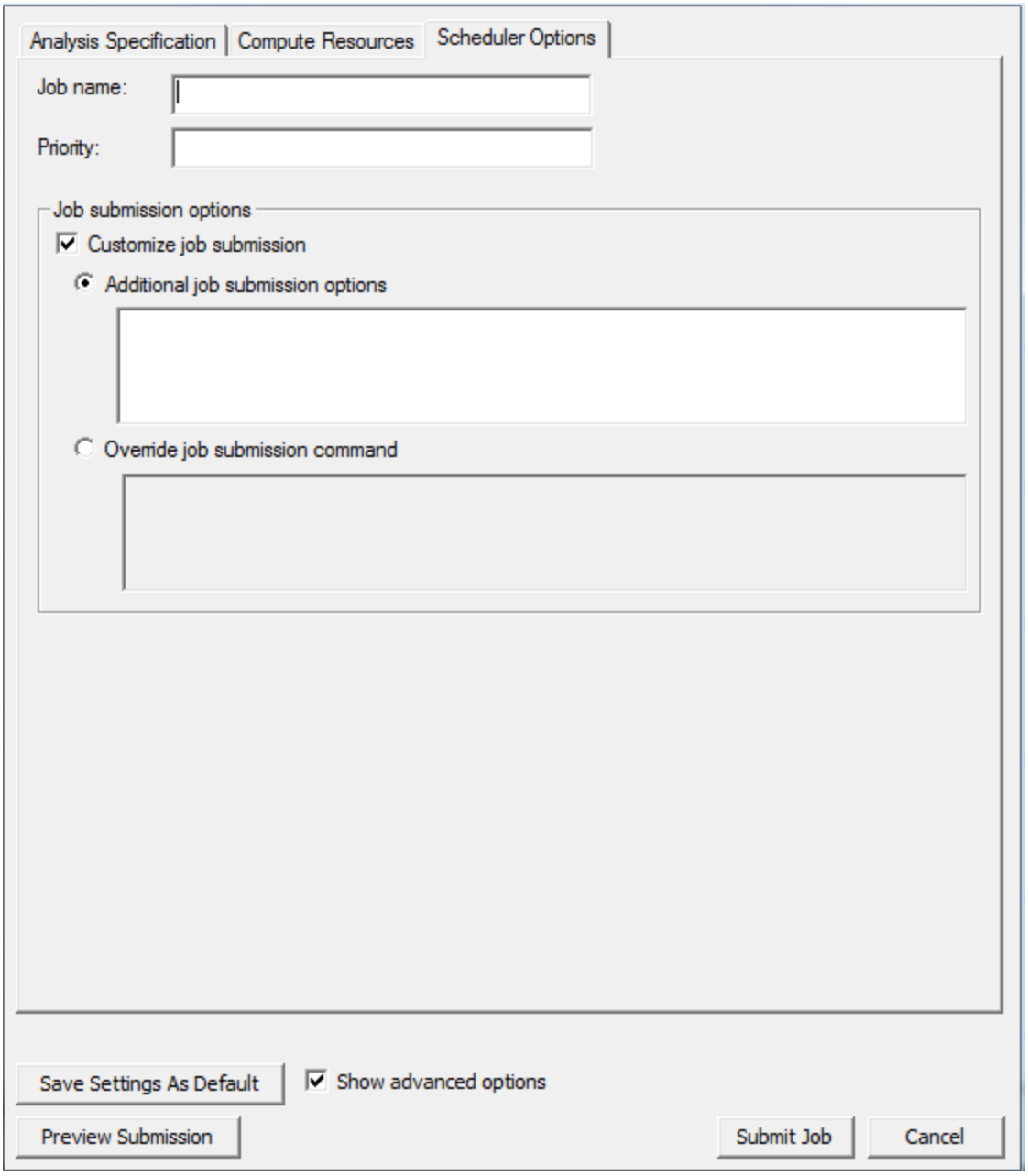
The argument also contains a Boolean parameter indicating whether nodes used for this job should be exclusively allocated to this job.

The **computeResources** argument is a reference to an object of type **UniformComputeResourcesStruct**. This **struct** contains zero or more resource attribute settings for the job. Each resource attribute setting consists or a resource name and a resource value. The resource name is the name of one of the resources defined in the **AttributeDefinitionsStruct** filled in by the **GetComputeResourceAttributes()** function. The resource attribute value is the value specified for the resource attribute using the **Compute Resource Selection Parameters** dialog box. If no resource attributes are specified in this dialog box, then the **computeResources** argument will contain no resource attribute settings.

The **jobOptions** argument contains the environment variable settings for the job.

The **jobAttributes** argument contains job submission attributes which are not necessarily related to the compute resources allocated to the job. The job name and the requested job priority are included in this data structure. Use the **SchedulerOptions** tab of the **Job Submission** dialog box to specify additional job submission options or all submission options, replacing the settings from the other **Job Submission** dialog box controls.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The user-specified submission options are included in this data structure, as well as a Boolean setting indicating whether the user-specified options are in addition to the generated options, or whether they replace the generated submission options.

The **submissionUser** argument contains information about the client user (the user running the Desktop process). The job is submitted to the scheduler to run as this user.

The **jobParametersCB** argument is a pointer to an object that implements the **IJobParameters** interface. This interface allows the scheduler proxy library to get additional information about the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

job. Specifically, the **GetWorkingDirectory()** interface function returns the working directory to be used for the job.

The **cmdLineInfo** argument contains the command line arguments. The first argument is the command name.

int SubmitNonUniformJob(std::string& msg, std::string& jobID, const CmdLineStruct& cmdLineInfo, const JobParallelizationStruct& jobParallelization, const NonUniformComputeResourcesStruct& computeResources, const JobOptionsStruct& jobOptions, const JobAtributesStruct& jobAttributes, const SubmissionUserStruct& submissionUser,const IJobParameters* jobParametersCB):

This function submits a job to the scheduler.

- On success, zero returns, and the system writes the job identifier of the newly submitted job to the **jobID** argument.
- On failure, a non-zero error code returns, and the system writes an error message to the **msg** argument.

This function is used to submit jobs to the scheduler in which the nodes to use and the number of engines to run on each node are user-specified. All other arguments are input arguments, as for the **SubmitUniformJob()** function. These input arguments are the same as for the **SubmitUniformJob()** function, except that the **computeResources** argument is a reference to a **NonUniformComputeResourcesStruct**, as described below:

The **computeResources** argument is a reference to an object of type **NonUniformComputeResourcesStruct**. This object contains a vector of pairs, where each pair consists of the name of a node in the cluster, and the number of engines to run on the node.

int PreviewUniformJob(std::string& msg, std::string& preview, const CmdLineStruct& cmdLineInfo, const JobParallelizationStruct& jobParallelization, const UniformComputeResourcesStruct& computeResources, const JobOptionsStruct& jobOptions, const JobAtributesStruct& jobAttributes, const SubmissionUserStruct& submissionUser, const IJobParameters* jobParametersCB):

This function is similar to the **SubmitUniformJob()** function, but instead of submitting the job, text representing how the job will be submitted is written to the preview argument. Typically the preview text includes the job submission command and the contents of the job script created for the job. For some schedulers, this content may not be meaningful, so the text returned could be different.

- On success, zero returns, and the system writes a job preview text to the **preview** argument.
- On failure, a non-zero error code returned, and the system writes an error message to the **msg** argument.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The other arguments are input arguments with the same meaning as for the **SubmitUniformJob()** function. The **submissionUser** argument is ignored for this function.

Testing Your Scheduler Integration

One way to test these functions is to run the analysis for an Ansys Electromagnetics product in batch mode. When running in batch mode, the system creates a batch log file in the same directory as the project file. The batch log file has the same base name as the project file, with a **.log** file extension. For example, if the project file is **TestProject123.aedt**, then the batch file is **TestProject123.log**. The batch log file contains useful information about the analysis run.

See the product-specific help for details on running the product in batch mode, and for the command line options to use for distributed analysis.

- Testing IsProductLaunchedInYourEnvironment
- Testing GetSchedulerDisplayName and GetThisJobID
- Testing GetTempDirectory
- Testing GetMachineListAvailableForDistribution
- Testing LaunchProcess
- Testing GetUseRsmForEngineLaunch

Testing IsProductLaunchedInYourEnvironment

This function should be tested first. If the Ansys Electromagnetics Desktop cannot load and run this function, or if it returns FALSE, then none of the other functions are called. If the batch analysis is running in a scheduler environment, and this function returns TRUE, then there will be an “info” message near the beginning of the batch log indicating that the analysis is running as a scheduler job. This message includes the scheduler display name returned by the **GetSchedulerDisplayName** function, and it will also include the job ID returned by the **GetThisJobID** function. If the batch analysis is not running in a scheduler environment, then none of the messages include a scheduler display name or job ID.

If this message does not appear when running in a scheduler environment, ensure that the scheduler proxy library is named correctly, built correctly, installed in the correct directory, and that the function name prefix is the same is the library prefix converted to upper case.

Testing GetSchedulerDisplayName and GetThisJobID

As described previously, when running a batch job in a scheduler environment the scheduler display name and job ID appear in an “info” message near the beginning of the batch log. The values returned by these functions are copied to this message, so you can compare them to the expected values.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Testing GetTempDirectory

Unfinished. The temp directory displayed in the batch log is the default installation setting, not the one from the scheduler. The scheduler temp directory is set in AnsoftCOMApplication::MainFunction(), so it happens for COM engines, but not for the Ansys Electronics Desktop.

Testing GetMachineListAvailableForDistribution

This function is used for distributed analysis. The analysis can be distributed across several machines if portions of the analysis are independent. For example, frequency sweeps, parametric analysis, and domain decomposition allow different portions of the analysis to be distributed across machines. The analysis in a batch job is distributed to multiple processors or hosts if the analysis includes a setup that may be distributed (for example, a frequency sweep or parametric analysis) and the **-Distributed** option is included in the desktop command line. The list of machines appears in an **info** message near the beginning of the batch log. You can compare the list in the info message to the expected list of machines.

To verify that the machine list is constructed correctly for a variety of cases, it may be necessary to test several jobs with different resource requirements and verify that the machine list is correct in each case. For example, one may run batch analyses with the following resource requirements:

- One processor on one host
- Several processors on one host
- One processor on each of several hosts
- Several processors on each of several hosts

Testing LaunchProcess

This function launches analysis engines when the analysis is distributed across multiple hosts. The analysis may be distributed across several machines if portions of the analysis are independent. For example, frequency sweeps, parametric analysis and domain decomposition allow different portions of the analysis to be distributed across machines. The analysis in a batch job is distributed to multiple processors or hosts if the analysis includes a setup that may be distributed (for example, a frequency sweep or parametric analysis) and the **-Distributed** option is included in the desktop command line. The list of machines appears in an **info** message near the beginning of the batch log. The batch log may also contain info messages when portions of the analysis distributed to different machines start or finish. These messages usually include the name of the host when the analysis ran or will run. You can verify that the analysis is actually running on the expected host or hosts using the Windows Task Manager.

In general, one analysis engine starts for each occurrence of each host in the list of machines available for distribution. For example, if the list of hosts is "hostA hostA hostA hostB hostB",

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

then a total of 5 engines would be started, three on hostA and two on hostB. In some cases, an additional engine is started to perform the portion of the analysis which is not distributed; if this is the case, the non-distributed engine is idle during the portion of the analysis which is distributed. If this occurs in the case where the list of hosts is "hostA hostA hostA hostB hostB", then a total of 6 engines would be started, but at most 5 engines would be active at any given time. When each analysis engine is running, it may start additional child processes to do a portion of the analysis, but these are not counted as additional analysis engines because the parent of the sub-engine is inactive (waiting for the sub-engine results) when the sub-engine is active.

Testing should be sufficient to demonstrate that the scheduler proxy library can start multiple engine processes on the desktop host, and can also start multiple engine processes on other hosts.

Testing GetUseRsmForEngineLaunch

In most cases, this function will not be implemented or tested. If this function is implemented and returns TRUE, the Ansys Electronics Desktop will not start the analysis engines using the LaunchProcess function directly. Instead, the Ansys Electromagnetics Desktop starts one AnsoftRSMService process on each host using LaunchProcess, and the engine processes are started by these AnsoftRSMService processes. You can check for these processes using the Windows Task Manager. One AnsoftRSMService process should run on each host. These processes are named **ansoftrsmservice.exe** or **AnsoftRSMService.exe**. These processes start on each host before any analysis engine starts on the host, and will remain running until the job is complete.

Troubleshooting Custom Scheduler Integration

- None of the Proxy Functions are Called
- Troubleshooting IsProductLaunchedInYourEnvironment Function
- Troubleshooting GetSchedulerDisplayName
- Troubleshooting GetThisJobID
- Troubleshooting GetTempDirectory
- Troubleshooting GetMachineListAvailableForDistribution
- Troubleshooting LaunchProcess
- Troubleshooting GetUseRsmForEngineLaunch

None of the Proxy Functions are Called

There are several problems which could result in none of the proxy functions being called.

The scheduler proxy library must be installed in the scheduler's subdirectory of the Ansys Electromagnetics installation directory. The installation directory is set in the registry entry **HKEY_CURRENT_USER/Software/Ansoft/***Product*/*Version*/**Desktop/InstallationDirectory**,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

where *Product* is the Ansys Electromagnetics product name (for example, Twin Builder) and *Version* is the Ansys Electromagnetics product version (for example, 14.0).

The scheduler proxy library name must match ***_scheduler.dll** on Windows. If the library name does not match this format, then the library will not be loaded. In addition, the function name prefix must be the same as the library name prefix converted to upper case. For example, if the library name prefix is **abc**, the function name prefix is **ABC**. In this example, the library name is **abc_scheduler.dll** on Windows. In this example, the full name of the IsProductLaunchedInYourEnvironment function is ABC_IsProductLaunchedInYourEnvironment on Windows, and it must have extern "C" linkage.

Verify that the compile and link flags follow the guidelines in ***Build Information for Scheduler Proxy Library***. Incorrect compile or link flags can prevent the library from being loaded by the Ansys Electromagnetics product.

If there is a problem with calling the IsProductLaunchedInYourEnvironment function, then none of the other functions will be called. The other functions are only called if the IsProductLaunchedInYourEnvironment function is called successfully and returns TRUE.

Troubleshooting IsProductLaunchedInYourEnvironment Function

Verify that the conditions specified in ***None of the Proxy Functions are Called*** are met.

Verify that this function returns TRUE when called in an environment running under the scheduler, and that it returns FALSE when called in an environment not running under the scheduler.

Troubleshooting GetSchedulerDisplayName

Verify that the **IsProductLaunchedInYourEnvironment** function returns TRUE when running in the scheduler environment.

Verify that the scheduler display name is a valid ASCII string.

Verify that, if argument buffer is NULL, the required length of the buffer is stored in the location to which argument length points, and returns TRUE. The required buffer length must include space for the string null terminator.

Verify that, if argument buffer is not NULL and the value to which argument length points (the buffer length) is large enough to contain the display name, including the terminal null byte, then the display name is copied to the buffer and returns TRUE.

Troubleshooting GetThisJobID

Verify that the **IsProductLaunchedInYourEnvironment** function returns TRUE when running in the scheduler environment.

Verify that the job ID is a valid ASCII string.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Verify that, if argument buffer is NULL, the required length of the buffer is stored in the location to which argument **length** points, and TRUE returns. The required buffer length must include space for the string null terminator.

Verify that, if argument buffer is not NULL and the value to which argument length points (the buffer length) is large enough to contain the job ID, including the terminal null byte, then the job ID is copied to the buffer and TRUE returns.

Troubleshooting GetTempDirectory

Verify that the **IsProductLaunchedInYourEnvironment**function returns TRUE when running in the scheduler environment. Verify that the temporary directory name is a valid ASCII string.

Verify that, if argument buffer is NULL, the required length of the buffer is stored in the location to which argument **length** points, and TRUE returns. The required buffer length must include space for the string null terminator.

Verify that, if argument buffer is not NULL and the value to which argument **length** points (the buffer length) is large enough to contain the temporary directory path name, including the terminal null byte, then the temporary directory path name is copied to the buffer and TRUE returns.

Troubleshooting GetMachineListAvailableForDistribution

Verify that the **IsProductLaunchedInYourEnvironment** function returns TRUE when running in the scheduler environment.

Verify that the list of hosts is a valid ASCII string containing a space separated list of host names. A host name will appear in the list a number of times equal to the number of processors or cores available to the job on that host.

Verify that, if argument **buffer** is NULL, then the required length of the buffer is stored in the location to which argument **length** points, and TRUE returns. The required buffer length must include space for the string null terminator.

Verify that, if argument **buffer** is not NULL and the value to which argument **length** points (the buffer length) is large enough to contain the list of hosts, including the terminal null byte, then the list of hosts is copied to the buffer and TRUE returns.

Troubleshooting LaunchProcess

Verify that the **IsProductLaunchedInYourEnvironment** function returns TRUE when running in the scheduler environment.

The **hostName** argument is a host name from the list returned by the **GetMachineListAvailableForDistribution** function. Verify that the **LaunchProcess** function

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

can accept host names in the format returned by the **GetMachineListAvailableForDistribution** function.

The **exePathName** argument is the path name of the analysis engine executable to start. This path name may contain spaces or other characters special to the shell. Ensure that the **LaunchProcess** function can handle such cases.

The **arg1** and **arg2** arguments may contain newlines, tabs, single quotes, spaces, dollar signs, and other characters which may be special to the shell. Ensure that the **LaunchProcess** function can handle such cases. If needed, the newline characters may be replaced by other whitespace characters. One or both of these arguments could also be an empty string; verify that the empty string is correctly passed to the engine process command line.

If you use a scheduler command to launch the engine process on a remote machine, the engine command line may be processed by the shell twice, once when the scheduler command is processed by the shell, and again when the engine command is processed by the shell. In such cases, the quoting of characters special to the shell will need to be take these two passes through the shell into account. In some implementations, it may be necessary or convenient to use different approaches for launching engine processes on the local machine and on remote machines; if this is done, verify that the approach used to determine whether the **hostName** argument represents the local machine is correct.

Troubleshooting GetUseRsmForEngineLaunch

In most cases, this function will not be implemented. If it is implemented, then follow the suggestions below.

Verify that the **IsProductLaunchedInYourEnvironment** function returns TRUE when running in the scheduler environment.

If the RSM should be used for launching engines, verify that this function returns TRUE. Otherwise, verify that it returns FALSE.

RSM Integration with Job Management UI

Ansys Electromagnetics supports its own Remote Simulation Management (RSM) software, along with other High Performance Computing (HPC) software management programs (see ***High Performance Computing (HPC) Integration***.).

When do you need RSM?

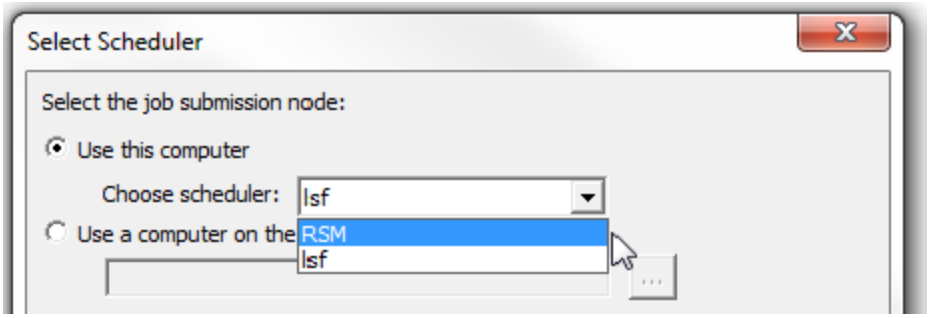
RSM is required to run remote or distributed simulations. However, if you have a separate scheduling system that Ansys Electromagnetics supports, and you plan to run batchsolve simulations only, then you may not need to install RSM. For details of installation and configuration of RSM, see the **Ansys Electromagnetics Installation Guides**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Job Management UI for RSM

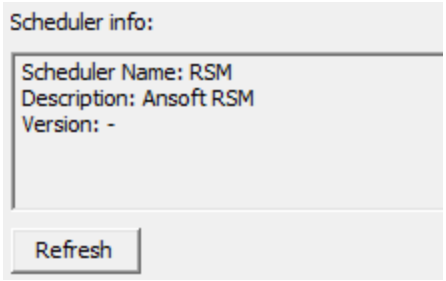
You can use the Job Management UI to submit batch jobs to RSM. Access the Job Management UI by running Ansys Electromagnetics Desktop on the designated 'Postprocessing node' of the cluster. The Desktop provides UI commands for Scheduler selection, Job submission, and Job monitoring/control. Select **Tools** > **Job Management** > **Select Scheduler** to access the Scheduler User Interface.

Click **Select Scheduler** to display the selection dialog box. A drop-down menu lists potential schedulers, (which can include RSM, lsf, Windows HPC, or others, depending on the environment).



If you select a scheduler that is not supported in your environment, you receive a warning message.

After selecting a scheduler, click **Refresh** to display information for that scheduler.

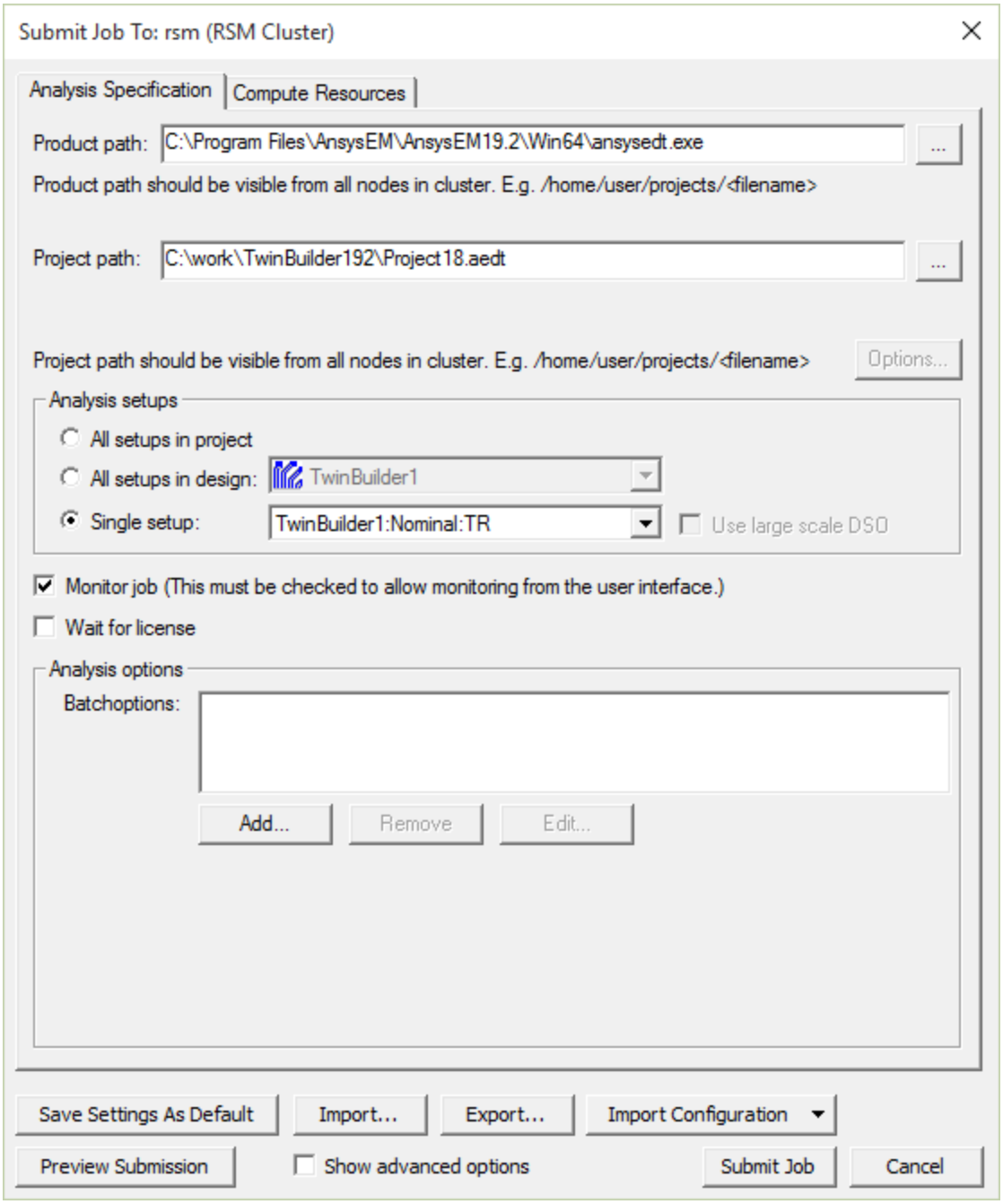


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Once you have selected a scheduler supported in your environment, follow these steps to submit a batch job.

1. Set up and prepare model on local workstation.
2. Copy the input project (or folder, if the project references external files) from a personal workstation to a shared drive on cluster (for example, project is copied to /home/projects/spool/test.aedt).
   - In the RSM environment, you are required to specify a machine configuration list. (See ***Setting HPC and Analysis Options*** and ***Editing Distributed Machine Configurations***) For example, if the machine list is: 3 cores from 'm1' and 3 cores from 'm2', for a total of 6 engines, select the list on the **Compute Resources** tab of the **Submit Job to: RSM** dialog box, as described below.
3. Open a remote desktop session (or equivalent such as vnc session) on the node corresponding to the first machine of job's machine list, 'm1' in this case. Launch Desktop graphically on 'm1'.
4. Run **Tools** > **Job Management** > **Submit Job** to open the standard **Job Submission** dialog box, shown below.

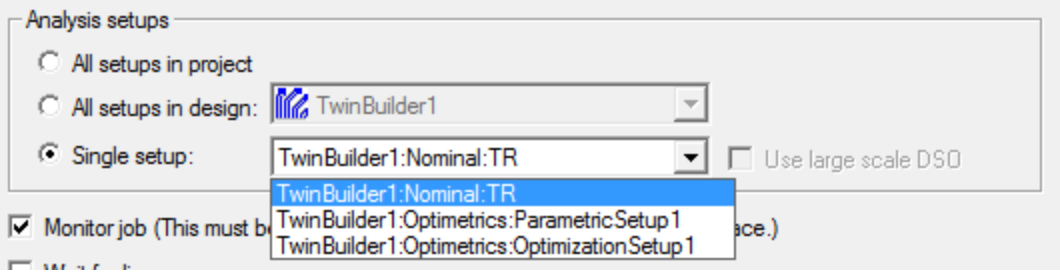
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- The **Analysis Specification** tab has parameters to specify the input project model, the analysis setup and analysis options (including batch options) that affect analysis algorithms.
- The **Compute Resources** tab specifies the amount of compute resources and how to select specific resources from the available pool.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Click ... to browse for a project.
6. In the **Analysis setups** area, you can select option buttons for **All setups in the project**, **All setups in the design**, or a **Single setup**. For instance, the following example includes setups for Nominal, Parametric, and Optimization.

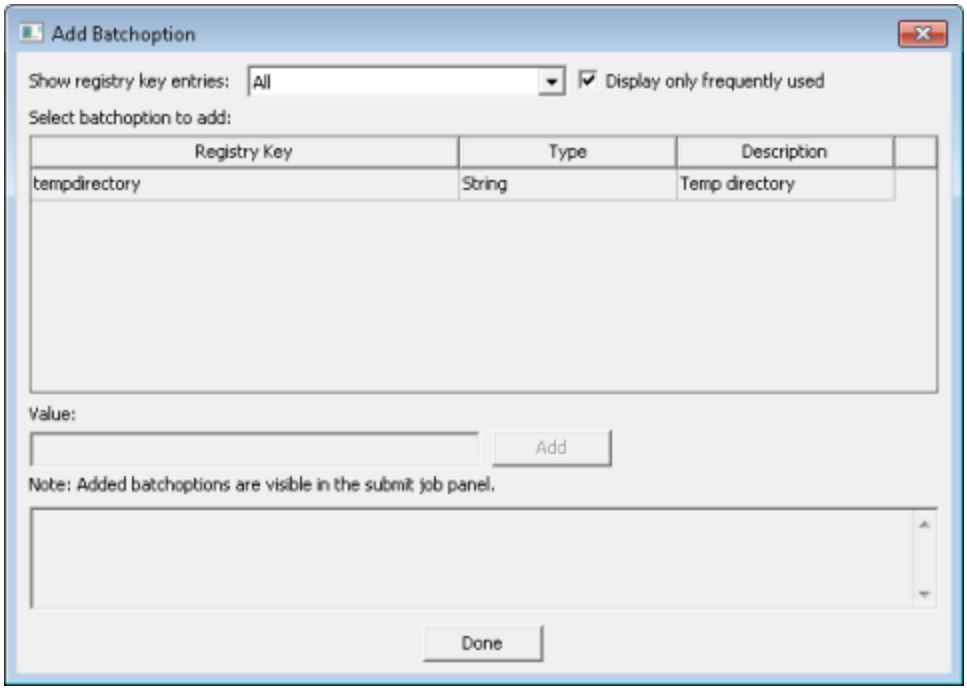


For Parametric setups, you can select **Use Large Scale DSO**. For details on how and when to use this feature, see ***Job Management Interface for Large Scale DSO***.

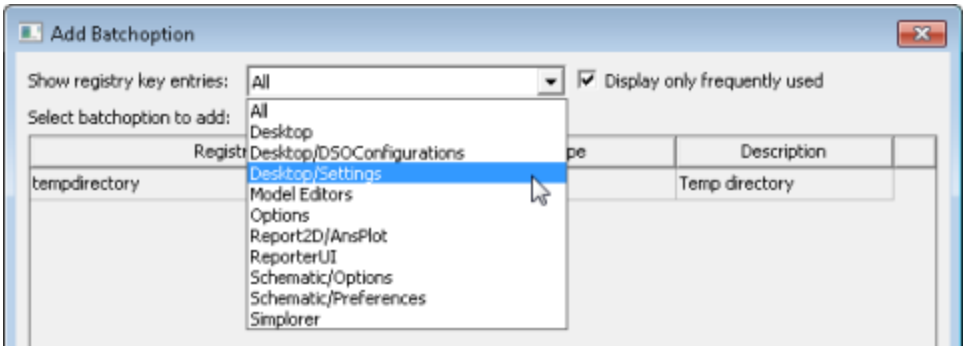
7. The Analysis options includes check boxes for monitoring the job, whether to wait for a license, and a field for adding Batchoptions via a graphical interface, or as text.
   - If you intend to monitor the job through a user interface, select **Monitor job**. You can then monitor this job with **Tools** > **Job Management** > **Monitor Jobs...**, or by selecting the dialog box that opens when you submit the job.
   - Use the **Batchoptions** field to add additional **-batchoptions** parameters, either as text, or by using a dialog box with selection menus. Click **Add** to view the **Add**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Batchoption dialog box.



Use the **Show registry key entries** field to filter the entries displayed, by means of a drop-down menu selection, and a check box to **Display only frequently used entries**.

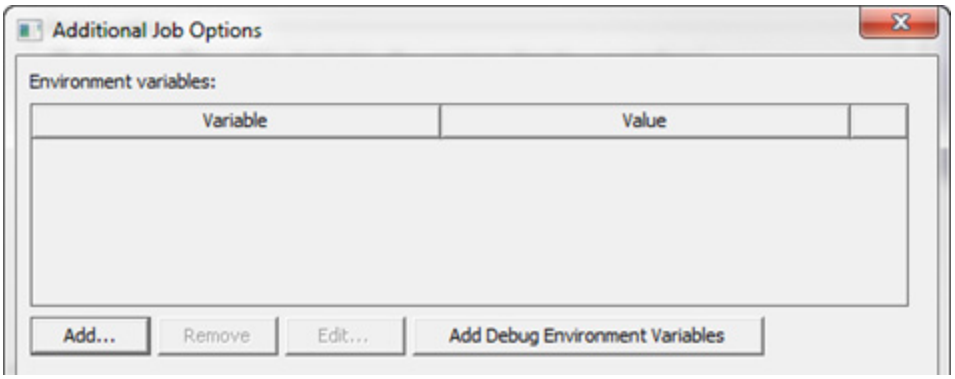


- When you have selected a batch option, type the value in the field and click **Add** to add the option to the batch command.
- In the **Submit Job To:** dialog box, select **Show advanced options** to display the

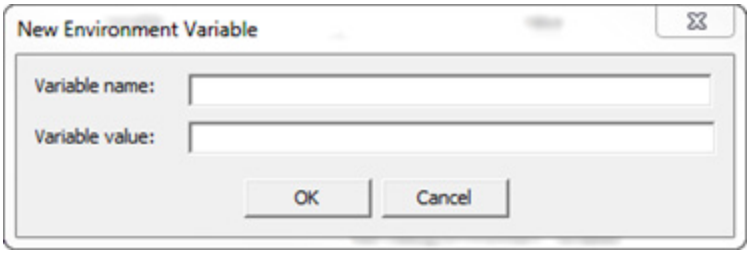
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Environment field.

This field is for environment variables, for instance, for debugging features or other variable controlled features. Click ... to open the **Additional Job Options** dialog box.

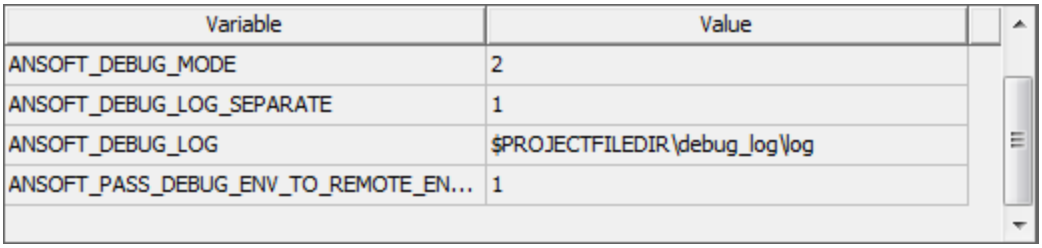


Click **Add...** to open a **New Environment Variable** dialog box in which you can include a variable name and value.



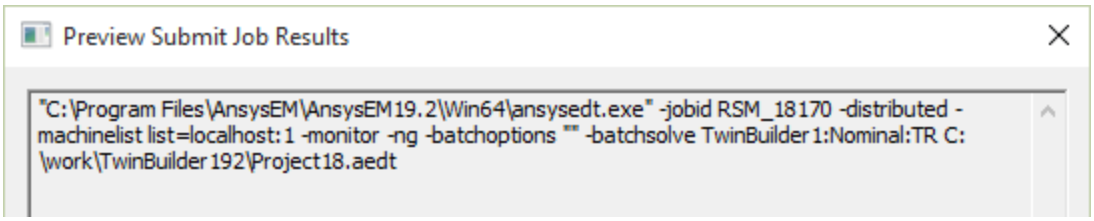
Click **Add Debug Environment Variables** to add a set of debug variables. This can be useful in working with Ansys Application Engineering support.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Select a variable to enable the **Remove** and **Edit** buttons. Click **Edit** to open a dialog box where you can change the variable and value.

8. To see the command line to be submitted to the scheduler, click **Preview Submission**. This opens a dialog box showing the command to be sent to the scheduler.

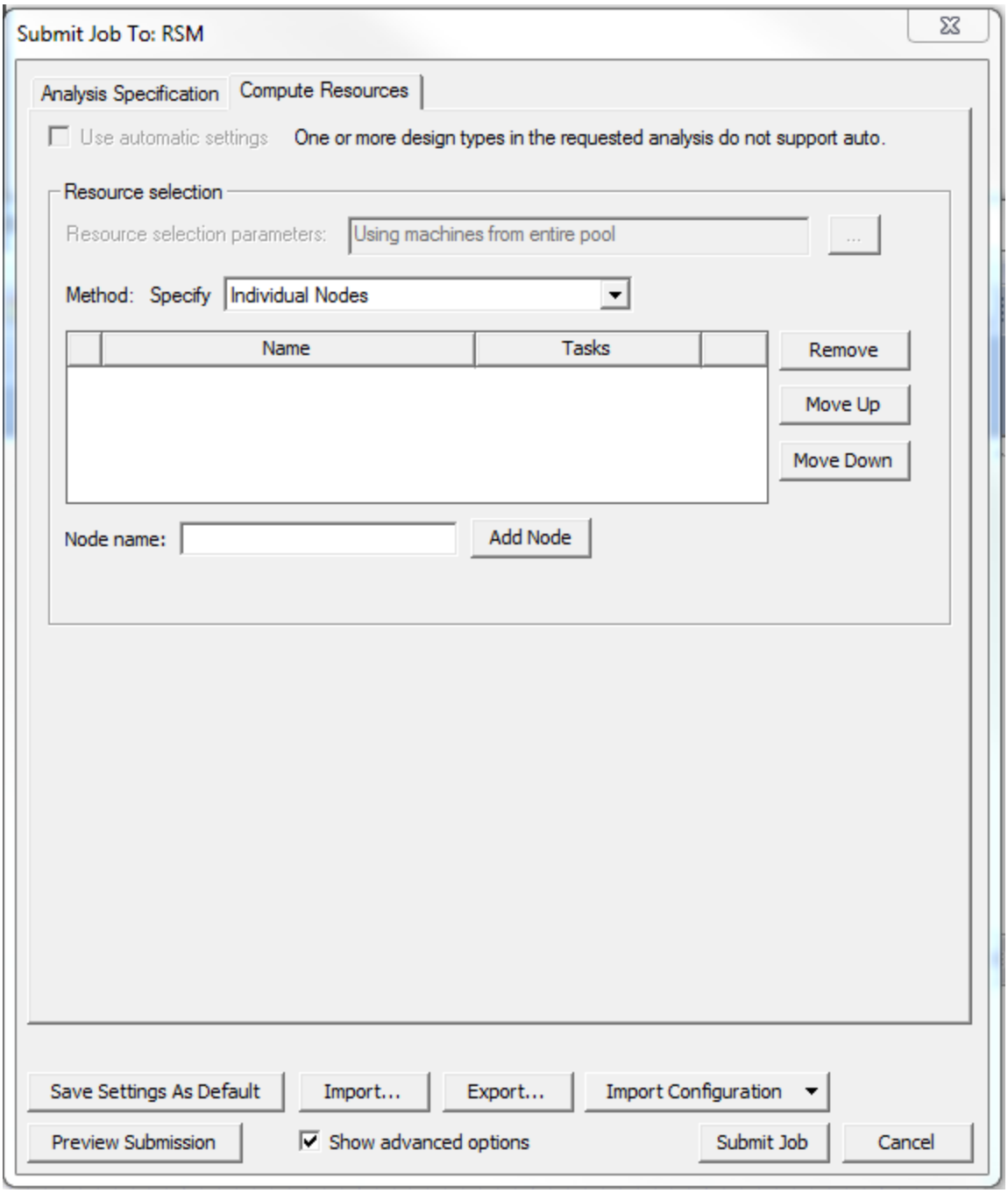


The text can be copied to the clipboard.

9. The **Compute Resources** tab of the **Submit Job To: RSM** dialog box displays other parameters. Depending on the resources available for a scheduler environment, some of

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the fields may be disabled.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

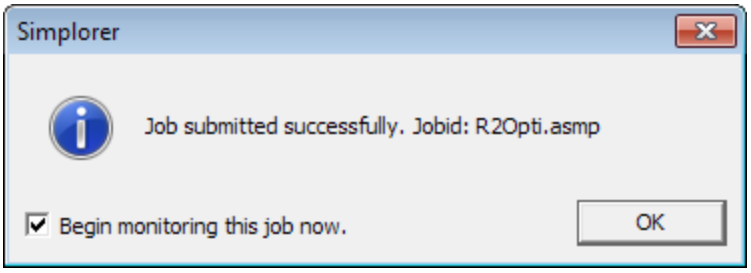
> **Note:**
>
> If you select **Use automatic settings,** the solver does not support automatic distribution of variations. This option does not support Optimetrics variations. It does distribute frequencies, domains, and use of multiple level domains.

You can either select **Use automatic settings** on the **Compute Resources** tab, or you can enter individual nodes. For each node, enter the node name and add the node. Click **Remove**, **Move Up**, and **Move Down** to edit and order the list.

10. To submit the command with the specified parameters, click **Submit Job**.

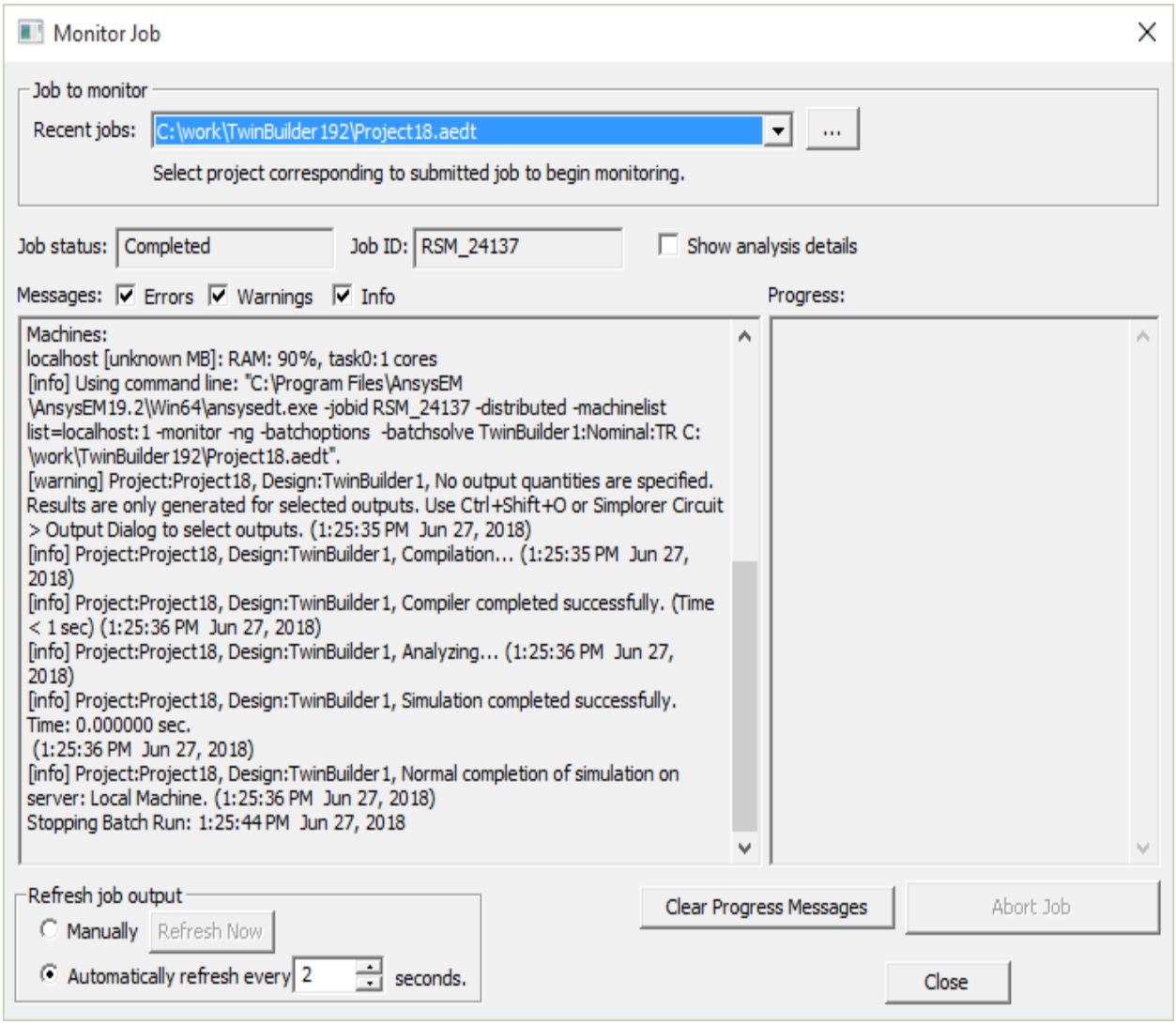
> **Note:**
>
> The RSM environment does not support queuing, so 'Submit Job' will immediately start running the job.

A dialog box appears in which you can select **Begin monitoring this job now**.



11. You can monitor this job either automatically (by checking the option) or with **Tools** > **Job Management** > **Monitor Jobs**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The dialog box contains fields reporting the job status, job ID, messages issues, and progress. You can filter the messages for Errors, Warnings, and Info. By option you refresh the job manually or at specified intervals.

Re-solving a Problem

When you modify a design after generating a solution, the solution in memory will no longer match the design.

Follow this procedure to generate a new solution after modifying a design.

1. Run a simulation.
2. Select a solution setup in the Project tree.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Click **Project** > **Analyze All**.

Resetting Symbol Animation

Select **Twin Builder** > **More Actions** > **Reset Symbol Animation** to reset output quantity and signal values to their defaults (they may have been set to other values during simulation) and re-evaluates the symbol animation conditions to determine the current graphics levels to show.

Related Topics

Design Settings

Animate

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

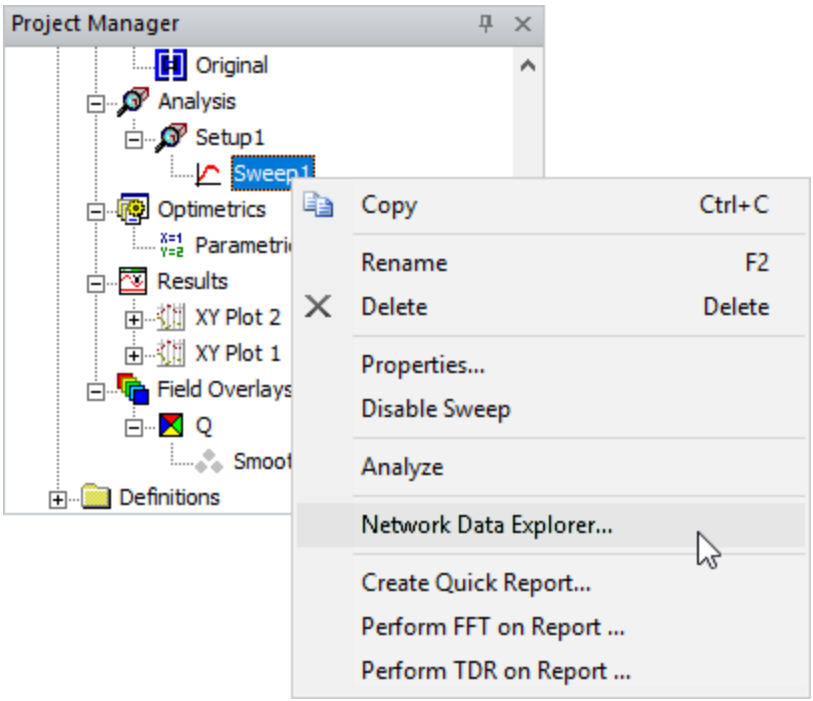
18 - Network Data Explorer

The **Network Data Explorer** provides visualization, analysis, and manipulation tools for network data.

To access **Network Data Explorer**, select **Tools** > **Network Data Explorer**.

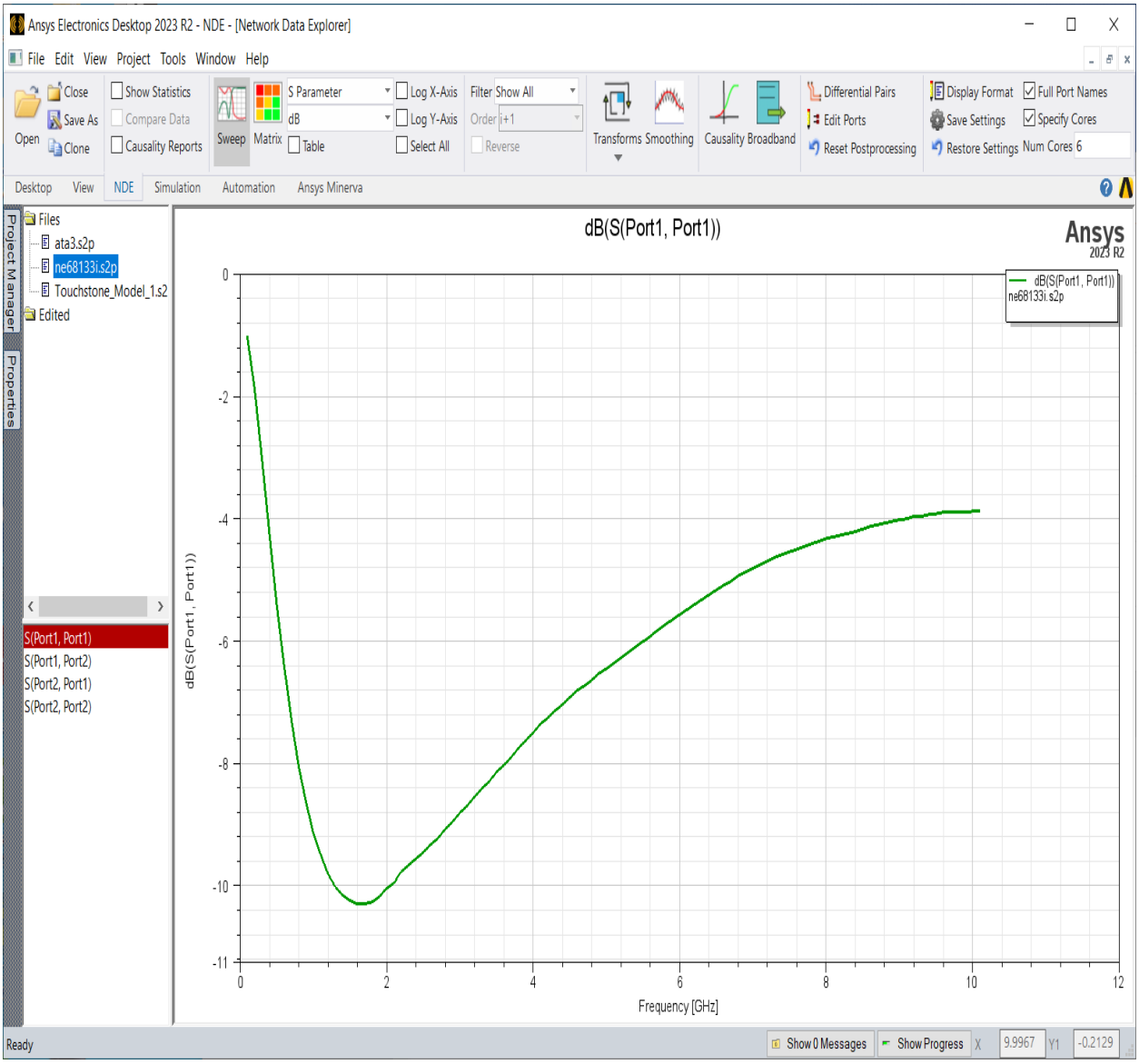
If you added a Frequency Sweep, right-click the sweep and select **Network Data Explorer**.

> **Note:**
>
> Accessing **Network Data Explorer** in this way loads sweep data.



The **Network Data Explorer** window appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The topics for this section include:

Network Data Explorer Overview

Loading Data Into Network Data Explorer

Exporting Data from Network Data Explorer

Right-Click Context Menu Options

Viewing Data and Modifying the Display

Creating a Statistics Plot

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Comparing Network Data

Comparing Variations

Displaying Plot Traces from Multiple Data Sources

Displaying Mixed-Mode Parameters

Changing Port Properties and Reducing Matrix Size

Smoothing

Causality Checking and Plots

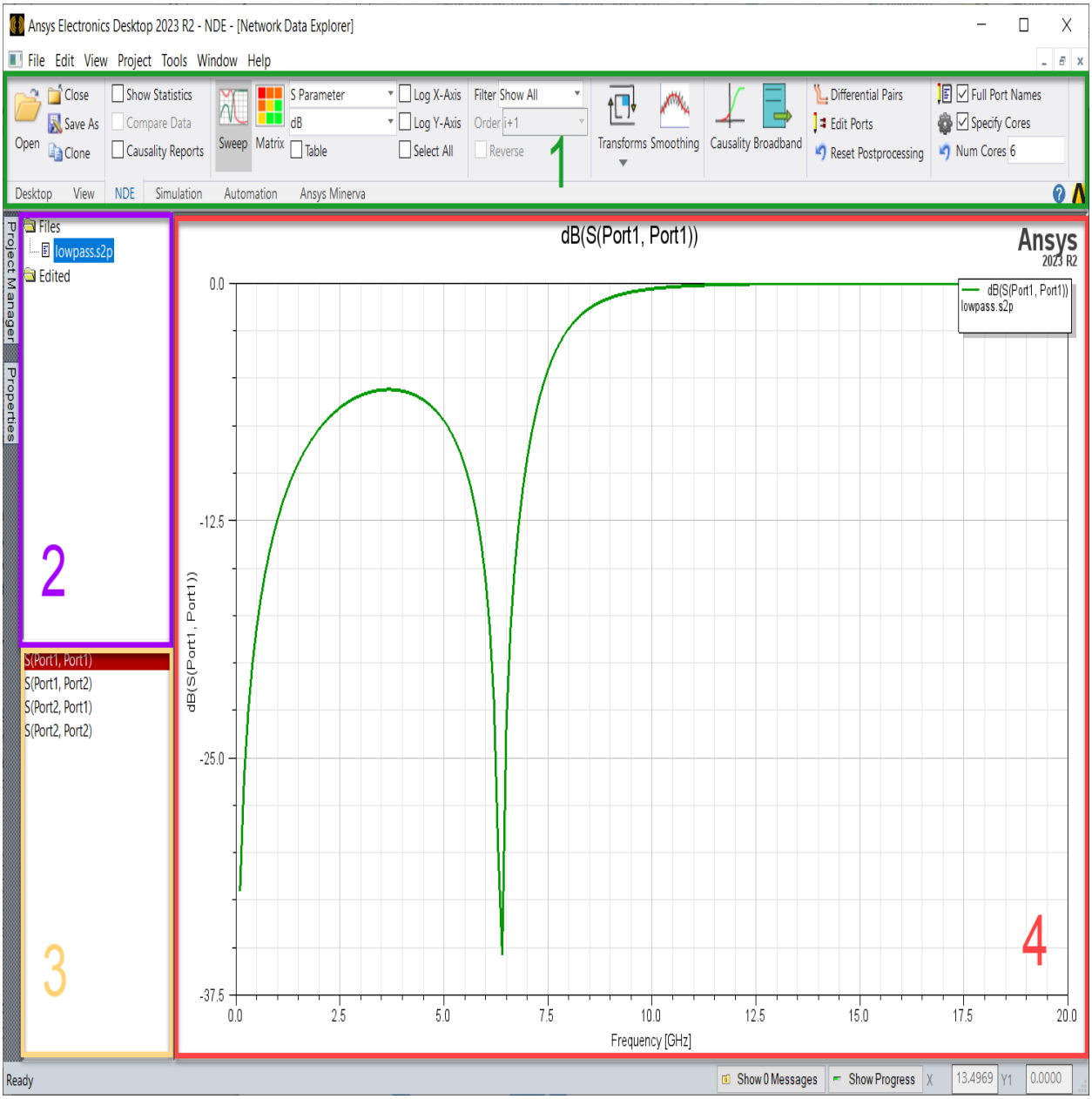
Network Data Explorer Overview

The **Network Data Explorer** window is divided into the following panes:

1. **NDE ribbon** – Perform many functions of the Network Data Explorer.
2. **Network Data Selection pane** – Select a network data file.
3. **Cell and Frequency Selection pane** – Narrow your selection.
4. **Data View pane** – Display data in table or plot format.

The panes are shown in the following figure. Additional information about each pane follows.

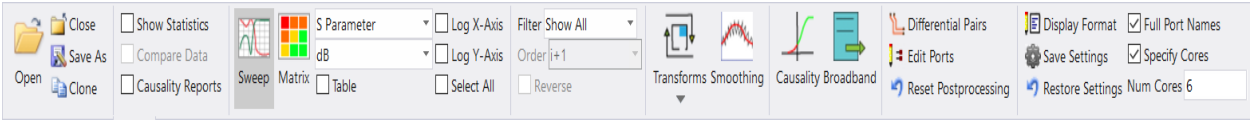
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



NDE Ribbon

The ribbon provides access to many of the Network Data Explorer's functions and display options.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

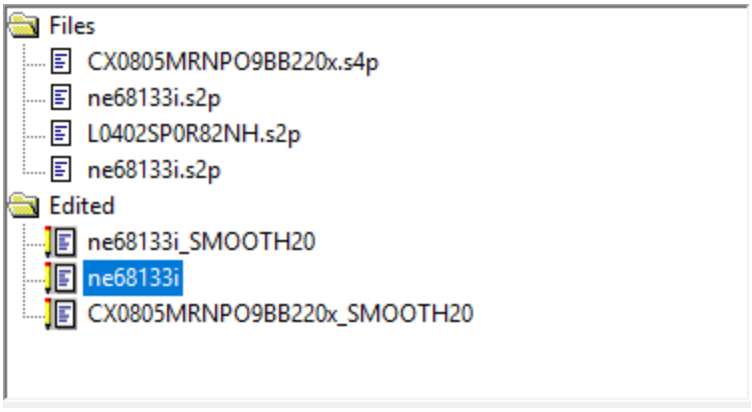


On this ribbon, you can control:

- **Plotting** – Determine how the data displays.
- **Quantity** – Select the type of quantity to display (parameter values, matrix statistics, or causality plots).
- **Parameter Type** – Choose the parameter for display (S, Y, or Z parameters, Port Impedance, or Gamma).
- **Format** – Decide the display function to apply to the data (for example, magnitude, phase, dB, real, imaginary).
- **Export** – Export either SYZ data (***.s1p**, ***.ts**, ***.nmf**, ***.tab**, ***.m**, ***.cit**) or Broadband data (***.sp**).
- **Check** – Check causality.
- **Cores** – Enable or disable multithreading.
- **Post-Process Selection** – Choose between Terminal Data and Differential Pairs, if your design includes differential pairs.

Network Data Selection Pane

Use this pane to view and compare various data sets. Original data sets appear under **Files**. Click a data set to see it as it was when it was opened. Altered data sets appear under **Edited**. These data sets appear here when they have been smoothed, transformed, or changed in some way.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Cell and Frequency Selection Pane

Use this pane to choose which frequencies or cells to display. Click **Sweep** or **Matrix** to choose a type of plot. Select the **Select All** check box on the NDE ribbon to select all frequencies or cell entries.

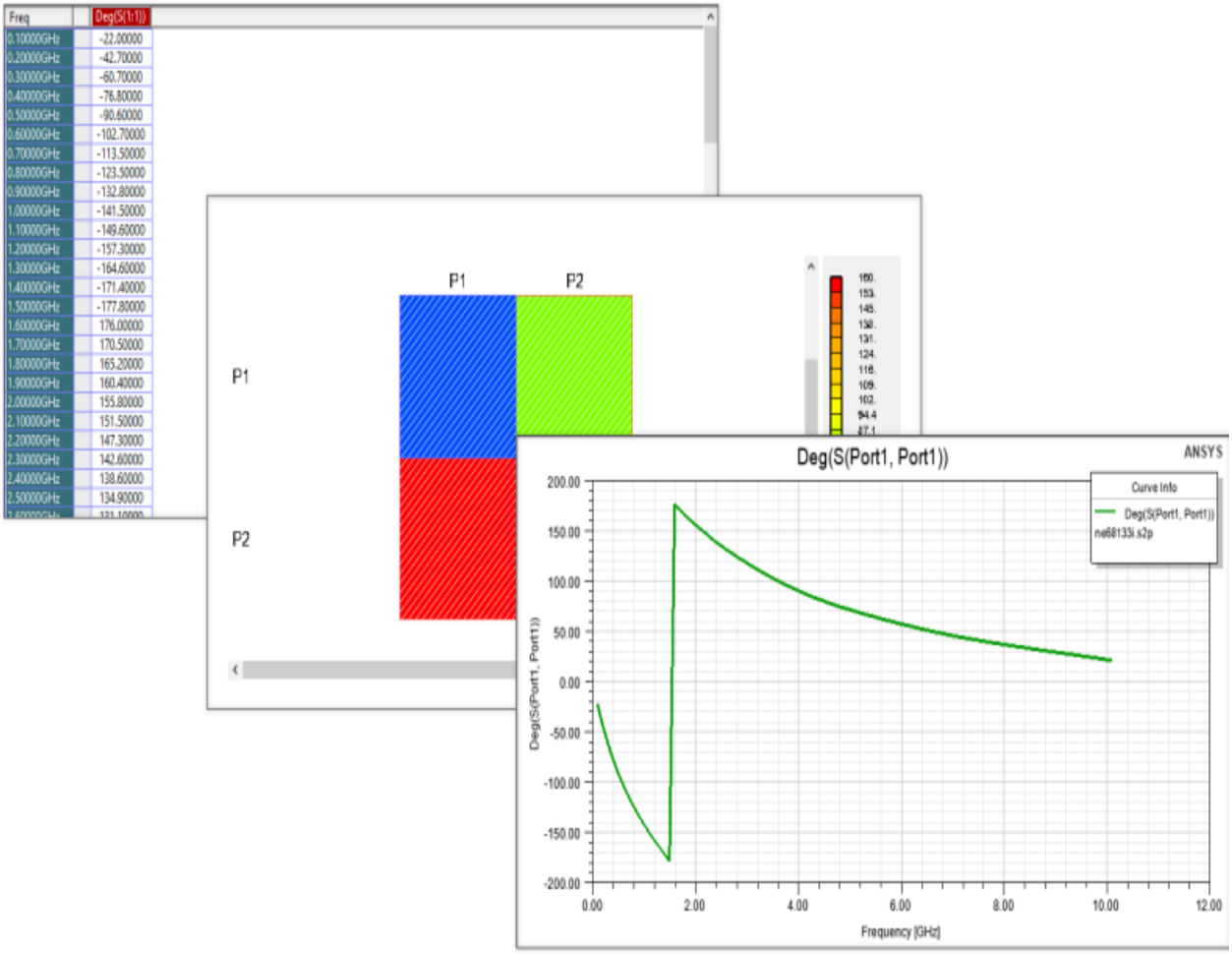
This pane can also list available variations. Selecting a variation activates it and affects table, plot, and statistics displays. When displayed by frequency, the entire matrix is presented in the **Data View** pane for each selected frequency. When displayed by matrix cell, the data for the individually chosen cells is shown across all frequencies. Select the **Select All** check box to select all variations.

> **Warning:**
>
> Given the volume of data in many network data sets, choosing **Select All** may take a considerable time to generate, especially in Plot view.

Data View Pane

This pane displays data for **Sweep** or **Matrix** in either plots or a table, depending on your selection.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Loading Data Into Network Data Explorer

If right-clicked a sweep to launch **Network Data Explorer**, the current solution data is loaded and ready for viewing. Otherwise you must load a data file.

You can import these file types:

- Touchstone Format (***.s*p**)
- Touchstone 2 Format (***.ts**)
- Citifile (***.cit**)
- Neutral Format (***.nmf**)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- State Space File (***.sss**)

> **Note:**
>
> When this type of file is loaded, **Network Data Explorer** regenerates s-parameter data based on the file.
>
> You can compare the regenerated s-parameters to the original data.

To import a file into Network Data Explorer, either drag and drop an analysis from the **Project Manager** into Network Data Explorer or

1. On the **NDE** ribbon, click **Open**. The **Open** dialog box appears.
2. Navigate to and select a file.
3. Click **Open**. The file appears in the **Files** tree.

You can open multiple files at once with the file browser. However, the displayed data always corresponds to the data set indicated in the Network Data Selection pane. Click the file you want in that pane to switch between data sets. Use Shift and Ctrl to select and display data sets from multiple files simultaneously.

Exporting Data from Network Data Explorer

Use **Network Data Explorer** to export data to a file in several formats.

You can also create a Twin Builder component during export, which exports the active data set back to the Twin Builder project.

Topics for this section include:

Creating a Twin Builder Component

Scripting Network Data Explorer

Export SYZ Data

Export Macro Model

Exporting SYZ Data

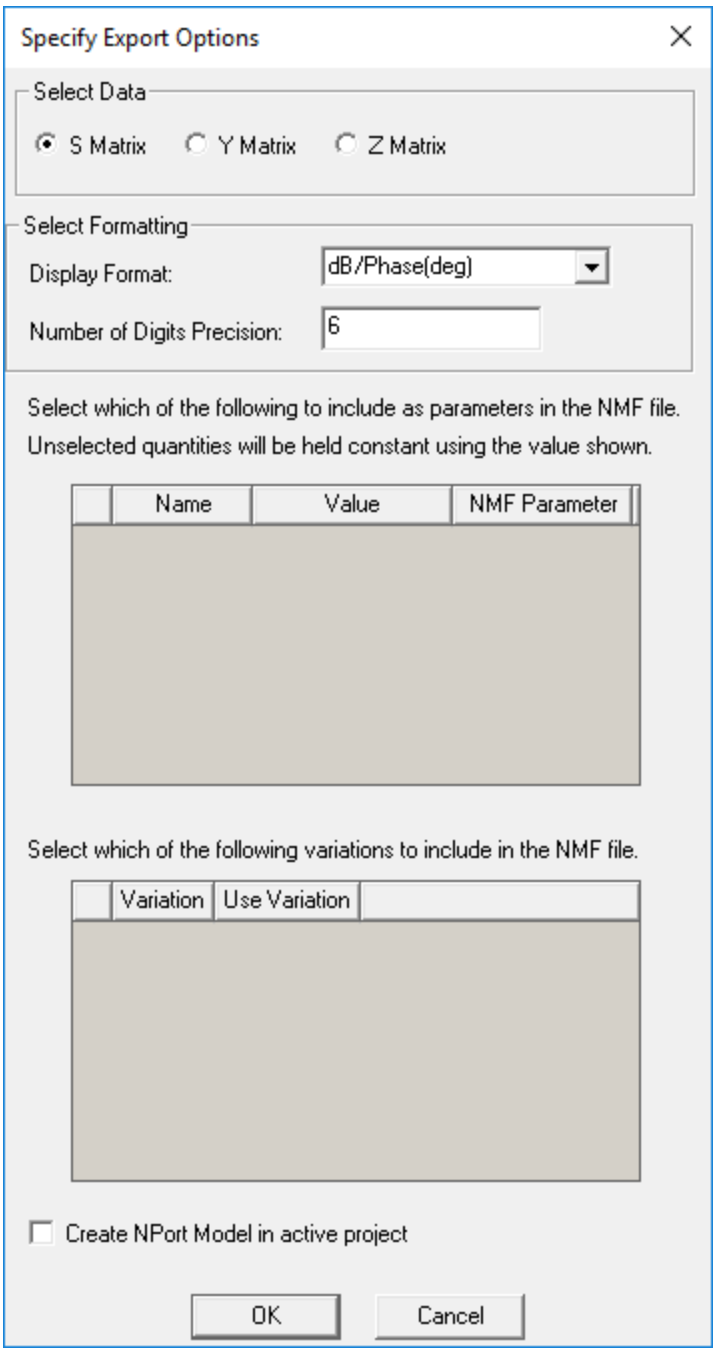
To export SYZ data from within Network Data Explorer, click the **Save In SYZ Format** icon on the **NDE** ribbon. The **Save File As** dialog boxappears. You can export data in any of six file types:

- Touchstone Format 1.0 (***.s*p**)
- Touchstone 2 Format (***.ts**)
- Neutral Format (***.nmf**)

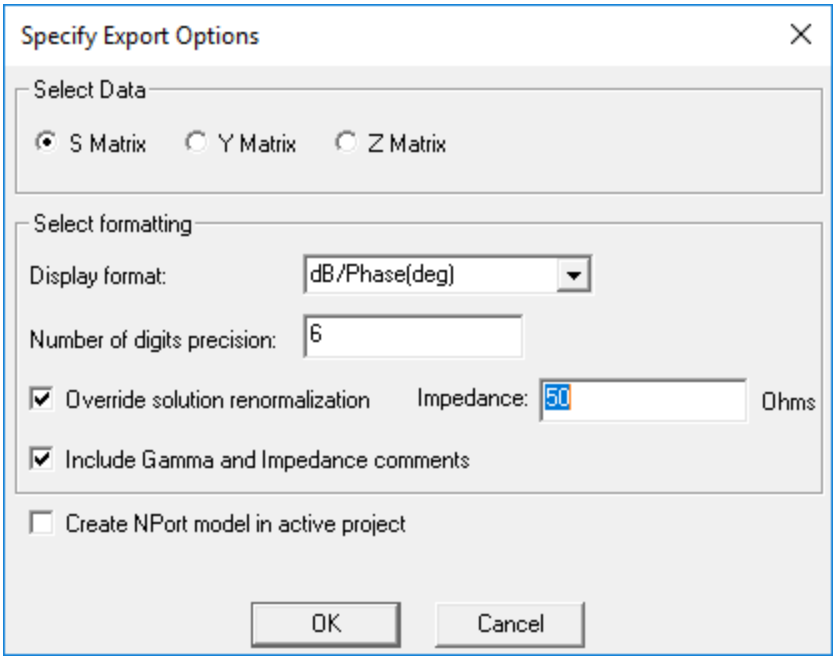
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Data Table Spreadsheet (***.tab**)
- MATLAB (***.m**)
- Citifile (***.cit**)

Select a file type and name for export. The **Specify Export Options** dialog box appears.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Depending on the type of export file, different options appear. You can always:

- Select from **S Matrix**, **Y Matrix**, and **Z Matrix** data.
- Select the **Display Format**.

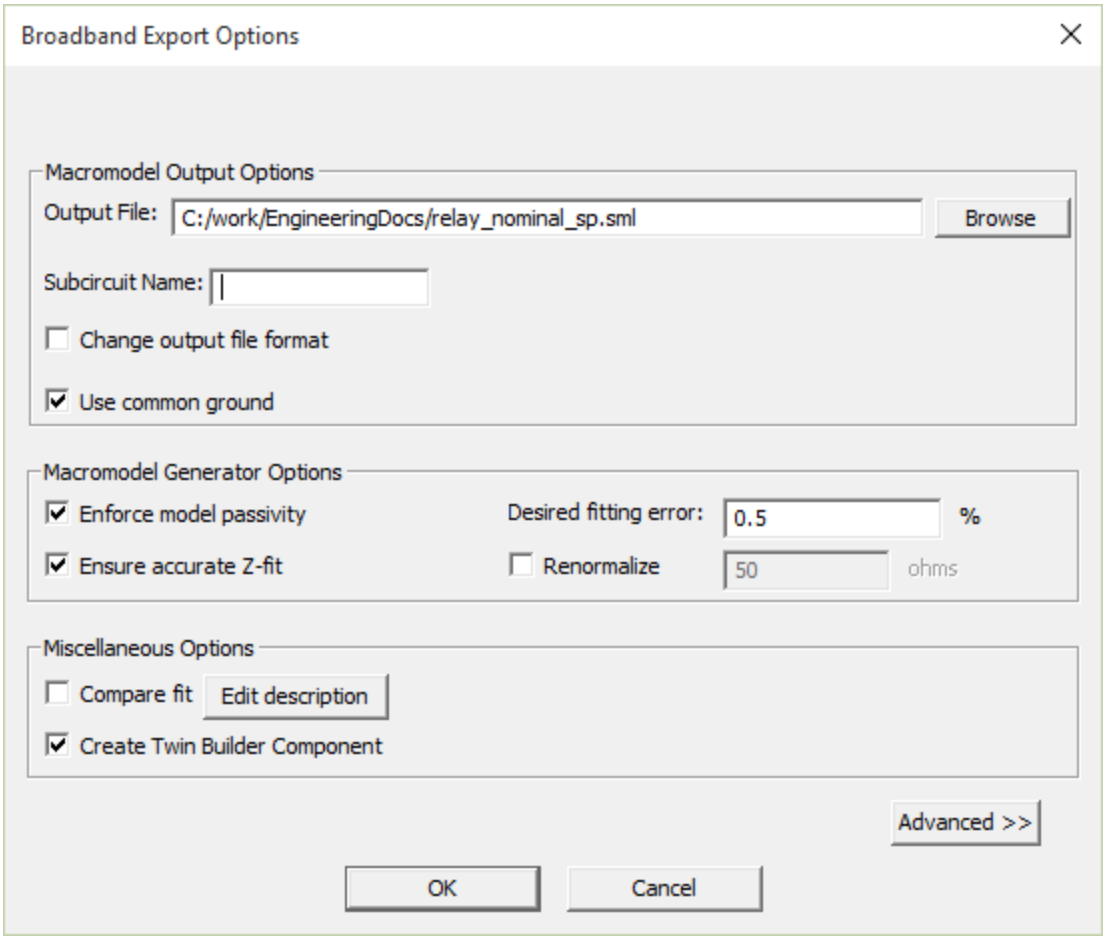
With some file types, you can create a Twin Builder component in the active project.

Exporting Macro Model

Network Data Explorer lets you export macro model data. To export data, click the **Broadband** icon on the **NDE** ribbon.

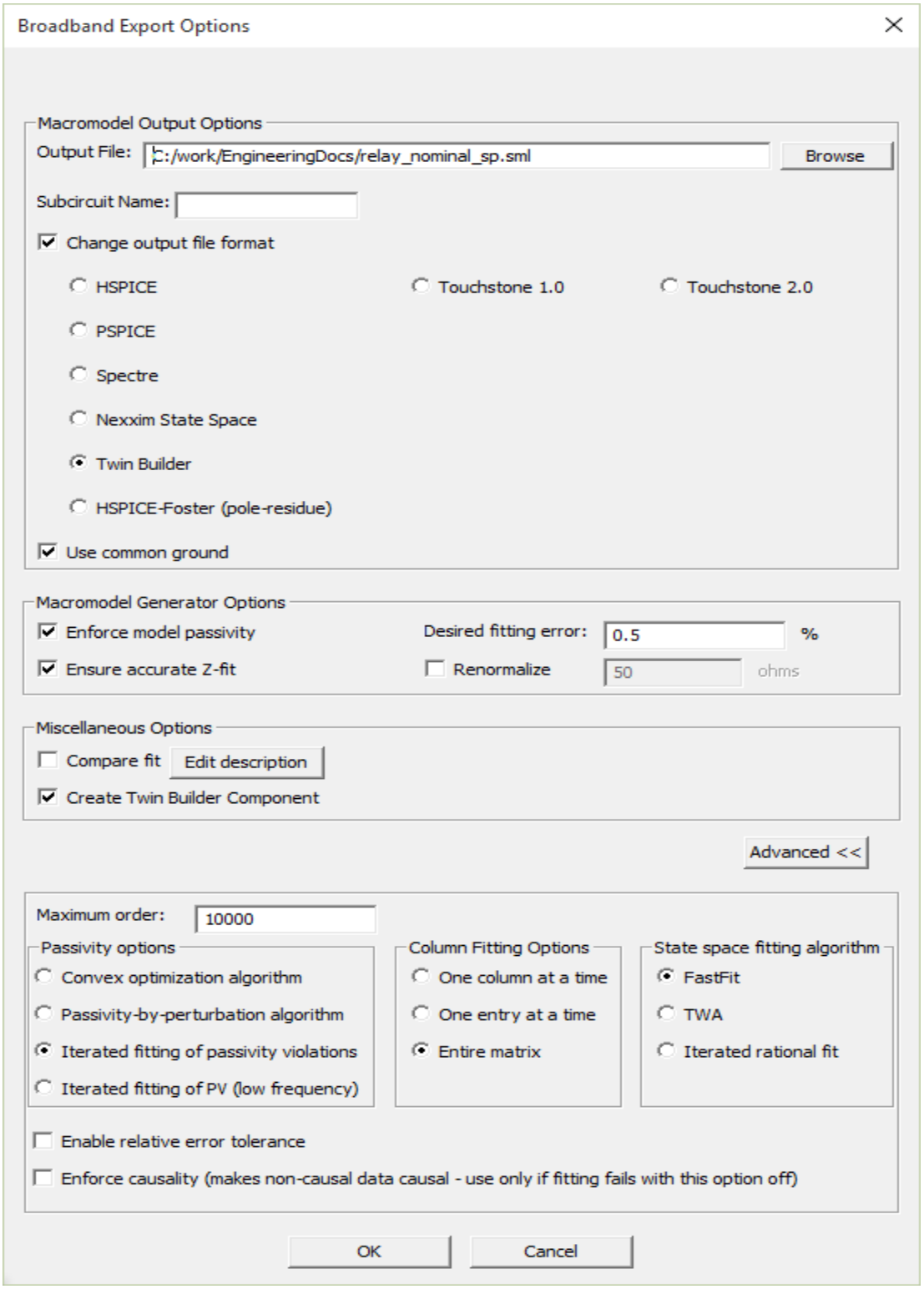
The **Broadband Export Options** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Click **Advanced >>** to view all options.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

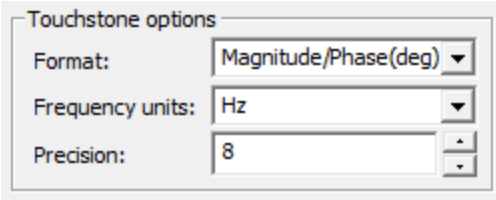


Macromodel Output Options

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Output File** – Choose the name and location of the file.
- **Subcircuit Name** – Name the subcircuit.
- **Change Output File Format** – Select this box to choose an output format from the list.

  When **Touchstone 1.0** or **Touchstone 2.0** is selected as the format, the following **Touchstone options** appear:



- **Use common ground** – Select this box to use common ground. When this option is on, ports are referenced to ideal ground (node 0). When this option is off, extra ports generate to provide reference levels. Common grounding is best when the pins are physically near to each other and ideal ground is suitable. For distant connections and circuits with non-ideal reference levels such as differential pairs, common grounding is not used.

> **Note:**
>
> - R and L values may be quite sensitive to the values of the S-parameters. This is an issue if the actual impedance value is much greater than, or much less than, the reference impedance of the S-parameters.
> - Since resistances of power cables is typically in the milliohms range at DC, using a reference impedance = 50 ohms is 5000 times higher. This causes any fitting errors in the state space model to get multiplied by 5000 times when the R and L values are computed.
> - As a general rule, for high power applications a reference impedance of 1 ohm is probably a better choice than 50 ohms.

Macromodel Generator Options include:

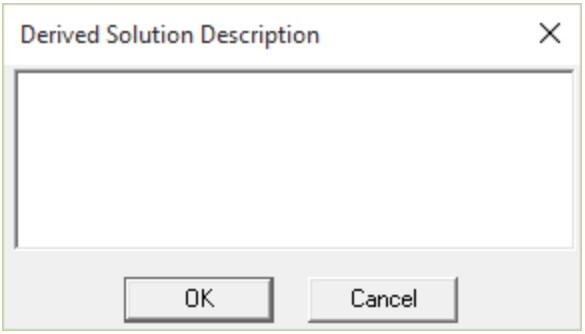
- **Enforce Model Passivity** – Select this box to enforce passivity. A uniform renormalization of 50 ohms is performed on the solution data for passivity checking.
- **Ensure Accurate Z-fit** – Select this box when state-space fitting of Y-parameters or Z-parameters does not produce an accurate fit.
- **Desired Fitting Error** – Select the value at which rational fitting fails (if the fitting error exceeds this value).
- **Renormalize** – Select this box to renormalize using the impedance specified in the adjacent text box. 50 ohms is the default value, but you can enter a different value.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> R and L values may be quite sensitive to the values of the S-parameters. This is an issue if the actual impedance value is much greater than (>>) or much less than (<<) the reference impedance of the S-parameters.
>
> Since resistances of power cables is typically in the milliohms range at DC, using a reference impedance = 50 ohms is 5000 times higher. This causes any fitting errors in the state space model to get multiplied by 5000 times when the R and L values are computed.
>
> As a general rule, for high power applications a reference impedance of 1 ohm is probably a better choice than 50 ohms.

Miscellaneous Options

- **Compare Fit** – When this box is selected, the original and derived solution will be available for comparison. Click **Edit Description** to open the **Derived Solution Description** dialog box and add a text description to better identify the export.



- **Create Twin Builder Component** – Creates a Twin Builder component. This option is available only for Twin Builder designs.

Advanced Options

The following advanced options are available:

- **Maximum order** – Number of poles. The Broadband models are built from a rational-function approximation to the data. The fidelity of this approximation can be controlled by setting the **Desired fitting error** and the **Maximum order** (number of poles).
- **Passivity Options** – If you enabled **Enforce Model Passivity**, this area lets you select the passivity enforcement method.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Convex optimization algorithm** – Guarantees a passive state-space realization, but is very slow and memory-intensive.
- **Passivity-by-perturbation algorithm** – Designed for systems with a large number of ports.
- **Iterated fitting of passivity violations (IFPV)** (default) – Less accurate than other algorithms but quickly estimates a fit with low memory usage.
- **Iterated fitting of PV (low frequency)** – Similar to IFPV while improving the fit to "Z" at DC and low frequencies.
- **Column Fitting Options**
- **One column at a time** – The set of poles is shared across all entries of a single column.
- **One entry at a time** – Each entry is fitted using a separate set of poles.
- **Entire Matrix** – The set of poles is shared across all entries of the matrix being fitted.

> **Note:**
>
> - Typically, using the same set for all entries is adequate, and yields the most compact models. However, if all the entries of the matrix have completely unrelated transfer functions, it may be better to fit them using separate pole sets.
> - The options **One column at a time** and **One entry at a time** do not work when either **Ensure accurate Z-fit** or **FastFit** is used.

- **State space fitting algorithm** – Select either **FastFit**, **TWA**, or **Iterated rational function**.
  - **FastFit** – FastFit is the Ansys-proprietary default method for state-space fitting. ndExplorer uses **FastFit** for calculating the state-space matrices from the network data. The FastFit algorithm for state-space fitting is an alternative to the Tsuk-White algorithm (TWA) and Iterated Rational Fitting (IRF) methods. FastFit is generally as accurate as TWA, but is significantly faster than both TWA and IRF. It also aims to fit the lower frequencies with higher fidelity.
  - **TWA** (Tsuk-White Algorithm) – An Ansys-proprietary method for fitting a state space model to extracted s-parameter data. It uses techniques based on Singular Value Decomposition (SVD) to quickly determine required number of poles for fitting a model.
  - **Iterated rational function** – This fitting approach takes a matrix of S-parameter data and, for each matrix entry, tries a succession of different pole-zero approximations (increasing the number of poles used at each iteration) until it can find an acceptable fit to the data. For broad frequency sweeps and large numbers of excitations, this

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

process can be time consuming because of all the iterations and is not guaranteed to produce a good fit to the data. It is retained as a fallback if the TWA algorithm fails.

- **Enable relative error tolerance** – Enable relative error tolerance, which works best with the **TWA** algorithm.

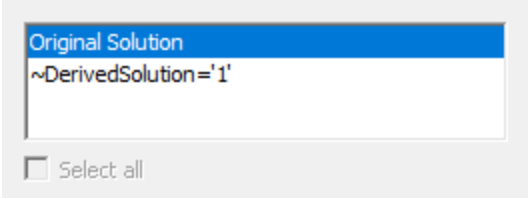
> **Note:**
>
> Although the **Enable relative error tolerance** option works best with the **TWA** fitting algorithm, is not recommended for use with the **Iterated rational function** algorithm, and is disabled when **FastFit** or **Ensure accurate Z-fit** is used.

- **Enforce Causality** (makes non-causal data causal-use only if fitting fails with this option off) – Select this box to enforce causality when needed for fitting to succeed. Measurement noise or numerical simulation errors can result in data that cannot be approximated accurately with a moderate number of poles. In particular, the data may resemble the response of a non-causal system, and since the Broadband model is causal by construction, it becomes impossible to achieve a good fit. In such cases, you may use the **Enforce Causality** option to approximate the data as that of a causal system prior to fitting. However, doing so may introduce significant fitting errors with respect to the original data. This option should be used only when the fitting fails without it.

Click **OK** to begin the export. The **Message Manager** pane details the export process.

Comparing Original S-Parameters with Exported S-Parameters

~DerivedSolution='1' is the exported solution. If you selected the **Compare fit** check box during export, the data selection pane updates to list both the original and derived solution.



Creating a Twin Builder Component

If you choose Twin Builder in the Full Wave Spice Format pane of the **Broadband Export Options** dialog box, you can also select an option to Create a Twin Builder. When you choose this option, **Network Data Explorer** can compile the exported model data and create a

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

component based on the data. The resulting component is then available for placement in a design.

Scripting for Network Data Explorer

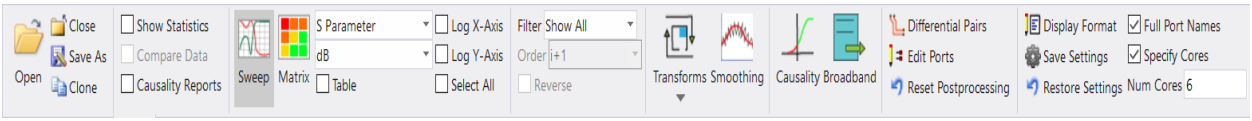
Scripting is available for each **Network Data Explorer** export method, which means that you can record a script to duplicate the export process. Invoke **Network Data Explorer** in the following contexts; scripting is available from the Project Context and the Design Instance Context (simulation setup).

- **No design** – When there is no design, the export functionality is not available, and so scripting is not available.
- **Project Context** – In the context of a project, you can open a Touchstone file and export it. Scripting is available.
- **Design Instance Context** – In the context of a solution (RCM from the simulation setup in a design), you can export the corresponding network data solution. Scripting is available.

For more information, see the *Twin Builder Scripting Guide*.

Network Data Explorer Ribbon

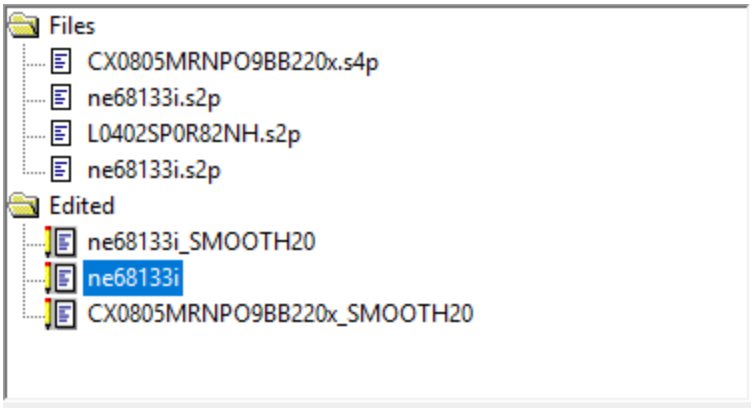
The **NDE** ribbon provides access to many of the Network Data Explorer's functions.



Data Sources

Use Network Data Explorer to easily view all data sources in the Network Data Section pane. This area displays the active source, any comparison source, and any derived variations.

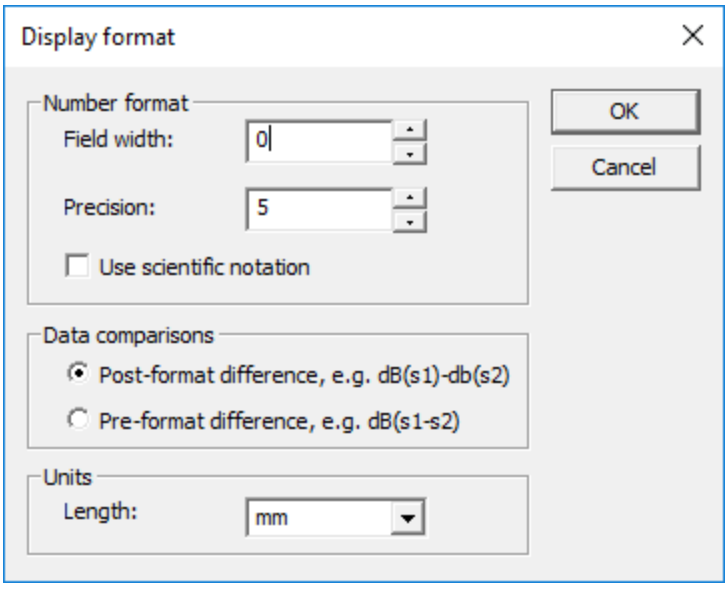
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Set Display Format

The **Display Format** dialog box affords additional control over the display of values in Network

Data Explorer.On the **NDE** ribbon, click **Edit how data is displayed in reports** . The **Display Format** dialog box appears.



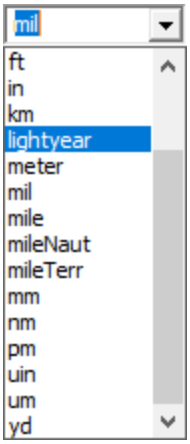
Use the **Number Format** options to specify **Field Width** (the minimum number of characters used to display a number) and **Precision** (the number of decimals to display). You can also select the **Use scientific notation** check box.

Use the **Data comparisons** options to choose post-format difference or pre-format difference.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Post-format difference** – When comparing data sets, subtract values after applying the formatting function (for example, dB, magnitude); the values displayed will be the difference between the magnitude, dB, and so on.
- **Pre-format difference** – When comparing data sets, subtract values before applying the formatting function (for example, dB, magnitude); the values displayed will be the magnitude, dB, and so on, of the complex difference.

Use the **Units** option to specify the **Length** unit (the unit used to display and interpret length values). The default is mm.

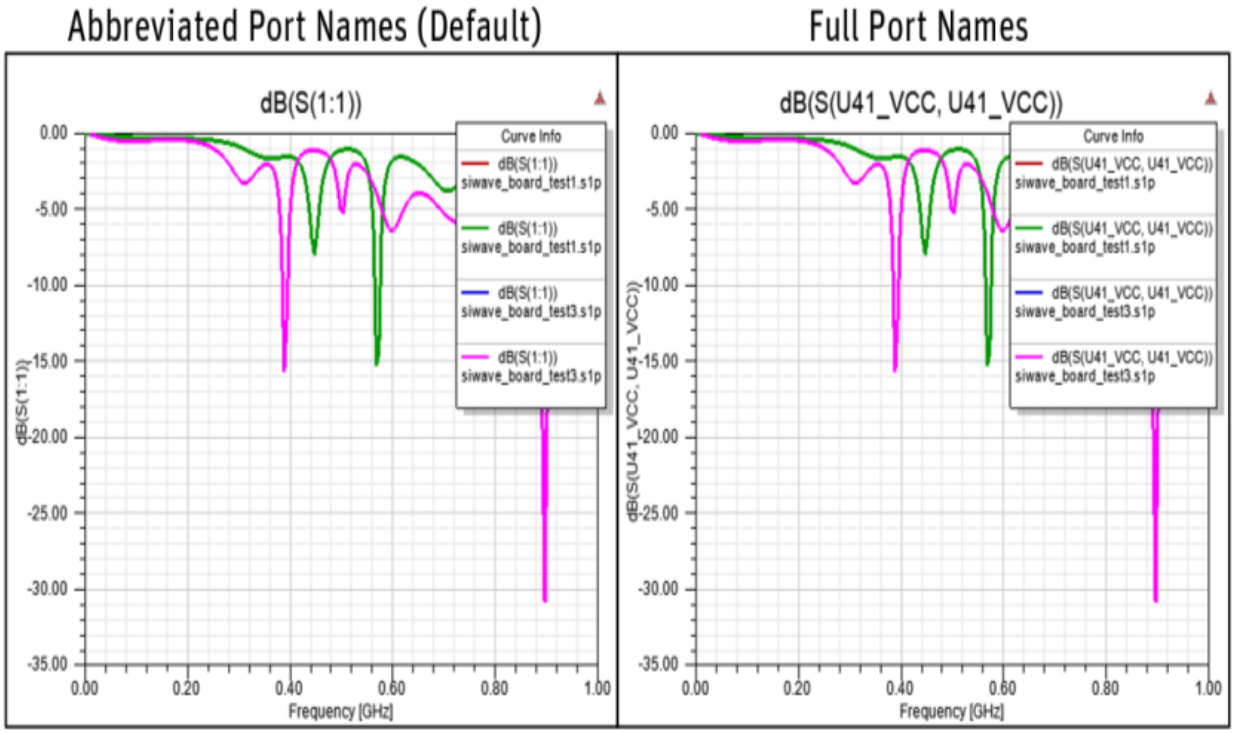


Display Full Port Names

By default, port names in Network Data Explorer appear in an abbreviated form: P1, P2, and so on. This applies to both the **Data Selection** pane and the Data View pane. To change this so that full port names are displayed, select the **Full Port Names** check box on the **NDE** ribbon.

The following figure shows the difference in display for a plot.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



> **Note:**
>
> Tool-tips always display the full port name.

Save or Reset Default Settings

To save field settings as the default, click on the **NDE** ribbon. The next time Network Data Explorer is opened, the chosen settings will be selected by default.

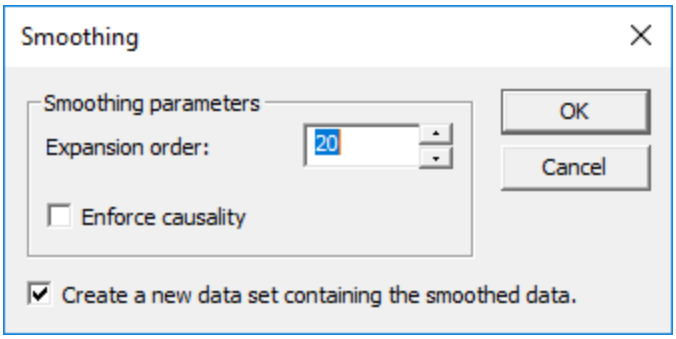
To restore the original defaults, click the on the **NDE** ribbon. The next time Network Data Explorer is opened, the original settings will be selected by default.

Smoothing

Follow this procedure to access data smoothing options.

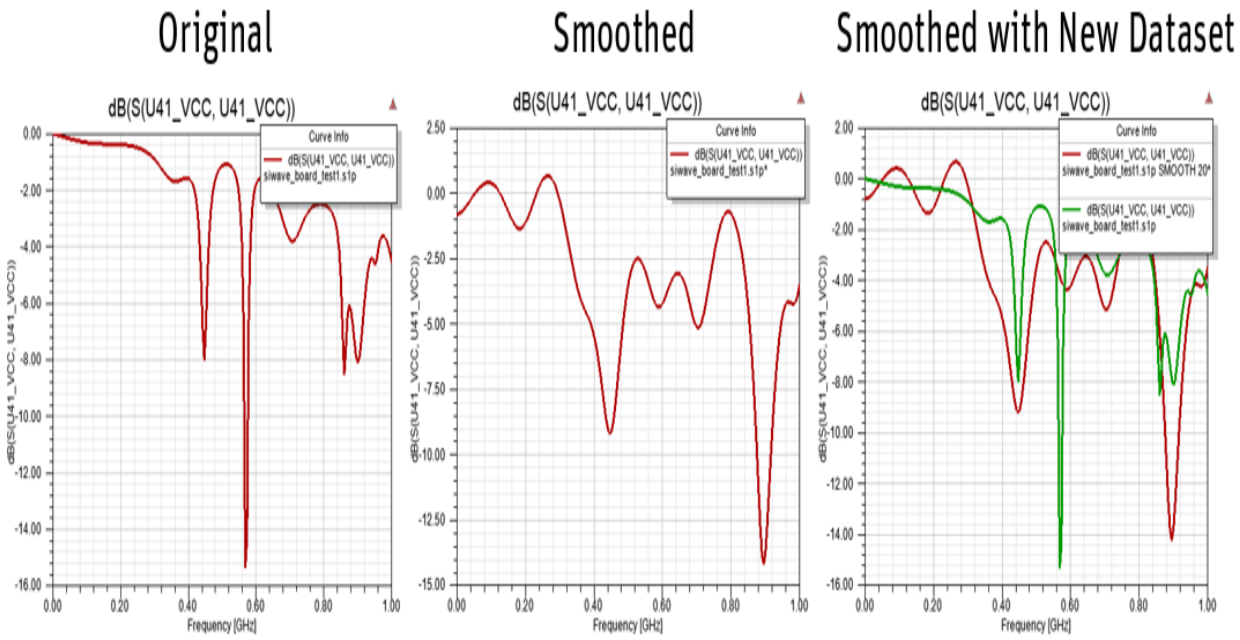
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Click on the **NDE** ribbon. The **Smoothing** dialog box appears.



2. The **Smoothing Parameters** area lets you choose the **Expansion Order**. This can be any discrete value between 1 and 150.
3. If desired, select the **Enforce Causality** check box.
4. If desired, select the **Create a new data set containing the smoothed data** check box. If selected, the smoothed data appears alongside the original data.
5. Click **OK**.
6. The data view pane updates. The following image shows a plotted result with **Create a new data set containing the smoothed data** selected and cleared.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

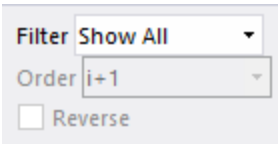


A least-squares polynomial fit of the specified order is used to interpolate new data points for the magnitude and phase components of the S-parameters.

- **Left window size** and **Right window size** control the number of data values collected on either side of the point to be interpolated. Note that these are point counts and not distances.
- **Spline order** can not exceed the total number of points sampled (that is, the sum of the two window sizes).

Cell Filtering

The cells available in the data selection pane may be restricted using cell filtering. The **Cell Filtering** control are located on the **NDE** ribbon.



Cell filtering is modeless, and filters are immediately applied to the cell list. Filtering remains in effect after the window closes.

For an n-port model with a total of 2n pins in the standard arrangement, the choices are:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

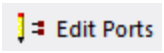
- **Show all** – Display all available cells. There are n-squared choices.
- **Return loss** – Show S(i, i). There are n choices.
- **Insertion loss** – Show S(i, i+1). There are n choices.
- **Lower triangle** – Show S(i, j) for all j < i. There are n(n-1)/2 choices.

Three pin arrangements are recognized:

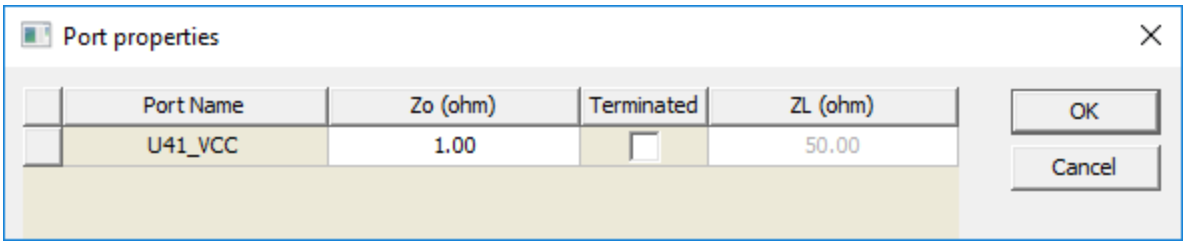
- **S(i, i+1)** and its **Reverse order**, S(i, i-1).
- **S(i, i+n)** and its **Reverse order**, S(i, i-n).
- **S(i, 2*n-i+1)** and its **Reverse order**, S(i, 2*n-i-1).

Changing Port Properties and Reducing Matrix Size

You can edit the normalization impedance, termination, port order, gamma values, and de-embedding distance with the **Port Properties** dialog box.

To access these options, click **Edit Ports** 

on the **NDE** tab. The **Ports Properties** dialog box opens.



Ports appear in a table. Click a column heading to sort by that column. Click within a cell to edit the port property:

- **Zo (ohm)** and **ZL (ohm)** – Specify Impedance values. Accepted syntaxes are:
  - real (e.g., 50)
  - real + imag i (e.g., 50+5i)
  - imag i (e.g., 5i).
- **Terminated** – Select this check box to terminate a port. Terminated ports are eliminated from the matrix, reducing the matrix size. Existing data sets with mismatching port numbers will no longer be available for data comparisons.
- **De-Embedding** – This column appears only if gamma values are available. Default units can be changed from the Set Display Format window.

Impedance values are specified in ohms and may be complex, the accepted syntaxes are:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Real. For example, 50.
- Real + imag i. For example, 50+5i .
- Imag i. For example, 5i .

Explicit units may be used with the impedance value (for example, 0.5kOhm) but will always be shown in ohms (when the dialog box is redisplayed).

Click and dragg a row to reorder ports. To see the port names in the data view displays, use the right-click menu option **Full Port Names**.

Terminated ports are eliminated from the matrix thereby reducing the matrix size. Existing data sets with mismatching port numbers will no longer be available for data comparisons.

Click **Edit** > **Reset All Port Properties** to restore port properties to their load time values.

The **de-embedding** column only appears if gamma values are available (specifically, if **ndExplorer** is invoked from the results menu within Twin Builder). Values are always displayed in the units indicated in the column header, however, values may be entered with specific units, such as "1.5in". The default units may be changed via the **Set Display Format** dialog box (available on the **Edit** menu or the right-click context menu).

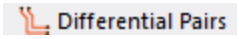
Once the changes have been committed, click **OK**. Select **File** > **Save As** to save the modified matrix values to a file.

Displaying Mixed-Mode Parameters Using Differential Pairs

Network Data Explorer displays mixed-mode parameters when differential pairs are both defined and activated.

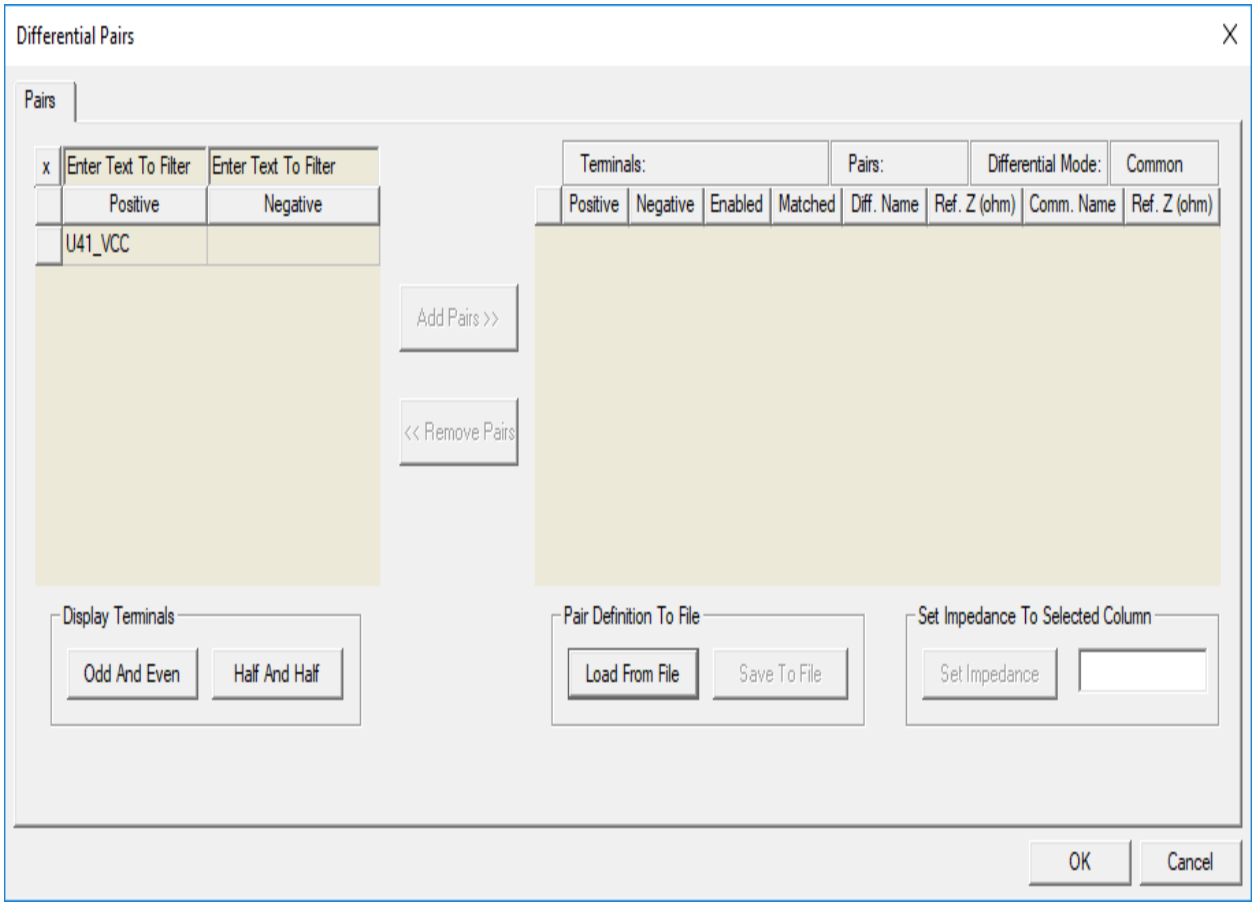
To define differential pairs:

1. Select existing ports.
2. Open Network Data Explorer (**Tools** > **Network Data Explorer**).
3. On the **NDE** ribbon, select **Differential Pairs**.

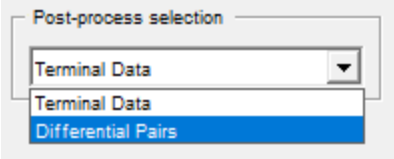


The **Differential Pairs** window appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. Select a pair from the list on the left and click **Add Pairs**.
5. Click **OK**.
6. Select **Differential Pairs** from the **Post-process selection** field to view mixed-mode parameters.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:** The Network Data Explorer **Edit** menu option **Reset All Port Properties** deactivates all pairs, but it does not clear the differential pair settings. And since **Reset All Port Properties** also clears reference impedances and terminations, it should not be used when the user simply wishes to disable all differential pairs. To disable all differential pairs, click the **Active** column header in the **Differential Pairs** dialog box to deselect all pairs.

Reset All Port Properties

Reset All Ports clears reference impedances and terminations, but does not disable all differential pairs. To disable all differential pairs, you must disable them in HFSS.

Right-Click Context Menu Options

The **Data View** pane presents different right-click menu options, depending on the context. Some commands are the same as those on the **NDE** ribbon. Others appear only in the context menus.

Topics for this section include:

Display Format

Multiple Frequency Statistics

Highlight Min/Max

Select Transpose

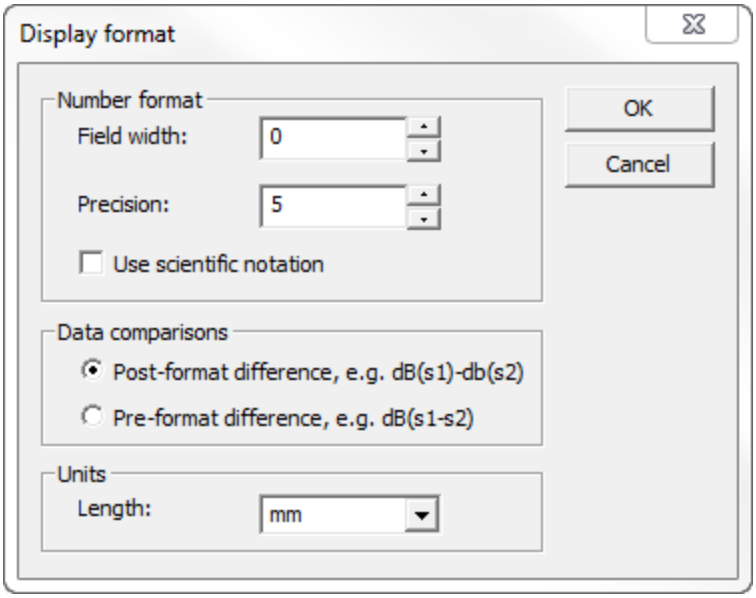
Full Port Names

Select Compare Variations

Display Format

Use the **Display format** dialog box for additional control over the display of values throughout **Network Data Explorer**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- **Field width** – The minimum number of characters used to display a number. Values are padded with blanks as necessary. Use to increase the blank space displayed in the data tables to force an increase in column width.
- **Precision** – The number of decimal places to display.
- **Use scientific notation** – Select this check box to display values using scientific notation (for example, 3.88124e-001).
- **Post-format difference** – When comparing data sets, subtract values after applying the formatting function (for example, dB, magnitude); the values displayed will be the difference between the magnitude, dB, and so on.
- **Pre-format difference** – When comparing data sets, subtract values before applying the formatting function (for example, dB, magnitude); the values displayed will be the magnitude, dB, and so on, of the complex difference.
- **Length** – The default units used to display and interpret length values (for example, de-embedding length).

Multiple Frequency Statistics

This option determines the statistical composite to display when multiple frequencies are selected for the matrix display. The statistical data is always the first matrix displayed, followed by matrices for each individual frequency. The **Multiple Frequency Statistics** also indicates the data used in the colored matrix plot when multiple frequencies are selected.

- **Average** – Display the average of the matrix values across the selected frequencies.
- **Minimum** – Display the minimum matrix values across the selected frequencies.
- **Maximum** – Display the maximum matrix values across the selected frequencies.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Highlight Min/Max

This option determines whether the minimum and maximum matrix entries are highlighted in the matrix table and color plot view.

Select Transpose

When selecting a cell in the table view of the matrix, this option determines if the transposed cell is highlighted as well.

Full Port Names

By default, port names appear in an abbreviated form: P1, P2, and so on. This applies to the matrix display and to the cell list display. Use this option to display the original port names. Note that tooltips always display the full port name.

Exploring Network Data and Modifying the Display

A number of features in the **Network Data Explorer** let you view data and modify various aspects of the display, including color plots, color coding, viewing across frequencies, and displaying individual statistics.

The topics for this section include:

Viewing the S, Y, or Z Matrix for a Selected Frequency

Color-Coded Matrix Plot

Changing the Color Scheme for a Matrix Color Plot

Viewing Matrix Cell Data Across All Frequencies

Cell Filtering

Displaying a Graph of a Cell Across All Frequencies

Displaying Statistics by Frequency

Displaying Individual Statistics for All Frequencies

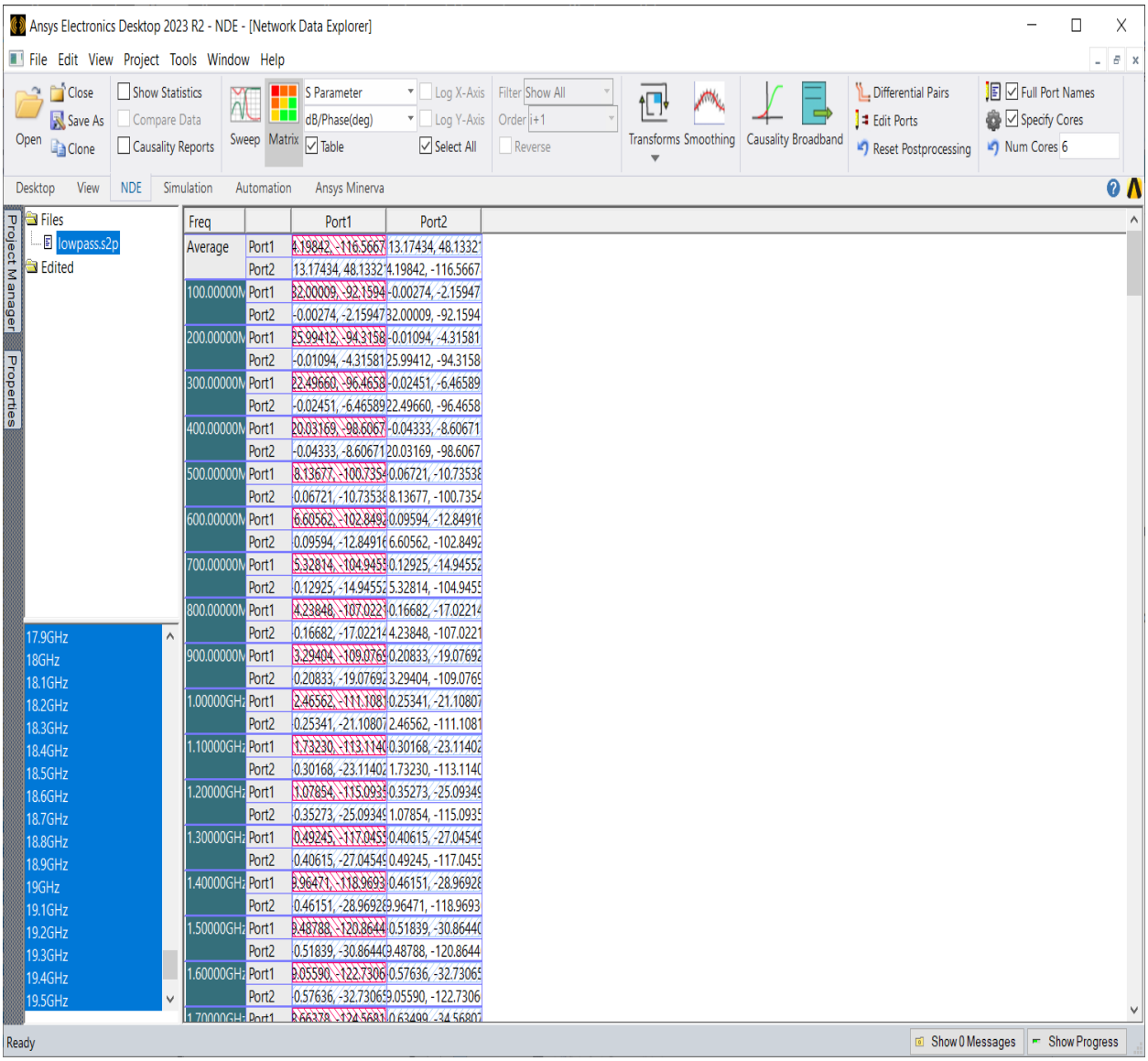
Viewing the S, Y, or Z Matrix for a Selected Frequency

Follow this procedure to view the S, Y or Z matrix:

1. On the **NDE** ribbon, use the **Parameter type** drop-down menu to select **S parameter**, **Y parameter**, or **Z parameter**.
2. Click **Matrix**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. In the **Cell and Frequency Selection** pane, select frequencies to display.
4. On the **NDE** ribbon, select the **Table** check box. The S, Y, or Z matrix appears.



Maximum values are highlighted in red stripes. If **Select Transpose** is enabled, transposes are highlighted in red stripes as well.

Minimum values are highlighted in blue stripes.

Selected cells appear in solid blue. Select **Select Transpose** to display the transpose in blue as well.

Double-click a cell to switch to a matrix cell view, in which values for all frequencies for that cell are displayed. The double-clicked frequency is highlighted with solid red shading

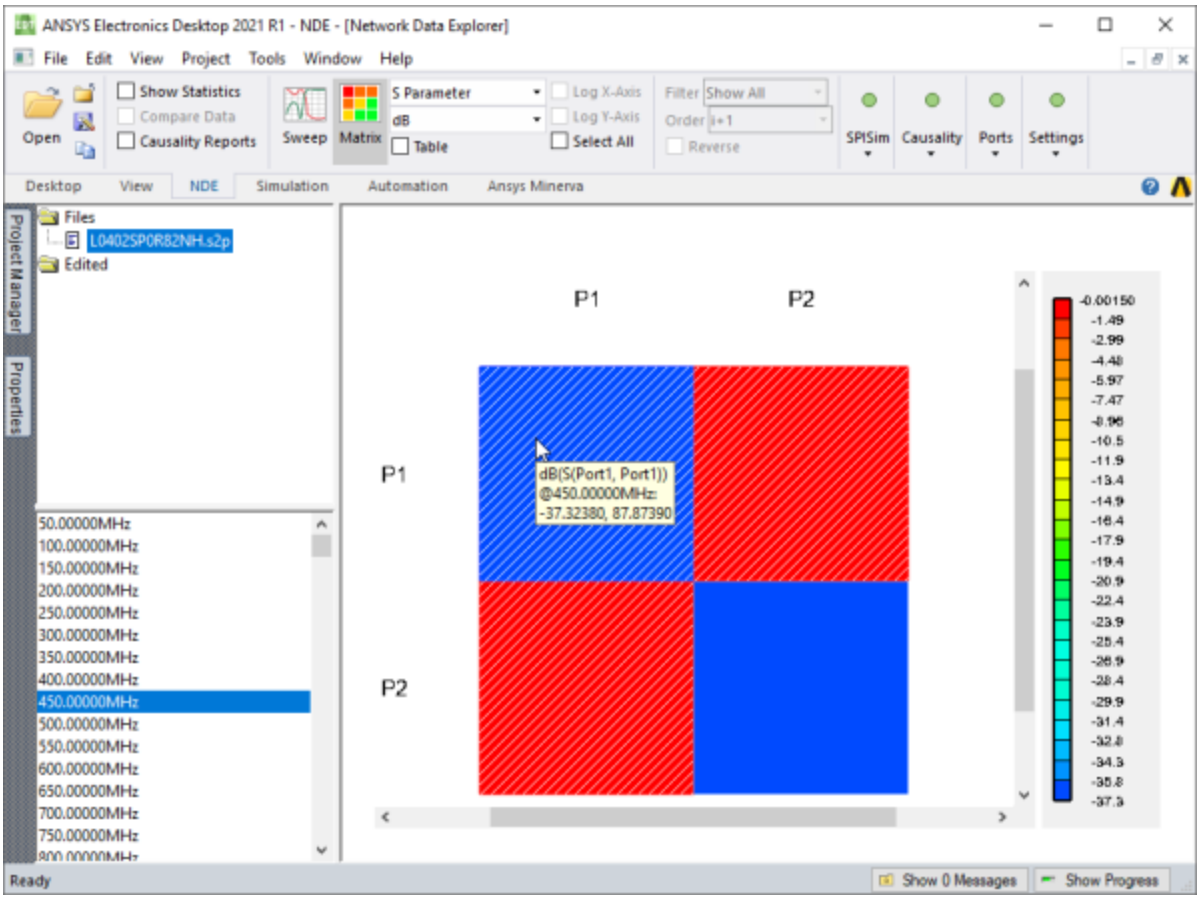
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Complex values are compared using their modulus. When multiple frequencies or variations are selected, the data display depends on the **Multiple Frequency Statistics** setting.

Color-Coded Matrix Plot

To view a color-coded matrix plot:

1. On the **NDE** ribbon, use the **Parameter type** drop-down menu to select **S parameter**, **Y parameter**, or **Z parameter**.
2. Click **Matrix**. The color-coded matrix plot displays.
3. In the Cell and Frequency Selection pane, select the frequencies you want to plot.



Matrix values display in a color-coded grid. If the selected **Format** is a complex value, only the real component determines the display color. When multiple frequencies or variations are selected, the data display depends on the **Multiple Frequency Statistics** setting.

Maximum values are highlighted in red.

Minimum values are highlighted in dark blue.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Hover the cursor over any cell to view information about it.

Click any cell to select it. Selected cells appear with a pink outline. If **Select Transpose** is enabled, the transpose is selected in pink as well.

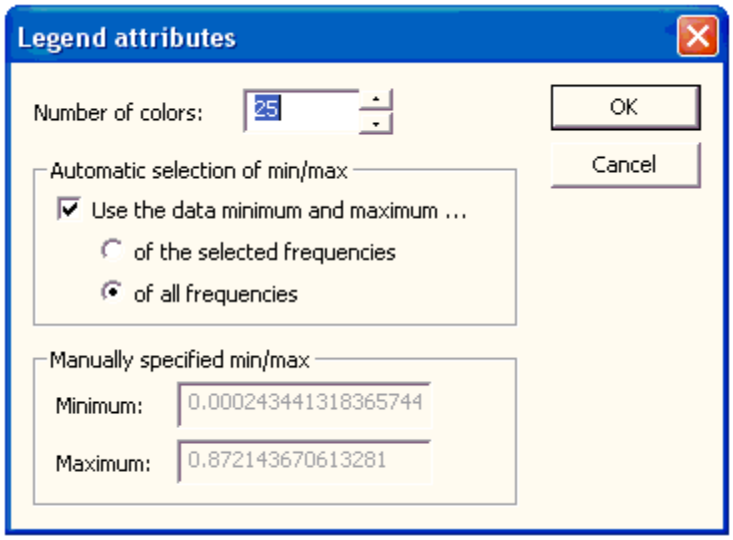
Double-click any cell to view a matrix cell plot in which all frequency values for that matrix cell are displayed as a graph.

Related Topics:

- Setting Multiple Frequency Statistics
- Changing Color Legend Attributes

Changing the Color Scheme for a Matrix Color Plot

Double-click the matrix color plot legend, or right-click within the plot itself and choose **Color Legend Attributes** to display the **LegendAttributes** dialog box, in which you can change the granularity of the color scheme and the value range, but not the colors themselves. Use a standard range across all frequencies to quantitatively compare plots; ndExplorer remembers legend settings for each data-type/display-format pair.



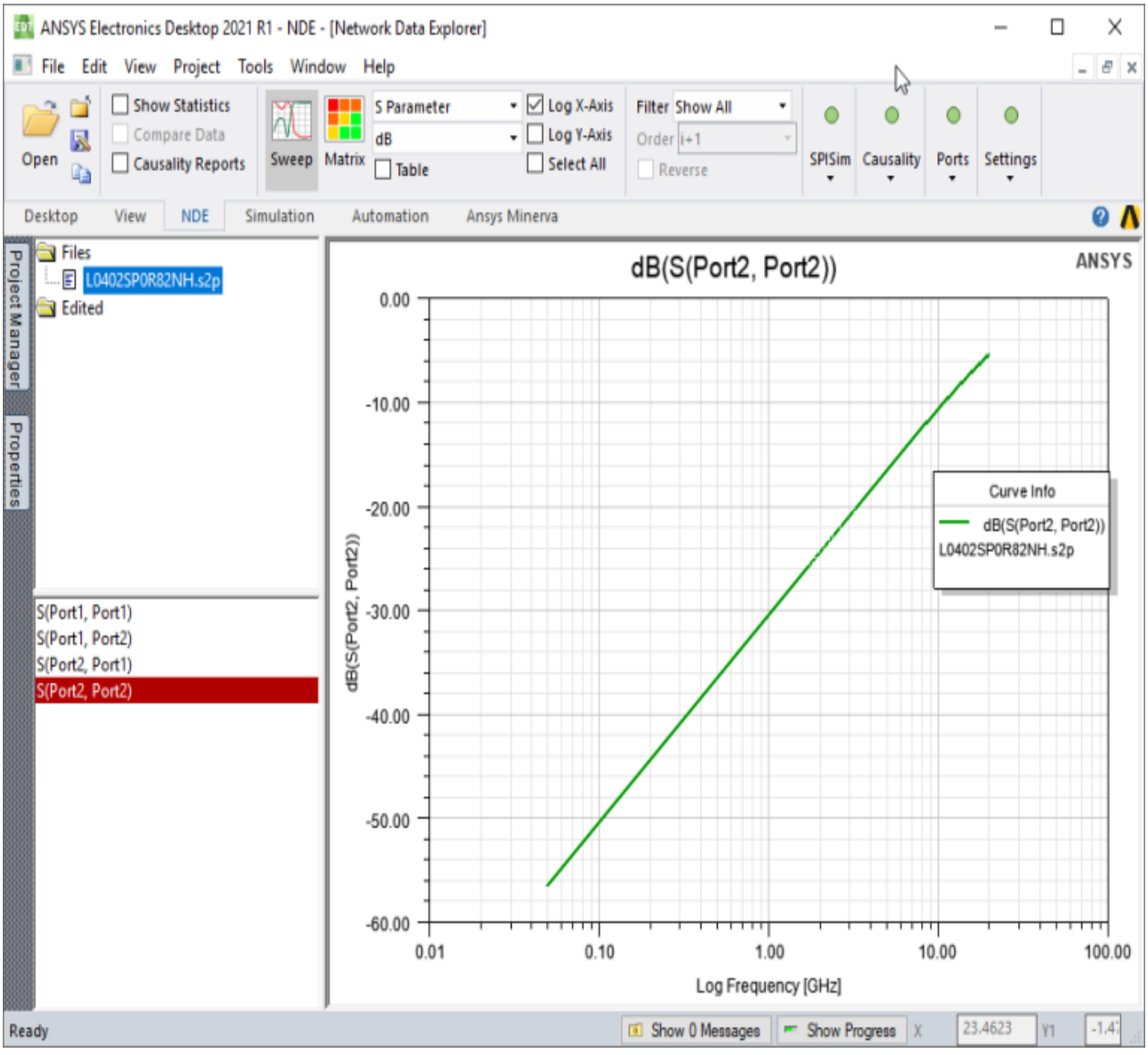
- **Number of colors** – Number of color entries in the legend; that is, the number of divisions between the start/end of the data range.
- **Use the data minimum and maximum** – Select the data range using the minimum and maximum values from either just the **selected frequencies** or **all frequencies** in the data set.
- **Minimum & Maximum** – When the range is not determined, use these fields to manually enter hard values. For example, for S parameter data magnitude data, a minimum of 0 and a maximum or 1 could be used.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Displaying a Graph of a Cell Across All Frequencies

Network Data Explorer can plot a cell across all frequencies.

1. On the **NDE** ribbon, select the **Parameter type** drop-down menu and choose **S parameter**, **Y parameter**, or **Z parameter**.
2. Click **Sweep**. The Data View pane updates if necessary.
3. In the Cell and Frequency Selection pane, select frequencies to display.
4. To add a log scale to the X-axis, click **Log X-Axis** on the **NDE** ribbon.
5. To add a log scale to the Y-axis, click **Log Y-Axis** on the **NDE** ribbon.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Viewing Matrix Cell Data Across All Frequencies

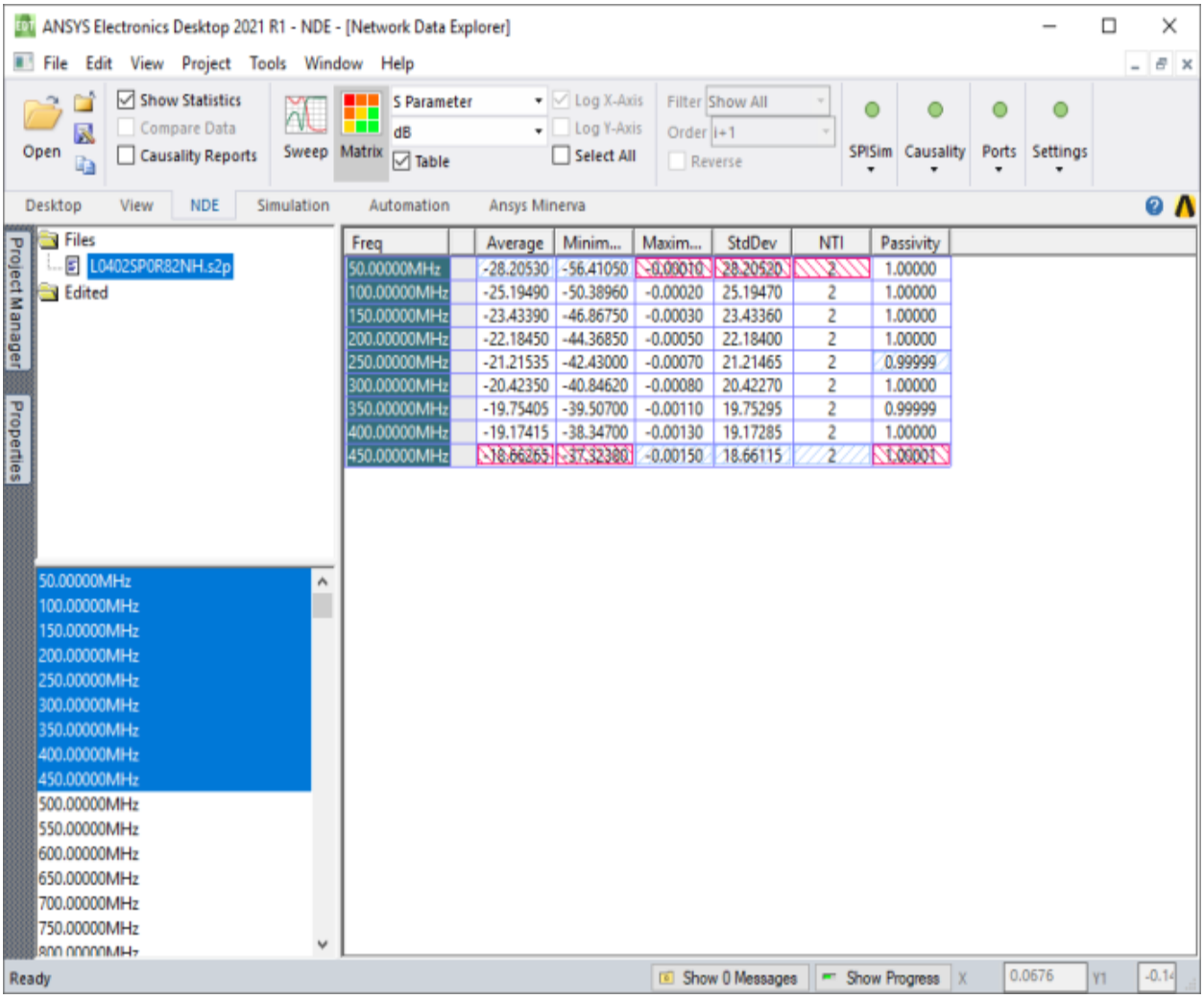
1. Select **Parametervalues** in the **Quantity** field.
2. Select **Matrix entries** in the data selection pane to the left.
3. Select **Table** in the data view pane to the right.
4. Choose the frequency and variation to display.
   - Values for all frequencies appear for each selected matrix cell. The data type and format are determined by the **Parametertype** and the **Format** settings in the control pane.
   - Place the cursor over a table cell to display a tooltip revealing more detail about the content of the cell. Click a cell to select it. Double-click a cell to see the matrix table view in which all cell values for the corresponding frequency appear; the double-clicked cell is highlighted with solid red shading.
   - Click a column header in the data table to highlight the column. The corresponding trace in the plot view is also highlighted.
   - Select **Plot** in the data view pane to switch the view to a graph of the cell data. The graph shares the same color coding as the columns; highlighted columns are highlighted in the plot.

Displaying Statistics by Frequency

Network Data Explorer can display various statistical measurements.

1. On the **NDE** ribbon, click **Matrix**.
2. Click **Table**.
3. Click **Show Statistics**.
4. In the Cell and Frequency Selection pane, select frequencies to display.
5. To change the statistics displayed, use the populated list to select statistics.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The various statistical measures for the current **Parametertype** and **Format** display for each frequency selected. Only real (not complex) data formats are offered for statistical analysis. **Passivity** is only available for S-parameter data (comparisons inactive). **NTI** refers to the number of trivial items. For S-parameters, this includes all zeros and ones; for all other data (and data comparisons), only zeros count as trivial. The minimum value for each column is highlighted in blue; the maximum is highlighted in red.

When you hover the cursor over a cell, a tooltip indicates the frequency and statistics displayed. Click a cell to select it. Multiple variations display as separate entries in the table; use the tooltip to identify the variation for a particular frequency.

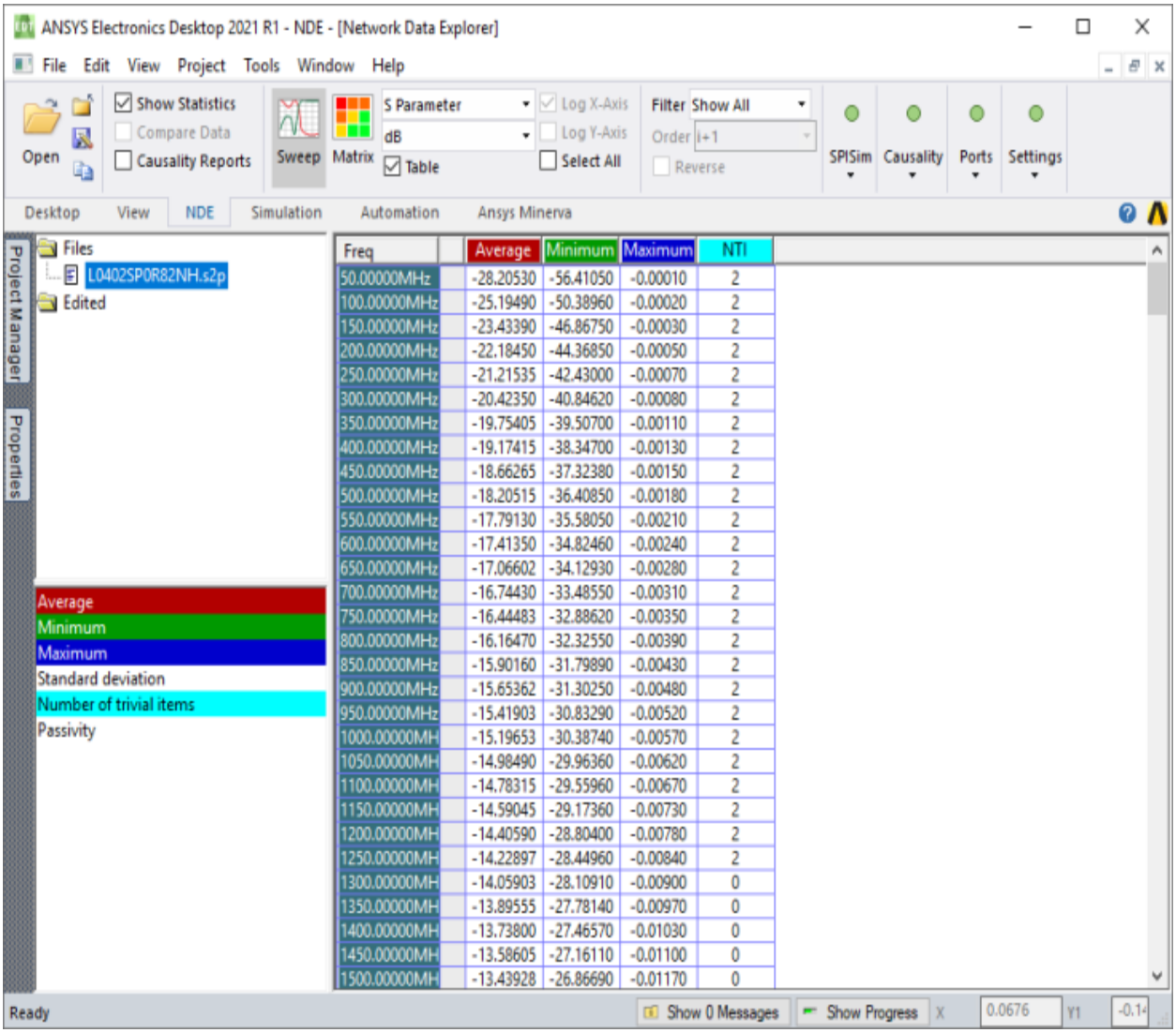
Click a column header to sort the data using that column for comparison.

Displaying Individual Statistics for All Frequencies

Network Data Explorer can display various statistical measurements.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. On the **NDE** ribbon, click **Sweep**.
2. Click **Table**.
3. Click **Show Statistics**.
4. In the Cell and Frequency Selection pane, select frequencies to display. The information displays in a table in the Data View pane.
5. To change the statistics displayed, use the populated list to select statistics.



The various statistical measures (for the current **Parametertype** and **Format**) are displayed for all frequencies. **Passivity** is only available for S-parameters (comparisons inactive). **NTI** refers to the number of trivial items. For S-parameters, this includes all zeros and ones; for all other data (and data comparisons), only zeros count as trivial.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

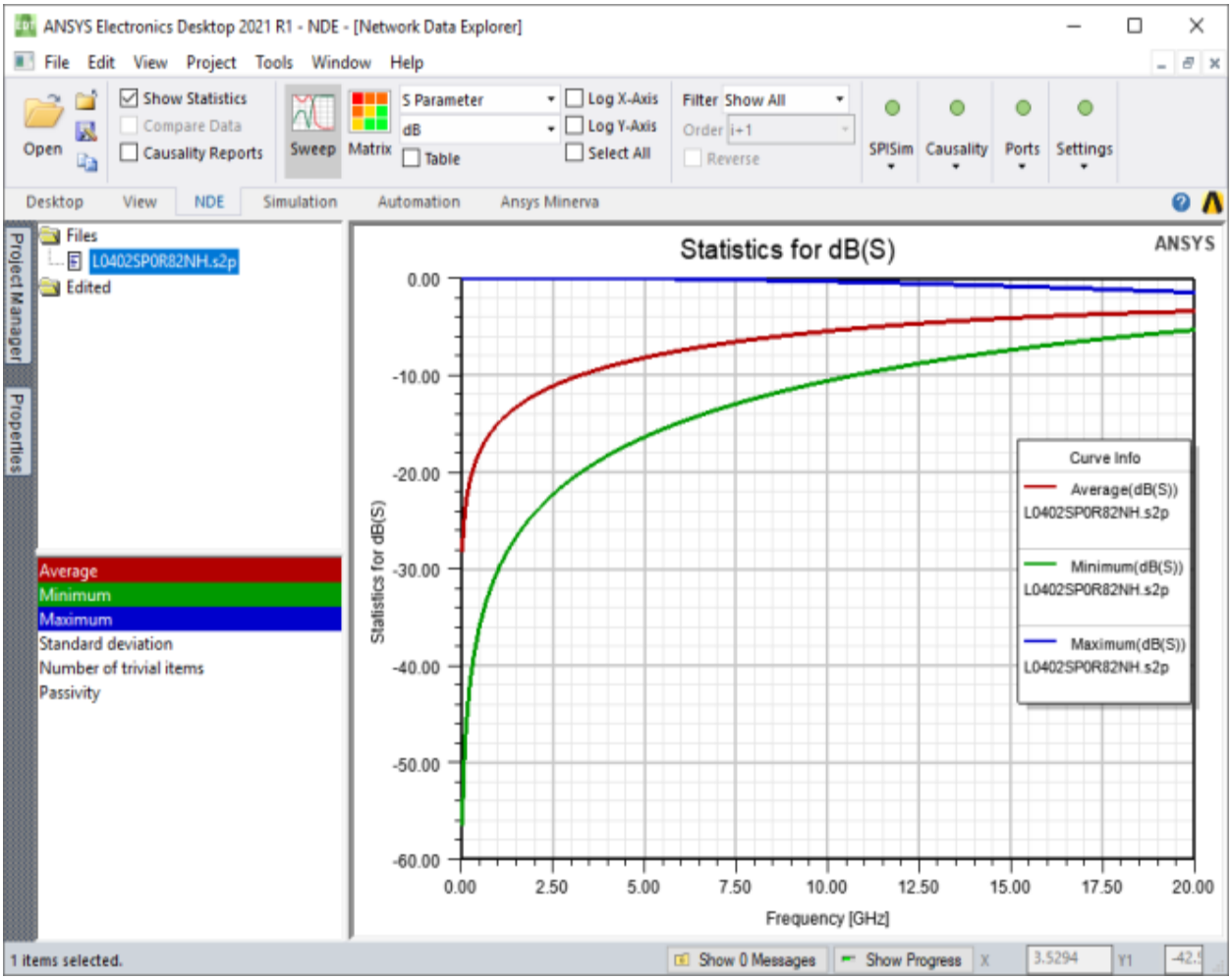
When you hover the cursor over a cell, a tooltip indicates the frequency and statistics displayed. Click a cell to select it. Multiple variations display as separate entries in the table; use the tooltip to identify the variation for a particular frequency.

Select **Plot** in the data view pane to switch the view to a graph of the statistical data. The graph shares the same color coding as the columns; highlighted columns are highlighted in the plot.

Creating a Statistics Plot

Network Data Explorer can display a graph of selected statistical measures across all frequencies.

1. On the **NDE** ribbon, click **Sweep**.
2. Click **Show Statistics**.
3. In the Cell and Frequency Selection pane, select statistics to display. The selected statistics are plotted.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Hover the cursor over a statistic to view more information about it.

Passivity is only available for S-parameters (comparisons inactive).

For S-parameters, the **Number of Trivial Items (NTI)** includes all values of 0 and 1. For other data and data comparisons, only values of 0 are counted as trivial.

Comparing Network Data

Use the Network Data Explorer to compare data in an open file with data in any other open file. In the **File Selection** pane, press and hold Ctrl and click the files whose data sets you want to display in the **Data View** pane. In the **Data Selection** pane, click which frequencies you want to view.

Comparing Variations

Network Data Explorer can compare variations for Neutral Format (*.**nmf**) files or multiple network data sets that are the same size.

1. On the **NDE** ribbon, click **Open**.
2. In the **Open** window, select two or more Neutral Format files or network data files that are the same size and click **Open**.
3. In the **Network Data Selection** pane, select the newly opened files.
4. On the **NDE** ribbon, click **Compare Data**.
5. In the **Cell and Frequency Selection** pane, select which ports you want to compare . For each value along the X-axis, the Y-axis values are subtracted, one from the other, to create the comparison plot.

It is not possible to compare a data set against itself unless the data set has been loaded twice.

Traces for a given cell or statistical measure are displayed for all data sets; you can use tool tips to distinguish between them.

If a single cell or statistical measure is displayed, different colors are used for each data trace. If multiple cells or statistical measures are selected, a single color is used for all data traces for each cell or statistical measure.

In a data comparison, traces are shown for all selected data sets. This is true for both cell and statistical traces.

Data sets with no variation information are always displayed. With multiple variation data sets, only those with values for the currently selected variations are displayed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

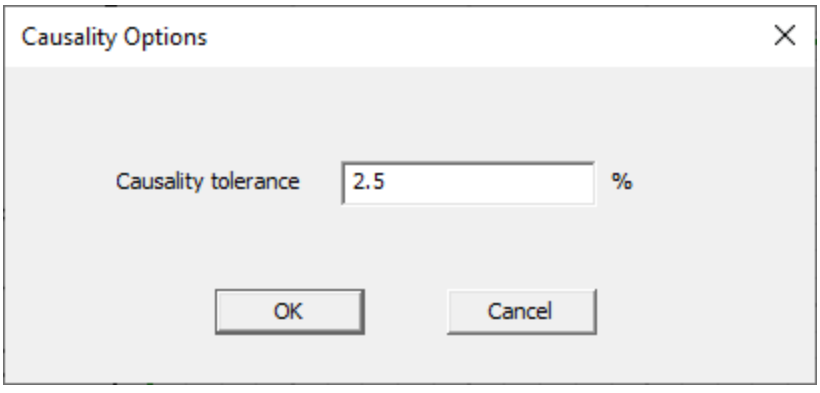
Displaying Plot Traces from Multiple Data Sources

1. Open multiple data sets, either through the main file browser or through the data comparison browser.
2. Use the **Edit** menu or the right-click menu to open the **Data Sources** dialog box.
3. Enable **Display plot traces for all data sources**.
   - Traces for a given cell or statistical measure appear for all data sets; you can use tooltips to distinguish between them.
   - If a single cell or statistical measure is displayed, different colors are used for each data trace.
   - If multiple cells or statistical measures are selected, a single color is used for all data traces for each cell or statistical measure.
   - If a data comparison is active, traces are shown for all data sets compared against it (with the exception of the comparison data set). This is true for both cell and statistical traces.
   - Data sets with no variation information always display. With multiple variation data sets, only those with values for the currently selected variations display. Note that the variation for the comparison data set is fixed by the selection made in the **Select Compare Variations** dialog box.

Causality Checking and Plots

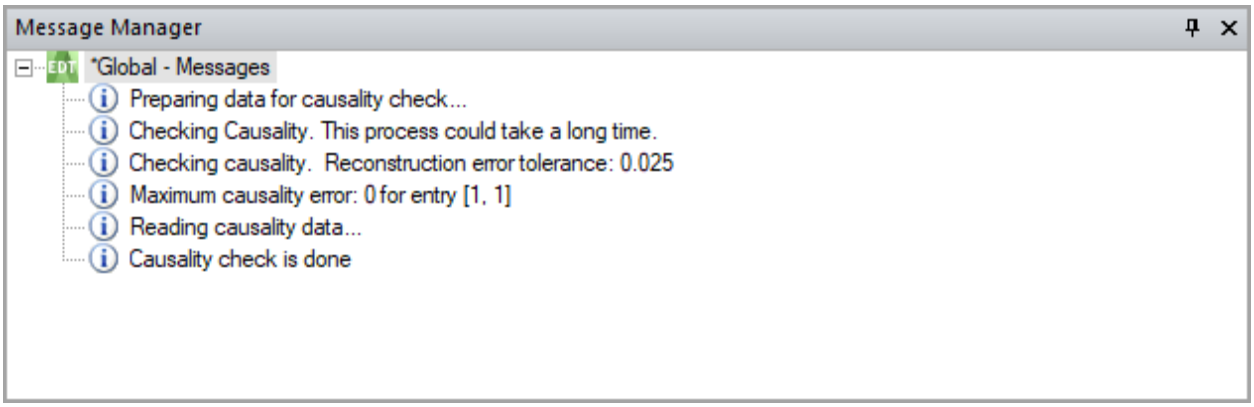
Network Data Explorer can perform a causality check on S-parameter data from any source (solution or file), and provide plots of the results in various formats.

When S-parameter data is loaded into **Network Data Explorer,** click **Causality** on the NDE ribbon. The **Causality Options** dialog box appears.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Enter a **Causality tolerance** and click **OK** to start the causality check. Depending on the size of the S-parameter data, the causality check may take several minutes to complete. The check's status displays in the **Message Manager** pane.



When the check completes, the **Message Manager** pane updates to display a summary of results.

Reconstruction Error Tolerance – Causality of a frequency response is determined by calculating the generalized Hilbert transform of the data at all frequencies. A causal frequency response is equal to its generalized Hilbert transform. The reconstruction error is the difference between the tabulated data and its transform at a given frequency. The message shows the maximum reconstruction error tolerance for a causal frequency response. The default tolerance of 0.01 is equal to the state-space fitting tolerance.

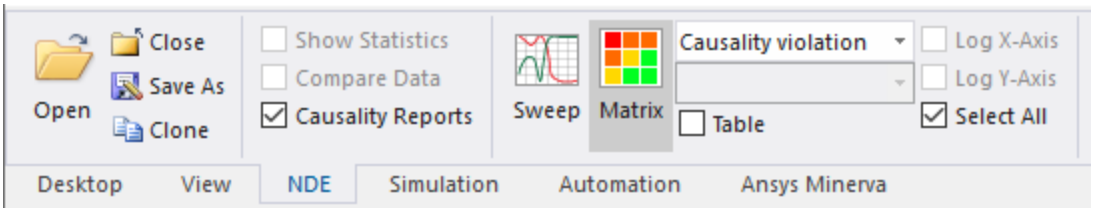
Maximum Causality Error – The maximum causality error for all port pairs and all frequencies, along with the matrix indices (port numbers) where the maximum noncausality occurs. A noncausal response is one where all matrix entries can be conclusively analyzed, and at least one entry exceeds the causality tolerance. The maximum reconstruction appears first, followed by port numbers in brackets (for example, **[***port number***,** *port number***]**). When all results are conclusive but no matrix reconstruction error exceeds the tolerance, the maximum causality error is reported as zero, and no matrix entry is listed.

If the data does not contain enough frequency points to determine whether the data is causal, the **Message Manager** pane will note an inconclusive result. **Network Data Explorer** will also report the data set as inconclusive if any cells are inconclusive, even if other entries exhibit causality violations.

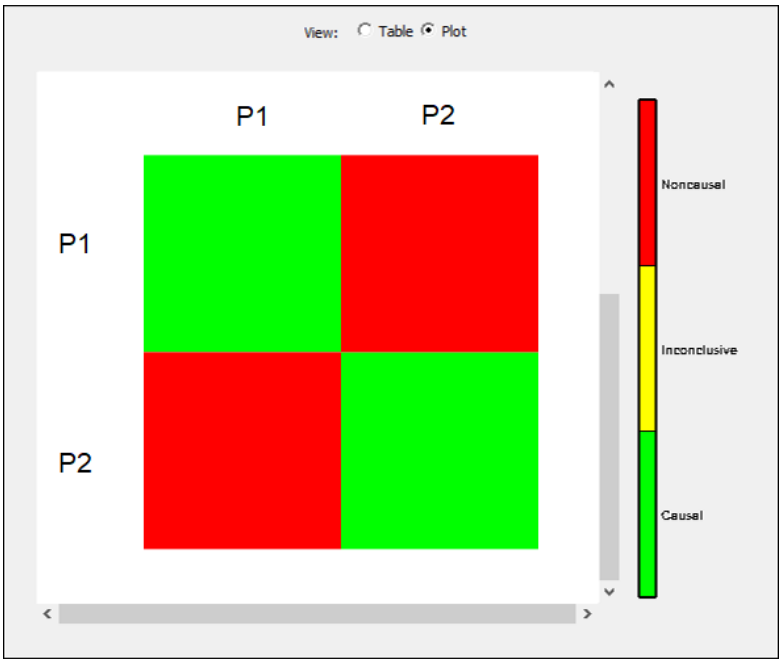
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To see the plotted results of the causality check:

1. On the **NDE** ribbon, select the **Causality Reports** check box.
2. Either click the Matrix icon or select **Causality violation** in the Parameter type drop-down menu.



3. A rectangular plot displays, with dimension N x N and color-coding to indicate the causality status of each port pairing.



In this plot, the cells go from (Port 1, Port 1) at the upper-left area to (Port N, Port N) at the lower-right corner. The result shows the causality over all frequencies in the data. In this example, the matrix is symmetric, so that both S12 and S21 are noncausal, while S11 and S22 are causal.

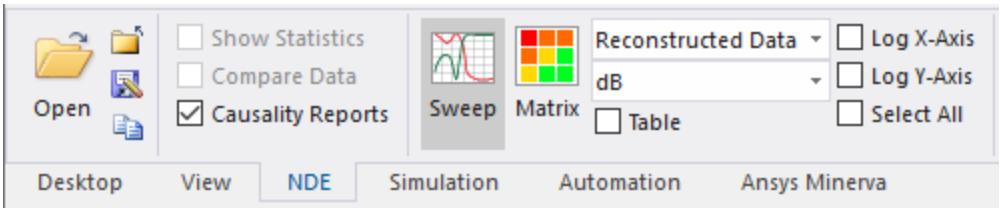
To see the details for each frequency, select the **Table** check box on the **NDE** ribbon.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Freq		Port1	Port2	Port1	Port2
0.10000GHz	Port1	0.01715, 0.00000	0.00050, 0.00000	0.01493, 0.00000	0.00098, 0.00000
	Port2	0.53111, 0.00000	0.00579, 0.00000	0.11471, 0.00000	0.00832, 0.00000
0.20000GHz	Port1	0.01367, 0.00000	0.00059, 0.00000	0.00736, 0.00000	0.00052, 0.00000
	Port2	0.27237, 0.00000	0.00394, 0.00000	0.05933, 0.00000	0.00423, 0.00000
0.30000GHz	Port1	0.01362, 0.00000	0.00105, 0.00000	0.00344, 0.00000	0.00028, 0.00000
	Port2	0.24509, 0.00000	0.00239, 0.00000	0.01921, 0.00000	0.00195, 0.00000
0.40000GHz	Port1	0.00907, 0.00000	0.00103, 0.00000	0.00171, 0.00000	0.00006, 0.00000
	Port2	0.16665, 0.00000	0.00163, 0.00000	0.01530, 0.00000	0.00084, 0.00000
0.50000GHz	Port1	0.00693, 0.00000	0.00062, 0.00000	0.00075, 0.00000	0.00006, 0.00000
	Port2	0.11877, 0.00000	0.00106, 0.00000	0.00542, 0.00000	0.00040, 0.00000
0.60000GHz	Port1	0.00295, 0.00000	0.00044, 0.00000	0.00047, 0.00000	0.00005, 0.00000
	Port2	0.05535, 0.00000	0.00042, 0.00000	0.00462, 0.00000	0.00015, 0.00000
0.70000GHz	Port1	0.00015, 0.00000	0.00009, 0.00000	0.00029, 0.00000	0.00003, 0.00000
	Port2	0.00510, 0.00000	0.00012, 0.00000	0.00114, 0.00000	0.00008, 0.00000
0.80000GHz	Port1	0.00058, 0.00000	0.00068, 0.00000	0.00047, 0.00000	0.00004, 0.00000
	Port2	0.02772, 0.00000	0.00088, 0.00000	0.00115, 0.00000	0.00010, 0.00000
0.90000GHz	Port1	0.00210, 0.00000	0.00081, 0.00000	0.00040, 0.00000	0.00013, 0.00000
	Port2	0.05492, 0.00000	0.00133, 0.00000	0.00173, 0.00000	0.00010, 0.00000
1.00000GHz	Port1	0.00221, 0.00000	0.00064, 0.00000	0.00033, 0.00000	0.00010, 0.00000
	Port2	0.07703, 0.00000	0.00142, 0.00000	0.00111, 0.00000	0.00009, 0.00000
1.10000GHz	Port1	0.00410, 0.00000	0.00045, 0.00000	0.00037, 0.00000	0.00008, 0.00000
	Port2	0.08851, 0.00000	0.00151, 0.00000	0.00137, 0.00000	0.00011, 0.00000
1.20000GHz	Port1	0.00433, 0.00000	0.00078, 0.00000	0.00017, 0.00000	0.00013, 0.00000
	Port2	0.09013, 0.00000	0.00100, 0.00000	0.00168, 0.00000	0.00017, 0.00000
1.30000GHz	Port1	0.00377, 0.00000	0.00072, 0.00000	0.00020, 0.00000	0.00018, 0.00000
	Port2	0.09487, 0.00000	0.00151, 0.00000	0.00081, 0.00000	0.00012, 0.00000
1.40000GHz	Port1	0.00380, 0.00000	0.00035, 0.00000	0.00020, 0.00000	0.00014, 0.00000
	Port2	0.08918, 0.00000	0.00136, 0.00000	0.00139, 0.00000	0.00009, 0.00000
1.50000GHz	Port1	0.00417, 0.00000	0.00005, 0.00000	0.00012, 0.00000	0.00011, 0.00000
	Port2	0.08091, 0.00000	0.00174, 0.00000	0.00094, 0.00000	0.00021, 0.00000
1.60000GHz	Port1	0.00315, 0.00000	0.00038, 0.00000	0.00006, 0.00000	0.00015, 0.00000
	Port2	0.07512, 0.00000	0.00208, 0.00000	0.00065, 0.00000	0.00011, 0.00000
1.70000GHz	Port1	0.00275, 0.00000	0.00034, 0.00000	0.00006, 0.00000	0.00021, 0.00000
	Port2	0.06069, 0.00000	0.00152, 0.00000	0.00101, 0.00000	0.00021, 0.00000

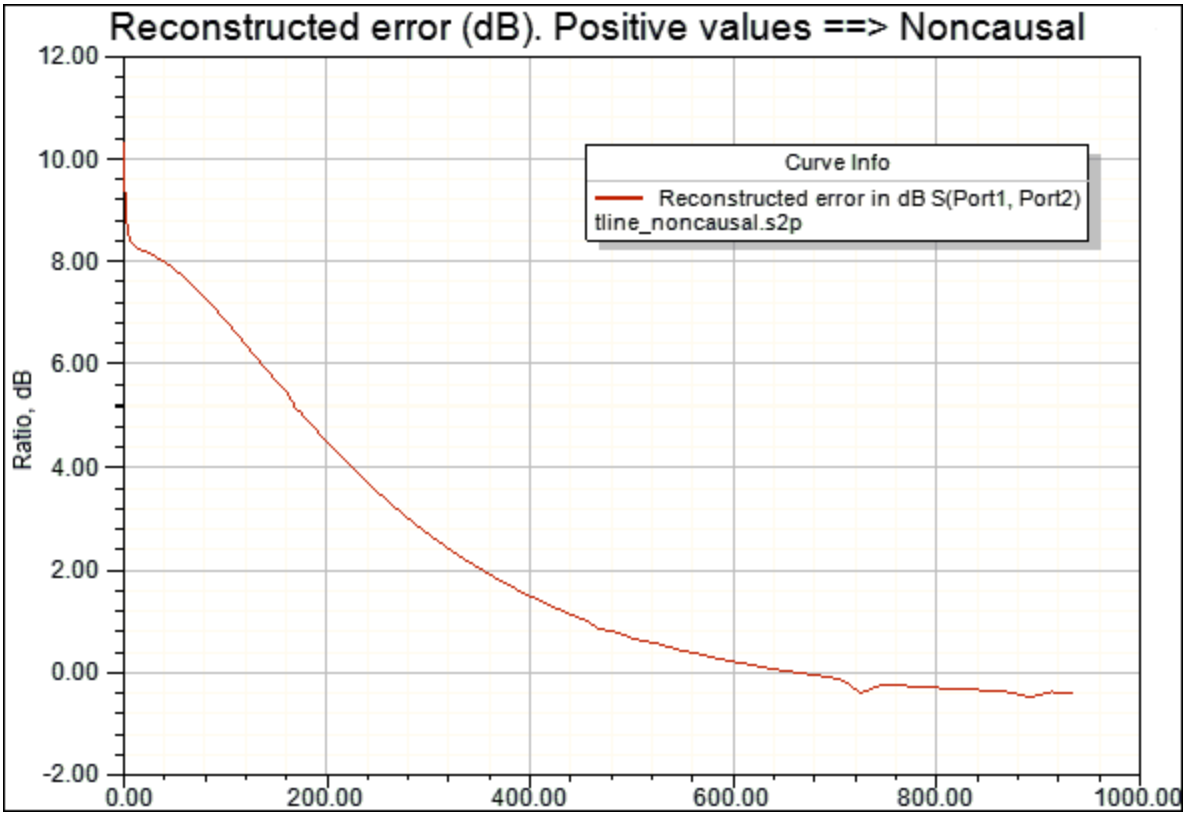
To plot the reconstructed frequency response generated by the causality checker:

1. On the **NDE** ribbon, click **Causality Reports**.
2. Either click the **Sweep** icon or select **Reconstructed Data** in the **Parameter type** drop-down menu.
3. In the **Format** drop-down menu, select the desired format. **dB** is selected by default.



4. The plot appears, showing the reconstruction error at each frequency divided by the tolerance.

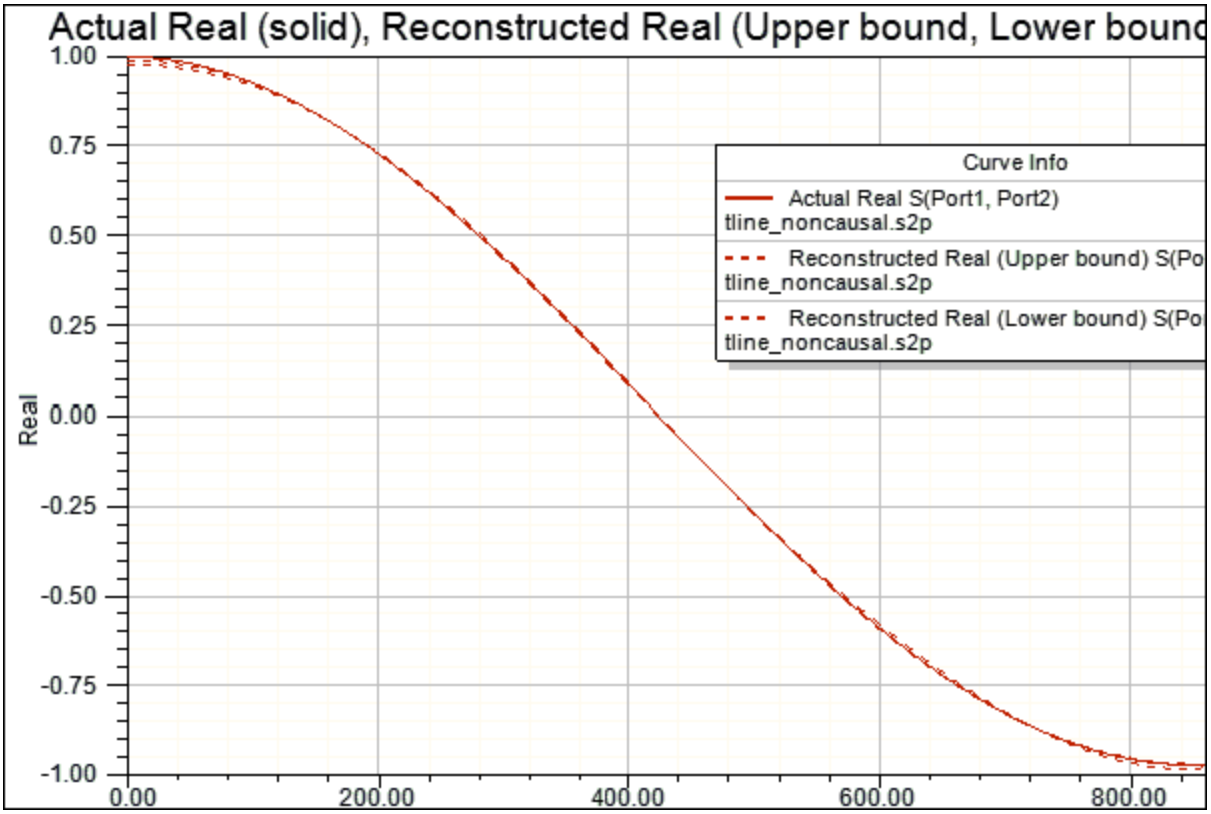
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The reconstruction error ratio for parameter S12 is positive for frequencies less than about 680MHz, indicating a broad range of noncausal behavior.

5. To compare the real part of the reconstructed data to the real part of the actual data, set the **Format** to **Real**.

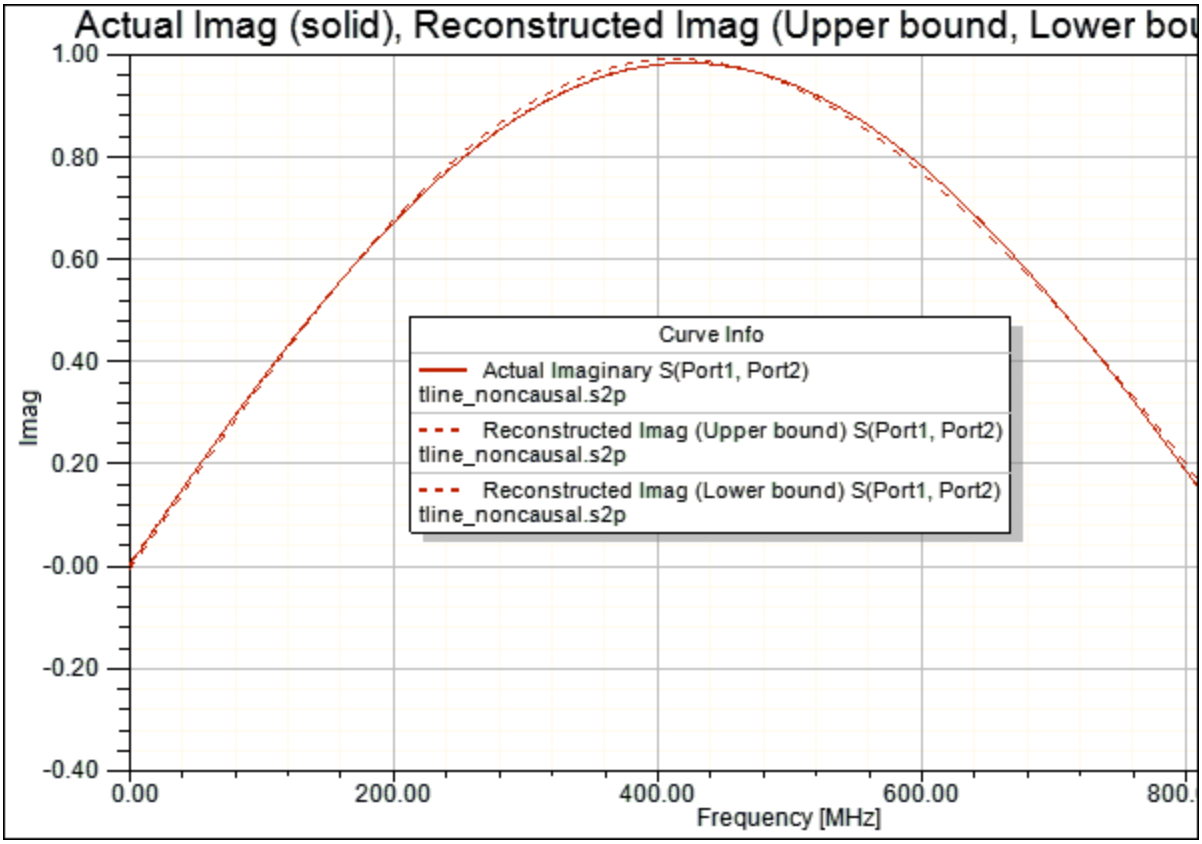
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



For a causal frequency response, the actual data (solid line) is in the upper and lower bounds of the reconstruction (dotted lines) at all frequencies.

6. To compare the imaginary part of the reconstructed data to the imaginary part of the actual data, set the **Format** to **Imaginary**.

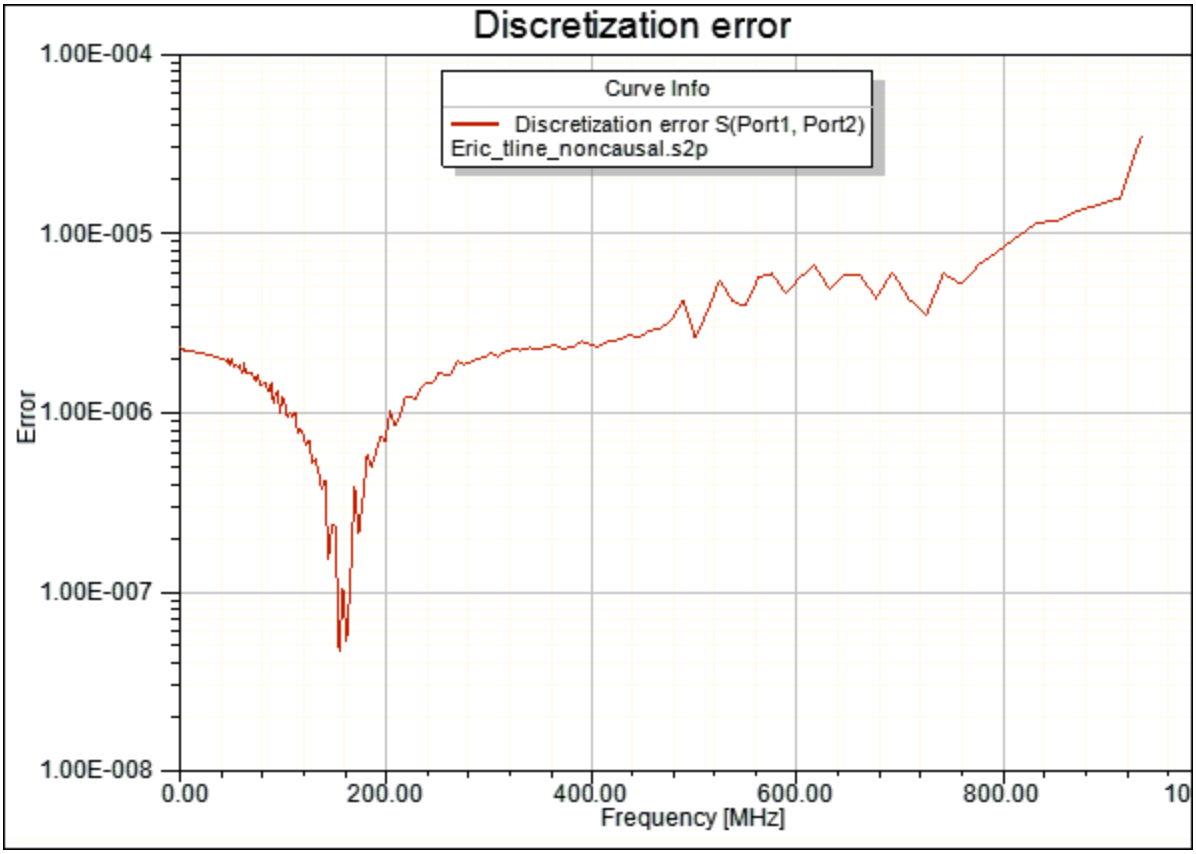
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



For a causal frequency response, the actual data (solid line) will be within the upper and lower bounds of the reconstruction (dotted lines) at all frequencies.

To view the frequency-dependent discretization error, set the **Format** to **Discretization**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The discretization error is the error that occurs because the data is available only at discrete frequencies rather than for a continuous spectrum. A discretization error near or greater than the causality tolerance renders the causality check inconclusive. Data at more frequencies could reduce the discretization error and render the analysis conclusive. This set of data exhibits low discretization errors (<<0.01) at all frequencies, and the causality check is conclusive (conclusively noncausal in this example).

Touchstone Calibration Wizard

Use the Ansys Calibration Wizard to characterize the effect of a probe or fixture, and remove the effect from a device under test, by reading and writing Touchstone files. There are three methods of calibration: Two-line, Thru-reflect-line, and Short-open-load-thru. Note that for measured data, it is often desirable to smooth the data before calibrating the fixtures.

Smoothing can be accomplished using the Network Data Explorer.

The topics for this section include:

Two-Line Method

Thru-Reflect-Line Method

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Short-Open-Load-Thru Method

Two-Line Method

To implement the Two-Line method, select **Tools** > **Calibration Wizard** to open the **Calibration Wizard** dialog box, then select **Two-line method** from the **Type** drop-down list. The two-line calibration method removes the effect of the probe fixtures from a device under test and assumes that the effect of fixture 1 is identical to fixture 2.

> **Note:** The two line method assumes the probe discontinuity can be represented as a simple shunt element. A larger probe will generate proportionally larger errors in calibration. Tolerance is at the user's discretion.

> **Note:** There can be only one propagating mode on the transmission line. If there are multiple propogating modes, then calibration may fail.

The required input is:

- Two Touchstone files (***.s2p**), each of a thru-line of a given length plus the probe fixtures. The lines must be of different lengths.
- The imported Zo uses Touchstone file (***.s2p**) of an ideal line with no fixture effects. This Touchstone file may come from measurement or simulation. If no file is specified, the wizard will approximate the Zo.
- A Touchstone file (***.s2p**) of the desired device under test with fixture effects.

The output is:

- A Touchstone file of the fixture.
- A Touchstone file of the device under test with effect of the fixtures removed.
- If you select **Export Zo**, the frequency dependent Zo of the calibration lines writes to the designated file.

Thru-Reflect-Line Method

To implement the Thru-Reflect-Line method (TRL), select **Tools** > **Calibration Wizard** to open the **Calibration Wizard** dialog box, then select **Thru-Reflect-Line method (TRL)** from the **Type** drop-down list. The Thru-reflect-line calibration method removes the effect of the probe fixtures from a device under test. Probe 1 and probe 2 are not assumed to identical.

The required input is:

- Two Touchstone files (***.s2p**), each of a thru-line of a given length plus the probe fixtures. The lines must be of different lengths.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- For each port, you can specify either a single-ended (***.s1p**) or a combined measurement (***.s2p**) file. When a ***.s2p** file is specified, Port 1 will use the S11 values and Port2 will use the S22 values:
  - For a short or matched measurement, the surplus inductance must be specified.
  - For an open circuit reflection, the excess capacitance must be specified.
  - If no value is given, an ideal reflect is assumed.
- The imported Zo uses Touchstone file (***.s2p**) of an ideal line with no fixture effects. This Touchstone file may come from measurement or simulation. If no file is specified, the wizard will approximate the Zo.
- A Touchstone file (***.s2p**) of the desired device under test with fixture effects.

The output is:

- Touchstone files for the fixtures corresponding to Port1 and Port2.
- A Touchstone file of the device under test with effect of the fixtures removed.
- If you select **Export Zo**, the frequency dependent Zo of the calibration lines writes to the designated file.

Short-Open-Load-Thru Method

To implement the Short-Open-Load-Thru method (SOLT), select **Tools** > **Calibration Wizard** to open the **Calibration Wizard** dialog box, then select **Short-Open-Load-Thru method** from the **Type** drop-down list. The Short-Open-Load-Thru calibration method removes the effect of the probe discontinuities from a device under test and does not assume the port 1 and port 2 are equal.

The required input is:

- One Touchstone file (***.s2p**) of a thru-line of any length.
- Three Touchstone files for each port corresponding to each measurement configuration: shorted, open and loaded. Each file can either be a single-ended (***.s1p**) or a combined measurement (***.s2p**) file. When a ***.s2p** file is specified, Port 1 will use the S11 values and Port2 will use the S22 values:
  - For a short or matched measurement, the surplus inductance must be specified.
  - For an open circuit reflection, the excess capacitance must be specified.
  - If no value is given, an ideal reflect is assumed.
- A Touchstone file (***.s2p**) of the desired device under test with fixture effects.

The output is:

- Touchstone files for the fixtures corresponding to Port1 and Port2.
- A Touchstone file of the device under test with effect of the fixtures removed.
- If you select **Export Zo**, the frequency dependent Zo of the calibration lines writes to the designated file.

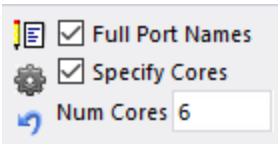
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Multithreading

By default, multithreading (execution on multiple cores) is enabled for Causality Checking and Macro Model export. Multithreading saves significant time in the Causality Check calculation, and improves the time for other state-space fitting operations. See ***Multithreading Technical Notes***.

The **Cores** field in the Network Data Explorer **Control** pane defaults to half the number of cores detected on your computer.

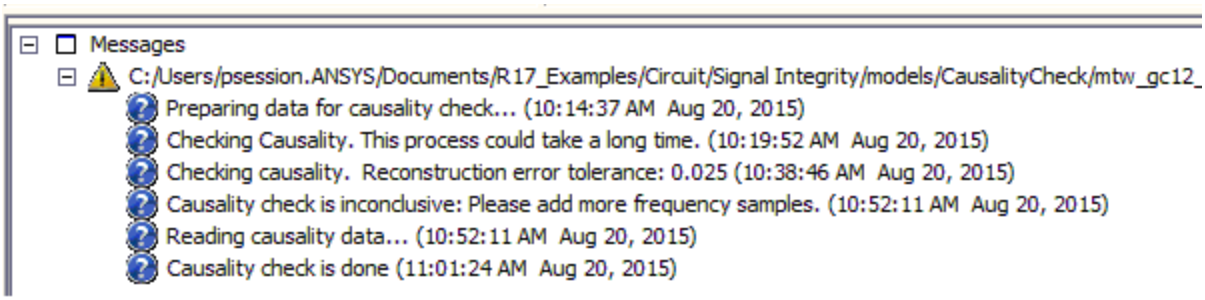
For best performance, disable hyper-threading on your computer. When you enable hyper-threading, the number of cores includes the physical cores and an equal number of logical cores. When you disable hyper-threading, the display shows only half the number of physical cores.



To disable multithreading, clear the check box.

Multithreading Technical Notes

Multithreading can save significant time in the causality check calculation, especially for data sets with large number of ports or a large number of frequencies. The Network Data Explorer contributes a fixed amount of overhead time in preparing the data, and this overhead is not reduced by multithreading. Here are the messages from the causality check of a 278-port Touchstone file using two cores:

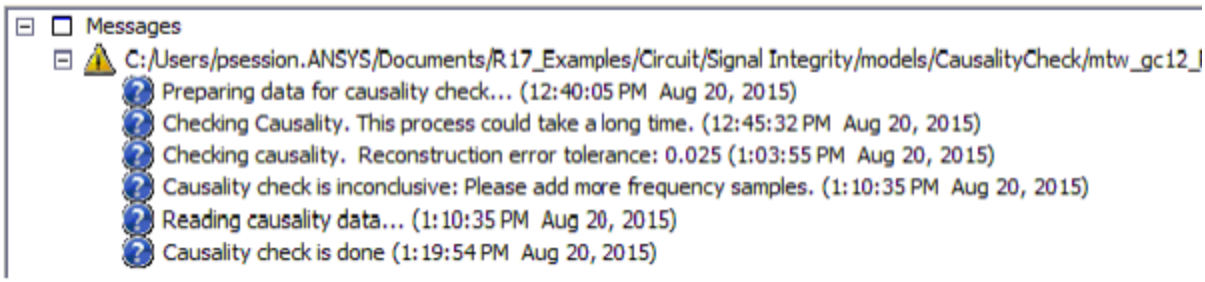


The causality check itself happens between the message Checking causality. Reconstruction error tolerance: 0.025 (time 10:38:46) and the message Causality check is inconclusive: Please add more frequency samples (time 10:52:11). The difference between these two times is the time for the causality check after setup, 13:25 or thirteen minutes and twenty-five seconds.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

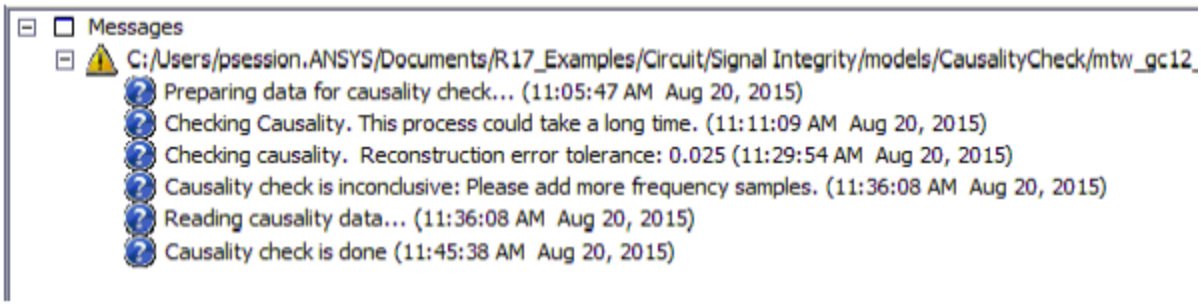
There is a fixed overhead of around 25 minutes involved in the overall time duration for this example.

Here are the causality check messages for the same 278-port file using 8 cores:



Now the causality check happens between times (1:03:55) and (1:10:35), a difference of (6:40) or six minutes and forty seconds, about half the time for the two-core example.

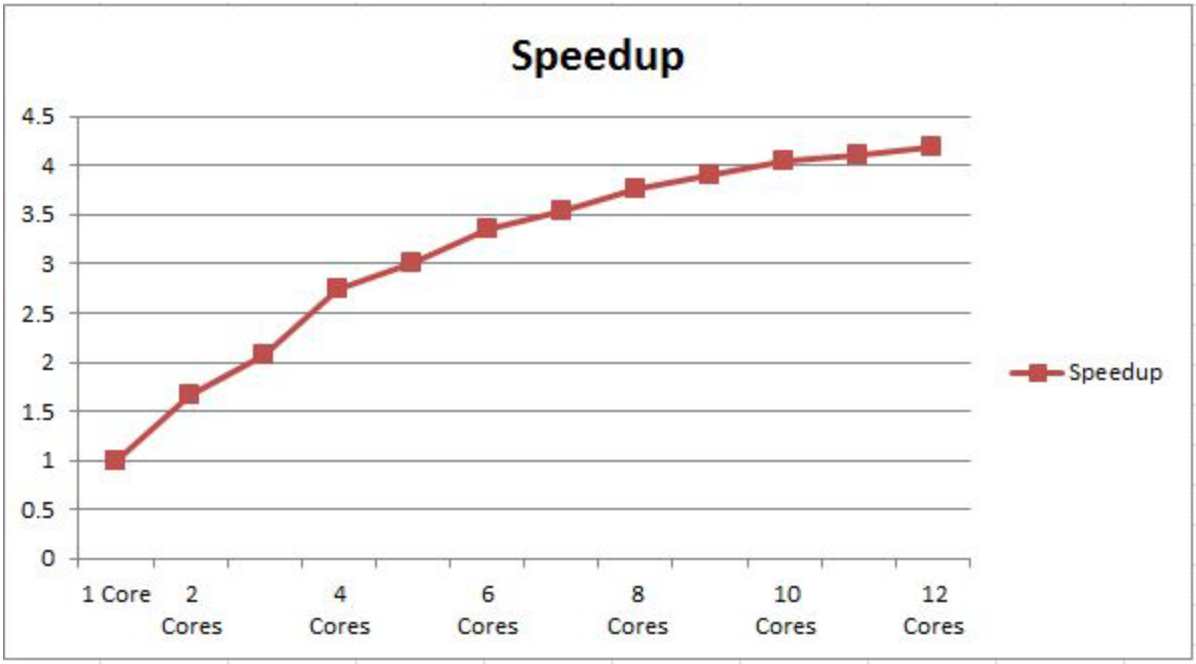
The speedup from multithreading is not linear in practice. Here are the messages from the same 278-port file using 16 cores:



Now the causality check happens between times (11:29:54) and (11:36:08), a difference of (6:14) or six minutes and fourteen seconds, not significantly better than the eight-core performance.

Here is a graph showing the speedup of the causality checker using multithreading on this 278-port file. The plot generates using data from runs with 1, 2, 4, 6, 8, 10, and 12 cores, then averaging the known times to approximate the speedup for 3, 5, 7, 9, and 11 cores. The speedup for N cores is the time with one core divided by the time with N cores.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

19 - Optimetrics

Use Optimetrics to determine the best design variation among a model's possible variations. You create the original model, the *nominal design*, then define the design parameters that vary, which can be nearly any design parameter assigned a numeric value in Twin Builder. For example, you can parameterize the model geometry or material properties. You can then perform the following types of analyses on your nominal Twin Builder design:

Parametric	In a parametric analysis, you define one or more *variable sweep definitions,* each specifying a series of variable values within a range. For example, you can parameterize component values. (See ***Working with Variables*** for more information.) Optimetrics solves the design at each variation. You can then compare the results to determine how each design variation affects the performance of the design. Parametric analyses are often used as precursors to optimization solutions because they help to determine a reasonable range of variable values for the optimization analysis.
Optimization	For an optimization analysis, you identify the cost function and the optimization goal. Optimetrics changes the design parameter values to meet that goal. The cost function can be based on any solution quantity that Twin Builder can compute.
Sensitivity	In a sensitivity analysis, you use Optimetrics to explore the vicinity of the design point to determine the sensitivity of the design to small changes in variables.
Tuning	Tuning lets you change variable values interactively while monitoring the performance of the design. If you want to ensure that tuning does not resolve variations already solved by parametric setup, select **Save Fields Mesh** in the **Options** tab of the Optimetrics setup.
Statistical	In a statistical analysis, you use Optimetrics to determine the distribution of a design's performance, which is caused by a statistical distribution of variable values.
Design of Experiments	Design of experiments involves the creation of a response surface that depicts and predicts how variable values will affect the design.
DesignXplorer	An optimization tool for studying a range of design variations, used with design of experiments.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Sweeping or using a complex variable is not allowed in any optimetrics setup, including optimization, statistical, sensitivity, and tuning setups.

Related Topics

Setting up a Parametric Analysis

Setting up an Optimization Analysis

Setting up a Sensitivity Analysis

Tuning a Variable

Setting up a Statistical Analysis

Parametric Overview

Optimization Overview

Sensitivity Analysis Overview

Statistical Analysis Overview

Tuning Overview

Using Distributed Analysis

Parametric Overview

Run a parametric analysis to simulate several design variations using a single model. You define a series of variable values within a range, or a variable sweep definition, and Twin Builder generates a solution for each design variation. You can then compare the results to determine how each design variation affects the performance of the design.

You can vary design parameters that are assigned a quantity, such as geometry dimensions, material properties, and boundary and excitation properties. (See the help topic for the specific parameter you want to vary.) The number of variations that can be defined in a parametric sweep setup is limited only by your computing resources.

To perform a parametric analysis, first create a nominal design. A nominal design is created like any other design, except that variables are assigned to those aspects of the model you want to change. All variables must be defined before you start the parametric analysis. Although you are not required to solve the nominal design before performing a parametric analysis, doing so helps ensure that the model is set up and operates as intended. Parametric analyses are often used

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

as precursors to optimization analyses because they let you determine a reasonable range of variable values for an optimization analysis.

Related Topic

Setting up a Parametric Analysis

Setting Up a Parametric Analysis

A *parametric setup* specifies all of the design variations that Optimetrics drives Twin Builder to solve. A parametric setup is made up of one or more *variable sweep definitions*, which are a set of variable values within a range for Twin Builder to solve when you run the parametric setup.

You can define more than one parametric setup per design.

> **Note:**
>
> Once you have created a parametric setup, you can copy and paste it, then make changes to the copy, rather than redoing the whole process for minor changes.

To add a parametric setup to a design:

1. Select **Twin Builder** > **Optimetrics Analysis** > **Add Parametric** .
   - Alternatively, right-click **Optimetrics** in the Project tree, and select **Add** > **Parametric**. The **Setup Sweep Analysis** dialog box appears.
2. Add a variable sweep definition.

   > **Note:**
   >
   > Sweeping or using a complex variable is not allowed in any optimetrics setup, including optimization, statistical, sensitivity, and tuning setups.

   After you define a parametric sweep, a shortcut menu becomes available when you right-click the setup name.
3. Use the **Options** tab to save the solution data for solved design variations in the parametric analysis, or to enable/disable use of a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses.

Related Topics

Adding a Variable Sweep Definition

Specifying a Solution Setup for a Parametric Setup

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Using Distributed Analysis

Parametric Overview

Adding a Variable Sweep Definition

A parametric setup is made up of one or more *variable sweep definitions*. A variable sweep definition is a set of variable values within a range that Optimetrics drives Twin Builder to solve when the parametric setup is analyzed. You can add one or more sweep definitions to a parametric setup.

> **Note:**
>
> Sweeping a complex variable is not allowed in any optimetrics setup, including optimization, statistical, sensitivity, and tuning setups.

1. Select **Twin Builder** > **Optimetrics Analysis** > **Add Parametric** .
   - Alternatively, right-click **Optimetrics** in the Project tree and select **Add** > **Parametric**.

   The **Setup Sweep Analysis** dialog box appears.
2. Under the **Sweep Definitions** tab, click **Add**.

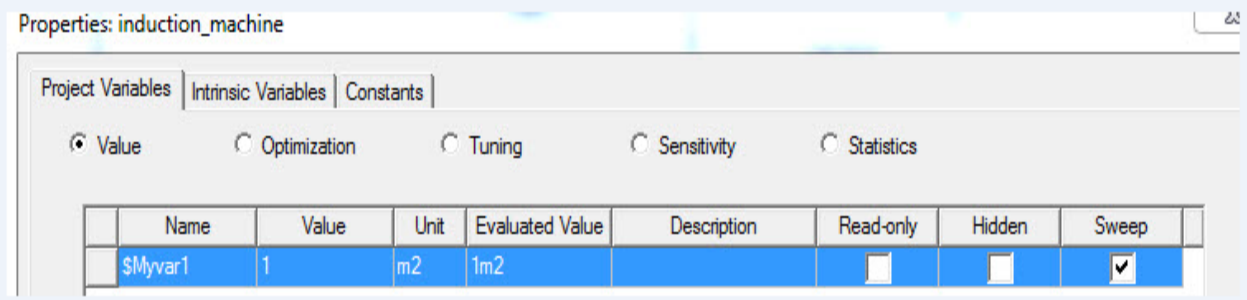
   The **Add/Edit Sweep** dialog box appears.

   All the independent variables associated with the design are listed in the **Variable** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

An *independent* variable is one that does not refer to another variable in its definition. When you select its **Sweep** property in the **Properties** dialog box as shown below, the variable appears in the **Variable** drop-down list. The variable's **Sweep** property is selected by default.



3. Click the variable for which you are defining the sweep definition from the **Variable** drop-down list of the **Add/Edit Sweep** dialog box.

   If you do not define a sweep definition for a variable in the list, the variable's current value in the nominal design is used in the parametric analysis.

4. Specify the variable values to be included in the sweep.
5. Click **Add**, then click **OK**.

   You return to the **Setup Sweep Analysis** dialog box. The variable sweep appears in the top half of the window.

6. View the design variations to be solved in table format under the **Table** tab. Viewing the sweep definition in table format lets you visualize the design variations to be solved and manually adjust sweep points if necessary.
7. Optionally, click **HPC and Analysis Options**, which lets you select or create an analysis configuration.
8. Click **OK**.

Related Topics

Specifying Variable Values for a Sweep Definitions

Synchronizing Variable Sweep Definitions

Modifying a Variable Sweep Definition Manually

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Overriding a Variable's Current Value in a Parametric Setup

Specifying Variable Values for a Sweep Definition

Follow this procedure to specify the variable values to include in a sweep definition:

1. Select one of the following in the **Add/Edit Sweep** dialog box:

Single value	Specify a single value for the sweep definition.
Linear step	Specify a linear range of values with a constant step size.
Linear count	Specify a linear range of values and the number, or count of points within this range.
Decade count	Specify a logarithmic (base 10) series of values, and the number of values to calculate in each decade.
Octave count	Specify a logarithmic (base 2) series of values, and the number of values to calculate in each octave.
Exponential count	Specify an exponential (base e) series of values, and the number of values to calculate.

2. If you selected **Single value**, type the value of the sweep definition in the **Value** box. If you selected another sweep type, do the following:
   a. Type the starting value of the variable range in the **Start** text box.
   b. Type the final value of the variable range in the **Stop** text box.

> **Warning:**
>
> Variable values must be single real numbers, or expressions that evaluate to single real numbers. Complex numbers cannot be used as the values of variables in any optimetric analysis.

3. If you selected **Linear step** as the sweep type, type the step size in the **Step** box.

The step size is the difference between variable values in the sweep definition. The step size determines the number of design variations between the start and stop values. Twin Builder will solve the model at each step in the specified range, including the start and stop values. The step size can be negative, when the **Stop** value is less than the **Start** value.

If you selected another sweep type, type the number of points, or variable values, in the sweep definition in the **Count** text box. For **Decade count** and **Octave count**, the **Count** value specifies the number of points to calculate in every decade or octave. For **Exponential count**, the **Count** value is the total number of points. The total number of points includes the start and stop values.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Synchronizing Variable Sweep Definitions

Synchronizing Variable Sweep Definitions

By default, variable sweep definitions are nested. Alternatively, you can synchronize the variable sweep definitions if they have the same number of sweep points.

For example, if you synchronize a sweep definition that includes values of 1, 2, and 3 inches with a second sweep definition that includes values of 4, 5, and 6 inches, Twin Builder will solve 3 design variations. The first variation is solved at the variable values of 1 and 4; the second variation is solved at the variable values 2 and 5; and the third variation is solved at the final variable values 3 and 6.

To synchronize variable sweep definitions:

1. Under the **Sweep Definitions** tab of the **Setup Sweep Analysis** dialog box, select the rows containing the sweep definitions you want to synchronize.
2. Click **Sync**.

The synchronized sweeps are given a group number, listed in the **Sync #** column.

Optionally, view the design variations to be solved in table format under the **Table** tab.

Related Topics

Specifying Variable Values for a Sweep Definitions

Modifying a Variable Sweep Definition Manually

You can manually modify the variable values solved for a parametric setup by changing, adding, or deleting existing points in a variable sweep definition under the **Table** tab of the **Setup Sweep Analysis** dialog box.

To manually modify a variable sweep definition:

1. Click the **Table** tab of the **Setup Sweep Analysis** dialog box.

   The design variations Twin Builder solves for the parametric setup are listed in table format.

2. Do one of the following:
   - To modify a variable value, click a value text box in the table and type a new value.
   - To delete a variable value from the sweep definition, click the row you want to delete and click **Delete**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- To add a new variable value to the sweep definition, click **Add** then click in the value text box and type a new value.

  > **Warning:**
  >
  > Variable values must be single real numbers, or expressions that evaluate to single real numbers. Complex numbers cannot be used as the values of variables in any optimetric analysis.

Your modifications are tracked and available for viewing at the bottom of the **Setup Sweep Analysis** dialog box under the **Sweep Definitions** tab. The operations you performed are listed with descriptions.

> **Warning:**
>
> If you modify an original sweep definition using the **Add/Edit Sweep** dialog box after manually modifying its table of design variations, your manual modifications become invalid and are removed. A warning informs you that your manual values are about to become invalid, so you can decide whether to proceed.

Related Topics

Adding a Variable Sweep Definition

Overriding a Variable's Current Value in a Parametric Setup

Overriding a Variable's Current Value in a Parametric Setup

If you choose not to sweep a variable, Twin Builder uses the variable's current value set for the nominal design when it solves the parametric setup. To override the current variable value for a parametric setup:

1. In the **Setup Sweep Analysis** dialog box, click the **General** tab.

   All of the current independent design variable values are listed under **Starting Point**.

2. Click the **Value** box of the variable with the value you want to override for the parametric setup.
3. Type a new value in the **Value** box and press **Enter**.

   The **Override** option is now selected; the value you entered will be used for the parametric setup. For this parametric setup, the new value will override the current value in the nominal design.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Alternatively, you can select the **Override** option first, then type a new variable value in the **Value** box.

4. Optionally, click a new unit in the **Units** box.

To revert to the current variable value, clear the **Override** option.

> **Warning:**
>
> Variable values must be single real numbers, or expressions that evaluate to single real numbers. Complex numbers cannot be used as the values of variables in any optimetric analysis.

Related Topics

Adding a Variable Sweep Definition

Modifying a Variable Sweep Definition Manually

Specifying a Solution Setup for a Parametric Setup

To specify the solution setup that Twin Builder analyzes when it solves a parametric setup:

1. In the **Setup Sweep Analysis** dialog box, click the **General** tab.
2. Select the solution setup you want Twin Builder to use when it solves the parametric setup.

Twin Builder solves the parametric setup using the solution setup you select. If you select more than one, results generate for all selected solution setups.

Related Topics

Specifying the Solution Quantity to Evaluate for Parametric Analysis

Specifying a Solution Quantity's Calculation Range

Specifying the Solution Quantity to Evaluate for Parametric Analysis

When you add a parametric setup, you can identify one or more solution quantities to be presented in the **Post Analysis Display** dialog box. The solution quantities are specified by mathematical expressions composed of basic quantities such as output variables. When you view the results, Twin Builder extracts the solution quantities and lists them in the results table.

1. In the **Setup Sweep Analysis** dialog box, click the **Calculations** tab.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

This displays a table that will show solutions and associated calculations. Below the table are buttons to **Setup Calculations...** and **Delete**.

2. Click **Setup Calculations**.

This displays the **Add/Edit Calculation** dialog box. The dialog box contains panes to set the **Context**, the **Trace** tab for the **Calculation Expression**, and the **Calculation Range** tab for the **Calculation Range**.

Follow the procedure to ***Setup Calculations for Optimetrics.***

3. Click **Add Calculation** to add the expression in the **Add/Edit Calculation** dialog box **Calculation Expression** field to the **Calculations** tab of the **Setup Sweep Analysis** dialog box.
4. Click **Done** to close the **Add/Edit Calculation** dialog box.

Related Topics

Specifying a Solution Quantity's Calculation Range

Setup Calculations for Optimetrics.

Setup Calculations for Optimetrics

The **Setup** dialog boxes for each of the Optimetrics types include a **Setup Calculations** button. Click **Setup Calculations** to open the **Add/Edit Calculation** dialog box. The dialog box contains distinct panes and tabs to set the **Context**, the **Calculation Expression**, and the **Calculation Range**.

The **Context** pane contains fields for the report type to use, the solution, and the domain.

The **Trace** tab contains fields for the calculation expression, and, to build the expression, a category list, a quantity list with a text filter field, and a list of functions available for the selected category. The function button opens a dialog box in which you can define a range function to apply a function to the expression.

The category list for the **Trace** tab includes variables and output variables. Click **Output Variables** to open a dialog box to define and edit the output variables.

To set up an Optimetrics calculation:

1. Click **Setup Calculations** to open the **Add/Edit/Calculation** dialog box.
2. In the **Report Type** text field in the **Context** pane, select from the drop-down list of available types.
3. In the **Solution** text box, select from the drop-down list of available solutions.
4. In the **Trace** tab, specify the solution category, a quantity, and functions. The resulting expression appears in the **Calculation Expression** field.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

a. Select the **Category** from the list.

The selection appears in the **Calculation Expression** field, and the Quantity and Function fields list what is available for the corresponding selection.

b. Select the **Quantity** from the list.

The selected quantity appears in the **Calculation Expression** field.

If the **Quantity** list is long, you can filter it for easier selection by typing in the text filter field. Only quantities that contain those alphanumeric characters anywhere in their name will remain visible in the list.

If you want to create an output variable that represents the solution quantity, do the following:

- Click **Output Variables**. The **Output Variables** dialog box appears.
- Add the expression you want to evaluate and click **Done**. The recently created output variable appears in the **Quantity** list.
- Click a new output variable in the **Quantity** drop-down list.

> **Note:**
>
> The calculation you specify must be able to be evaluated into a single, real number.

The selected quantity appears in the **Calculation Expression** field.

c. Select the **Function** from the list. The selected function is applied to the **Quantity** in the **Calculation Expression** field.

5. To apply a **Range function** to the **Calculation Expression**, see ***Setting a Range Function***.
6. Click **Add Calculation** to add the expression in the **Add/Edit Calculation** dialog box **Calculation Expression** field to the **Calculations** tab of the **Setup Sweep Analysis** dialog box.
7. Click **Done** to close the **Add/Edit Calculation** dialog box.

Related Topics

Specifying a Solution Quantity to Evaluate

Setting a Range Function

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Specifying a Solution Quantity's Calculation Range

The calculation range of a solution quantity determines the value of intrinsic variables such as frequency (Freq) at which the solution quantity is extracted. For a parametric setup, the calculation range must be a single value.

1. In the **Setup Sweep Analysis** dialog box, click the **Calculations** tab.
2. Click **Setup Calculations**. The **Add/Edit/Calculation** dialog box appears.
3. Select the **Calculation Range** tab.
4. In the **Variable** list, click an intrinsic variable. **Single Value** is selected by default.
5. In the **Value** box, click a value at which the solution quantity is extracted.
6. Click **Update** and click **Edit**.

Viewing Results for Parametric Solution Quantities

1. In the Project tree, right-click the parametric setup for which you want to view the results calculated for the solution quantities, and select **View Analysis Result**. The **Post Analysis Display** dialog box appears.
2. Select the parametric setup with the results to view from the drop-down list at the top of the dialog box.
3. Select the **Result** tab.
4. To view the results in tabular form, select **Table** as the view type.

The results for the selected solution quantities appear in table format for each solved design variation.

5. Optionally, select **Show complete output name**.

The complete name of the solution for which the results are being displayed appears in the column headings.

6. Optionally, click a design variation in the table and click **Apply** (at the far right side of the dialog box).

Twin Builder now points to the selected design variation as the nominal solution and as a result, the design changes to represent the selected design variation.

Click **Revert** to return the design in the view window to the original value.

7. To view the results in graphic form, select **Plot** as the view type.
8. Select the variable with the swept values to plot on the X-axis from the **X** drop-down list.
9. Only one sweep variable at a time can be plotted against solution quantity results. Any other variables swept during the parametric analysis remain constant.

Optionally, to modify the constant values of other swept variables:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

a. Click **Set Other Sweep Variables Value**.

The **Setup Plot** dialog box appears. All of the other solved variable values are listed.

b. Click the row with the variable value to use as the constant value in the plot and click **OK**.

10. Select the solution quantity results to plot on the Y-axis from the **Y** drop-down list.

The x -y plot appears in the view window.

Right-click the graph area to modify the display. See Reports for details on these operations.

11. To view profile information about the analysis, click the **Profile** tab on the **Post Analysis Display** dialog box.
12. When more than one parametric analysis has run, use the left and right arrows to select a profile.
13. Click **Close** to close the **Post Analysis Display** dialog box.

Related Topics

Plotting Solution Quantity Results vs. a Swept Variable

Viewing Solution Data for an Optimetrics Design Variation

Adding a Parametric Sweep from a File

You can specify the parameters for a parametric sweep in a spreadsheet that uses a **.csv** (comma delimited) or **.txt** (tab delimited) format. You can then import the parametric sweep using **Twin Builder** > **Optimetrics Analysis** > **Add Parametric from File**.

For example, a **.txt** spreadsheet file could resemble the following:

a $b $c[in] d[m] $e $f

0.1 mil 2mm 11 21 0.6in 8

0.2mil 3mm 1.3 2.6mm 3 9cm

...

The first row lists the project and design variable names, and when followed by parentheses or square brackets, the units. The following rows provide the variable values and units. Project or design variables must be defined before they are accepted from a file. The characters in variable names are not case sensitive. Consecutive separators, tabs for **.txt** files, or commas for **.csv** files, are treated as one separator.

Related Topic

Setting up a Parametric Analysis

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Optimization Overview

Optimetrics works with Ansys Electromagnetics products to optimize a wide variety of design parameters based on variable geometry, materials, excitations, component values, and so on. Optimization is the process of locating the minimum of a user-defined cost function. Optimetrics modifies the variable values until the minimum is reached with acceptable accuracy.

Related Topics

Setting Up an Optimization Analysis

Choosing an Optimizer

Choosing an Optimizer

Conducting an optimization analysis lets you determine an optimum solution for your problem. In Twin Builder optimization analyses, you have the choice of different optimizers, though in most cases, we recommend the Sequential Nonlinear Programming optimizer.

- Sequential Nonlinear Programming (Gradient) (SNLP)
- Merit-based Sequential Quadratic Programming (Gradient) (MBSQ)
- Sequential Mixed Integer NonLinear Programming (Gradient and Discrete) (SMINLP)
- Quasi-Newton (Gradient)
- Pattern Search (Search-based)
- Genetic Algorithm (Random search)
- MATLAB Optimizer

Additional Optimizers

These use a decision support process (DSP) based on satisfying criteria as applied to the parameter attributes using a weighted aggregate method. In effect, the DSP is a postprocessing action on the Pareto fronts as generated from the results of the various optimization methods.

- **Screening (Search based)** – This is a non-iterative direct sampling method that uses a quasi-random number generator based on the Hammersley algorithm. You can start with Screening to locate the multiple tentative optima, then refine with NLPQL or MISQP to zoom in on the individual local maximum or minimum value. Usually Screening is used for preliminary design, which can lead you to apply one of the other approaches for more refined optimization results.
- **Multi-Objective Genetic Algorithm** – This iterative random search algorithm optimizes problems with continuous input parameters. It is better for calculating the global optima. You can start with MOGA to locate the multiple tentative optima, then refine with NLPQL or MISQP to zoom in on the individual local maximum or minimum value.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Nonliner Programming by Quadratic Lagrangian (Gradient)** – This is a gradient-based, single-objective optimizer based on quasi-Newton methods. Ideally suited for local optimization.
- **Mixed-Integer Sequential Quadratic Programming (Gradient and Discrete)** – This is a gradient-based, single-objective optimizer that solves mixed-integer non-linear programming problems by a modified sequential quadratic programming (SQP) method. Ideally suited for local optimization.
- **Adaptive Multiple Objective (Gradient)** – This is an iterative, multi-objective optimizer that employs a Kriging response surface and MOGA. In this method, the use of a Kriging response surface allows for a more rapid optimization process because all design points are not evaluated except when necessary and part of the population is simulated by evaluations of the Kriging response surface, which is constructed of all design points submitted by Multi-Objective Genetic Algorithm (MOGA).
- **Adaptive Single Objective (Gradient)** – This is a gradient-based, single-objective optimizer that employs an OSF (Optimal Space-Filling) DOE, a Kriging response surface, and MISQP.

All optimizers assume that the nominal problem you are analyzing is close to the optimal solution; therefore, you must specify a domain that contains the region in which you expect to reach the optimum value.

All optimizers let you define a maximum limit to the number of iterations to be executed. This prevents you from consuming your remaining computing resources and lets you analyze the obtained solutions. From this reduced range, you can further narrow the domain of the problem and regenerate the solutions.

All optimizers also let you enter a coefficient in the **Add Constraints** dialog box to define the linear relationship between the selected variables and the entered constraint value. For the SNLP and NMINLP optimizers, the relationship can be linear or nonlinear. For the quasi-Newton (Gradient) and Pattern Search (Search-based) optimizers, the relationship must be linear.

Cost functions can be quite nonlinear. As a result, during the function evaluations of the algorithm, the cost function can vary significantly. It is important to understand the relationship between optimization function evaluation and iteration. Every iteration, depending on the number of parameters to be optimized, performs several function evaluations. These function evaluations, depending on how nonlinear the cost function is, could show drastic changes. The presence of drastic changes has no bearing on whether the optimization algorithm converged or not.

In the case of non-gradient search-based optimization algorithms, such as "pattern search," which are entirely based on function evaluations, one could see drastic changes in the function evaluations depending on how nonlinear the cost function is. This could seem misleading as if the algorithm did not converge since in theory one expects the cost function to decrease from one iteration to the next. The optimetrics, however, reports function evaluations and not necessarily the optimizer performance per iteration.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

The MATLAB optimizer displays function evaluation when you select the **Show all functions evaluation** check box. If you clear the check box, it displays iteration.

Quasi-Newton (Gradient)

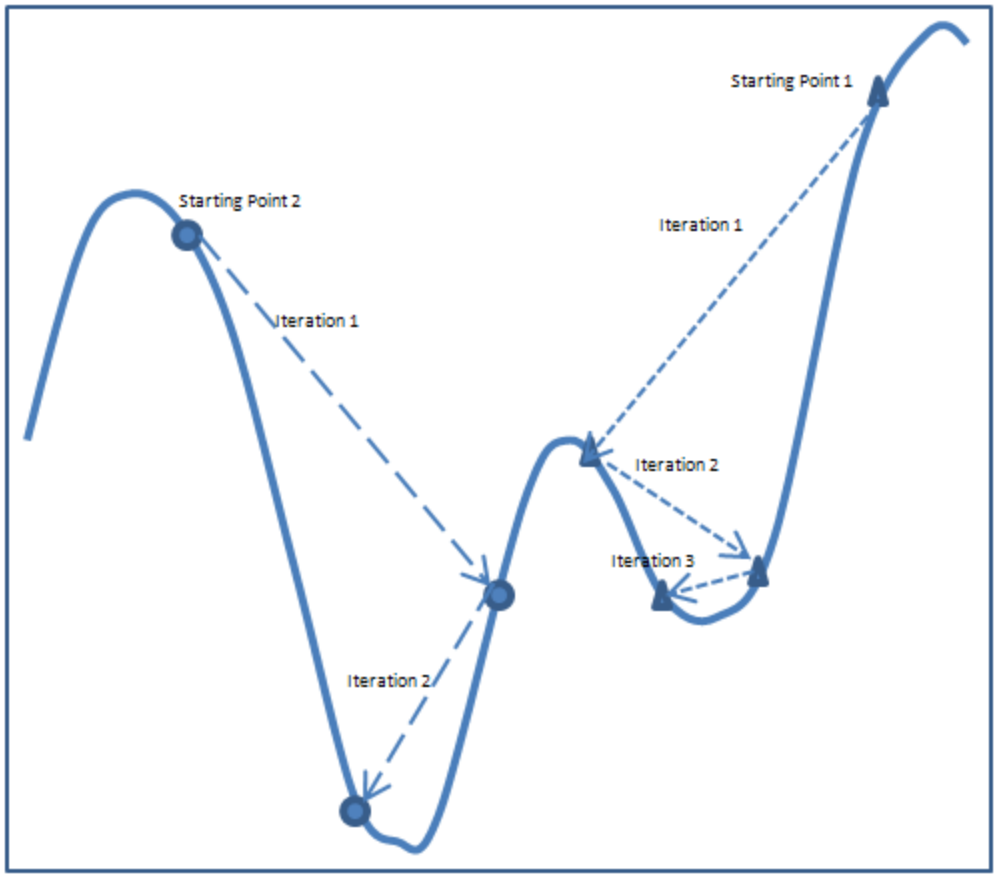
If the Sequential Non Linear Programming Optimizer has difficulty, and if the numerical noise is insignificant during the solution process, use the quasi-Newton optimizer to obtain the results. The quasi-Newton optimizer works on the basis of finding a minimum or maximum of a cost function which relates variables in the model or circuit to overall simulation goals. You define one or more variables in the problem definition and a cost function in the optimization setup. The cost function relates the variable values to field quantities, design parameters like force or torque, power loss, and so on. The optimizer can then maximize or minimize the value of the design parameter by varying the problem variables.

Sir Isaac Newton first showed that the maximum or minimum of any function can be determined by setting the derivative of a function with respect to a variable (x) to zero and solving for the variable. This approach leads to the exact solution for quadratic functions. However, for higher order functions or numerical analysis, an iterative approach is commonly taken. The function is approximately locally by a quadratic and the approximation is solved for the value of x. This value is placed back into the original function and used to calculate a gradient which provides a step direction and size for determining the next best value of x in the iteration process.

In the quasi-Newton optimization procedure, the gradients (Hessian) are not well behaved functions and are calculated numerically. Essentially, the change in the estimate of x and the change in the gradient are used to estimate the Hessian for the next iteration. The ratio of the change in the gradients to the change in the values of x provides the Hessian for the next step and is known as the quasi-Newton condition. In order to perform the quasi-Newton optimization, at least three solutions are required for each parameter being varied. This can have a significant computational cost depending upon the type of analysis being performed.

There are numerous methods described in the literature for solving for the Hessian and the details of the method used by Optimetrics are beyond the scope of this document. However, as the quasi-Newton method is, at its heart, a gradient method, it suffers from two fundamental problems common to optimization. The first is the possible presence of local minima. The following figure illustrates the problem of local minima.

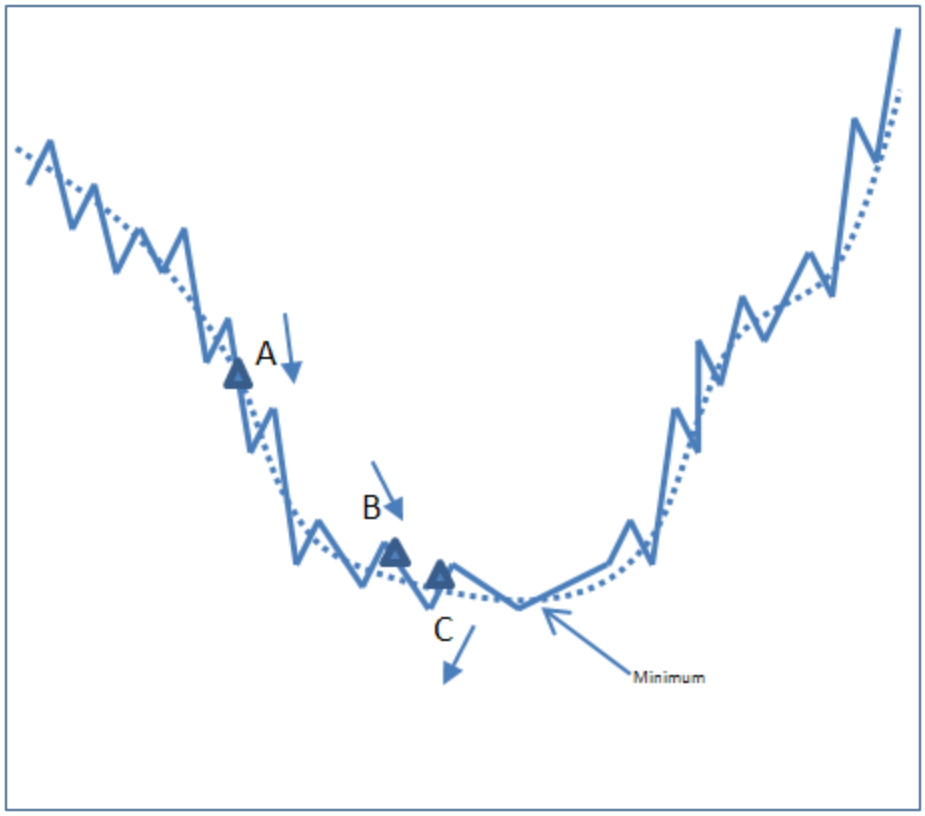
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



In this scenario, you can see that in order to find the minimum of the function over the domain, a number of factors will determine the overall success including the initial starting point, the initial set of gradients calculated, the allowable step size, and so on. Once the optimizer has located a minimum, the quasi-Newton approach will locate the bottom and will not search further for other possible minima. In the example shown, when the optimizer begins at the point labeled **Starting Point 1** the minima it finds is a local minima and not a good global solution to the problem.

The second basic issue with quasi-Newton optimization is numerical noise. In gradient optimization, the derivatives are assumed to be smooth, well behaved functions. However, when the gradients are calculated numerically, the calculation involves taking the differences of numbers that get progressively smaller. At some point, the numerical imprecision in the parameter calculations becomes greater than the differences calculated in the gradients and the solution will oscillate and may never reach convergence. To illustrate this, consider the figure shown below.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



In this scenario, the optimizer is looking for the point labeled **Minimum**. Three possible solutions are labeled **A**, **B**, and **C**, with each arrow indicating the direction of the derivative of the function at that point. If points **A** and **B** represent the last two solution points for the parameter, then it is easy to see that the changes in the magnitude and the consistent direction of the derivatives will serve to push the solution closer to the desired minimum. If, however, points **A** and **C** are the last two solution points respectively, the magnitude indicates the proper direction of movement, but the derivatives are opposite, possibly causing the solution to move away from the minimum, back in the direction of point **A**.

In order to use the quasi-Newton optimizer effectively, the cost function should be based on parameters that exhibit a smooth characteristic (little numerical noise) and a starting point of the optimization should be chosen somewhat close to the expected minimum based on an understanding of the physical problem being optimized. This becomes increasingly difficult, however, when multiple parameters are being varied or when multiple parameters are to be optimized. In addition, the computational burden of multivariate optimization with quasi-Newton increases geometrically with the number of variables being optimized. As a result, this method should only be attempted when 1 or 2 variables are being optimized at a time.

For more information regarding quasi-Newton optimization methods, see:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Schoenberg, Ronald. *Optimization with the Quasi-Newton Method*. Aptech Systems, Inc. 2001.

Related Topics

Optimization Setup for Quasi-Newton (Gradient) Optimizer

Pattern Search (Search-based)

If the noise is significant in the nominal project, use the Pattern Search optimizer to obtain the results. It performs a grid-based simplex search, which makes use of simplices: triangles in 2D space or tetrahedra in 3D space. A simplex is a Euclidean geometric spatial element having the minimum number of boundary points, such as a line segment in one-dimensional space, a triangle in two-dimensional space, or a tetrahedron in three-dimensional space.

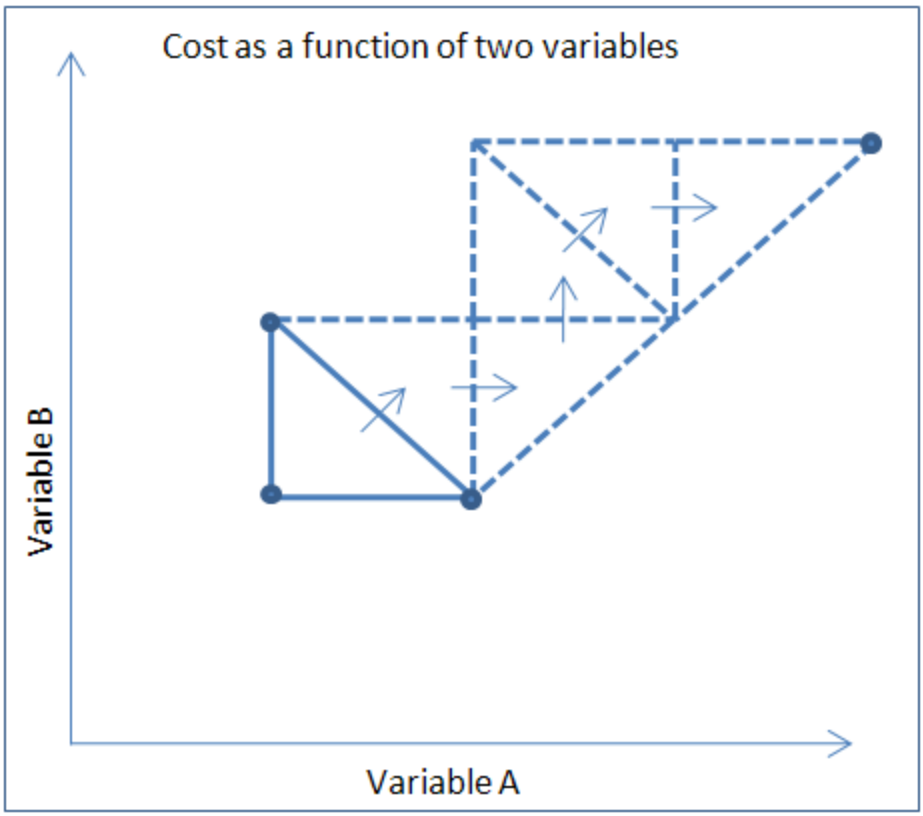
The cost value is calculated at the vertices of the simplex. The optimizer mirrors the simplex across one of its faces based on mathematical guidelines and determines if the new simplex provides better results. If it does not produce a better result, the next face is used for mirroring and the pattern continues. If no improvement occurs, the grid is refined. If improvement occurs, the step is accepted and the new simplex is generated to replace the original one. The figures below illustrates a triangular simplex mirrored several times to demonstrate the pattern search approach in two variables and the simplices superimposed on a 2D cost function to demonstrate the convergence toward a minimum in the cost function.

Cost functions can be quite nonlinear. As a result, during the function evaluations of the algorithm, the cost function can vary significantly. Also, it is important to understand the relationship between optimization function evaluation and iteration. Every iteration, depending on the number of parameters to be optimized, performs several function evaluations. These function evaluations could show drastic changes, depending on how nonlinear the cost function is. The presence of drastic changes has no bearing on whether the optimization algorithm converged.

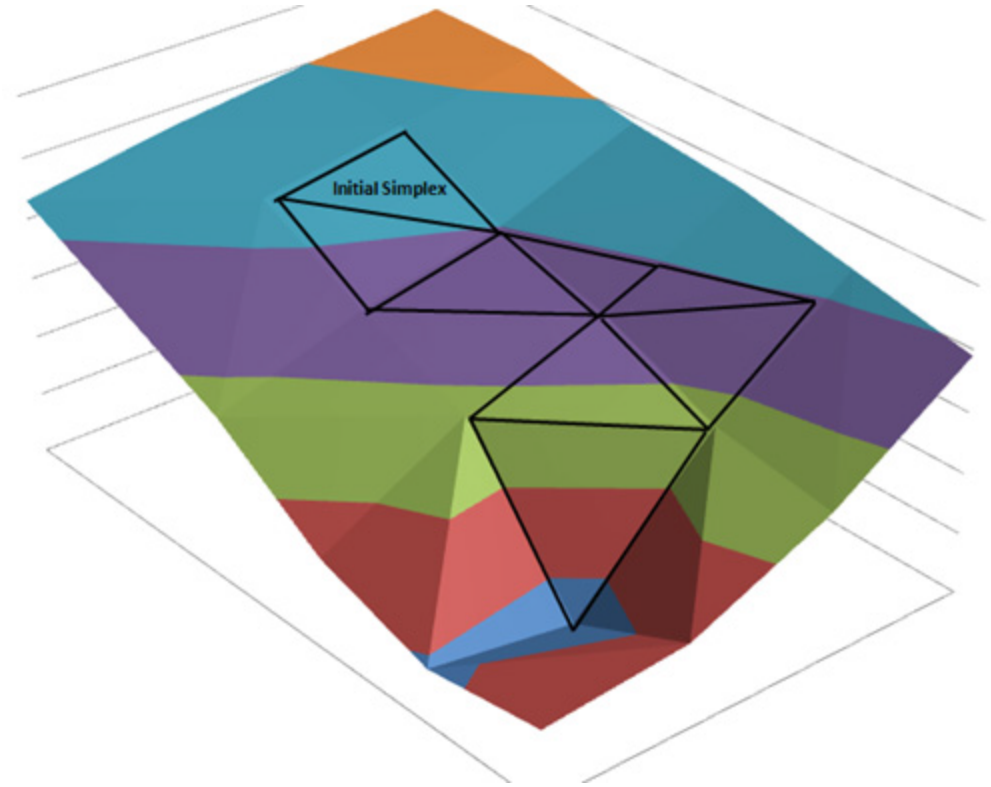
In the case of non-gradient search-based optimization algorithms, such as pattern search, which are entirely based on function evaluations, you could see drastic changes in the function evaluations depending on how nonlinear the cost function is. This could seem misleading as if the algorithm did not converge since in theory one expects the cost function to decrease from one iteration to the next. The optimetrics, however, report function evaluations and not necessarily the optimizer performance per iteration.

> **Note:**
>
> The MATLAB optimizer displays function evaluation when you select the **Show all functions evaluation** check box. Clear the check box to display iteration.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The Pattern Search algorithms are extensible to three variable optimization by using tetrahedral simplices, however, they are not easily represented in graphical form. Generally, Pattern Search algorithms are not used when more than three variables are used in the optimization.

When there is not improvement in the cost function regardless of the direction the simplex is mirrored, then the simplex is subdivided into smaller simplices and the process restarted.

Pattern Search algorithms have several advantages over quasi-Newton algorithms. First, they are less sensitive to noise because the cost function is evaluated at all node points on the simplex and the numerical noise averages out over the simplex. The second advantage is that the number of initial solutions is generally smaller as shown in the table. However, since the pattern search does not use gradient information to locate the minimum the process converges more slowly toward the true minimum, taking more steps to successively divide the simplices as the minimum is approached.

Related Topics

Optimization Setup for Pattern Search(Search-based) Optimizer

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Sequential Non-linear Programming (Gradient) (SNLP)

The main advantage of SNLP over quasi-Newton is that it handles the optimization problem in more depth. This optimizer assumes that the optimization variables span a continuous space. As a result, there is no Minimum Step Size specified in this optimizer and the variables may take any value within the allowable constraints and within the numerical precision limits of the simulator. Like quasi-Newton, the SNLP optimizer assumes that the noise is not significant. It does reduce the effect of the noise, but the noise filtering is not strong.

The SNLP optimizer approximates the FEA characterization with Response Surfaces (RS). With the FEA-approximation and with light evaluation of the cost function, SNLP has a good approximation of the cost function in terms of the optimization variables. This approximation lets the SNLP optimizer estimate the location of improving points. The overall cost approximations are more accurate. This allows the SNLP optimizer a faster practical convergence speed than that of quasi-Newton.

The SNLP optimizer creates the response surface using a Taylor Series approximation from the FEA simulation results available from past solutions. The response surface is most accurate in the local vicinity. The response surface is used in the optimization loop to determine the gradients and calculate the next step direction and distance. The response surface acts as a surrogate for the FEA simulation, reducing the number of FEA simulations required and greatly speeding the problem. Convergence improves as more FEA solutions are created and the response surface approximation improves.

The SNLP method is similar to the Sequential Quadratic Programming (SQP) method in two ways. Both are sequential, updating the optimizer state to the current optimal values and iterating. Sequential optimization can be thought of a walking a path, step by step, toward an optimal goal. SNLP and SQP optimizers are also similar in that both use local and inexpensive surrogates. However, in the SNLP case, the surrogate can be of a higher order and is more generally constrained. The goal is to achieve a surrogate model that is accurate enough on a wider scale, so that the search procedures are well lead by the surrogate, even for relatively large steps. All functions calculated by the supporting finite element product (for example, Maxwell 3D or HFSS) is assumed to be expensive, while the rest of the cost calculation (for example, an extra user-defined expression) — which is implemented in Optimetrics — is assumed to be inexpensive. For this reason, it makes sense to remove inexpensive evaluations from the finite element problem and, instead, implement them in Optimetrics. This optimizer holds several advantages over the quasi-Newton and Pattern Search optimizers.

Most importantly, due to the separation of expensive and inexpensive evaluations in the cost calculation, the SNLP optimizer is more tightly integrated with the supporting FEA tools. This tight integration provides more insight into the optimization problem, resulting in a significantly faster optimization process. A second advantage is that the SNLP optimizer does not require cost-derivatives to be approximated, protecting against uncertainties (noise)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

in cost evaluations. In addition to derivative-free state of the RS-based SNLP, the RS technique also proves to have noise suppression properties.

Related Topics

Optimization Setup for Sequential Non-linear Programming (Gradient) Optimizer

Sequential Mixed Integer NonLinear Programming (Gradient and Discrete)

The Sequential Mixed Integer Nonlinear Programming (SMINLP) optimizer is equivalent to the SNLP optimizer with one difference. Many problems require variables take only discrete values. One example might be to optimize on the number of turns in a coil. To optimize on number of turns or quarter turns, the optimizer must handle discrete optimization variables. The SMINLP optimizer can mix continuous variables among the integers, or can have only integers, and works if all variables are continuous. The setup resembles the setup for SNLP, except that you must flag the integer variables.supporting integer variables. You can set up internal variables based on the integer optimization variable.

For example, consider N to be an integer optimization variable. By definition it can only assume integer values. You can establish another variable, which further depends on this one: K = 2.345 * N, or K = sin(30 * N). This way K has a discrete value, but is not necessarily integer. Or, you can use N directly as a design parameter.

Related Topic

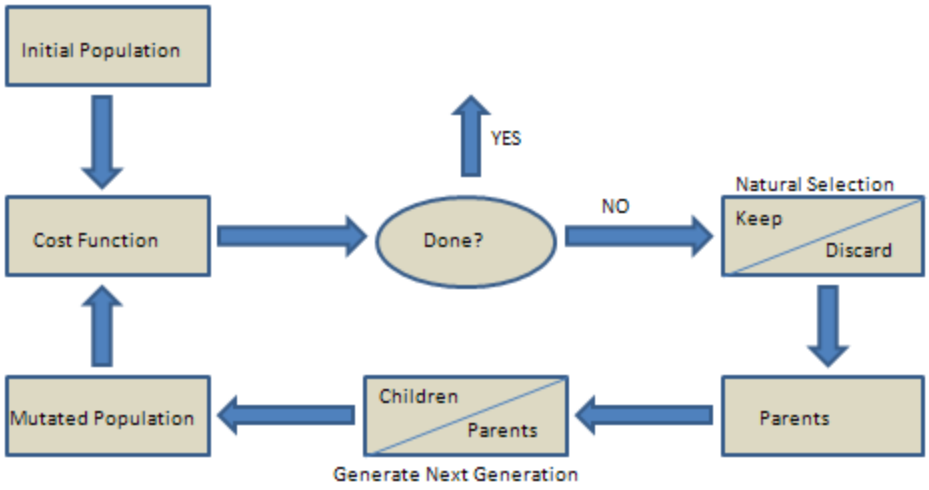
Optimization Setup for Sequential Mixed Integer NonLinear Programming (Gradient and Discrete) Optimizer

Genetic Algorithm (Random search)

Genetic Algorithm (GA) optimizers are part of a class of optimization techniques called stochastic optimizers. They do not use the information from the experiment or the cost function to determine where to further explore the design space. Instead, they use a type of random selection and apply it in a structured manner. The random selection of evaluations to proceed to the next generation has the advantage of allowing the optimizer to jump out of a local minima at the expense of many random solutions which do not provide improvement toward the optimization goal. As a result, the GA optimizer will run many more iterations and may be prohibitively slow.

The Genetic Algorithm search is an iterative process that goes through a number of generations (see image below). In each generation some new individuals (Children / Number of Individuals) are created and the so grown population participates in a selection (natural-selection) process that in turn reduces the size of the population to a desired level (Next Generation / Number of Individuals).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



When creating a smaller set of individuals from a bigger set, the GA selects individuals from the original set. During this process, better fit (in relation to the cost function) individuals are preferred. In the elitist selection, the best so many individuals are selected, but if you turn on the roulette selection, then the selection process gets relaxed. An iterative process starts selecting the individuals and fill up the resulting set, but instead of selecting the best so many, we use a roulette wheel that has for each selection-candidate divisions made proportional to the fitness level (relative to the cost function) of the candidate. This means that the fitter the individual is, the larger the probability of survival will be.

Related Topics

Optimization Setup for Genetic Algorithm (Random search) Optimizer

Optimization Variables in Design Space

Cost Function

Advanced Genetic Algorithm Optimizer Options

MATLAB optimizer

The MATLAB optimizer option lets you pass a script to MATLAB to perform the optimization. When optimization starts, MATLAB launches and a script passes in to MATLAB to perform the optimization. During optimization, MATLAB calls back into Twin Builder to perform the solve and compute the cost. The cost is reported back to MATLAB, and MATLAB's optimization then determines the next step in the optimization.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The optimization script is specified as part of the optimization setup. By modifying the optimization script, you can change optimization parameters and optimization method as well as use the full power of MATLAB.

Running the Optimization

A MATLAB optimization launches like any other optimization. The **Message Manager** pane displays status messages when MATLAB launches, and status messages generate for each solve being performed.

In most cases, MATLAB terminates when the optimization completes. Some reasons why MATLAB may fail to terminate include:

- You have modified the MATLAB script to not terminate MATLAB after the optimization.
- A syntax error or some other error has occurred.
- You have added some other code, which runs after the optimization completes.

System Requirements

In order to use MATLAB to perform optimizations from your application:

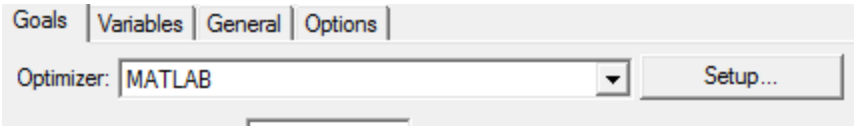
- A version of MATLAB must be installed on your system.
- The computing platform (that is, 64-bit) of MATLAB must match the platform of the Ansys application you are using it with.
- You must have the MATLAB Optimization Toolkit installed.

Specifying the MATLAB Location

The **Miscellaneous** tab at **Tools** > **Options** > **General Options** contains a setting for the MATLAB application location. This setting must point to the version of MATLAB that will perform the optimization. The platform (64-bit) of the specified version of MATLAB must match the platform of this application.

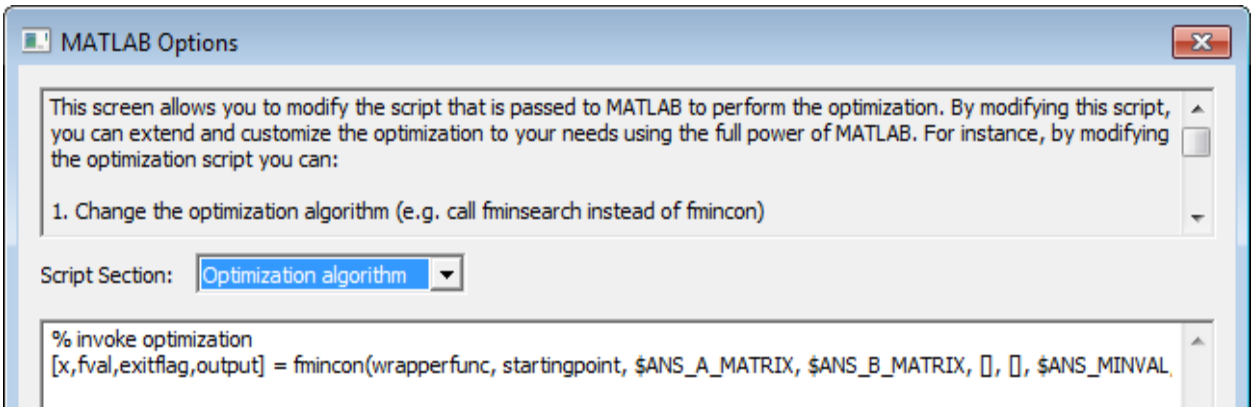
MATLAB Optimization Setup

MATLAB optimization starts by creating an optimization and selecting MATLAB from the optimizer drop-down list. If you select MATLAB as the optimizer, the **Setup Optimization** dialog box displays a **Setup** button.



Click **Setup** to open the **MATLAB Options** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The upper text panel is informative. The **Script Section** drop-down list lets you select a lower panel display for **Optimization algorithm**, **Options**, or the **Full** script template.

This panel lets you modify the script passed to MATLAB to perform the optimization. The complete script contains all the instructions necessary for MATLAB to connect to Twin Builder and perform the optimization. The drop-down selection lets you view only the portion of code of interest without having to view the full script. The choices are:

- **Optimization algorithm** – Displays only the line of code invoking the actual optimization function. By changing this line, you can use a different MATLAB function for optimization. By default **fmincon()** is used, which is a derivative-based constrained optimization. By modifying this line, you could replace the **fmincon()** call with **fminsearch()** to use an unconstrained pattern searching optimizer or another optimization function. See the MATLAB documentation for details about available optimization functions.
- **Options** – Each optimization function contains many options and parameters, which are set in the MATLAB script prior to calling the optimization function. By modifying these options, the optimization can be customized as desired. For instance, options can be set for **fmincon()** to specify the algorithm that it uses internally. See the MATLAB documentation for details about options available for each optimization function.
- **Full script template** – This choice displays the full optimization script passed to MATLAB.

The initial Script Section display for the Optimization algorithm shows the following:

```
% invoke optimization
[x,fval,exitflag,output] = fmincon(wrapperfunc, startingpoint, [],
[], [], [], $ANS_MINVAL, $ANS_MAXVAL, nlcon, options)
```

The initial Script Section Options display shows the following:

```
% customers can add their own options below
options = optimset(options, 'display', 'iter')
options = optimset(options, 'Algorithm', 'interior-point')
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
% options = optimset(options, 'PlotFcns', @optimplotfval)
```

You can modify the script to extend and customize the optimization to your needs. You must ensure that the script follows MATLAB syntax. For instance, by modifying the optimization script you can:

- Change the optimization algorithm (for example, call **fminsearch** instead of **fmincon**).
- Change the parameters/options of the optimization algorithm (see the MATLAB documentation for details).
- Specify a plot function to provide graphical output during optimization.
- Specify a user-defined output function to be called at completion or per iteration.

Symbols

When modifying the MATLAB code, you can use symbols to represent values from the optimization setup. The symbols and their definitions are listed below.

$ANS_VARIABLE_LIST:	List of variables we are optimizing.
$ANS_STARTING_POINT:	Vector of starting values of variables used in the optimization.
$ANS_MAXITERATIONS:	Maximum number of iterations specified in optimization setup.
$ANS_MINVAL:	Vector of minimum values from optimization setup.
$ANS_MAXVAL:	Vector of maximum values from optimization setup.
$ANS_MINSTEP:	Vector of minimum step sizes from optimization setup.
$ANS_MAXSTEP:	Vector of maximum step sizes from optimization setup.
$ANS_A_MATRIX	Matrix of linear constraint coefficients (left-hand side) generated from optimization setup.
$ANS_B_MATRIX	Matrix of linear constraint bounds (right-hand side) generated from optimization setup.

Note:

The linear constraints as generated for MATLAB have the form [A][x] <= [B], where [A] is the coefficient matrix, [x] is the variable list matrix (column vector), and [B] is the bounds matrix (column vector).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> While modifying the script, make sure that the script follows MATLAB syntax.

MATLAB Optimization Script Template

The script template shown in the Script Section is as follows:

```
% make sure platform matches
if strcmp(computer, '$ANS_EXPECTED_PLATFORM') ~= 1
   h = msgbox('32/64 platform does not match calling application,
  exiting')
   uiwait(h)
   exit
end

% add installation dir to search path so .mex file can be found
originalpath = addpath('$ANS_EXEDIR')

% connect back to opticomengine
callbackinterface = optimex('connect', '$ANS_CONNECTIONSTRING')

% set up optimization
% variables are: $ANS_VARIABLELIST
startingpoint = $ANS_STARTINGPOINT
options = optimset('MaxIter', $ANS_MAXITERATIONS)
iterationCallbackWrapper = @(x, optimValues, state) optimex
('notifyiterationcomplete', callbackinterface, x, optimValues.fval,
state)
options = optimset(options, 'OutputFcn', iterationCallbackWrapper)

% halt execution so debugger can be attached
% h = msgbox('attach debugger if desired')
% uiwait(h)

% attributes that user can pass to optimization algorithm
% variables are: $ANS_VARIABLELIST

% this is the objective function which returns cost
wrapperfunc = @(x)optimex('eval', callbackinterface, x)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
% this is our non linear constraint function, returns no constraints
returnempty = @(x)[];
nlcon = @(x) deal(returnempty(x), returnempty(x));

% DO NOT EDIT THIS LINE - START OPTIONS SECTION

% customers can add their own options below
options = optimset(options, 'display', 'iter')
options = optimset(options, 'Algorithm', 'interior-point')
% options = optimset(options, 'PlotFcns', @optimplotfval)

% DO NOT EDIT THIS LINE - END OPTIONS SECTION
% DO NOT EDIT THIS LINE - START OPTIMIZATION ALGO SECTION
% invoke optimization
[x,fval,exitflag,output] = fmincon(wrapperfunc, startingpoint, $ANS_
A_MATRIX, $ANS_B_MATRIX, [], [], $ANS_MINVAL, $ANS_MAXVAL, nlcon,
options)

% DO NOT EDIT THIS LINE - END OPTIMIZATION ALGO SECTION

% write exit message to Ansoft message window
(warning=0,error=1,info=2)
optimex('postansoftmessage', callbackinterface, 2, output.message)

% notify opticomengine that optimization is finished
optimex('optimizationfinished', callbackinterface, exitflag)

% restore original path
path = originalpath

% note: comment below line if you want MATLAB to remain
% running after optimization
exit
```

Related Topics

Optimization Setup for the MATLAB Optimizer

General Options: Miscellaneous

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Screening (Shifted Hammersley Sampling) Optimization

Use the Screening (Shifted Hammersley Sampling) algorithm for sample generation by all methods except NLPQL. The conventional Hammersley sampling algorithm is a quasi-random number generator, which has very low discrepancy and is used for quasi-Monte Carlo simulations.

A low-discrepancy sequence is a sequence of points that approximate the equidistribution in a multi-dimensional cube in an optimal way. This means that the design space is populated almost uniformly by these sequences and that dimensionality is not a problem because of the inherent properties of Monte Carlo sampling. The number of points does not increase exponentially with an increase in the number of input parameters.

The conventional Hammersley sampling algorithm is constructed by using the radical inverse function. Any integer n can be represented as a sequence of digits n_0, n_1, n_2,..., n_m by the following equation:

$$n=n_0\,n_1\,n_2\,n_3\cdots n_m$$

For example, consider the integer 687459, which can be represented this way as n_0=6, n_1=8, and so on. Because this integer is represented with radix 10, you can write it as 687459=9+5*10+4*100 and so on. In general, for a radix R representation, the equation is:

$$n=n_m+n_{m-1}*R+\cdots+n_0$$

The inverse radical function is the function that generates a fraction in (0, 1) by reversing the order of the digits in the previous equation about the decimal point, as shown.

$$\Phi_R(n)=0\,n_m\,n_{m-1}\,n_{m-2}\cdots n_0$$
$$=n_m*R^{-1}+n_{m-1}*R^{-2}+\cdots+n_0*R^{-(m-1)}$$

Thus, for a k-dimensional search space, the Hammersley points are given by the following expression:

$$H_k(i)=[i/N,\,\Phi_{R1}(i),\,\Phi_{R2}(i),\cdots,\,\Phi_{Rk-1}(i)]$$

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Where *i*=0, ..., *N* indicates the sample points. Now, from the plot of these points, you can see that the first row (corresponding to the first sample point) of the Hammersley matrix is zero and the last row is not 1. This implies that, for the *k*-dimensional hypercube, the Hammersley sampler generates a block of points skewed more toward the origin of the cube and away from the far edges and faces. To compensate for this bias, a point-shifting process is proposed that shifts all Hammersley points by the amount that follows:

$$\Delta = \frac{1}{2}N$$

This moves the point set more toward the center of the search space and avoids unnecessary bias. Thus, the initial population always provides unbiased, low-discrepancy coverage of the search space.

Related Topics

Setup Screening (Search Based) Optimizer

Optimization Variables in Design Space

Cost Function

Multi-Objective Genetic Algorithm (MOGA)

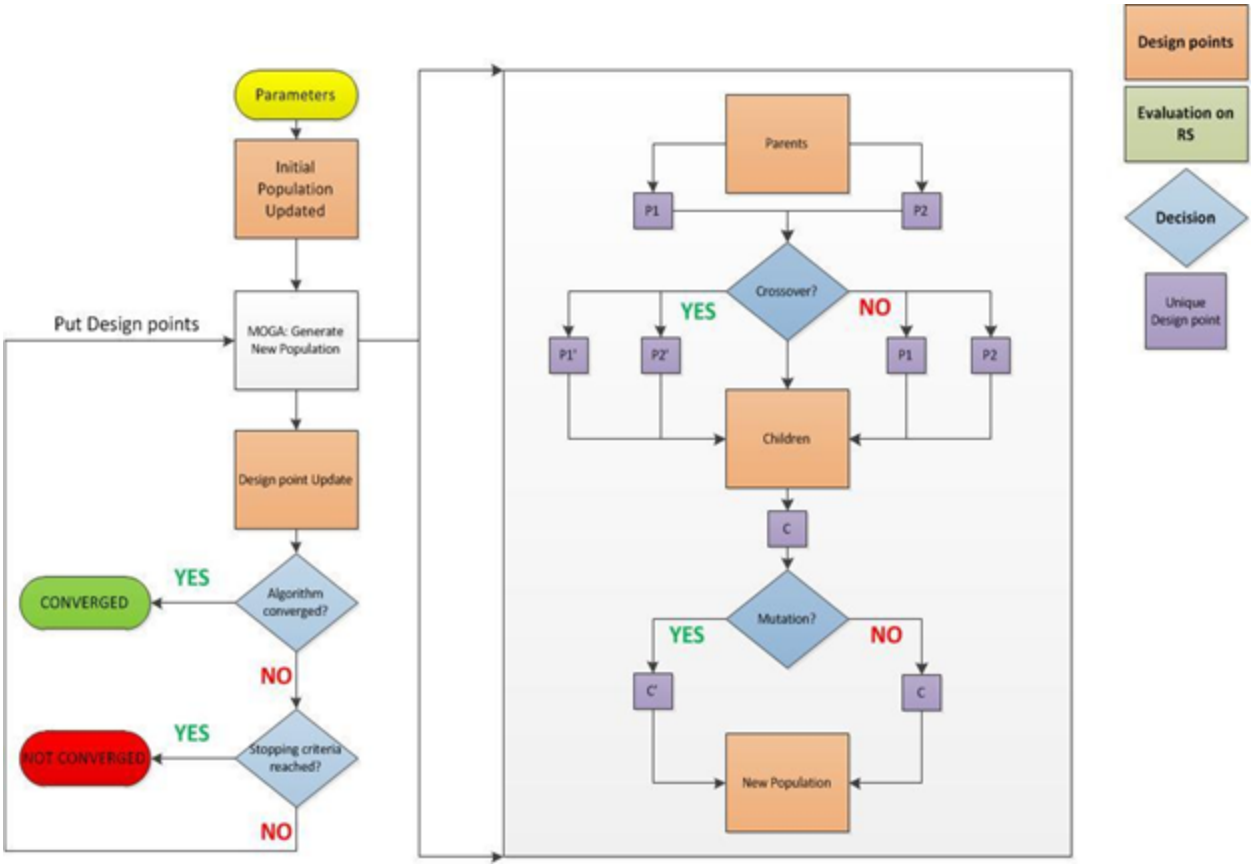
The Multi-Objective Genetic Algorithm (MOGA) used in GDO is a hybrid variant of the popular NSGA-II (Non-dominated Sorted Genetic Algorithm-II) based on controlled elitism concepts. It supports all types of input parameters. The Pareto ranking scheme is done by a fast, non-dominated sorting method that is an order of magnitude faster than traditional Pareto ranking methods. The constraint handling uses the same non-dominance principle as the objectives. Therefore, penalty functions and Lagrange multipliers are not needed. This also ensures that the feasible solutions are always ranked higher than the infeasible solutions.

The first Pareto front solutions are archived in a separate sample set internally and are distinct from the evolving sample set. This ensures minimal disruption of Pareto front patterns already available from earlier iterations. You can control the selection pressure (and, consequently, the elitism of the process) to avoid premature convergence by altering the Maximum Allowable Pareto Percentage property. For more information about this and other MOGA properties, see Setup Multi-Objective Genetic Algorithm.

MOGA Workflow

The MOGA workflow follows:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



MOGA Steps

1. **First Population of MOGA**

   The initial population is used to run MOGA.

2. **MOGA Generates a New Population**

   MOGA is run and generates a new population via cross-over and mutation. After the first iteration, each population is run when it reaches the number of samples defined by the Number of Samples Per Iteration property. See **MOGA Steps to Generate New Population** below.

3. **Design Point Update**

   The design points in the new population are updated.

4. **Convergence Validation**

   The optimization is validated for convergence.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Yes: Optimization Converged**

MOGA converges when the Maximum Allowable Pareto Percentage or the Convergence Stability Percentage has been reached.

- **No: Optimization Not Converged**

  If the optimization is not converged, the process continues to the next step.

5. **Stopping Criteria Validation**

   If the optimization has not converged, it is validated for fulfillment of stopping criteria.

   - **Yes: Stopping Criteria Met**

     When the Maximum Number of Iterations criterion is met, the process stops without having reached convergence.

   - **No: Stopping Criteria Not Met**

     If the stopping criteria have not been met, MOGA runs again to generate a new population (return to Step 2).

6. **Conclusion**

   Steps 2 through 5 are repeated in sequence until the optimization converges or the stopping criteria are met. When one of these things occurs, the optimization concludes.

MOGA Steps to Generate a New Population

The process MOGA uses to generate a new population has two main steps: **Cross-over** and **Mutation**.

1. **Cross-over**

   Cross-over combines (mates) two chromosomes (parents) to produce a new chromosome (offspring). The idea behind cross-over is that the new chromosome can be better than both of the parents if it takes the best characteristics from each of the parents. Cross-over occurs during evolution according to a user-definable cross-over probability.

   - **Cross-over for Continuous Parameters**

A cross-over operator that linearly combines two parent chromosome vectors to produce two new offspring according to the following equations:

$$\text{Offspring1}=\mathbf{a}*\text{Parent1}+\left(1-\mathbf{a}\right)*\text{Parent2}$$

$$\text{Offspring2}=\left(1-\mathbf{a}\right)*\text{Parent1}+\mathbf{a}*\text{Parent2}$$

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Consider the following two parents (each consisting of four floating genes), which have been selected for cross-over:

$$\text{Parent 1: } (0.3)(1.4)(0.2)(7.4)$$
$$\text{Parent 2: } (0.5)(4.5)(0.1)(5.6)$$

If a = 0.7, the following two offspring would be produced:

$$\text{Offspring1: } (0.36)(2.33)(0.17)(6.86)$$
$$\text{Offspring2: } (0.402)(2.981)(0.149)(6.842)$$

- **Cross-over for Discrete Parameters and Continuous Parameters with Manufacturable Values**

Each discrete parameter or continuous parameter with manufacturable values is represented by a binary chain corresponding to the number of levels. For example, a parameter with two values (levels) is encoded to one bit, a parameter with seven values is encoded to three bits, and an -bits chain represents a parameter with values.

The concatenation of these chains forms the chromosome, which crosses over with another chromosome.

Three different kinds of cross-over are available:

- One-Point

A one-point cross-over operator that randomly selects a cross-over point within a chromosome then interchanges the two parent chromosomes at this point to produce two new offspring.

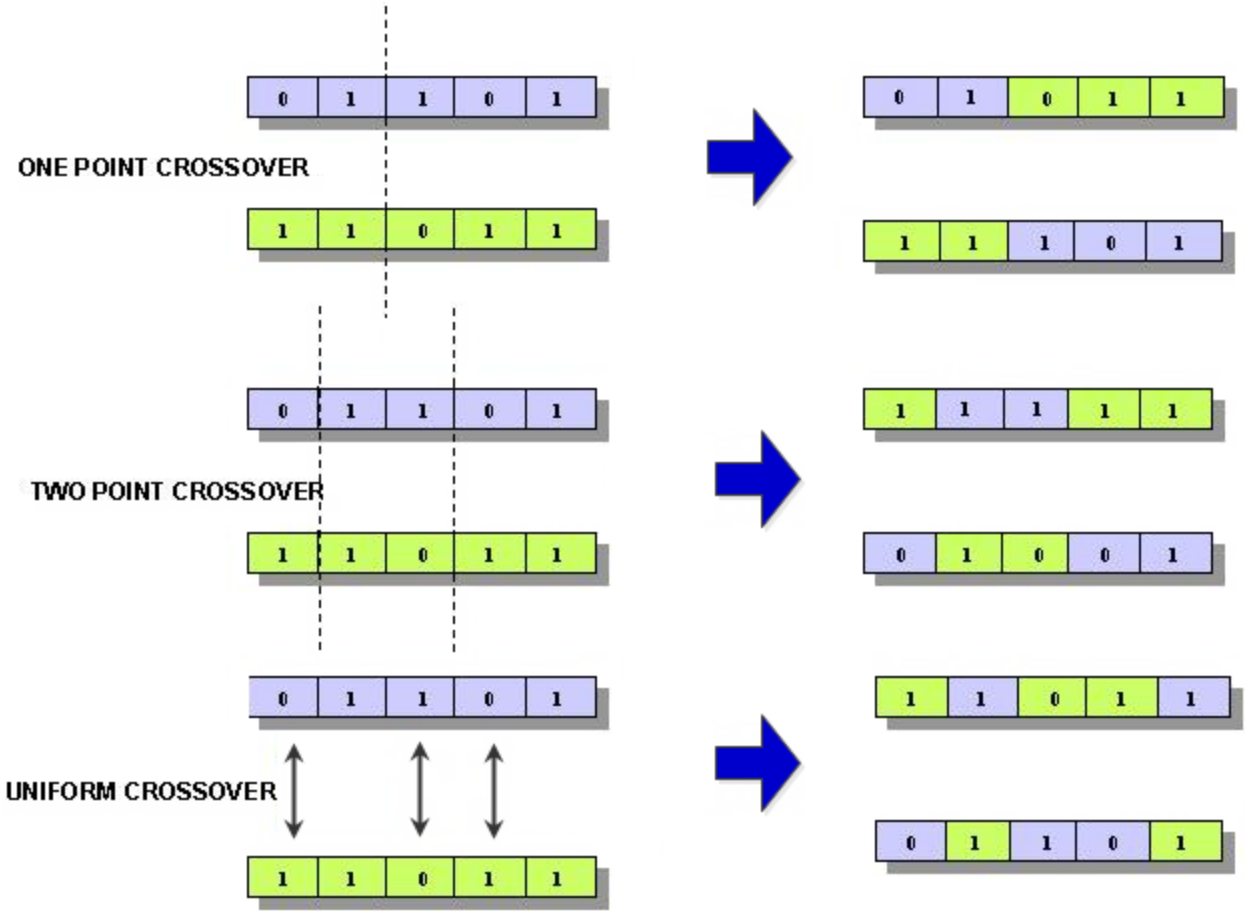
- Two-Point

A two-point cross-over operator randomly selects two cross-over points within a chromosome then interchanges the two parent chromosomes between these points to produce two new offspring.

- Uniform

A uniform cross-over operator decides (with some probability, which is known as the "mixing ratio") which parent contributes each of the gene values in the offspring chromosomes. This allows the parent chromosomes to be mixed at the gene level rather than the segment level (as with one and two-point cross-over). For some problems, this additional flexibility outweighs the disadvantage of destroying building blocks.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. **Mutation**

Mutation alters one or more gene values in a chromosome from its initial state. This can result in entirely new gene values being added to the gene pool. With these new gene values, the genetic algorithm might be able to arrive at a better solution than was previously possible. Mutation is an important part of the genetic search, as it helps to prevent the population from stagnating at any local optima. Mutation occurs during evolution according to a user-defined mutation probability.

- **Mutation for Continuous Parameters**

For continuous parameters, a polynomial mutation operator is applied to implement mutation.

$$C = P + (\text{UpperBound} - \text{LowerBound})\delta$$

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

where C is the child, P is the parent, and δ is a small variation calculated from a polynomial distribution.

- **Mutation for Discrete Parameters and Continuous Parameters with Manufacturable Values**

For discrete parameters or continuous parameters with manufacturable values, a mutation operator inverts the value of the chosen gene (0 goes to 1 and 1 goes to 0) with a probability of 0.5. This mutation operator can only be used for binary genes. The concatenation of these chains forms the chromosome, which crosses over with another chromosome.

Related Topics

Setup Multi-Objective Genetic Algorithm (MOGA) Optimizer

Optimization Variables in Design Space

Cost Function

Convergence Criteria in MOGA-Based Multi-Objective Optimization

Convergence criteria are the conditions that indicate when the optimization has converged. In the Multi-Objective Genetic Algorithm (MOGA)-based multi-objective optimization methods, the following convergence criteria are available:

Maximum Allowable Pareto Percentage

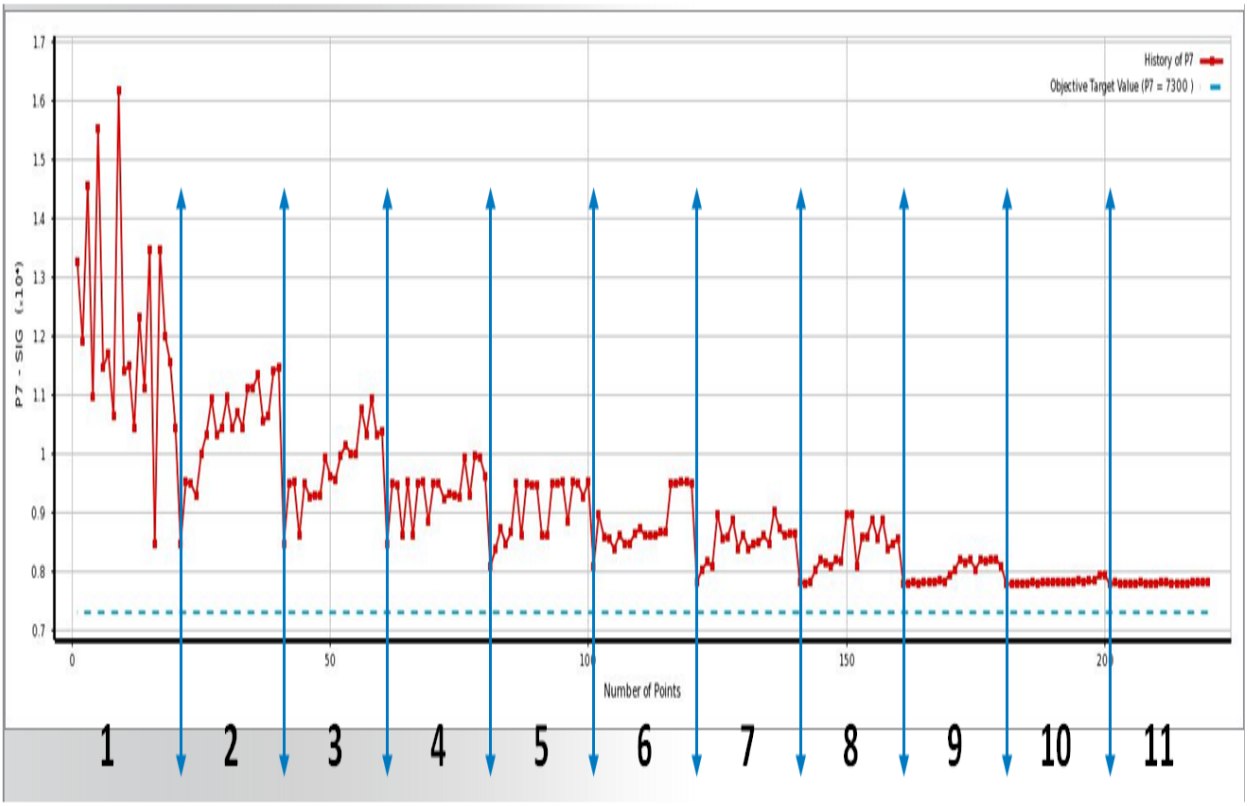
The maximum allowable Pareto percentage criterion looks for a percentage that represents a specified ratio of Pareto points per Number of Samples Per Iteration. When this percentage is reached, the optimization is converged.

Convergence Stability Percentage

The convergence stability percentage criterion looks for population stability, based on mean and standard deviation of the output parameters. When a population is stable with regards to the previous one, the optimization is converged. The criterion functions in the following sequence:

- **Population 1** – When the optimization runs, the first population is not taken into account. Because this population was not generated by the MOGA algorithm, it is not used as a range reference for the output range (for scaling values).
- **Population 2** – The second population sets the range reference. The minimum, maximum, range, mean, and standard deviation are calculated for this population.
- **Populations 3 – 11** – Starting from the third population, the minimum and maximum output values are used in the next steps to scale the values (on a scale of 0 to 100). The mean variations and standard deviation variations are selected. If both of these are smaller than the value for the convergence stability percentage property, the algorithm is converged.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



At each iteration and for each active output, convergence occurs if:

$$\frac{|Mean_i - Mean_{i-1}|}{Max - Min} < \frac{S}{100}$$

and

$$\frac{|StdDev_i - StdDev_{i-1}|}{Max - Min} < \frac{S}{100}$$

Where:

S=Stability Percentage

$Mean_i$= Mean of the *i*-th Population

$StdDev_i$ = Standard Deviation of the *i*-th Population

Max = Maximum Output Value calculated on the first generated population of MOGA

Min = Minimum Output Value calculated on the first generated population of MOGA

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

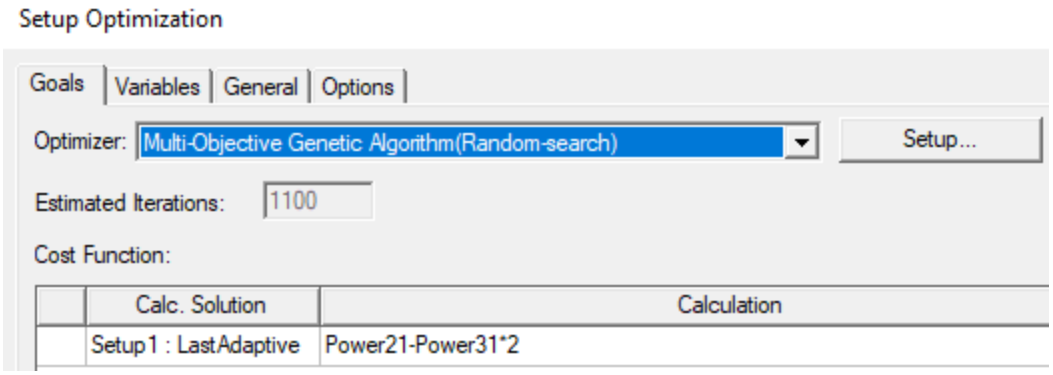
Setting Up Multi-Objective Genetic Algorithm (Random Search) Optimizer

Follow this procedure to set up an optimization analysis using the Multi-Objective Genetic Algorithm (Random Search) Optimizer. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

The Multi-Objective Genetic Algorithm (MOGA) is a hybrid variant of the popular NSGA-II (Non-dominated Sorted Genetic Algorithm-II) based on controlled elitism concepts. It supports all types of input parameters. The Pareto ranking scheme is done by a fast, non-dominated sorting method that is an order of magnitude faster than traditional Pareto ranking methods. The constraint handling uses the same non-dominance principle as the objectives. Therefore, penalty functions and Lagrange multipliers are not needed. This also ensures that the feasible solutions are always ranked higher than the infeasible solutions.

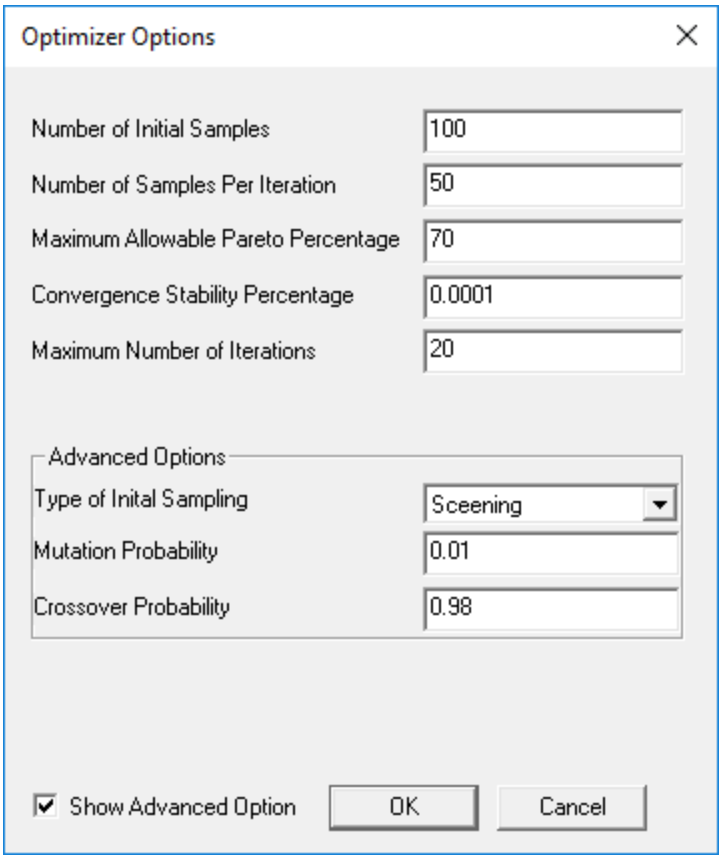
The first Pareto front solutions are archived in a separate sample set internally and are distinct from the evolving sample set. This ensures minimal disruption of Pareto front patterns already available from earlier iterations. You can control the selection pressure (and, consequently, the elitism of the process) to avoid premature convergence by altering the maximum allowable Pareto percentage property.

1. Set up the variables to optimize in the **Design Properties** dialog box. The variables must be swept in a Parametric setup.
2. Click **HFSS** or **Q3D Extractor** or **2D Extractor** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Multi-Objective Genetic Algorithm (Random Search)** from the **Optimizer** drop-down list.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Optionally click **Setup** to open the **Optimizer Options** dialog box.



- **Number of Initial Samples** – Initial number of samples to use. This number must be greater than the number of enabled input parameters. The minimum recommended number of initial samples is 10 times the number of enabled input parameters. The larger the initial sample set, the better your chances of finding the input parameter space that contains the best solutions.

The number of enabled input parameters is also the minimum number of samples required to generate the Sensitivities chart. You can enter a minimum of **2** and a maximum of **10000**. The default is **100**.

If you switch from the Screening method to the MOGA method, MOGA generates a new sample set. For the sake of consistency, enter the same number of initial samples as you used for the Screening method.

- **Number of Samples Per Iteration** – Number of samples to iterate and update with each iteration. This number must be greater than the number of enabled input parameters but less than or equal to the number of initial samples. The default is for a Direct Optimization system.

You can enter a minimum of **2** and a maximum of **10000**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Maximum Allowable Pareto Percentage** – Convergence criterion. Percentage value that represents the ratio of the number of desired Pareto points to the number of samples per iteration. When this percentage is reached, the optimization converges. For example, a value of **70** with Number of Samples Per Iteration set to **200** means that the optimization should stop once the resulting front of the MOGA optimization contains at least **140** points. Of course, the optimization stops before that if the maximum number of iterations is reached.

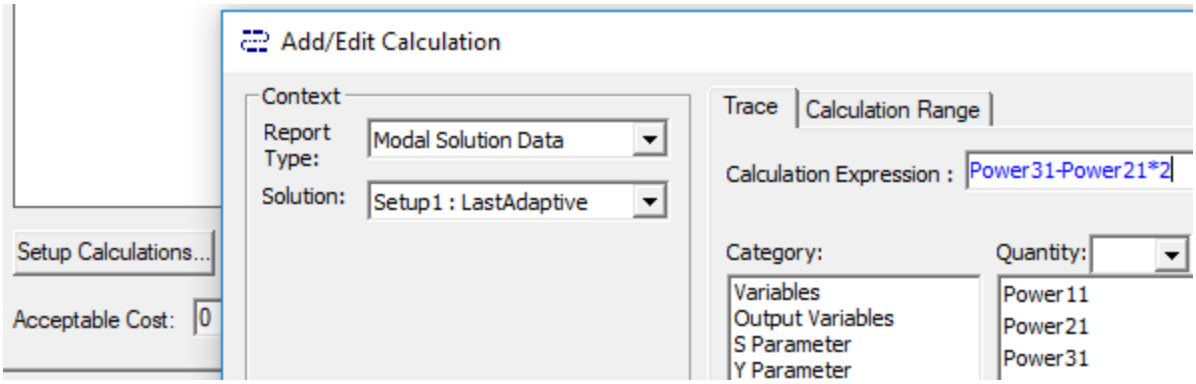
  If the **Maximum Allowable Pareto Percentage** is too low (below **30**), the process can converge prematurely. If the value is too high (above **80**), the process can converge slowly. The value of this property depends on the number of parameters and the nature of the design space itself. The default is **70**. Using a value between **55** and **75** works best for most problems. For more information, see ***Convergence Criteria in MOGA-Based Multi-Objective Optimization***.
- **Convergence Stability Percentage** – Convergence criterion. Percentage value that represents the stability of the population based on its mean and standard deviation. This criterion lets you minimize the number of iterations performed while still reaching the desired level of stability. When the specified percentage is reached, the optimization is converged. The default percentage is **2**. To not take the convergence stability into account, set to **0**. For more information, see ***Convergence Criteria in MOGA-Based Multi-Objective Optimization***.
- **Maximum Number of Iterations** – Stop criterion. Maximum number of iterations that the algorithm is to execute. If this number is reached without the optimization having reached convergence, iterations stop. This also provides an idea of the maximum possible number of function evaluations needed for the full cycle, as well as the maximum possible time it can take to run the optimization. For example, the absolute maximum number of evaluations is given by:

  Number of Initial Samples + Number of Samples Per Iteration * (Maximum Number of Iterations - 1)
- **Type of Initial Sampling** – Advanced option for generating different kinds of sampling. If you do not have any parameter relationships defined, set to **Screening** (default) or **Optimal Space-Filling**. If you have parameter relationships defined, the initial sampling must be performed by the constrained sampling algorithms because parameter relationships constrain the sampling. For such cases, this property is set to **Constrained Sampling**.
- **Mutation Probability** – Advanced option for specifying the probability of applying a mutation on a design configuration. The value must be between **0** and **1**. A larger value indicates a more random algorithm. If the value is **1**, the algorithm becomes a pure random search. A low probability of mutation (<0.2) is recommended. The default is **0.01**. For more information on mutation, see ***MOGA Steps to Generate a New Population***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

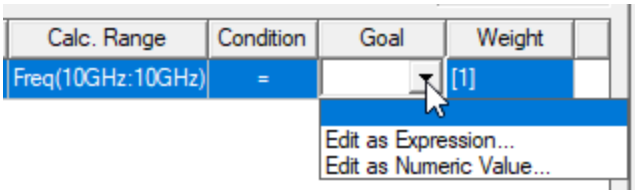
- **Crossover Probability** – Advanced option for specifying the probability with which parent solutions are recombined to generate offspring solutions. The value must be between **0** and **1**. A smaller value indicates a more stable population and a faster (but less accurate) solution. If the value is **0**, the parents are copied directly to the new population. A high probability of crossover (>0.9) is recommended. The default is **0.98**.

5. To Add a cost function, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box.

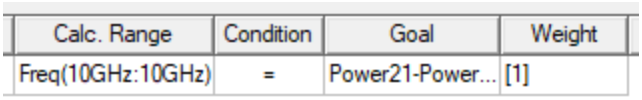


When you have created the calculation, click **Add Calculation** to add it to the **Optimization** setup, and **Done** to close the **Add/EditCalculation** dialog box.

6. In the Optimization setup, in the **Goal** column drop-down list, select either **Edit as Expression** or **Edit as Numeric Value**.



This reopens the **Add/Edit Calculation** dialog box. If you are satisfied with the expression or value displayed, click **Done** to close the dialog box. This enters the expression/value to the **Goal** column.



7. In the **Optimization** setup, if you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

A norm is a function that assigns a positive value to the cost function.

For **L1** norm, the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default), the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors, which means that it is always less than zero. For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.

The norm type doesn't impact goal setting that use as condition the "minimize" or "maximize" scenarios.

8. Optionally, set the Acceptable Cost and Cost Function Noise.
9. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.
10. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization.
    - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.
    - Optionally, set Linear constraints.
    - Select the **View all columns** check box to see all columns, including hidden columns.
11. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

    Select the **Update design parameters' value after optimization** check box to cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.
12. Under the **Options tab**, if you want to save the field solution data for every solved design variations in the optimization analysis, select **Save Fields And Mesh**.

> **Note:**
>
> Do not select this option when requesting a large number of iterations as the data generated will be very large and the system may become slow due to the large I/O requirements.

You may also select **Copy geometrically equivalent meshes** to reuse the mesh when geometry changes are not required, for example when optimizing on a material property or source excitation. This will provide some speed improvement in the optimization process.

Related Topic

Multi-Objective Genetic Algorithm (MOGA)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Convergence Rate % and Initial Finite Difference Delta % in NLPQ and MISQP

Typically, the use of Nonlinear Programming by Quadratic Lagrangian (NLPQL) or Mixed-Integer Sequential Quadratic Programming (MISQP) optimizers is suggested for continuous problems when there is only one objective function. The problem might or might not be constrained and must be analytic. This means that the problem must be defined only by continuous input parameters and that the objective functions and constraints should not exhibit sudden "jumps" in their domain.

The main difference between these algorithms and Multi-Objective Genetic Algorithm (MOGA) is that MOGA is designed to work with multiple objectives and does not require full continuity of the output parameters. However, for continuous single objective problems, the use of NLPQL or MISQP gives greater accuracy of the solution as gradient information and line search methods are used in the optimization iterations. MOGA is a global optimizer designed to avoid local optima traps, while NLPQL and MISQP are local optimizers designed for accuracy.

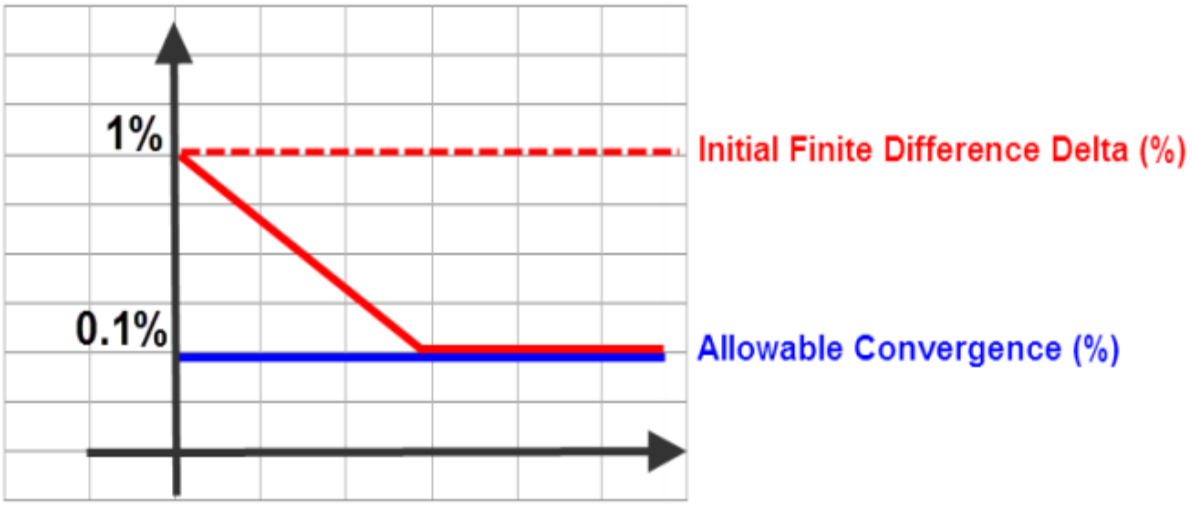
For NLPQL and MISQP, the default convergence rate, which is specified by the Allowable Convergence (%) property, is set to 0.1% for a Direct Optimization system. The maximum value for this property is 100%. This is computed based on the (normalized) Karush-Kuhn-Tucker (KKT) condition. This implies that the fastest convergence rate of the gradients or the functions (objective function and constraint) determine the termination of the algorithm.

The default convergence rate is used in conjunction with the initial finite difference delta percentage value, which is specified by the Initial Finite Difference Delta (%) advanced property. This property defaults to 1% for a Direct Optimization system. Use this property to specify a percentage of the variation between design points to ensure that the Delta use in the calculation of finite differences is large enough to be seen over simulation noise. The specified percentage is defined as a relative gradient perturbation between design points.

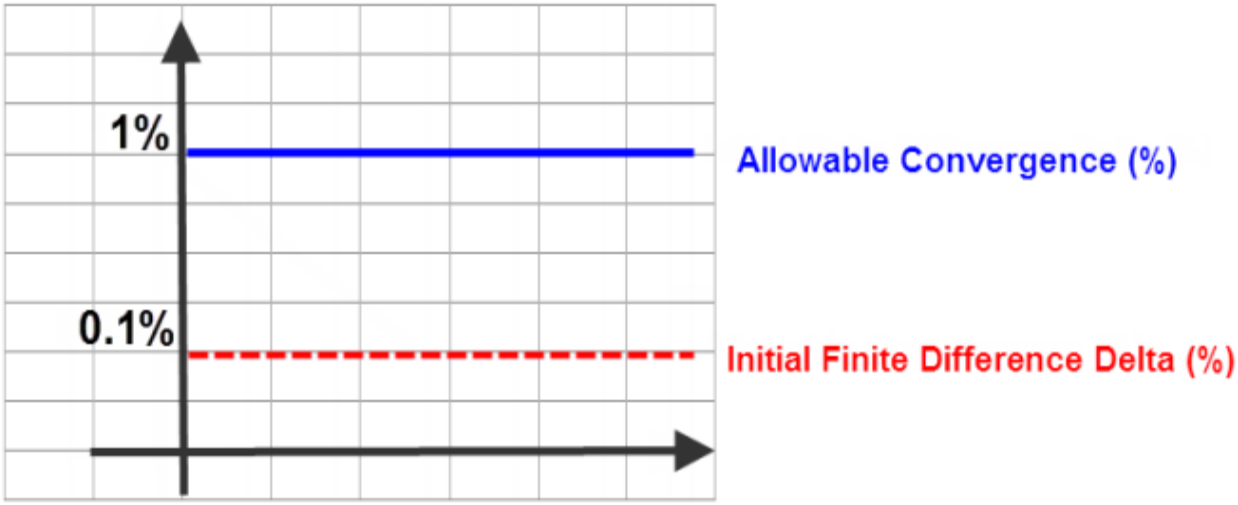
The advantage of this approach is that for large problems, it is possible to get a near-optimal feasible solution quickly without being trapped into a series of iterations involving small solution steps near the optima. To work most effectively with NLPQL and MISQP, keep the following guidelines in mind:

- If the Initial Finite Difference Delta (%) is greater than the Allowable Convergence (%), the relative gradient perturbation gets iteratively smaller, until it matches the allowable convergence rate. At this point, the relative gradient value stays the same through the rest of the analysis.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- If the Initial Finite Difference Delta (%) is less than or equal to the Allowable Convergence (%), the current relative gradient step remains constant through the rest of the analysis.



- Both the Initial Finite Difference Delta (%) and Allowable Convergence (%) should be higher than the magnitude of the noise in your simulation.

When setting the values for these properties, you have the usual trade-offs between speed and accuracy. Smaller values result in more convergence iterations and a more accurate (but slower) solution, while larger values result in fewer convergence iterations and a less accurate (but faster) solution. At the same time, however, you must be aware of the amount of noise in your model. For the input variable variations to be visible in the output variables, both values must be greater than the magnitude of the simulation's noise.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In general, default values for Initial Finite Difference Delta (%) and Allowable Convergence (%) cover the majority of optimization problems. For example, if you know that the noise magnitude in your direct optimization problem is 0.0001, then the default values (Allowable Convergence (%) = 0.001 and Initial Finite Difference Delta (%) = 0.01) are good.

When the defaults are not a good match for your problem, you can adjust the values to better suit your model and your simulation needs. If you require a more numerically accurate solution, set the convergence rate to as low as 1.0E-10%, then set the Initial Finite Difference Delta (%) accordingly.

Related Topics

Setup Nonlinear Programming by Quadratic Lagrangian (Gradient) Optimizer

Mixed-Integer Sequential Quadratic Programming

Mixed-Integer Sequential Quadratic Programming (MISQP) is a mathematical optimization algorithm as developed by Oliver Exler, Thomas Lehmann, and Klaus Schittkowski (NLPQL). This method solves Mixed-Integer Non-Linear Programming (MINLP) of the form:

Minimize:

$$f(x,y)$$

Subject to:

$$g_j(x,y)=0,\ \ j=1,\ldots,m_e,$$
$$g_j(x,y)\geq 0,\ \ j=m_e+1,\ldots,m$$

Where:

$$x\in\mathbb{R}^{n_c}, y\in\mathbb{N}^{n_i}$$
$$x_l\leq x\leq x_u$$
$$y_l\leq y\leq y_u$$

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The symbols and denote the vectors of the continuous and integer variables, respectively. It is assumed that problem functions and are continuously differentiable subject to all. It is not assumed that integer variables can be relaxed. In other words, problem functions are evaluated only at integer points and never at any fractional values in between.

MISQP solves MINLP by a modified sequential quadratic programming (SQP) method. After linearizing constraints and constructing a quadratic approximation of the Lagrangian function, mixed-integer quadratic programs are successively generated and solved by an efficient branch-and-cut method. The algorithm is stabilized by a trust region method as originally proposed by Yuan for continuous programs. Second order corrections are retained. The Hessian of the Lagrangian function is approximated by BFGS updates subject to the continuous and integer variables. MISQP is able to solve also non-convex nonlinear mixed-integer programs.

Related Topics

Setting Up Mixed Integer Quadratic Sequential Optimization

Adaptive Multiple Objective Optimization

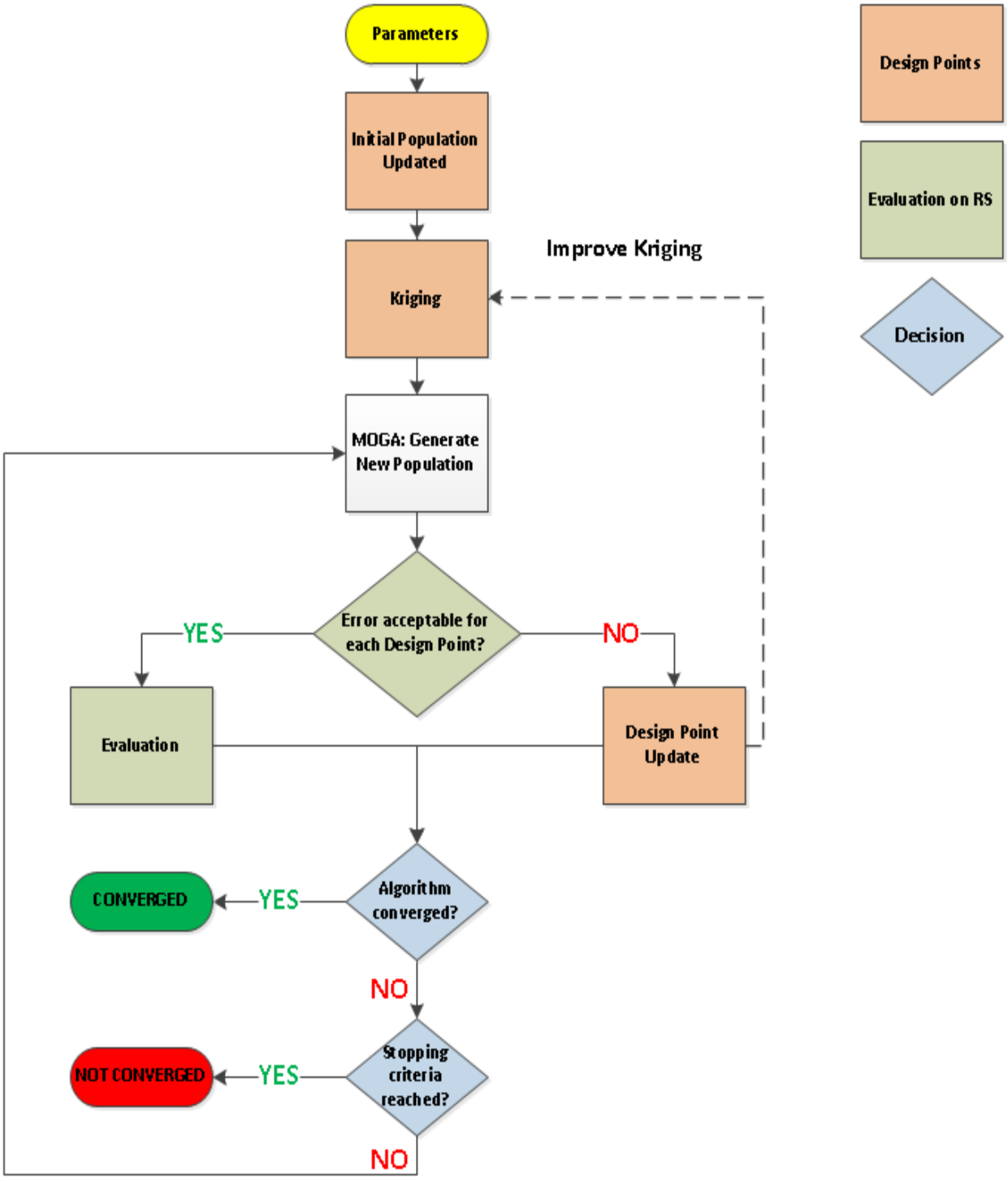
Adaptive Multiple-Objective (AMO) is a mathematical optimization that combines a Kriging response surface and MOGA. It lets you generate a new sample set or use an existing set, providing a more refined approach than the Screening method. Except when necessary, the optimizer does not evaluate all design points. The general optimization approach is the same as MOGA, but a Kriging response surface is used. Part of the population is "simulated" by evaluations of the Kriging response surface. The Kriging error predictor reduces the number of evaluations used in finding the first Pareto front solutions.

AMO supports multiple objectives and multiple constraints. It is limited to continuous parameters, including those with manufacturable values. It is available only for a Direct Optimization system. When discrete parameters are used, MOGA is the more efficient optimization method. For more information, see Multi-Objective Genetic Algorithm (MOGA).

AMO Workflow

The workflow of AMO follows:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



AMO Steps

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. **First Population of MOGA**

   The initial population runs MOGA.

2. **Kriging Generation**

   A Kriging response surface generates for each output. It is based on the first population and improved during simulation with the addition of new design points.

3. **MOGA**

   MOGA runs, using the Kriging response surface as an evaluator. After the first iteration, each population runs when it reaches the number of samples defined by the Number of Samples Per Iteration property.

4. **Evaluate the Population**
5. **Error Check**

   The Kriging error predictor is checked for each point.

   - **Yes: Error Acceptable**

     Each point is validated for error. If the error for a given point is acceptable, the approximated point is included in the next population to be run through MOGA. Return to Step 3.

     - **No: Error Not Acceptable**

       If the error is not acceptable, the points are promoted as design points. The new design points are used to improve the Kriging response surface (return to Step 2) and are included in the next population to be run through MOGA. Return to Step 3.

6. **Convergence Validation**

   The optimization is validated for convergence.

   - **Yes: Optimization Converged**

     MOGA converges when the maximum allowable Pareto percentage is reached. When this happens, the process stops.

   - **No: Optimization Not Converged**

   If the optimization is not converged, the process continues to the next step.

7. **Stopping Criteria Validation**

   If the optimization has not converged, it is validated for fulfillment of the stopping criteria.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Yes: Stopping Criteria Met**

  When the maximum number of iterations has been reached, the process is stopped without having reached convergence..

- **No: Stopping Criteria Not Met**

  If the stopping criteria have not been met, the MOGA algorithm is run again. Return to Step 3.

8. **Conclusion**

   Repeat steps 2 through 7 until the optimization converges or the stopping criteria are met. When one of these occurs, the optimization concludes.

Related Topics

Setup Multi-Objective Genetic Algorithm (MOGA) Optimizer

Optimization Variables in Design Space

Cost Function

Adaptive Single Objective Optimization

Adaptive Single-Objective (ASO) is a mathematical optimization method that combines an OSF (Optimal Space-Filling) DOE, a Kriging response surface, and MISQP. It is a gradient-based algorithm based on a response surface, which provides a refined, global, optimized result.

ASO supports a single objective and multiple constraints. It is available for continuous parameters, including those with manufacturable values. It does not support the use of parameter relationships in the optimization domain and is available only for a direct optimization system.

Like MISQP, ASO solves constrained nonlinear programming problems of the form:

Minimize:

$$f = f(\{x\})$$

Subject to:

$$g_k(\{x\}) \leq 0, \forall k = 1,2,\ldots,K$$

$$h_l(\{x\}) = 0, \forall I = 1,2\ldots,L$$

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

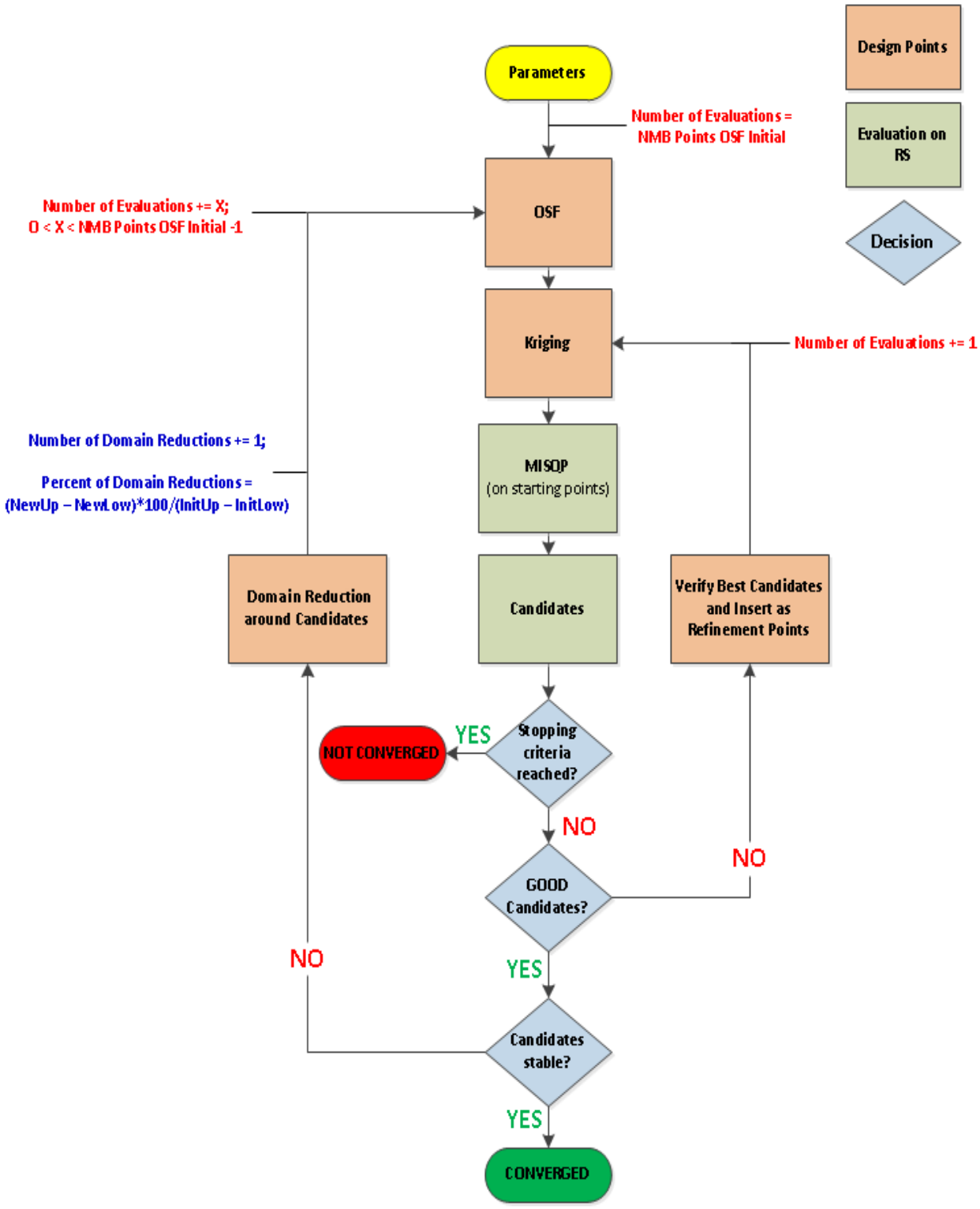
Where:

$$\{x_L\} \leq \{x\} \leq \{x_U\}$$

ASO Workflow

The workflow of ASO follows:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



AMO Steps

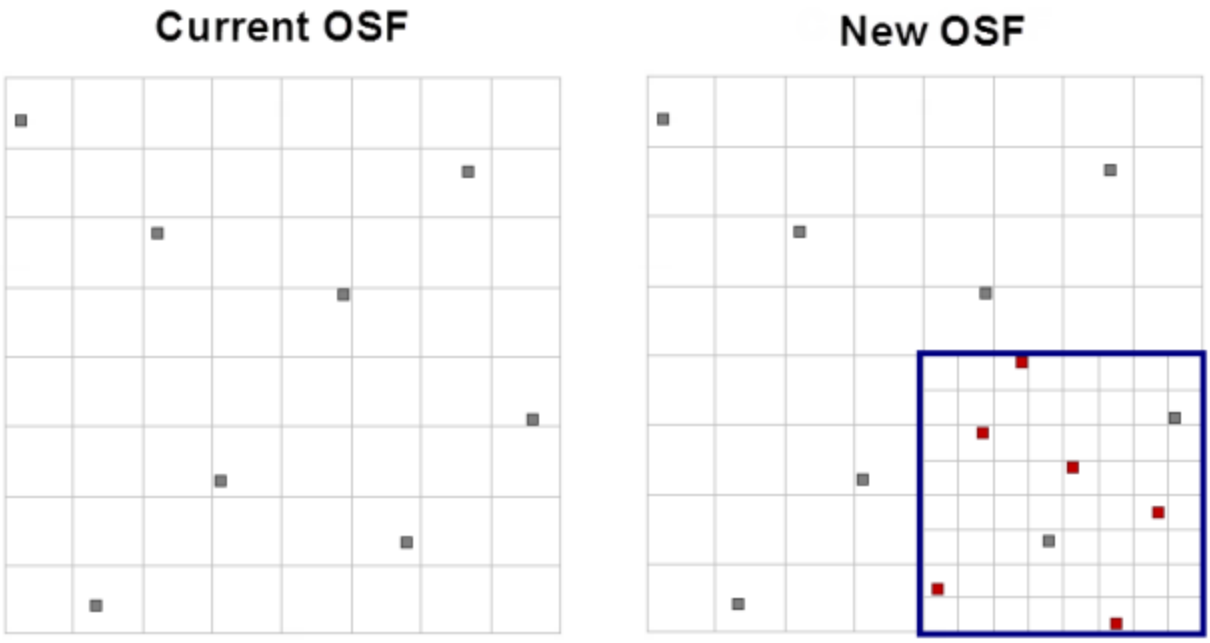
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. **OSF Sampling**

   OSF (Optimal Space-Filling Design) is used for the Kriging construction. In the original OSF, the number of samples equals the number of divisions per axis and there is one sample in each division.

   When a new OSF is generated after a domain reduction, the reduced OSF has the same number of divisions as the original and keeps the existing design points within the new bounds. New design points are added until there is a point in each division of the reduced domain.

   In the following example, the original domain has eight divisions per axis and contains eight design points. The reduced domain also has eight divisions per axis and includes two of the original design points. To have a design point in each division, six new design points must be added.



2. **Kriging Generation**

   A response surface generates for each output. It is based on the current OSF and consequently on the current domain bounds.

   A Kriging response surface generates for each output. It is based on the first population and improved during simulation with the addition of new design points.

3. **MISQP**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

MISQP runs on the current Kriging response surface to find potential candidates. A few MISQP processes run at the same time, beginning with different starting points, and consequently give different candidates.

4. **Candidate Point Validation**

   All the obtained candidates are validated or not, based on the Kriging error predictor. The candidate point is checked to see if further refinement of the Kriging surface changes the selection of this point. A candidate is acceptable if there aren't any points that call it into question, according to this error prediction. If the quality of the candidate is not called into question, the domain bounds are reduced. Otherwise, the candidate is calculated as a verification point.

   - **Refinement Point Creation** (If the selection is *not* to be changed.)

     When a new verification point is calculated, it is inserted in the current Kriging response surface as a refinement point and the MISQP process is restarted.

   - **Domain Reduction** (If the selection is to be changed.)

     When candidates are validated, new domain bounds must be calculated. If all of the candidates are in the same zone, the bounds are reduced, centered on the candidates. Otherwise, the bounds are reduced as an inclusive box of all candidates. At each domain reduction, a new OSF is generated (conserving design points between the new bounds) and a new Kriging response surface is generated based on this new OSF.

5. **Convergence and Stop Criteria**

   The optimization is converged when the candidates found are stable. This occurs when all of the MISQP processes run on the response surface converge to the same verified candidate point. However, there are four stop criteria that can stop the algorithm: Maximum Number of Evaluations, Maximum Number of Domain Reductions, Percentage of Domain Reductions, and Convergence Tolerance.

Related Topics

Setup Adaptive Single-Objective (Gradient) Optimizer

Optimization Variables in Design Space

Cost Function

Optimization Variables and the Design Space

Once the optimization variables are specified, the optimizer handles each of them as an *n*-dimensional vector *x*. Any point in the design space corresponds to a particular *x*-vector and to a design instance. Each design instance may be evaluated via FEA and assigned a cost value;

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

therefore, the cost function is defined over the design space (cost(x): $R^n \rightarrow R$, where n is the number of optimization variables.

In practice, a solution of the minimization problem is sought only on a bounded subset of the Rn space. This subset is called the feasible domain and is defined via linear constraints.

Setting Up an Optimization Analysis

Use optimization to vary predefined variables in the nominal design to search for the solution that best satisfies a set of user-defined goals or cost functions. Optimetrics modifies the variable values until the minimum is reached with acceptable accuracy.

Note:

- You can define more than one optimization analysis setup per design.
- You can create an Optimization setup before defining variables, but all variables must be defined before you start the Optimization analysis.
- Once you have created an optimization analysis setup, you can copy and paste it, then make changes to the copy, rather than redoing the whole process for minor changes.

To provide a broad range of capability, Optimetrics incorporates the following types of numerical optimizers:

- Sequential Nonlinear Programming (Gradient) (SNLP)
- Merit-based Sequential Quadratic Programming (Gradient) (MBSQ)
- Sequential Mixed Integer Nonlinear Programming (Gradient and Discrete) (SMINLP)
- Quasi-Newton (Gradient)
- Pattern Search (Search-based)
- Genetic Algorithm (Random search)
- MATLAB
- Screening (Search based)
- Multi-Objective Genetic Algorithm
- Non-linear Programming by Quadratic Lagrangian (Gradient)
- Mixed-Integer Sequential Quadratic Programming (Gradient and Discrete)
- Adaptive Multiple Objective (Gradient)
- Adaptive Single Objective (Gradient)

Click the links above to view the setup procedure for each optimizer. Options for the analysis are listed in the table.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

You can also use these *optional* optimization solution setup options:

- Modify the starting variable value.
- Modify the minimum and maximum values of variables that will be optimized.
- Exclude variables from optimization.
- Modify the values of fixed variables that are not being optimized.
- Set the minimum and maximum step size between solved design variations (For the quasi-Newton and Patterns Search optimizers, Variables tab).
- Set the minimum and maximum focus size. (For the SNLP and SMINLP optimizers, Variables tab).
- Set Linear constraints.
- Request that Optimetrics solve a parametric sweep before an optimization analysis.
- Request that Optimetrics solve a parametric sweep during an optimization analysis.
- Automatically update optimized variables to the optimal values during an optimization or after an optimization analysis is completed.
- Change the norm used for the cost function calculation (Advanced Option)
- Set **HPC and Analysis Options** to select or create an analysis configuration.
- Enable a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses. By default, the fast calculation-update option is enabled whenever it is applicable.

> **Note:**
>
> Sweeping or using a complex variable is not allowed in any optimetrics setup, including optimization, statistical, sensitivity, and tuning setups.

> **Note:**
>
> Improper or undefined simulation setups will cause errors during Optimetrics Analysis. To verify the analysis setups, select **Analysis** > **Analyze** in the **Project Manager** pane to analyze the nominal circuit and review the messages in the Twin Builder **Message Manager** pane prior to running the Optimetrics Analysis.

Related Topics

Optimization Overview

Choosing an Optimizer

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Optimization Setup for the Quasi-Newton (Gradient) Optimizer

Follow this procedure to set up an optimization analysis using the quasi-Newton (Gradient) Optimizer. Once you create a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

1. Set up the variables to optimize in the **Design Properties** dialog box.
2. Select **Twin Builder** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Quasi Newton(Gradient)** from the **Optimizer** drop-down list. The **Acceptable Cost** and **Noise** fields become active.
4. Type the maximum number of iterations you want Optimetrics to perform during the optimization analysis in the **Max. No. of Iterations** text box.
5. Under **Cost Function**, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box, and add a cost function.
6. Type the value of the cost function at which the optimization process should stop in the **Acceptable Cost** text box.
7. Type the cost function noise in the **Noise** text box.
8. If you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

     A norm is a function that assigns a positive value to the cost function.

     For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization.***)

     The norm type does not affect goal settings that use either the Minimize or Maximize condition.
9. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.
10. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization, and the **Min/Max Step Size** for the analysis.
    - You can also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Optionally, set Linear constraints.
- Select the **View all columns** check box to see all columns, including hidden columns.

11. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

    When you select the **Update design parameters' value after optimization** check box, Optimetrics modifies the variable values in the nominal design to match the final values from the optimization analysis.

12. Use the **Options** tab if you want to enable use of a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses, and to save the solution data for solved design variations in the analysis.

Optimization Setup for the Pattern Search (Search-based) Optimizer

Follow this procedure to set up an optimization analysis using the Pattern Search(Search-based) Optimizer. Once you create a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

1. Set up the variables you want to optimize in the **Design Properties** dialog box.
2. Select **Twin Builder** > **Optimetrics Analysis** > **Add Screening & Optimization**. The **Setup Optimization** dialog box appears.
3. Under the **Goal**s tab, select the optimizer by selecting **Pattern Search(Search-based)** from the **Optimizer** drop-down list. The **Acceptable Cost** and **Noise** fields are enabled.
4. Type the maximum number of iterations you want Optimetrics to perform during the optimization analysis in the **Max. No. of Iterations** text box.
5. Under **Cost Function**, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box, and add a cost function.
6. Type the value of the cost function at which the optimization process should stop in the **Acceptable Cost** text box.
7. Type the cost function noise in the **Noise** text box.
8. If you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

     A norm is a function that assigns a positive value to the cost function.

     For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the cost function uses the maximum among all the weighted goal errors. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

The norm type does not affect Goal settings that use either the Minimize or Maximize condition.

9. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.
10. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization, and the **Min/Max Step Size** for the analysis.
    - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.
    - Optionally, set Linear constraints.
    - Select the **View all columns** check box to see all columns, including hidden columns.
11. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

    Select the **Update design parameters' value after optimization** check box to cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.
12. Use the **Options** tab if you want to enable use of a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses, and to save the solution data for solved design variations in the analysis.

Optimization Setup for the Merit-based Sequential Quadratic Programming (Gradient) Optimizer

Follow this procedure to set up an optimization analysis using the Merit-based Sequential Quadratic Programming (Gradient) Optimizer or Merit-based SQP Optimizer. Once you create a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

1. Set up the variables you want to optimize in the **Design Properties** dialog box.
2. Select **Twin Builder** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Merit-based Sequential Quadratic Programming (Gradient)** from the **Optimizer** drop-down list.
4. Type the maximum number of iterations you want Optimetrics to perform during the optimization analysis in the **Max. No. of Iterations** text box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Under **Cost Function**, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box, and add a cost function.
6. To select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

   A norm is a function that assigns a positive value to the cost function.

   For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

   The norm type does not affect Goal settings that use either the Minimize or Maximize condition.
7. Optionally, click ***HPC and Analysis Options*** to select or create an analysis configuration.
8. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization, and the **Min/Max Focus** for the analysis.
   - You can also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
   - Optionally, modify the values of fixed variables that are not being optimized.
   - Optionally, set Linear constraints.
   - Select the **View all columns** check box to see all columns, including hidden columns.
9. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

   Selecting the **Update design parameters' value after optimization** check box causes Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.
10. Use the **Options** tab if you want to enable use of a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses, and to save the solution data for solved design variations in the analysis.

Optimization Setup for the Sequential Nonlinear Programming (Gradient) Optimizer

Follow this procedure to set up an optimization analysis using the Sequential Nonlinear Programming (Gradient) Optimizer, or SNLP Optimizer. Once you create a setup, you can **Copy**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

1. Set up the variables to optimize in the **Design Properties** dialog box.
2. Select **Twin Builder** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Sequential Nonlinear Programming(Gradient)** from the **Optimizer** drop-down list.
4. Type the maximum number of iterations for Optimetrics to perform during the optimization analysis in the **Max. No. of Iterations** text box.
5. Under **Cost Function**, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box, and add a cost function.
6. If you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

     A norm is a function that assigns a positive value to the cost function.

     For **L1** norm, the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default), the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm, the cost function uses the maximum among all the weighted goal errors. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

     The norm type does not affect Goal settings that use either the Minimize or Maximize condition.
7. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.
8. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization, and the **Min/Max Focus** for the analysis.
   - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
   - Optionally, modify the values of fixed variables that are not being optimized.
   - Optionally, set Linear constraints.
   - Select the **View all columns** check box to see all columns, including hidden columns.
9. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Selecting the **Update design parameters' value after optimization** check box causes Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.

10. Use the **Options** tab if you want to enable use of a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses, and to save the solution data for solved design variations in the analysis.

Optimization Setup for the Sequential Mixed Integer Nonlinear Programming (Gradient and Discrete) Optimizer

Follow this procedure to set up an optimization analysis using the Sequential Mixed Integer Nonlinear Programming (Gradient and Discrete) Optimizer, or SMINLP Optimizer. Once you create a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

1. Set up the variables you want to optimize in the **Design Properties** dialog box.
2. Select **Twin Builder** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Sequential Mixed Integer Nonlinear Programming(Gradient and Discrete)** from the **Optimizer** drop-down list.
4. Type the maximum number of iterations you want Optimetrics to perform during the optimization analysis in the **Max. No. of Iterations** text box.
5. Under **Cost Function**, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box, and add a cost function.
6. If you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

     A norm is a function that assigns a positive value to the cost function.

     For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

     The norm type does not affect Goal settings that use either the Minimize or Maximize condition.
7. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

8. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization, and the **Min/Max Focus** for the analysis.
   - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
   - Optionally, modify the values of fixed variables that are not being optimized.
   - Optionally, set Linear constraints.
   - Select the **View all columns** check box to see all columns, including hidden columns.
9. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

   Selecting the **Update design parameters' value after optimization** check box causes Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.
10. Use the **Options** tab if you want to enable use of a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses, and to save the solution data for solved design variations in the analysis.

Optimization Setup for the Genetic Algorithm (Random Search) Optimizer

Follow this procedure to set up an optimization analysis using the Genetic Algorithm (Random Search) Optimizer. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

1. Set up the variables to optimize in the **Design Properties** dialog box.
2. Select **Twin Builder** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Genetic Algorithm(Random search)** from the **Optimizer** drop-down list.
4. Click **Setup** to modify the Advanced Genetic Algorithm Optimizer Options.
5. Under **Cost Function**, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box, and add a cost function.
6. If you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

   A norm is a function that assigns a positive value to the cost function.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the maximum norm the cost function uses the maximum among all the weighted goal errors. For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.

The norm type does not affect Goal settings that use either the Minimize or Maximize condition.

7. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.

8. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization, and the **Min/Max Focus** for the analysis.
   - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
   - Optionally, modify the values of fixed variables that are not being optimized.
   - Optionally, set linear constraints.
   - Select the **View all columns** check box to see all columns, including hidden columns.
9. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

   Select the **Update design parameters' value after optimization** check box to cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.

10. Use the **Options** tab if you want to enable use of a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses, and to save the solution data for solved design variations in the analysis.

Optimization Setup for the MATLAB Optimizer

Follow this procedure to set up an optimization analysis using the MATLAB Optimizer. Once you create a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

1. Set up the variables you want to optimize in the **Design Properties** dialog box.
2. Select **Twin Builder** > **Optimetrics Analysis**> **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **MATLAB** from the **Optimizer** drop-down list. Selecting MATLAB enables the **Acceptable Cost** and **Noise** fields.
4. Click **Setup...** to modify the MATLAB Optimizer Options.
5. Type the maximum number of iterations you want Optimetrics to perform during the optimization analysis in the **Max. No. of Iterations** text box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. Under **Cost Function**, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box, and add a cost function.
7. Type the value of the cost function at which the optimization process should stop in the **Acceptable Cost** text box.
8. Type the cost function noise in the **Noise** text box.
9. If you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

   A norm is a function that assigns a positive value to the cost function.

   For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

   > The norm type does not affect Goal settings that use either the Minimize or Maximize condition.

10. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.

11. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization, and the **Min/Max Step Size** for the analysis.
    - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.
    - Optionally, set Linear constraints.
    - Select the **View all columns** check box to see all columns, including hidden columns.
12. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

    Select the **Update design parameters' value after optimization** check box to cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.
13. Under the **Options** tab, if you want to save the field solution data for every solved design variation in the optimization analysis, select **Save Fields And Mesh**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Do not select this option when requesting a large number of iterations as the data generated will be very large and the system may become slow due to the large I/O requirements.

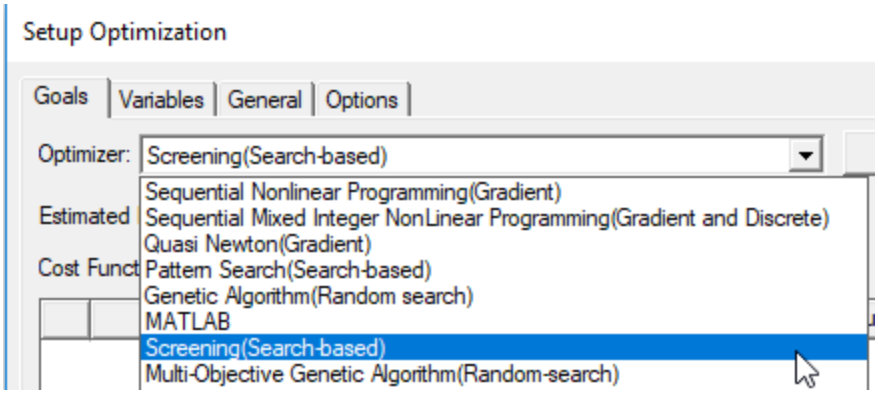
14. You can also select **Copy geometrically equivalent meshes** to reuse the mesh when geometry changes are not required, for example when optimizing on a material property or source excitation. This will provide some speed improvement in the optimization process.

Setting Up Screening (Search Based) Optimizer

Follow this procedure to set up an optimization analysis using the Screening Optimizer. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

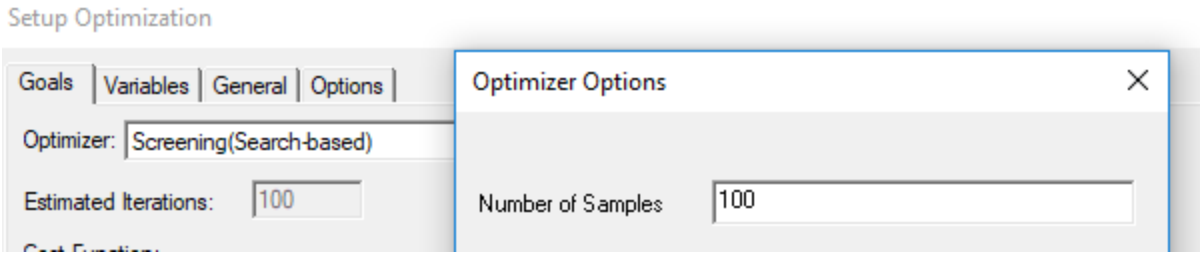
This is a non-iterative direct sampling method that uses a quasi-random number generator based on the Hammersley algorithm. Start with Screening to locate the multiple tentative optima, then refine with NLPQL or MISQP to zoom in on the individual local maximum or minimum value. Usually Screening is used for preliminary design. If you then want to refine, use these candidate points as starting points for gradient methods.

1. Set up the variables you want to optimize in the **Design Properties** dialog box. The variables must be swept in a Parametric setup.
2. Select **Twin Builder** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Screening (Search Based)** from the **Optimizer** drop-down list.



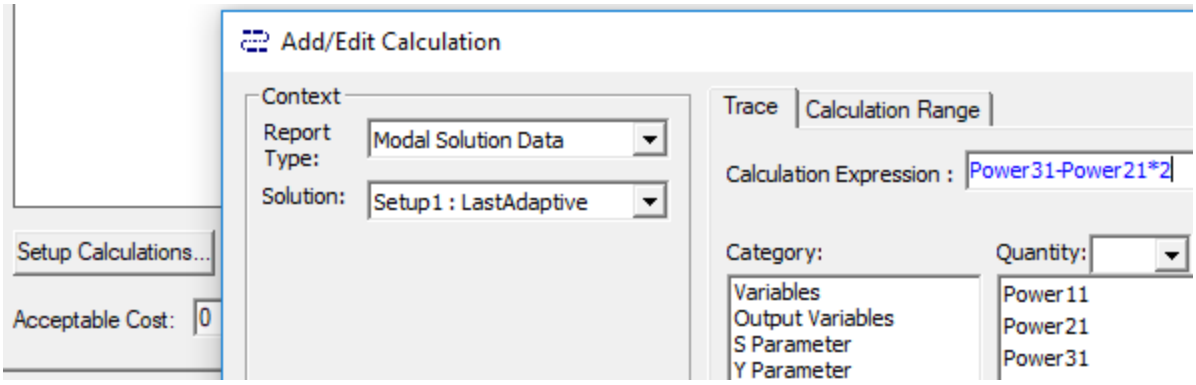
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Optionally click **Setup** to open the **Optimizer Options** dialog box to change the default number of samples from **100**.



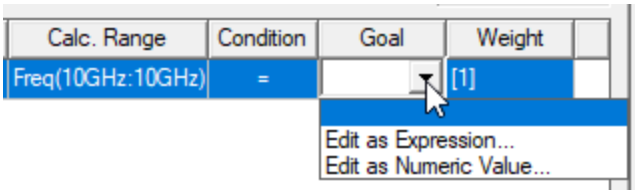
   The number of samples must be greater than the number of enabled input parameters. The number of enabled input parameters is also the minimum number of samples required to generate the Sensitivities chart. You can enter a minimum of **2** and a maximum of **10,000**. The default is **100** for a Direct Optimization system.

5. To Add a cost function, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box.



   When you have created the calculation, click **Add Calculation** to add it to the **Optimization** setup, and **Done** to close the **Add/EditCalculation** dialog box.

6. In the Optimization setup, in the drop-down for the **Goal** column, select **Edit as Expression** or **Edit as Numeric Value**.



7. This reopens the **Add/Edit Calculation** dialog box. If you are satisfied with the expression or value displayed, click **Done** to close the dialog box. This enters the expression/value to the **Goal** column.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Calc. Range	Condition	Goal	Weight
Freq(10GHz:10GHz)	=	Power21-Power...	[1]

8. In the **Optimization** setup, if you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

     A norm is a function that assigns a positive value to the cost function.

     For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors, which means that it is always less than zero. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

     The norm type doesn't impact goal setting that use as condition the "minimize" or "maximize" scenarios.

8. Optionally, set the Acceptable Cost and Cost Function Noise.
9. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.
10. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization.
    - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.
    - Optionally, set Linear constraints.
    - Select the **View all columns** check box to see all columns, including hidden columns.
11. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

    Select the **Update design parameters' value after optimization** check box to cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.

12. Under the **Options tab**, if you want to save the field solution data for every solved design variations in the optimization analysis, select **Save Fields And Mesh**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Do not select this option when requesting a large number of iterations as the data generated will be very large and the system may become slow due to the large I/O requirements.

You can also select **Copy geometrically equivalent meshes** to reuse the mesh when geometry changes are not required, for example when optimizing on a material property or source excitation. This will provide some speed improvement in the optimization process.

Related Topics

Screening (Shifted Hammersley Sampling) Optimization

Setting Up Multi-Objective Genetic Algorithm (Random Search) Optimizer

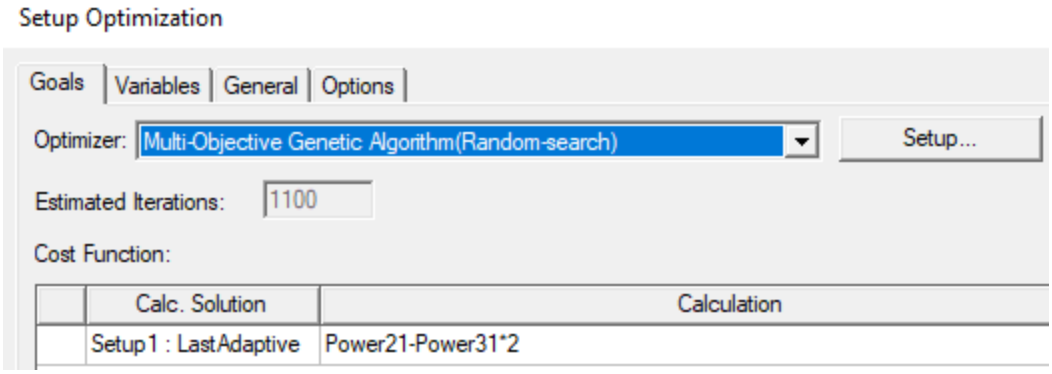
Follow this procedure to set up an optimization analysis using the Multi-Objective Genetic Algorithm (Random Search) Optimizer. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

The Multi-Objective Genetic Algorithm (MOGA) is a hybrid variant of the popular NSGA-II (Non-dominated Sorted Genetic Algorithm-II) based on controlled elitism concepts. It supports all types of input parameters. The Pareto ranking scheme is done by a fast, non-dominated sorting method that is an order of magnitude faster than traditional Pareto ranking methods. The constraint handling uses the same non-dominance principle as the objectives. Therefore, penalty functions and Lagrange multipliers are not needed. This also ensures that the feasible solutions are always ranked higher than the infeasible solutions.

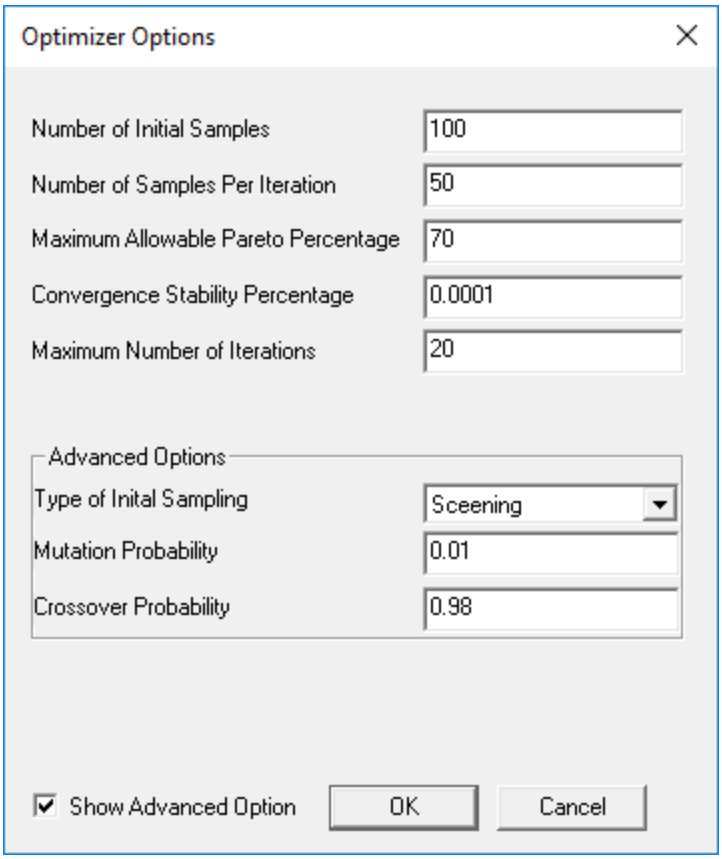
The first Pareto front solutions are archived in a separate sample set internally and are distinct from the evolving sample set. This ensures minimal disruption of Pareto front patterns already available from earlier iterations. You can control the selection pressure (and, consequently, the elitism of the process) to avoid premature convergence by altering the maximum allowable Pareto percentage property.

1. Set up the variables to optimize in the **Design Properties** dialog box. The variables must be swept in a Parametric setup.
2. Click **HFSS** or **Q3D Extractor** or **2D Extractor** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Multi-Objective Genetic Algorithm (Random Search)** from the **Optimizer** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. Optionally click **Setup** to open the **Optimizer Options** dialog box.



- **Number of Initial Samples** – Initial number of samples to use. This number must be greater than the number of enabled input parameters. The minimum recommended number of initial samples is 10 times the number of enabled input parameters. The larger the initial sample set, the better your chances of finding the input parameter space that contains the best solutions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The number of enabled input parameters is also the minimum number of samples required to generate the Sensitivities chart. You can enter a minimum of **2** and a maximum of **10000**. The default is **100**.

If you switch from the Screening method to the MOGA method, MOGA generates a new sample set. For the sake of consistency, enter the same number of initial samples as you used for the Screening method.

- **Number of Samples Per Iteration** – Number of samples to iterate and update with each iteration. This number must be greater than the number of enabled input parameters but less than or equal to the number of initial samples. The default is for a Direct Optimization system.

  You can enter a minimum of **2** and a maximum of **10000**.

- **Maximum Allowable Pareto Percentage** – Convergence criterion. Percentage value that represents the ratio of the number of desired Pareto points to the number of samples per iteration. When this percentage is reached, the optimization converges. For example, a value of **70** with Number of Samples Per Iteration set to **200** means that the optimization should stop once the resulting front of the MOGA optimization contains at least **140** points. Of course, the optimization stops before that if the maximum number of iterations is reached.

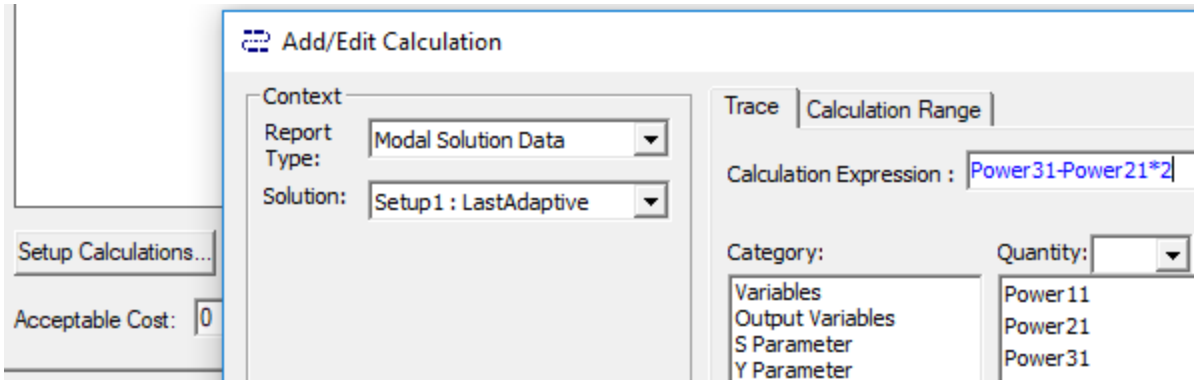
  If the **Maximum Allowable Pareto Percentage** is too low (below **30**), the process can converge prematurely. If the value is too high (above **80**), the process can converge slowly. The value of this property depends on the number of parameters and the nature of the design space itself. The default is **70**. Using a value between **55** and **75** works best for most problems. For more information, see ***Convergence Criteria in MOGA-Based Multi-Objective Optimization***.
- **Convergence Stability Percentage** – Convergence criterion. Percentage value that represents the stability of the population based on its mean and standard deviation. This criterion lets you minimize the number of iterations performed while still reaching the desired level of stability. When the specified percentage is reached, the optimization is converged. The default percentage is **2**. To not take the convergence stability into account, set to **0**. For more information, see ***Convergence Criteria in MOGA-Based Multi-Objective Optimization***.
- **Maximum Number of Iterations** – Stop criterion. Maximum number of iterations that the algorithm is to execute. If this number is reached without the optimization having reached convergence, iterations stop. This also provides an idea of the maximum possible number of function evaluations needed for the full cycle, as well as the maximum possible time it can take to run the optimization. For example, the absolute maximum number of evaluations is given by:

  Number of Initial Samples + Number of Samples Per Iteration * (Maximum Number of Iterations - 1)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

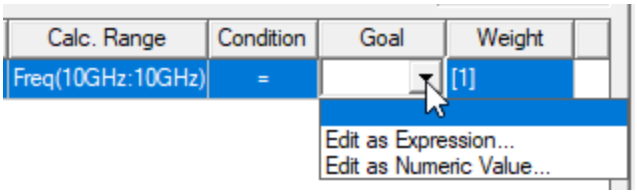
- **Type of Initial Sampling** – Advanced option for generating different kinds of sampling. If you do not have any parameter relationships defined, set to **Screening** (default) or **Optimal Space-Filling**. If you have parameter relationships defined, the initial sampling must be performed by the constrained sampling algorithms because parameter relationships constrain the sampling. For such cases, this property is set to **Constrained Sampling**.
- **Mutation Probability** – Advanced option for specifying the probability of applying a mutation on a design configuration. The value must be between **0** and **1**. A larger value indicates a more random algorithm. If the value is **1**, the algorithm becomes a pure random search. A low probability of mutation (<0.2) is recommended. The default is **0.01**. For more information on mutation, see ***MOGA Steps to Generate a New Population***.
- **Crossover Probability** – Advanced option for specifying the probability with which parent solutions are recombined to generate offspring solutions. The value must be between **0** and **1**. A smaller value indicates a more stable population and a faster (but less accurate) solution. If the value is **0**, the parents are copied directly to the new population. A high probability of crossover (>0.9) is recommended. The default is **0.98**.

5. To Add a cost function, click **Setup Calculations** to open the **Add/Edit Calculation** dialog box.



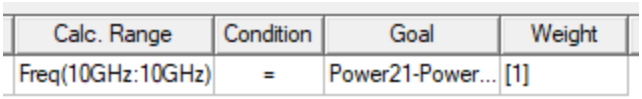
When you have created the calculation, click **Add Calculation** to add it to the **Optimization** setup, and **Done** to close the **Add/EditCalculation** dialog box.

6. In the Optimization setup, in the **Goal** column drop-down list, select either **Edit as Expression** or **Edit as Numeric Value**.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

This reopens the **Add/Edit Calculation** dialog box. If you are satisfied with the expression or value displayed, click **Done** to close the dialog box. This enters the expression/value to the **Goal** column.



Calc. Range	Condition	Goal	Weight
Freq(10GHz:10GHz)	=	Power21-Power...	[1]

7. In the **Optimization** setup, if you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

     A norm is a function that assigns a positive value to the cost function.

     For **L1** norm, the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default), the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors, which means that it is always less than zero. For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.

     The norm type doesn't impact goal setting that use as condition the "minimize" or "maximize" scenarios.
8. Optionally, set the Acceptable Cost and Cost Function Noise.
9. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.
10. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization.
    - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.
    - Optionally, set Linear constraints.
    - Select the **View all columns** check box to see all columns, including hidden columns.
11. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

    Select the **Update design parameters' value after optimization** check box to cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.
12. Under the **Options tab**, if you want to save the field solution data for every solved design variations in the optimization analysis, select **Save Fields And Mesh**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Do not select this option when requesting a large number of iterations as the data generated will be very large and the system may become slow due to the large I/O requirements.

You may also select **Copy geometrically equivalent meshes** to reuse the mesh when geometry changes are not required, for example when optimizing on a material property or source excitation. This will provide some speed improvement in the optimization process.

Related Topic

Multi-Objective Genetic Algorithm (MOGA)

Setting Up Nonlinear Programming by Quadratic Lagrangian (Gradient) Optimizer

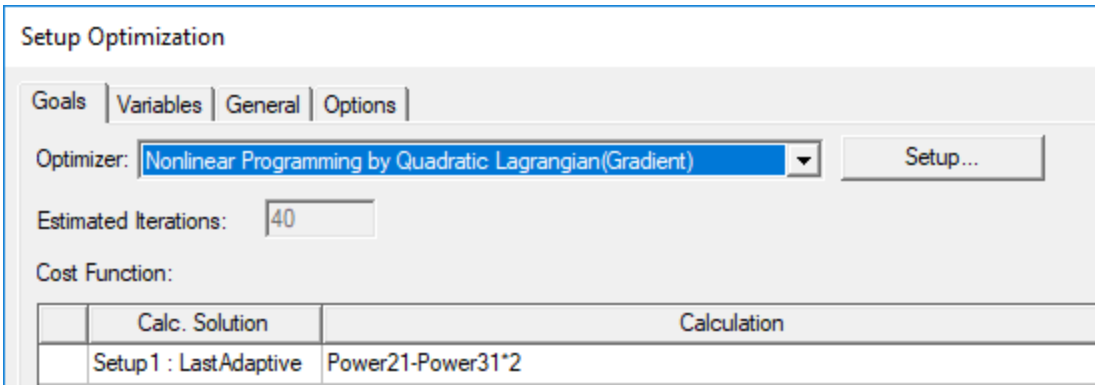
Follow this procedure to set up an optimization analysis using the Nonlinear Programming by Quadratic Lagrangian (Gradient) Optimizer. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

The NLPQL (Nonlinear Programming by Quadratic Lagrangian) method can be used for Direct Optimization systems. It lets you generate a new sample set to provide a more refined approach than the Screening method. Available for continuous input parameters only, NLPQL can handle only one output parameter goal. Other output parameters can be defined as constraints. For more information, see ***Convergence Rate % and Initial Finite Difference Delta % in NLPQL and MISQP and Nonlinear Programming by Quadratic Lagrangian (NLPQL)***.

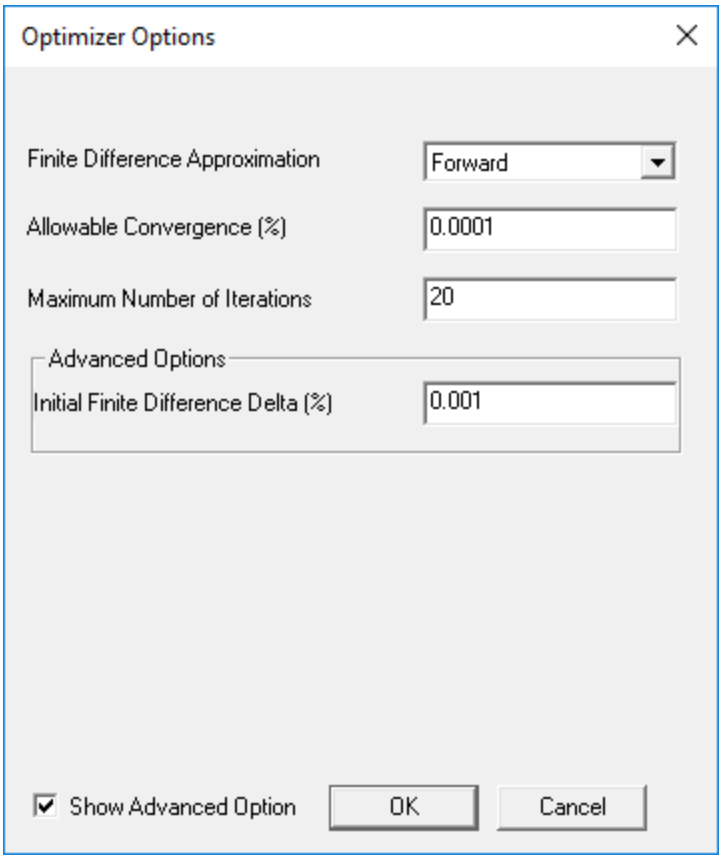
To generate samples and perform an NLPQL optimization:

1. Set up the variables you want to optimize in the **Design Properties** dialog box. The variables must be swept in a Parametric setup.
2. Click **HFSS** or **Q3D Extractor** or **2D Extractor** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Nonlinear Programming by Quadratic Lagrangian (Gradient)** from the **Optimizer** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. Optionally, click **Setup** to open the **Optimizer Options** dialog box.



- **Finite Difference Approximation** – When analytical gradients are not available, NLPQL approximates them numerically. This property lets you specify the method of approximating the gradient of the objective function. Choices are:
  - **Central** – Increases the accuracy of the gradient calculations by sampling from both sides of the sample point, but increases the number of design point evaluations by 50%. This method makes use of the initial point, as well as the forward point and rear point.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

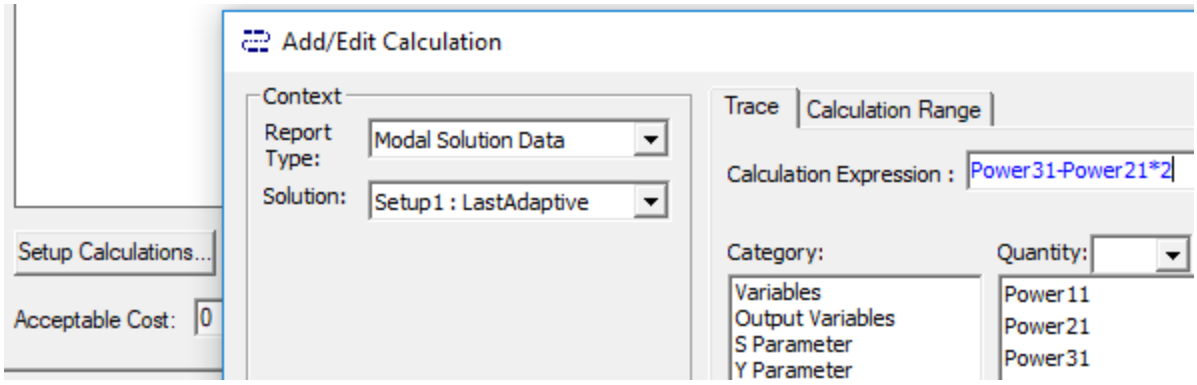
  - **Forward** – Uses fewer design point evaluations, but decreases the accuracy of the gradient calculations. This method makes use of only two design points, the initial point and forward point, to calculate the slope forward. This is the default method for new Direct Optimization systems.
- **Maximum Number of Iterations** – Stop criterion. Maximum number of iterations that the algorithm is to execute. If convergence happens before this number is reached, the iterations stop. This also provides an idea of the maximum possible number of function evaluations that are needed for the full cycle. For NLPQL, the number of evaluations can be approximated according to the Finite Difference Approximation gradient calculation method, as follows:
  - For **Central**: number of iterations * (2*number of inputs +1)
  - For **Forward**: number of iterations * (number of inputs+1)
- **Allowable Convergence (%)** – Stop criterion. Tolerance to which the Karush-Kuhn-Tucker (KKT) optimality criterion is generated during the NLPQL process. A smaller value indicates more convergence iterations and a more accurate (but slower) solution. A larger value indicates fewer convergence iterations and a less accurate (but faster) solution. For a Direct Optimization system, the default percentage value is **0.1**. The maximum percentage value is **100.** These values are consistent across all problem types because the inputs, outputs, and gradients are scaled during the NLPQL solution.

- **Initial Finite Difference Delta (%)** – Advanced option for specifying the relative variation used to perturb the current point to compute gradients. Used in conjunction with Allowable Convergence (%) to ensure that the delta in NLPQL's calculation of finite differences is large enough to be seen above the noise in the simulation problem. This wider sampling produces results that are more clearly differentiated so that the difference is less affected by solution noise and the gradient direction is clearer. The value should be larger than both the value for Initial Finite Difference Delta (%) and the noise magnitude of the model. However, smaller values produce more accurate results, so set Initial Finite Difference Delta (%) only as high as necessary to be seen above simulation noise.

- The default percentage value is **1**. The minimum is **0.0001**, and the maximum is **1**. For parameters with Allowed Values set to **Manufacturable Values** or Snap to Grid, the value for Initial Finite Difference Delta (%) is ignored. In such cases, the closest allowed value is used to determine the finite difference delta.

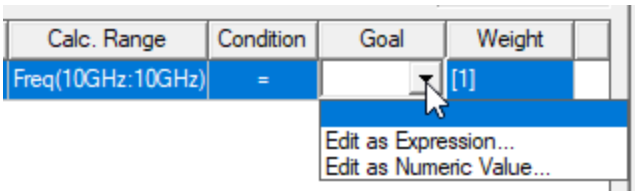
5. Add a cost function - Click **Setup Calculations** to open the **Add/Edit Calculation** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

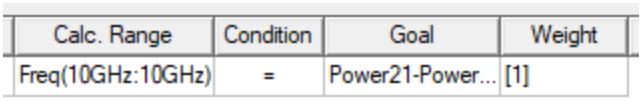


When you have created the calculation, click **Add Calculation** to add it to the **Optimization** setup, and **Done** to close the **Add/EditCalculation** dialog box.

6. In the Optimization setup, in the drop-down list for the **Goal** column, select **Edit as Expression** or **Edit as Numeric Value**.



7. This reopens the **Add/Edit Calculation** dialog box. If you are satisfied with the expression or value displayed, click **Done** to close the dialog box. This enters the expression/value to the **Goal** column.



8. In the **Optimization** setup, if you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

     A norm is a function that assigns a positive value to the cost function.

     For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors, which

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

means that it is always less than zero. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

The norm type doesn't impact goal setting that use as condition the "minimize" or "maximize" scenarios.

9. Optionally, set the Acceptable Cost and Cost Function Noise.
10. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.
11. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization.
    - You can also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.
    - Optionally, set Linear constraints.
    - Select the **View all columns** check box to see all columns, including hidden columns.
12. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

    Select the **Update design parameters' value after optimization** check box to cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.
13. Under the **Options tab**, if you want to save the field solution data for every solved design variations in the optimization analysis, select **Save Fields And Mesh**.

> **Note:**
>
> Do not select this option when requesting a large number of iterations as the data generated will be very large and the system may become slow due to the large I/O requirements.

You may also select **Copy geometrically equivalent meshes** to reuse the mesh when geometry changes are not required, for example when optimizing on a material property or source excitation. This will provide some speed improvement in the optimization process.

Related Topics

Convergence Rate % and Initial Finite Difference Delta % in NLPQ and MISQP

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

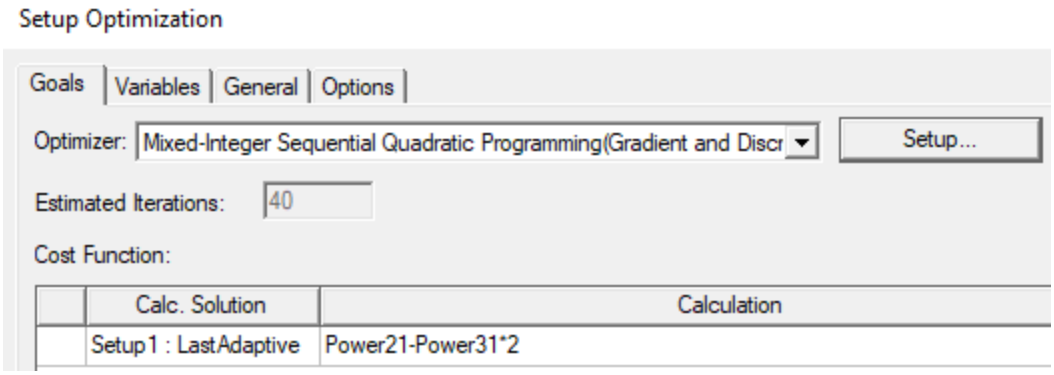
Setting Up Mixed-Integer Sequential Quadratic Programming (Gradient and Discrete) Optimizer

Follow this procedure to set up an optimization analysis using the Mixed-Integer Sequential Quadratic Programming optimizer. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

The Mixed-Integer Sequential Quadratic Programming (Nonlinear Programming by Quadratic Lagrangian) method (MISQP) can be used for Direct Optimization systems. It lets you generate a new sample set to provide a more refined approach than the Screening method. MISQP is available for both continuous and discrete input parameters, which is why "mixed" is in its name. MISQP can handle only one output parameter goal. Other output parameters can be defined as constraints. For more information, see Convergence Rate % and Initial Finite Difference Delta % in NLPQL and MISQP and Nonlinear Programming by Quadratic Lagrangian (NLPQL).

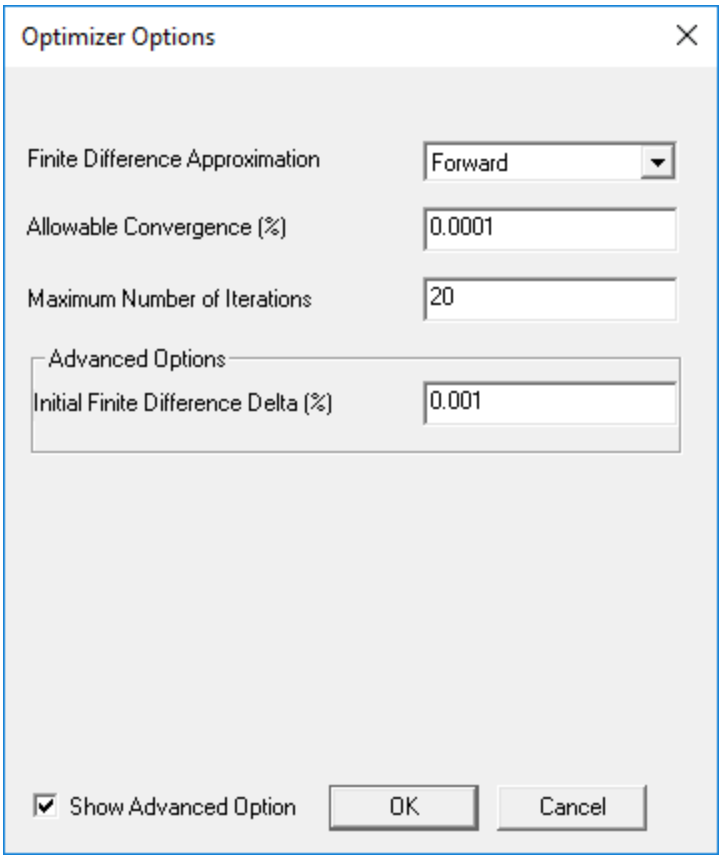
To generate samples and perform an NLPQL optimization:

1. Set up the variables you want to optimize in the **Design Properties** dialog box. The variables must be swept in a Parametric setup.
2. Click **HFSS** or **Q3D Extractor** or **2D Extractor** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Mixed-Integer Sequential Quadratic Programming (Gradient and Discrete)** from the **Optimizer** drop-down list.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Optionally click **Setup** to open the **Optimizer Options** dialog box.



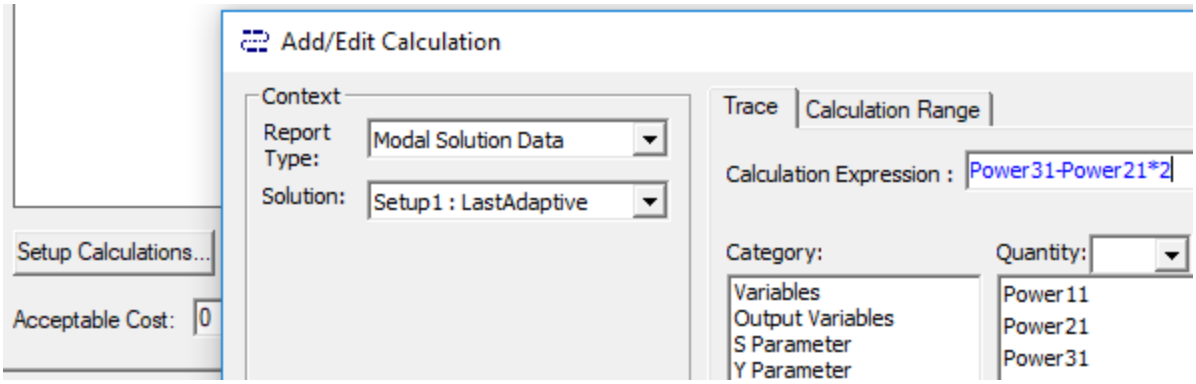
- **Finite Difference Approximation** – When analytical gradients are not available, MISQP approximates them numerically. This property lets you specify the method of approximating the gradient of the objective function. Choices are:
  - **Central** – Increases the accuracy of the gradient calculations by sampling from both sides of the sample point, but increases the number of design point evaluations by 50%. This method makes use of the initial point, as well as the forward point and rear point.
  - **Forward** – Uses fewer design point evaluations, but decreases the accuracy of the gradient calculations. This method makes use of only two design points—the initial point and forward point—to calculate the slope forward. This is the default method for new Direct Optimization systems.
- **Maximum Number of Iterations** – Stop criterion. Maximum number of iterations that the algorithm is to execute. If convergence happens before this number is reached, the iterations stop. This also provides an idea of the maximum possible number of function evaluations that are needed for the full cycle. For MISQP, the number of evaluations can be approximated according to the Finite Difference Approximation gradient calculation method, as follows:
  - For **Central**: number of iterations * (2*number of inputs +1)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

  - For **Forward**: number of iterations * (number of inputs+1)
- **Allowable Convergence (%)** – Stop criterion. Tolerance to which the Karush-Kuhn-Tucker (KKT) optimality criterion is generated during the MISQP process. A smaller value indicates more convergence iterations and a more accurate (but slower) solution. A larger value indicates fewer convergence iterations and a less accurate (but faster) solution. For a Direct Optimization system, the default percentage value is **0.1**. The maximum percentage value is **100**. These values are consistent across all problem types because the inputs, outputs, and gradients are scaled during the MISQP solution.
- **Initial Finite Difference Delta (%)** – Advanced option for specifying the relative variation used to perturb the current point to compute gradients. Used in conjunction with Allowable Convergence (%) to ensure that the delta in MISQP's calculation of finite differences is large enough to be seen above the noise in the simulation problem. This wider sampling produces results that are more clearly differentiated so that the difference is less affected by solution noise and the gradient direction is clearer. The value should be larger than both the value for Initial Finite Difference Delta (%) and the noise magnitude of the model. However, smaller values produce more accurate results, so set Initial Finite Difference Delta (%) only as high as necessary to be seen above simulation noise.

  The default percentage value is **1**. The minimum is **0.0001**, and the maximum is **1**. For parameters with Allowed Values set to **Manufacturable Values** or Snap to Grid, the value for Initial Finite Difference Delta (%) is ignored. In such cases, the closest allowed value is used to determine the finite difference delta.

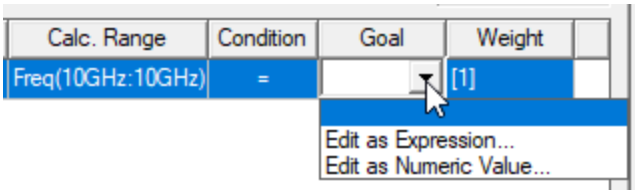
5. Add a cost function - Click **Setup Calculations** to open the **Add/Edit Calculation** dialog box.



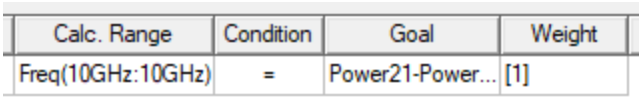
When you have created the calculation, click **Add Calculation** to add it to the **Optimization** setup, and **Done** to close the **Add/EditCalculation** dialog box.

6. In the Optimization setup, in the **Goal** column drop-down list, select **Edit as Expression** or **Edit as Numeric Value**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



7. This reopens the **Add/Edit Calculation** dialog box. If you are satisfied with the expression or value displayed, click **Done** to close the dialog box. This enters the expression/value to the **Goal** column.



8. In the **Optimization** setup, if you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

     A norm is a function that assigns a positive value to the cost function.

     For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors, which means that it is always less than zero. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

     The norm type doesn't impact goal setting that use as condition the "minimize" or "maximize" scenarios.
9. Optionally, set the Acceptable Cost and Cost Function Noise.
10. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.
11. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization.
    - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.
    - Optionally, set Linear constraints.
    - Select the **View all columns** check box to see all columns, including hidden columns.
12. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Select the **Update design parameters' value after optimization** check box to cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.

13. Under the **Options tab**, if you want to save the field solution data for every solved design variations in the optimization analysis, select **Save Fields And Mesh**.

> **Note:**
>
> Do not select this option when requesting a large number of iterations as the data generated will be very large and the system may become slow due to the large I/O requirements.

You may also select **Copy geometrically equivalent meshes** to reuse the mesh when geometry changes are not required, for example when optimizing on a material property or source excitation. This will provide some speed improvement in the optimization process.

Related Topics

Mixed-Integer Sequential Quadratic Programming

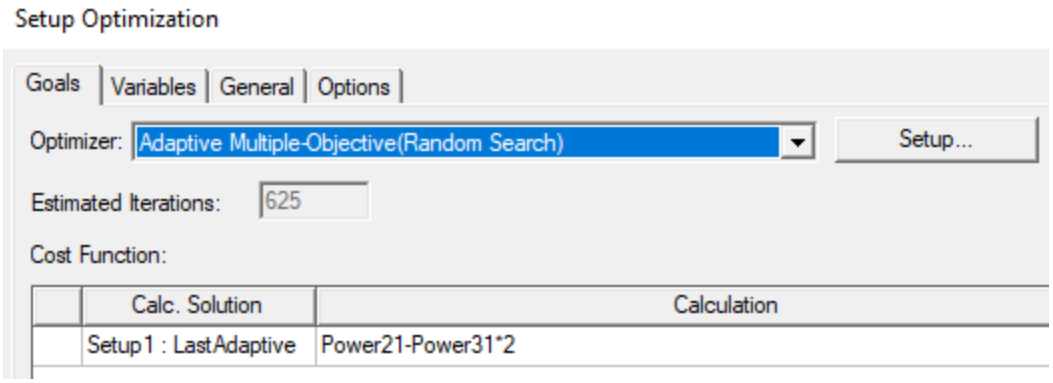
Setting Up Adaptive Multiple-Objective (Random Search) Optimizer

Follow this procedure to set up an optimization analysis using the Adaptive Multiple Objective (Random Search) Optimizer. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

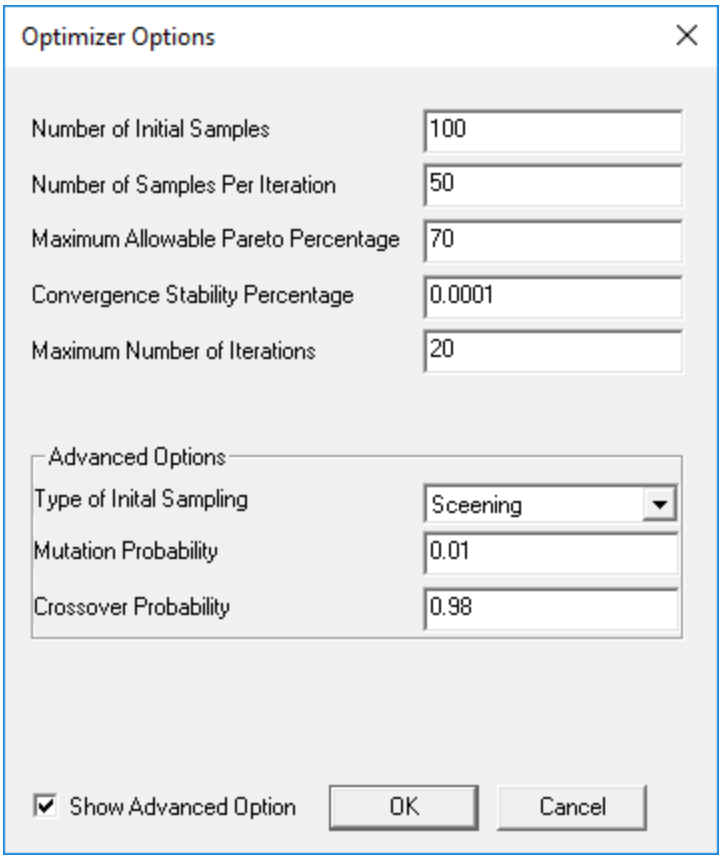
The Adaptive Multiple Objective (Kriging + MOGA) is an iterative algorithm that lets you generate a new sample set or use an existing set, providing a more refined approach than the Screening method. It uses the same general approach as MOGA, but applies the Kriging error predictor to reduce the number of evaluations needed to find the global optimum. The Adaptive Multiple-Objective method is available only for continuous input parameters, including those with manufacturable values. It can handle multiple objectives and multiple constraints. For more information, see ***Adaptive Multiple-Objective Optimization***.

1. Set up the variables you want to optimize in the **Design Properties** dialog box. The variables must be swept in a Parametric setup.
2. Click **HFSS** or **Q3D Extractor**or **2D Extractor** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Adaptive Multiple Objective (Random-search)** from the **Optimizer** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. Optionally click **Setup** to open the **Optimizer Options** dialog box.



- **Number of Initial Samples** – Initial number of samples to use. This number must be greater than the number of enabled input parameters. The minimum recommended number of initial samples is 10 times the number of enabled input parameters. The larger the initial sample set, the better your chances of finding the input parameter space that contains the best solutions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The number of enabled input parameters is also the minimum number of samples required to generate the Sensitivities chart. You can enter a minimum of **2** and a maximum of **10000**. The default is **100**.

If you switch from the Screening method to the MOGA method, MOGA generates a new sample set. For the sake of consistency, enter the same number of initial samples as you used for the Screening method.

- **Number of Samples Per Iteration** – Number of samples to iterate and update with each iteration. This number must be greater than the number of enabled input parameters but less than or equal to the number of initial samples. The default is for a Direct Optimization system.

You can enter a minimum of **2** and a maximum of **10000.**

- **Maximum Allowable Pareto Percentage** – Convergence criterion. Percentage value that represents the ratio of the number of desired Pareto points to the number of samples per iteration. When this percentage is reached, the optimization is converged. For example, a value of **70** with Number of Samples Per Iteration set to **200** would mean that the optimization should stop once the resulting front of the MOGA optimization contains at least **140** points. Of course, the optimization stops before that if the maximum number of iterations is reached.

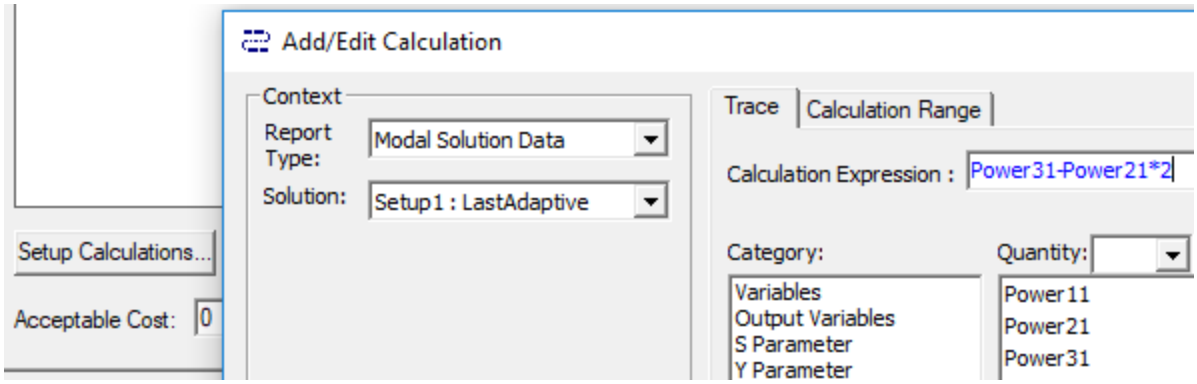
  If the **Maximum Allowable Pareto Percentage** is too low (below 30), the process can converge prematurely. If the value is too high (above 80), the process can converge slowly. The value of this property depends on the number of parameters and the nature of the design space itself. The default is **70**. Using a value between **55** and **75** works best for most problems. For more information, see ***Convergence Criteria in MOGA-Based Multi-Objective Optimization***.
- **Convergence Stability Percentage** – Convergence criterion. Percentage value that represents the stability of the population based on its mean and standard deviation. This criterion lets you minimize the number of iterations performed while still reaching the desired level of stability. When the specified percentage is reached, the optimization is converged. The default percentage is **2**. To not take the convergence stability into account, set to **0**. For more information, see ***Convergence Criteria in MOGA-Based Multi-Objective Optimization***.
- **Maximum Number of Iterations** – Stop criterion. Maximum number of iterations that the algorithm is to execute. If this number is reached without the optimization having reached convergence, iterations stop. This also provides an idea of the maximum possible number of function evaluations that are needed for the full cycle, as well as the maximum possible time it can take to run the optimization. For example, the absolute maximum number of evaluations is given by:

Number of Initial Samples + Number of Samples Per Iteration * (Maximum Number of Iterations - 1)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

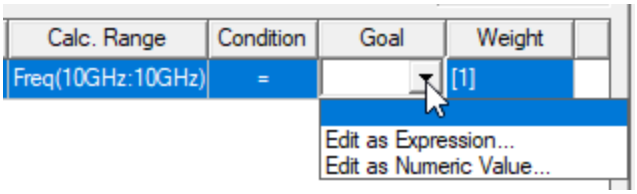
- **Type of Initial Sampling** – Advanced option for generating different kinds of sampling. If you do not have any parameter relationships defined, set to **Screening** (default) or **Optimal Space-Filling**. If you have parameter relationships defined, the initial sampling must be performed by the constrained sampling algorithms because parameter relationships constrain the sampling. For such cases, this property is set to Constrained Sampling.
- **Mutation Probability** – Advanced option for specifying the probability of applying a mutation on a design configuration. The value must be between **0** and **1**. A larger value indicates a more random algorithm. If the value is **1**, the algorithm becomes a pure random search. A low probability of mutation (<0.2) is recommended. The default is **0.01**. For more information on mutation, see ***MOGA Steps to Generate a New Population***.
- **Crossover Probability** – Advanced option for specifying the probability with which parent solutions are recombined to generate offspring solutions. The value must be between **0** and **1**. A smaller value indicates a more stable population and a faster (but less accurate) solution. If the value is **0**, the parents are copied directly to the new population. A high probability of crossover (>0.9) is recommended. The default is **0.98**.

5. Add a cost function - Click **Setup Calculations** to open the **Add/Edit Calculation** dialog box.



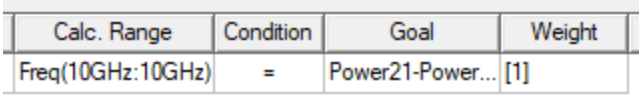
When you have created the calculation, click **Add Calculation** to add it to the **Optimization** setup, and **Done** to close the **Add/EditCalculation** dialog box.

6. In the Optimization setup, in the **Goal** column drop-down list, select **Edit as Expression** or **Edit as Numeric Value**.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7. This reopens the **Add/Edit Calculation** dialog box. If you are satisfied with the expression or value displayed, click **Done** to close the dialog box. This enters the expression/value to the **Goal** column.



Calc. Range	Condition	Goal	Weight
Freq(10GHz:10GHz)	=	Power21-Power...	[1]

8. In the **Optimization** setup, if you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

   A norm is a function that assigns a positive value to the cost function.

   For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors, which means that it is always less than zero. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

   The norm type doesn't impact goal setting that use as condition the "minimize" or "maximize" scenarios.

9. Optionally, set the Acceptable Cost and Cost Function Noise.
10. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.

11. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization.
    - You may also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.
    - Optionally, set Linear constraints.
    - Select the **View all columns** check box to see all columns, including hidden columns.
12. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

Selecting the **Update design parameters' value after optimization** check box will cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.

13. Under the **Options tab**, if you want to save the field solution data for every solved design variations in the optimization analysis, select **Save Fields And Mesh**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Do not select this option when requesting a large number of iterations as the data generated will be very large and the system may become slow due to the large I/O requirements.

You may also select **Copy geometrically equivalent meshes** to reuse the mesh when geometry changes are not required, for example when optimizing on a material property or source excitation. This will provide some speed improvement in the optimization process.

Related Topics

Set up Adaptive Multiple-Objective (Random Search) Optimizer

Setting Up Adaptive Single-Objective (Gradient) Optimizer

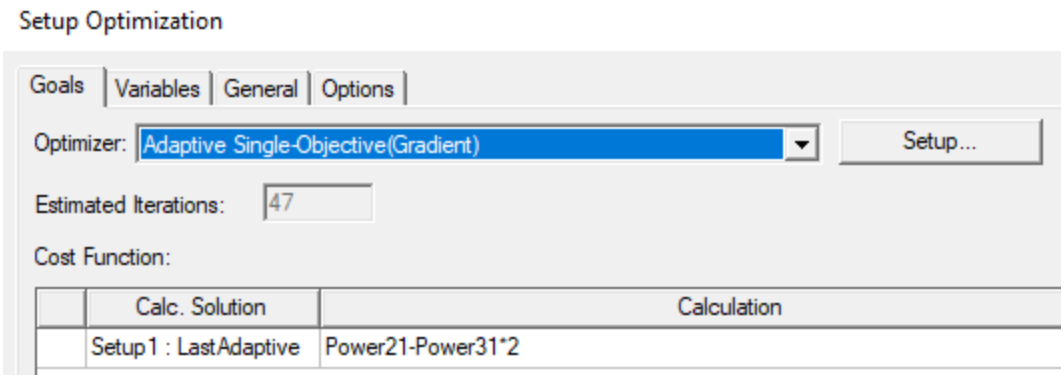
Follow this procedure to set up an optimization analysis using the Adaptive Single-Objective (OSF + Kriging + MISQP) Optimizer. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

This gradient-based method employs automatic intelligent refinement to provide the global optima. It requires a minimum number of design points to build the Kriging response surface, but in general this method reduces the number of design points necessary for the optimization. Failed design points are treated as inequality constraints, making it fault-tolerant.

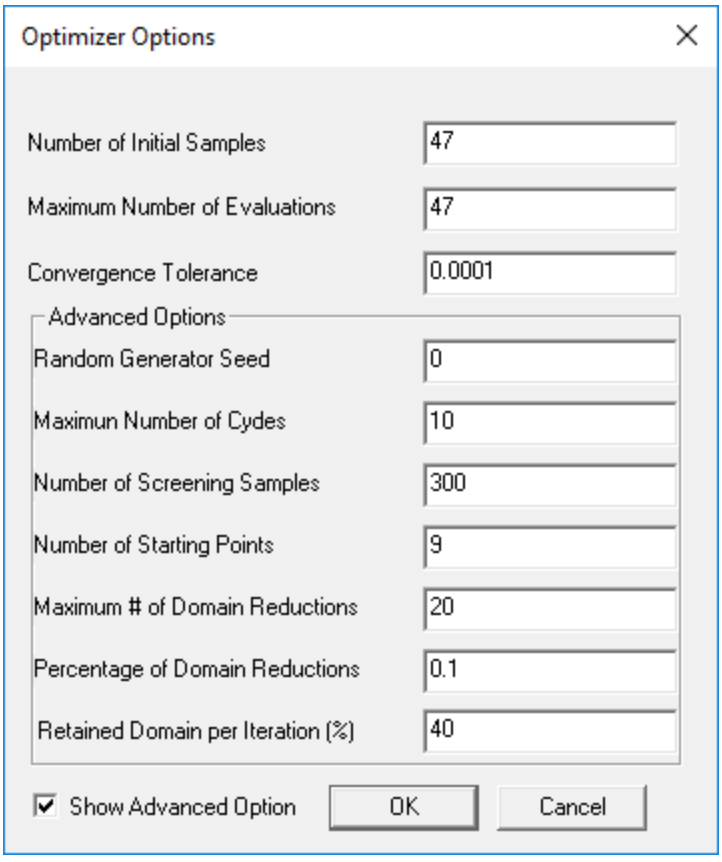
The Adaptive Single-Objective method is available for input parameters that are continuous, including those with manufacturable values. It can handle only one output parameter goal, although other output parameters can be defined as constraints. It does not support the use of parameter relationships in the optimization domain. For more information, see ***Adaptive Single-Objective Optimization***. It requires advanced options. Ensure that the **Show Advanced Options** check box is selected.

1. Set up the variables you want to optimize in the **Design Properties** dialog box. The variables must be swept in a Parametric setup.
2. Click **HFSS** or **Q3D Extractor** or **2D Extractor** > **Optimetrics Analysis** > **Add Screening & Optimization** . The **Setup Optimization** dialog box appears.
3. Under the **Goals** tab, select the optimizer by selecting **Adaptive Single Objective (Gradient)** from the **Optimizer** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



4. Optionally click **Setup** to open the **Optimizer Options** dialog box and select **Advanced Options**.



- **Number of Initial Samples** – Number of samples generated for the initial Kriging and after all domain reductions for the construction of the next Kriging. You can enter a minimum of **(NbInp+1)*(NbInp+2)/2** (also the minimum number of OSF samples required for the Kriging construction) or a maximum of **10,000**. The default is **(NbInp+1)*(NbInp+2)/2**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Because of the Adaptive Single-Objective workflow (in which a new OSF sample set is generated after each domain reduction), increasing the number of OSF samples does not necessarily improve the quality of the results and significantly increases the number of evaluations.

  - **Maximum Number of Evaluations** – Stop criterion. Maximum number of evaluations (design points) that the algorithm is to calculate. If convergence occurs before this number is reached, evaluations stop. This value also provides an idea of the maximum possible time it takes to run the optimization. The default is **20*(NbInp +1).**

- **Convergence Tolerance** – Stop criterion. Minimum allowable gap between the values of two successive candidates. If the difference between two successive candidates is smaller than the value for Convergence Tolerance multiplied by the maximum variation of the parameter, the algorithm is stopped. A smaller value indicates more convergence iterations and a more accurate (but slower) solution. A larger value indicates fewer convergence iterations and a less accurate (but faster) solution. The default is **1E-06**.
- **Random Generator Seed** – The value for initializing the random number generator invoked internally by OSF. The value must be a positive integer. This property lets you generate different samplings by changing the value or to regenerate the same sampling by keeping the same value. The default is **0**.
- **Maximum Number of Cycles** – Number of optimization loops that the algorithm needs, which in turn determines the discrepancy of the OSF. The optimization is essentially combinatorial, so a large number of cycles slows down the process. However, this makes the discrepancy of the OSF smaller. The value must be greater than **0**. For practical purposes, 10 cycles is usually good for up to 20 variables. The default is **10**.
- **Number of Screening Samples** – Number of samples for the screening generation on the current Kriging. This value is used to create the next Kriging (based on error prediction) and verified candidates.

  You can enter a minimum of **(NbInp+1)*(NbInp+2)/2** (also the minimum number of OSF samples required for the Kriging construction) or a maximum of **10,000**. The default is **100*NbInp** for a Direct Optimization system. There is no default for a Response Surface Optimization system.

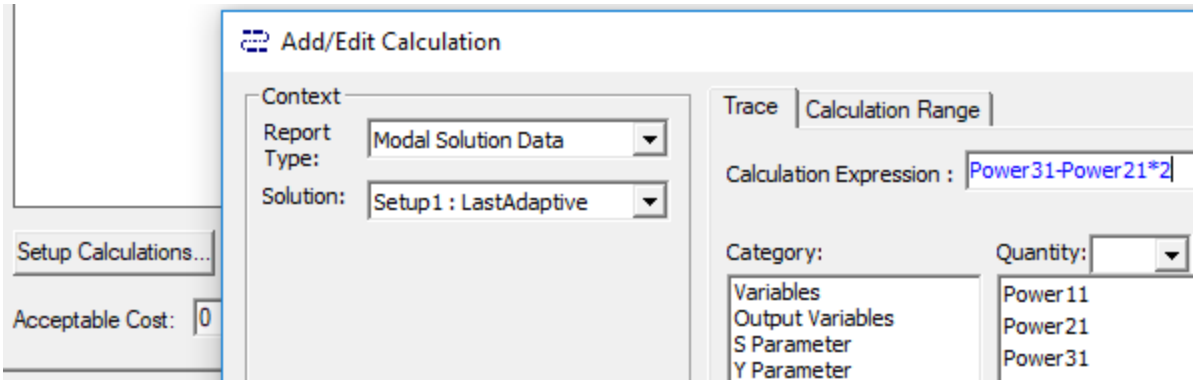
  The larger the screening sample set, the better the chances of finding good verified points. However, too many points can result in a divergence of the Kriging.

- **Number of Starting Points** – Determines the number of local optima to explore. The larger the number of starting points, the more local optima explored. In the case of a linear surface, for example, it is not necessary to use many points. This value must be less than the value for **Number of Screening Samples** because these samples are selected in this sample. The default is the value for **Number of Initial Samples**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

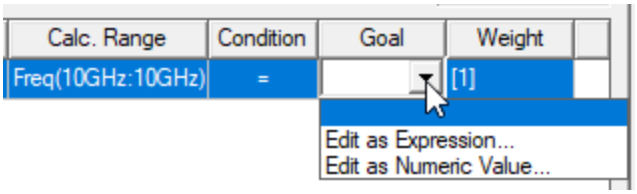
- **Maximum Number of Domain Reductions** – Stop criterion. Maximum number of domain reductions for input variation. (No information is known about the size of the reduction beforehand.) The default is **20**.
- **Percentage of Domain Reductions** – Stop criterion. Minimum size of the current domain according to the initial domain. For example, with one input ranging between **0** and **100**, the domain size is equal to **100**. The percentage of domain reduction is 1%, so the current working domain size cannot be less than 1 (such as an input ranging between 5 and 6). The default is **0.1**.
- **Retained Domain per Iteration (%)** – Advanced option that lets you specify the minimum percentage of the domain you want to keep after a domain reduction. The percentage value must be between **10** and **90**. A larger value indicates less domain reduction, which implies better exploration but a slower solution. A smaller value indicates a faster and more accurate solution, with the risk of it being a local one. The default percentage value is **40**.

5. Add a cost function - Click **Setup Calculations** to open the **Add/Edit Calculation** dialog box.



When you have created the calculation, click **Add Calculation** to add it to the **Optimization** setup, and **Done** to close the **Add/EditCalculation** dialog box.

6. In the Optimization setup, in the **Goal** column drop-down list, select **Edit as Expression** or **Edit as Numeric Value**.



7. This reopens the **Add/Edit Calculation** dialog box. If you are satisfied with the expression or value displayed, click **Done** to close the dialog box. This enters the expression/value to

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the **Goal** column.

Calc. Range	Condition	Goal	Weight
Freq(10GHz:10GHz)	=	Power21-Power...	[1]

8. In the **Optimization** setup, if you want to select a **Cost Function Norm Type**:
   - Select the **Show Advanced Option** check box. The **Cost Function Norm Type** drop-down list appears.
   - Select **L1**, **L2**, or **Maximum**.

     A norm is a function that assigns a positive value to the cost function.

     For **L1** norm the actual cost function uses the sum of absolute weighted values of the individual goal errors. For **L2** norm (the default) the actual cost function uses the weighted sum of squared values of the individual goal error. For the Maximum norm the cost function uses the maximum among all the weighted goal errors, which means that it is always less than zero. (For further details, see ***Explanation of the L1, L2, and Max Norms in Optimization***.)

     The norm type doesn't impact goal setting that uses as condition the "minimize" or "maximize" scenarios.
9. Optionally, set the Acceptable Cost and Cost Function Noise.
10. Optionally, click **HPC and Analysis Options** to select or create an analysis configuration.
11. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization.
    - You can also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.
    - Optionally, modify the values of fixed variables that are not being optimized.
    - Optionally, set Linear constraints.
    - Select the **View all columns** check box to see all columns, including hidden columns.
12. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

    Select the **Update design parameters' value after optimization** check box to cause Optimetrics to modify the variable values in the nominal design to match the final values from the optimization analysis.
13. Under the **Options tab**, if you want to save the field solution data for every solved design variations in the optimization analysis, select **Save Fields And Mesh**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Do not select this option when requesting a large number of iterations as the data generated will be very large and the system may become slow due to the large I/O requirements.

You may also select **Copy geometrically equivalent meshes** to reuse the mesh when geometry changes are not required, for example when optimizing on a material property or source excitation. This will provide some speed improvement in the optimization process.

Related Topics

Adaptive Single Objective Optimization

Setting the Maximum Iterations for an Optimization Analysis

The **Max. No. of Iterations** value is the maximum number of design variations that you want Optimetrics to solve during an optimization when using the **Sequential Nonlinear Programming (Gradient)**, **Sequential Mixed Integer NonLinear Programming**, **Quasi-Newton (Gradient)**, or **Pattern Search (Search-based)** optimizer. This value is a stopping criterion; if the maximum number of iterations has completed, the optimization analysis stops. If the maximum number of iterations has not completed, the optimization continues by performing another iteration, that is, by solving another design variation.

If the maximum number of iterations has not been reached, the optimizer performs iterations until the acceptable cost function is reached or until the optimizer cannot proceed as a result of other optimization setup constraints, such as when it searches for a variable value with a step size smaller than the minimum step size.

> **Note:**
>
> The **Genetic Algorithm** optimizer does not use the **Max. No. of Iterations** criteria.

To set the maximum number of iterations for an optimization analysis:

1. Open the **Setup Optimization** dialog box.
2. Click the **Goals** tab.
3. Type a value in the **Max. No. of Iterations** text box.

Related Topics

Adding a Cost Function

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Cost Function

Optimetrics manipulates the model's design variable values to find the minimum location of the cost function; therefore, you should define the cost function so that a minimum location is also the optimum location.

When using the quasi-Newton optimizer, which is appropriate for designs that are not sensitive to noise, the best cost function is a smooth, second-order function that can be approximated well by quadratics in the vicinity of the minimum; the slope of the cost function should decrease as Optimetrics approaches the optimum value. The preferred cost function takes values between **0** and **1**. In practice, most functions that are smooth around the minimum are acceptable as cost functions. Most importantly, the cost function should not have a sharp dip or pole at the minimum. A well designed cost function can significantly reduce the optimization process time.

The cost function is defined in the **Setup Optimization** dialog box or the **Design of Experiments** setup when you set up an optimization analysis. If you know the exact syntax of the solution quantity on which you want to base the cost function, you can type it directly in the **Calculation** text box. You can also use **Setup Calculations** to add a solution quantity via the **Add/Edit Calculation** dialog box, or to create an output variable that represents the solution quantity in the **Output Variables** dialog box.

Related Topics

Adding a Cost Function

Acceptable Cost

Cost Function Noise

Linear Constraints

Goal Weight

Step Size

Explanation of L1, L1, Norm Costs in Optimization

Setting Up Design of Experiments

Acceptable Cost

The acceptable cost is the value of the cost function at which the optimization process should stop; it is also known as the *stopping criterion*. The cost function value must be equal to or below the acceptable cost value for the optimization analysis to stop. The acceptable cost may be a negative value.

Related Topics

Cost Function

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding a Cost Function

Cost Function Noise

Numerical analysis introduces various sources of noise to the cost function. You must provide the optimizer with an estimate of the noise. The noise indicates whether a change during the solution process is significant enough to support achievement of the cost function.

For example, if the cost function *c* is

$$c = 10000 \cdot |L_{11}|^2$$

where $|L_{11}|$ is the magnitude of the inductance, at the minimum, $|L_{11}|$ is expected to be very small, $|L_{11}| \approx 0$.

From the solution setup, the error in $|L_{11}|$ is expected to be $E_{L11} \approx 0.01$. The perturbed cost function is therefore

$$c_{perturbed} = 10000 \cdot (|L_{11}|_{min} + E_{L11})^2$$

Near the minimum, the error in the cost function E_c is given by

$$E_c = c_{perturbed} - c_{min} = 10000 \cdot (0.0 + 0.01)^2 - (10000 \cdot 0.0) = 1.0$$

Therefore, the cost function noise would be 1.0.

Related Topics

Cost Function

Adding a Cost Function

A cost function can include one or more goals for an optimization analysis. Optimetrics manipulates the model's design variable values to fulfill the cost function. The optimization stops

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

when the solution quantity meets the acceptable cost criterion.

Following is the general procedure for adding a cost function with a single goal:

1. Under the **Goals** tab of the **Setup Optimization** dialog box or the **Design of Experiments** dialog box, click **Setup Calculations**. The **Add/Edit Calculation** dialog box appears.
2. In the **Add/Edit Calculation** dialog box, follow these steps to set up a cost function.
    a. Set the **Context** for the calculation.
    b. Choose the **Category** of available data type depending upon the **Solution** type of the design being optimized.
    c. Select the **Quantity** to add to the **Calculated Expression** field. Available quantities depend upon the **Category** selection.
    d. You may optionally make a selection from the function list to apply to the calculated expression.
    e. When the **Calculation Expression** has the desired equation, click **Add Calculation** to add the expression to the cost function table.
    f. Repeat to add additional calculations to the cost function or click **Done** to exit the **Add/Edit Calculation** dialog box and return to **Setup Optimization**.
3. To modify the **Solution** on which the calculation is based, click the **Solution** column and select the solution from which the cost function is to be extracted.
4. To edit the calculation on which to base the cost function goal, select **Edit** from the drop-down list.
5. In the **Condition** text box, select one of these conditions from the drop-down list:

<=	Less than or equal to.
=	Equal to.
>=	Greater than or equal to.
Minimize	Reduce the cost function to a minimum value.
Maximize	Identify a maximized condition.

6. In the **Goal** text box, type the value of the solution quantity you want to achieve during the optimization analysis. If the solution quantity is a complex calculation, the goal value must be complex; two goal values must be specified. The **Minimize** and **Maximize** options do not require you to specify a **Goal** value.
7. Optionally, if you have multiple goals and want to assign higher or lower priority to a goal, type a different value for the goal's weight in the **Weight** text box. The goal with the greater weight is given more importance. If the goal is a complex value, the weight value must be complex; two weight values must be specified. The weight value cannot be variable

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

dependent.

> **Note:**
>
> Click **Edit Goal/Weight** to open the **Edit Goal Value/Weight** dialog box. You can modify weights for all goals simultaneously, as well as set the **Goal Values** to expressions.

8. Optionally, **HPC and Analysis Options** to select or create an analysis configuration.
9. Specify other options (such as acceptable cost, noise, and number of passes) and click **OK**.

The optimization stops when the solution quantity meets the acceptable cost criterion.

Related Topics

Setting a Goal Value

Cost Function

Acceptable Cost

Goal Weight

Adding/Editing a Cost Function Calculation

Use the **Add/Edit Calculation** dialog box to define the mathematical equation for one or multiple cost functions. It represents the calculation to be performed on the optimization variables to compare to the goal values. To set up a calculation for a cost function:

1. In the **Context** section of the dialog box:
   - Select the **Report Type** from the drop-down list containing the available types for this design.
   - Select the **Solution** from the drop-down list. This lists the available setups and sweeps. As a minimum, the **LastAdaptive** solution is available.
   - Select the Geometry from the drop-down list or select none (the default). This modifies the list of quantities available to the ones that apply to the specific geometry.
2. Click **Output Variables** to open the **Output Variables** dialog box; use this dialog box to create special output variables to be used in the cost function.
3. The **Calculated Expression** field in the **Trace** tab is used to enter the equation to be used for the cost function. To enter an expression, type it directly into the field or use the **Category**, **Quantity**, and **Function** lists as follows:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Select the **Category**. These depend on the Solution type and the design. This lets you specify the category of information to be used in the cost function.
- Select a **Quantity** from the list. Available quantities depend upon the Solution type, as well as the Geometry and Category selection. Selecting a Quantity enters it into the **Calculated Expression** field.
- Select a **Function** to apply to the value in the calculated expression.
- For swept variables, the **Range Function** button opens the **Set Range Function** dialog box to apply functions to the expression that apply over the sweep range.

4. The **Calculation Range** tab applies to swept variables and lets you specify the range of the sweep over which to apply the calculation.
5. When the desired **Calculated Expression** is obtained, click **Add Calculation** to add the entry to the cost function table. Add multiple entries to the table by changing the **Calculated Expression** and using the **Add Calculation** button.
6. To update or edit a selected cost function, enter the desired Calculated Expression and click the **Update Calculation** button.
7. Click **Done** to return to the **Setup Calculations** dialog box.

Specifying a Solution Quantity for a Cost Function Goal

When setting up a cost function, you must identify the solution quantity on which to base each goal. Solution quantities are specified by mathematical expressions composed of basic quantities, such as matrix parameters and output variables.

1. Add a row (a goal) to the cost function table:
   a. Under the **Goals** tab of the **Setup Optimization** dialog box, click **Add**. A new row appears in the **Cost Function** table.
   b. In the **Solution** column, click the solution from which the cost function is to be extracted.
2. In the **Solution** text box, click the solution from which the solution quantity is to be extracted.
3. In the **Calculation** text box, specify the solution quantity in one of the following ways:
   - If you know the syntax of the mathematical expression or the output variable's name, type it in the **Calculation** text box.
   - If you want to create an output variable that represents the solution quantity, do the following:
   a. Click **Edit Calculation**. The **Output Variables** dialog box appears.
   b. Add the expression you want to evaluate and click **Done**.
   c. Click **Done** to close the **Output Variables** dialog box.
   d. In the **Setup Optimization** dialog box, the most recently created output variable appears in the **Calculation** text box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

e. To specify a different defined output variable, click the **Calculation** text box. It becomes a drop-down list that displays all of the defined output variables. Click an output variable from the drop-down list.

Setting the Calculation Range of a Cost Function Goal

The calculation range is the range within which you want a cost function goal to be calculated. It can be a single value or a range of values, depending on the solution or solution quantity selected for the goal.

1. Under the **Goals** tab in the **Setup Optimization** dialog box, click **Edit Cal. Range**.
2. In the **Variable** drop-down list, click a variable.

   If you chose to solve a parametric setup during the optimization analysis, the variables swept in that parametric setup are available in the **Variable** drop-down list. If you sweep a variable in the parametric setup that is also being optimized, that variable is excluded from the optimization.

   Other examples of available variables include frequency, if the solution quantity is an S-parameter quantity, and phi or theta, if the solution quantity is a radiated field quantity.

3. After you select a variable from the **Variable** drop-down list, you can select a range of values for the calculation range as follows:
   a. Select **Range**.
   b. In the **Start** text box, type the starting value of the range.
   c. In the **Stop** text box, type the final value of the range.
4. To select a single value for the calculation range:
   a. Select **Single Value**.
   b. In the **Value** text box, type the value of the variable at which the cost function goal is to be extracted.
5. Click **Update**, then click **OK**.

Setting a Goal Value

A goal is the value you want a solution quantity to reach during an optimization analysis. It can be a real value or a complex value. If the solution quantity is a complex calculation, the goal value must be complex. You can type the goal value in the **Goal** text box. Alternatively, you can use the **Edit Goal/Value Weight** dialog box to specify the goal value as a single value, a mathematical expression, or a value dependent on a variable such as frequency.

Related Topics

Specify a single goal value

Specify an expression as the goal value

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Specify a variable-dependent goal value

Specifying a Single Goal Value

1. Under the **Goals** tab in the **Setup Optimization** dialog box, click **Edit Goal/Weight**. The **Edit Goal/Weight** dialog box appears.
2. Under the **Goal Value** tab, select **Simple Numeric Value** from the **Type** list.
3. If the goal value is complex, select **real/imag** from the drop-down list to specify the real and imaginary parts of the goal value.

   Alternatively, click **mag/ang** to specify the magnitude and angle of the goal value.

4. Type the goal value in the **Goal Value** table.

   If the goal value is complex, type both parts of the goal value in the text box below the **Goal Value** heading. For example, type **1, 1** to specify the real part of the goal value as 1 and the imaginary part as 1.

   If the goal value is real, type a real goal value in the text box below the **Goal Value** heading.

5. Click **OK**. The goal value you specified appears in the **Goal** text box.

Specifying an Expression as a Goal Value

1. Under the **Goals** tab in the **Setup Optimization** dialog box, click **Edit Goal/Weight**. The **Edit Goal/Weight** dialog box appears.
2. Under the **Goal Value** tab, select **Expression** from the **Type** list.
3. If you know the syntax of the mathematical expression or the existing output variable's name, type it in the text box below the **Goal Value** heading.

   Alternatively, if you want to create an output variable that represents the goal value, do the following:

   a. Click **Edit Expression**. The **Output Variables** dialog box appears.
   b. Add the expression you want to be the goal value and click **Done**. Twin Builder enters the most recently created output variable in the text box below the **Goal Value** heading.

4. Click **OK**. The goal value you specified appears in the **Goal** text box.

Specifying a Variable-Dependent Goal Value

1. Under the **Goals** tab in the **Setup Optimization** dialog box, click **Edit Goal/Weight**. The **Edit Goal/Weight** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Under the **Goal Value** tab, select **Variable Dependent** from the **Type** list.
3. Select a variable from the drop-down list to the left of the table.
4. Type the value of that variable in the first column of the table.

> **Warning:**
>
> Variable values must be single real numbers, or expressions that evaluate to single real numbers. Complex numbers cannot be used as the values of variables in any optimetric analysis.

5. Type a corresponding goal value for that variable value in the text box below the **Goal Value** heading.
6. Click **Add** to add another row to the reference curve.
7. Repeat steps 4, 5, and 6 until you have specified the reference curve.
8. Click **OK**. The goal value is listed as being variable dependent in the **Goal** text box.

Goal Weight

If an optimization setup has a cost function made up of multiple goals, you can assign a different weight to each goal. The goal with the greater weight is given more importance during the cost calculation.

The error function value is a weighted sum of the sub-goal errors. Each sub-goal, at each frequency at which it is evaluated, gives rise to a (positive) error value that represents the discrepancy between the simulated response and the goal value limit. If the response satisfies the goal value limit, then the error value is **0**. Otherwise, the error value depends on the differences between the simulated response and the respective goal limit. The error function may be defined as follows:

$$\sum_{j}^{G} \frac{W_j}{N_j} \sum_{i}^{N_j} e_i$$

where

- G is the number of sub-goals.
- W_j is the weight factor associated with the j^{th} sub-goal.
- N_j is the number of frequencies for the j^{th} sub-goal.
- e_i is the error contribution from the j^{th} sub-goal at the i^{th} frequency.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The value of e_i is determined by the band characteristics, target value, and the simulated response value. The choices for band characteristics are <=, =, and >=.

Band Characteristics (Condition)	**e_i evaluation where s_i is the simulated response and g_i is the desired limit.**
<=	$e_i = \begin{cases} 0 & s_i \le g_i \\ s_i - g_i & s_i > g_i \end{cases}$
=	$e_i = \lvert s_i - g_i \rvert$
>=	$e_i = \begin{cases} 0 & s_i \ge g_i \\ g_i - s_i & s_i < g_i \end{cases}$

If the total error value is within the acceptable cost, the optimization stops.

Related Topics

Adding a Cost Function

Cost Function

Modifying the Starting Variable Value for Optimization

A variable's starting value is the first value solved during the optimization analysis. Optimetrics sets the starting value of a variable to be the current value set for the nominal design. You can modify this value for each optimization setup.

> **Note:**
>
> If you choose to solve a parametric setup before an optimization analysis, a variable's starting value is ignored if a more appropriate starting value is calculated during the parametric analysis.

1. In the **Setup Optimization** dialog box, click the **Variables** tab.

   All of the variables selected for the optimization analysis are listed.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Type a new value in the **Starting Value** text box for the value you want to override, then press **Enter**.

   The **Override** option is now selected. This indicates that the value you entered is used for this optimization analysis, and the current value set for the nominal model is ignored.

   - Alternatively, you can select the **Override** option and type a new variable value in the **Starting Value** text box.
3. Optionally, click a new unit system in one of the **Units** text boxes.

> **Note:**
>
> To revert to the default starting value, clear the **Override** check box.

Related Topics

Setting the Minimum and Maximum Variable Values for Optimization

Step Size

Setting the Min and Max Focus

Modifying the Starting Variable Value for Sensitivity Analysis

Modifying the Starting Variable Value for Statistical Analysis

Setting the Minimum and Maximum Variable Values for Optimization

For every optimization setup, Optimetrics sets the minimum and maximum values it will consider for a variable being optimized. Optimetrics sets a variable's minimum value equal to approximately 50% of its starting value. (The starting value is the variable's current value set for the nominal design.) Optimetrics sets the variable's maximum value equal to approximately 150% of the starting value. During the optimization analysis, variable values that lie outside of this range are not considered.

> **Warning:**
>
> Variable values must be single real numbers, or expressions that evaluate to single real numbers. Complex numbers cannot be used as the values of variables in any optimetric analysis.

Related Topics

Text Entry for Calc. Range or Edit Calculation Range Dialog

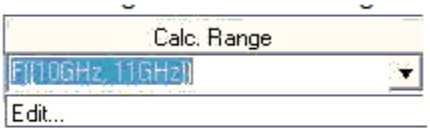
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Override the default min and max variable values for a single optimization setup

Change the default min and max variable values for every optimization setup

Text Entry for Calc. Range or Edit Calculation Range Dialog

In the **Setup Optimization** dialog box, select a line item in the **Cost Function** field. To enter the Calc. Range Sweep Min/Max value, click the **Calc. Range** field and enter a value.



You can also use the **Calc. Range** drop-down list and select **Edit**. The **Edit Calculation Range** dialog box appears. Click in this dialog box to specify a range.

The **Calc. Range** field accepts the following forms of text:

- sweep that lets you select different discrete values:

  discrete values, for example, **F(10GHz, 11GHz)**

  min/max range, for example, **F([10GHz, 11GHz])**

- editable sweep, which lets you customize values (that is, a sweep that has an enabled "edited" radio button in sweep selection dialog box):

  The min/max is used on top of selected values. For example, if you use the sweep dialog box and choose "0 deg, 60 deg, 180 deg, 240 deg", then [60deg, 240deg] will select values "60 deg, 180 deg, 240 deg".

- sweep that uses a full range:

  all values, for example, Time(All)

  min/max range, for example, Time([1ms, 2ms]) HFSS/MAXWELL/TWINBUILDER

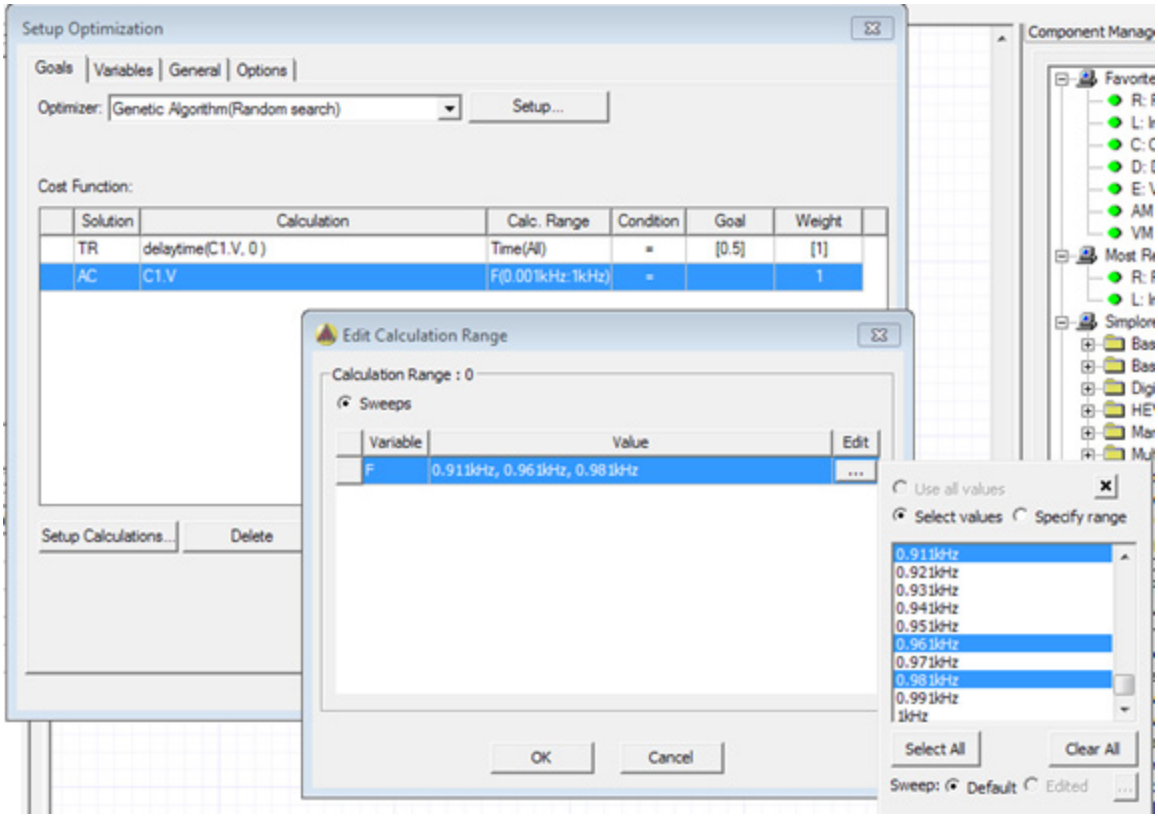
- You solve 1 to 20 GHz step .1 and specify F[10.381GHz, 11.381GHz]: it is equivalent to selecting values between 10.4GHz and 11.3GHz.
- You can specify multiple sweep values by separating those with a comma.

  For example, F(1GHz), cap(1pf, 1.2pf)

  For example, Distance(All), Freq([1ghz,2ghz]), Phase(0 deg)

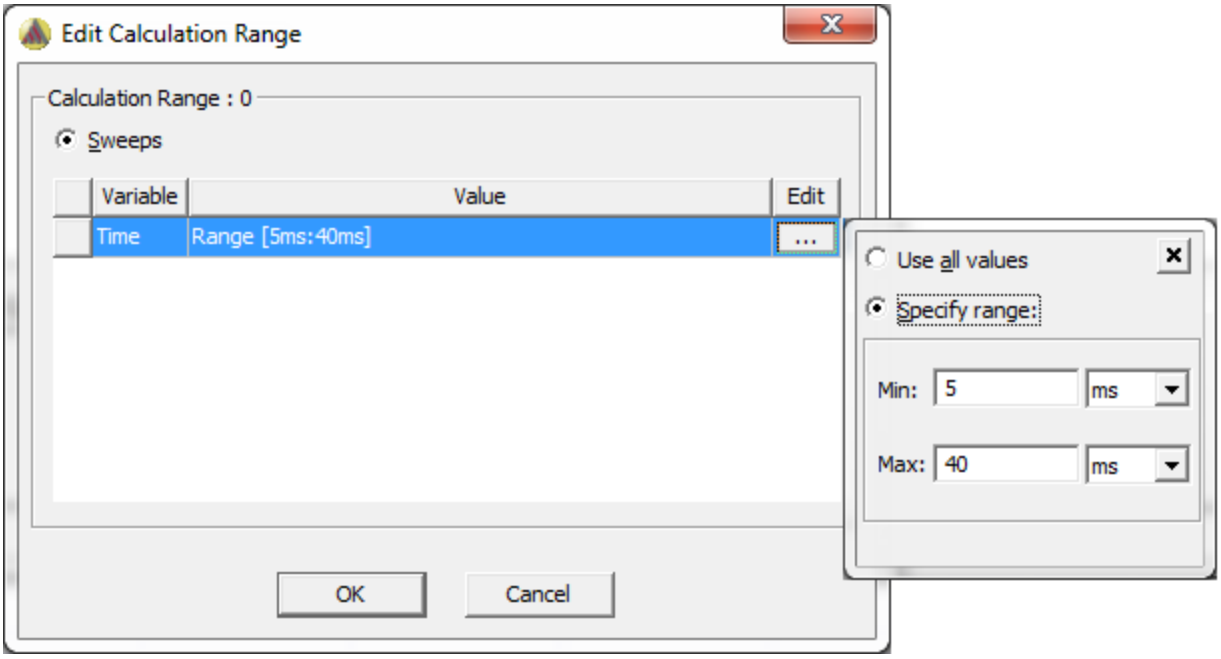
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Click **Calc. Range** > **Edit** to open the **Edit Calculation Range** dialog box. Click ... and select **Use all values**, **Select values**, or **Specify range**. The **Select values** option is available depending on the solution type.



The image below shows how the range appears in the **Calc. Range** field when you specify a range.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



You can also enter a range directly into the **Calc. Range** field.

Overriding the Min. and Max. Variable Values for a Single Optimization Setup

1. In the **Setup Optimization** dialog box, click the **Variables** tab. All of the variables selected for optimization analysis are listed.
2. Type a new value in the **Min** or **Max** text box for the value to override and press **Enter**.

   The **Override** option is now selected. This indicates that the value you entered is used for this optimization analysis; the variable's original **Min** or **Max** value in the nominal design is ignored.

   - Alternatively, you can select the **Override** option and type a new value in the **Min** or **Max** text box.
3. Optionally, click a new unit system in one of the **Units** text boxes.

   To revert to the default minimum and maximum values, clear the **Override** option.

Changing the Min. and Max. Variable Values for Every Optimization Setup

1. Make sure that the variable's minimum and maximum values are not being overridden in any single optimization setup.
2. If the variable is a design variable click **Twin Builder** > **Design Properties**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If the variable is a project variable click **Project** > **Project Variables**.

The **Properties** dialog box appears.

3. Select **Optimization**.
4. Type a new value in the **Min** or **Max** text box for the value you want to override and press **Enter**.
5. Click **OK**.

When Optimetrics solves an optimization setup, it does not consider variable values that lie outside of this range.

Step Size

To make the search for the minimum cost value reasonable, the search algorithm is limited in two ways. First, you do not want the optimizer to continue the search if the step size becomes irrelevant or small. This limitation impacts the accuracy of the final optimum. Second, in some cases you do not want the optimizer to take large steps either. In case the cost function is suspected to possess large variations in a relatively small vicinity of the design space, large steps may result in too many trial steps, which do not improve the cost value. In these cases, it is safer to proceed with limited size steps and have more frequent improvements.

For these two limitations, the optimizer uses two independent distance measures. Both are based on user-defined quantities: the minimum and maximum step limits for individual optimization variables. Since the particular step is in a general direction, these measures are combined together in order to derive the limitation for that particular direction.

The step vector between the i^{th} and $(i+1)^{th}$ iterate is as follows:

$$s_i = x_{i+1} - x_i$$

The natural distance measure is

$$\|s_i\| = \sqrt{s_i^T s_i}$$

which is the Euclidean norm.

A more general distance measure incorporates some “stretching” of the design space:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$\|s_i\|_D = \sqrt{s_i^T D^T D s_i}$$

where the matrix *D* incorporates the linear operation of the stretching of design space. The simplest case is when the *D* matrix is diagonal, meaning that the design space is stretched along the orthogonal direction of the base vectors.

The optimizer stops the search if

$$\|s_i\|_{D_{min}} < 1$$

where D_{min} consists of diagonal elements equal to the inverse of the **Min. Step** value assigned to the corresponding optimization variable. Similarly the optimizer truncates steps for which

$$\|s_i\|_{D_{max}} > 1$$

where D_{max} has diagonal elements equal to the inverse of **Max. Step** values of the corresponding optimization variables.

Related Topics

Setting the Min. and Max. Step Sizes

Cost Function

Adding a Cost Function

Setting the Min. and Max. Step Sizes

For the quasi-Newton and Pattern Search optimizers, the step size is the difference in a variable's value between one solved design variation and the next. The step size is determined when Optimetrics locates the next design variation that should be solved in an effort to meet the cost function.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. In the **Setup Optimization** dialog box, click the **Variables** tab.
2. Optimetrics displays **Min Step** and **Max Step** columns, with default values for each variable to be optimized.
3. In the **Min Step** text box, type the minimum step size value. Optionally, modify the unit system in the **Units** text box.
4. In the **Max Step** text box, type the maximum step size value. Optionally, modify the unit system in the **Units** text box.
5. Click **OK**.

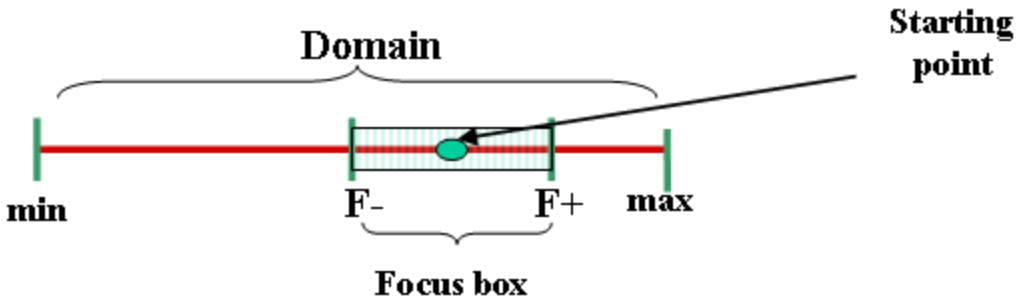
Hint	Use a value of zero for the minimum step size.

Related Topic

Step Size

Setting the Min and Max Focus

For the SNLP, SMINLP, and Genetic Algorithm optimizers, the minimum focus and maximum focus criteria let you specify a sub-range of parameter values where the optimizer should look when performing the optimization. This focus box is where you suspect the optimal solution will be, so it is a hint for the optimizer.



- The domain limits the search. The domain = physical limits.
- The focus box does not limit the search. Rather, the Focus box = an initial guess of optimum search domain. The starting point is the center of the focus box, but the search does extend beyond the box.
- This focus must be inside the domain limits. Consequently, it has to be equal or smaller size. An error message generates if you specify a focus outside the domain.
- The focus box must be at least one hundredth of the domain size. Otherwise an error message is sent.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Equalizing the influence of different optimization variables

The optimizer seeks optimal values for the optimization variables. These variables are usually quantities with specified units. The change in one variable could be measured in **mm** and the change in other variable could be measured in **mA**. Instead of those units, the optimizer uses internal abstract units, so that a change in one variable changes the design behavior about as much as the same change in another variable, where changes are measured in the respective internal abstract units. When you define the focus box, the unit of the abstract internal unit equals the difference between the upper and lower focus limits. This way you can use the focus box to equalize the influence of different optimization variables on the design behavior.

To set the Min and Max Focus values

1. In the **Setup Optimization** dialog box, click the **Variables** tab.
2. Optimetrics displays **Min. Focus** and **Max. Focus** columns, with default values for each optimized variable.

If you do not have an initial guess based on your knowledge of the problem, make the focus box equal to the domain – that is, the physical limits. This tells SNLP to search the entire decision space.

- In the **Min. Focus** text box, type the minimum value of the focus range. Optionally, modify the unit system in the **Units** text box.
- In the **Max. Focus** text box, type the maximum value of the focus range. Optionally, modify the unit system in the **Units** text box.
- Click **OK**.

Solving a Parametric Setup Before an Optimization

Solving a parametric setup before an optimization setup is useful for guiding Optimetrics during an optimization.

To solve a parametric setup before an optimization setup:

1. In the **Setup Optimization** dialog box, click the **General** tab.
2. In the **Parametric Analysis** drop-down list, click the parametric setup you want Optimetrics to solve before optimization.

> **Note:**
>
> The parametric setup must include sweep definitions for the variables you are optimizing.

3. Select **Solve the parametric sweep before optimization**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If the parametric setup has not yet been solved, Optimetrics solves it. Optimetrics uses the cost value evaluated at each parametric design variation to determine the next step in the optimization analysis. This lets you guide the direction in which the optimizer searches for the optimal design variation.

Related Topics

Solving a Parametric Setup During an Optimization

Solving a Parametric Setup During an Optimization

Solving a parametric setup during an optimization analysis is useful when you want Optimetrics to solve every design variation specified in the parametric setup at each optimization iteration. A cost function goal could then depend on the value of the variable swept in the parametric setup.

To solve a parametric setup during an optimization analysis:

1. In the **Setup Optimization** dialog box, click the **General** tab.
2. In the **Parametric Analysis** drop-down list, select a parametric setup for Optimetrics to solve during an optimization.
3. Select **Solve the parametric sweep during optimization**.
4. Optionally, you can adjust the sweep values to use during the optimization.
   a. Click the **Goals** tab and select **Setup Calculations** to specify a calculation. The **Add/Edit Calculation** dialog box appears.
   b. Click the **Calculation Range** tab.
   c. Click **Edit** for the sweep to be modified. A dialog box appears.
   d. Select the sweep values to use.
   e. Close the dialog box. Click **Done** to close the **Add/Edit Calculation** dialog box.

Automatically Updating a Variable's Value after Optimization

When Optimetrics finds an optimal variable value by solving an optimization setup, it updates that variable's current value set for the nominal model to the optimal value.

1. In the **Setup Optimization** dialog box, click the **General** tab.
2. Select **Update design parameters' values after optimization**.

When optimization is complete, the current variable value for each optimized variable changes to the optimal value.

Changing the Cost Function Norm

You can select the norm used in the calculation of the cost goal.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. In the **Setup Optimization** dialog box, click the **Goals** tab.
2. Select **Show Advanced Options**.
3. Select a norm from the drop-down list in the **Cost Function Norm Type** field. The options are **L1**, **L2**, and **Maximum**. **L2** is the default.

Related Topics

Explanation of L1, L2 and Max Norms in Optimization

Cost Function

Explanation of L1, L2 and Max norms in Optimization

When you set multiple goals for an optimization, the question arises as to what is actually going to drive the optimizer which is not a multi-objective one. The cost function will have a lot to do with it. The following discussion explains how the cost function is put together when there are multiple goals.

The general goal setting structure in Optimetrics is a logical sentence with the format:

$Calculation_{(i)}\ Condition_{(i)}\ Goal_{(i)}\ Weight_{(i)}$

The cost function that the optimizer uses is built based on the norm setting as long as there are multiple goals and none of those use the **minimize** or **maximize** conditions. Thus, in this case the error associated with each individual goal (weighted) is combined in a way that is specific for each norm type chosen.

For **L1** norm, the actual cost function uses the sum of absolute weighted values of the individual goal errors:

$$Cost = \sum_{1}^{N} \left| w_i \varepsilon_i \right|$$

For **L2** norm, the actual cost function uses the weighted sum of absolute values of the individual

$$Cost = \sum_{1}^{N} w_i \varepsilon_i^2$$

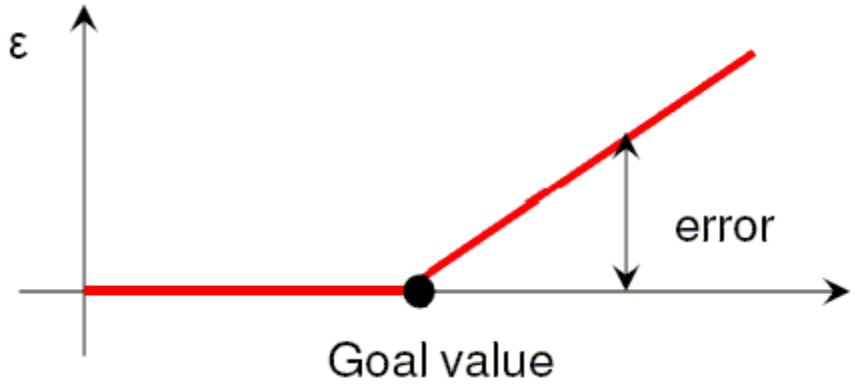
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For the **Maximum** norm, the cost function uses the maximum among all the weighted goal errors:

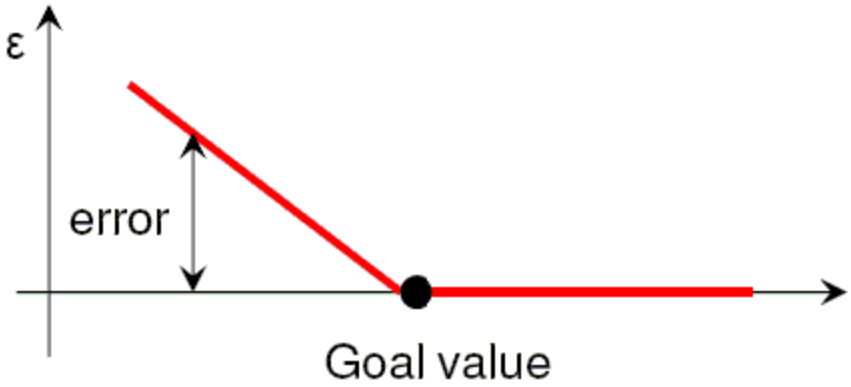
$$Cost = \underset{1}{\overset{N}{Max}} W_i \cdot \varepsilon_i$$

For all the above situations, N is the number of individual goals, while $w_i\varepsilon_i$ are individual weighting factors and residual errors. A minimization of the cost function is performed during optimization since it makes sense to minimize the error in the sense of the chosen norm type.

The graphical representation of the error is possible and depends upon the actual condition being used. If a “<” condition is used, the error can be represented as below:

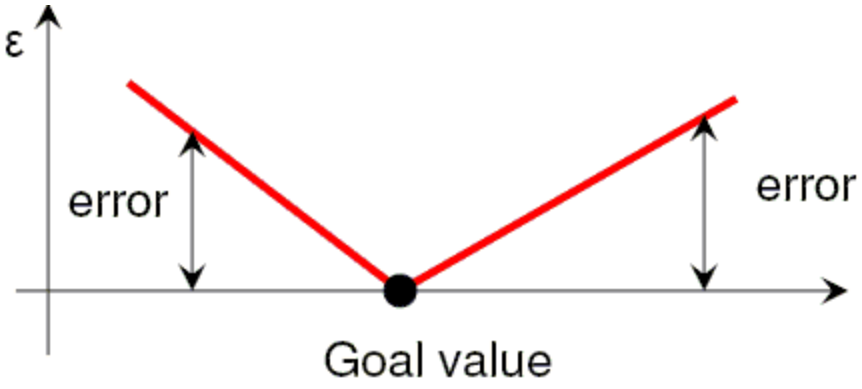


If a “>” condition is used, the error can be represented as below:



If a “=” condition is used, the error is double-sided and can be represented as below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The norm type does not affect Goal settings that use either the **minimize** or **maximize** condition. Note that when using **minimize** or **maximize** settings for the condition there should be a single goal setting which in this case coincides with the cost function.

Related Topics

Cost Function

Advanced Genetic Algorithm Optimizer Options

The Genetic Algorithm (GA) search for Optimization analysis is an iterative process. In each generation some new individuals (Children / Number of Individuals) are created and the so grown population participates in a selection (natural-selection) process that in turn reduces the size of the population to a desired level (Next Generation / Number of Individuals).

If you select the Genetic Algorithm for an Optimization analysis, the **Setup** button is enabled on the **Setup Optimization** page.

1. Click **Setup** to open the **Advanced Genetic Algorithm Optimizer Options** dialog box.
2. Select the Stopping Criteria. You can select any combination.
   - **Maximum number of generations** – Enables a value field.
   - **Elapsed time** – Enables a drop-down list with times ranging from five minutes to two weeks.
   - **Slow convergence**.
3. Specify the Parents.

   The first step toward mating is a selection process that determines the participating individuals. Potential parents are selected from the Current Generation. This is a set of individuals that is always a subset of the current generation.

   - **Number of individuals** – Specify the number of parents for the optimizer to use. You can set the Number of Individuals to less than or equal to the size of the “Current

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Generation". One reason to consider fewer parents than the possible maximum is to steer the GA toward improvement by selecting the better portion of the current generation to be able to mate.

- **Roulette selection** – Enables the **Selection pressure** value field. This number defines how many times more probable is the selection of the best individual over the worst individual in an elementary spin of the roulette wheel.

4. Specify the Mating pool.

   Create the Mating pool by selecting randomly from the parents. Note that with each selection, the parent gets "cloned" so can select it again and again.

   - **Number of individuals** – Specify the number of individuals to include in the mating pool.
   - **Reproduction setup** – Opens the **Genetic Algorithm Optimizer Reproduction Setup** dialog box.

5. Click **Reproduction setup** to specify the Crossover setup and the Mutation setup.

   The crossover and mutation operator have different roles: *Crossover* mixes "features" of the parents in a new combination, while *mutation* slightly alters the "features" of the individuals. Both need to be present in a GA. The crossover is a way to discover new combinations while the mutation acts as a local search or fine-tuning step. Mutation also keeps diversity in a population, which is a must for GA.

   The crossover operator has two steps. It first alters the variable values of the parents according to a distribution. This tends to produce one child that looks a lot like one parent, and one child that looks a lot like the other parent. Next, some of the variable values of the two children can be exchanged in order to achieve more variation.

   For crossover there are four possible parameters.

   - **Individual Crossover Probability** – Determines, for each pair in the mating pool, the probability that their features are mixed. Usually, this probability should be close or equal to one. If you set it set less than one, some parents will produce two children which are exact clones of the parents. This means that some children inherit all the features of their parents unchanged.
   - **Variable Crossover Probability** – Determines, for each variable, the probability of mixing, if the parent is a candidate for mixing. Parents often have multiple variables. This is usually set high to ensure that most or all variables mix.
   - **Variable Exchange Probability** – After the slight change in the variable values is made, the crossover operation can exchange the values of the variables between the two children that are being constructed. The Variable Exchange Probability governs the likelihood of exchange of any variable.
   - **Mu** – A general parameter that defines the sharpness of the distribution that might be used for the **Variable Crossover Probability**. Mu should be greater than one. There is no theoretical upper limit, but we recommend not exceeding 30.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. Select one of the four **Crossover types** from the drop-down list.

   The crossover type selected affects the options available.

Uniform	Individual crossover probability Variable crossover probability
One point	Individual crossover probability
Two point	Individual crossover probability
Simulated binary crossover	Individual crossover probability Variable crossover probability Variable exchange probability Mu

7. Select a **Mutation type** from the drop-down list:
   - **Uniform Distribution**
   - **Gaussian Distribution**
   - **Polynomial Mutation.**
8. For the selected mutation type, set the following parameters:
   - **Uniform Mutation Probability** – If this is more than zero (recommendation is to have still a small probability here), then there will be some children whose features are a completely random design (design variables randomly selected over the domain).
   - **Individual Mutation Probability** – Controls, for each child, the likelihood of a mild mutation.
   - **Variable Mutation Probability** – If the child will be mutated, this probability controls at the variable level the likelihood of a mutation of the variables.
   - **Standard Deviation** – The standard deviation of the selected distribution being used for the mutation. It is measured relatively to the optimization-domain.
9. When you have completed the Reproduction setup in the **Genetic Algorithm Optimizer Reproduction Setup** dialog box, click **OK** to close it and return to the **Advanced Genetic Algorithm Optimizer Options** dialog box.
10. In the **Advanced Genetic Algorithm Optimizer Options** dialog box, specify the children as a Number of Individuals.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

11. Set the **Pareto Front** value.

    This the number of the very best individuals (identified relative to the cost function) to keep for future generations.

12. Set the Next Generation parameters. The Next Generation is selected from the Parents, the children, and the Pareto front.
    - **Number of individuals** – Specify the number of individuals to survive to form the next generation for the optimizer to use.
    - **Roulette selection** – Enables the **Selection pressure** field. This number defines how many times more probable is the selection of the best individual over the worst individual in an elementary spin of the roulette wheel.
13. Click **OK** to accept the settings for the Genetic Algorithm and to close the dialog box.

Related Topics

Setting up an Optimization Analysis

Adding a cost function

Optimization Overview

Acceptable Cost

Explanation of L1, L2, and Max Norms in Optimization

Choosing an Optimizer

Sensitivity Analysis Overview

During a sensitivity analysis, Optimetrics explores the vicinity of the design point to determine the sensitivity of the design to small changes in variables. The variables and their attributes define the design point, the problem around which the sensitivity analysis is performed.

When Optimetrics performs a sensitivity analysis, its goal is to calculate the second-order regression polynomials for all of the design's output parameters. The algorithm first determines an appropriate interval for each variable. The intervals are further subdivided according to the available number of iterations and variables. If the master output is not used, the specified initial displacement values define those intervals.

When all of the design calculations are complete, the second-order polynomials are fitted for all the output parameters. Optimetrics then reports the following quantities:

- Regression value at the current variable value.
- First derivative of the regression.
- Second derivative of the regression.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Setting Up a Sensitivity Analysis

Selecting a Master Output

Selecting a Master Output

During a sensitivity analysis, the design variations that Optimetrics selects to solve are close to the design point, but not so close that numerical noise (from the finite element mesh) affects the analysis. The algorithm Optimetrics uses to determine the design variations to solve must be based on only one output parameter and that output parameter's numerical noise. Therefore, if you have defined more than one output parameter, be sure to select **Master Output** for the output variable on which you want the selection of design variations to be based.

Related Topics

Setting Up an Output Parameter

Setting Up a Sensitivity Analysis

Setting Up a Sensitivity Analysis

Follow this procedure to set up a sensitivity analysis. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes.

1. Before a variable can be included in a sensitivity analysis, you must specify that you intend for it to be used during a sensitivity analysis in the **DesignProperties** dialog box.
2. Select **Twin Builder** > **Optimetrics Analysis** > **Add Sensitivity** . The **Setup Sensitivity Analysis** dialog box appears.
3. Under the **Calculations** tab, type the maximum number of iterations per variable value for Twin Builder to perform in the **Max. No. of Iterations/Sensitivity Variable** text box.
4. Set up an output parameter calculation and select a Master Output.
5. Specify the value of the design point at which the sensitivity analysis should stop in the **Approximate Error in Master Output** text box.
6. In the **Variables** tab, specify the **Min/Max** values for variables included in the optimization, and the **Initial Displacement (Initial Disp.)** for the analysis.

   You can also override the variable starting values by selecting the **Override** check box and entering the desired value in the **Starting Value** field.

7. In the **General** tab, specify whether Optimetrics should use the results of a previous Parametric analysis or perform one as part of the optimization process.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

8. The following optional sensitivity analysis setup options can also be used:
    - Modify the starting variable value.
    - Modify the minimum and maximum values of variables that will be solved.
    - Exclude variables from the sensitivity analysis.
    - Set the initial displacement.
    - Modify the values of fixed variables that are not being modified during the sensitivity analysis.
    - Set linear constraints.
    - Request that Optimetrics solve a parametric sweep **before** a sensitivity analysis.
    - You can also request that Optimetrics solve a parametric sweep **during** a sensitivity analysis.
    - Set **HPC and Analysis Options**, which lets you select or create an analysis configuration. Enable a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses. By default, the fast calculation-update option is enabled whenever it is applicable.
    - Select the **Perform Worst Case Analysis** option for an extreme value analysis that focuses on the upper and lower boundaries of all the analyzed parameters. Some setup is required before performing a worst case analysis.

> **Note:**
>
> Sweeping or using a complex variable is not allowed in any optimetrics setup, including optimization, statistical, sensitivity, and tuning setups.

> **Note:**
>
> Improper or undefined simulation setups will cause errors during Sensitivity Analysis. To verify the analysis setups, Select **Analysis** > **Analyze** in the **Project Manager** pane to analyze the nominal circuit and review the messages in the Twin Builder **Message Manager** pane prior to running the Sensitivity Analysis.

Related Topics

Sensitivity Analysis Overview

Setting the Maximum Iteration Per Variable

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Setting the Maximum Iterations Per Variable

The **Max. No. of Iterations/Sensitivity Variable** value is the maximum number of design variations that Optimetrics solves per variable during a sensitivity analysis. This value is a stopping criterion; if the maximum number of iterations has been completed, the sensitivity analysis stops. If the maximum number of iterations has not been completed, the sensitivity analysis continues by performing another iteration – that is, by solving another design variation. It performs iterations until the approximate error in master output value is reached or until Optimetrics cannot proceed as a result of other sensitivity setup constraints, such as when it searches for a variable value that is larger than the maximum value.

To set the maximum number of iterations for a sensitivity analysis:

1. Open the **Setup Sensitivity Analysis** dialog box.
2. Click the **Calculations** tab.
3. Type a value in the **Max. No. of Iterations/Sensitivity Variable** text box.

Related Topics

Setting Up an Output Parameter

Setting Up an Output Parameter

Follow this procedure to add an output parameter to a sensitivity setup:

1. Under the **Calculations** tab of the **Setup Sensitivity Analysis** dialog box, click **Setup Calculations** to open the **Add/Edit Calculations** dialog box.
2. In the **Add/Edit Calculations** dialog box, set up output parameter calculations to be evaluated for sensitivity.
3. To modify the solution from which the output parameter is to be extracted, click in the **Solution** column and select an option.
4. You can modify the Calculation specified. Click the output parameter in the table and select **Edit**.
5. For output parameters based on swept variable, you must choose a single value in the **Calculation Range** at which to evaluate the output parameter.
6. If you have more than one output parameter, select **Master Output** if you want Optimetrics to use the output parameter to base its selection of solved design variations.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> During a sensitivity analysis, the design variations that Optimetrics selects to solve are close to the design point, but not so close that numerical noise (from the finite element mesh) affects the analysis. The algorithm that Optimetrics uses to determine the design variations to solve must be based on only one output parameter and that output parameter's numerical noise. If you have defined more than one output parameter, be sure to select **Master Output** for the output variable on which you want the selection of design variations to be based.

Related Topics

Selecting a Master Output

Specifying a Solution Quantity for an Output Parameter

When setting up an output parameter, you must identify the solution quantity on which to base the output parameter. Solution quantities are specified by mathematical expressions composed of basic quantities such as matrix parameters, and output variables.

The **Add/Edit Calculation** dialog box lets you define the mathematical equation for one or multiple output parameters. To set up an output parameter:

1. In the **Context** section of the dialog box:
   - Select the **Report Type** with a drop-down list containing the available types for this design.
   - Select the **Solution** from the drop-down list. This lists the available setups and sweeps. As a minimum, the **LastAdaptive** solution is available.
   - Select the **Geometry** from the drop-down list or select none (the default). This modifies the list of quantities available to the ones that apply to the specific geometry.
   - When selecting a geometry, you may also be required to specify a point within the geometry where the calculation is performed.
2. Click **Output Variables** to open the **Output Variables** dialog box, in which you can create special output variables to be used in the output parameter.
3. Use the **Calculation Expression** field in the **Trace** tab to enter the equation for the output parameter. To enter an expression, type it directly into the field or use the **Category**, **Quantity**, and **Function** lists as follows:
   - Select the **Category**. These depend on the Solution type and the design. This lets you specify the category of information used in the output parameter.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Select a **Quantity** from the list. Available quantities depend upon the Solution type, as well as the Geometry and Category selection. Selecting a Quantity enters it into the **Calculation Expression** field.
- Select a **Function** to apply to the value in the calculated expression.
- For swept variables, click **Range Function** to open the **Set Range Function** dialog box to apply functions to the expression that apply over the sweep range.

4. The **Calculation Range** tab applies to swept variables and lets you specify the range of the sweep over which to apply the calculation.
5. When you obtain the desired **Calculation Expression**, click **Add Calculation** to add the entry to the calculation table in the **Setup Sensitivity Analysis** dialog box. To add multiple entries to the table, change the **Calculation Expression** and click **Add Calculation**.
6. To update or edit a selected cost function, enter the desired Calculation Expression and click **Update Calculation**.
7. Click **Done** to return to the **Setup Sensitivity Analysis** dialog box.

> **Note:**
>
> The specified solution quantity must be able to be evaluated to a single, real number.

Related Topics

Setting the Calculation Range of an Output Parameter

Setting the Calculation Range of an Output Parameter

The calculation range of a solution quantity determines the intrinsic variable value at which the solution quantity is extracted. For a sensitivity setup, the calculation range must be a single value. If you specified that the solution quantity be extracted from a frequency sweep solution, Optimetrics uses the starting frequency in the sweep by default.

1. Under the **Calculations** tab of the **Setup Sensitivity Analysis** dialog box, click the **Calculation Range** column of the table for the calculation to be modified. The **Edit Calculation Range** dialog box appears.
2. In the **table**, click **Edit** in the row to be modified.

If you choose to solve a parametric setup during the sensitivity analysis, the variables swept in that parametric setup are available in a separate dialog box. If you sweep a variable in the parametric setup that is also a sensitivity variable, that variable is excluded from the sensitivity analysis.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Click the value for the calculation range in the list and dismiss the dialog box.
4. Click **OK** in the **Edit Calculation Range** dialog box to accept the new value for the intrinsic variable, and return to the **Setup Sensitivity Analysis** dialog box.

Related Topics

Setting Up an Output Parameter

Modifying the Starting Variable Value for Sensitivity Analysis

The design point of the sensitivity analysis is the starting value of the sensitivity variable; it is usually the first variation to be solved. Optimetrics sets the starting value of a variable to the current value set for the nominal design. You can modify the design point for each sensitivity setup.

> **Warning:**
>
> Variable values must be single real numbers, or expressions that evaluate to single real numbers. Complex numbers cannot be used as the values of variables in any optimetric analysis.

1. In the **Setup Sensitivity Analysis** dialog box, click the **Variables** tab. All of the variables that were selected for the sensitivity analysis appear.
2. Type a new value in the **Starting Value** text box for the value to override and press **Enter**.

   The **Override** option is now selected. This indicates that the value you entered is to be used for this sensitivity analysis; the current value set for the nominal model will be ignored.

   - Alternatively, you can select the **Override** option, then type a new variable value in the **Starting Value** text box.
3. Optionally, click a new unit system in one of the **Units** text boxes.

To revert to the default starting value, clear the **Override** option.

Related Topics

Setting Up a Sensitivity Analysis

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Setting the Min. and Max. Variable Values

For every sensitivity setup, Optimetrics sets the minimum and maximum values that it will consider for a sensitivity variable. Optimetrics sets a variable's minimum value equal to approximately one-half its starting value. (The starting value is the variable's current value set for the nominal design.) Optimetrics sets the variable's maximum value equal to approximately 1.5 times the starting value. During sensitivity analysis, variable values outside this range are not considered.

> **Warning:**
>
> Variable values must be single real numbers, or expressions that evaluate to single real numbers. Complex numbers cannot be used as the values of variables in any optimetric analysis.

Related Topics

Override the default minimum and maximum variable values for a single sensitivity setup

Change the default minimum and maximum variable values for every sensitivity setup

Overriding the Min. and Max. Variable Values for a Single Sensitivity Setup

1. In the **Setup Sensitivity Analysis** dialog box, click the **Variables** tab. All of the variables selected for sensitivity analysis appear.
2. Type a new value in the **Min** or **Max** text box for the value to override and press **Enter**.

The **Override** option is now selected. This indicates that the value you entered is to be used for this sensitivity analysis; the variable's current **Min** or **Max** value set in the nominal design is ignored.

- Alternatively, you can select the **Override** option and type a new value in the **Min** or **Max** text box.

3. Optionally, click a new unit system in one of the **Units** text boxes.

To revert to the default minimum and maximum values, clear the **Override** option.

Related Topics

Setting Up a Sensitivity Analysis

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Changing the Min. and Max. Variable Values for Every Sensitivity Setup

1. Make sure the variable's minimum and maximum values are not overridden in any sensitivity setup.
2. If the variable is a design variable, click **Twin Builder** > **Design Properties**.

   If the variable is a project variable, click **Project** > **Project Variables**.

   The **Properties** dialog box appears.
3. Select **Sensitivity**.
4. Type a new value in the **Min** or **Max** text box for the value to override and press **Enter**.

   When Optimetrics solves a sensitivity setup, it does not consider variable values that lie outside of this range.

Related Topics

Setting Up a Sensitivity Analysis

Setting the Initial Displacement

The initial displacement is the difference between a variable's starting value and the next solved design variation. During the sensitivity analysis, Optimetrics does not consider an initial variable value that is greater than this step size away from the starting variable value.

1. In the **Setup Sensitivity Analysis** dialog box, click the **Variables** tab.
2. Optimetrics displays the **Initial Disp.** column, with default values for each sensitivity variable.
3. In the **Initial Disp.** text box, type the initial displacement value. Optionally, modify the unit system in the **Units** text box.

Related Topics

Setting Up a Sensitivity Analysis

Solving a Parametric Setup Before a Sensitivity Analysis

Solving a parametric setup before a sensitivity setup is useful for guiding Optimetrics in a sensitivity analysis.

To solve a parametric setup before a sensitivity setup:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. In the **Setup Sensitivity Analysis** dialog box, click the **General** tab.
2. Click the parametric setup for Optimetrics to solve from the **Parametric Analysis** drop-down list.

   > **Note:**
   >
   > The parametric setup must include sweep definitions for the sensitivity variables.

3. Select **Solve the parametric sweep before analysis**.

If the parametric setup has not yet been solved, Optimetrics solves it. Optimetrics uses the results (of the solution calculation you requested under the **Goals** tab of the **Setup Sensitivity** dialog box) to determine the next design variation to solve for the sensitivity analysis.

Related Topics

Setting Up a Sensitivity Analysis

Solving a Parametric Setup during a Sensitivity Analysis

Solving a parametric setup during a sensitivity analysis is useful when you want Optimetrics to solve every design variation in the parametric setup at each sensitivity analysis iteration. An output parameter goal could then depend on the value of the variable swept in the parametric setup.

To solve a parametric setup during a sensitivity analysis:

1. In the **Setup Sensitivity Analysis** dialog box, click the **General** tab.
2. Click the parametric setup you want Optimetrics to solve from the **Parametric Analysis** drop-down list.
3. Select **Solve the parametric sweep during analysis**.

Related Topics

Setting Up a Sensitivity Analysis

Performing Worst Case Analysis

Two popular worst case analysis techniques can be implemented: extreme value analysis and Monte Carlo analysis.

Some setup is required before performing worst case analysis. First, identify uncertainties in design and create a local or project variable for each of them. Second, determine the variation

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

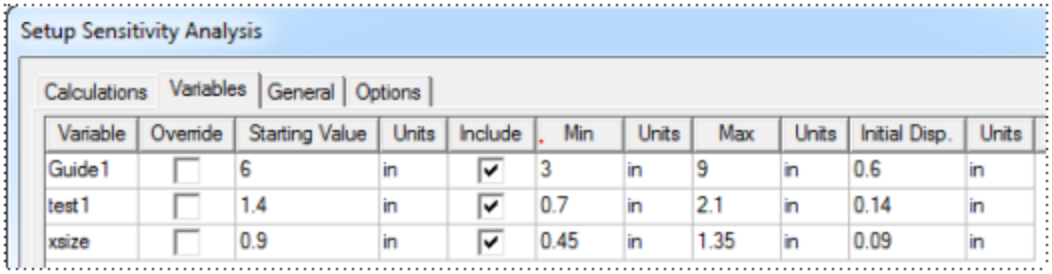
range of each variable (Min and Max); its statistical distribution is optional. Third, determine a measurement of performance, especially for extreme value analysis.

Extreme Value Analysis

This is one of the most popular methods to estimate worst-case performance. First perform a sensitivity analysis. The results (sensitivities/first derivative) let you pick an extreme value (upper or lower bound) for each variable. The corresponding simulation result is used to predict upper and lower bound of performance. The assumption is that extreme performance is reached at boundary value. Note that in certain cases, making such an assumption is not valid.

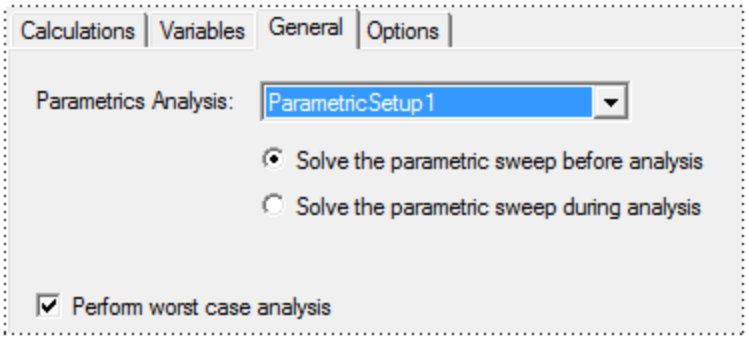
To perform an extreme value analysis:

1. Create a new analysis under **Optimetrics Analysis** > **Add Sensitivity**, and include variables in a sensitivity analysis. To do so, go to your list of variables, specify Min and Max values of each variable, and select **Include**. The following figure shows an example with three variables included:



Variable	Override	Starting Value	Units	Include	Min	Units	Max	Units	Initial Disp.	Units
Guide1	☐	6	in	☑	3	in	9	in	0.6	in
test1	☐	1.4	in	☑	0.7	in	2.1	in	0.14	in
xsize	☐	0.9	in	☑	0.45	in	1.35	in	0.09	in

2. Follow ***Setting up a Sensitivity Analysis***. During this procedure, set your performance measurement in the **Calculations** tab, select all variables in the **Variables** tab, and select **Perform worst case analysis** in the **General** tab.



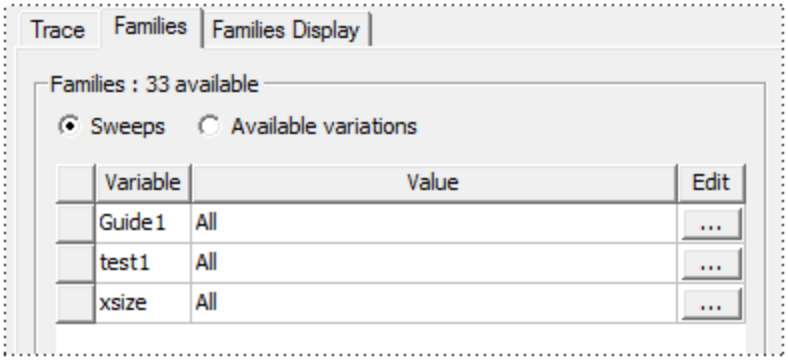
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Select **Perform worst case analysis** to calculate first derivatives for each variable. If you have three variables and for Var1 the first derivative is negative, for Var2 the first derivative is positive, and for Var3 the first derivative is positive, then for Worst Case Analysis, you should request two more variations:
>
> Var1@minimum, Var2@maximum, Var3@maximum
>
> and Var1@maximum, Var2@minimum, Var3@minimum

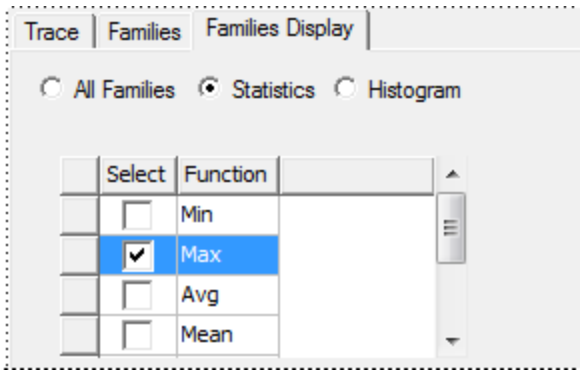
3. After setup, analyze your **Sensitivity Setup** under **Optimetrics**.

To view a worst-case result (upper bound only):

1. Create a **Data Table** report.
2. In the **Context** pane in the **Report** dialog box, select matching **Solution** and **Optimetrics** setup.
3. On the **Families** tab, change the setting to **All** in the **Value** column for each variable.



4. On the **Families Display** tab, select **Statistics**, then **Max**.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

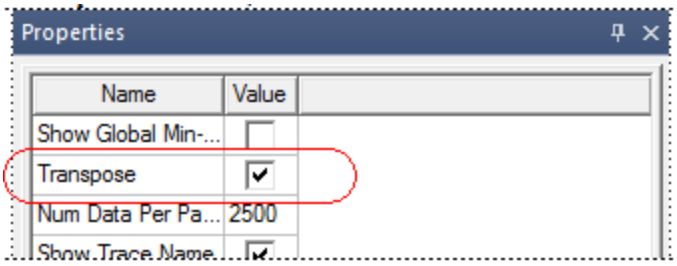
The associated data table shows the Max values.

	Freq [GHz]	dB(S(1:1,1:1)) Setup1 : Sweep Max
1	3.000000	-0.000000
2	4.000000	-0.000000
3	5.000000	-0.105323
4	6.000000	-0.248322
5	7.000000	-0.267220
6	8.000000	-0.202039
7	9.000000	-0.298840

To see the corresponding variable values, select **All Families** on the **Families Display** tab, and locate the max value to see the values of variables.

	Freq [GHz]	dB(S(1:1,1:1)) Setup1 : Sweep Guide1='3in' test1='2.1in' xsize='0.45in'
1	3.000000	-0.545922
2	4.000000	-2.300003
3	5.000000	-10.634331
4	6.000000	-4.586505
5	7.000000	-5.950253
6	8.000000	-9.019574
7	9.000000	-3.210097

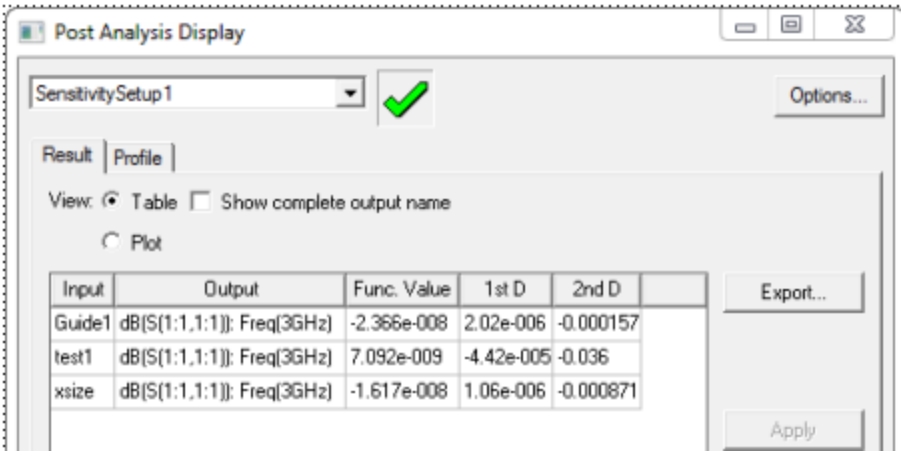
Tip: Transpose the table for an alternate view (double-click the data table to view the **Properties** dialog box, and on the **Data Table** tab select **Transpose**).



To estimate lower bound of performance:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Set the sensitivity of performance for each variable. (Follow the documentation for ***Viewing Output Parameter Results for a Sensitivity Analysis***). Identify the variables that have major influence on the performance.



2. In your project, manually change these variables to the corresponding bounds: choose Min for a positive first derivative and Max for a negative first derivative.

Related Topics

Example of a Worst Case Analysis

Monte Carlo Analysis

This method does not assume a circuit is linear; better accuracy is achieved with more iterations. The cost is computing time and resources.

To perform a Monte Carlo Analysis, create a new analysis under **Optimetrics Analysis** > **Add Statistical**. See ***Setting Up a Statistical Analysis*** to set this up. The upper and lower bound of performance can be found on the edge of performance distribution.

Tip: If the distribution of performance is not of interest, set all variables as uniformly distributed.

Related Topics

Sensitivity Analysis Overview

Statistical Analysis Overview

Example of a Worst Case Analysis

This example contains a simple voltage divider circuit illustrating worst case analysis (WCA).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The following schematic contains two resistors (R1 and R2) that divide voltage supply E1. The output of interest is VM1.V.

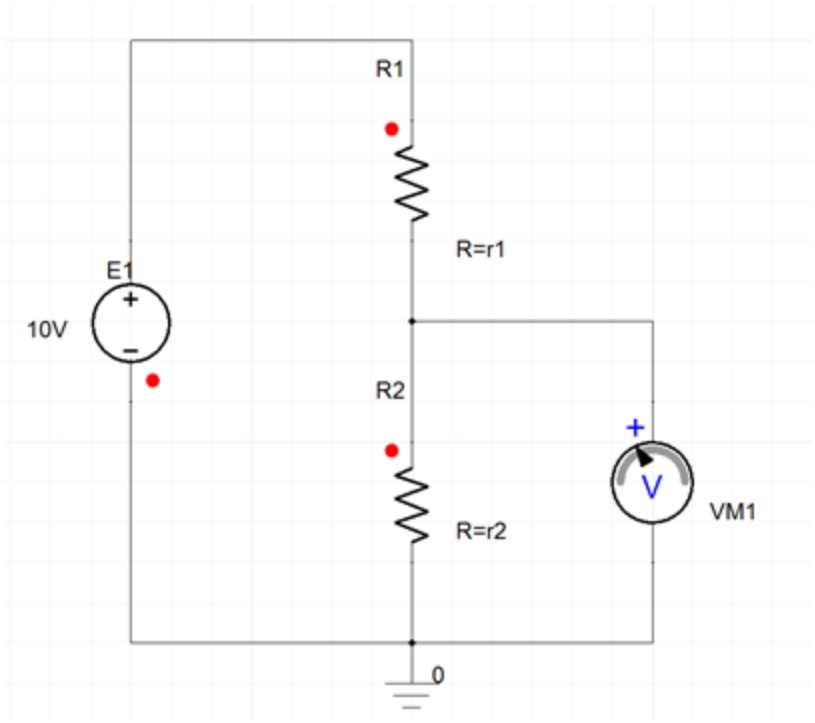


Figure 19-1 Simple Voltage Divider

The following information is available from the manufacturer:

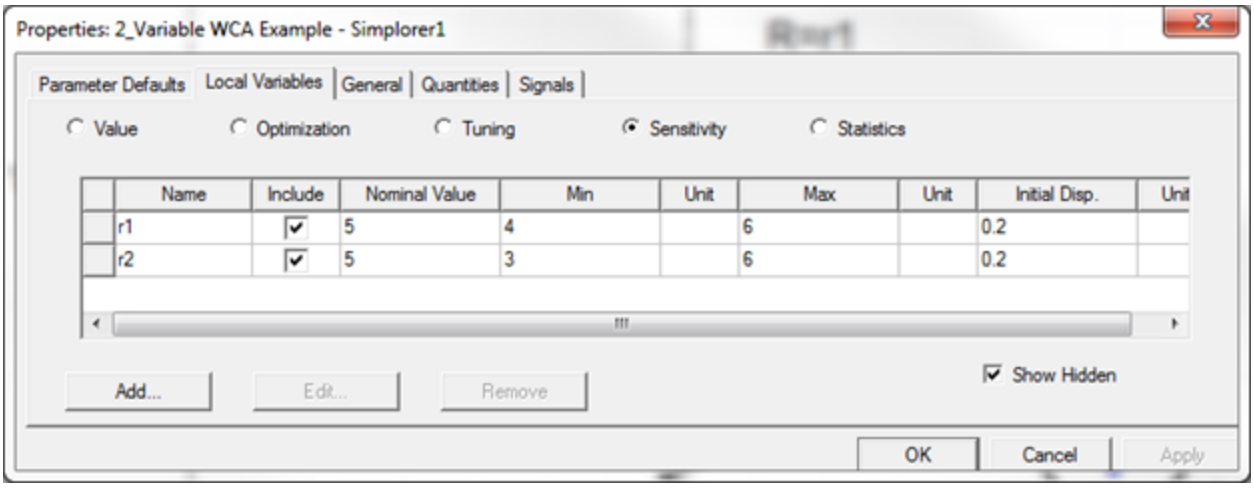
	Nominal	**Tolerance**
R1	5 ohm	+/- 20%
R2	5 ohm	+20% , -40%

To determine the worst values of output VM1.V caused by uncertainty of the resistors, you want to find both the upper and lower bounds of VM1.V because either low or over voltage can cause problems.

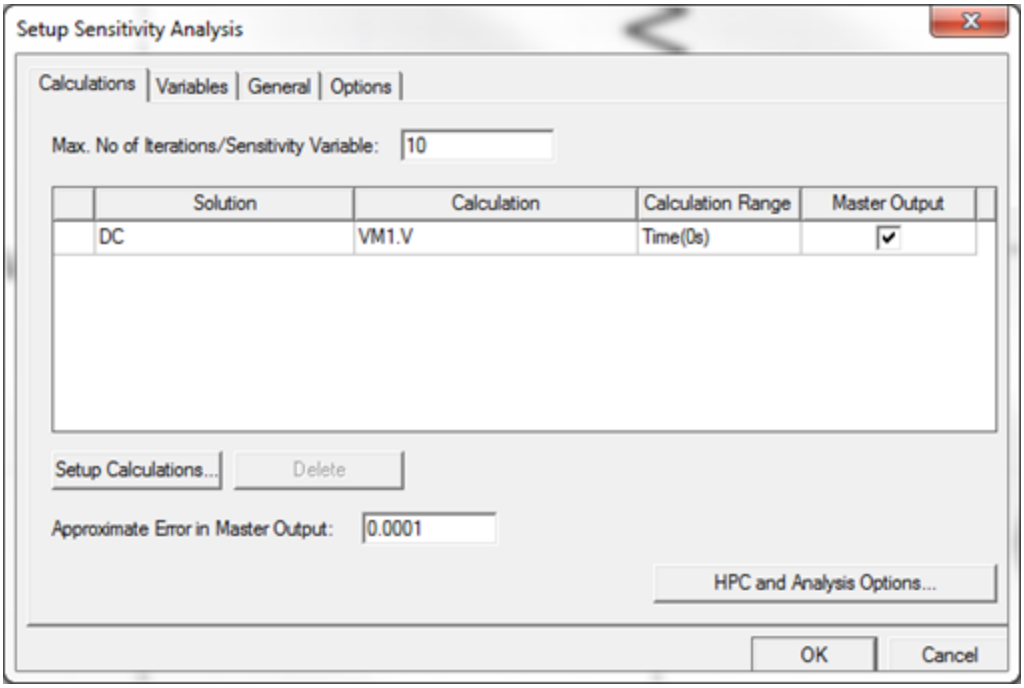
In Twin Builder, to estimate the worst case based on the sensitivity analysis result:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Set the range of variables accordingly.

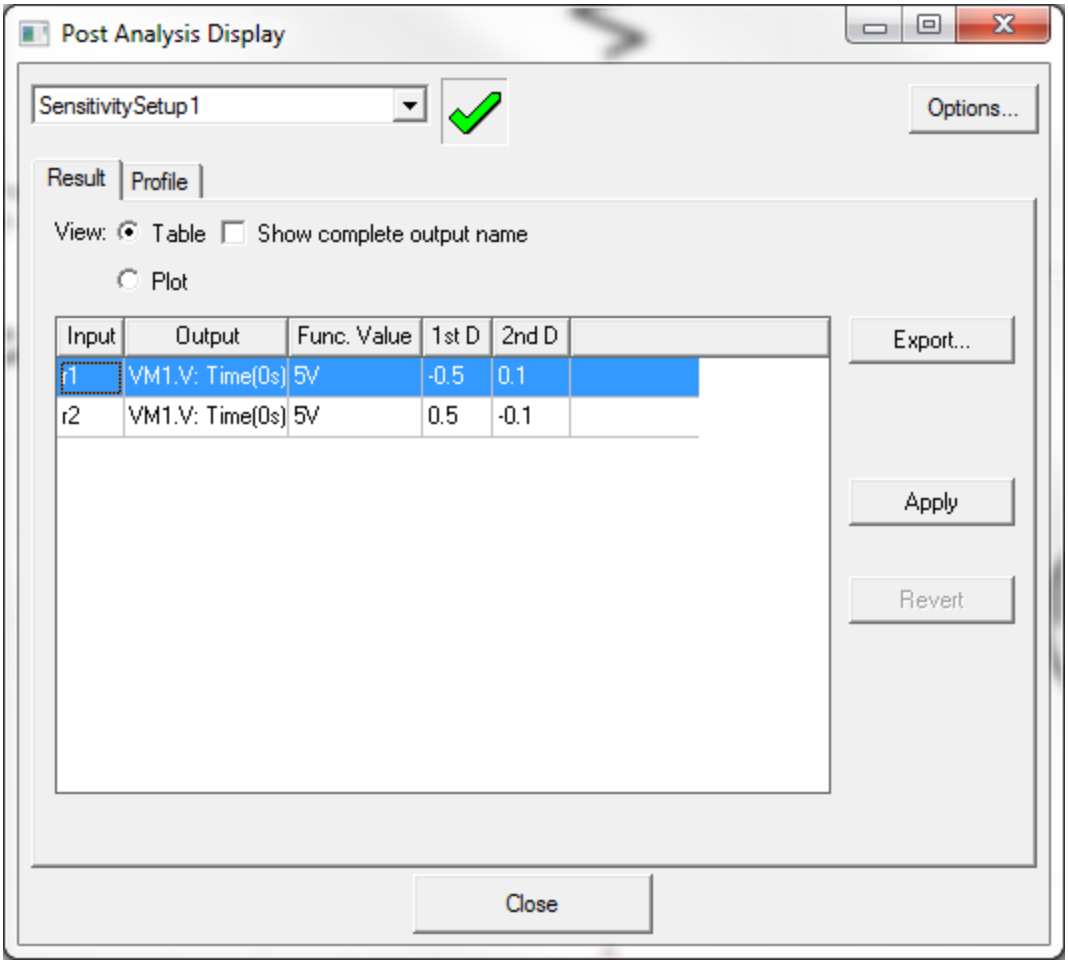


2. Set up the sensitivity analysis of DC analysis of VM1.V.



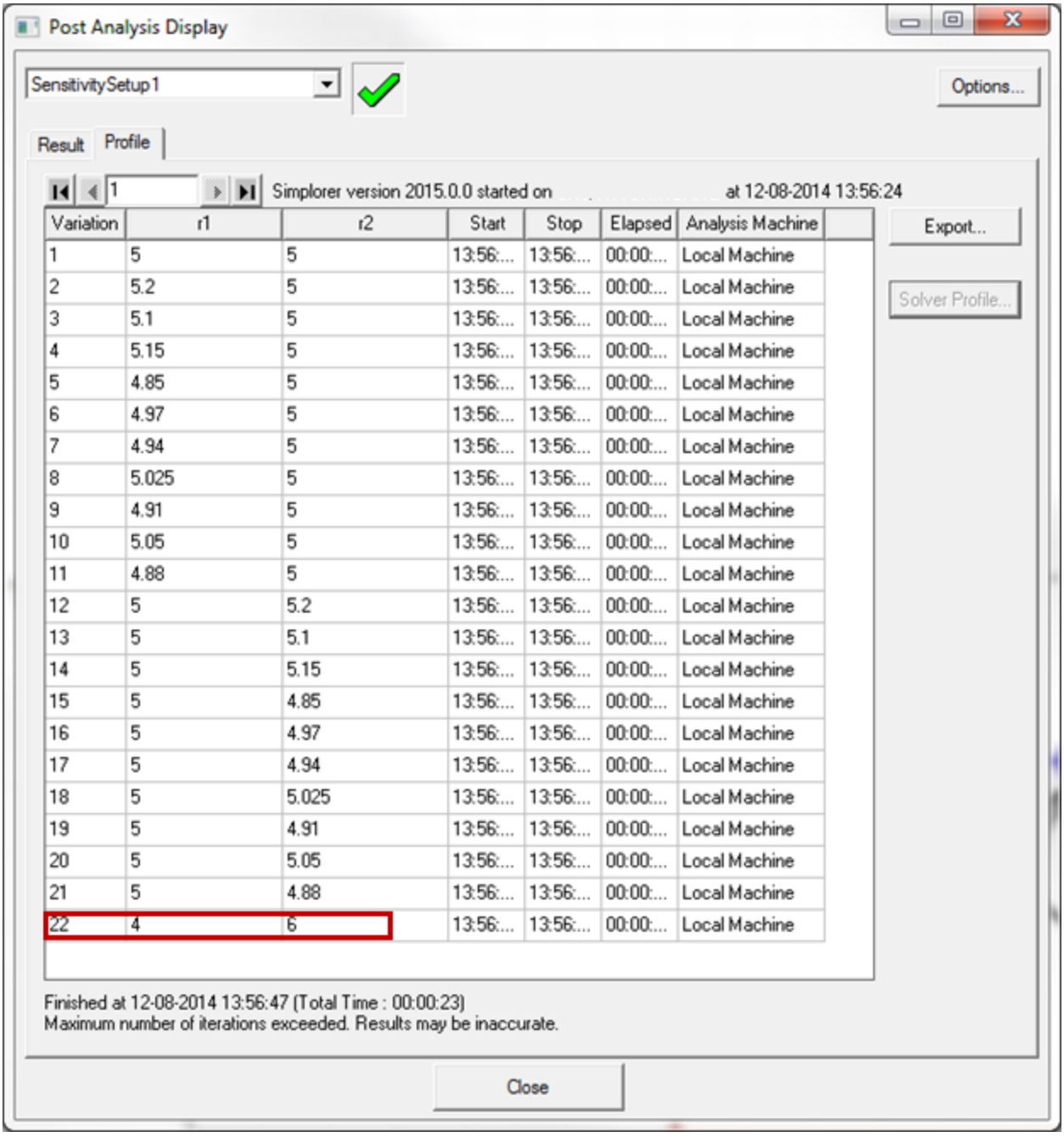
3. Review the sensitivity analysis result. Note the first derivative is negative to r1 and positive to r2. Assuming the circuit is linear, which is true in this case, VM1.V is maximized (6V) with a minimal r1 and a maximal r2 (that is, r1=4 and r2=6), and minimized (3.3V) with r1=6 and r2=3.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



This method can be generalized to multiple variables, and it requires two extra analyses, one for upper bound and one for lower bound, regardless of the number of variables. The following example shows the upper bound in variation 22.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Variation	r1	r2	Start	Stop	Elapsed	Analysis Machine
1	5	5	13:56:...	13:56:...	00:00:...	Local Machine
2	5.2	5	13:56:...	13:56:...	00:00:...	Local Machine
3	5.1	5	13:56:...	13:56:...	00:00:...	Local Machine
4	5.15	5	13:56:...	13:56:...	00:00:...	Local Machine
5	4.85	5	13:56:...	13:56:...	00:00:...	Local Machine
6	4.97	5	13:56:...	13:56:...	00:00:...	Local Machine
7	4.94	5	13:56:...	13:56:...	00:00:...	Local Machine
8	5.025	5	13:56:...	13:56:...	00:00:...	Local Machine
9	4.91	5	13:56:...	13:56:...	00:00:...	Local Machine
10	5.05	5	13:56:...	13:56:...	00:00:...	Local Machine
11	4.88	5	13:56:...	13:56:...	00:00:...	Local Machine
12	5	5.2	13:56:...	13:56:...	00:00:...	Local Machine
13	5	5.1	13:56:...	13:56:...	00:00:...	Local Machine
14	5	5.15	13:56:...	13:56:...	00:00:...	Local Machine
15	5	4.85	13:56:...	13:56:...	00:00:...	Local Machine
16	5	4.97	13:56:...	13:56:...	00:00:...	Local Machine
17	5	4.94	13:56:...	13:56:...	00:00:...	Local Machine
18	5	5.025	13:56:...	13:56:...	00:00:...	Local Machine
19	5	4.91	13:56:...	13:56:...	00:00:...	Local Machine
20	5	5.05	13:56:...	13:56:...	00:00:...	Local Machine
21	5	4.88	13:56:...	13:56:...	00:00:...	Local Machine
22	4	6	13:56:...	13:56:...	00:00:...	Local Machine

Related Topics

Performing Worst Case Analysis

Sensitivity Analysis Overview

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Statistical Analysis Overview

Use statistical analysis to explore the effects of random combinations of values of selected variables on selected global or local available analysis results. Before a variable is included in a statistical analysis, specify that you intend for it to be used during a statistical analysis. For each variable you must specify the type of distribution (Uniform, Gaussian, Lognormal or User Defined) and the corresponding parameters of the selected distribution.

> **Note:**
>
> A statistical analysis is currently limited to using a maximum of 30 variables.

In addition to specifying the variables to use in the statistical analysis and the parameters of the chosen distribution, you must also specify the output quantities of interest. These quantities can be global ones such as previously defined parameters (force/torque, inductance/capacitance, and so on), other named quantities, quantities defined in the field calculator as global (such a domain integral of a certain field quantity) or local (such as field value at a certain location). The calculations performed during the statistical analysis are specified during setup, in a manner similar to other types of analysis in Optimetrics.

Following the analysis the statistical distribution of the output quantities can be visualized in histogram format. To access available reports, after the statistical analysis is complete, right-click the respective Statistical analysis setup and select **View Analysis Result**.

Related Topics

Setting Up a Statistical Analysis

Setting Up a Statistical Analysis

Follow this general procedure to set up a statistical analysis. Once you have created a setup, you can **Copy** and **Paste** it, then make changes to the copy, rather than redoing the whole process for minor changes. You can create a statistical setup before defining variables but all variables must be defined before you start the statistical analysis.

> **Note:**
>
> A statistical analysis is currently limited to using a maximum of 30 variables.

1. Before a variable is included in a statistical analysis, you must specify that you intend to use it during a statistical analysis in the **Properties** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Select **Twin Builder** > **Optimetrics Analysis** > **Add Statistical** . The **Setup Statistical Analysis** dialog box appears.
3. Under the **Calculations** tab, type the maximum number of iterations for Twin Builder to perform in the **Maximum Iterations** text box.
4. If you want to specify an initial seed value, select the **Specify initial seed value** check box and enter a positive value in the text box. Each different seed value creates a new statistical sequence.
5. Specify a solution quantity to evaluate.
6. In the **Calculation** text box, set the value at which the solution quantity is to be computed.
7. Optionally, modify the distribution criteria to use.
8. The following ***optional*** statistical analysis setup options can also be used:
   - Modify the starting variable value.
   - Exclude variables from the statistical analysis.
   - Modify the values of fixed variables that are not being modified during the statistical analysis.
   - Request that Optimetrics solve a parametric sweep during a statistical analysis.

     > **Note:**
     >
     > Sweeping or using a complex variable is not allowed in any optimetrics setup, including optimization, statistical, sensitivity, and tuning setups.

   - Set **HPC and Analysis Options** to select or create an analysis configuration. Enable a fast calculation-update algorithm to speed up Optimetrics and report updates during Optimetrics analyses. By default, the fast calculation-update option is enabled whenever it is applicable.

     > **Note:**
     >
     > Improper or undefined simulation setups will cause errors during Statistical Analysis. To verify the analysis setups, click **Analysis** > **Analyze** in the **Project Manager** pane to analyze the nominal circuit and review the messages in the **Message Manager** pane prior to running the Statistical Analysis.

Related Topics

Statistical Analysis Overview

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Statistical Analysis with Netlist Components

When conducting statistical analyses with Twin Builder netlist components it is important to ensure that component definitions contain a ModelName property. If a component is netlisted as **X@ID**, the component's ModelName property must be set to **X**. Next, the statistical variable is set to **X@ID** and the model engine is able to locate it.

Additionally, you must also have the same parameter names listed in your library subcircuit netlist and in your UI parameters. For example, with a subcircuit netlist of:

```
CAP 0805: Netlist string: X@ID %0 %1 CAP0805 PARAMS: Cval=@C Rval=@R
```

The UI is only aware of **C** and requests statistical information on **X@32~~C**, while the subcircuit is only aware of **Cval**. Because of this, you must set up the same component parameter name which the actual model refers to. Consequently, if you add properties **Cval** and **Rval** in this component example, you must also change the netlist to read:

```
X@ID %0 %1 CAP0805 PARAMS: Cval=@Cval Rval=@Rval
```

As a result, statistical analysis and tuning will work for both **Cval** and **Rval**.

Note The netlist template is also capable of accessing RefDes (Reference Designator) information for netlist definitions of the form **X@RefDes %0 %1 %2 ...**.

Setting the Maximum Iterations for a Statistical Analysis

The **Maximum Iterations** value is the maximum number of design variations Optimetrics solves during a statistical analysis. This value is a stopping criterion; if the maximum number of iterations has been completed, the analysis stops. If the maximum number of iterations has not been completed, Optimetrics continues by performing another iteration (that is, by solving another design variation).

To set the maximum number of iterations for a statistical analysis:

1. Open the **Setup Sensitivity Analysis** dialog box.
2. Select the **Calculations** tab.
3. Type a value into the **Maximum Iterations** text box.

Related Topics

Setting up a Statistical Analysis

Specifying an Initial Seed Value for a Statistical Analysis

The initial seed value is a positive integer used as the starting point during a statistical analysis. When you enter a new seed value, a new statistical sequence is created.

To set an initial seed value for a statistical analysis:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Open the **Setup Sensitivity Analysis** dialog box.
2. Click the **Calculations** tab.
3. Select the **Specify initial seed value** check box.
4. Type a positive value in the text box.

Related Topics

Setting up a Statistical Analysis

Specifying the Solution Quantity to Evaluate for Statistical Analysis

When you add a statistical setup, you can identify one or more solution quantities to evaluate. The solution quantities are specified by mathematical expression composed of basic quantities. When you view the results, Twin Builder displays the distribution of the solution quantities.

1. In the **Calculations** tab of the **Setup Statistical Analysis** dialog box, click **SetupCalculations**. The **Add/Edit Calculations** dialog box appears. Use this dialog box to define one or more mathematical expressions for statistical evaluation.
2. In the **Context** section of the dialog box:
   - Select the **Report Type** with a drop-down list containing the available types for this design.
   - Select the **Solution** from the drop-down list. This lists the available setups and sweeps. As a minimum, the **LastAdaptive** solution is available.
   - Select the **Geometry** from the drop-down list or select none. This modifies the list of quantities available to the ones that apply to the specific geometry.
   - When selecting a geometry, you may also be required to specify a point within the geometry where the calculation is to be performed.
3. Click **Output Variables** to open the **Output Variables** dialog box. Use this dialog box to create special output variables used in the output parameter.
4. Use the **Calculation Expression** field in the **Trace** tab to enter the equation used for the solution quantities. To enter an expression, type it directly into the field or use the **Category**, **Quantity**, and **Function** lists as follows:
   - Select the **Category**. These depend on the Solution type and the design. This lets you specify the category of information used in the output parameter.
   - Select a **Quantity** from the list. Available quantities depend upon the Solution type, as well as the **Geometry** and **Category** selection. Selecting a **Quantity** enters it into the **Calculation Expression** field.
   - Select a **Function** to apply to the value in the calculated expression.
   - For swept variables, click **Range Function** to open the **Set Range Function** dialog box to apply functions to the expression that apply over the sweep range.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. The **Calculation Range** tab applies to swept variables and lets you specify the range of the sweep over which to apply the calculation.
6. When the desired **Calculation Expression** is obtained, click **Add Calculation** to add the entry to the calculation table in the **Setup Statistical Analysis** dialog box. To add multiple entries to the table, change the **Calculated Expression** and click **Add Calculation**.
7. To update or edit a selected cost function, enter the desired Calculation Expression and click **Update Calculation**.
8. Click **Done** to return to the **Setup Statistical Analysis** dialog box.

> **Note:**
>
> The solution quantity you specify must be able to be evaluated to a single, real number.

Related Topics

Setting up a Statistical Analysis

Setting the Maximum Iterations for a Statistical Analysis

Setting the Solution Quantity's Calculation Range

The calculation range of a solution quantity determines the intrinsic variable value at which the solution quantity is extracted. For a statistical setup, the calculation range must be a single value. If you specified that the solution quantity be extracted from a frequency sweep solution, Optimetrics will use the starting frequency in the sweep by default. Set the calculation range during the setup of the solution quantity for statistical evaluation. To modify the calculation range:

1. Under the **Calculations** tab of the **Setup Statistical Analysis** dialog box, click in the **Calculation Range** column of the table for the calculation to be modified. The **Edit Calculation Range** dialog box appears.
2. In the **table**, click **Edit** in the row to be modified.

   If you choose to solve a parametric setup during the statistical analysis, the variables swept in that parametric setup are available in a separate dialog box. If you sweep a variable in the parametric setup that is also a statistics variable, that variable is excluded from the statistics analysis.

3. Click the value for the calculation range in the list and dismiss the separate dialog box.
4. Click **OK** in the **Edit Calculation Range** dialog box to accept the new value for the intrinsic variable, and return to the **Setup Statistical Analysis** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Setting up a Statistical Analysis

Setting the Distribution Criteria

For every statistical setup, Optimetrics sets the distribution criteria to be uniform within 10% of the variable's starting value. You can modify the distribution type and criteria for a single statistical setup or for every statistical setup.

Related Topics

Override the default distribution criteria for a single statistical setup.

Change the default distribution criteria for every statistical setup.

Overriding the Distribution Criteria for a Single Statistical Setup

Follow this procedure to override the default distribution criteria for a single statistical setup:

1. In the **Setup Statistical Analysis** dialog box, click the **Variables** tab. All of the variables that were selected for statistical analysis appear.
2. Select or clear the **Include** check box for each variable to define the specific variables included in the statistical analysis setup.
3. For each included variable, select **Uniform**, **Gaussian**, **Lognormal**, or **User Defined** in the **Distribution** column for the variable you want to override.

   If you changed the distribution type, the **Override** option is now selected. This indicates that the distribution type you selected will be used for this optimization analysis; the current distribution type selected for the variable in the nominal design is ignored in this statistical analysis.

   - Alternatively, you can select the **Override** option first, then select a different distribution type in the **Distribution** text box.
4. Optionally, if you want to change the distribution criteria, click in the **Distribution Criteria** column for the variable you want to override. The **Edit Distribution** dialog box appears.
5. If the distribution type is **Gaussian**:
   a. Enter the standard deviation in the **Std. Dev** text box.
   b. Enter the lower limit of the distribution in the **Low Cutoff** text box.
   c. Enter the upper limit of the distribution in the **High Cutoff** text box.

      Twin Builder will solve design variations using a Gaussian distribution within the low and high cutoff values.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. If the distribution type is **Uniform**, enter a tolerance value in the text box.

   Twin Builder solves design variations within the tolerance range of the starting value, using an even distribution.

7. If the distribution type is **Lognormal**:
   a. Enter the cutoff probability in the **Cutoff Probability** text box.
   b. Enter the **sigma** value of the distribution in the **Sigma** text box and select a unit from the drop-down list.
   c. Enter the **m** value of the distribution in the **M** text box.
   d. Enter the **theta** value in the Theta text box and select a unit from the drop-down list.
8. If the distribution type is **User Defined**:
   a. Enter the cutoff probability in the **Cutoff Probability** text box.
   b. Click **Edit XY Data** to open the **Edit Datasets** dialog box in which you can select an existing dataset, or create a new one.
9. By default, all variables are set to sample using **Latin Hypercube** sampling. This sampling method provides for greater variability than random sampling by keeping track of chosen samples and guaranteeing that samples cannot be repeated. Revert to random sampling by clearing the check box in the **Latin Hypercube** column for any desired variable.
10. Click **OK**.

    To revert to the default distribution settings, clear the **Override** option.

Related Topics

Statistical Cutoffs

Changing the Distribution Criteria for Every Statistical Setup

To change the default distribution criteria for every statistical setup:

1. Make sure that the variable's distribution criteria are not overridden in any statistical setup.
2. If the variable is a design variable, do the following: On the **Twin Builder** menu, click **Design Properties**.

   If the variable is a project variable, do the following: Select **Project** > **Project Variables**.

   The **Properties** dialog box appears.

3. Select **Statistics**.
4. Click the **Distribution** column for the variable you want to change, then select **Uniform**, **Gaussian**, **Lognormal**, or **User Defined**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Optionally, if you want to change the distribution criteria, click in the **Distribution Criteria** column for the variable to change.

If the distribution type is **Gaussian**, the **Gaussian Distribution** dialog box appears. If the distribution type is **Uniform**, the **Uniform Distribution** dialog box appears.

6. If the distribution type is **Gaussian**:
   a. Type a cutoff probability value in the **Cutoff Probability** text box.
   b. Type the standard deviation in the **Std. Dev** text box.
   c. Type the mean value of the distribution in the **Mean** text box.

      Twin Builder will solve design variations using a Gaussian distribution within the low and high cutoff values.

7. If the distribution type is **Uniform**:
   a. Type a cutoff probability value in the **Cutoff Probability** text box.
   b. Type mean and tolerance values in the corresponding text boxes.

      Twin Builder will solve design variations within the tolerance range of the starting value, using an even distribution.

8. If the distribution type is **Lognormal**:
   a. Type a cutoff probability value in the **Cutoff Probability** text box.
   b. Type values for Sigma, M, and Theta in the corresponding text boxes.
9. If the distribution type is **User Defined**:
   a. Type a cutoff probability value in the **Cutoff Probability** text box.
   b. Click **Edit XY Data** to open the **Edit Dataset** dialog box.
   c. Either type or import the X and Y data values for the distribution in the **Edit Dataset** dialog box.
10. Click **OK**.

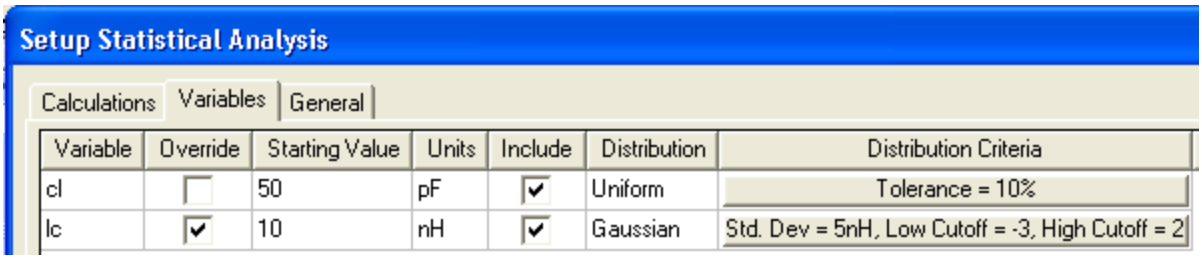
Related Topics

Statistical Cutoffs

Statistical Cutoffs

The low and high cutoff values multiply the Gaussian Standard Deviation value. In the image below, the **Ic** variable uses a Gaussian distribution that extends three standard deviations below 10nH and two standard deviations above 10nH. Outside these values, the Gaussian distribution is truncated, effectively giving a Gaussian distribution on a pedestal.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



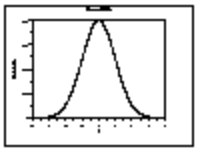
Uniform distributions such as variable **cl** above use only the Tolerance value, and do not have cutoffs.

Edit Distribution

When setting the distribution type for a variable, you can change the distribution parameters from the default values.

1. If the distribution type is **Gaussian**:
   a. Type the lower limit of the distribution in the **Cutoff Probability** text box.
   b. Type the mean value of the distribution in the **Mean** text box.
   c. Type the standard deviation of the distribution in the **Std Dev** text box.

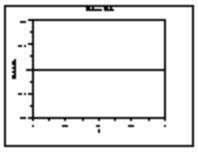
   Twin Builder solves design variations using a Gaussian distribution within the specified mean and standard deviation values.



2. If the distribution type is **Uniform**:
   a. Type the lower limit of the distribution in the **Cutoff Probability** text box.
   b. Type the mean value of the distribution in the **Mean** text box.
   c. Enter the tolerance in the **Tolerance** text box.

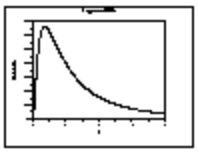
   Twin Builder solves design variations within the tolerance range of the starting value, using an even distribution.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



3. If the distribution type is **Lognormal**:
   a. Type the lower limit of the distribution in the **Cutoff Probability** text box.
   b. Enter the shape parameter of the distribution in the **Sigma** text box.
   c. Enter the scale parameter in the **M** text box. The scale parameter should be set to **1** for the standard lognormal distribution.
   d. Enter the location parameter value for **Theta** in the text box. The value for a standard lognormal distribution is **0**.

   Twin Builder solves design variations with a logarithmic distribution using the shape, scale and location parameters provided.



4. If the distribution type is **User Defined**:
   a. Type the lower limit of the distribution in the **Cutoff Probability** text box.
   b. Click **Edit XY Data** to manually define the data distribution using datasets.

Related Topics

Adding Datasets

Changing the Distribution Criteria for Every Statistical Setup

Overriding the Distribution Criteria for a Single Statistical Setup

Modifying the Starting Variable Value for Statistical Analysis

A variable's starting value is the first value solved during the statistical analysis. Optimetrics sets the starting value of a variable to be the current value set for the nominal design. You can modify this value for each statistical setup.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Warning:**
>
> Variable values must be single real numbers, or expressions that evaluate to single real numbers. Complex numbers cannot be used as the values of variables in any optimetric analysis.

1. In the **Setup Statistical Analysis** dialog box, click the **Variables** tab. All of the variables selected for the statistical analysis appear.
2. Type a new value in the **Starting Value** text box for the value to override, then press **Enter**.

The **Override** option is now selected. This indicates that the value you entered will be used for this statistical analysis; the current value set for the nominal model will be ignored.

- Alternatively, you can select the **Override** option first, then type a new variable value in the **Starting Value** text box.

3. Optionally, click a new unit system in one of the **Units** text boxes.

To revert to the default starting value, clear the **Override** option.

Related Topics

Setting up a Statistical Analysis

Solving a Parametric Setup during a Statistical Analysis

Solving a parametric setup during a statistical analysis is useful when you want Optimetrics to solve every design variation in the parametric setup at each statistical analysis iteration.

To solve a parametric setup during a statistical analysis:

1. Open the **Setup Statistical Analysis** dialog box.
2. Click the **General** tab.
3. Click the parametric setup you want Optimetrics to solve during the statistical analysis from the **Parametric Analysis** drop-down list.
4. Select **Solve the parametric sweep during analysis**.

Related Topics

Setting up a Statistical Analysis

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

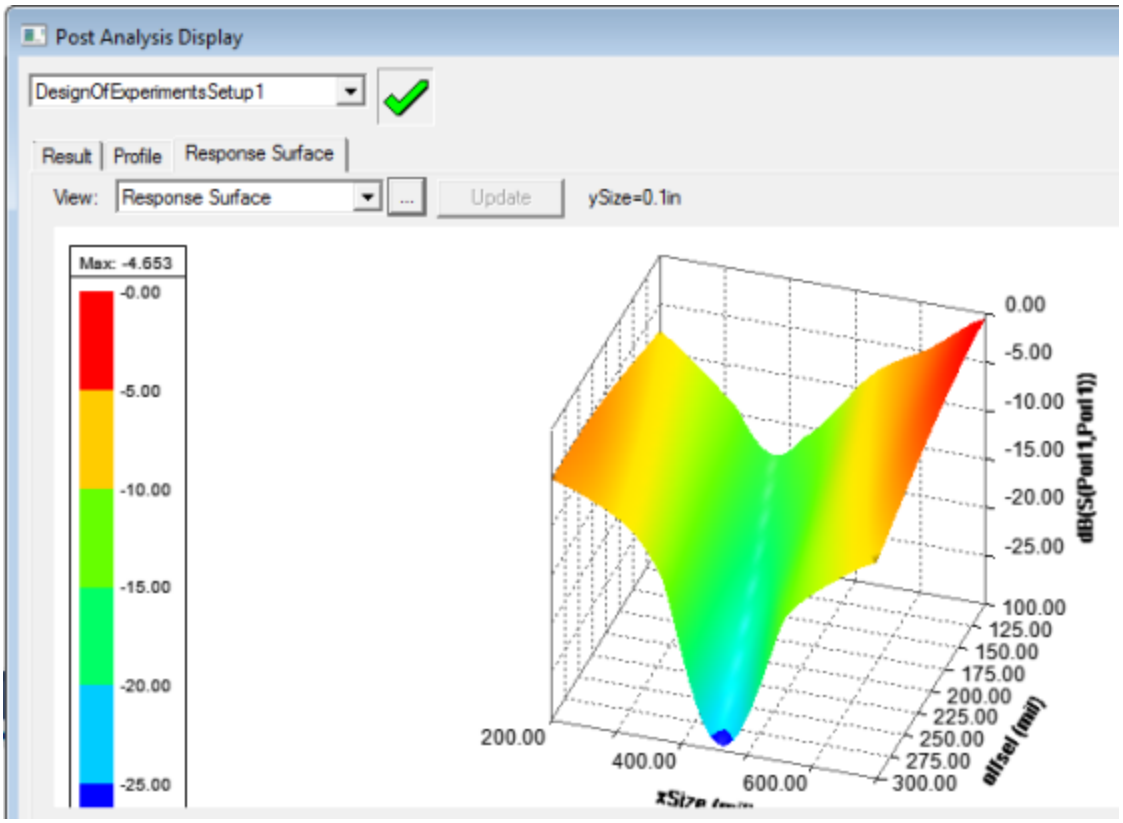
Using Design of Experiments

Use the Design of Experiments (DOE) technique to scientifically determine the location of sampling points. It is included as part of the Response Surface, Goal Driven Optimization, and Analysis systems. Design of Experiments used with a mathematical approximation of output parameters lets you:

- Reduce the number of simulations.
- Interactively explore the design space before running optimization.

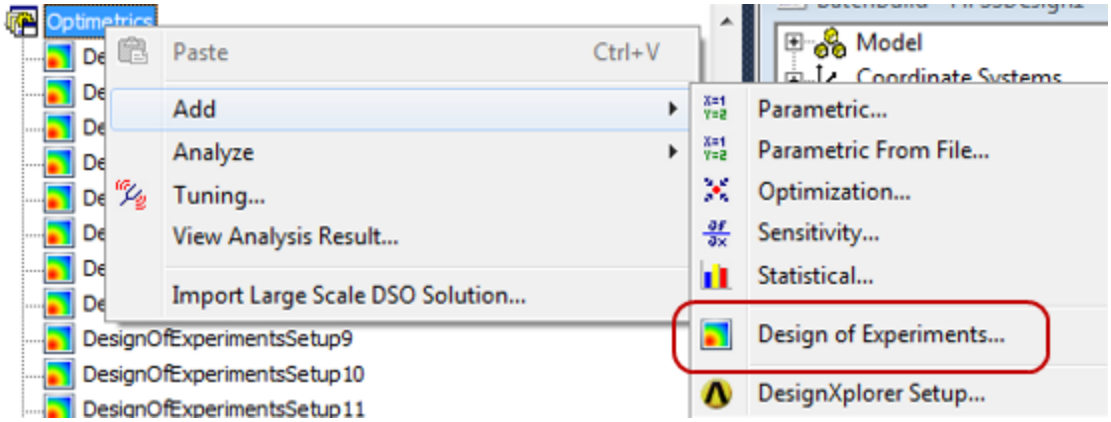
Design of Experiments describes the relationship between the design variables and the performance of the product by using DOE, combined with response surfaces. DOE and response surfaces provide all of the information required to achieve Simulation Driven Product Development. Once the variation of the performance with respect to the design variables is known, it becomes easy to understand and identify all changes required to meet the requirements for the product.

The goal is to create a response surface by interpolating through calculated points (a best curve fit). For each design, you can create a response surface for each output parameter. Once the response surfaces are created, you can share the information in easily understandable terms: curves, surfaces, sensitivities, and so on. They can be used at any time during the development of the product without requiring additional simulations to test a new configuration.

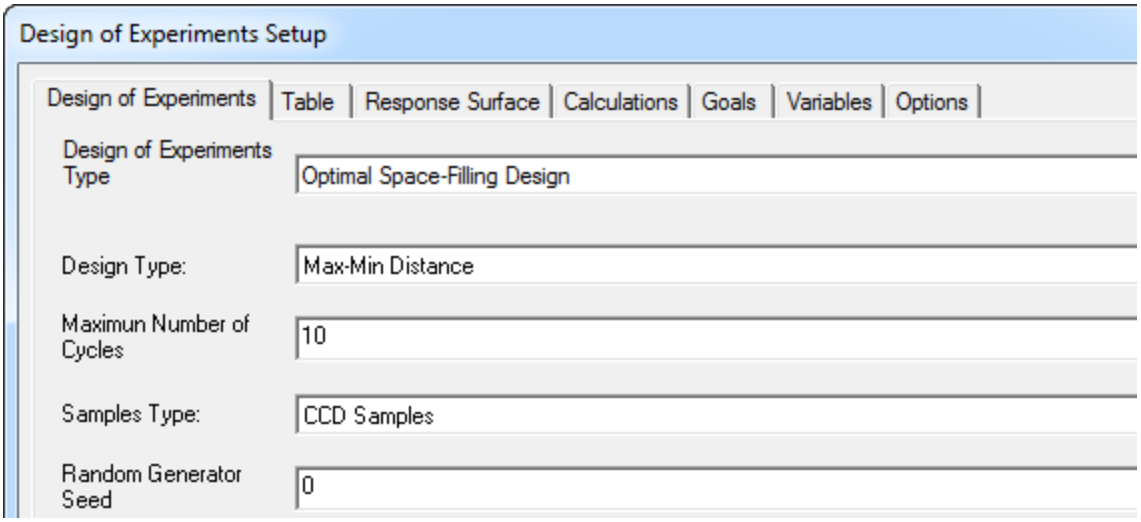


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The Design of Experiments feature is integrated inside Electronics Desktop. Combined with Electronics Desktop‘s distributed solve feature, you can build the response surfaces from the DOE variation table much faster.



Selecting a Design of Experiments under Optimization opens a dialog box with several tabs:



In the Design of Experiments setup, you select the DOE type, select the Response Surface, specify goals, view and include variables.

There are a wide range of DOE algorithms or methods available in engineering literature. These techniques all have one common characteristic: they try to locate the sampling points such that the space of random input parameters is explored in the most efficient way, or obtain the required information with a minimum of sampling points. Sample points in efficient locations only reduce the required number of sampling points and increases the accuracy of the response surface generated. For more information on the available types of DOE, see ***Design of Experiments Tab***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Once you have set up your input parameters, you can update the DOE, which submits the generated design points to the analysis system for solution. Design points are solved simultaneously if the analysis system is set up to do so; they are solved sequentially if not. After the solution is complete, you can update the Response Surface cell, which generates response surfaces for each output parameter based on the data in the generated design points.

> **Note:**
>
> Requirements and recommendations regarding the number of input parameters vary according to DOE type. For more information, see ***Design of Experiments Tab***.

If you change the Design of Experiments type after doing an initial analysis and preview the Design of Experiments Table, any design points generated for the new algorithm that are the same as design points solved for a previous algorithm will appear to be up to date. Only the design points that are different from any previously submitted design points must be solved.

You should set up your DOE Properties before generating your DOE Design Point matrix. The following topics describe setting up and solving your Design of Experiments, and viewing the results.

Related Topics

Setting Up Design of Experiments

Viewing the Result for Design of Experiments

Setup Calculations for Optimetrics

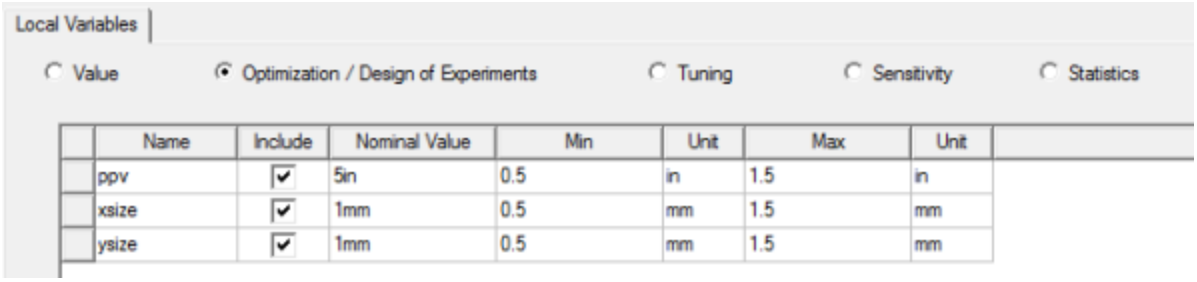
Using the Fast Calculation-Update Algorithm

Setting Up Design of Experiments

Follow this procedure to set up a Design of Experiments (DOE) analysis:

Choose the Variables for Design Exploration

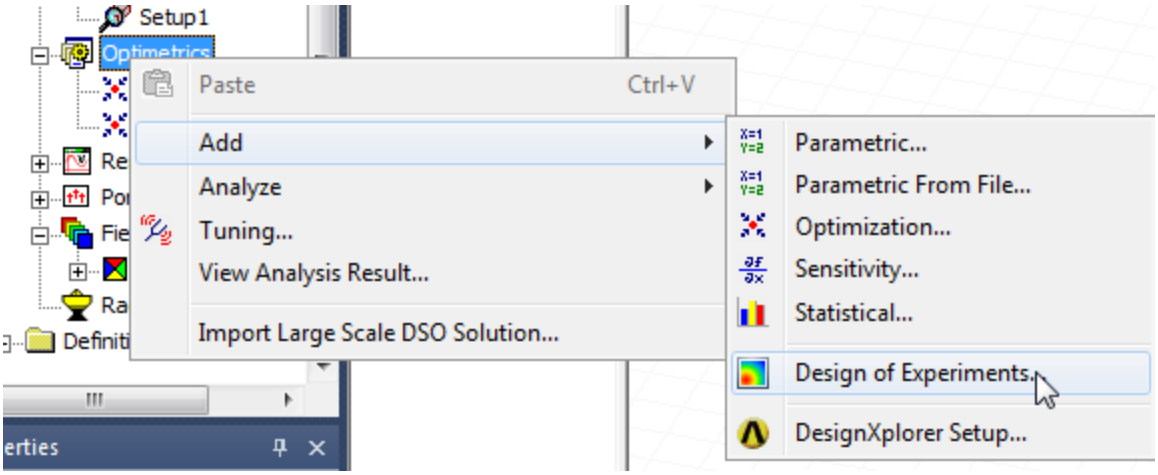
You must define local or project variables as **Optimization / Design of Experiments** variables for the Design of Experiments setup to include the variable.



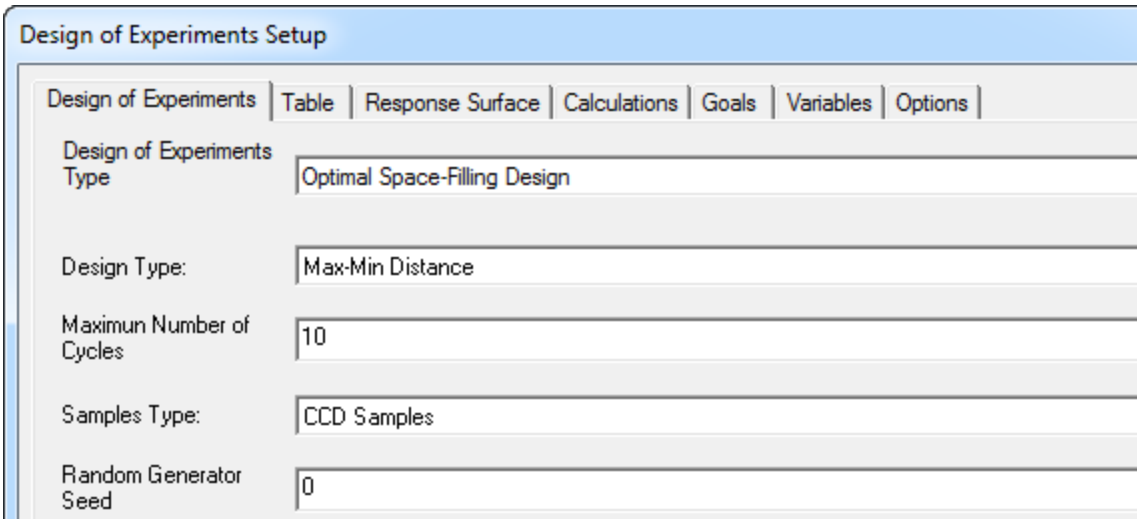
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Add the Design of Experiments Setup

Right-click **Optimetrics** in the **Project Manager** pane and select **Add** > **Design of Experiments**. You can also use **Twin Builder** > **Optimetrics Analysis** > **Add Design of Experiments**.



The **Design of Experiments Setup** dialog box appears.



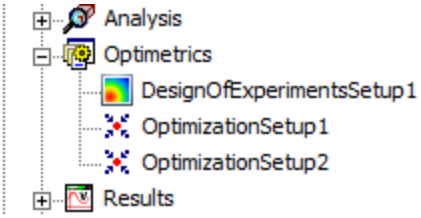
It has tabs for:

- **Design of Experiments** – Specify the sampling points and associated parameters.
- **Table** – Show the sampling points defined by the DOE settings you provide. If you selected a custom DOE type in the **Design of Experiments** tab, you can import data files as well as add or edit rows. You can also export files of DOE sampling points you have defined by any method.
- **Response Surface** – Specify the response surface type and refinement parameters.

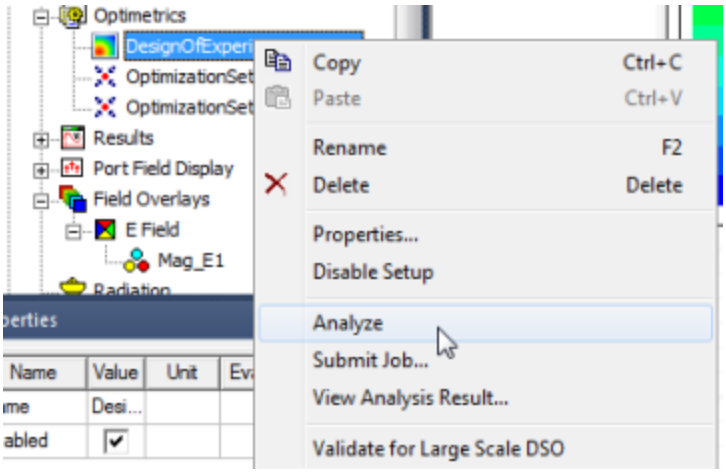
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Calculations** – Access Optimetrics calculations.
- **Goals** – Also includes cost functions calculations and norm type.
- **Variables** – Previously defined for Optimization/Design of Experiments as project or design variables, and whether to include them, treat as discrete, whether to use manufacturable variables, and the available levels.
- **Options** – To save solutions to the database and to use fast calculation-update algorithm.

Once you have set parameters and click **OK**, the **Design of Experiments** setup appears under the Optimetrics icon in the Project tree.



From here, right-click the DOE setup and select **Analyze**, **Submit Job**, or **Validate for Large Scale DSO**.



Related Topics

Using Design of Experiments

Design of Experiments Tab

Table Tab for Design of Experiments

Response Surface Tab for Design of Experiments

Variables Tab for Design of Experiments

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

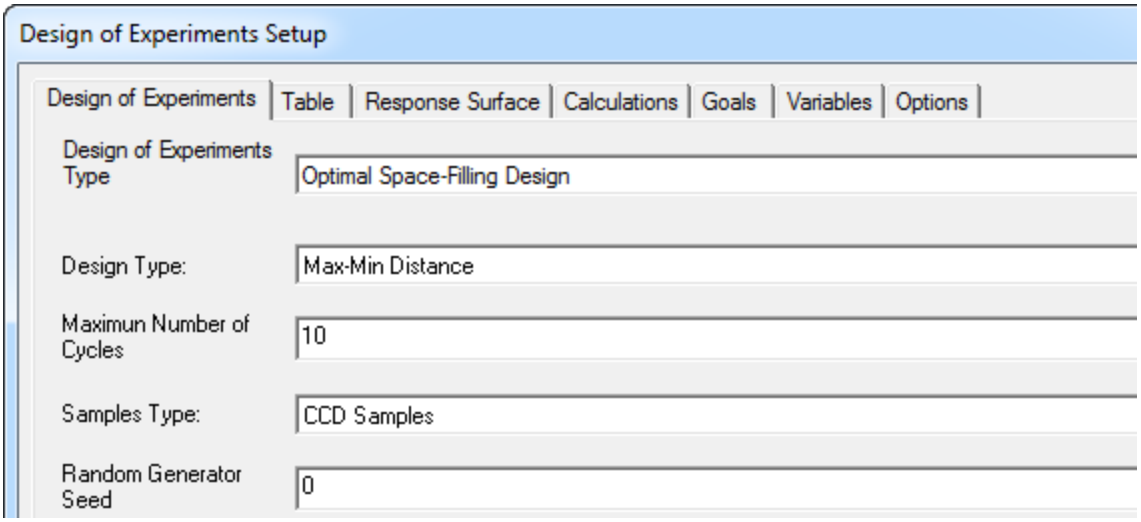
Setup Calculations for Optimetrics

Using the Fast Calculation-Update Algorithm

View Analysis Results for Design of Experiments

Design of Experiments Tab

The **Design of Experiments** tab in the **Design of Experiments Setup** dialog box includes selections for defining the sampling points that define your experiment. Each selection for **Design of Experiments Type** has a different set of associated parameters so the appearance of the dialog box changes to show the parameters for your selection.



The DOE types available in the Desktop include:

Design of Experiments Types	Brief Description (click links for more details)
Optimal Space Filling (Default)	An optimized Latin Hypercube Sampling maximizing distance between experiments. Several design type criteria are available: • Max-Min Distance • Centered L2 • Maximum Entropy Several sampling types available which determine the number of samples in the design: • CCD samples (Central Composite Designs are five level factorial designs that are suitable for calibrating the quadratic response

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Design of Experiments Types	Brief Description (click links for more details)
	model) • Linear model samples • Pure quadratic model samples • Full quadratic model samples • User-Defined samples You also specify a Maximum Number of Cycles and a Random Generator Seed.
Central Composite Design	Several design types available: • Face-Centered • Rotatable • VIF-Optimality • G-Optimality • Auto-defined You can also choose a Standard or Enhanced Template.
Box-Behnken	Avoids critical configurations in the corner of the design space. Maximum number of input parameters is 12.
Custom	Lets you customize a DOE matrix by editing values, adding or removing samples, and/or importing samples from a CSV file. Select **Custom** to enable the **Import** button and the **Table** tab, as well as buttons to **Add editable rows** or **Delete selected rows**. If you previously solved the DOE using one of the other algorithms, those design points are retained and you can add new design points to the table. You can also import and export design points into the custom DOE table from the parameter set.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Design of Experiments Types	Brief Description (click links for more details)
Latin Hypercube Sampling	Statistical design where no two experiments share input parameters of the same value. The samples Type can be: • CCD Samples • Linear Model Samples • Pure Quadratic Model Samples • Full Quadratic Model Samples • User-Defined Samples, for which you also specify the Number of Samples. For each samples type, you also specify a Random Generator Seed.

The **Table tab** provides a preview view of the design points defined by your selections.

Related Topics

Setting Up Design of Experiments

Optimal Space Filling Design (OSF)

The goal in Design of Experiments (DOE) is to determine the smallest sufficient set of points required to calculate a response surface. Therefore, you choose the type depending on the parametric problem and targeted response surface. The number of points depends on the number of input parameters, or is user-defined.

Optimal Space-Filling Design (OSF) creates optimal space filling DOE plans according to some specified criteria. Essentially, OSF is a Latin Hypercube Sampling Design (LHS) that is extended with post-processing. It is initialized as an LHS and optimized several times, remaining a valid LHS (without points sharing rows or columns) while achieving a more uniform space distribution of points (maximizing the distance between points).

To offset the noise associated with physical experimentation, classical DOE types such as CCD focus on parameter settings near the perimeter of the design region. Because computer simulation is not quite as subject to noise, though, the OSF design can distribute the design parameters equally throughout the design space with the objective of gaining the maximum insight into the design with the fewest number of points. This advantage makes it appropriate when a more complex meta-modeling technique such as Kriging, Non-Parametric Regression, or Neural Networks is used.

OSF shares some of the same disadvantages as LHS, though to a lesser degree. Possible disadvantages of an OSF design are:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- When the CCD Samples sample type is selected, a maximum of 20 input parameters is supported.
- Extremes, such as the corners of the design space, are not necessarily covered.
- The selection of too few design points can result in a lower quality of response prediction.

The following properties are available for the OSF DOE type.

- **Design Type** – The following choices are available:
  - **Max-Min Distance** (default) – Maximizes the minimum distance between any two points. This strategy ensures that no two points are too close to each other. For a small size of sampling (N), the Max-Min Distance design generally lies on the exterior of the design space and fill in the interior as N becomes larger. Generally, this is the faster algorithm.
  - **Centered L2** – Minimizes the centered L2-discrepancy measure. The discrepancy measure corresponds to the difference between the empirical distribution of the sampling points and the uniform distribution. This means that the centered L2 yields a uniform sampling. This design type is computationally faster than the **Maximum Entropy** type.
  - **Maximum Entropy** – Maximizes the determinant of the covariance matrix of the sampling points to minimize uncertainty in unobserved locations. This option often provides better results for highly correlated design spaces. However, its cost increases non-linearly with the number of input parameters and the number of samples to be generated. Thus, it is recommended only for small parametric problems.
- **Maximum Number of Cycles** – Determines the number of optimization loops the algorithm needs, which in turn determines the discrepancy of the DOE. The optimization is essentially combinatorial, so a large number of cycles slows down the process. However, this makes the discrepancy of the DOE smaller. For practical purposes, 10 cycles is generally good for up to 20 variables. The value must be greater than 0. The default is 10.
- **Samples Type** – Determines the number of DOE points the algorithm should generate. This option is suggested if you have some advanced knowledge about the nature of the metamodel. The following choices are available:
  - **CCD Samples** (default) – Supports a maximum of 20 inputs. Generates the same number of samples a CCD DOE would generate for the same number of inputs. You can use this to generate a space filling design that has the same cost as a corresponding CCD design.
  - **Linear Model Samples** – Generates the number of samples as needed for a linear metamodel.
  - **Pure Quadratic Model Samples** – Generates the number of samples as needed for a pure quadratic metamodel (no cross terms).
  - **Full Quadratic Samples** – Generates the number of samples needed to generate a full quadratic model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

  - **User-Defined Samples** – Specify the desired number of samples.
- **Seed Value** – Set the value used to initialize the random number generator invoked internally by the LHS algorithm. Although the generation of a starting point is random, the seed value consistently results in a specific LHS. This property lets you generate different samplings by changing the value or regenerate the same sampling by keeping the same value. The default is 0.
- **Number of Samples** – Enabled when **Samples Type** is set to **User-Defined Samples**. Specifies the default number of samples. The default is 10.

Related Topics

Setting Up Design of Experiments

Central Composite Design (CCD)

The goal in Design of Experiments (DOE) is to determine the smallest sufficient set of points required to calculate a response surface. Therefore, you choose the type depending on the parametric problem and targeted response surface. The number of points depends on the number of input parameters, or is user defined.

Central Composite Design (CCD) provides a screening set to determine the overall trends of the metamodel to better guide the choice of options in Optimal Space-Filling Design. The CCD DOE type supports a maximum of 20 input parameters.

The following properties are available for the CCD DOE type.

- **Design Type** – By specifying the **Design Type** for CCD, you can help to improve the response surface fit for DOE studies. For each CCD type, the alpha value is the location of the sampling point that accounts for all quadratic main effects. The following CCD design types are available:
  - **Face-Centered** – A three-level design with no rotatability. The alpha value equals 1.0. A Template Type setting appears, with Standard and Enhanced options. Choose Enhanced for a possible better fit for the response surfaces.
  - **Rotatable** – A five-level design that includes rotatability. The alpha value is calculated based on the number of input variables and a fraction of the factorial part. A design with rotatability has the same variance of the fitted value regardless of the direction from the center point.
  - **VIF-Optimality** – A five-level design in which the alpha value is calculated by minimizing a measure of non-orthogonality known as the Variance Inflation Factor (VIF). The more highly correlated the input variable with one or more terms in a regression model, the higher the VIF.
  - **G-Optimality** – Minimizes a measure of the expected error in a prediction and minimizes the largest expected variance of prediction over the region of interest.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Auto-Defined** – Design exploration selects the Design Type based on the number of input variables. Use of this option is recommended for most cases as it switches between the G-Optimality if the number of input variables is 5 or VIF-Optimality otherwise.
  However, you can use the Rotatable design if the default option does not provide good values for the Goodness of Fit from the response surface plots. Additionally, you can use the Enhanced template if the default Standard template does not fit the response surfaces well.
- **Template Type** – Enabled for the Rotatable and Face-Centered design types. The following options are available:
  - **Standard**
  - **Enhanced** – Choose this option for a possible better fit for the response surfaces.

Related Topics

Setting Up Design of Experiments

Box-Behnken Design (CCD)

The goal in Design of Experiments (DOE) is to determine the smallest sufficient set of points required to calculate a response surface. Therefore, you choose the type depending on the parametric problem and targeted response surface. The number of points depends on the number of input parameters, or is user defined.

A **Box-Behnken Design** is a three-level quadratic design that does not contain fractional factorial design. The sample combinations are treated in such a way that they are located at midpoints of edges formed by any two factors. The design is rotatable (or in cases, nearly rotatable).

One advantage of a Box-Behnken design is that it requires fewer design points than a full factorial CCD and generally requires fewer design points than a fractional factorial CCD. Additionally, a Box-Behnken Design avoids extremes, so you can work around extreme factor combinations. Consider using the Box-Behnken Design DOE type if your project has parametric extremes (for example, has extreme parameter values in corners that are difficult to build). Since the Box-Behnken DOE doesn't have corners and does not combine parametric extremes, it can reduce the risk of update failures.

Possible disadvantages of a Box-Behnken design are:

- Prediction at the corners of the design space is poor and that there are only three levels per parameter.
- A maximum of 12 input parameters is supported.

No additional properties are available for the Box-Behnken Design DOE type.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Setting Up Design of Experiments

Custom DOE Type

The goal in Design of Experiments (DOE) is to determine the smallest sufficient set of points required to calculate a response surface. Therefore, you choose the type depending on the parametric problem and targeted response surface. The number of points depends on the number of input parameters, or is user defined.

The **Custom DOE** type allows for definition of a custom DOE Table. You can manually add new design points, entering the input and (optionally) output parameter values directly into the table. If you previously solved the DOE using one of the other algorithms, those design points are retained and you can add new design points to the table. You can also import and export design points into the custom DOE Table from the parameter set.

You can change the edition mode of the DOE table to edit the output parameter values. You can also right-click and select **Import Design Points** to copy and paste data and import data from a CSV file.

Related Topics

Setting Up Design of Experiments

Latin Hypercube Sampling

The goal in Design of Experiments (DOE) is to determine the smallest sufficient set of points required to calculate a response surface. Therefore, you choose the type depending on the parametric problem and targeted response surface. The number of points depends on the number of input parameters, or is user defined.

In the **Latin Hypercube Sampling** design DOE type, the DOE is generated by the LHS algorithm, an advanced form of the Monte Carlo sampling method that avoids clustering samples. In a Latin Hypercube Sampling, the points are randomly generated in a square grid across the design space, but no two points share the same value. This means that no point shares a row or a column of the grid with any other point.

Possible disadvantages of a Latin Hypercube Sampling design are:

- When the CCD Samples sample type is selected, a maximum of 20 input parameters is supported. For more information, see ***Design of Experiments Tab***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Extremes, such as the corners of the design space, are not necessarily covered. Additionally, the selection of too few design points can result in a lower quality of response prediction.

Note: The Optimal Space-Filling Design DOE type is an LHS design that is extended with post-processing.

The following properties are available for the LHS DOE type:

- **Samples Type** – Determines the number of DOE points the algorithm should generate. This option is suggested if you have some advanced knowledge about the nature of the metamodel. The following choices are available:
  - **CCD Samples** (default) – Supports a maximum of 20 inputs. Generates the same number of samples a CCD DOE would generate for the same number of inputs. You can use this to generate a space filling design that has the same cost as a corresponding CCD design.
  - **Linear Model Samples** – Generates the number of samples as needed for a linear metamodel.
  - **Pure Quadratic Model Samples** – Generates the number of samples as needed for a pure quadratic metamodel (no cross terms).
  - **Full Quadratic Samples** – Generates the number of samples needed to generate a full quadratic model.
  - **User-Defined Samples** – Specify the desired number of samples.
- **Seed Value** – Set the value used to initialize the random number generator invoked internally by the LHS algorithm. Although the generation of a starting point is random, the seed value consistently results in a specific LHS. This property lets you generate different LHS samplings (by changing the value) or to regenerate the same LHS sampling (by keeping the same value). The default is 0.
- **Number of Samples** – Enabled when **Samples Type** is set to **User-Defined Samples**. Specifies the default number of samples. The default is 10.

Related Topics

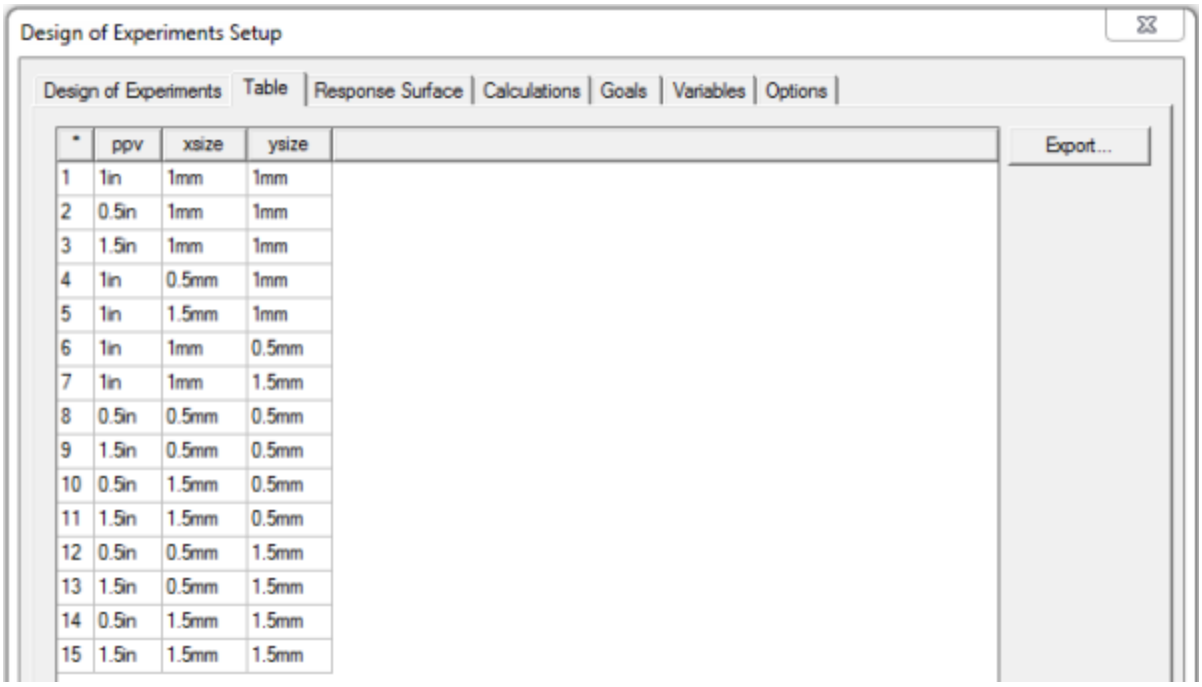
Setting Up Design of Experiments

Table Tab for Design of Experiments

The **Table** tab in the **Design of Experiments Setup** dialog box displays a preview of the design points designed by your selections on the **Design of Experiments** tab. There is one column for

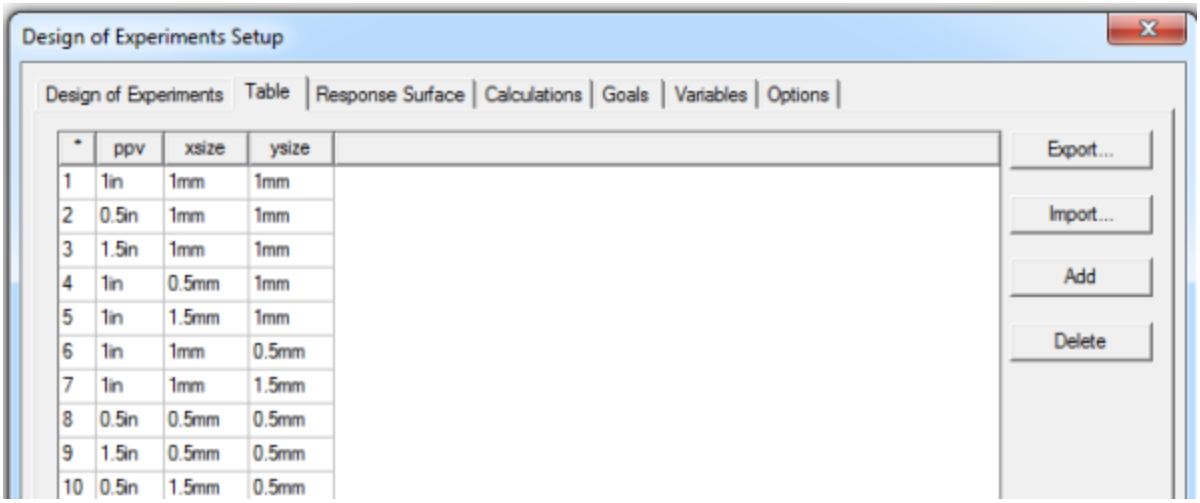
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

each defined variable. Click **Export** to create a file of the table in a format you specify.



*	ppv	xsize	ysize
1	1in	1mm	1mm
2	0.5in	1mm	1mm
3	1.5in	1mm	1mm
4	1in	0.5mm	1mm
5	1in	1.5mm	1mm
6	1in	1mm	0.5mm
7	1in	1mm	1.5mm
8	0.5in	0.5mm	0.5mm
9	1.5in	0.5mm	0.5mm
10	0.5in	1.5mm	0.5mm
11	1.5in	1.5mm	0.5mm
12	0.5in	0.5mm	1.5mm
13	1.5in	0.5mm	1.5mm
14	0.5in	1.5mm	1.5mm
15	1.5in	1.5mm	1.5mm

If you have specified **Custom** as the Design of Experiments Type, the table is editable and the **Table** tab includes buttons for adding and deleting rows. The added rows are editable. You can add new rows by entering values in the * row of the table. You enter values in the input parameter columns. Once you have entered a value in one column in the * row, the row is added to the table and the values for the remaining input parameters are set to the initial values of the parameters. You can then edit that row in the table and change any of the other input parameter values if needed. Output parameter values are then calculated when the design is solved or updated.



*	ppv	xsize	ysize
1	1in	1mm	1mm
2	0.5in	1mm	1mm
3	1.5in	1mm	1mm
4	1in	0.5mm	1mm
5	1in	1.5mm	1mm
6	1in	1mm	0.5mm
7	1in	1mm	1.5mm
8	0.5in	0.5mm	0.5mm
9	1.5in	0.5mm	0.5mm
10	0.5in	1.5mm	0.5mm

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Depending on the context, the tables are read-only and filled automatically or they are partially or completely editable. The background color of a cell indicates if it is editable or not:

- A gray background indicates a read-only cell.
- A white background indicates an editable cell.

Output parameter values calculated from a simulation (a design point update) appear in black text.

The Custom Table view also includes an **Import** button. Import and export files can be:

- Comma delimited data (CSV) files.
- Tab delimited data (TAB) files.
- Ansoft Plot Data (DAT) files.
- Post Processor format data (TXT) files.

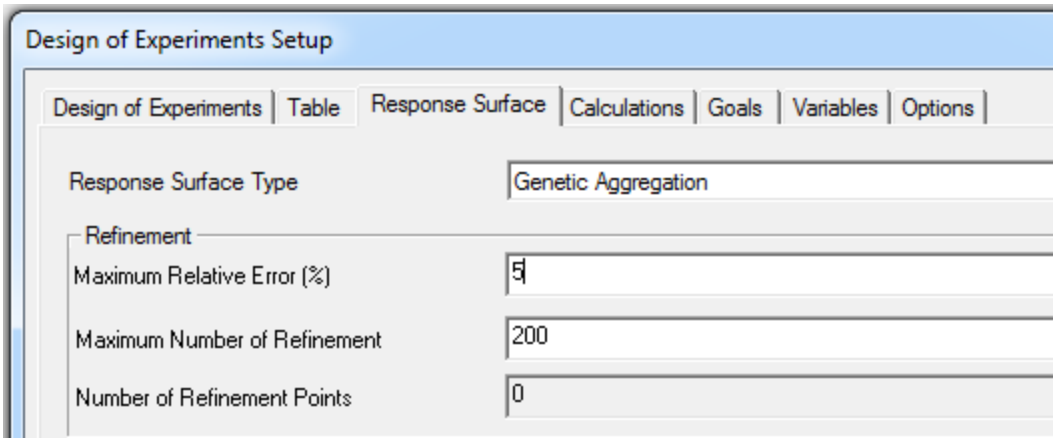
The table updates when you change your selections on the **Design of Experiments** tab.

Related Topics

Custom DOE Type

Response Surface Tab for Design of Experiments

The **Response Surface** tab in the **Design of Experiments Setup** dialog box lets you select the Response Surface type as Genetic Aggregation or Standard Response Surface.



The selection here specifies the refinement applied to the initial Design of Experiments (DOE). The Genetic Aggregation Response Surface finds the best possible response surface for each output by combining:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Metamodels
- Settings
- Kernal Variation
- Polynomial Regression

For each output, a fitness factor works to minimize error, including cross-validation errors. The automatic refinement adds design points to the DOE until the response surface accuracy meets user requirements. You can specify requirements for:

- Maximum Relative Error %
- Maximum Number or Refinement attempts

You do have the option of selecting Standard Response Surface- Full 2nd Order Polynomial.

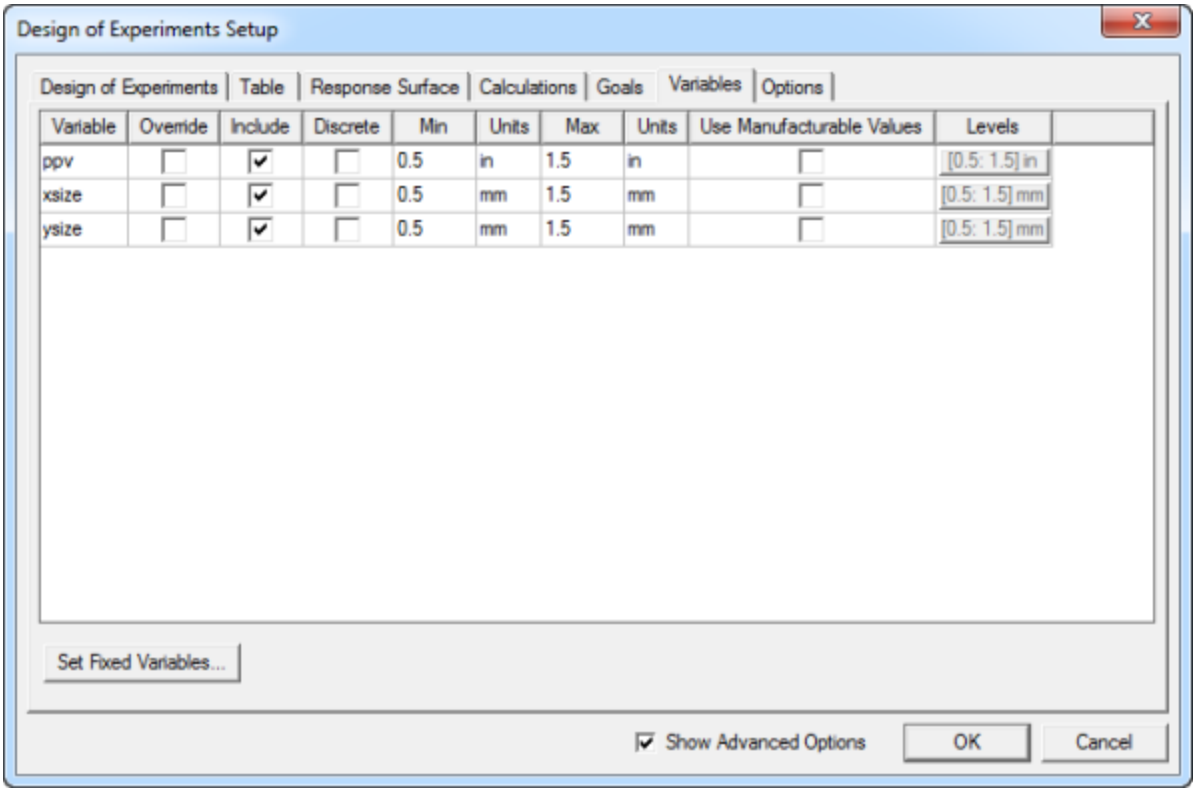
After you have completed an analysis, you can view the generated plot.

Related Topics

Setting Up Design of Experiments

Variables Tab for Design of Experiments

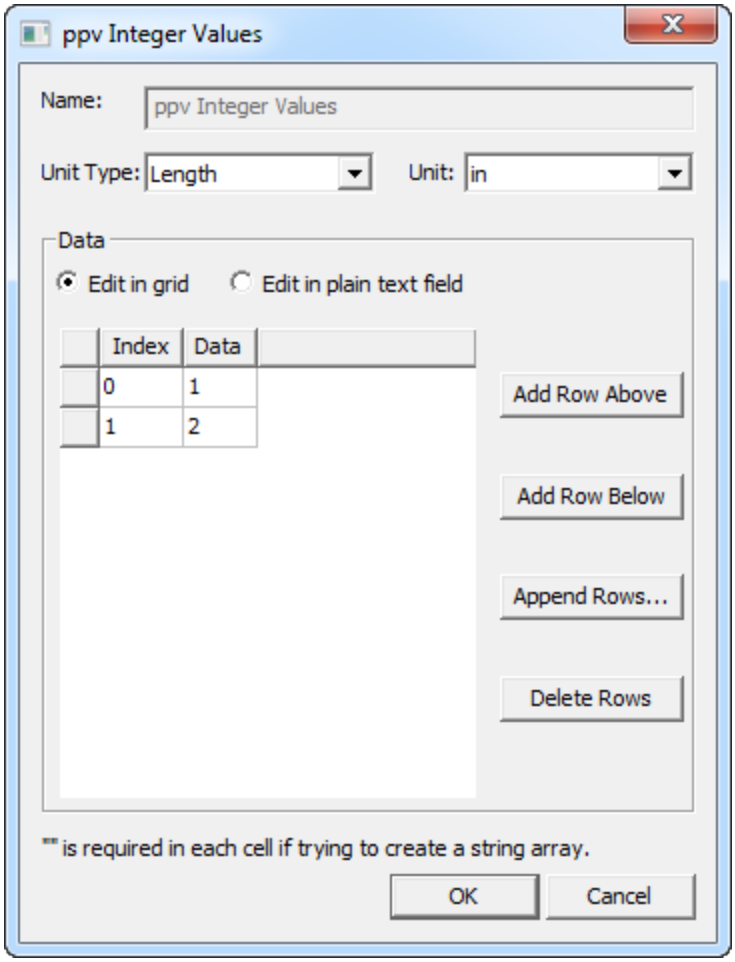
The **Variables** tab in the **Design of Experiments Setup** dialog box displays a list of the variables defined for the design and project as Optimetrics/Design of Experiments variables.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The columns list the variable names, the current minimum and maximum values and units, and provides options for the following:

- **Override** – Select this check box to override the current design value. Clearing this check box causes a prompt to appear asking you to confirm the return to the design value.
- **Include** – Choose whether to include the variable in an analysis.
- **Discrete** – **Discrete Variables** physically represent different configurations or states of the model. Select the check box in the **Discrete** column to enable the button in the **Levels** column. The discrete values can be bounded by a minimum/maximum range and/or manufacturable values. Click **Levels** for the row to edit the discrete values. An edit dialog box for the variable appears.

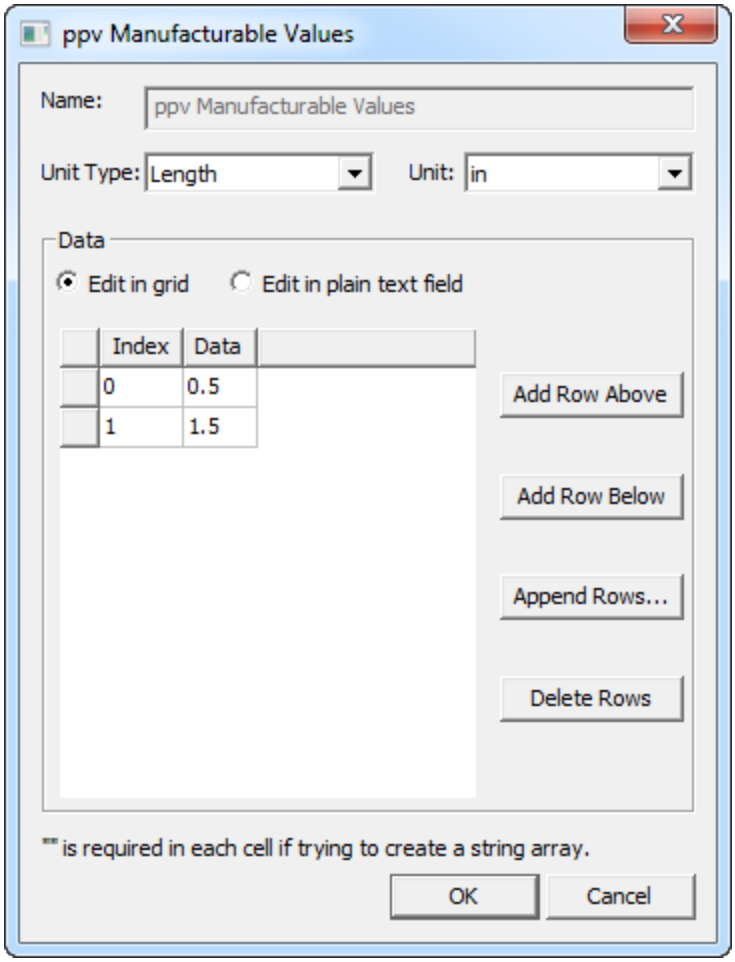


- **Continuous Variables** – These variables physically vary in a continuous manner between a lower and an upper bound (min/max) defined by you. With continuous variables, you can also apply a manufacturable values filter to the variable. Manufacturable values represent real-world manufacturing or production constraints. The

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

minimum and maximum values are the upper and lower constraints on the available manufacturable values bounded by the range.

- Select the check box in the **Use Manufacturable Values** column to enable the button in the **Levels** column. Click the button to edit the values. An edit dialog box for the variable appears.



Related Topics

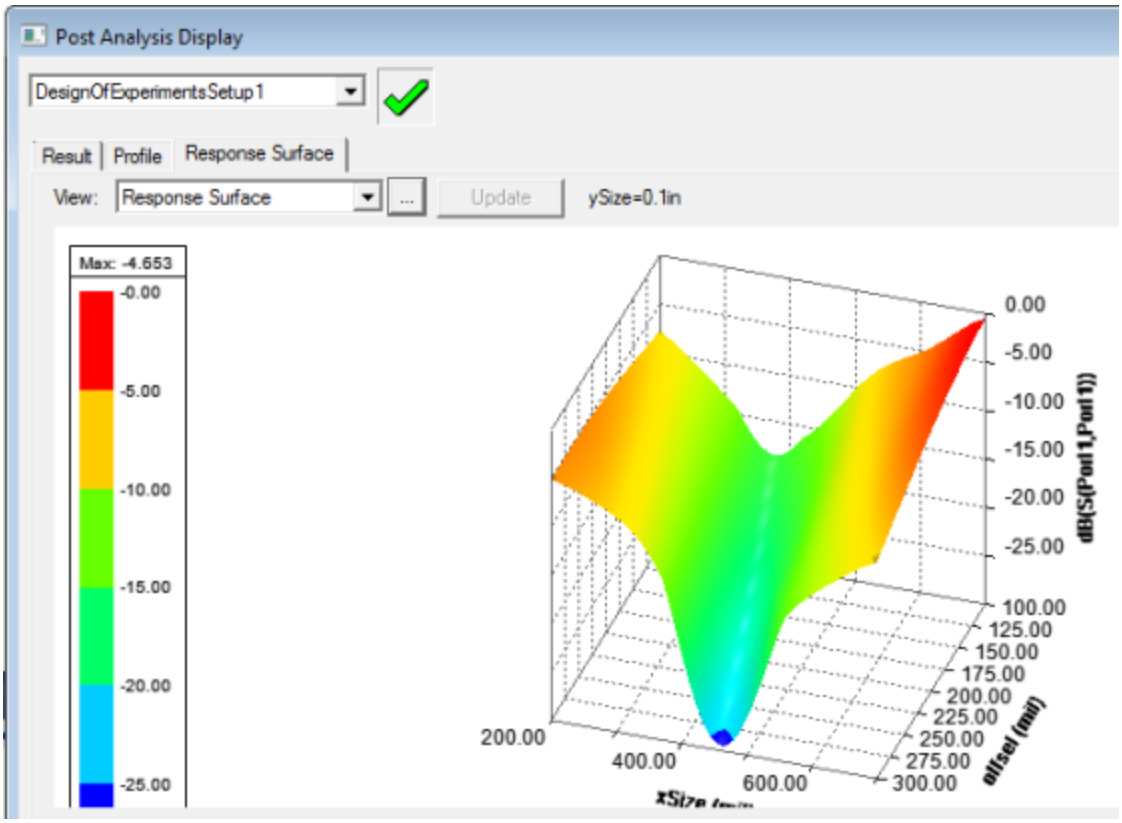
Setting Up Design of Experiments

View Analysis Result for Design of Experiments

After completing an analysis, right-click the Design of Experiments setup in the **Project Manager** pane and select **View Analysis Result**. This opens a dialog box that includes a **Result** tab that lists the variations and variable values, a solution **Profile** tab with start, stop, time elapsed, and machines used, as well as a **Response Surface** tab. When you include more than

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

one variable in the setup, response surface view is available. Click ... next to the view list box to choose any two variables as the X-, Y-axis, and choose an output calculation as the Z-axis.



From the **Response Surface** tab, the **View** list box lets you select all available views of the selected response-surface-setup.

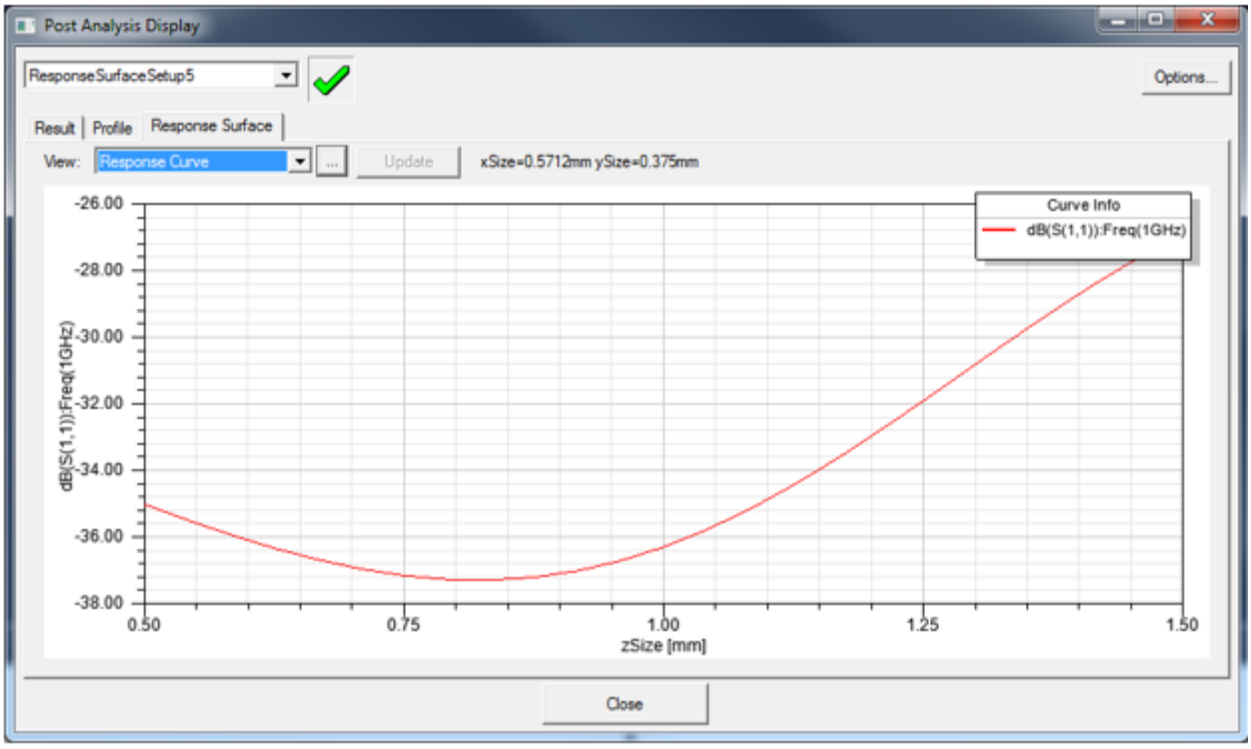
- Min Max Search
- Refinement points table
- Response points table
- Verification points table
- Goodness of Fit
- Response Curve (see below)
- Response Curve (2D Slices, see below)
- Response Surface

The **Update** button is disabled when the response surface is up to date. After the setup, if you modify a verification point or refinement points, it is enabled. Click **Update** to re-generate the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

response surface with new settings. It may start new simulations if any of the design points in the DOE, refinement points, or verification points has not been solved.

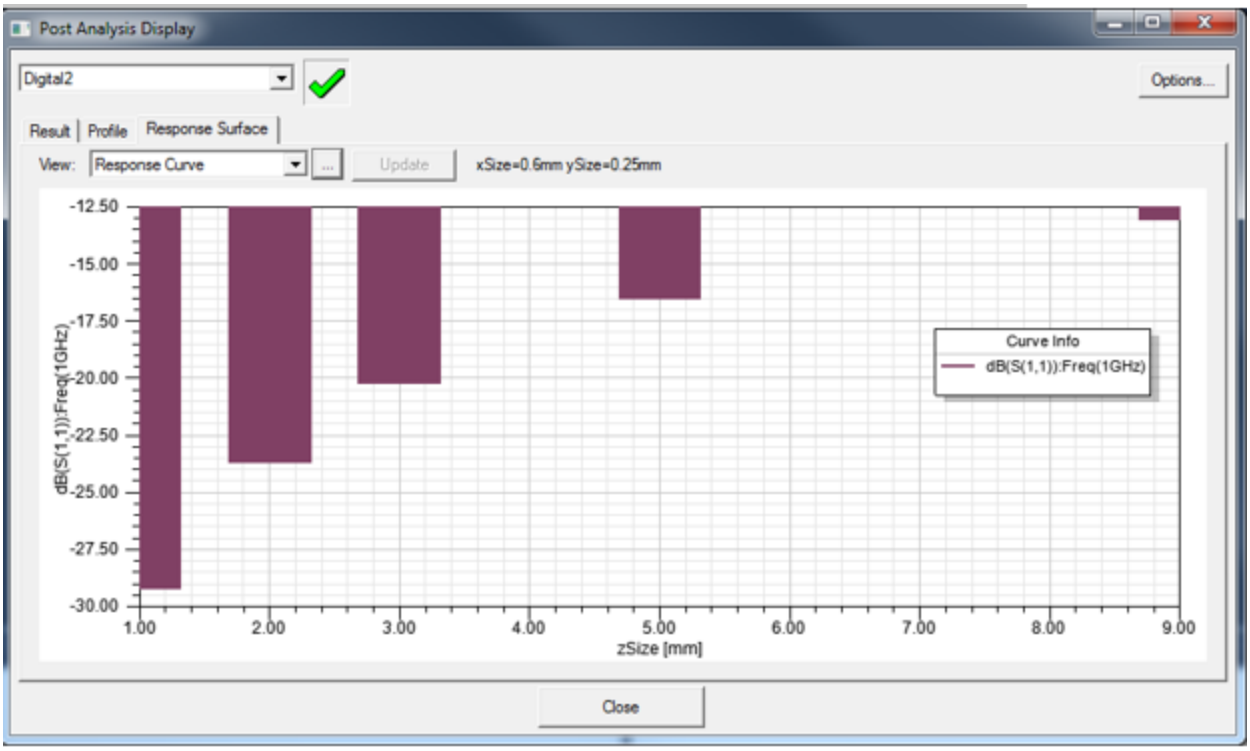
Response Curve



Discrete Variable

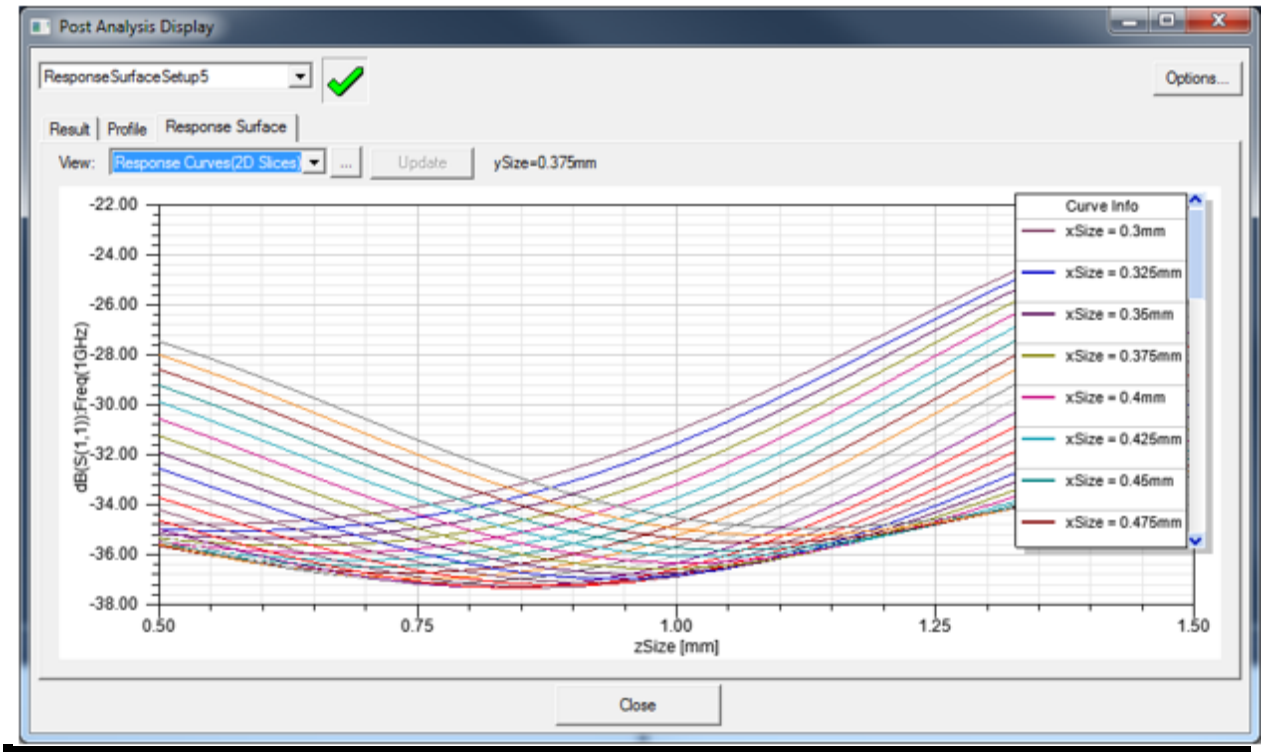
If you choose **Discrete Variable** as the X-axis, a bar chart plot appears:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Response Curves (2D Slices)

If you have included more than one variable in the setup, the Response Curves (2D) view is available:



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

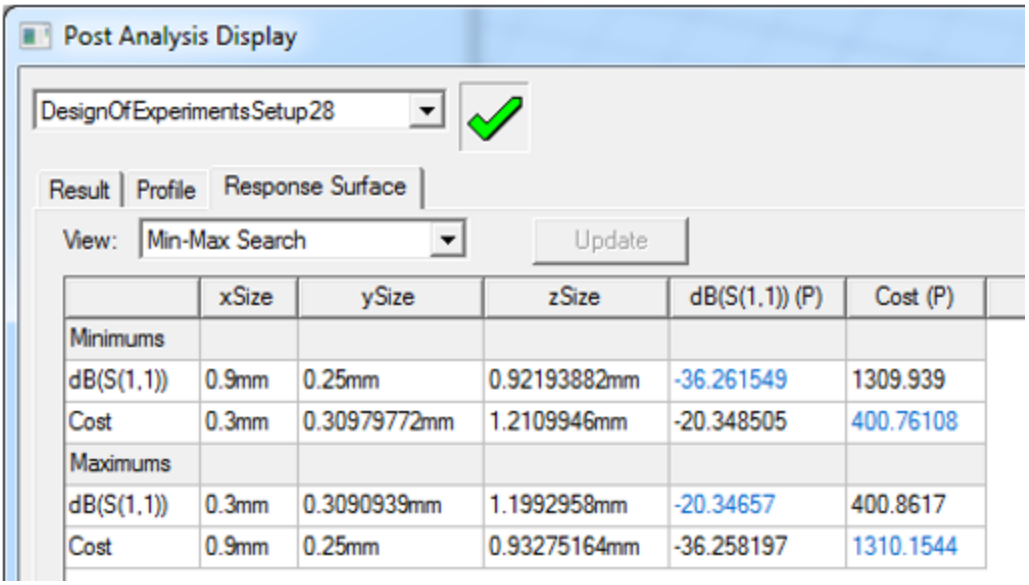
Using Design of Experiments

Setting Up Design of Experiments

Min-Max Search for Design of Experiments

In the **Response Surface** tab of the Design of Experiments **Post Analysis Display** dialog box, the **View** list box lets you select all available views of the selected response-surface-setup.

The **Min-Max Search** view examines the entire output parameter space from a response surface to approximate the minimum and maximum values of each output parameter. When you select a Min/Max row, the **Apply** button is enabled, and you can then apply the selected variation variable values to the variables' nominal values.



	xSize	ySize	zSize	dB(S(1,1)) (P)	Cost (P)
Minimums					
dB(S(1,1))	0.9mm	0.25mm	0.92193882mm	-36.261549	1309.939
Cost	0.3mm	0.30979772mm	1.2109946mm	-20.348505	400.76108
Maximums					
dB(S(1,1))	0.3mm	0.3090939mm	1.1992958mm	-20.34657	400.8617
Cost	0.9mm	0.25mm	0.93275164mm	-36.258197	1310.1544

You can export the table in the following file formats:

- Comma delimited data (CSV).
- Tab delimited data (TAB).
- Ansoft Plot data (DAT).
- Post Processor format data (TXT).

Related Topics

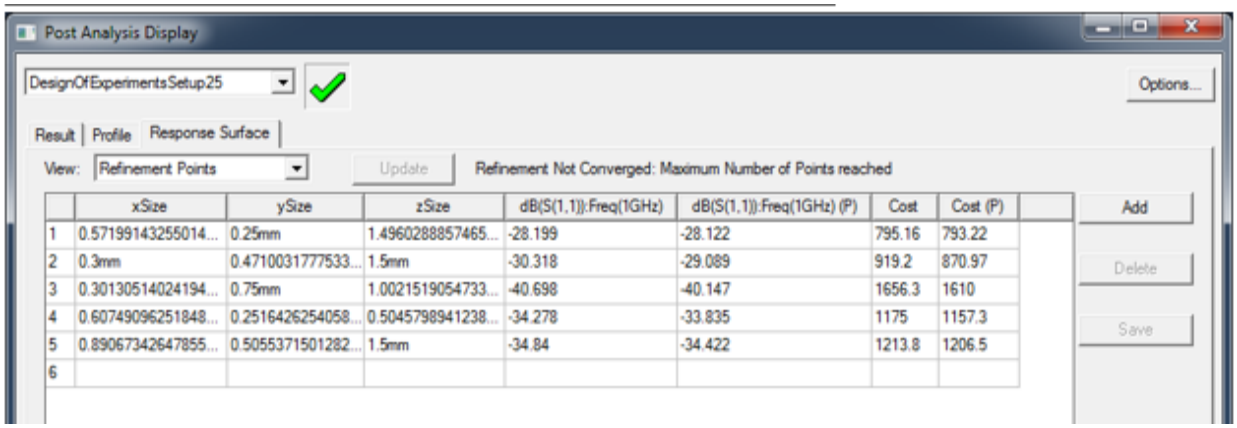
Using Design of Experiments

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Setting Up Design of Experiments

Refinement Points Table

From the **Response Surface** tab, the **View** list box lets you select all available views of the selected response-surface-setup. All refinement points are shown in the **Refinement Points** table view:



	xSize	ySize	zSize	dB(S(1,1)):Freq(1GHz)	dB(S(1,1)):Freq(1GHz) (P)	Cost	Cost (P)
1	0.57199143255014...	0.25mm	1.4960288857465...	-28.199	-28.122	795.16	793.22
2	0.3mm	0.4710031777533...	1.5mm	-30.318	-29.089	919.2	870.97
3	0.30130514024194...	0.75mm	1.0021519054733...	-40.698	-40.147	1656.3	1610
4	0.60749096251848...	0.2516426254058...	0.5045798941238...	-34.278	-33.835	1175	1157.3
5	0.89067342647855...	0.5055371501282...	1.5mm	-34.84	-34.422	1213.8	1206.5
6							

Refinement points are points added to your model to enrich and improve your response surface. They can be generated with the response surface update, or added manually, as described in **Performing a Manual Refinement** below. As with design points, DesignXplorer must perform a design point update (a "real solve") in order to obtain the output parameters for the refinement points.

Upon update, the refinement points build the response surface and are taken into account for the generation of verification points. Along with DOE points, refinement points are also learning points for Goodness of Fit calculations.

Performing a Manual Refinement

Manual refinement is a way to force the response surface to take into account points of your choice, in addition to the points already in the Design of Experiments. You can insert a refinement point in the **Refinement Points** table view, and do not need to do an initial solve of the response surface (without the refinement point) before updating your response surface with your manual refinement. Manual refinement is available for all response surface types except for Sparse Grid.

Click **Add** or **Delete** to add, delete, or modify refinement points; you can also modify the point values directly in the grid.

Manual Refinement Point can be inserted from the Response point table and Verification Points table.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

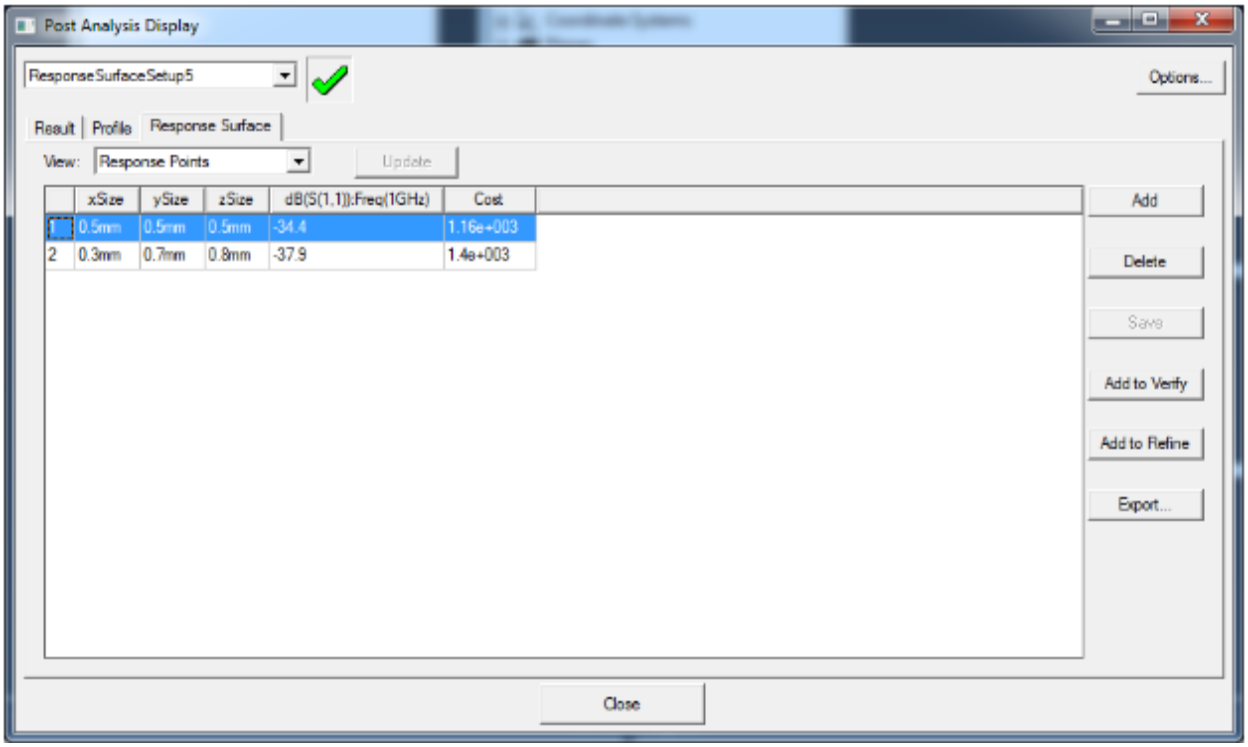
Related Topics

Setting Up Design of Experiments

Using Design of Experiments

Response Points Table

From the **Response Surface** tab, the **View** list box lets you select all available views of the selected response-surface-setup. All response points are shown in the **Response Points** table view.



A response point is defined by a snapshot of variable values where output calculation values were calculated in Ansys DesignXplorer from a response surface. As such, the output calculation (or cost) values are approximate and calculated from response surfaces.

You can use the buttons to **Add**, **Delete**, **Save**, or **Export** response points, or you can modify them directly in the grid.

Click **Add to Verify** or **Add to Refine** to insert the selected response point to the verification table or refinement table.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

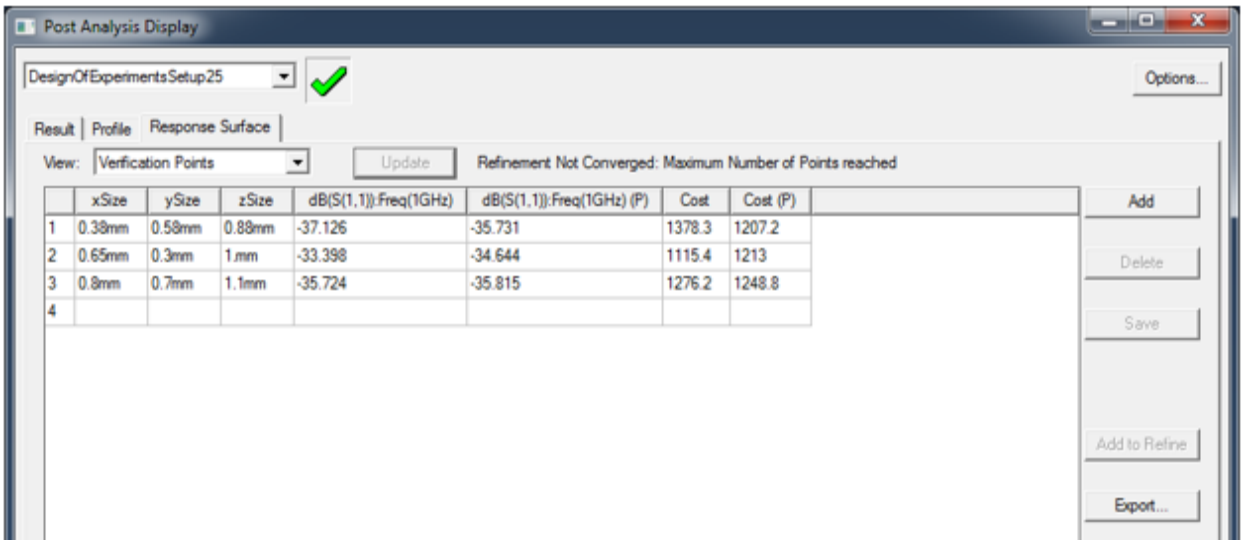
Related Topics

Setting Up Design of Experiments

Using Design of Experiments

Verification Points Table

From the **Response Surface** tab, use the **View** list box to select all available views of the selected response-surface-setup. All verification points appear in the **Verification Points** table view:



Use verification points to verify that the response surface accurately approximates the output parameter values. They compare the predicted and observed values of the output parameters.

You can add, delete and modify verification points manually.

- Same as add/delete/modify refinement points.
- Insert from **Response Points** table view.

Click **Add to Refine** to insert the selected response point to the refinement table.

A design point update (that is, a "real solve") calculates each verification point. These verification point results are then compared with the response surface predictions and the difference is calculated.

Verification points are useful in validating any type of response surface. In particular, however, you should always use verification points to validate the accuracy of interpolated response surfaces, such as Kriging or Sparse Grid.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

You can add, delete, save, or export verification points with the command buttons, or you can modify verification points manually by modifying point values directly in the grid.

Related Topics

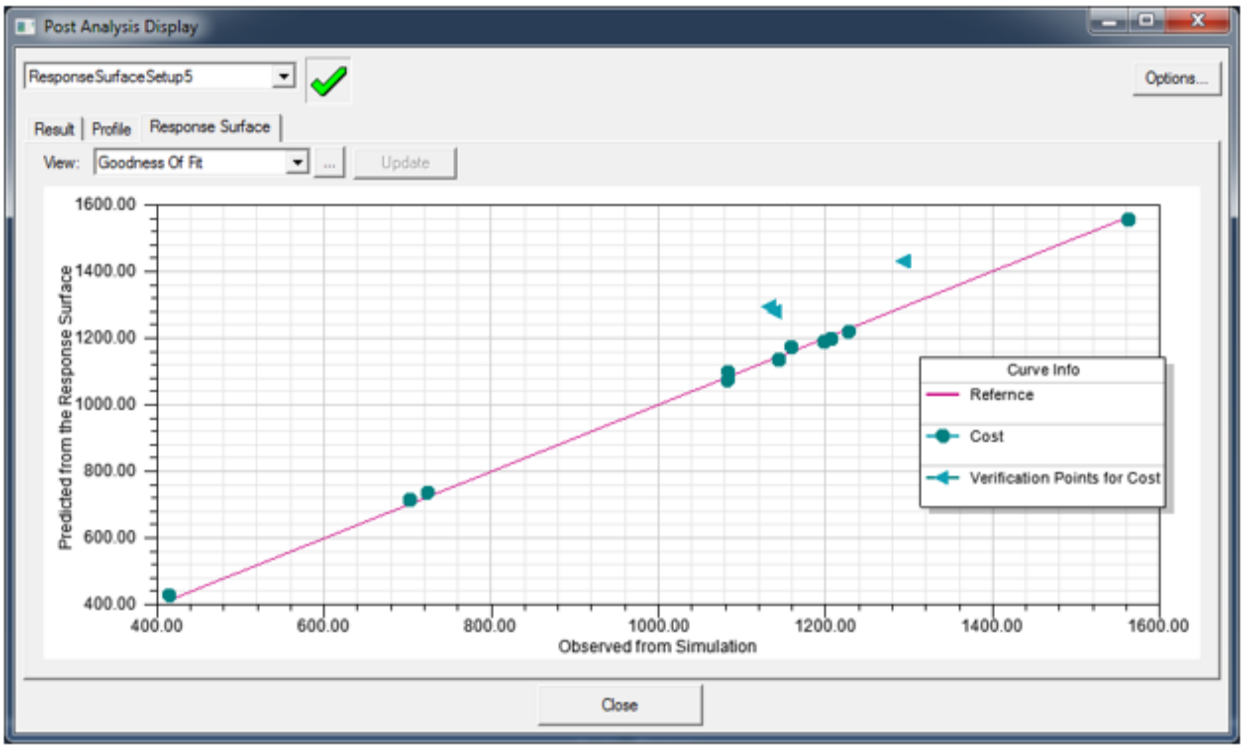
Setting Up Design of Experiments

Using Design of Experiments

Goodness of Fit (Predicted vs Observed Chart)

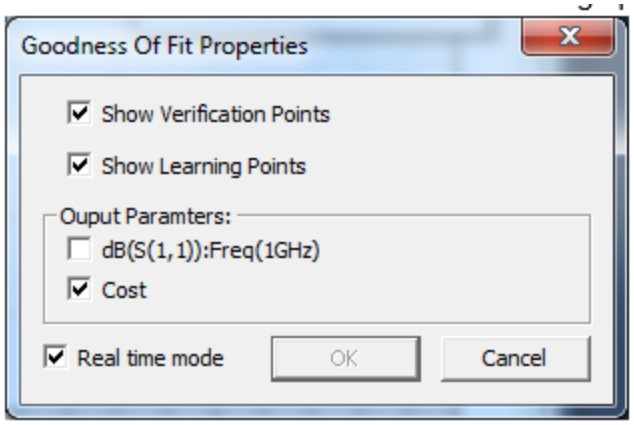
From the **Response Surface** tab, the **View** list box lets you select all available views of the selected response-surface-setup. Response surfaces are built from design points in the Design of Experiments (DOE) and refinement points (collectively, called learning points). The Goodness of Fit calculations compare the response surface outputs with the DOE results used to create them.

The closer the points are to the diagonal line, the better the response surface fits the points.



You can view Goodness of Fit information for any of the output parameters in a response surface. To do so click ... to bring up the **Goodness of Fit Properties** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Related Topics

Setting Up Design of Experiments

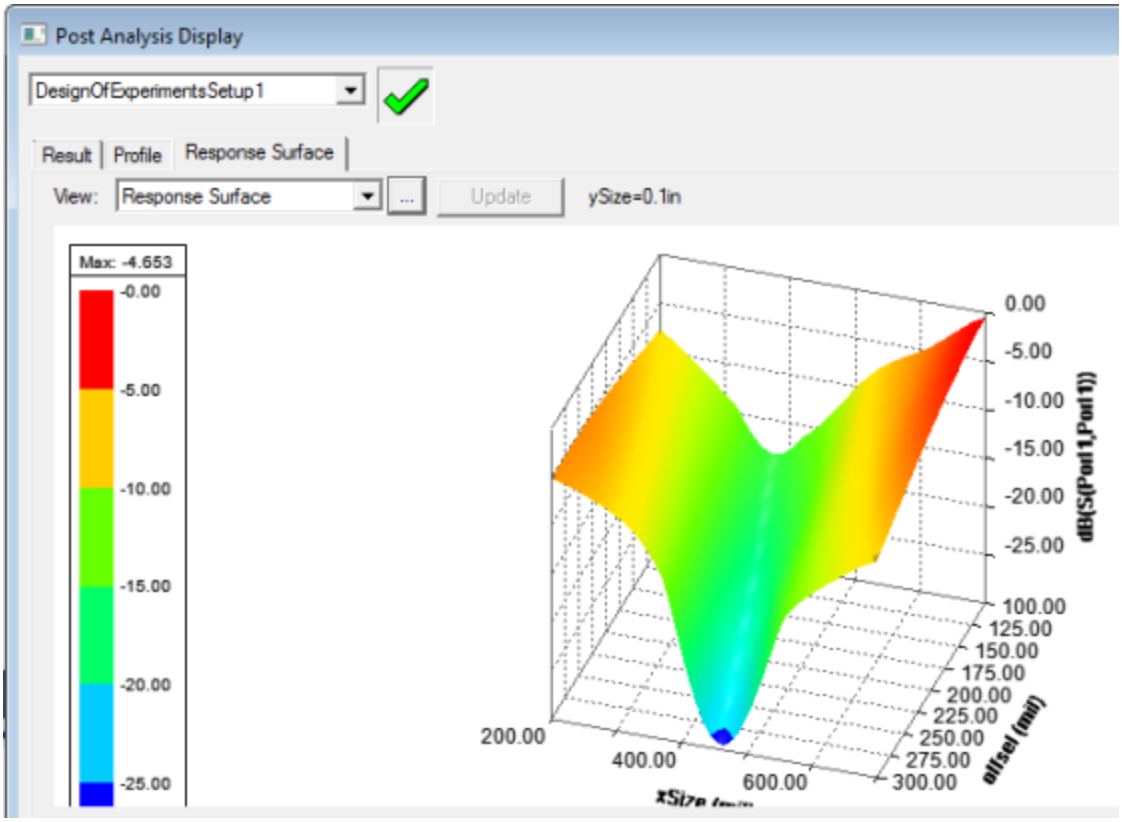
Using Design of Experiments

Response Surface Results Design of Experiments Result

When more than one variable is included in the setup, the **Response Surface** view is available. You can choose any two variables as the X- or Y-axis, and choose an output calculation as the Z-axis; click ... and browse for the variables. From the **Response Surface** tab of the Design of Experiments **Post Analysis Display** dialog box, the **View** list box lets you select all available views of the selected response-surface-setup.

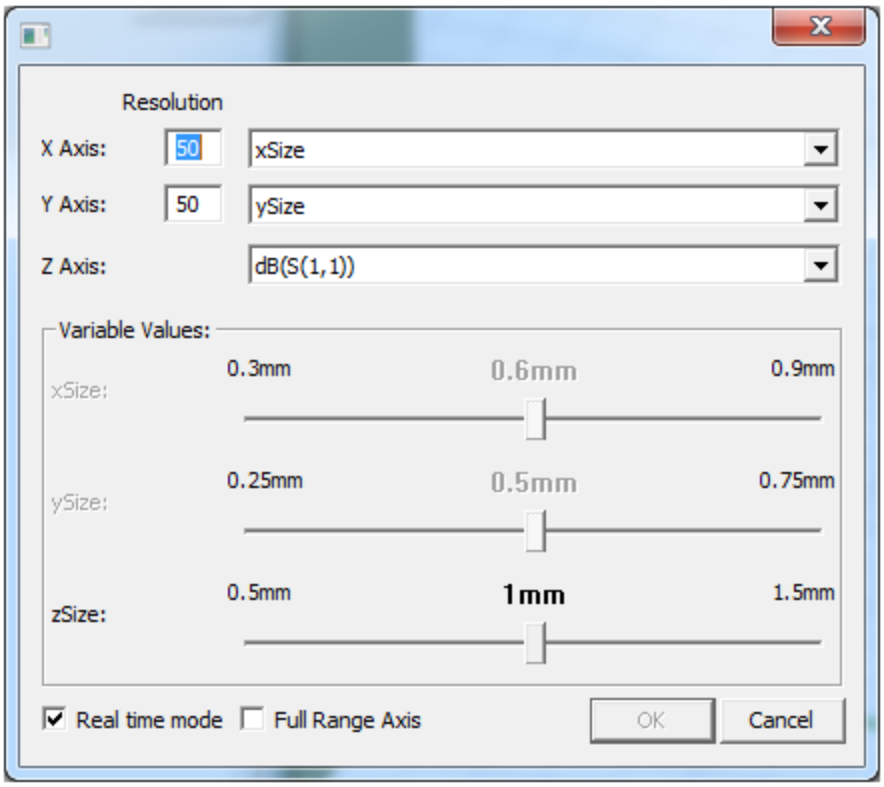
Response surfaces are functions of varying natures in which the output parameters are described in terms of the input parameters. Built from the Design of Experiments (DOE), they quickly provide the approximate values of the output parameters throughout the design space without having to perform a complete solution. The accuracy of a response surface depends on several factors: the complexity of the variations of the solution, the number of points in the original DOE, and the response surface type. Once a response surface is generated, you can create and manage response points and charts. These postprocessing tools help you to understand how each output parameter is driven by input parameters and how you can modify your design to improve its performance.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Click ... to adjust the variables selected and the values applied. A separate dialog box, pictured below, appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Tuning a Response Surface

For the X-axis and Y-axis, you can specify a resolution, and the variable to use. For the Z-axis you can select the cost or calculation. For variables not selected for the X- and Y-axis, a slider is enabled that lets you adjust the value to see the effect on the response surface plot. You can enable or disable **Real time mode** with the check box at the lower left.

Full Range Axis check box

When you select this check box, all axes are set to their maximum ranges, and the ranges won't be changed while tuning unless you change the axis variable.

When you clear this check box, the Y-axis (2D) and Z-axis (3D) axis range updates to fit the curve/surface.

Accumulate Response Curve

The check box is cleared when the tuning dialog box opens, and it won't restore its last state. When selected, it will retain the existing curves, and add new curves to the plot. Clearing the check box won't clear the accumulated curves, but new curves will no longer be accumulated. When the axis variable changes, all accumulated curves are cleared.

Exporting Response Surface Data

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

You can export the response surface data as a table in the following formats:

- Comma delimited data (CSV) files.
- Tab delimited data (TAB) files.
- Ansoft Plot data (DAT) files.
- Post Processor format data (TXT) files.

You can import the exported files into a report.

Related Topics

Using Design of Experiments

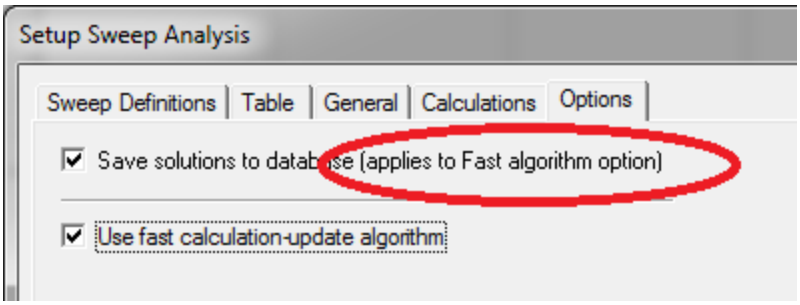
Setting Up Design of Experiments

Using the Fast Calculation-Update Algorithm

A fast calculation-update algorithm is available to speed up Optimetrics and report updates during Optimetrics analyses. The fast calculation-update algorithm generates the same Optimetrics results, only faster, and is available for all Optimetrics analyses *except* Tuning analysis. You can enable the fast calculation-update option using the **Options** tab of the Optimetrics **Setup** dialog box. To enable the fast calculation-update algorithm:

1. Select **Twin Builder** > **Optimetrics Analysis** > **Add** <*OptimetricsType*>.

   A typical **SetupSweep Analysis** dialog box is shown below with the **Options** tab selected.



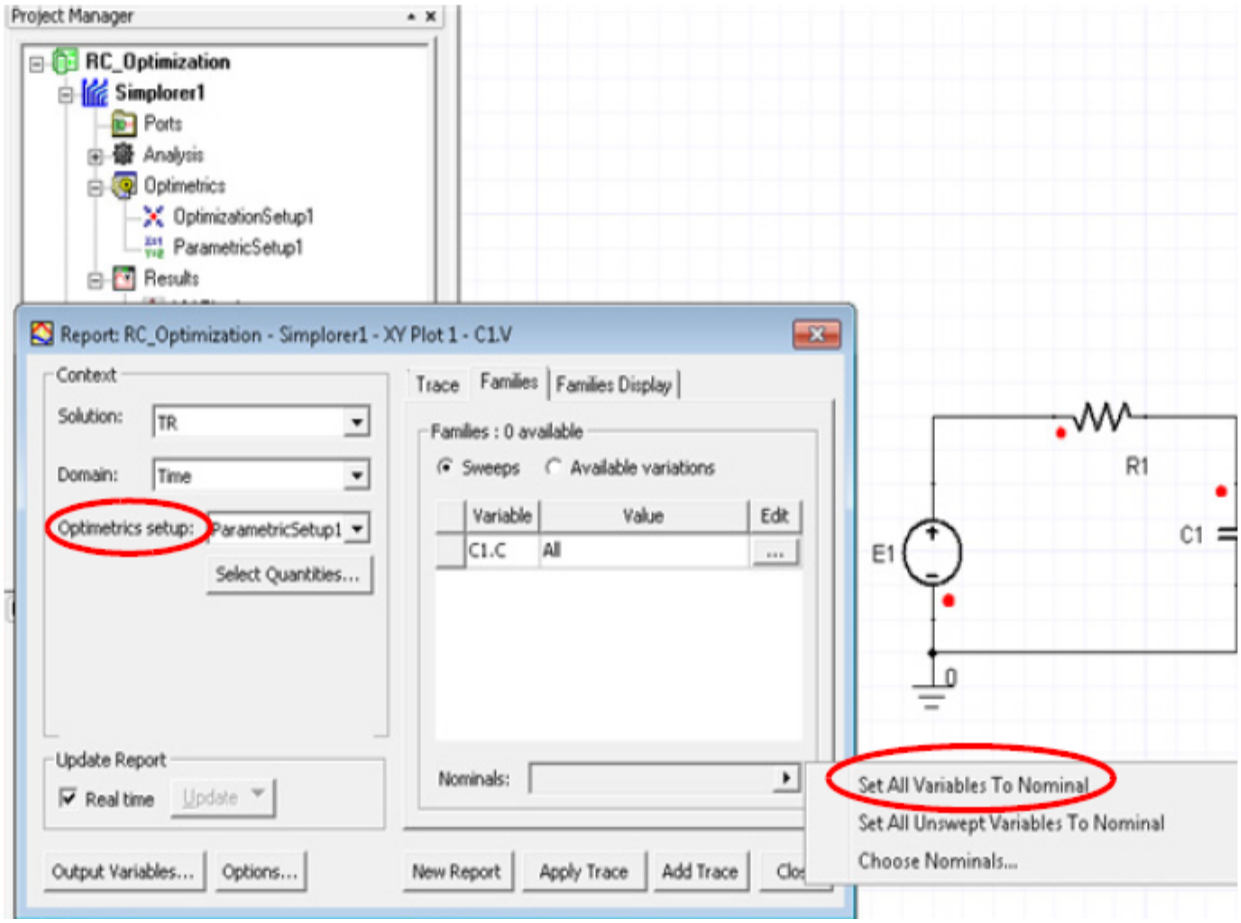
   The **Options** tab is the same for all optimization setups.

2. Select the **Use fast calculation-update algorithm** check box to enable the algorithm. (See also *Fast Calculation-Update Algorithm Limitations* below.)

   When the fast calculation-update algorithm is enabled:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- You can enable some reports to be updated during the optimetrics analysis if you **Set All Variables To Nominal** in the Report/Trace setup dialog box:

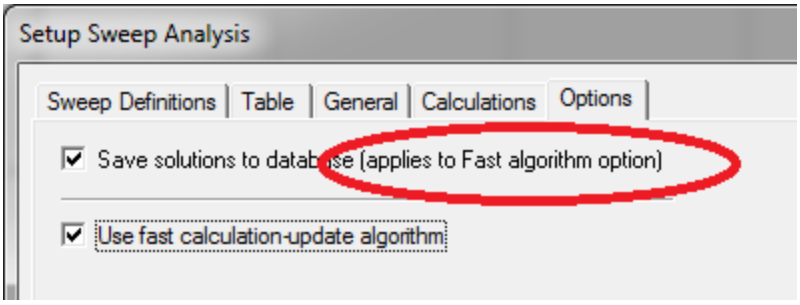


- You can see each trace (overwriting the previous) by setting the **Optimetrics setup** in the **Report** dialog box to **None** in addition to having all variables set to Nominal. At the end of the analysis, you will see the last calculated value.

3. If you have enabled **Use fast calculation-update algorithm** and want to save the solution data for every solved design variation in the Optimetrics analysis, select **Save solutions to database** as shown below.

> **Note:**
>
> Do not select this option when requesting a large number of iterations as the data generated will be very large and the system may become slow due to the large I/O requirements.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- When the **Save solutions to database** option is selected, a plot with traces based on the Optimetrics Setup just run can be updated through a menu command (right-click the desired report in the **Project Manager Results** folder and select **Update Report**) and will show results as appropriate (family, if chosen).
- Without the **Save solutions to database** option selected, you can examine analysis data in the **Post Analysis Display** dialog box. Right-click the Optimetrics Setup and choose **View Analysis Result**.

Fast Calculation-Update Algorithm Limitations

The fast calculation-update algorithm **cannot** be used if **any** optimetrics calculation uses:

- **Project/Design variables** – If the project/design variable is ***not*** swept in the Optimetrics analysis, and you would like to use it in the expression, you can create an output variable for the Project/Design variable, assign the value of the Project/Design variable to the output variable, and use the output variable in the expression instead.
- **More than one range function** – For example, when range function is not the outermost function or when range function takes arguments.
- **More than one calculation range** – Also applies if the calculation range is not for primary sweep.

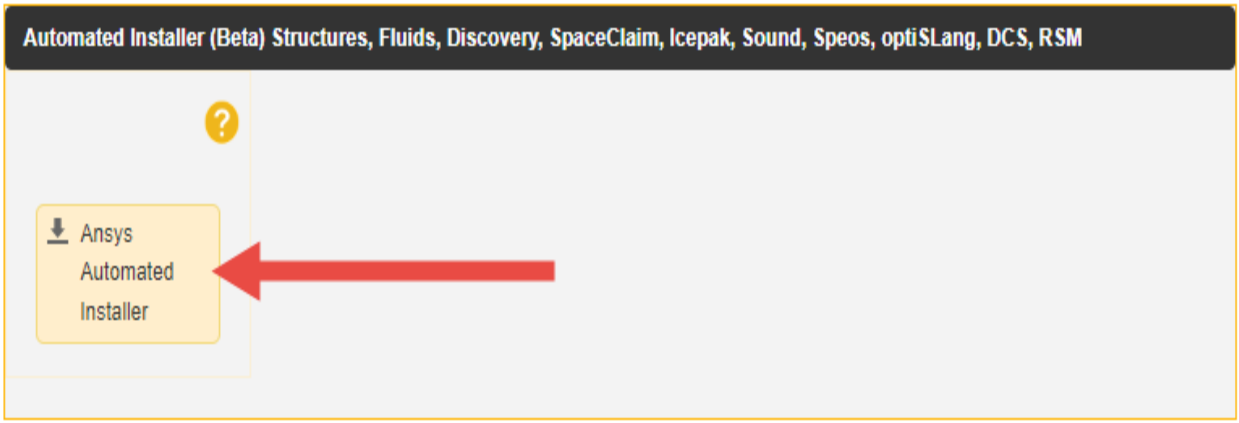
Similarly, when the fast calculation-update algorithm is enabled, a trace cannot be updated during an analysis if the trace expression uses:

- **Project/Design variables**. However, you can use the same workaround described above.
- **Any range function.**

optiSLang Integration with Electronics Desktop

optiSLang is a tool for graphical programming, process integration, and automation that can be integrated with Ansys Electronics Desktop (AEDT) for Optimetrics analysis. It is available as part of the Ansys Automated Installer package on the Ansys Customer Portal.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Comprehensive online help for optiSLang is available from the optiSLang Help menu, or at:

https://ansyshelp.ansys.com/account/secured?returnurl=/Views/Secured/corp/v242/en/opti_ug/opti_ug_intro.html

This section covers only its integration with Ansys Electronics Desktop.

optiSLang in AEDT allows AEDT and optiSLang to stay in continuous interaction by exchanging API commands between each other. optiSLang algorithms can be executed without exiting AEDT because optiSLang setups follow Optimetrics norms and have an explicit association with a specific analysis setup. AEDT stays continuously open while optiSLang algorithms demand set after set of designs.

Each optiSLang setup resides in one specific design of an AEDT project and is linked to one specific analysis setup. When an optiSLang setup is executed within AEDT, optiSLang runs in the background in batch mode and executes the algorithm that the optiSLang setup represents. optiSLang administers an optiSLang project file (*.opf) and a normal optiSLang project directory (*.opd), stored relative to the AEDT project.

The **standalone optiSLang GUI** allows for further editing of the optiSLang project structure, with each analysis setup represented as a single Ansys EDT node:

If the original Ansys EDT node and its copies are all kept in the optiSLang Setup run mode, they will remain linked to the specific optiSLang setup existing in the referenced AEDT project. The optiSLang project can contain and execute multiple algorithm systems, which means that the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

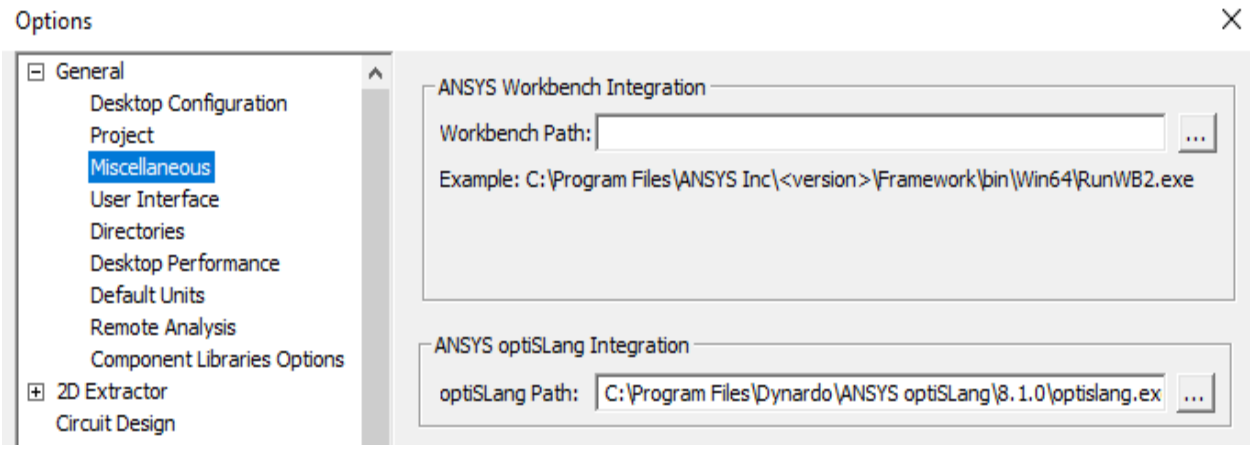
linked optiSLang setup in the AEDT project represents a connector, not an algorithm, in that case.

The topics in this section cover:

- optiSLang User Workflow
- Creating an optiSLang Setup in AEDT
- Solving an optiSLang Setup in AEDT
- optiSLang Menu Options
- Viewing optiSLang Postprocessing Results

Prerequisites

Before working with optiSLang in AEDT, specify the path to the optiSLang installation using **Tools** > **General Options** > **General** > **Miscellaneous**.



Additionally, ensure that the following are true:

- The project is solved
- Parameters exist (See: Parametrization for optiSLang Integration)
- Results reports exist (See: Results and Reports for optiSLang Integration)

optiSLang User Workflow

This topic explores the general workflow for creating an optiSLang project from Ansys Electronics Desktop (AEDT).

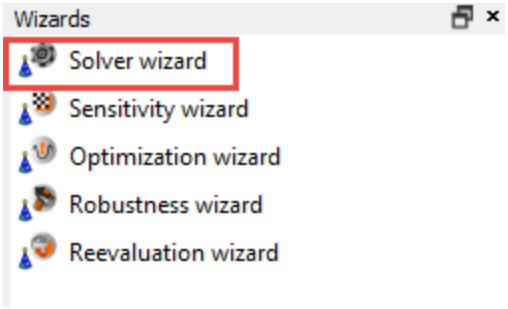
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Prepare the AEDT project by ensuring:
   - Parameters exist.

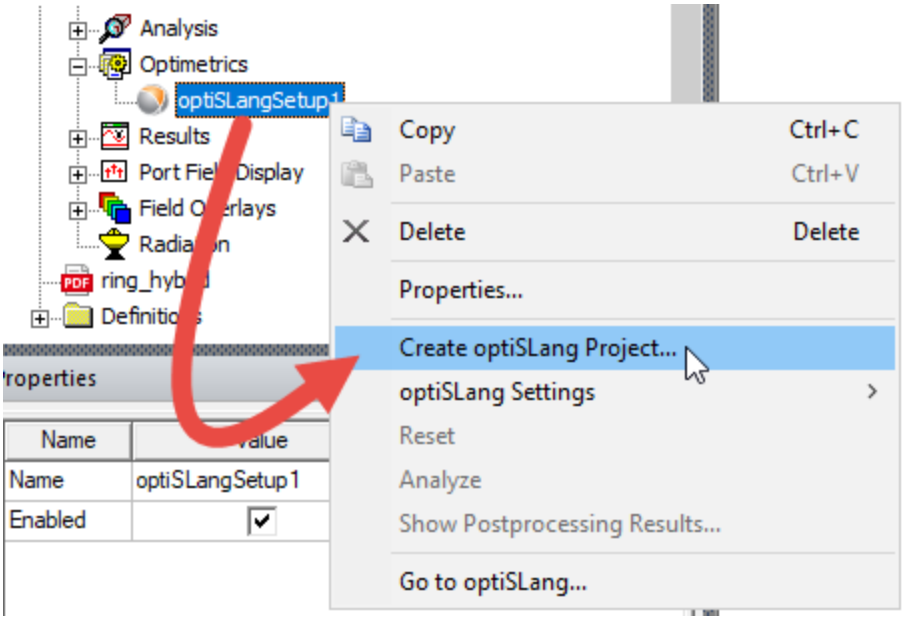
     > **Note:**
     >
     > optiSLang scans parameters at both the project and design level. Hidden parameters are also scanned. See: Parametrization for optiSLang Integration.

   - Project is solved.
   - Results reports exist and are prepared for the specific use case
     - For a standard use case, set all variables to nominal.
     - For a use case with a user-defined sweep, display all variations.

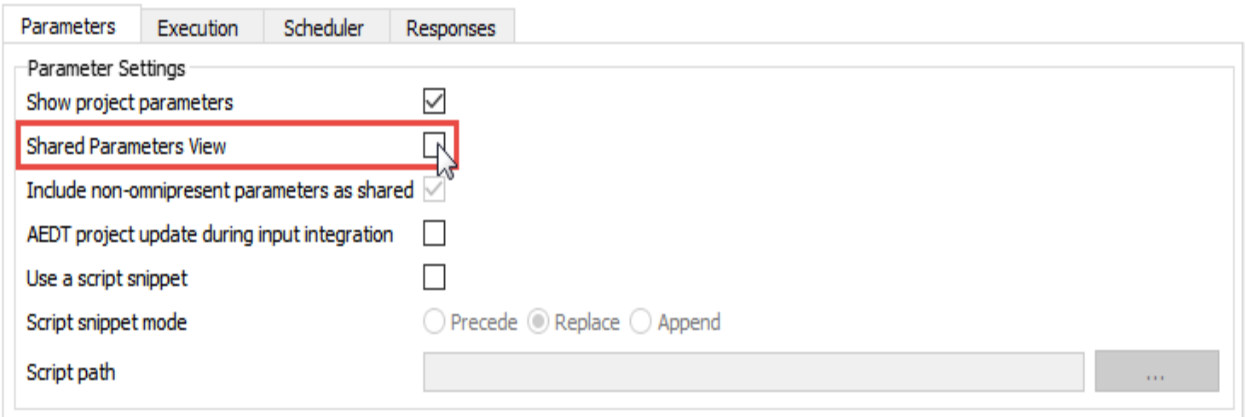
     See: Results and Reports for optiSLang Integration.
2. Invoke the Solver Wizard in optiSLang or create an optiSLang project directly from AEDT.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



3. Create a one-node setup or a two-node setup.
   - A one-node setup registers both parameters and responses. This is the type of setup created by the wizard.
   - A two-node setup registers parameters in the first node and responses in the second. This is a more complicated setup that requires cloning the integration node, setting function switches, and setting absolute or relative paths. Consult the optiSLang help for more information.
4. View results in either tabular or tree view.
5. Enable **Shared Parameters View**. By default, parameters from different models are treated separately and a prefix based on the model name ensure uniqueness of names. In shared mode, sub-models are kept in sync andprefixes are unnecessary.



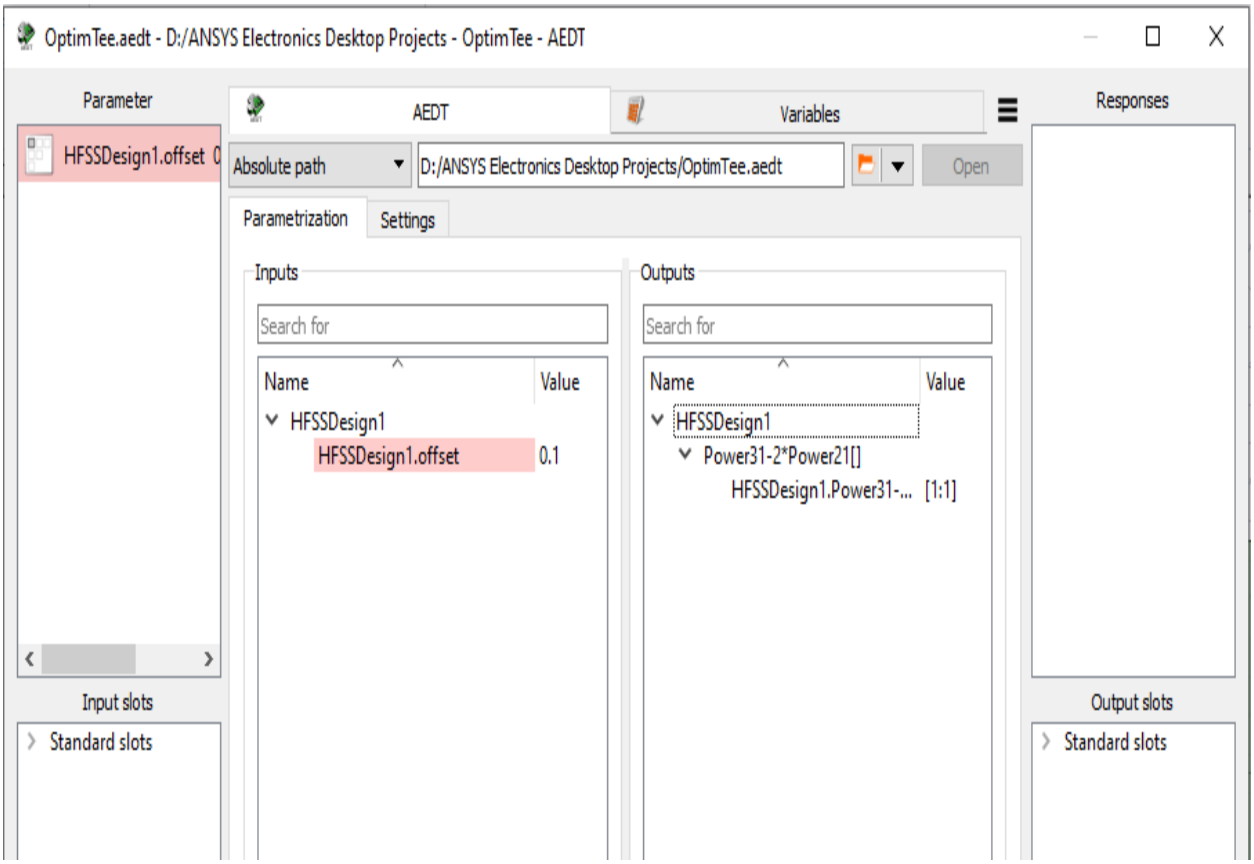
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. Reload the AEDT project in optiSLang to see the effects of changed settings.

Parametrization for optiSLang Integration

Certain AEDT data must exist before optiSLang can use it.

Variables act as **Inputs** in optiSLang, while simulation results (reports) act as **Outputs**.



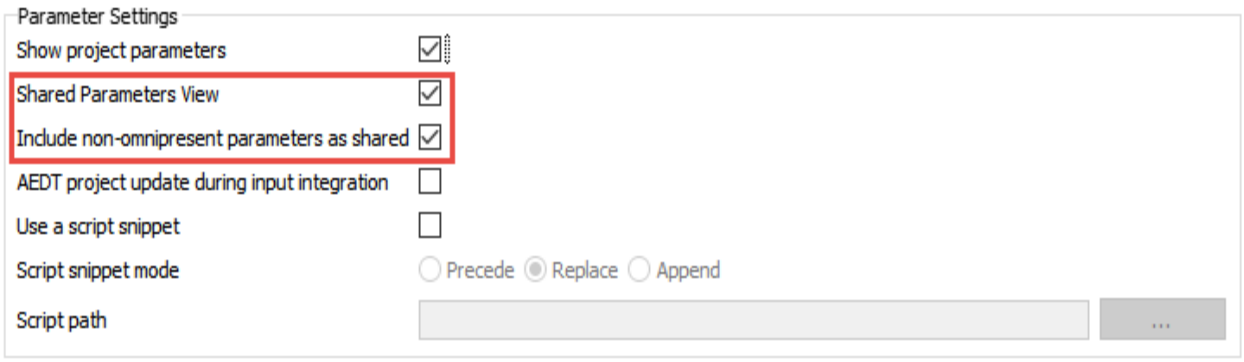
Inputs

When integrated with AEDT, optiSLang automatically detects project variables and design variables for every design in the project. Note that:

- Dependent parameters are skipped.
- Any parameter marked as "hidden" is always included.

Users decide whether to treat parameters from different models in the project as independent or communal (kept synchronized). In optiSLang, the AEDT integration has a **Shared Parameters View** with an additional **Include non-omnipresent parameters as shared** option.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



In AEDT, project variables can be used to steer model parameters.

It is best to leverage features on both sides of the optiSLang-AEDT integration for managing overlapping (that is, partially synced) parameter sets. optiSLang's standard parametrization management tools offer the parameter type Dependent for aligning or otherwise controlling parameters.

In the context of bundling optiSLang-demanded designs for simultaneous execution as DSO jobs, remember that DSO design variation tables are hosted in the Optimetrics section of each design, and that there is no Optimetrics branch at the project level. So, even varying only project variables will still yield separate DSO tables and thus separate jobs for each model in the project.

A project walker mechanism inquires a given AEDT project and stores the generated project structure info in a JSON file for optiSLang. If the same reference project is linked for use in a different AEDT node even from a different optiSLang project, the stored project inquiry results can be recalled and the parametrization can be instantly displayed. This mechanism spares users wait time for the project inquiry call.

Outputs

The framework for result data transfer from AEDT to optiSLang is very simple: each report trace shows up in optiSLang response listings as a signal object. You can register the signals directly or apply math functions offered by optiSLang's calculator with each integration node. The transfer vessels are CSV files.

Even such plots as Smith charts or 3D polar plots can be collected as groups of signal objects using the uniform CSV file export function offered in AEDT.

Parameterization with optiSLang Analysis Goals in Mind

The parametrization scheme should connect to the goals of the optiSLang analysis.

Bad parametrization might hide the design optimum from the best optimizer or destroy the connection of a robustness sampling with reality. Thoughtfully improved parametrization can

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

achieve simulation goals with a fraction of the designs that would be needed for a task within the most simple ad hoc parametrization scheme.

The algorithms being applied in a specific scenario should also guide your parametrization scheme. For example, it makes sense to give a robustness sampling algorithm the chance to examine a parameter space where the simulation can represent the relevant physical effects happening in the real world. An optimizer algorithm, meanwhile, needs the chance to hit feasible designs frequently.

Some tips:

- Think ahead to devise a stable and meaningful parametrization
- Be attentive to the kinds of errors encountered by failing designs
- Be quick to eliminate the origins of errors that appear most frequently in error logs
- Learn to use optiSLang post-processing as a diagnostic toolbox
- Apply feedback and lessons learned to your parametrization schemes

Results and Reports for optiSLang Integration

AEDT can export simulation reports as CSV files that can be read in optiSLang.

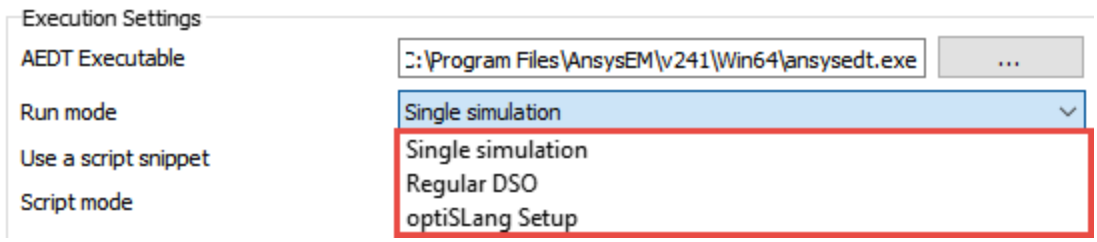
Additionally, all Optimetrics setups in AEDT (incuding optiSLang setups) have a **Calculations** tab, allowing for additional results. See: Setting up Calculations for Optimetrics.

When the scripted AEDT integration machinery within optiSLang exports reports, it is just as if a user had exported a CSV file from within the AEDT UI. Any reports that can be exported as CSV (rectangular plots, data tables, polar plots, and so on) are sent in CSV format, and the reports' traces can be registered as responses in optiSLang. As a general rule, each data column of a CSV file is exposed as a signal trace.

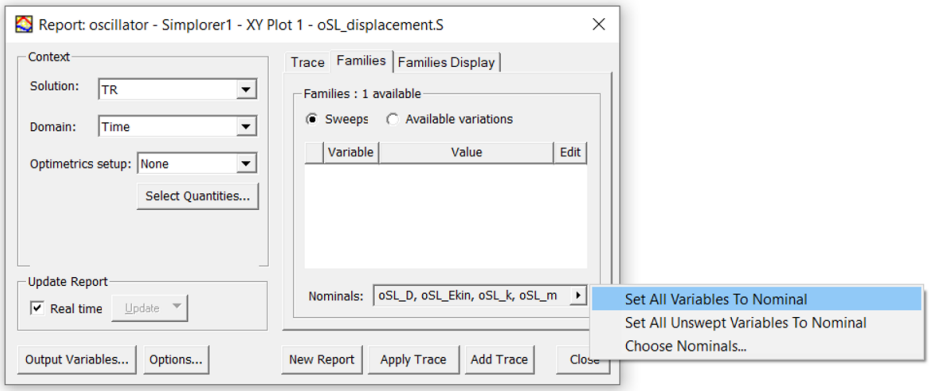
> **Note:**
>
> For 3D Polar Plots, this can cause a small disadvantage by yielding, for example, 180 signal objects instead of a matrix. However, the simplicity of the routine achieves good performance and robustness, and allows the AEDT integration node in optiSLang to cover a large array of use cases.

Below, find considerations for setting up reports for use in optiSLang's different run modes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



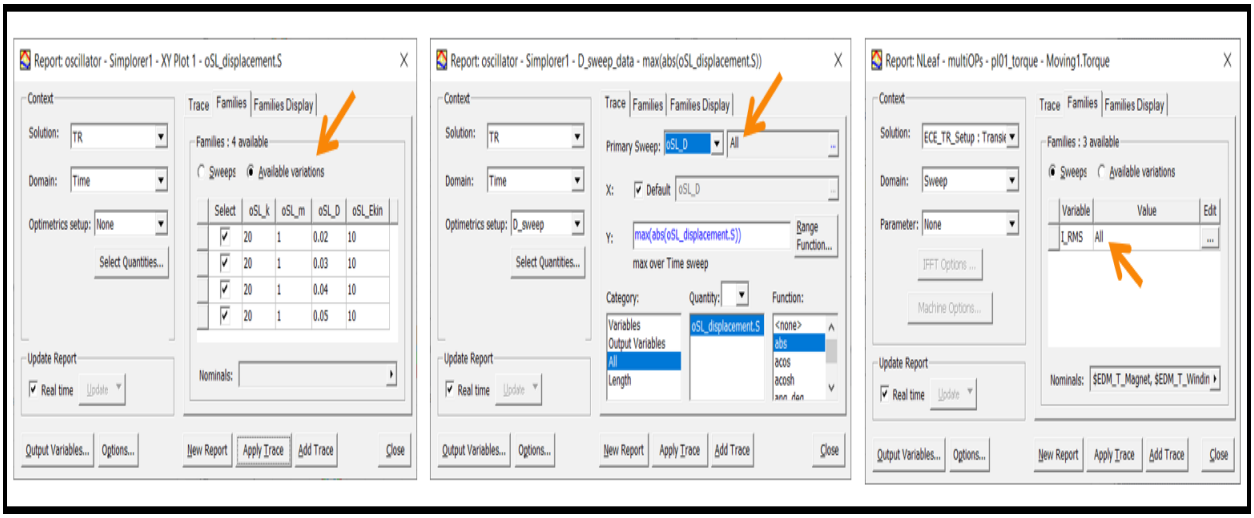
In optiSLang's **Single Simulation** run mode, optiSLang modifies the parameters of the nominal design and triggers the solution process. Therefore, reports used in this way should be configured to display the nominal design if they are to contain traces after modification and solution update. To ensure all report parameters are set to nominal, open the report settings and select the **Families** tab. From the **Nominals** field, select **Set All Variables to Nominal**.



In optiSLang's **Regular DSO** mode, the result export script cycles over all design variations of the Optimetrics setup created by previous script commands for holding the set of design variations demanded by optiSLang. In the loop, it applies the next design to be displayed as the nominal design, forces a refresh of all reports, and exports all reports. This loop is repeated for all design variations and the CSV files are stored each time in the corresponding design folder. Therefore, reports used in this mode should also be configured to display the nominal design. To ensure all report parameters are set to nominal, open the report settings and select the **Families** tab. From the **Nominals** field, select **Set All Variables to Nominal**.

In optiSLang's **optiSLang Setup** mode, the software solves a pre-existing Optimetrics setup (either serially or as a DSO job). This is commonly used to represent a parametric sweep or a prescribed set of operating points. In these use cases, reports should contain all data from all design variations to support a single export report call. To ensure the reports are adequate, open the report settings and enable all available variations:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

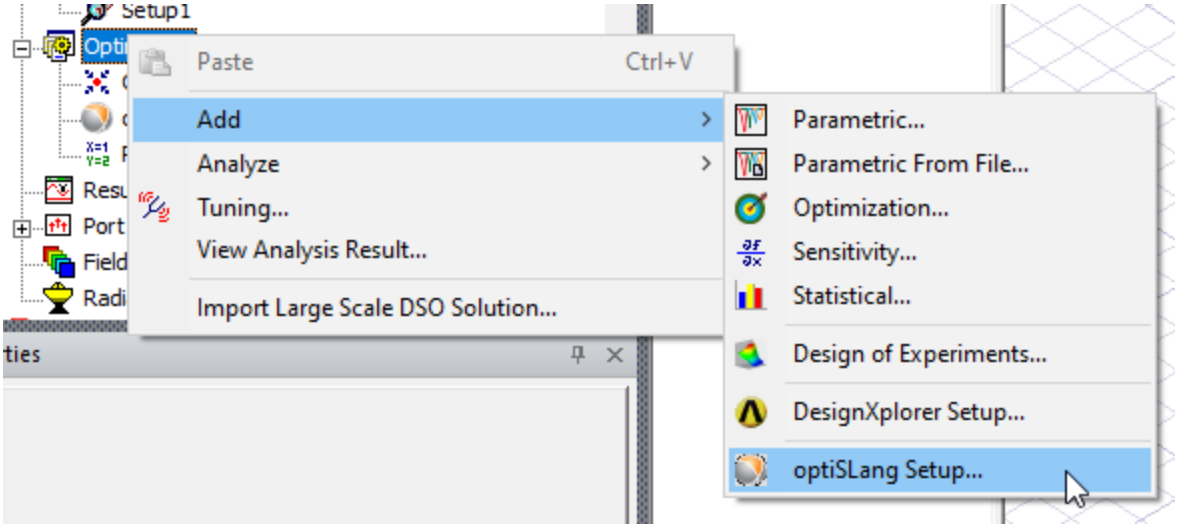


Creating an optiSLang Setup in AEDT

optiSLang setups are defined similarly to Optimetrics setups.

Create an optiSLang setup one of two ways:

- From the **Project Manager**, right-click **Optimetrics** and select **Add** > **optiSLang Setup**.
- Click (missing or bad snippet) > **Optimetrics Analysis** > **Add optiSLang Setup**.

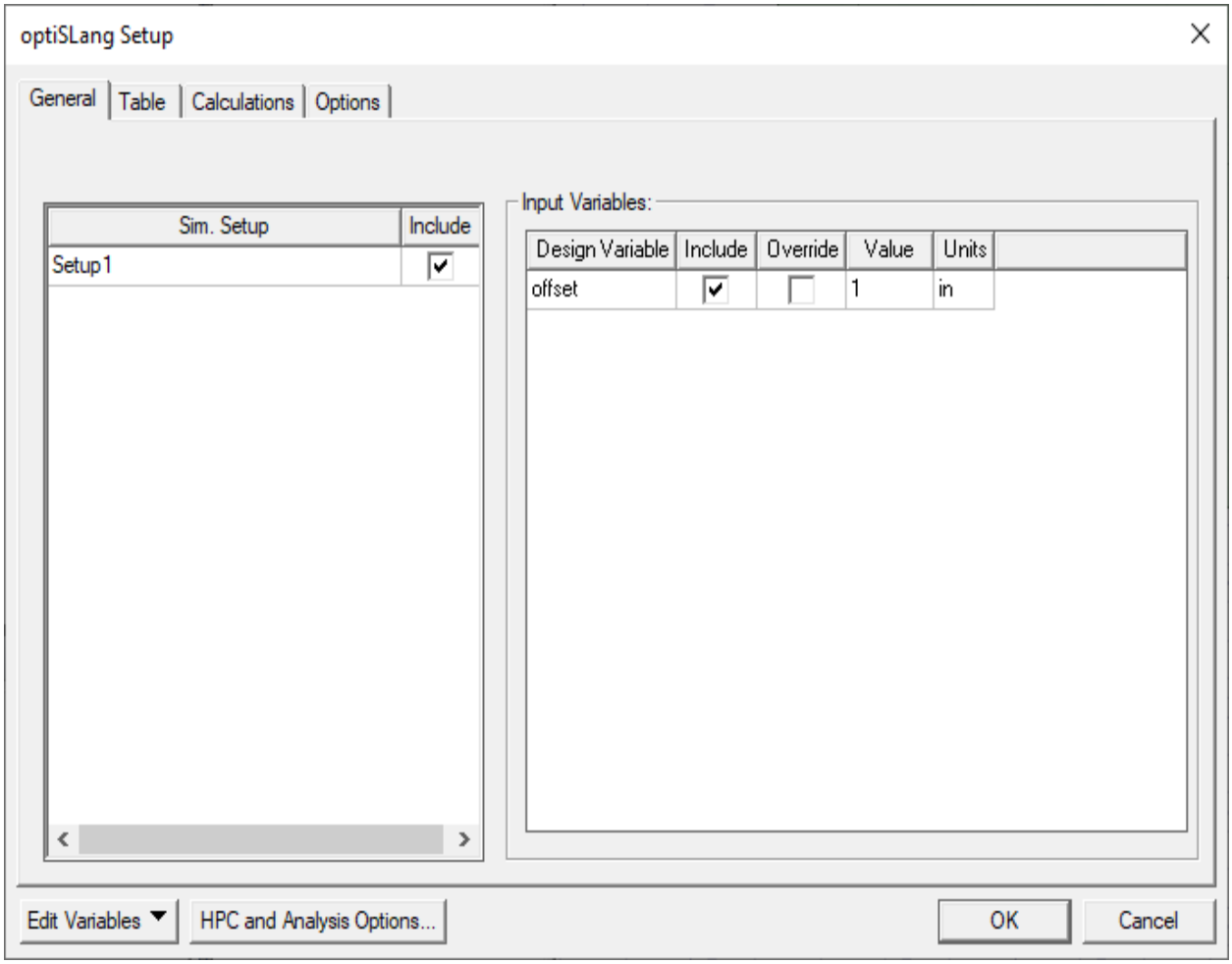


The **optiSLang Setup** window appears, with content based on the current design.

This window contains several tabs:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **General** – lists solution setups and design variables associated with the current design, and allows you to specify which setups and variables to include, as well as to override a variable's value.



- **Table** – lists the values of the any variable(s) selected for inclusion.
- **Calculations** – allows for the setup of optiSLang calculations, similar to Optimetrics or Parametric setup. See: Setting up Calculations below.
- **Options** – lists mesh options. See: Setting Options below.

Each tab of the window also provides access to Edit Variables and HPC and Analysis Options.

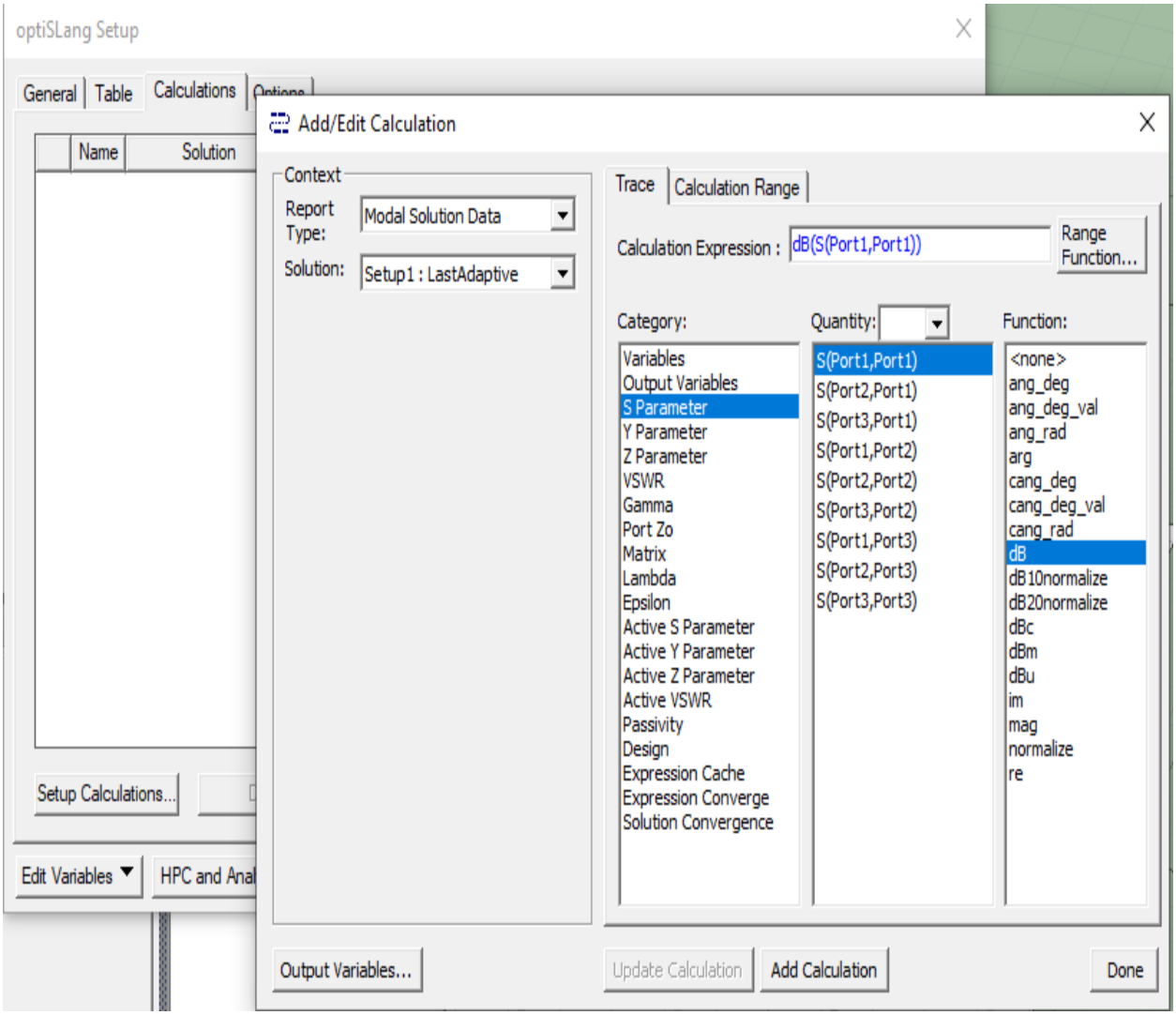
Setting up Calculations

Initially, the **Calculations** tab is empty.

To set up a calculation, click **Setup Calculations**.

The **Add/Edit Calculation** window appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



This window contains **Context**, **Category**, **Quantity**, and **Function** fields that can be used to build an optimization calculation expression, just as for other optimization or parametric calculations.

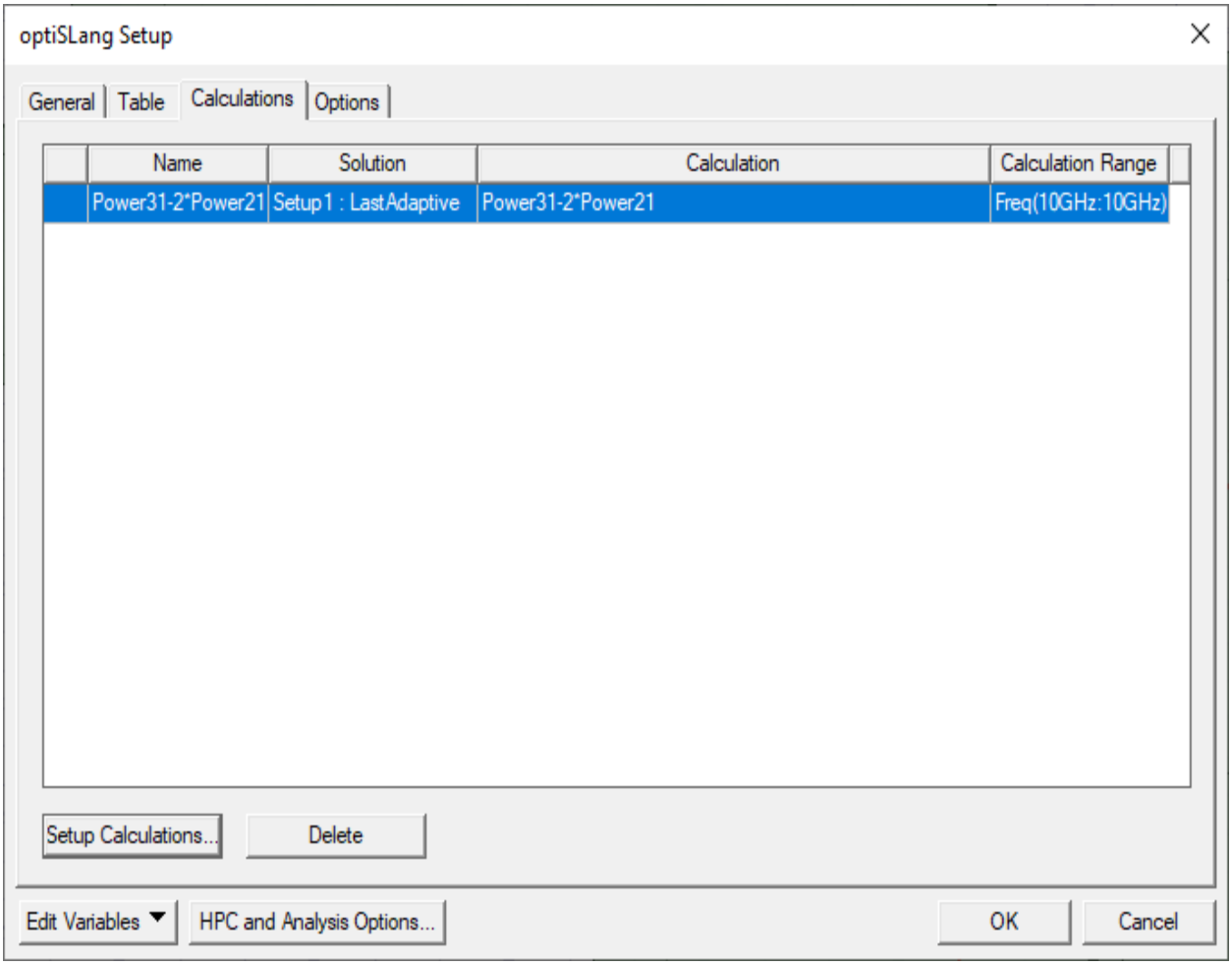
See: Setting Up Calculations for Optimetrics.

Once you have created an expression, click **Add Calculation** to add it to the **optiSLang** setup.

Click **Done** to close the **Add/Edit** window and return to the **Calculations** tab.

The new calculation is listed:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



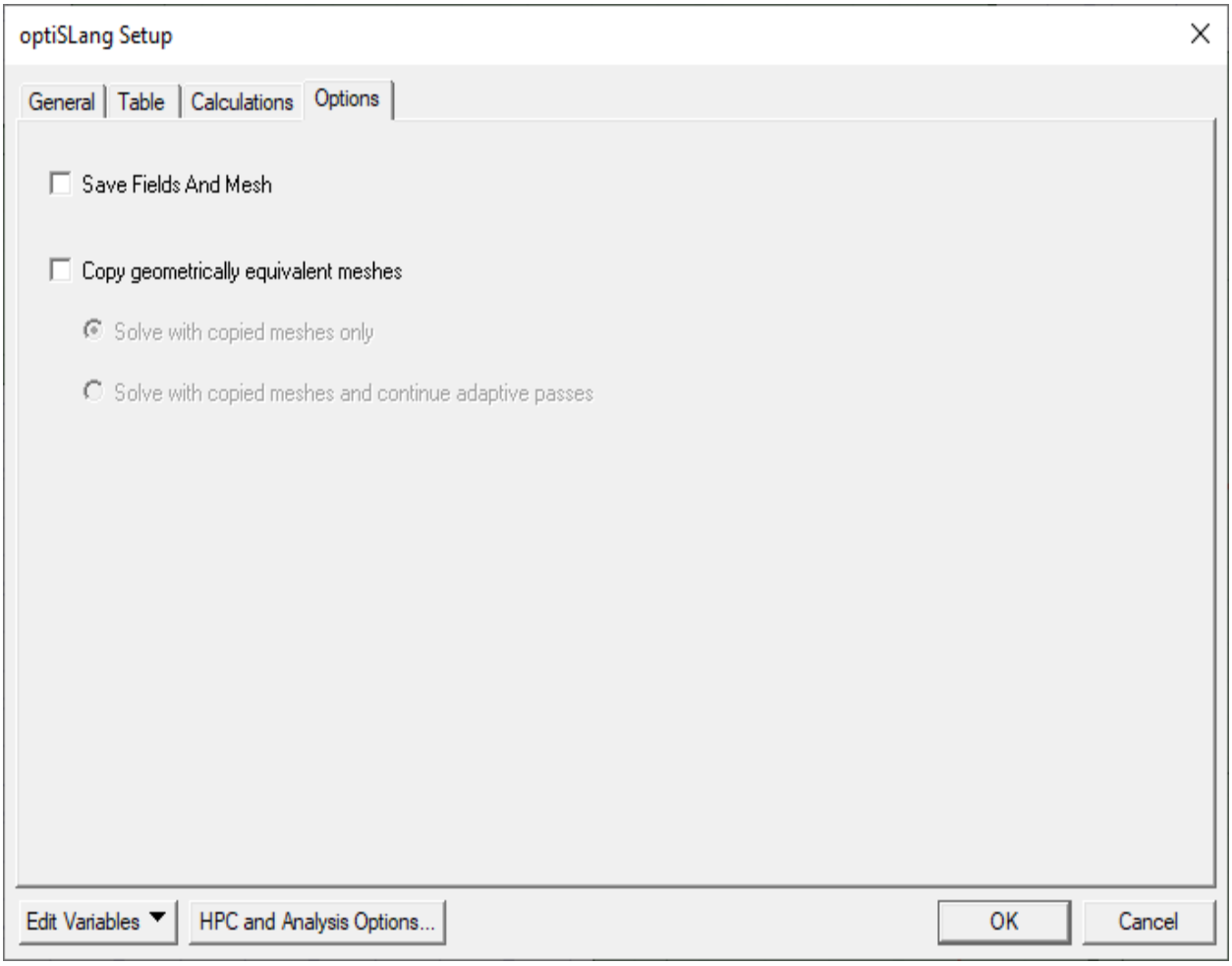
Setting Options

The **Options** tab contains options determining whether to **Save Fields and Mesh** and whether to **Copy geometrically equivalent meshes**.

In order to preserve disk space, by default AEDT does not save field solution data for every solved design variation. **Save Fields and Mesh** overrides this behavior.

Copy geometrically equivalent meshes directs AEDT to copy a mesh that was calculated for one sweep variation for reuse on a geometrically equivalent sweep variation. See: Copying Meshes in Optimetrics Sweeps.

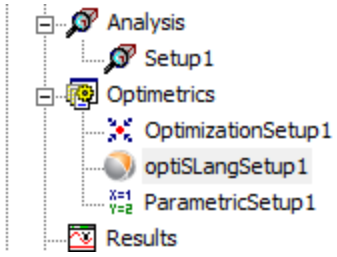
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Finishing Setup

Click **OK** when you have completed the setup.

The optiSLang setup appears in the **Project Manager**, under **Optimetrics**:



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Example optiSLang Setups

The following are example scenarios illustrating different considerations that might be taken when preparing an optiSLang setup. See: Solving an optiSLang Setup.

Settings in these examples are handled both in AEDT and in optiSLang.

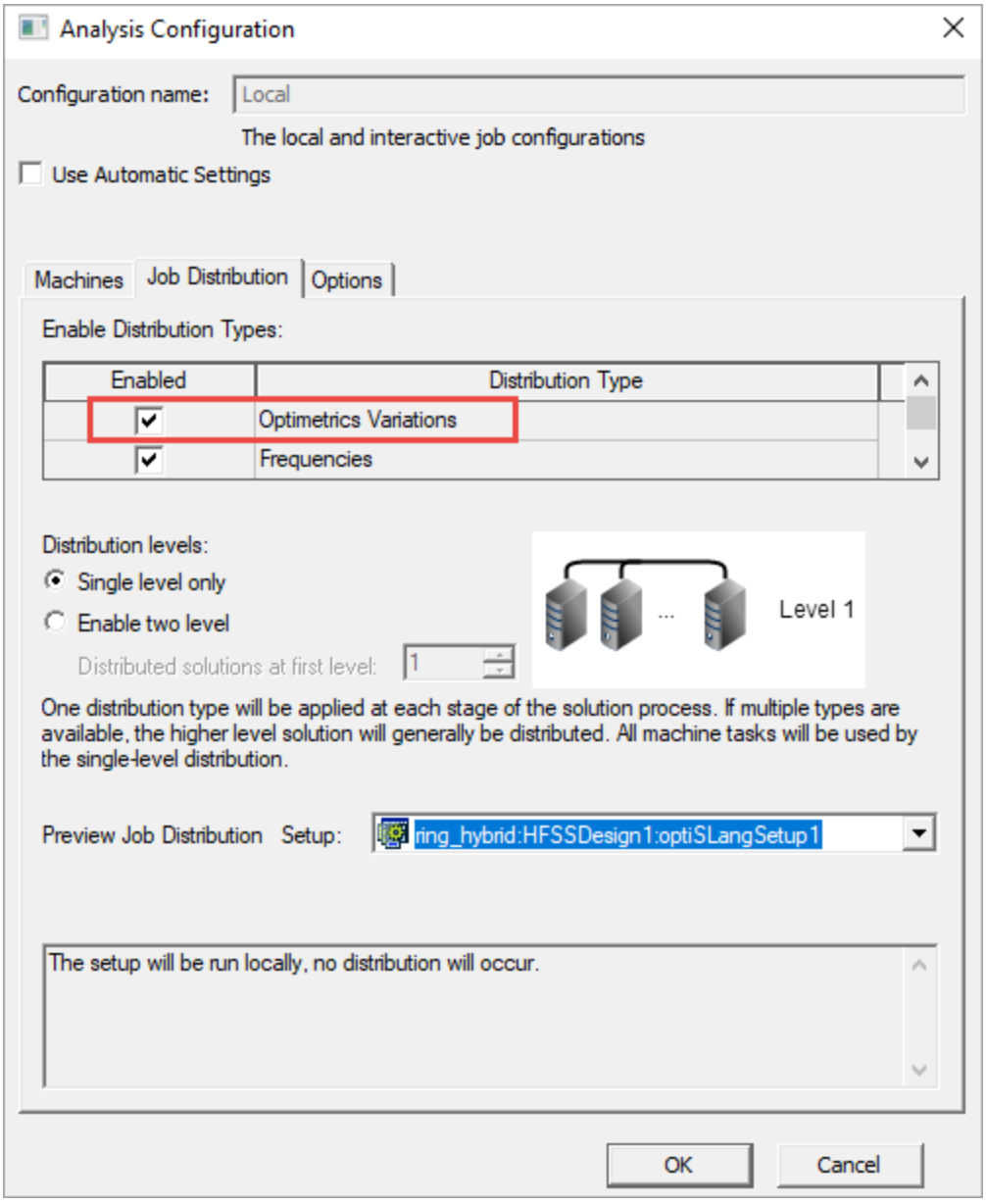
Example 1: Solving a User-created, Pre-existing optiSLang Setup

Recall that optiSLang setups are handled as Optimetrics setups in AEDT.

By preparing an optiSLang setup in an AEDT project, you can create regular DSO jobs even though the AEDT integration node is left in the single simulation run mode. In every design folder, the clone of the reference project contains the optiSLang setup and its solution is triggered under the **Analyze All** umbrella. Depending on the HPC settings active in AEDT for the model type in question, this yields optiSLang DSO jobs.

A key setting is the check box for enabling **Optimetrics Variations** in the **Job Distribution** tab of the **Analysis Configuration** window:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

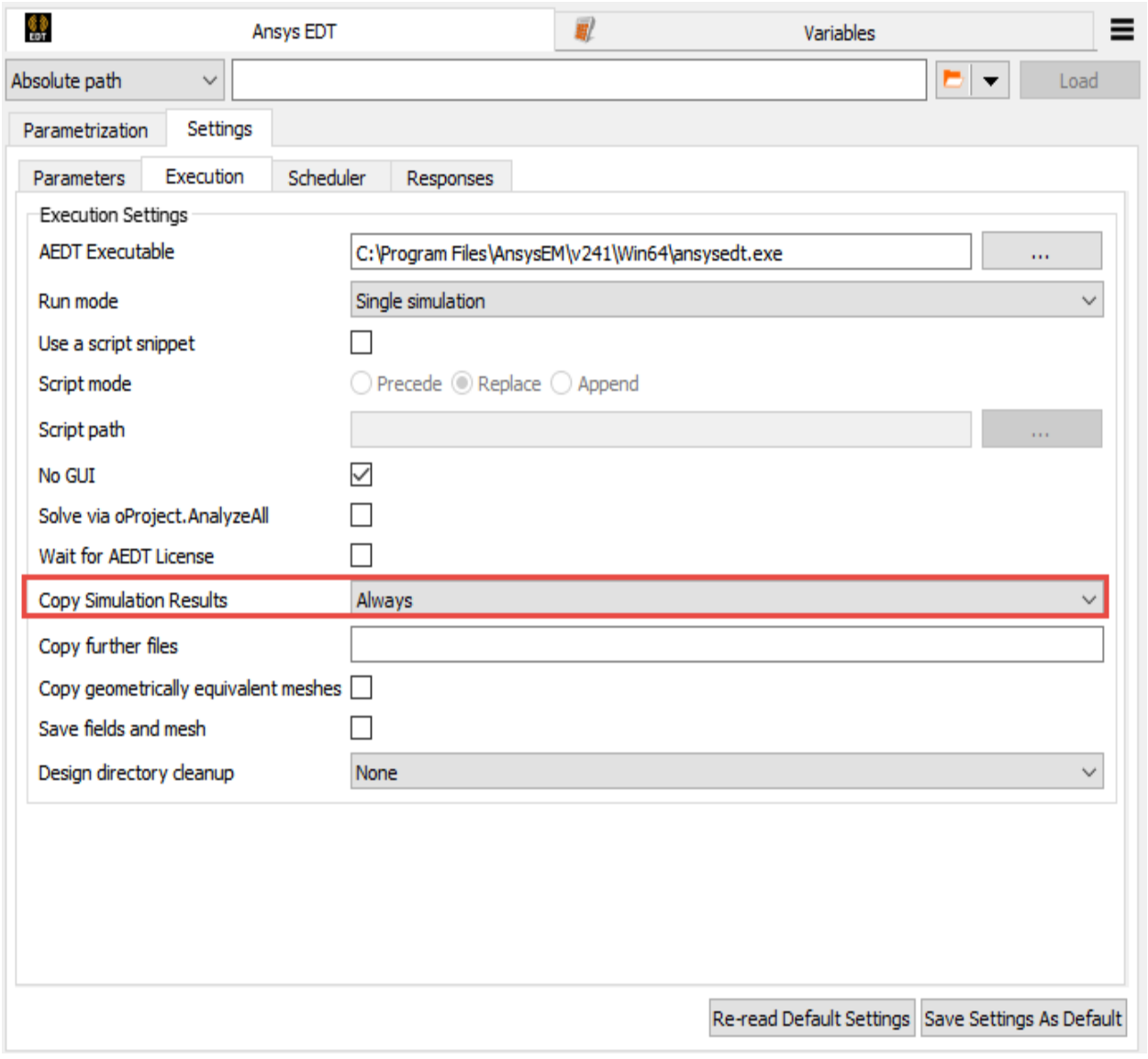
The **Use Automatic Settings** check box must be disabled to view this tab.

The working points being solved completely independently (simultanization) enables nice speed-up degrees for all designs within the optiSLang setup.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Example 2: Increasing Performance by Avoiding Wasteful Solution of the Nominal Design

In Example 1, performance drag can exist if the analysis setup of the nominal design is also solved in every optiSLang design folder in series with a desired Optimetrics setup. Keeping the optiSLang setting **Copy Simulation Results** on the value **Always** can avoid redundant solution of the nominal design at the cost of the time for copying the *.aedtresults folder of the reference design repeatedly into every design folder.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Example 3: Avoiding Redundant Computations while Exploiting AEDT Features around Postprocessing Variables

In optiSLang, an AEDT node can be used to copy a reference project along with its associated *.aedtresults directory of solution data. When this node is used for evaluating different combinations of postprocessing variables, AEDT does not call any solver and that the evaluation time per design is accordingly fast. The only drag consists in loading a fresh instance of the AEDT program for every new design.

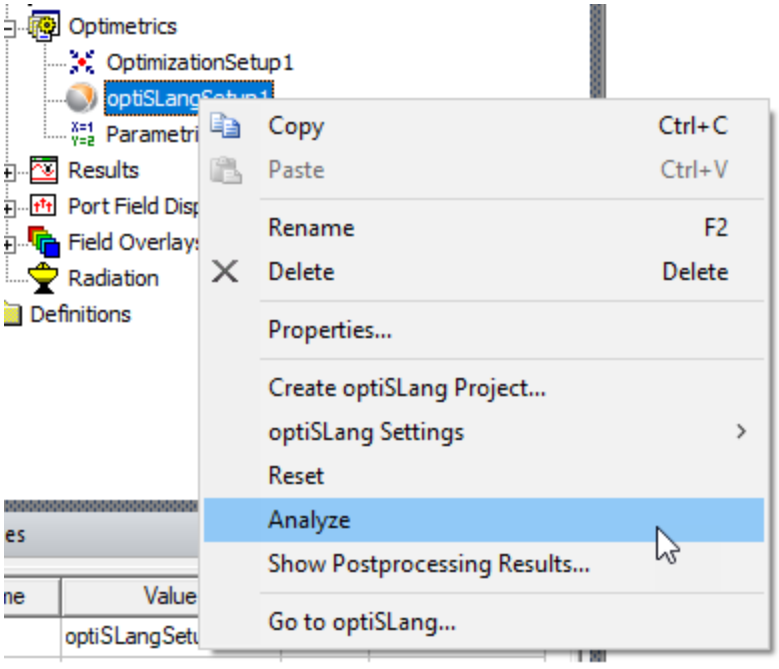
Example 4: Parametrization of an AEDT Design for optiSLang Integration

Consider a 2D Maxwell model of a permanent magnet synchronous motor. For design optimization, the magnet positions and angles can be the crucial parameters and due to symmetry a quarter model can represent the entire machine. For questions of robustness, reliability, tolerances, and achieving specs, it might be necessary to parametrize eccentricity and ellipticity. If magnet quality scatter is important, this might be another aspect requiring a full 360 degree model because it would be drastically unrealistic to assume that 16 worst-case magnets can find their way into one single machine. See: Parametrization for optiSLang Integration.

Solving an optiSLang Setup

To analyze an optiSLang setup within Electronics Desktop:

- From the **Project Manager**, right-click a setup and select **Analyze**.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Considerations for Using the Analyze All Command

You can use **Analyze All** to solve all projects in a design, including Optimetrics setups. See: Running More Than One Simulation.

> **Note:**
>
> Before using **Analyze All**, remove any unnecessary analysis setups from the project.

Some reasons for doing so:

- During regular DSO run mode, optiSLang creates Optimetics DSO design tables. By leaving just one setup for analysis, there is no ambiguity about which analysis setup is associated with any new table.
- Numerical convergence settings steer the performance trade-off between speed and accuracy. If a small number of designs take exceptionally long to converge, it can make a substantial performance difference whether designs are solved independently or in groups (DSO jobs). In group jobs, one slowly converging simulation can delay the entire group. This increases the importance of upper bounds on iterations for mesh refining, interpolation point addition, and so on.
- The unintended presence of Optimetrics setups (including optiSLang setups) can lead to long execution times. When used intentionally, however, Optimetrics setups can be used to define parametric sweeps. These sweeps are easy to set up and represent the sweep per design as a nested system in optiSLang. Note that reports must be adjusted accordingly, so that data is drawn from the entire sweep and not a single point, in order to reach the optiSLang database.
- Analysis setups contain settings that drastically impact the resulting file and folder sizes (for example, the **Save Fields** check box). If saved fields consume disk space in the range of gigabytes for a single project, then the creation of hundreds of design folders with solution data by optiSLang can quickly fill a hard drive.

Considerations for Distributed Analysis

There are additional considerations for using Distributed Analysis and Large-Scale DSO to analyze optiSLang setups.

- Using the AEDT node's run mode "regular DSO" is not the only way to create DSO jobs. Also in run mode "single simulation" DSO jobs can be spawned. For example, you can use **Analyze All** to create a pre-existing Optimetrics setup that could be solved for every one of optiSLang's design evaluations.The integration node's **Designs Per Execution** setting can work with HPC and DSO settings for AEDT solvers in many different ways. If set to 4, optiSLang can send 4 designs per AEDT call when AEDT is geared at solving 4 design variations simultaneously. If **Designs Per Execution** is set to 9999, however,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

each optiSLang algorithm will always send all designs waiting in the pipeline at once (a robustness sampling of 1000 designs can be cast into one single DSO job). Using HPC settings, AEDT can still be made to break it down with ntasks=ncores=4. The drawback is that the success/failure info becomes visible in optiSLang only at the very end of the job. A large value for **Designs Per Execution** can make sense for optimization algorithms demanding design sets of fluctuating size.

- With population-based optimizers, it can be useful to match the population size with compute resources to avoid job load fluctuations.
- Several optiSLang nodes offer the feature to list files to be copied from the reference location into each design folder. However, this may not work for directories. A simple solution is to use a Python node and the shutil.copytree command.
- Avoid redundant solve actions. It may be helpful to copy not only the bare *.aedt file but also the *.aedtresults directory into each design folder. When the AEDT node issues the **Analyze All** command on the project level, pre-existing solution data can reduce the scope of analysis setups that are triggered to be solved. Compare the time for copying the data of solved projects against the time gain due to less computation. Avoid disc space concerns by activating design directory purging rules to be applied to every design folder after finished evaluation.
- AEDT offers a special type of postprocessing variables. When they are varied while other parameters stay constant, a new design variation can be evaluated with very little computational burden as compared to solving from scratch. The AEDT integration node provides copy functionalities for *.aedtresults folders allowing to benefit from AEDT features around postprocessing variables in certain use cases.

See: Example optiSLang Setups.

optiSLang Menu Options in AEDT

Once an optiSLang setup has been created in AEDT, additional menu options become available.

Right-click options include:

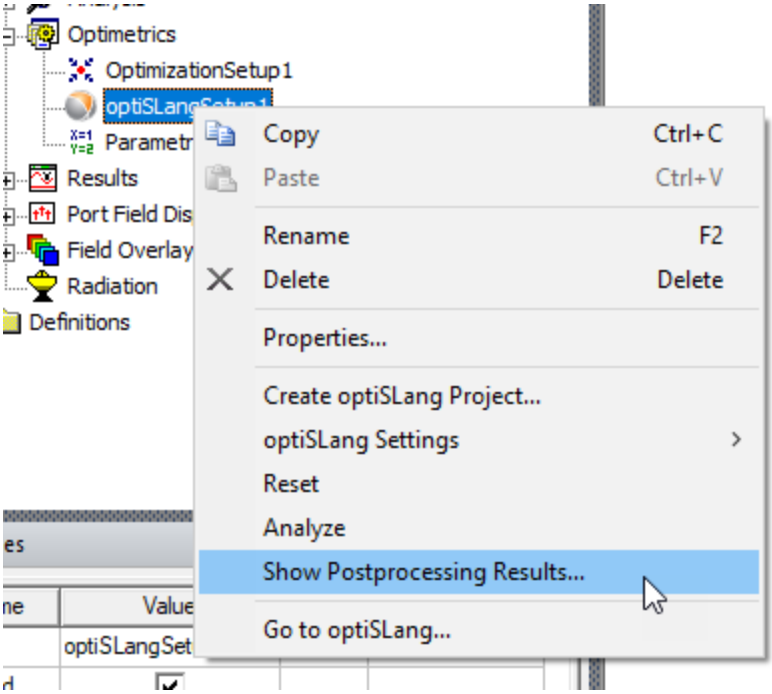
- **Create optiSLang Project** – creates an optiSLang project from the current design.
- **Edit AEDT Node Settings** – launches optiSLang with the AEDT project connected, on the Parametrization tab. For more information on node settings, consult the optiSLang help.
- **Edit Algorithm settings** – launches optiSLang and opens the Analysis Wizard, with options to analyze for Sensitivity, Optimization, or Robustness/Reliability. For more information on the Analysis Wizard, consult the optiSLang help.
- **Reset** – resets the optiSLang project.
- **Analyze** – launches the analysis of the optiSLang Optimetrics setup. When it has finished, post-processing results are available.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Show Postprocessing Results** – launches an optiSLang postprocessing results window. This menu option only available after analysis has been performed.
- **Go to optiSLang** – launches optiSLang.

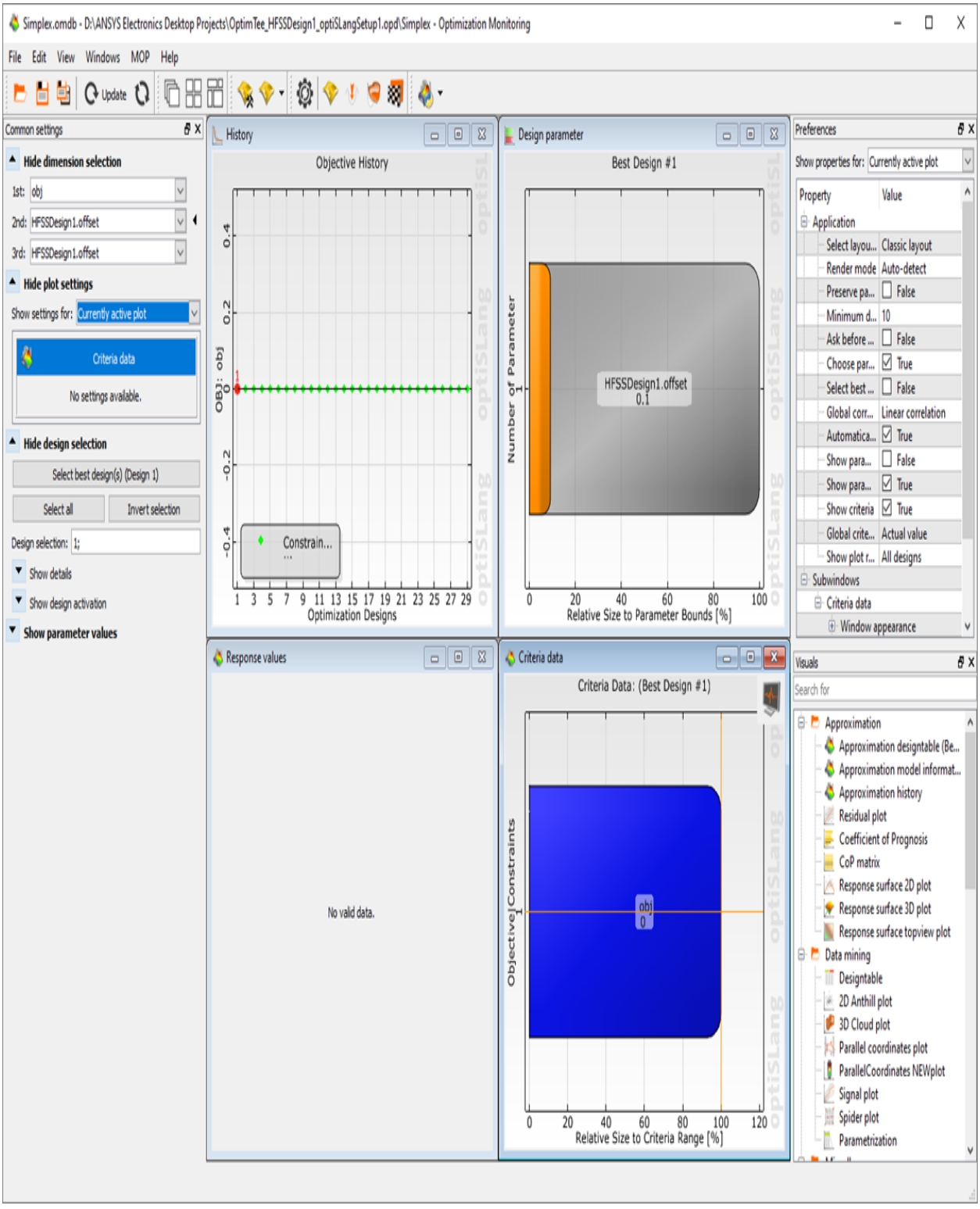
Viewing optiSLang Postprocessing Results

Once you have created and run an optiSLang setup, right-click it and select **Show Postprocessing Results**.



An optiSLang window opens, showing postprocessing features.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



For more information, use optiSLang's **Help** menu to access the optiSLang online help.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Tuning Overview

Tuning a variable is useful when you want to manually modify its value and immediately perform an analysis of the design. For example, it is useful after performing an optimization analysis, in which Optimetrics determines an optimal variable value, and you want to fine tune the value to see how the design results are affected.

You can update a design after a tuning analysis to reflect a design variation solved during a tuning analysis and the results of each solved design variation are saved for post processing.

Related Topics

Tuning a Variable

Tuning a Variable

1. Before tuning a variable, you must specify that you intend for it to be used during a tuning analysis in the **Properties** dialog box.
2. On the **Twin Builder** menu, click **Tune** . The **Tune** dialog box appears.
3. Clear the **Real Time** option. If this option is selected, a simulation begins immediately after you move the slider.
4. In the **Sim. Setups** column, select the solution setup you want Twin Builder to use when it solves the specified design variation. Twin Builder solves the analysis using the solution setup you select. If you select more than one, results are generated for all selected solution setups.
5. In the **Nominal** text box for the variable you want to tune, type the value of the variable for Twin Builder to solve, or drag the slider to increase or decrease its value.

   > **Warning:**
   >
   > Variable values must be single real numbers, or expressions that evaluate to single real numbers. Complex numbers cannot be used as the values of variables in any optimetric analysis.

   Alternatively, if you want Twin Builder to solve a range of values, specify a linear range of values with a constant step size:

   a. Select the **Sweep** check box.
   b. In the text box below the **Step** value, type the starting value in the variable range.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

c. Type the step size, or difference between variable values in the sweep definition, in the **Step** text box. The step size determines the number of design variations between the start and stop values. Twin Builder solves the model at each step in the specified range, including the start and stop values.

d. In the text box just below the variable name, type a stopping value in the variable range.

6. Click **Tune**.

   You can click **Abort** to abort the tuning process, if desired.

> **Note:**
>
> Sweeping or using a complex variable is not allowed in any optimetrics setup, including optimization, statistical, sensitivity, and tuning setups.

7. When finished tuning, you can:
   - Save the tuned state.
   - Apply the tuned state to the design.
   - Revert to a saved tuned state.
   - Reset variable values.

> **Note:**
>
> Improper or undefined simulation setups will cause errors during Tuning Analysis. To verify the analysis setups, select **Analysis** > **Analyze** in the **Project Manager** pane to analyze the nominal circuit and review the messages in the Twin Builder **Message Manager** pane prior to running the Tuning Analysis.

Related Topics

Applying a Tuned State to a Design

Tuning Overview

Resetting Variable Values after Tuning

Applying a Tuned State to a Design

You can apply the variable values solved during a tuning analysis to the nominal design in one of these ways:

- When closing the **Tune** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

a. Click **Close** to exit the **Tune** dialog box. The **Apply Tuned Variation** dialog box appears.
b. Click the design variation to apply, then click **OK**. The variable values from the solved design variation become the current variable values for the nominal design.

- When saving a tuned state.
- When reverting to a tuned state.

Saving a Tuned State

You can save the settings in the **Tune** dialog box, including the variable values you specified for a tuning analysis. Saved states are only available during the current session of the **Tune** dialog box; they are not stored for the next session.

1. After tuning a variable, click **Save** in the **Tune** dialog box. A **Save As** dialog box appears.
2. Type a name for the tuned state in the text box.
3. Select **Apply tuned values to design** if you want to update the model to the new variable values.
4. Click **OK** to return to the **Tune** dialog box.

Related Topics

Reverting to a Saved Tuned State

Reverting to a Saved Tuned State

You can revert to a group of saved settings in the **Tune** dialog box, including the variable values you specified for a specific tuning analysis. Saved states are only available during the current session of the **Tune** dialog box; they are not stored for the next session.

1. In the **Tune** dialog box, click **Revert**. The **Revert** dialog box appears.
2. Type the name of the tuned state you want to apply or click a name in the drop-down list.
3. Select **Apply tuned values to design** to update the model to the selected tuned state's variable values.
4. Click **OK** to return to the **Tune** dialog box.

Related Topics

Saving a Tuned State

Resetting Variable Values after Tuning

If you want to reset variable values to where they were when you started the current session, click **Reset** in the **Tune** dialog box after tuning a variable.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Solutions for the design variations solved during tuning analyses remain available for post processing.

Adding an Expression in the Output Variables Window

When you are in the **Output Variables** dialog box (click **Edit Calculation** from the one of the setup analysis windows), follow this procedure to specify an expression:

1. Type a name for the expression in the **Name** text box.
2. Follow this procedure in the **Calculation** section of the dialog box to insert a quantity into the expression:
   a. Select the **Report Type** and **Solution** from the drop-down lists.
   b. Select a **Category**, **Quantity**, and **Function** from the lists, and click **Insert Quantity Into Expression**.
   c. To insert a specific pre-defined function, select one from the **Function** drop-down list, and click **Insert Function**.
3. You can also type numbers or expression by hand directly into the **Expression** area.

Excluding a Variable from an Optimetrics Analysis

To exclude a variable from being optimized or included in a sensitivity or statistical analysis:

1. Do one of the following:
   - In the **Setup Optimization** dialog box, click the **Variables** tab.
   - In the **Setup Sensitivity Analysis** dialog box, click the **Variables** tab.
   - In the **Setup Statistical Analysis** dialog box, click the **Variables** tab.

   All of the independent variables selected for the optimization analysis appear.
2. Clear the **Include** option for the variable you want to exclude from the analysis.

   The **Override** option is now selected. This indicates that, for this optimization analysis, the variable is not included.

> **Note:**
>
> You can also select the **Override** option first, then clear the **Include** option for the variable you want to exclude.

3. Click **OK**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Modifying the Value of a Fixed Variable

If you are not including a variable in an optimization, sensitivity, or statistical analysis, Optimetrics uses that variable's current value during the analysis.

To override the current value of a fixed variable for an Optimetrics setup:

1. Do one of the following:
   - In the **Setup Optimization** dialog box, click the **Variables** tab.
   - In the **Setup Sensitivity Analysis** dialog box, click the **Variables** tab.
   - In the **Setup Statistical Analysis** dialog box, click the **Variables** tab.
2. Click **Set Fixed Variables** to open the **Setup Fixed Variables** dialog box. Under **Fixed Variables**, all of the current independent variable values appear.
3. Click the **Value** text box of the variable with the value you want to override.
4. Type a new value in the **Value** text box and press **Enter**.

   The **Override** option is now selected. This indicates that the value you entered is used for this Optimetrics setup; the current variable value set for the nominal design is ignored.

> **Note:**
>
> Alternatively, you can select the **Override** option and type a new value in the **Value** text box.

5. Optionally, click a new unit system in the **Units** text box.
6. Click **OK**.

To revert to a default variable value, clear the **Override** option.

Linear Constraints

Once the optimization variables are specified, the optimizer handles each of them as an *n*-dimensional vector *x*. Any point in the design space corresponds to a particular *x*-vector and to a design instance. Each design instance may be evaluated via Finite Element Analysis and assigned a cost value; therefore, the cost function is defined over the design space (cost(*x*): $R^n \rightarrow R$), where *n* is the number of optimization variables.

In practice, a solution of the minimization problem is sought only on a bounded subset of the R^n space. This subset is called the feasible domain and is defined via linear constraints.

You may constrain the feasible domain of a design variable by defining linear constraints for the optimization process. The feasible domain is defined as the domain of all design variables that satisfy all upper and lower bounds and constraints. Linear constraints are defined by the following inequalities:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$\sum_i \alpha_{ij} x_i < c_j \forall j$$

where

- α_{ij} are coefficients.
- c_j is a comparison value for the j^{th} linear constraint.
- x_i is the i^{th} designer parameter.

Related Topics

Setting a Linear Constraint

Setting a Linear Constraint

A linear constraint defines the linear relationship between variables. Setting linear constraints in Optimetrics is useful for establishing limitations involving linear combinations of variable values.

1. Do one of the following:
   - If you are setting up an optimization analysis, open the **Setup Optimization** dialog box and click the **Variables** tab.
   - If you are setting up a sensitivity analysis, open the **Setup Sensitivity Analysis** dialog box and click the **Variables** tab.
2. Click **Linear Constraint**. The **Linear Constraint** dialog box appears.
3. Click **Add**. The **Edit Linear Constraint** dialog box appears.
4. Click a **Coeff** text box and type a positive or negative coefficient value.
5. Click a condition, **<** (less than) or **>** (greater than), from the drop-down list.
6. Type the inequality value, which should be a constant value, in the text box to the right of the condition.
7. Click **OK**.

You return to the **Linear Constraint** dialog box. The left-hand side of the constraint appears in the **LHS** (left-hand side) column. The condition is listed in the **Condition** column, and the inequality value is listed in the **RHS** (right-hand side) column.

Related Topics

Modifying a Linear Constraint

Deleting a Linear Constraint

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Linear Constraints

Modifying a Linear Constraint

1. Do one of the following:
   - If you are setting up an optimization analysis, open the **Setup Optimization** dialog box and click the **Variables** tab.
   - If you are setting up a sensitivity analysis, open the **Setup Sensitivity Analysis** dialog box and click the **Variables** tab.
2. Click **Linear Constraint**. The **Linear Constraint** dialog box appears.
3. Click the row listing the constraint you want to modify and click **Edit**. The **Edit Linear Constraint** dialog box appears.
4. Optionally, click a **Coeff** text box and type a new coefficient value.
5. Optionally, click a different condition, **<** (less than) or **>** (greater than), in the drop-down list.
6. Optionally, type a different inequality value in the text box to the right of the condition and click **OK**.

You return to the **Linear Constraint** dialog box. The new coefficient value, the condition, and the inequality value appear in the **LHS** (left-hand side), **Condition**, and **RHS** (right-hand side) columns, respectively.

Deleting a Linear Constraint

1. Do one of the following:
   - If you are setting up an optimization analysis, open the **Setup Optimization** dialog box and click the **Variables** tab.
   - If you are setting up a sensitivity analysis, open the **Setup Sensitivity Analysis** dialog box and click the **Variables** tab.
2. Click **Linear Constraint**. The **Linear Constraint** dialog box appears.
3. Select the row listing the constraint to delete and click **Delete**.

Running an Optimetrics Analysis

Once you have created all necessary Optimetrics based analyses, you have several options for running the simulations.

- To use the **Analyze All** command at the project or design level to simulate the nominal problem and subsequently run all Optimetrics setups:

  In the **Project Manager** pane, right-click the **project** or **design** name and select **Analyze All**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- To use **Analyze All** from the Optimetrics menu to simulate only the Optimetrics based setups:

  In the **Project Manager** pane, right-click **Optimetrics** and select **Analyze All**.

- You can choose to analyze only the setups related to a specific Optimetrics type of analysis. To simulate setups of a specific type:

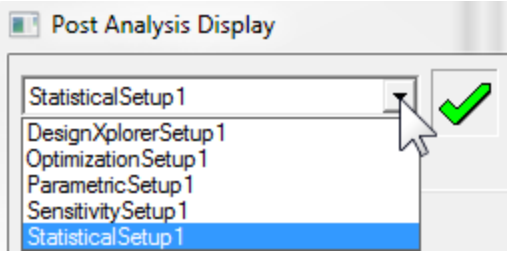
  In the **Project Manager** pane, right-click **Optimetrics** and select **Analyze All** {*TYPE*}, where *TYPE* is the specific analysis type of interest: **Parametric**, **Optimization**, **Sensitivity**, or **Statistical**.

> **Note:**
>
> Improper or undefined simulation setups will cause errors during Optimetrics Analysis. To verify the analysis setups, select **Analysis** > **Analyze** in the **Project Manager** pane to analyze the nominal circuit and review the messages in the Twin Builder **Message Manager** pane prior to running the Parametric Analysis.

Viewing Analysis Results for Optimetrics Solutions

Follow this general procedure to view data specific to an Optimetrics solution:

- In the Project tree, right-click the Optimetrics setup for which you want to view the results and select **View Analysis Result**. The **Post Analysis Display** dialog box appears.
- Select from available setups by using the drop-down menu.



- Select the **Result** tab to view results in plot or table form. When you view results in table form, you can sort the results based on each column.
  - Click the **Iteration** column header to sort the results by setup number.
  - Click the **Variable name** column to sort the results by step value.
  - Click the **Cost** column header to sort the results by cost.
- Click **Options** to specify the maximum number of significant digits to display when showing the analysis result. The default is **4**.
- Select the **Profile** tab to view start, stop, and elapsed times for each variable, and the analysis machine for each variation. Click the column headers to sort the table by

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

variation number, variable value, start, stop, or elapsed time, or (if you have run a distributed analysis) machine.

See the topics in this section for more details about viewing Optimetrics analysis results.

Related Topics

Viewing an Optimetrics Solution's Profile Data

Viewing Cost Results for an Optimization Analysis

Viewing Output Parameter Results for Sensitivity Analysis

Viewing Distribution Results for Statistical Analysis

Viewing an Optimetrics Solution's Profile Data

At any time during or after the Optimetrics solution process, you can see an overview of the computing resources or profile data used by Twin Builder as it solved each design variation. The profile data indicates the how long each design variation took to solve.

1. In the Project tree, right-click the Optimetrics solution setup of interest, and select **View Analysis Result**. The **Post Analysis Display** dialog box appears.
2. Click the **Profile** tab.
3. Select the Optimetrics setup with the results you want to view from the drop-down list at the top of the dialog box.
4. Optionally, to examine more detailed profile data for a specific design variation:
   a. Click a design variation in the table.
   b. Click **Solver Profile**.

The **Solutions** dialog box appears with the profile data for the selected design variation.

The profile line for the matrix solver is in the following format:

`Solver` *`123`*

where:

- *1* is the precision type: M (mixed) or D (double)
- *2* is the matrix data type: R (real) or C (complex)
- *3* is the symmetry type: S (symmetric), A (asymmetric), H (hermitian)

Plotting Solution Quantity Results vs. a Swept Variable

To plot solution quantity results versus a swept variable's values on a rectangular (x - y) plot:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. In the Project tree, right-click the parametric setup for which you want to view the results and select **View Analysis Result**. The **Post Analysis Display** dialog box appears.
2. Select **Plot** as the view type.
3. Select the variable with the swept values to plot on the X-axis from the **X** drop-down list.
4. Only one sweep variable at a time can be plotted against solution quantity results. Any other variables swept during the parametric analysis remain constant.

   Optionally, to modify the constant values of other swept variables:

   a. Click **Set Other Sweep Variables Value**. The **Setup Plot** dialog box appears. All of the other solved variable values are listed.
   b. Click the row with the variable value to use as the constant value in the plot and click **OK**.
5. Select the solution quantity results to plot on the Y-axis from the **Y** drop-down list. The xy plot appears in the view window.
6. Right-click in the plot area to get the shortcut menu where you can set modify the plots display properties, print, copy to the clipboard, or export the data to a file.

Viewing Cost Results for an Optimization Analysis

To view cost values versus completed iterations in data table format:

1. In the Project tree, right-click the optimization setup for which you want to view the cost results and select **View Analysis Result**. The **Post Analysis Display** dialog box appears.
2. Under the **Result** tab, select **Table** as the view type. The cost value at each solved design variation appears in table format.
3. Optionally, select a design variation in the table and click **Apply**.

Twin Builder now points to the selected design variation as the nominal solution and as a result, the design displayed in the **Modeler** window changes to represent the selected design variation.

Click **Revert** to return the design in the view window to the original value.

Related Topics

Plotting Cost Data for an Optimization Analysis

Plotting Cost Results for an Optimization Analysis

To view cost values versus completed iterations in rectangular (x-y) plot format:

1. In the Project tree, right-click the optimization setup for which you want to view the cost results and select **View Analysis Result**. The **Post Analysis Display** dialog box

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

appears.

2. Under the **Result** tab, select **Plot** as the view type. A plot of the cost value at each iteration appears.

Viewing Output Parameter Results for a Sensitivity Analysis

To view actual output parameter values versus design point in data table format:

1. In the Project tree, right-click the sensitivity setup for which you want to view the parameter results and select **View Analysis Result**. The **Post Analysis Display** dialog box appears.
2. Under the **Result** tab, select **Table** as the view type. The following values appear in table format:
   - The regression value of the output parameter at the design point is in the **Func. Value** column.
   - The first derivative of the regression is in the **1st D** column.
   - The second derivative of the regression is in the **2nd D** column.
3. Click **Apply**.

Twin Builder now points to the selected design variation as the nominal solution and as a result, the design displayed in the **Modeler** window changes to represent the selected design variation.

Click **Revert** to return the design in the view window to the original value.

Related Topics

Plotting Output Parameter Results for a Sensitivity Analysis

Plotting Output Parameter Results for a Sensitivity Analysis

Follow this procedure to plot output parameter results versus sensitivity variable values on a rectangular (xy) plot:

1. In the Project tree, right-click the sensitivity setup for which you want to view the output parameter results and select **View Analysis Result**. The **Post Analysis Display** dialog box appears.
2. Under the **Result** tab, select **Plot** as the view type.
3. Select the sensitivity variable with the sweep values to plot on the X-axis from the **X** drop-down list.
4. Select the output parameter results to plot on the Y-axis from the **Y** drop-down list.

The xy plot appears in the **Post Analysis Display** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The plot displays actual output parameter results for each solved design variation. It also displays a parabola that best fits these results. The parabola is a more accurate representation of sensitivity around the design point than any individual solved design variation.

Viewing Distribution Results for a Statistical Analysis

1. In the Project tree, right-click the statistical setup for which you want to view the distribution results calculated for the solution quantities and select **View Analysis Result**. The **Post Analysis Display** dialog box appears.
2. Select the statistical setup with the results to view from the drop-down list at the top of the dialog box.
3. To view the results in tabular form, select **Table** as the view type. The distribution results for the selected solution quantities appear in table format for each solved design variation.
4. Optionally, select a design variation in the table and click **Apply** (at the far right side of the dialog box). The design displayed in the **3D Modeler** window changes to represent the selected design variation.
5. To view the results in graphic format, select **Plot** as the view type.
6. Type the number of bins to plot on the X-axis.
7. Select the solution quantity for which you want to plot distribution results on the Y-axis from the **Y** drop-down list. A histogram plot appears in the **Post Analysis Display** dialog box. It displays the distribution of the selected solution quantity.
8. Optionally, select a design variation in the table and click **Apply** (at the far right side of the dialog box).

Twin Builder now points to the selected design variation as the nominal solution and as a result, the design displayed in the **Modeler** window is changed to represent the selected design variation.

Click **Revert** to return the design in the view window to the original value.

Related Topics

Plotting Distribution Results for a Statistical Analysis

Plotting Distribution Results for a Statistical Analysis

1. In the Project tree, right-click the statistical setup for which you want to view the distribution results calculated for the solution quantities and select **View Analysis Result**. The **Post Analysis Display** dialog box appears.
2. Select the statistical setup with the results to view from the drop-down list at the top of the dialog box.
3. Select **Plot** as the view type.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Type the number of bins to plot on the X-axis.
5. Select the solution quantity for which you want to plot distribution results on the Y-axis from the **Y** drop-down list.

A histogram plot appears in the **Post Analysis Display** dialog box. It displays the distribution of the selected solution quantity.

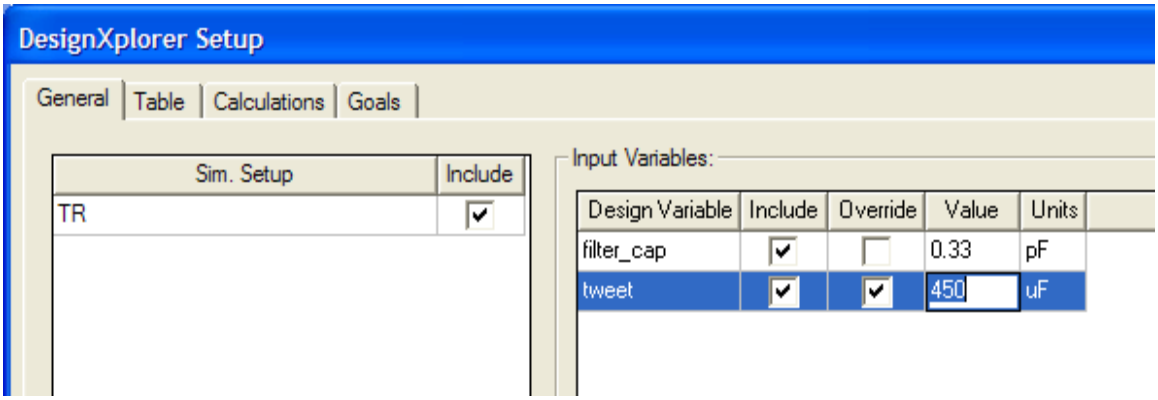
Link to DesignXplorer

You can export an XML file containing information on a Twin Builder setup, optimization variables, and output variables that enable Ansys DesignXplorer to manage Twin Builder simulations, for example, for design of experiments and optimization. DesignXplorer will launch Twin Builder simulations of design variations and evaluate the Twin Builder outputs.

To do so:

1. Click **Twin Builder** > **Optimetrics Analysis** > **Add DesignXplorer Setup** or right-click **Optimetrics** in the **Project** window, and select **Add DesignXplorer Setup**.

   The **DesignXplorer** dialog box appears. It lists the setups available in the current project, and the input variables it contains.



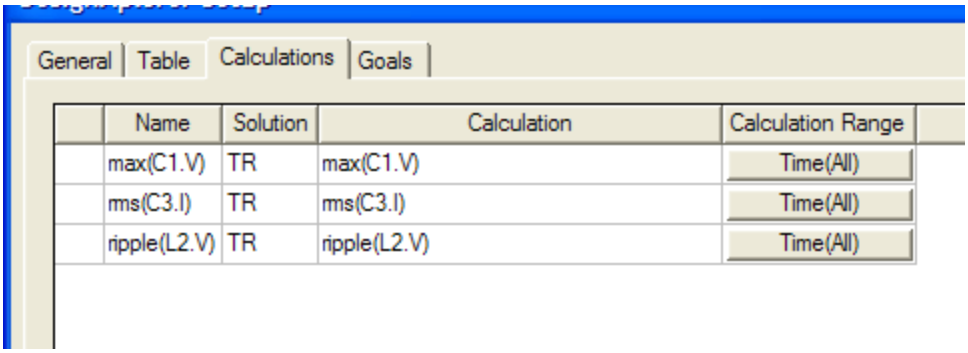
2. On the **General** tab, select the **Include** check box for the simulation setups you want to use.
3. In the **Input Variables** panel, select the **Include** check box for the **Design Variables** to use. You can also override the value of a design variable. Edit the **Value** and **Units** fields to enable **Override**. Clear the **Override** check box to return the value and unit to their original states.
4. To setup output calculations, select the **Calculation** tab and click **Setup Calculations**.

   The **Add/Edit Calculation** dialog box appears. Here you can define the simulation results of interest. The dialog box contains distinct panes and tabs to set the **Context**, the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

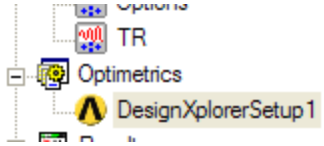
Calculation Expression, and the **Calculation Range**. See ***Setup Calculations for Optimetrics*** for details.

Click **Add Calculation** to add expressions to the Calculations table of the **DesignXplorer Setup** dialog box, **Calculations** tab.



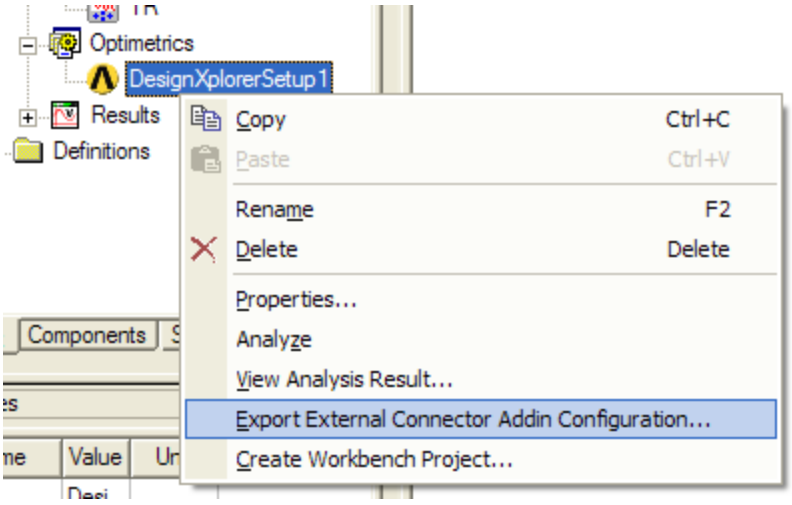
5. After adding the calculations of interest, click **OK** to save the setup.

   An icon for the DesignXplorer setup appears under **Optimetrics** in the **Project Manager** pane.



6. To create an XML file with the setup information for DesignXplorer, first **Save** your project.

7. Right-click the setup and select **Export External Connector Addin Configuration.**



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

This displays a browser dialog box that you can use to navigate through your file system, and name and save the XML file. The XML file contains information regarding the Twin Builder path along with the setup, variables, and simulation results that you specified.

8. If you have an Ansys Workbench installation you can perform additional steps. A path to the Workbench installation must be specified in **Tools** > **Options** > **General Options** on the **Miscellaneous** panel.
9. Click **Create Workbench Project**.

   This lets you name a Workbench project containing the information in the setup. The Ansys Workbench launches with the connection to the Twin Builder project established. To this connection, you can add a DesignXplorer Setup. See the documentation of Ansys Workbench for details on DesignXplorer.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

20 - Generating Reports and Postprocessing

After Twin Builder generates a solution, the results for that solution are available for analysis. One of the ways you can analyze your solution data is to create a 2D or 3D report, or graphical representation that displays the relationship between a design's values and the corresponding analysis results.

For each solution type (Transient, AC, and DC), use **Twin Builder** > **Results** > **Create Standard Report** > **Results** > **Bode Plot** or **Nyquist Report** to select from a cascading menu that lists the display types available for that report.

If you created custom report templates (for example, including your company name or other format changes), you can also create a report based on that template by selecting **Twin Builder** > **Results** > **Report Templates** > <*templateName*>. You can also access previously defined templates using **Report2D** > **Report Templates** > **Apply Settings**.

You can also use the **Report2D** > **Export** feature and select the report data format (RDAT) file which you can then select for **Create Report from File**.

When you select one of the standard report types, the **Report** dialog box appears.

Related Topics

The Report Dialog Box

Creating a New Report

Creating a Report from a Report Data File

Modifying Reports

Modifying the Background Properties of a Report

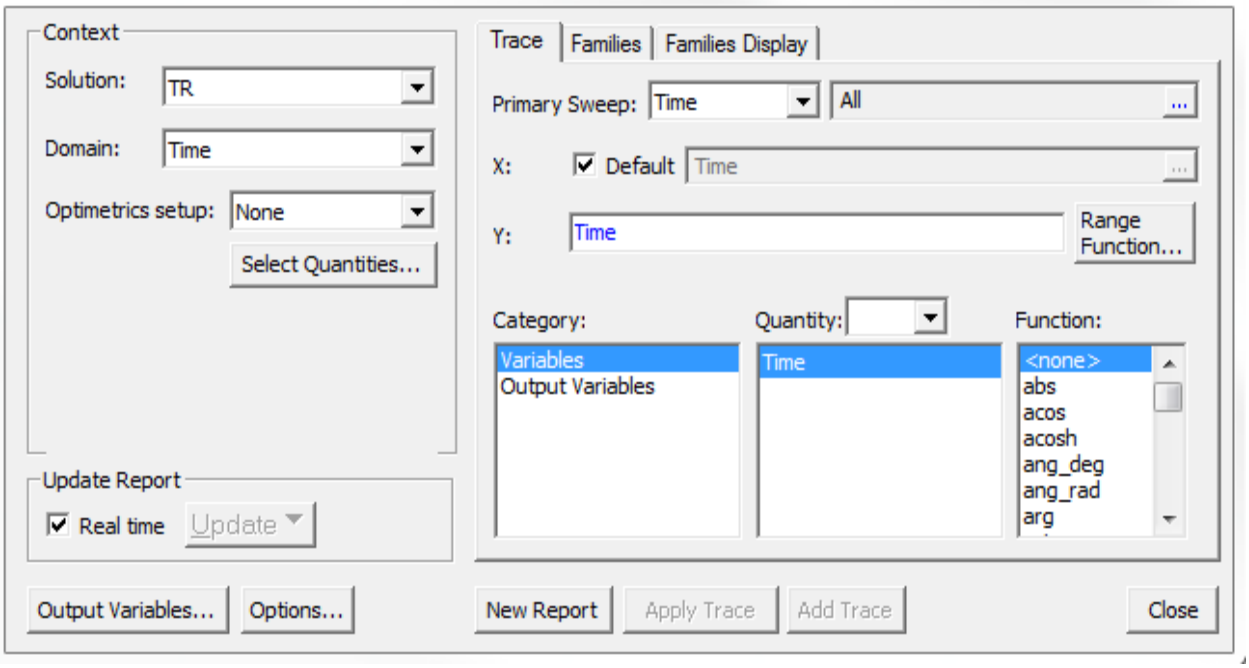
Creating Custom Report Templates

User Defined Outputs

The Report Dialog Box

The **Report** dialog box has the following sections:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Context Section:

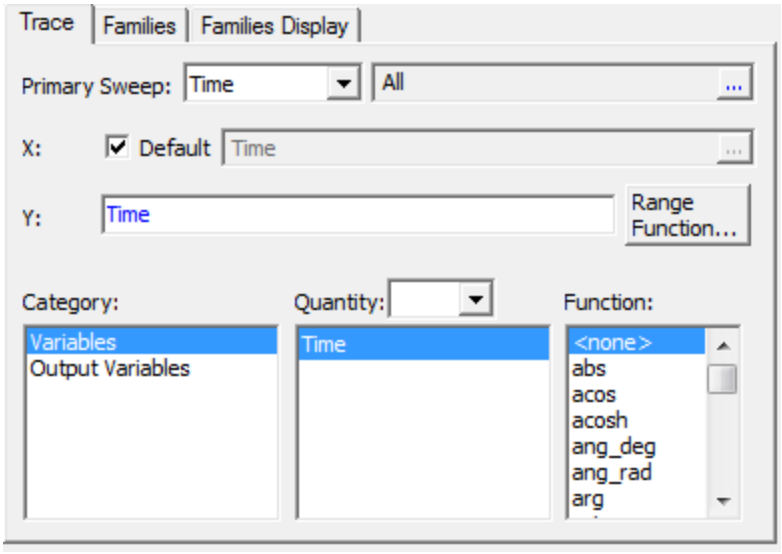
- **Solution** field – A drop-down list of available solutions (TR, AC, or DC).
- **Domain** field – A drop-down list of domains. Available selections may include **Sweep**, **Time**, or **Spectral** - depending on the **Solution** type.

  Select **Spectral** to display additional fields in which you can make settings for plotting spectral domain data.

- **Optimetrics setup** – All defined Optimetrics analysis setups.
- **Select Quantities** – Opens the **Select Quantities** dialog box in which you can select additional quantities for plotting.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

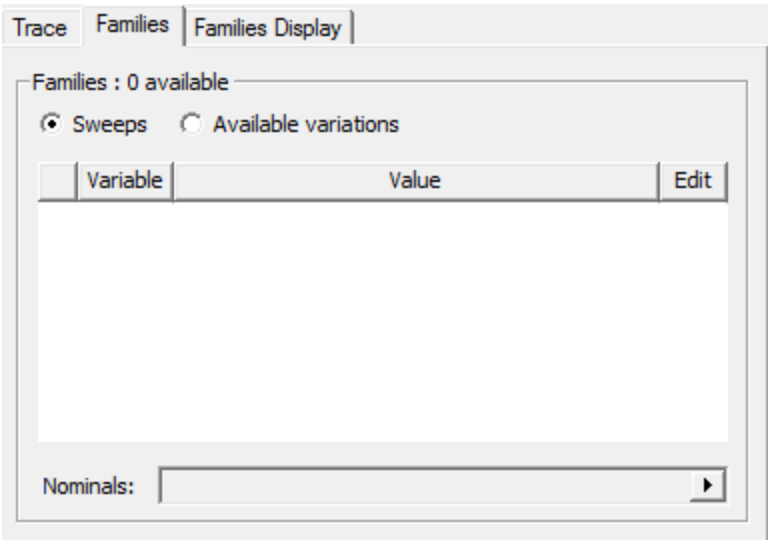
Trace Tab



- **X (Primary Sweep)** section
  - This contains a drop-down menu for selecting the values and a browse button for selecting from a list of sweeps (if available).
- **Y** section
  - Value field displays the currently specified quantity and function.
  - **Range Function** – Opens the **Set Range Function** dialog box. This applies currently specified quantity and function.
  - **Category** – Available categories depend on the solution type and the design. This lets you specify the category of information for the Y component.
  - **Quantity** for Y.
  - **Function** to apply to the Y quantities.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Families Tab:



- Each member of a family defines one point on a curve.
- Lists the number of families available.
- **Sweeps** display or **Available variations** display option buttons.

Sweeps lets you edit the swept variable values that appear when you click ... in the **Edit** field, then choose the desired variable values in the resulting dialog box. All values are selected by default.

Available variations lets you select individual combinations of values using check boxes. Click the **Select** column heading to select or clear all check boxes simultaneously. Click the variable name column heading to sort the listing.

- **Nominals** field (disabled if none exist in the design) lets you **Set All Variables to Nominal**, **Set All Unswept Variables to Nominal**, or to **Choose Nominals** to open a dialog box in which you can select the variables you want to set to nominal values.

Families Display Tab

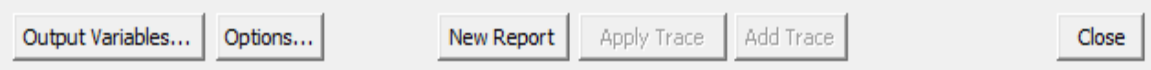
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Contains three option buttons: **All Families**, **Statistics**, and **Histogram**.
- **All Families** (default) – Enables traces for all families selected on the **Families** tab to display.
- **Statistics** – Select various statistical functions to apply to the traces selected on the **Families** tab. The selected functions can then be plotted on a new report, or added to the currently active report.
- **Histogram** – Generate a histogram plot based on the family of curves selected on the **Families** tab. The histogram representation is meaningful as long as the number of variations on which it is based is greater than one. The source of these variations can be from parametric, statistical, or optimization analyses, and even from multiple runs of the same design generated by changing the value of existing design/project variables. Observe the following general guidelines when generating histograms:
  - Use Time as a primary sweep for TR solutions. Use Frequency as a **Primary Sweep** for AC solutions.
  - For the X-axis definition, only the available **Primary Sweep** quantity can be used.
  - For the Y-axis definition, any available quantity can be selected.
  - A set of *N* variations, where *N* is greater than one, must be selected under the **Families** tab.
  - Set the desired **Number of bins** (the maximum number of rectangles used to represent the number of outcomes). The number of bins must be between 2 and 1000.
  - Set the **Value to sample at** - the instance value of the selected **Primary Sweep** in milliseconds for Time, or kilohertz for Frequency- at which the number of outcomes are computed among all *N* selected variations.

Update Report

- **Real Time** – Select to enable real time updates for all reports. Clear to enable the drop-down menu to **Update All Reports** or **Update This Report**. Traces are updated to latest data available if you click **Update This Report** or after last pass has been solved.

Report Dialog Box Buttons



- **Output Variables** – Opens the **Output Variables** dialog box.
- **Options** – Opens the **Report Setup Options** dialog box. This contains a check box for using the advanced mode for editing and viewing trace components. Advanced mode is automatic if the trace requires it. It also contains a field for setting the maximum number of significant digits to display for numerical quantities.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **New Report** – Adds a report to the Project tree under the **Results** icon. The new report appears in the **Project Manager** pane.
- **Add Trace** – Enabled when you add a new report, or select an existing report to modify. When you add traces, they appear in the **Project Manager** pane under the report.
- **Apply Trace** – Updates the selected traces in a report based on further processing or changes. When you edit a trace, this button applies the current values to that trace.
- **Close** – Closes the **Report** dialog box.

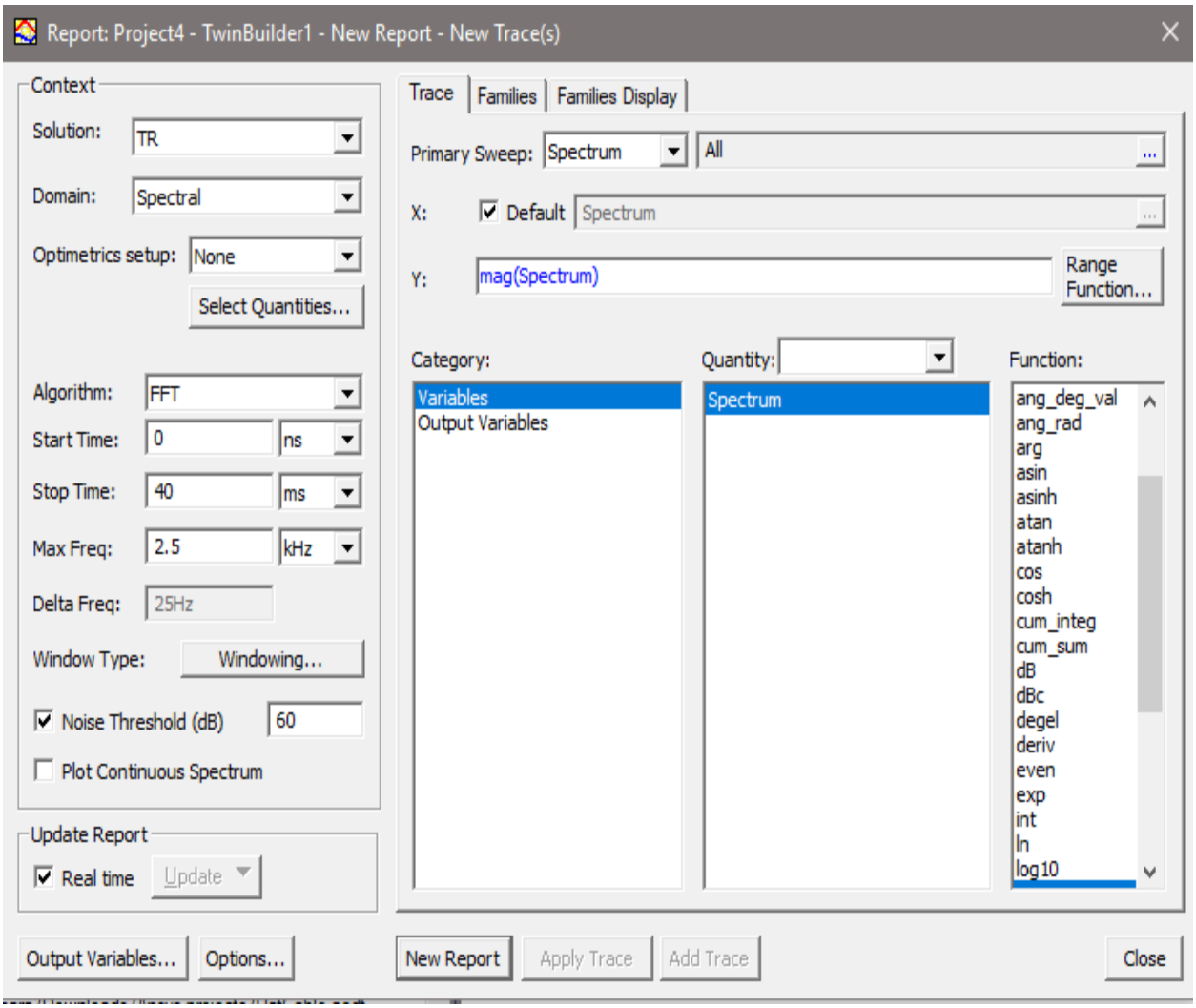
> **Note:**
>
> The evaluated value of an expression is always interpreted in SI units. However, when a quantity is plotted in a report, you have the option to plot values in units other than SI. For example, the expression
>
> ```
> 1+ang_deg(S11)
> ```
>
> represents an “angle” quantity evaluated in radians, though plotted in degree units. To represent an angle quantity in degrees, you would specify units as
>
> ```
> 1 deg + ang_deg(S11)
> ```

Plotting Spectral Domain Data

Follow this procedure to plot spectral domain data from a transient simulation in the Reporter:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Open the **Report** dialog box. The following dialog box appears.



2. Select the desired transient solution and set the **Domain** to **Spectral** in the **Context** area.
3. Set the **Start Time** and **Stop Time** as desired to avoid circuit startup artifacts or late time data.
4. **Max Freq** sets the maximum frequency for spectral analysis and plotting.

   Delta Freq is supplied as $1/T_0$ Hz, T_0 = Stop Time - Start Time.

   The **Algorithm** drop-down menu contains 3 options: **Fourier Integration (FI)**, **FFT**, and **Fourier Transform**.

   Fourier Integration and **FFT** both provide Fourier Series (or Fourier Coefficients) of the transient signal with DFT, where periodicity is implicitly assumed and the period is defined as T_0 = **Stop Time - Start Time**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Fourier Integration computes Fourier series by numerically integrating the input waveform from its transient simulation. The sampled waveform may not be on a set of evenly spaced time points. **Fourier Integration** is much slower compared with **FFT**, but it can be more accurate since it preserves the shape of underlying waveform without re-sampling.

The **FFT** algorithm re-samples the input waveform by interpolating to evenly spaced time points. It uses the FFTW library that is publicly available and whose computational complexity is

$$O(Nlog_2N)$$

where **N** is the sample size after re-sampling and N = 2 *#Harmonics +1, where #Harmonics = **Max Freq/Delta Freq**.

> **Note:**
>
> Based on the Nyquist-Shannon Theorem or Sampling Theorem, the sampling rate should be at least twice the bandwidth of the signal to avoid aliasing.
>
> Spectral plots based on Fourier Integration and FFT are single-sided with magnitudes of non-zero harmonics doubled because the negative frequency components are folded to the positive side.

Fourier Transform (FT) provides the Fourier transform of a signal defined in **[Start Time, Stop Time]** by leveraging the **FFT** algorithm. The Fourier transform of a time limited signal f(t) in [0, T_0] is

$$\mathrm{F}(f) = \int_{-\infty}^{\infty} \mathrm{f}(t) e^{-i2\pi ft} dt \; = \int_{0}^{T_0} f(t) e^{-i2\pi ft} dt$$

The Fourier series of a periodic signal that is equal to f(t) in [0, T_0] and with period T_0 is

$$c_n = \frac{1}{T_0} \int_{0}^{T_0} f(t) e^{-i\frac{2\pi}{T_0}nt} dt$$

Hence,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$c_n = \frac{1}{T_0}\mathbb{F}\left(\frac{n}{T_0}\right)$$

In other words, the Fourier series of f(t) based on **FFT** can be multiplied by T_0 in order to get the **Fourier Transform** of f(t) at sampled frequencies n/T_0.

> **Note:**
>
> Spectral plots based on **Fourier Transform** only show the positive frequency side.

5. Select the **Plot Continuous Spectrum** check box for a continuously plotted spectrum; clear the check box for a discrete spectrum.
6. You can toggle a **Noise threshold** using the check box (default value is 60 dB). This setting filters out data that is more than the specified amount below the maximum magnitude.
7. To control advanced settings such as **Window Type** and **Kaiser Param**, click **Windowing** to open the dialog box.
   a. Windowing functions cause the FFT of the signal to have non-zero values away from ω. Each window function trades off the ability to resolve comparable signals and frequencies versus the ability to resolve signals of different strengths and frequencies. You can apply a window to the transient data to help resolve closely spaced harmonics and to reduce spectral leakage due to the finite nature of the time signal. Choose the **Window Type** to apply to the data from the following list:

Window Function	Preferred Use
Rectangular (default)	A low dynamic range function offering good resolution for signals of comparable strength. Poor when signals have very different amplitudes. $w(n)=1$.
Bartlett	A high dynamic range function, with lower resolution, designed for wide band applications.
Blackman	A high dynamic range function, with lower resolution, designed for wide band applications.
Hamming	A moderate dynamic range function, designed for narrow band applications.
Hanning	A moderate dynamic range function, designed for narrow band applications.
Kaiser	Selecting the Kaiser plot also enables the **Kaiser Param** field to

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Window Function	Preferred Use
	specify an associated Kaiser parameter. The larger the Kaiser parameter, the wider the window. The parameter controls the trade-off between width of the central lobe and the area of the side lobes.
Welch	This approach divides the spectrum into overlapping periodogram bins in order to more accurately depict data away from the center of the signal.
Weber	
Lanzcos	The Lanczos window offers a windowed form of the infinite sinc filter, providing the central lobe of a horizontally-stretched sinc, sinc(x/a) for -a = x = a.

b. If desired, select the **Adjust Coherent Gain** check box, so that the signal levels are adjusted (based on the window type) as if no window were used. When non-rectangular windows are used, the window changes the power level of the signal. This is known as the “coherent gain” or “processing loss” of the window.

> **Note:**
>
> **Fourier Integration** may be slightly more accurate than FFT, but can be *much* slower, requiring a long time to create results.

6. When finished, click **OK** to save the settings.

For more information about the use of windows in harmonic analysis see: F.J. Harris, “On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform,” pp. 51-83, Proc. IEEE, vol. 66, no. 1, Jan 1978.

The Select Quantities Dialog Box

For a given simulation, you typically have an interest in looking at specific conditions in the schematic. You can specify these outputs for a simulation with the **Output** dialog box.

After running a simulation, the outputs you specified in the **Output** dialog box are available for selection in the **Report** dialog box **Trace** tab for the corresponding **Category**, such as voltage or current.

While in the **Report** dialog box, if you want to select additional quantities to plot, click **Select Quantities** to open the **Select Quantities** dialog box in which you to can select additional quantities for plotting. The **Select Quantities** dialog box is similar in operation to the **Output** dialog box. See ***Setting the Outputs for Simulation*** for details.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

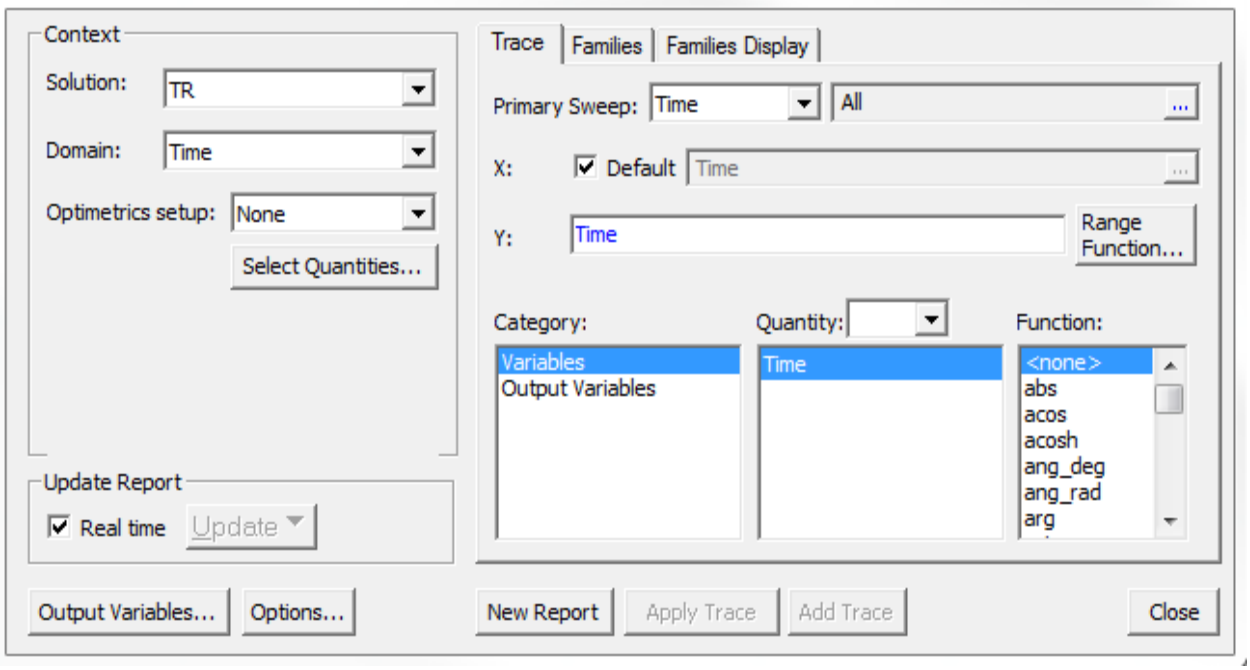
Creating a New Report

Follow this general procedure to create a new report. See the detailed procedures for the various report types.

1. Click **Twin Builder** > **Results** > **Create Standard Report**. You can also right-click **Results** in the **Project Manager** pane.

   If you have created custom report templates (for example, including your company name or other format changes), you can also create a report based on that template by selecting **Twin Builder** > **Results** > **Report Templates** > **PersonalLib** > *<templateName>*. You can also make such changes the default for new reports; right-click a modified report and select **Report Templates** > **Save Settings as default**.

   The **Report** dialog box appears.



2. In the **Context** section, make selections from the following fields, depending on the design and solution type.
   - **Solution** – The available solutions, whether sweeps or adaptive passes.
   - **Domain** – Domains relevant to the chosen **Solution**. For transient reports, the domain can be **Spectral** or **Time**. For AC reports, the domain is **Sweep**. For DC reports, the domain is **Time**.

     When **Spectral** is selected, additional fields display in which you can make settings for plotting spectral domain data.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Optimetrics setup** – All defined Optimetrics analysis setups.
- **Select Quantities** – Opens the **Select Quantities** dialog box in which you can select additional quantities for plotting.

3. The **Update Report** sections control whether reports are updated in real time. By default, reports are updated in real time. If the **Real time** check box is cleared, click **Update** to update either the current report or all reports manually.
4. In the **Y** Component section of the dialog box, make selections for the following:
   - **Category** – The contents of this list depend on the **Solution** type and the design. This field lets you specify the category of information for the Y component.
   - **Quantity** – Lists the Y component items available in the selected **Category**.

     > **Note:**
     >
     > The **Quantity** text field can be used to filter the **Quantity** list by typing in text. It is enabled if the selected category produces a lengthy **Quantities** list.

   - **Function** to apply to the Y quantities.
   - **Ytext box** – This text box displays the currently specified **Quantity** and **Function**. You can edit this field directly.

     > **Note:**
     >
     > The text color shows whether the expression is valid (blue for valid expressions, red for invalid).

   - **Range** - Click **Range** to open the **Set Range Function** dialog box. This applies currently specified quantity and function.
5. In the **X (Primary Sweep)** section, make selections for the following:
   - Select the Primary Sweep values from the drop-down list.
   - If sweeps are available, click to display a dialog box that lets you select particular sweep or sweeps, or all sweeps.
6. The **Families** tab provides a way to select from valid solutions for sweeps where a simulation has multiple variables defined (for example, for a parametric sweep). If so, the variables other than the one chesen as the **X (Primary sweep)**, appear under the **Families** tab with columns for the variable, the value, and an **Edit** column with a button. You can make selections for the following:
   - Select either **Sweeps** display or **Available variations** display option buttons.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Sweeps** – Edit the swept variable values. Click in the **Edit** field, then choose the desired variable values in the resulting dialog box.
- **Available variations** lets you select individual combinations of values via check boxes. Click the **Select** column heading to select or clear all check boxes simultaneously. Click the variable name column heading to sort the listing in either descending or ascending order.

- In the **Nominals** field (disabled if none exist in the design) choose either **Set All Variables to Nominal**, **Set All Unswept Variables to Nominal**, or to **Choose Nominals** to open a dialog box in which you can select the variables you want to set to nominal values.

7. On the **Families Display** tab, make selections for the following:
   - Choose **All Families** (default selection) to enable traces for all families selected on the **Families** tab to be displayed.
   - Choose **Statistics** to select various statistical functions to apply to the traces selected on the **Families** tab. The selected functions can then be plotted on a new report, or added to the currently active report.
   - Choose **Histogram** to generate a histogram plot based on the family of curves selected on the **Families** tab. Observe the following general guidelines when generating histograms:
     - Use Time as a primary sweep for **TR** solutions. Use Frequency as a **Primary Sweep** for **AC** solutions.)
     - For the X-axis definition, you can only use the available **Primary Sweep** quantity.
     - For the Y-axis definition, you can select any available quantity.
     - You must select a set of *N* variations, where *N* is greater than one, under the **Families** tab.
     - Set the desired **Number of bins** (the maximum number of rectangles used to represent the number of outcomes). The number of bins must be between 2 and 1000.
     - Set the **Value to sample at** to the instance value of the selected **Primary Sweep** – in milliseconds for Time, or kiloHertz for Frequency – at which the number of outcomes are computed among all *N* selected variations.
8. Click the **Report** buttons to create a new report with the settings you provide, or to modify an existing report.
   - **New Report** – Adds a report to the Project tree under the **Results** icon. The new report appears in the **Project Manager** pane.
   - **Add Trace** – Enabled when you add a new report or select an existing report to modify. Click this to add one or more traces to the report. New traces appear in the **Project Manager** pane under the report. This is enabled when you have created or selected a report.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Apply Trace** – Updates the selected traces in a report based on further processing or changes. When you edit a trace, this button applies the current values to that trace.
- **Output Variables** – Opens the **Output Variables** dialog box.
- **Options** – Opens the **Report Setup Options** dialog box. This contains a check box for using the advanced mode for editing and viewing trace components. This mode is automatic if the trace requires it. It also contains a field for setting the maximum number of significant digits to display for numerical quantities.
- **Close** – Closes the **Modify Report** dialog box.

9. Click **New Report** to create a new report in the Project tree.

   The report appears in the view window. It will be listed in the Project tree under **Results**, with the default name based on the report category you selected, for example, S Parameter Plot *n* or Output Variables Plot *n*. You can edit the plot names in the Project tree and the plot header text in the report synchronizes. Traces within the report also appear in the Project tree. Traces within the report also appear in the Project tree.

   Some plots may take time to complete. Saving in such cases after the plot has been created lets you review the plot later without having to repeat the calculation time when you reopen the project later.

10. To speed redraw times for changed plots, save. This saves the data that comprises expressions. For example if re(S11)*re(S22) is requested over multiple widths, each of the S11 and S22 are stored when you save. If you do not do a save of a changed plot, the changed version is not stored.

Related Topics

Modifying Reports

Creating Custom Report Templates

Creating a Report from a Report Data File

Using the Report Data File (RDAT) format, you can save (export) a report from one design, and import it into a different design (or even the same design). This lets you take a static snapshot of a set of simulation data and view it at a later stage of the design process, or view the same data set in one or more different designs.

To save a report to a Report Data File:

1. Create a report.
2. Right-click in the report window (or right-click the desired report icon in the Project tree) and select **Export** to open the **Export Report** dialog box.
3. Select the **Report Data** files option (**.rdat**) in the **Save as type** drop-down list.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Navigate to the desired location and either create a file name or select an existing file for the exported report, then click **Save**.

To create a report from an existing Report Data File:

1. Right-click the **Results** folder in the Project tree and select **Create Report From File**.
2. Browse to select the desired Report data file (**.rdat**) and click **Open**.
3. The selected report is added to the design and opens in a new report window.

Related Topics

Exporting Plot Data

Report File Formats

Modifying Reports

To modify the data that is plotted in a report:

1. In the Project tree, click the report you want to modify.
2. Right-click the report and select **Modify Report**. The **Report** dialog box appears.
3. Modify the report settings as desired. See ***Creating a New Report*** for detailed information on the various settings for this dialog box.

You can also view and edit the properties of reports and their traces via their properties windows. See ***Modifying Background Properties of a Report***.

Related Topics

Showing and Hiding Active View Objects

Modifying Background Properties of a Report

Modifying the Legend in a Report

Working with Traces

Editing the Display Properties of Traces

Adding a Characteristic to a Trace

Adding Data Markers to Traces

Creating Custom Report Templates

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Showing and Hiding Active View Objects

The **Active View Visibility** dialog box lets you control the visibility of various objects that appear on reports. Tabs in the dialog box list these objects by type, such as: **Traces**, **Notes**, **Legends**, and **Color Keys**.

Follow this procedure to toggle the display of objects in a report.

1. On the **View** menu or on the menu bar, click **Active View Visibility**. The **Active View Visibility** dialog box appears.
2. Select the tab for the object type to show or hide.

   By default, objects on a tab are listed by name in alphabetical order. Click the **Name** bar above the **Name** field to invert the order. For reports with large numbers of objects, you can resize the dialog box for easier selection.

3. On the desired tab, select the **Visibility** check box for objects you want to show in the active view window. Clear the box to hide objects.

   Click the **Visibility** bar above the check box field to toggle the visibility of all objects in the list.

   You can also use the **Name** text box to type in an object name, as well as control visibility for the named object. Click **Show** and **Hide**.

4. When finished making changes, click **Done** to close the dialog box.

Modifying the Background Properties of a Report

The standard zoom, pan, and fit commands operate on reports. When zooming on a report, axis labels and ticks adjust during the zoom operation and rescale to their final value after the zoom operation is complete.

Follow this procedure to modify the appearance of a report, or to modify the display properties of any object in a report, including traces, axis labels, grids, colors, fonts, legends, color maps, and so on:

1. Open the report you want to modify.

   > **Tip:**
   >
   > You must select an editable object in the report to be able to edit its properties.

2. Click an object to select it and to view its properties in the docked window. To open the **Properties** window, either double-click an object, or select **Edit** > **Properties** on the toolbar.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The **Properties** tabs and options display for editable plot objects varies depending on the report type (for example, whether 2D rectangular, 2D polar, Smith, Stacked, or 3D), but can include the following:

- **Header** – Edit the properties for the text displayed at the top of the report, including the title font, company name, show design name, subtitle font. The plot title is tied to the report's name and is not a header property. If you change the report name in the Project tree, plot title synchronizes. The company name and the show design name check boxes are grouped in the **Properties** dialog box as **Subtitle**. Edits to the subtitle font property affect both of them.
- **General** – This dialog box (or **General** tab for other report properties windows) lets you edit the background color (the perimeter around the trace display) for the plot, the contrast color (the trace display background), the field width, the precision, and whether to use scientific notation for marker and delta marker displays. (X and Y notation display is set separately, in the axis property tabs.)
- **Legend** – Lets you edit the properties for whether to show trace name, show solution name, and variation key. At least one of these three must be selected. You can also edit the font, the background color of the legend box, the border color, the border width, grid color (for the lines between trace descriptions), and the grid line width. See ***Modifying the Legend in a Report***.
- **Color Key** – For 3D plots, to control the appearance of the color key (colors, transparency, border appearance, fonts, number format, field width and precision.
- **Contour** – For 3D plots, to control the appearance of the color map, including map type, ramp color, spectrum, IsoValType, levels, number of contours, and values shown.
- **Radiation Pattern** – For 2D polar plots, whether to show the circular grid and angle lines.
- **Stacked** – For stacked plots, properties for X scroll bar, thumb properties, and stack layout, auto fit, and stack height.
- **Smith** – For Smith charts, whether to show grids for Imp., Adm., Cir, and angle lines.
- **Traces** – Select traces either in the legend or on the plot. The trace will turn green when selected. Editable properties for traces include color, line style, line width, trace type, whether to show a symbol, symbol frequency, symbol style, whether to fill symbol, symbol color, and whether to show arrows. See ***Editing the Display Properties of Traces***.
- **Grid** – Editable grid properties include grid border color, major and minor grid line colors and line styles, and whether to show the major and minor X and Y grids.
- **Axis** for X, Y or Z, or for Phi, Theta or Rho – The defaults for most of these values are set in the Report 2D Options Axis tab.
  - **Specify Name** – Select this to change the axis name.
  - **Name** – This describes the axis to which the following properties/options refer. These are selected in the **Modify Report** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Axis Color** – Set the color. Click the color bar to display the **Set Color** dialog box. Select a default or custom color and click **OK**.
- **Axis Font** – Click the cell to display the **Edit Font** dialog box in which you can select from a list of available fonts, styles, sizes, and colors. The dialog box also contains a preview field. Click **OK** to apply the font edits and close the dialog box.
- **Show Units** – Specify whether to display units.
- **Window (section)**
  - **Window Mode** – Can be axis range, continuous moving window, or step moving window.
  - **Window Width** (in) – Provide an integer value for the previous selection.
- **Manual Format** (section)
  - **Number format** – Select **Auto**, **Decimal**, or **Scientific notation** from the drop-down list.
  - **Field Width** – Enter a real value.
  - **Field Precision** – Enter a real value.
- X or Yn or Z **Scaling Tab** – These properties provide control over scaling.
- **Axis Scaling** – Use the drop-down list to select scaling as linear or log. For the Y-axis, all zero and negative values are discarded before log scaling is applied.
- **Specify Min** – Enables the **Min** field.
- **Min** – Minimum axis value. Text entry in same units as axis units. Saved as SI internally.
- **Specify Max** – Select this check box to set the maximum value to display. Clear to let Twin Builder set the maximum value.
- **Max** – Maximum axis value. Text entry in same units as axis units. Saved as SI internally.
- **Specify Spacing** – Select this check box to set the spacing value to display. Clear to let Twin Builder set the spacing value.
  - **Spacing** – The spacing of major tick divisions within the axis scaling range. Value must be greater than zero. If the value exceeds the specified axis range, spacing is adjusted (entry is in same units as axis units). Saved as SI internally.
  - **Minor Tick Divs** – The number of minor tick divisions displayed on the axis between major tick divisions. Must be between 1 and 20.
- **-Manual Units** (section)
  - **Auto Units** – Use the check box to auto-compute the correct units for the axis.
  - **Units** – Click the cell to select from a menu of available units if you have not selected **Auto Units**.
- **-Infinity Visualization** (section)
  - **Map Infinity Mode** – Check box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Each axis can be set to treat infinity values in a user defined way. When you check the map infinity mode, any infinity values in the input data get the infinityMap value (negative infinity gets the value*-1 and positive infinity the positive value specified). This can be useful if there are zeros, or very small values that Twin Builder treats as zero, in the data, for example, dB Gain.

- **Map Infinity To** – Enter a real value for the **Map Infinity Mode**.
- **Grid** – Properties for grid labels and grid style, appearance, and scaling. For the 3D rectangular plots, there are separate tabs for the XY, YZ and ZX axes, and for 3D Polar plots, tabs for phi-rho-theta, or theta-rho-phi grids.

3. When finished editing properties, click **OK** to apply the changes and close the dialog box.

Related Topics

Modifying Reports

Working with Traces

Modifying the Legend in a Report

Editing the Display Properties of Traces

Setting Report2D options

Creating Custom Report Templates and Defaults

Modifying the Legend in a Report

The legend in a report is a list of the traces being plotted. For each trace, the legend gives the name, shows the line color, and lists the setup and the adaptive pass used to generate the trace.

To show or hide a legend in a report:

1. Make the report the active view.
2. Select **View** > **Active View Visibility** or right-click the legend and select **View** > **Visibility** to display or hide the report.

   Either command displays the **Active View Visibility** dialog box.

3. Select the **Legends** tab. This lists the legend (or legends) in the report.
4. Use the **Visibility** check box to toggle display of the legend. Click **OK** to apply the change and close the dialog box.

To edit the display properties of a legend:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select the legend in a report. Click the Curve Info panel to display a docked properties window, or right-click the legend and select **Edit** > **Properties** to display the floating properties window. This lets you edit the properties for whether to show trace name, solution name, and variation key. At least one of these must be selected.

   You can also edit the font. Click the font cell to display the **Edit Text Font** dialog box. This dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and script. It also contains a preview field. Make your selections and click **OK** to apply the font edits and close the dialog box.

   You can also edit the background color of the legend box, the border color, the border width, grid color (for the lines between trace descriptions), and the grid line width.

2. Click **OK** to close the **Properties** dialog box and apply the selections.

Related Topics

Showing Objects

Hiding Objects

Modifying Reports

Spinning a 3D Report

You can set a 3D report spinning around any axis, to enable viewing from multiple angles without having to rotate repeatedly.

1. Select **View** > **Spin**. Alternatively, right-click the 3D graph and select **View** > **Spin**. The cursor changes to the spin mode cursor (curved arrows).
2. Click and drag the report in the direction and at the speed you want to spin the view. The image begins spinning continually.
3. To stop spinning, click in the report window. You can start another spin in the same or another direction and speed by repeating step 2.
4. To end spin mode, press **ESC** or deselect **Spin** on the **View** menu.

Working with the X-Axis Scrollbar (Rectangular, Stacked, Bode, and Nyquist Plots)

To facilitate finding data markers on 2D rectangular, stacked, Bode, and Nyquist plots, you can enable the X-axis scrollbar. To access the scrollbar settings, click any report object (trace, grid, legend, and so on), or right-click an object and select **Properties**.

The scrollbar settings are located on the **Cartesian** tab for rectangular, stacked, and Nyquist plots; or on the **Bode** tab for Bode plots, in either the docked **Properties** window or the **Properties** dialog box. Cartesian tab settings include:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Show X Scrollbar** – Select the box to display a scrollbar beneath the X-axis on the report.

The scrollbar spans the entire visible width of the plot. Names of data markers you added to the plot appear on the scrollbar showing their relative locations on the plot. A small tab (and vertical line - depending on the **Thumb trans** setting) appears under the scrollbar to mark the locations of X-data markers present on the plot.

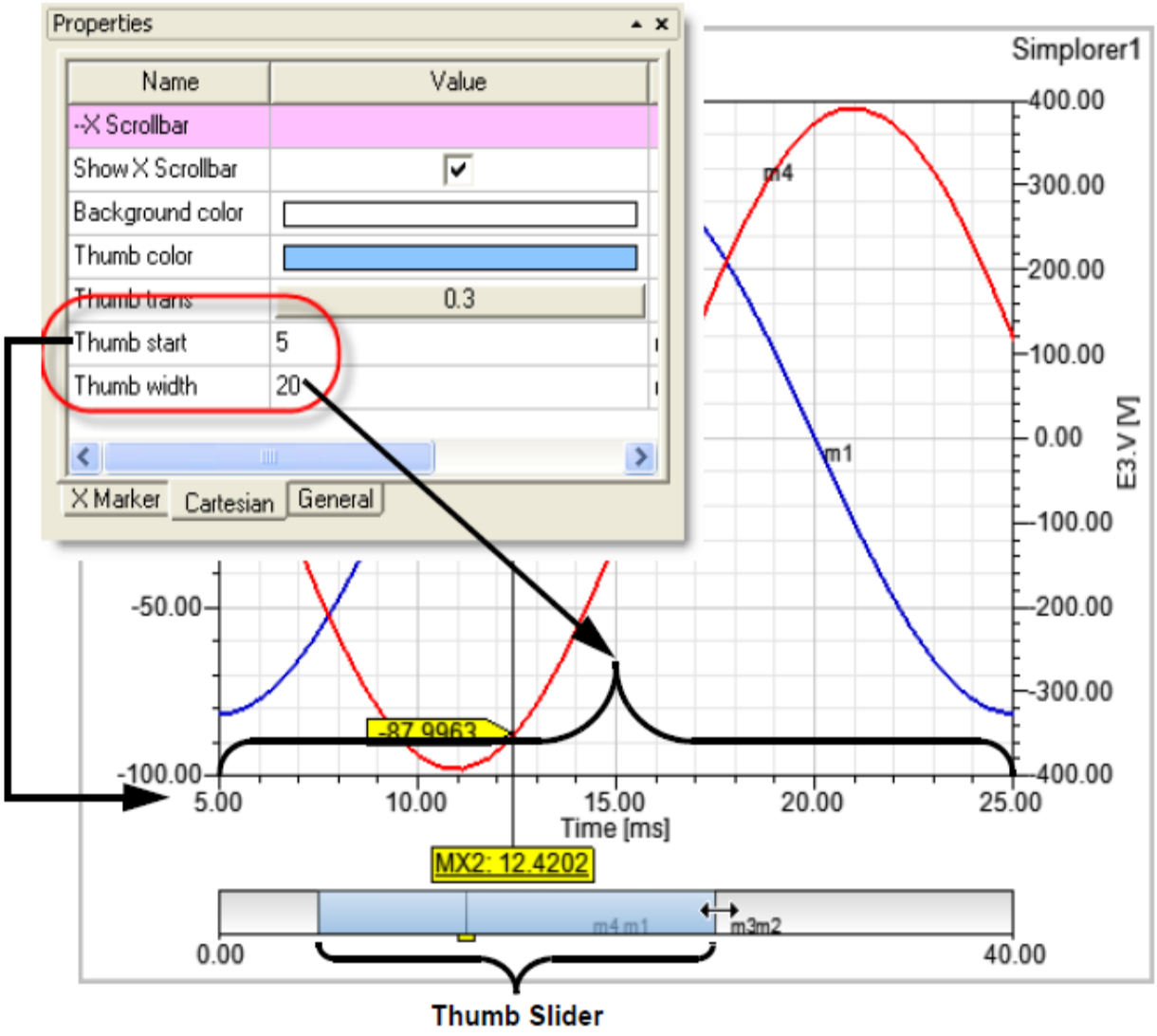
The scrollbar contains a “thumb” slider. When first displayed, the slider completely fills the scrollbar. In addition to the **Thumb width** setting in the **Properties** window, you can change the width dynamically by moving the mouse pointer over either the left or right edge of the thumb slider until the pointer changes to a horizontal two-headed arrow. Click on and slide the edge of the thumb to resize it. As the thumb width changes, the scale of the graph and the portion of the plot shown change accordingly — making the thumb narrower zooms in, widening the thumb zooms out. Sliding the thumb scrolls the plot horizontally in the graph window.

- **Background color** – Open a color selection box for setting the background color of the scrollbar.
- **Thumb color** – Open a color selection box for setting the “thumb” slider color.
- **Thumb trans** – Open a **Set Transparency** dialog box containing a slider that sets the transparency of the thumb. A text entry box shows the current setting. You can also enter a value between 0 (opaque) and 1 (transparent) directly in the box to set transparency.
- **Thumb start** – Sets the starting X-value of the plot displayed in the graph window.
- **Thumb width** – Sets the X-axis value range of the plot displayed in the graph window. The width of the thumb slider changes accordingly

For example, if the original plot’s X-axis range was 0 to 40, setting **Thumb start** to 5, and setting **Thumb width** to 20 changes the scale to show a range between 5 and 25 - effectively “stretching” or zooming in on the plot to allow a more detailed view.

> **Note:**
>
> You can restore the plot to its original view. Right-click anywhere in the plot window and select **View** > **Fit All**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Selecting the Display Type

You can configure report data to display in several formats. Select from the following **Display Type** formats in the **Create** <*type*> **Report** submenus:

Rectangular Plot	A 2D rectangular (x-y) graph.
Polar Plot	A 2D circular chart divided by spherical coordinates.
Data Table	A grid with rows and columns that displays, in numeric form, selected quantities against a swept variable or other quantity.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3D Rectangular Plot	A 3D rectangular (x-y-z) graph.
Rectangular Stacked Plot	A series of 2D rectangular (x-y) graphs stacked vertically, sharing a common X-axis scale, but with each trace having its own Y-axis scale.
Rectangular Contour Plot	
Digital Plot	A 2D rectangular graph especially designed to display quantities for VHDL models. Each quantity has its own coordinate system.
Bode Plot	A 2D rectangular (x-y) graph consisting of a magnitude plot of log magnitude versus log frequency, and a phase plot of phase versus log frequency.
Nyquist Plot	A 2D polar chart in which the gain (magnitude) and phase of a frequency response are plotted showing phase as the angle and magnitude as the distance from the origin.

> **Note:**
>
> Outputs that are Boolean (enum-type) can only be plotted in digital plots or in data tables.

Creating Rectangular Plots

A rectangular plot is a 2D, x-y graph of simulation results.

> **Note:**
>
> You must define a solution setup before attempting to create a rectangular plot.

To create a rectangular plot:

1. Select **Twin Builder** > **Results** > **Create Standard Report** > **Rectangular Plot**. The **Report** dialog box appears.
2. In the **Context** section make selections from the following field or fields, depending on the design and solution type.
   - **Solution** – This lists the available solutions, whether sweeps or adaptive passes.
   - **Domain** – Whether this field appears, and the domains listed depend on the solution type and the <*type*> selected. The domain can be **Sweep**, **Spectral**, or **Time**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

When **Spectral** is selected, additional fields display in which you can make settings for plotting spectral domain data. The magnitude function **mag()** is also applied to quantities selected for plotting in the Y field.

- **Optimetrics setup** – All defined Optimetrics analysis setups.
- **Select Quantities** – Opens the **Select Quantities** dialog box in which you can select additional quantities for plotting.

3. Under the **Trace** tab, **Y** component section, specify the information to plot along the Y-axis:
   - **Category** – Select the type of information to plot.
   - **Quantity** – Select the value to plot.
   - **Function** – Select the mathematical function of the quantity to plot.
   - The **Y** field displays the currently specified quantity and function. You can edit this field directly.

     > **Note:**
     >
     > - The text color shows whether the expression is valid (blue for valid expressions, red for invalid).
     > - When **Spectral** is selected, the magnitude function **mag()** is applied to quantities selected for plotting in the Y field.

   - **Range Function** – Click to open the **Set Range Function** dialog box. This applies currently specified quantity and function.
4. On the **Trace** tab, **X** (primary sweep) line, specify the quantity to plot along the X-axis in one of the following ways:
   - Select the sweep variable to use from the drop-down list.
   - If sweeps are available, to find and select sweeps. The quantity is plotted against the primary sweep variable listed.
5. On the **Families** tab, confirm or modify the sweep variables to be plotted.
6. Click **New Report**.

   This creates a new report in the Project tree, displays the report with the defined trace, and enables the **Add Trace** button on the **Report** dialog box. The default name is based on the report category you selected (for example, S Parameter Plot *n* or rE Plot *n*). You can edit the plot names in the Project tree and the plot header text in the report synchronizes.

   The function of the selected quantity is plotted against the swept variable values or quantities you specified on an x-y graph. The plot is listed under **Results** in the Project tree and the traces are listed under the plot. The default name is based on the report

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

category you selected, (for example, S Parameter Plot *n* or rE Plott *n*). You can edit the plot names in the Project tree, and the plot header text in the report synchronizes. When you move the mouse pointer close to a trace, the quantity name of that trace is displayed in a box that appears next to the cursor. When you hover the mouse over a trace in the **Curve Info** box, the corresponding trace changes color. When you select the traces or plots, their properties display in the **Properties** window. You can edit these properties to modify the plot.

7. Optionally, add another trace to the plot by following the procedure above, using **Add Trace** rather than **New Report**.

Related Topics

Sweeping a Variable

Working with Traces

Delta Markers in 2D Reports

Modifying Background Properties of a Report

Creating a Plot-On-Schematic

Creating Polar Plots

A polar plot is a 2D circular chart divided by the spherical coordinates *R* and *theta*, where *R* is the radius, or distance from the origin, and *theta* is the angle from the X-axis. Follow this general procedure to draw a polar graph of results:

1. Select **Twin Builder** > **Results** > **Create Standard Report** > **Polar plot**. The **Report** dialog box appears.
2. In the **Context** section, make selections from the following field or fields, depending on the design and solution type.
   - **Solution** – Lists the available solutions, whether sweeps or adaptive passes.
   - **Domain** – Whether this field appears, and the domains listed depend on the Solution type and the <*type*> selected. The domain can be **Sweep**, **Spectral**, or **Time**.

   When **Spectral** is selected, additional fields display in which you can make settings for Plotting Spectral Domain Data.

   - **Optimetrics setup** – All defined Optimetrics analysis setups.
   - **Select Quantities** – Click to open the **Select Quantities** dialog box in which you can select additional quantities for plotting.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. On the **Trace** tab, **Polar** area, specify the information to plot:
   - **Category** – Select the type of information to plot.
   - **Quantity** – Select the values to plot. Use Ctrl-click to make multiple selections.
   - **Function** – Select the mathematical function to apply to the quantity for the plot.
   - **Value** – Displays the currently specified quantity and function. You can edit this field directly.

     > **Note:**
     >
     > The text color shows whether the expression is valid (blue for valid expressions, red for invalid).

   - **Range** – Click to open the **Set Range Function** dialog box. This applies to the currently specified quantity and function.
4. On the **PrimarySweep** line, select the sweep variable from the drop-down list and the values against which to plot the polar component.
5. Click **New Report**.

   This creates a new report in the Project tree, displays the report with the defined trace, and enables the **Add Trace** button in the **Report** dialog box. The default name is based on the report category you selected (for example, S Parameter Plot *n* or rE Plot *n*). You can edit the plot names in the Project tree and the plot header text in the report synchronizes.

   The function of the selected quantity is plotted against the swept variable values or quantities you specified on an x-y graph. The plot appears under **Results** in the Project tree and the traces appear under the plot. When you move the mouse close to a trace, the quantity name of that trace displays in a box next to the cursor. When you hover the mouse over a trace listed in the **Curve Info** box, the corresponding trace changes color. When you select the traces or plots, their properties display in the **Properties** dialog box. You can edit these properties to modify the plot.

6. Optionally, add another trace to the plot by following the procedure above, using **Add Trace** rather than **New Report**.

Related Topics

Reviewing 2D Polar Plots

Sweeping a Variable

Working with Traces

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Reviewing 2D Polar Plots

For a polar plot of S-parameters, Twin Builder displays this derived information about the cursor's location in the lower-left corner :

MP	The magnitude and phase of the point.
RX	The normalized resistance (**R**) and reactance (**X**).
GB	An alternate view of the normalized resistance and reactance in the form of $R + jX = \frac{1}{G + jB}$ where • ***G*** = conductance • ***B*** = susceptance
Q	The quality factor.
VSWR	The voltage standing wave ratio, calculated from the equation $\frac{1 + \lvert S_{ij} \rvert}{1 - \lvert S_{ij} \rvert}$.

A scale below the plot displays the scale of points along the R-axis.

Related Topics

Creating 2D Polar Plots

Creating a Data Table or Numeric Display

A data table is a grid of rows and columns that displays, in numeric form, selected quantities against a swept variable or other quantities.

1. Select **Twin Builder** > **Results** > **Create Standard Report** > **Data Table**. The **Report** dialog box appears.
   - Similarly, if you drew a data table directly on a schematic using **Draw** > **Report** > **DataTable** or **Draw** > **Report** > **Numeric Display**, double-click the data table or numeric display on the schematic to open the **Report** dialog box.
2. In the **Context** section, make selections from the following field or fields, depending on the design and solution type.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Solution** – Lists the available solutions, whether sweeps or adaptive passes.
- **Domain** – Whether this field appears, and the domains listed depend on the solution type and the <*type*> selected. The domain can be **Sweep**, **Spectral**, or **Time**.

When you select **Spectral**, additional fields display in which you can make settings for plotting spectral domain data.

- **Optimetrics setup** – All defined Optimetrics analysis setups.
- **Select Quantities** – Click to open the **Select Quantities** dialog box in which you can select additional quantities for plotting.
- **Show last value only** – This check box is cleared by default for plot types in general. It is selected by default if a transposed data table has been added to the sheet using **Draw** > **Report** > **Numeric Display**.

3. On the **Trace** tab, select the quantity you are interested in and its associated function:
   - **Category** – Select the type of information to plot. The category selected provides the default name for the plot.
   - **Quantity** – Select the values to plot. Ctrl-click to make multiple selections.
   - **Function** – Optionally, select the mathematical function to apply to the quantity.
   - The **Y** field displays the currently specified quantity and function. You can edit this field directly.

     > **Note:**
     >
     > The text color shows whether the expression is valid (blue for valid expressions, red for invalid).

   - **Range Function** – Click to open the **Set Range Function** dialog box. This applies currently specified quantity and function.
4. On the **Primary Sweep** line, select the sweep variable from the drop-down list, and specify **All** values or select values.
5. For the **X** field, either select the **Default** check box to use the default *x* quantity, or clear it to choose another quantity in the **Select X Component** dialog box.
6. Click **New Report**.

   This creates a new report in the Project tree, displays the report with the defined traces, and enables **Add Trace** on the **Report** dialog box. The default name is based on the report category you selected, for example, S Parameter Plot *n* or Output Variables Plot *n*. You can edit the plot names in the Project tree and the plot header text in the report synchronizes.

   The Y quantity is listed at each variable value or additional quantity value you specified. The data table is under **Results** in the Project tree, and the defined traces are under the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

data table. When you select the traces or plots, their properties display in the **Properties** dialog box. You can edit these properties to modify the plot.

7. Optionally, add other traces to the plot by following the procedure above, using **Add Trace** rather than **New Report**.

Related Topics

Working with Data Tables

Sweeping a Variable

Working with Traces

Working with Data Tables

This section describes the various options and settings you can make when working with data tables.

- To change the column order in a data table, drag a column head to the desired position and drop it. Column and row headings remain visible when scrolling.
- To restore the original column order if columns have been moved (reordered), right-click anywhere in the table and select **Reset**.
- To add a new set of data to the table each time an analysis runs, right-click anywhere in the table and select **Accumulate**.
- To add a note to the data table, right-click anywhere in the table, select **Add Note**, and type the note text in the resulting dialog box. Once created, you can edit the note's font, background and border colors and visibility, and the border width.

The following tabs appear in both the **Properties** window and **Properties** dialog box for the data table.

Data Table Tab

- **Show Global Min-Max** – Displays the global minimum and maximum values of the unfiltered data values in each column (or row if the table has been transposed). These values appear at the beginning of the table.
- **Show Local Min-Max** – This field appears only if you choose to specify minimum and maximum values on the **Data Filter** tab. Displays the local minimum and maximum values of the data values in each column (or row if the table has been transposed). These values appear at the beginning of the table.
- **Transpose** – Rearranges the data table such that each column of data becomes a row. Column and row headings remain visible when scrolling.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Num Data Per Page** – Sets the number of rows of data displayed per page. When multiple pages are available for viewing, use the arrows and page numbers at the bottom of the table to navigate through the pages.
- **Goto Page** – This field appears only if multiple pages of data are present. Click to open the **Goto Page** dialog box where you can choose the desired data page.
- **Show Trace Name** – This check box toggles display of all trace names.
- **Show Solution Name** – This check box toggles display of all solution names.
- **Show Variation Key** – This check box toggles display of all variation key names.
- **Font** – Opens the **Font** dialog box in which you can set the font characteristics for the table.
- **Back Color** – Sets the table background color.
- **Border Color** – Sets the table border color.
- **Border Width** – Sets the table border width.
- **Grid Color** – Sets the table grid line color.
- **Grid Line Width** – Sets the table grid line width.

Data Filter Tab

You can filter table data via the **Properties** window **Data Filter** tab which can be opened by selecting any table column. (The **Data Filter** tab is also available in the table **Properties** dialog box; double-click any column heading.) You can also set table units and formatting on the **Data Filter** tab.

The following settings are available:

- **Units** – Sets the unit of measure for the selected field.
- **Number Format** – Sets whether the data in the selected field displays in decimal or scientific notation.
- **Field Width** – Sets the total number of digits displayed for the data in the selected field.
- **Field Precision** – Sets the number of digits displayed to the right of the decimal point for the data in the selected field.
- **Specify Min** and **Specify Max** – When selected, lets you set the **Min** and/or **Max** data values of the selected field to be included in the filtered list of data.
- **Pare to** – Filters the data table on the selected field using Pare to **Minimum**, **Maximum**, or user-specified **Pare To Value**.

Header Tab

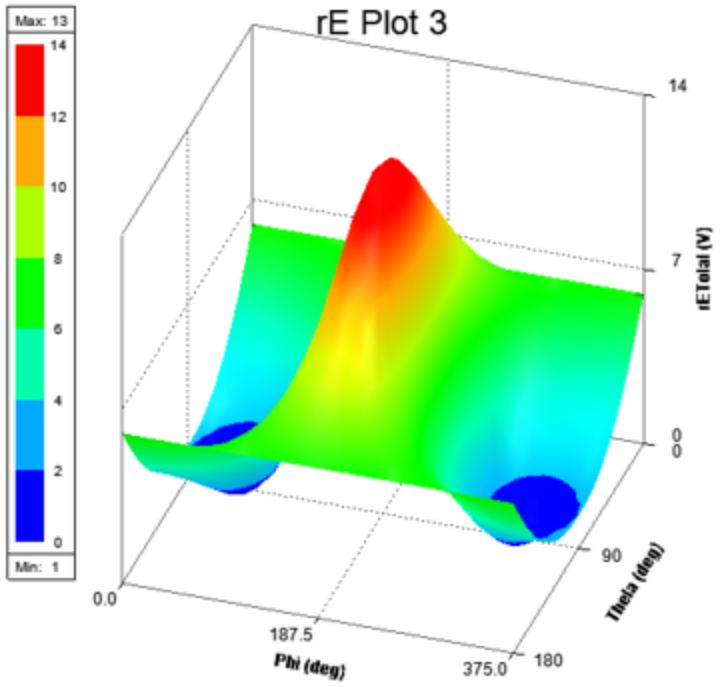
- **Title Font** – Opens the **Font** dialog box in which you can set the font characteristics for the table title.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Sub Title Font** – Opens the **Font** dialog box in which you can set the font characteristics for the table subtitle.
- **Company Name** – Field in which you can enter text for the subtitle.
- **Design Name** – Toggles display of the design name in the table header.

Creating 3D Rectangular Plots

A 3D rectangular plot is an x-y-z graph of results.



Working with a 3D Rectangular Plot

You can rotate, zoom, and pan a plot. When you rotate, the Cartesian grid responds so that the curve always remains in front and the grids behind.

Click a plot entity to select it and highlight it in bold.

Double-click anywhere in the plot to open the **Properties** dialog box. The properties are grouped appropriately under various tabs, which correspond to plot entities:

- **General** – For general plot properties such as visual detail level and background color.
- **Header** – Properties related to plot header/title.
- **Axis [X|Y|Z]** – Properties related to the 3 axes.
- **Grid [XY|YZ|ZX]** – Properties related to the 3 grids.
- **ColorKey** – Properties related to ColorKey, including borders, background, min and max, as well as number format and precision.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Contour** – Properties related to contouring of all curves/surfaces.
- **Surface** – Properties related to the curve.

Selecting a property also displays its properties in the **Property** window. You can edit the properties to customize the appearance of the plot. See ***Controlling Visual Detail in a 3D Rectangular Plot***.

Creating a 3D Rectangular Plot

1. Select **Twin Builder** > **Results** > **Create** > **Standard Report** > **3D Rectangular Plot**. The **Report** dialog box appears.
2. In the **Context** section, make selections from the following field or fields, depending on the design and solution type.
   - **Solution** – Lists the available solutions, whether sweeps or adaptive passes.
   - **Domain** – Whether this field appears, and the domains listed depend on the Solution type and the <*type*> selected. The domain can be **Sweep**, **Spectral**, or **Time**.

     When **Spectral** is selected, additional fields display in which you can make settings for plotting spectral domain data. The magnitude function **mag()** is also applied to quantities selected for plotting in the Z field.

     - **Optimetrics setup** – Lists all defined Optimetrics analysis setups.
     - **Select Quantities** – Click to open the **Select Quantities** dialog box in which you can select additional quantities for plotting.
3. Under the **Z** Component area, specify the information to plot along the Z-axis:
   - **Category** – Select the type of information to plot.
   - **Quantity** – Select the value to plot.
   - **Function** list – Select the mathematical function of the quantity to plot.
   - The **Z** field displays the currently specified quantity and function. You can edit this field directly.

     > **Note:**
     >
     > - The text color shows whether the expression is valid (blue for valid expressions, red for invalid).
     > - When **Spectral** is selected, the magnitude function **mag()** is applied to quantities selected for plotting in the Z field.

   - **Range Function** – Click to open the **Set Range Function** dialog box. This applies currently specified quantity and function.
4. On the **Y** (secondary sweep) lines, specify the information to plot along the Y-axis in one of the following ways:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Select the sweep variable to use from the drop-down list.
- If sweeps are available, click to select particular values. The quantity is plotted against the primary sweep variable listed.

5. On the **X** (primary sweep) lines, specify the information to plot along the X-axis in one of the following ways:
   - Select the sweep variable to use from the drop-down list.
   - If sweeps are available, click to select particular values. The quantity is plotted against the primary sweep variable listed.
6. Click **New Report**.

   This creates a new report in Project tree, displays the report with the defined trace, and enables **Add Trace** on the **Report** dialog box. The default name is based on the report category you selected, (for example, S Parameter Plot *n* or rE Plot *n*). You can edit the plot names in the Project tree and the plot header text in the report synchronizes.

   The function of the selected quantity or quantities is plotted against the values you specified on an x-y-z graph. The plot is listed under **Results** in the Project tree. When you select the traces or plots, axis or grid labels, plot header, color key, or variable labels, their properties appear in the **Properties** dialog box. You can edit the properties for each plot element to modify the plot content and appearance. See ***Modifying the Background Properties of a Report***.
7. Optionally, add another trace to the plot by following the procedure above, using **Add Trace** rather than **New Report**.

Related Topics

Sweeping a Variable

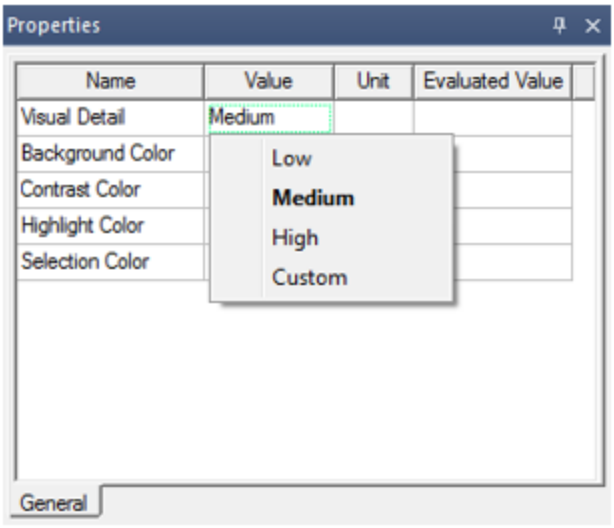
Working with Traces

Adding Characteristics to a Trace

Controlling Visual Detail in a 3D Rectangular Plot

If a particular plot seems busy with information, you can edit plot properties, such as axis and grid attributes, for discrete levels of visual detail to improve readability. Double-click anywhere on a plot to display the **Properties** dialog box. The **Visual Detail** property on the **General** tab also provides control suited to different screen and plot sizes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

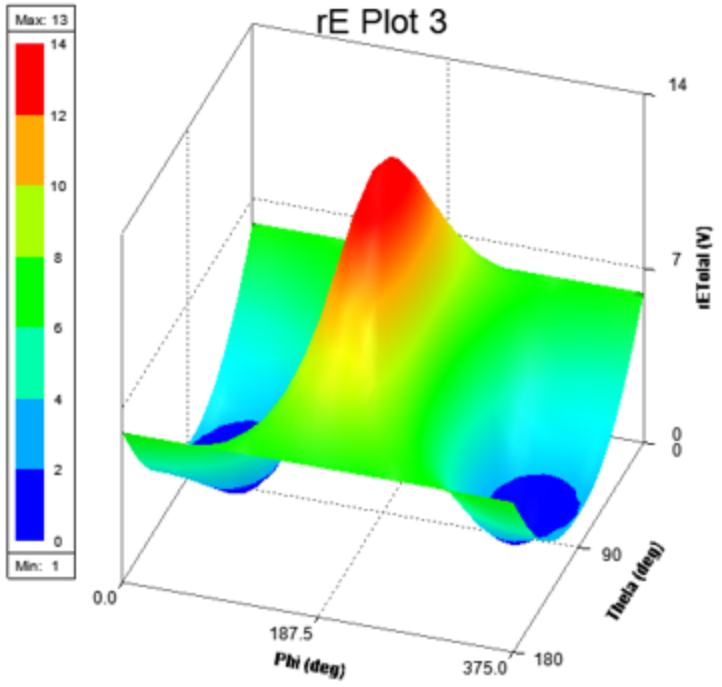


The **Visual Detail** menu has four options: **Low**, **Medium**, **High**, and **Custom**. If you select any visual detail, the 3D plot renders according to the selected visual detail level and the properties reflect the values chosen for the selected visual detail level. From this predefined visual detail level, if you modify any properties, **Visual Detail** is set to **Custom** (or to another predefined visual detail level if the edits happen to match the settings for that level).

You can also manually set **Visual Detail** to **Custom**. In such a case, **Custom** will inherit property values corresponding to the previous level. This ensures that you can customize settings starting from a baseline provided by the preconfigured **Low**, **Medium**, or **High** visual detail levels.

3D Rectangular Plot with Medium Visual Detail

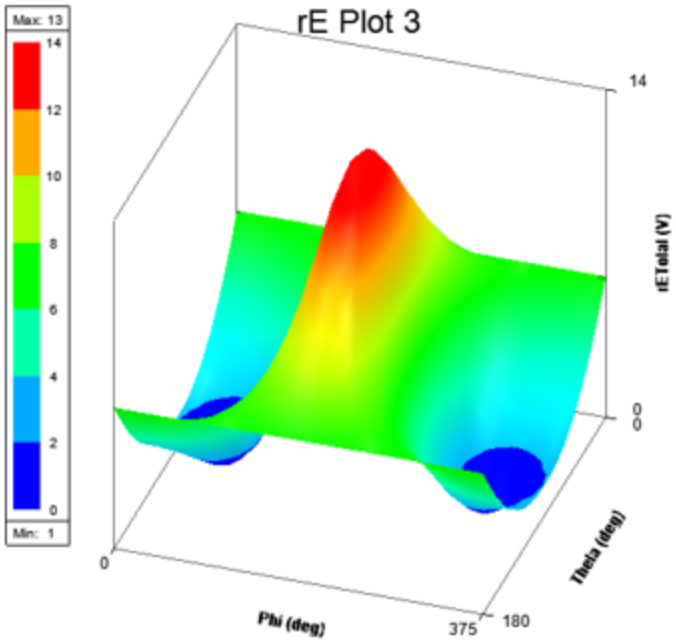
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



On creation, a 3D rectangular plot has **Visual Detail** set to **Medium** and looks and feels as shown above. Specifically, under medium visual detail level, a 3D rectangular plot has 3 ticks per axis (X-, Y-, Z-axis) which will show minimum, maximum, and middle value. This setting also shows axes labels.

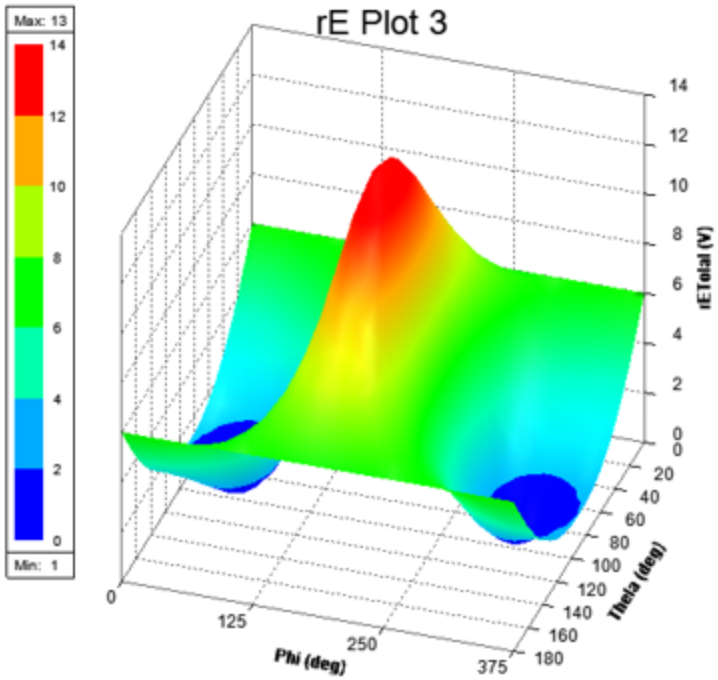
3D Rectangular Plot with Low Visual Detail

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



With **Visual Detail** set to **Low**, a 3D rectangular plot shows axes with two ticks corresponding to minimum and maximum values. It also shows axes labels and grid borders.

3D Rectangular Plot with High Visual Detail

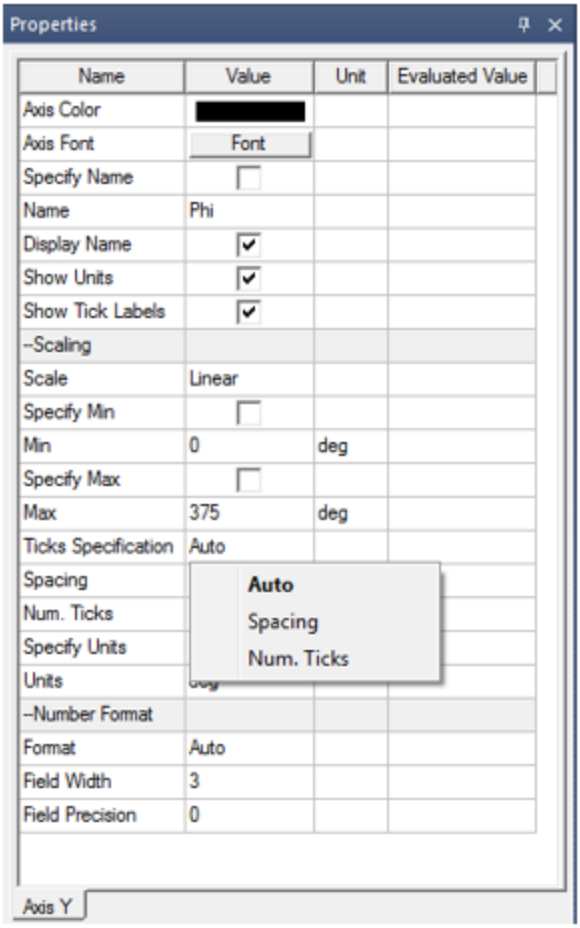


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

With **Visual Detail** set to **High**, a 3D rectangular plot shows all Cartesian axes and grids together with all ticks and axes labels.

Axis Properties: Ticks Specification and Num. Ticks

Ticks Specification is available on axis properties, as shown below:



The **Ticks Specification** menu contains the **Auto**, **Spacing**, and **Num. Ticks** values. If **Ticks Specification** is **Auto**, then a spacing value is calculated and used to calculate and display the tick labels. **Spacing** shows the calculated value, and **Num. Ticks** shows the number of ticks based on this spacing value as shown below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

--Scaling			
Scale	Linear		
Specify Min	☐		
Min	0	deg	
Specify Max	☐		
Max	375	deg	
Ticks Specification	Auto		
Spacing	125	deg	
Num. Ticks	2		

You can edit the **Spacing** field when **Ticks Specification** is set to **Spacing**; otherwise, it is read-only.

You can edit the **Num. Ticks** field when **Ticks Specification** is **Num. Ticks**; otherwise, it is read-only.

Valid **Num. Ticks** are between **0** and **100**, including **0** and **100**. If you enter an invalid value, an error message displays. If you enter a spacing value that results in number of ticks greater than 100, then an appropriate value is shown.

- If **Num. Ticks** is **0**, then no ticks appear on the axis.
- If **Num. Ticks** is **1**, then only the maximum value tick appears on the axis.
- If **Num. Ticks** is **2**, then only the minimum and maximum value ticks appear on the axis.
- If **Num. Ticks** is greater than **2**, then evenly spaced ticks (including minimum and maximum) appear on the axis.

> **Note:**
>
> With the addition of the **Ticks Specification** property to axis properties, the **Specify Spacing** property was removed as an axis property.
>
> - When you open an R18.0 or R18.1 project with the **Specify Spacing** check box cleared, **Ticks Specification** is set to **Auto**.
> - When you open an R18.0 or R18.1 project with the **Specify Spacing** check box selected, **Ticks Specification** is set to **Spacing**.

Related Topics

Sweeping a Variable

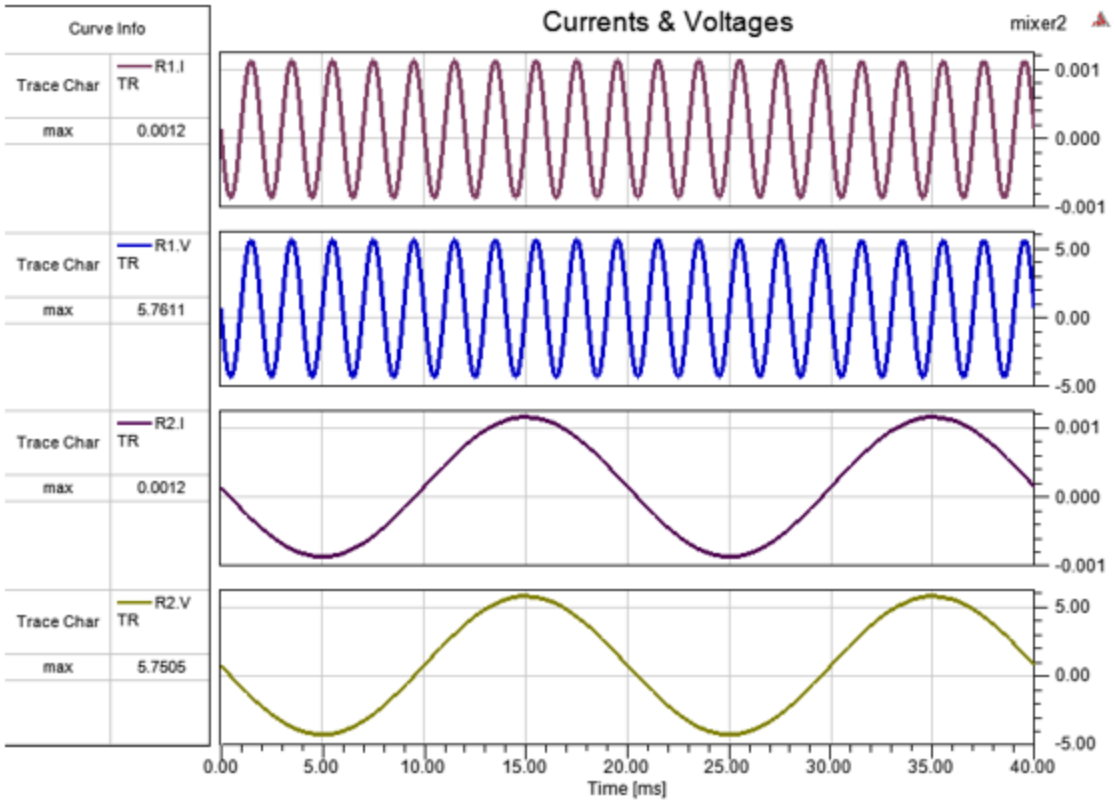
Working with Traces

Adding Characteristics to a Trace

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

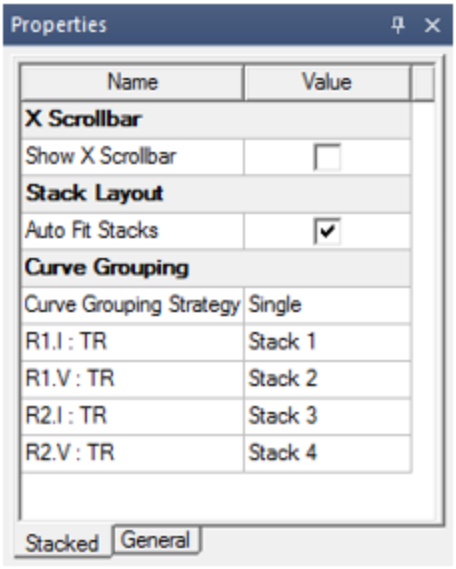
Creating a Rectangular Stacked Plot

A rectangular stacked plot is a 2D, x-y graph of results, with each trace displayed on a separate plot, or with multiple curves grouped in a single stack. The following figure shows one curve per stack.

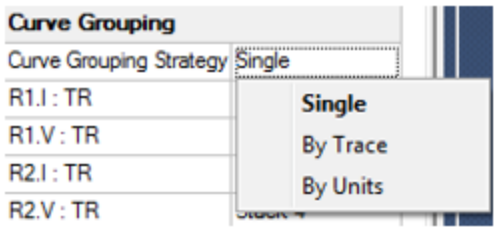


The **Stacked** tab in the **Properties** window shows the **Curve Grouping Strategy** property, as shown below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

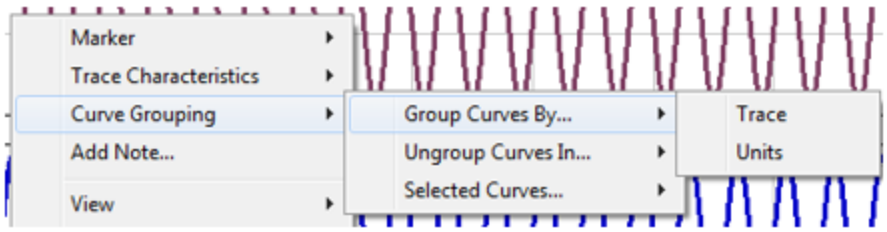


The curves (curve names) and their stack associations (stack names) appear under the **Curve Grouping Strategy** property in the **Stacked** tab. The **Curve Grouping Strategy** property has a drop-down list with three options:

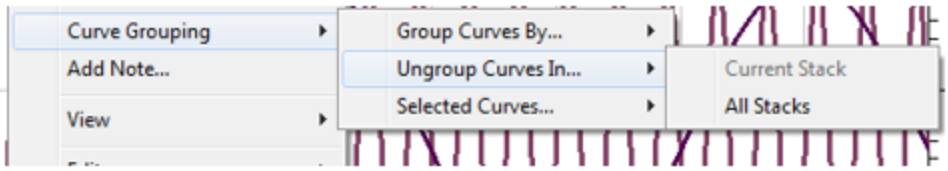


- **Single** – All curves are ungrouped and each curve is in its own stack.
- **By Trace** – All curves are grouped by their trace.
- **By Units** – All curves are grouped by their unit type.

The same options are available when you right-click a plot, as shown below.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Creating a Rectangular Stacked Plot

1. Select **Twin Builder** > **Results** > **Create Standard Report** > **Rectangular Stacked Plot**. The **Report** dialog box appears.
2. In the **Context** section, make selections from the following field or fields, depending on the design and solution type.
   - **Solution** – Lists the available solutions, whether sweeps or adaptive passes.
   - **Domain** – Whether this field appears, and the domains listed depend on the Solution type and the *<type>* selected. For modal and terminal solution data reports, the domain can be **Sweep** or **Time**.
   - **Geometry** – For field and radiated field reports, this applies the quantity to a geometry or radiated field setup.
3. Under the **Trace** tab, **Y** component section, specify the information to plot along the Y-axis:
   - **Category** – Select the type of information to plot. The category you select provides the default plot name.
   - **Quantity** – Select the value to plot.
   - **Function** – Select the mathematical function of the quantity to plot.
   - **Value** – Displays the currently specified quantity and function. You can edit this field directly.

     > **Note:**
     >
     > Color shows valid expression.

   - **Range Function** – Click to open the **Set Range Function** dialog box. This applies currently specified quantity and function.
4. On the **Trace** tab, **X** (primary sweep) line, specify the quantity to plot along the X-axis in one of the following ways:
   - Select the sweep variable to use from the drop-down list.
   - If sweeps are available, click ... to select particular sweeps. The quantity is plotted against the primary sweep variable listed.
5. On the **Families** tab, confirm or modify the sweep variables to plot.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

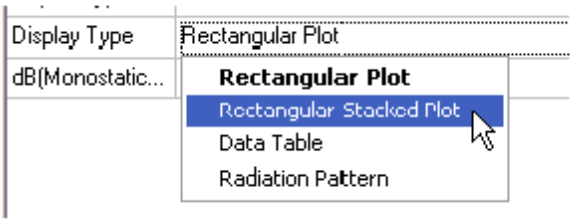
6. Click **New Report**.

   This creates a new report in Project tree, displays the report with the defined trace, and enables the **Add Trace** button on the **Report** dialog box. The default name is based on the report category you selected, (for example, S Parameter Plot *n* or rE Plot *n*). You can edit the plot names in the Project tree and the plot header text in the report synchronizes.

   The function of the selected quantity is plotted against the swept variable values or quantities you specified on an x-y graph. The plot is listed under **Results** in the Project tree and the traces appear under the plot. When you select the traces or plots, their properties display in the **Properties** window. You can edit these properties directly to modify the plot.

7. Optionally, add another trace to the plot by following the procedure above, using **Add Trace** rather than **New Report**.

   You can also modify the display type of an existing plot from the **Properties** dialog box for that plot. Select the **Report** icon in the Project tree to display the **Properties** dialog box. Select the **Display Type** field to display a menu with selections available for that plot.



   Once you make a selection, the plot display updates for the current selection.

Multiple Curves in a Stack in Cartesian Stacked Plots

Now multiple curves can be grouped together in a single stack. There are two ways in which grouping of curves can be achieved:

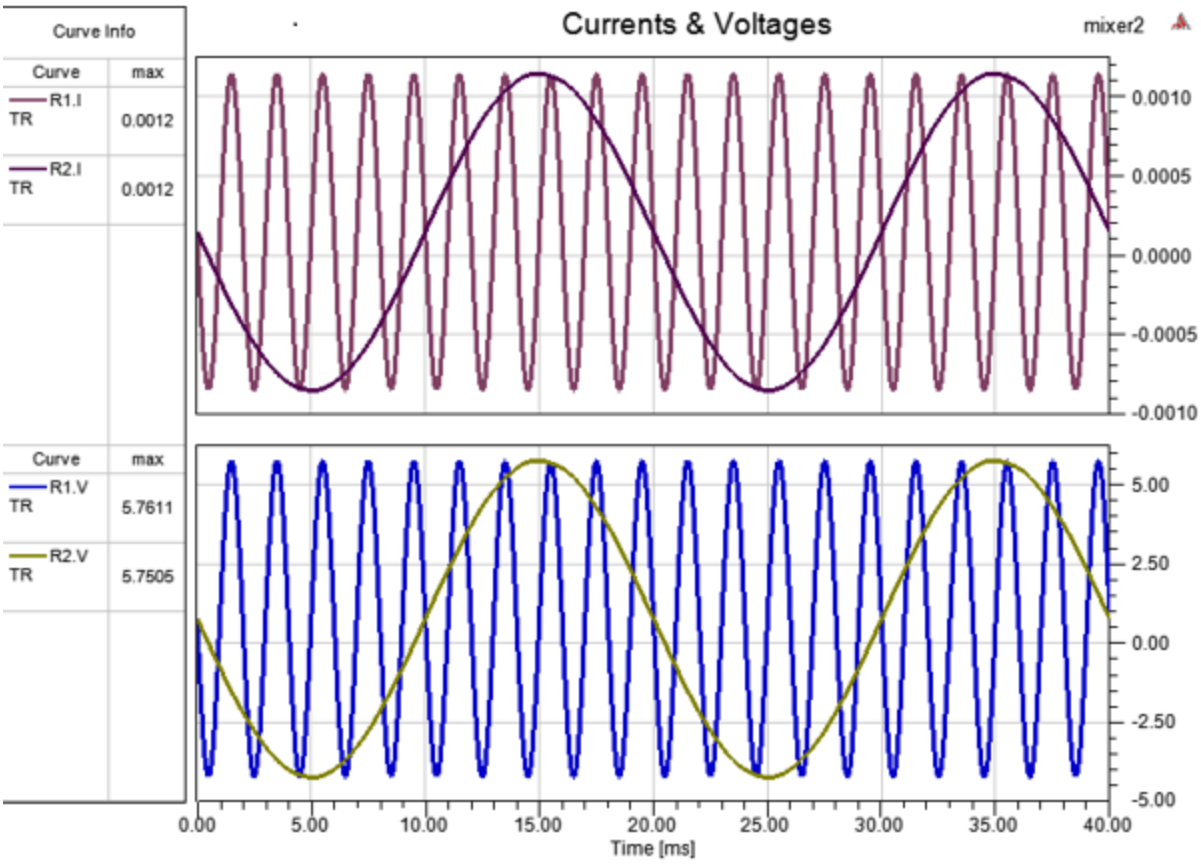
- Automatic Grouping
- Manual Grouping

Automatic Grouping

The automatic grouping feature for stacked plots groups the curves generated by a report into stacks based on a similarity metric: trace name or units. This is controlled by the **Curve Grouping Strategy** property which can also be used to turn grouping off when the single strategy is selected. New curves respect the selected grouping.

The following image shows the same stacked plot with curves grouped by units; that is, current and voltage curves are grouped together in their own stacks:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

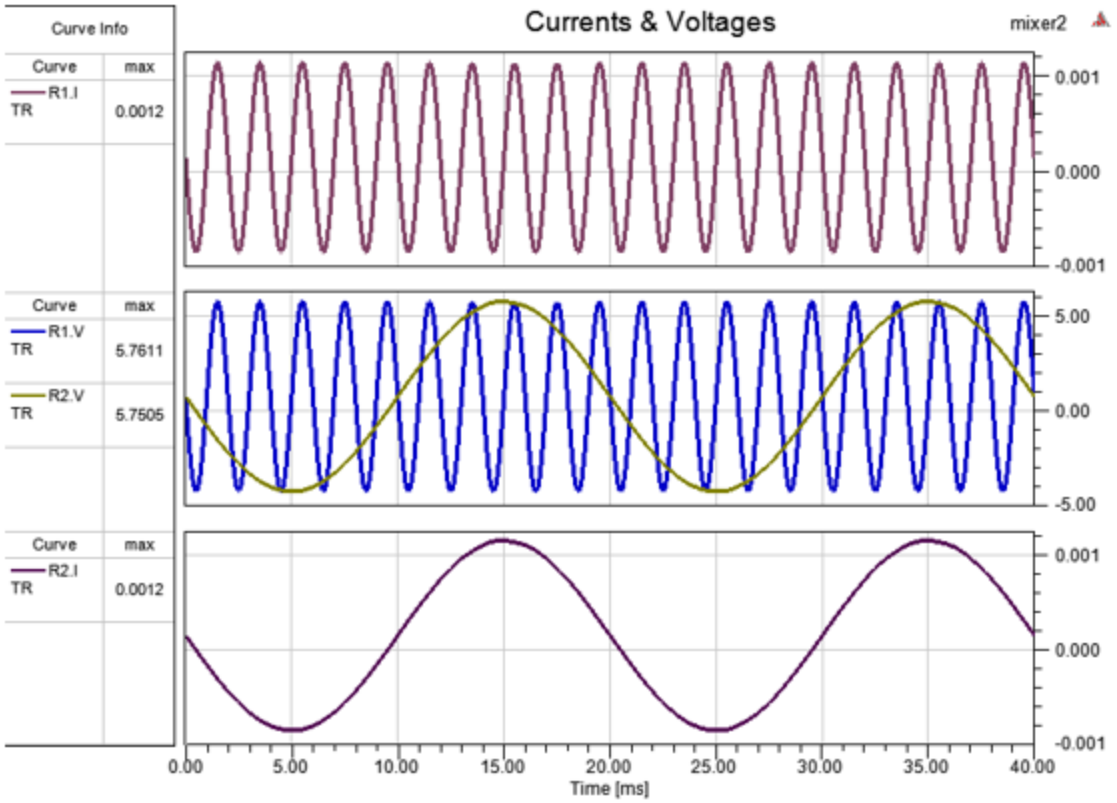


Manual Grouping

Manual grouping of curves allows arbitrary grouping of curves. When you manually group curves, the **Curve Grouping Strategy** property becomes **Custom**. Each new curve goes into a new stack.

The following image shows an example of manual grouping, where two *voltage* curves are grouped together while *current* curves are in their own stacks:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

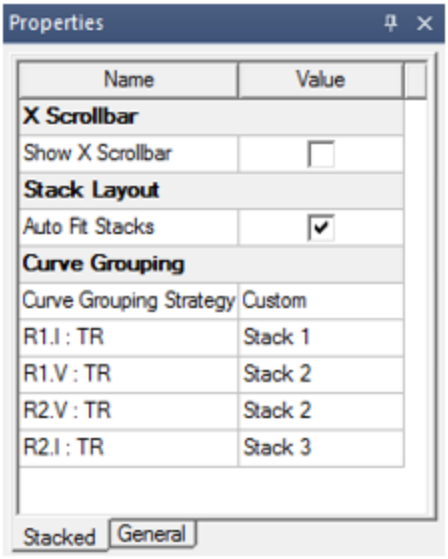


You can perform manual grouping in two ways:

- Through the **Stacked** tab of the **Property** window.
- Through the right-click context menu.

Performing manual grouping sets the **Curve Grouping Strategy** property in **Stacked** tab in the **Properties** window as **Custom** as shown below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

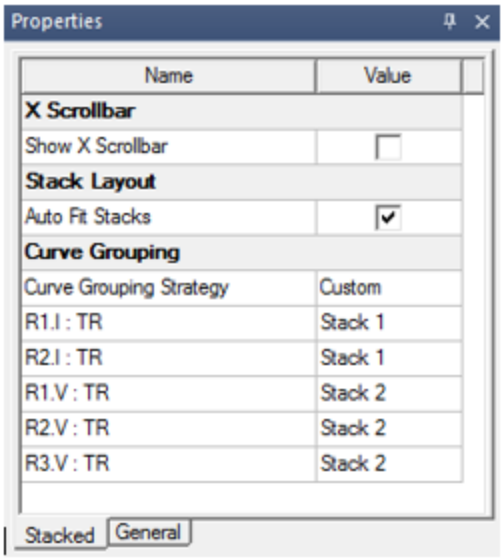


Manual Grouping through the Stacked Property Tab

To perform manual grouping though the **Properties** window, follow these steps:

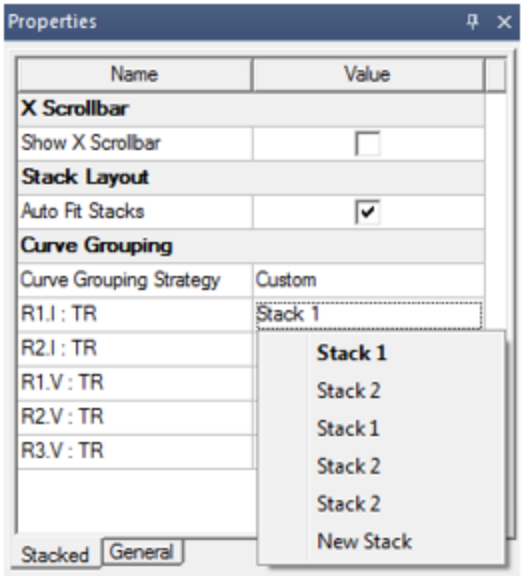
1. Click in an empty area on the plot.
2. In the **Properties** window, select the **Stacked** tab.

   The **Stacked** tab shows the curve name and its stack *<number>* association, as shown in following image:



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Each **Stack** <*number*> field has a drop-down list as shown in following image:



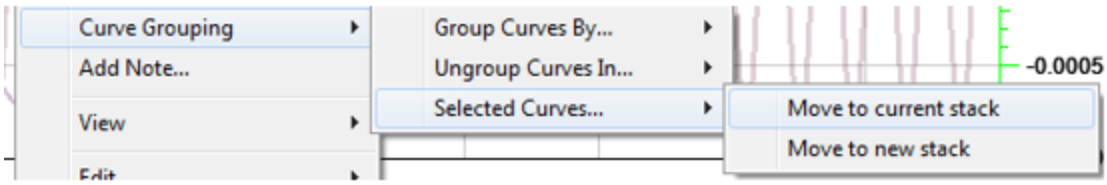
3. To change the stack of a curve, choose a different stack in this drop down menu.

> **Note:**
>
> The **New Stack** option is only available when there are multiple curves in the stack, which is the current stack of the curve.
>
> When you choose **New Stack**, the curve moves to a new stack. This lets you ungroup curves in a stack.

Manual Grouping Through the Context Menu

To manually move curves to another stack:

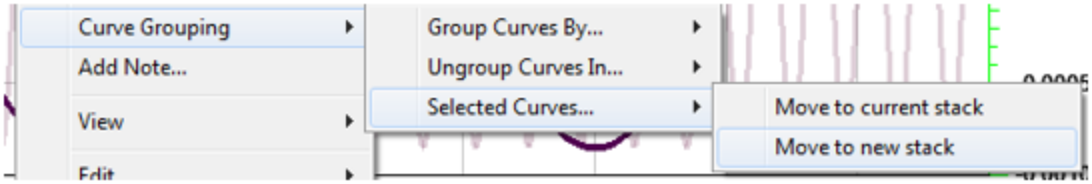
1. Select the curves.
2. Right-click within the target stack.
3. Select **Curve Grouping** > **Selected Curves** > **Move to current stack** as shown below:



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

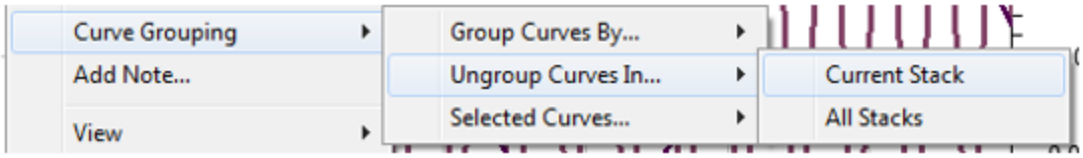
To manually move curves to a new stack:

1. Select the curves.
2. Right-click within the plot.
3. Choose **Curve Grouping** > **Selected Curves** > **Move to new stack** as shown below:



To ungroup all curves in a stack:

1. Right-click within the stack.
2. Choose **Curve Grouping** > **Ungroup Curves In** > **Current Stack** as shown below.



Legend Optimizations

The legend gets additional layout changes to optimize space usage. When there is single curve per stack, the legend shows the curve names in columns and trace characteristics in rows to optimize use of available vertical space.

If there are multiple curves in any stack, then curve names are shown in rows and trace characteristics are shown in columns. Since automatic grouping tends to increase the number of curves rapidly for the same selected trace characteristics, this keeps the plot area maximized under such growth scenarios.

Y-Markers

If there is a Y-Marker in a stack which has curves, then:

- If all those curves move together to a new stack, the Y-Marker migrates as well.
- If only a subset of the curves move to a new stack, the Y-Marker remains in the stack with the remaining curves.
- If all the curves move to different stacks, the Y-Marker is dropped.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Sweeping a Variable

Working with Traces

Adding a Characteristic to a Trace

Delta Markers in 2D Reports

Modifying Background Properties of a Report

Discarding Report Values Below a Specified Threshold

Setting Report2D options

Y Markers in stacked XY plots

Creating a Rectangular Contour Plot

To display a contour chart of simulation results after a successful analysis:

1. Select **Twin Builder** > **Results** > **Create Standard Report** > **Rectangular Contour Plot**. The **Report** dialog box appears.
2. Select the output quantities of interest and click **Add Trace**.
3. Click **Done**. A Report window opens to display a graph of the analysis results.

Related Topics

Sweeping a Variable

Working with Traces

Delta Markers in 2D Reports

Modifying Background Properties of a Report

Creating a Bode Plot

A **Bode Plot** typically is used to display the frequency response of a linear, time-invariant system. Bode plots consist of a *magnitude* plot stacked above a *phase* plot. The stacked plots share a common X-axis scale.

- The Bode *magnitude* plot is a graph of log magnitude (gain) versus log frequency. The gain values by default are expressed in decibels.
- The Bode *phase* plot is a graph of phase angle versus log frequency. The phase angle by default is expressed in degrees.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> You must define an AC analysis setup before attempting to create a Bode plot.

To create a Bode plot:

1. Select **Twin Builder** > **Results** > **Create Standard Report** > **Bode Plot**. The **Report** dialog box appears.
2. In the **Context** section, make selections from the following field or fields, depending on the design and solution type.
   - **Solution** – Lists the available solutions, whether sweeps or adaptive passes. Select **AC:AC**.
   - **Domain** – Select **Sweep**.
   - **Optimetrics setup** – Lists all defined Optimetrics analysis setups.
   - **Select Quantities** – Click to open the **Select Quantities** dialog box in which you can select additional quantities for plotting.
3. On the **Trace** tab, specify the information to plot:
   a. For the **Primary Sweep** field, specify the quantity to plot along the X-axis in one of the following ways:
      - Select a sweep variable **F** (frequency) from the drop-down list.
      - If sweeps are available, click to select sweeps. The quantity is plotted against the primary sweep variable listed.
   b. For the **X** field, select the **Default** check box to ensure that frequency is plotted along the X-axis.
   c. In the **Category** list, select the type of information to plot along the Y-axis.
   d. In the **Quantity** list, select the value to plot along the Y-axis.
   e. Optionally, in the **Function** list, select the mathematical function to be applied to the quantity to plot.

      The **Signal** field displays the currently specified **Quantity** and **Function** that will be plotted along the Y-axis. You can edit this field directly.

      > **Note:**
      >
      > The text color shows whether the expression is valid (blue for valid expressions; red for invalid).

   f. **Range Function** – Click to open the **Set Range Function** dialog box. This applies the currently specified quantity and function.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Optionally, on the **Families** tab, confirm or modify the sweep variables to plot.
5. Click **New Report**.

   This creates a new Bode plot in the Project tree, displays the plot with the defined trace, and enables the **Add Trace** button on the **Report** dialog box.

   The function of the selected quantity is plotted against the swept variable values or quantities you specified on x-y *magnitude* and *phase* graphs. The plot is listed under **Results** in the Project tree and the traces are listed under the plot. When you select the traces or plots, their properties display in the **Properties** dialog box. You can edit these properties directly to modify the plot.

6. Optionally, add another trace to the plot by following the procedure above, using **Add Trace** rather than **New Report**.

Related Topics

Sweeping a Variable

Working with Traces

Delta Markers in 2D Reports

Modifying Background Properties of a Report

Creating a Nyquist Plot

A Nyquist plot typically is used to assess the stability of a closed-loop feedback system by plotting the gain and phase of a system's frequency response using polar coordinates.

> **Note:**
>
> An AC analysis setup must be defined before attempting to create a Nyquist plot.

Follow this procedure to create a Nyquist plot:

Select **Twin Builder** > **Results** > **Create Standard Report** > **Nyquist Plot**. The **Report** dialog box appears.

2. In the **Context** section, make selections from the following field or fields, depending on the design and solution type:
   - **Solution** – Lists the available solutions. For Nyquist Plots, select **AC:AC**.
   - **Domain** – Select **Sweep**.
   - **Optimetrics setup** – Lists all defined Optimetrics analysis setups.
   - **Select Quantities** – Click to open the **Select Quantities** dialog box in which you can select additional quantities for plotting.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. On the **Trace** tab, specify the information to plot:
   a. For the **Primary Sweep** field, the default sweep is **F** (frequency).
   b. In the **Category** list, select the type of information to plot.
   c. In the **Quantity** list, select the value to plot. Typically this is the output signal you want to investigate.
   d. Optionally, in the **Function** list, select the mathematical function to apply to the quantity to plot.

      The **Signal** field displays the currently specified **Quantity** and **Function** that will be plotted along the Y-axis. You can edit this field directly.

> **Note:**
>
> The text color shows whether the expression is valid (blue for valid expressions; red for invalid).

   e. Optionally, click **Range Function** to open the **Set Range Function** dialog box, in which you can select a range function to apply to the currently specified quantity and function.
   f. For the **Base** field, choose the desired quantity to plot from the **Select Base Component** dialog box.
4. Optionally, on the **Families** tab, confirm or modify the sweep variables that will be plotted.
5. Click **New Report**.

   This creates a new Nyquist plot in the Project tree, displays the plot with the defined trace, and enables the **Add Trace** button on the **Report** dialog box.

   The function of the selected quantity is plotted against the swept variable values or quantities you specified on a polar Nyquist graph. The plot is listed under **Results** in the Project tree and the traces appear under the plot. When you select the traces or plots, their properties display in the **Properties** dialog box. You can edit these properties directly to modify the plot.
6. Optionally, add another trace to the plot by following the procedure above, using **Add Trace** rather than **New Report**.

Related Topics

Sweeping a Variable

Working with Traces

Delta Markers in 2D Reports

Modifying Background Properties of a Report

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Creating a Digital Plot

A digital plot is a 2D, x-y graph of simulation results.

> **Note:**
>
> You must define a solution setup before creating a digital plot.

To create a digital plot:

1. Select **Twin Builder** > **Results** > **Create Standard Report** > **Digital Plot**. The **Report** dialog box appears.
2. In the **Context** section, make selections from the following field or fields, depending on the design and solution type.
   - **Solution** – Lists the available solution types.
   - **Domain** – Whether this field appears, and the domains listed depend on the solution type and the <*type*> selected. The domain can be **Sweep**, **Spectral**, or **Time**.

     If you select **Spectral**, additional fields display in which you can make settings for plotting spectral domain data.

   - **Optimetrics setup** – Lists all defined Optimetrics analysis setups.
   - **Select Quantities** – Click to open the **Select Quantities** dialog box in which you can select additional quantities for plotting.
3. Under the **Trace** tab, **Y** component section, specify the information to plot along the Y-axis:
   - **Category** – Select the type of information to plot.
   - **Quantity** – Select the value to plot.
   - **Function** – Select the mathematical function of the quantity to plot.
   - Value field displays the currently specified quantity and function. You can edit this field directly.

     > **Note:**
     >
     > The text color shows whether the expression is valid (blue for valid expressions; red for invalid).

Creating a Plot-On-Schematic

After Twin Builder generates a solution, you can analyze your solution data by creating a 2D report that displays the relationship between a design's values and the corresponding analysis results. One convenient way to generate a 2D report is to create a **Plot-On-Schematic**, which

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

generates a standard 2D rectangular plot *associated with the selected component*, displaying the plot alongside the component in the Schematic editor.

You can create a plot-on-schematic using either **Probe**, **Quick Probe**, or **Probe Selection** as follows:

Probe Command

1. Right-click the desired component and select **Probe**.

   A list of quantities that can be plotted, such as voltage across the component, appears. The list of quantities is determined by the **Probe Quantities & Signals** settings on the **Twin Builder** tab of the **Schematic Options** dialog box, and any output that is selected in the **Defined Outputs** set in the **Output** dialog box. By default, the probe settings for the Twin Builder schematic have output quantities turned-on.

2. Select the quantity you want to plot from the list. The probed quantity is plotted and placed on the schematic next to the selected component.
   - The plot is associated with the component so that, as the component moves, the plot moves with it.
   - The plot can also be relocated directly by selecting and dragging it.
   - The plot can also be resized by dragging any of its handles.
   - Text is resized with the plot.
   - The created plot also appears in the **Results** folder in the Project tree. Double-click the plot listing in the tree to open it in a new window.

     > **Note:**
     >
     > If you change the display type to **Data Table**, the on-sheet plot will go blank (data tables cannot be shown on the schematic). Instead, a separate **Data Table** dialog box appears.

   - Each component/net/port can have only one on-sheet plot associated with it. You can have multiple plots on the same schematic, but only one such plot for each component.
   - If you double-click a plot in the schematic, the plot becomes fully editable. You can move the legend bar around, change the color of traces, change scaling for X- and Y-axis – every operation you can normally perform in a regular plot window. You can also add plot markers. For more information see ***Modifying Reports***. When you click outside of the plot, it reverts back to being another schematic object.
   - Plots are dynamically updated when you modify an analysis.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Quick Probe Command

Quick Probe command operates similarly to **Probe**. However, instead of offering a list of choices to plot, it plots the first quantity for the selected component or net as defined in the list of quantities for the **Probe** command. Derivative quantities (for example, dV) are not available for the **Quick Probe** command.

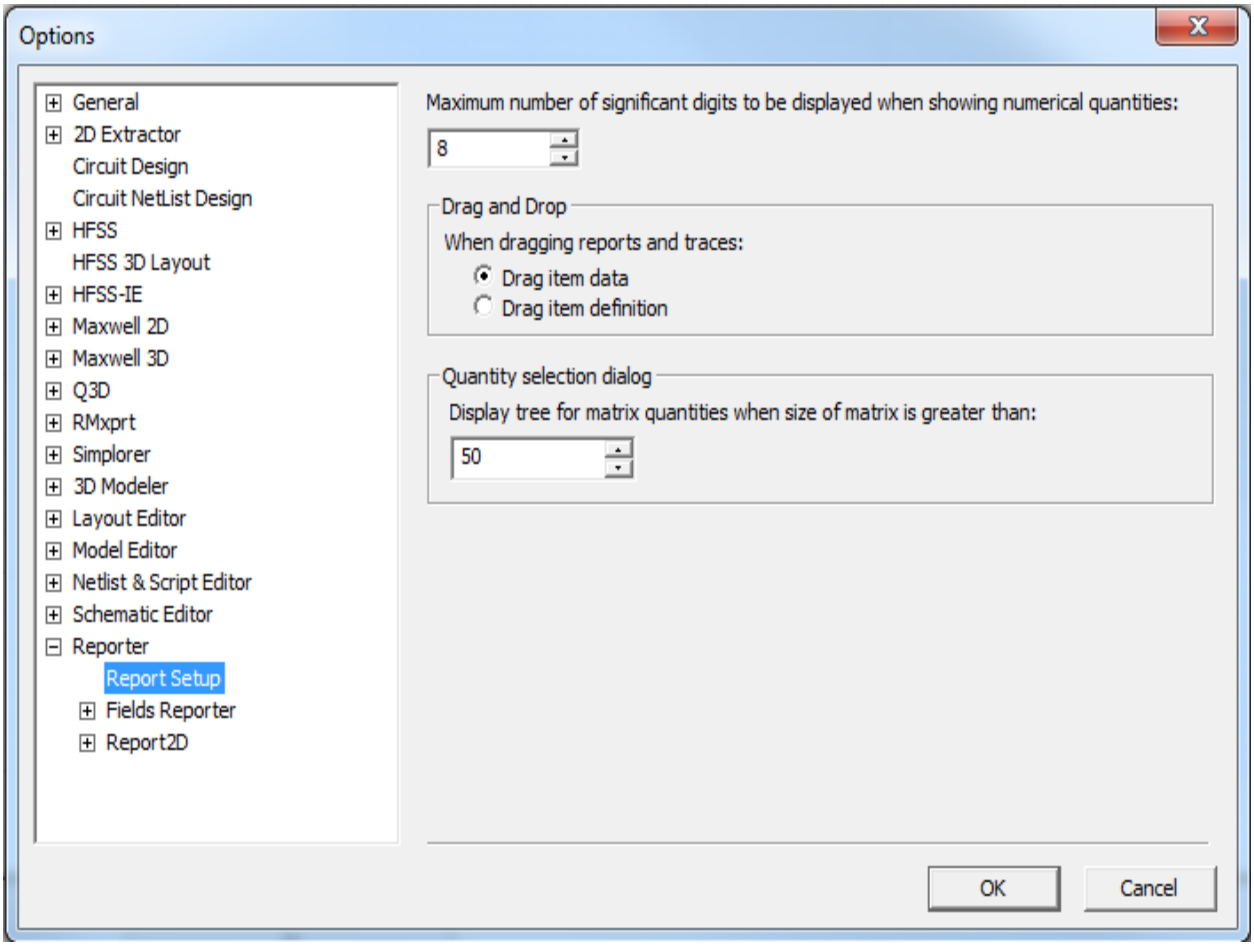
Probe Selection Command

The **Schematic** > **Probe Selection** command operates like the **Quick Probe** command: plotting the first quantity for the selected component or net.

Setting Report Setup Options

To set report setup options in Twin Builder:

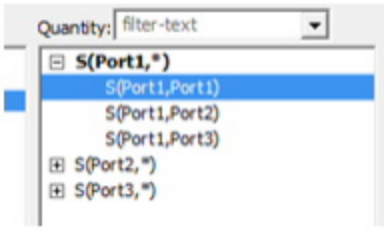
1. Click **Tools** > **Options** > **General Options** > **Reporter** > **Report Setup**.
2. Enter a number in the **Maximum number of significant digits to be displayed when showing numerical quantities** field.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Specify the drag and drop behavior with one of the following options:
   - **Drag item data**
   - **Drag item definition**
4. The **Quantity selection** dialog box specifies the matrix size for using a tree display for matrix quantities in the **Reporter** dialog box. This is helpful when dealing with larger matrices. The default is **50**. When the number of matrix elements is larger than the number, the **Quantities** field uses a tree structure to divide matrix quantities into groups by their first element name. The initial display shows groups, without initially listing the group members.

   A folder node is not selectable. Click a folder node to expand or collapse it. When any of a folder's child nodes is selected, it becomes bolded.



   Click a quantity node (tree leaf node). Shift and Ctrl keys apply only to multiple selection dialog boxes:

- Without Shift and Ctrl key – Select the quantity and deselect all previous selected quantities.
- Only Ctrl key down – Toggle the selection of the quantity. No effect on other selected quantities.
- Only Shift key down – Do a range selection, and deselect any selected quantity that is outside of the range.
- Both Shift and Ctrl key down – Do a range selection, but don't deselect any selected quantity.
- Ctrl+a – Select all quantities in a multiple selections dialog box.

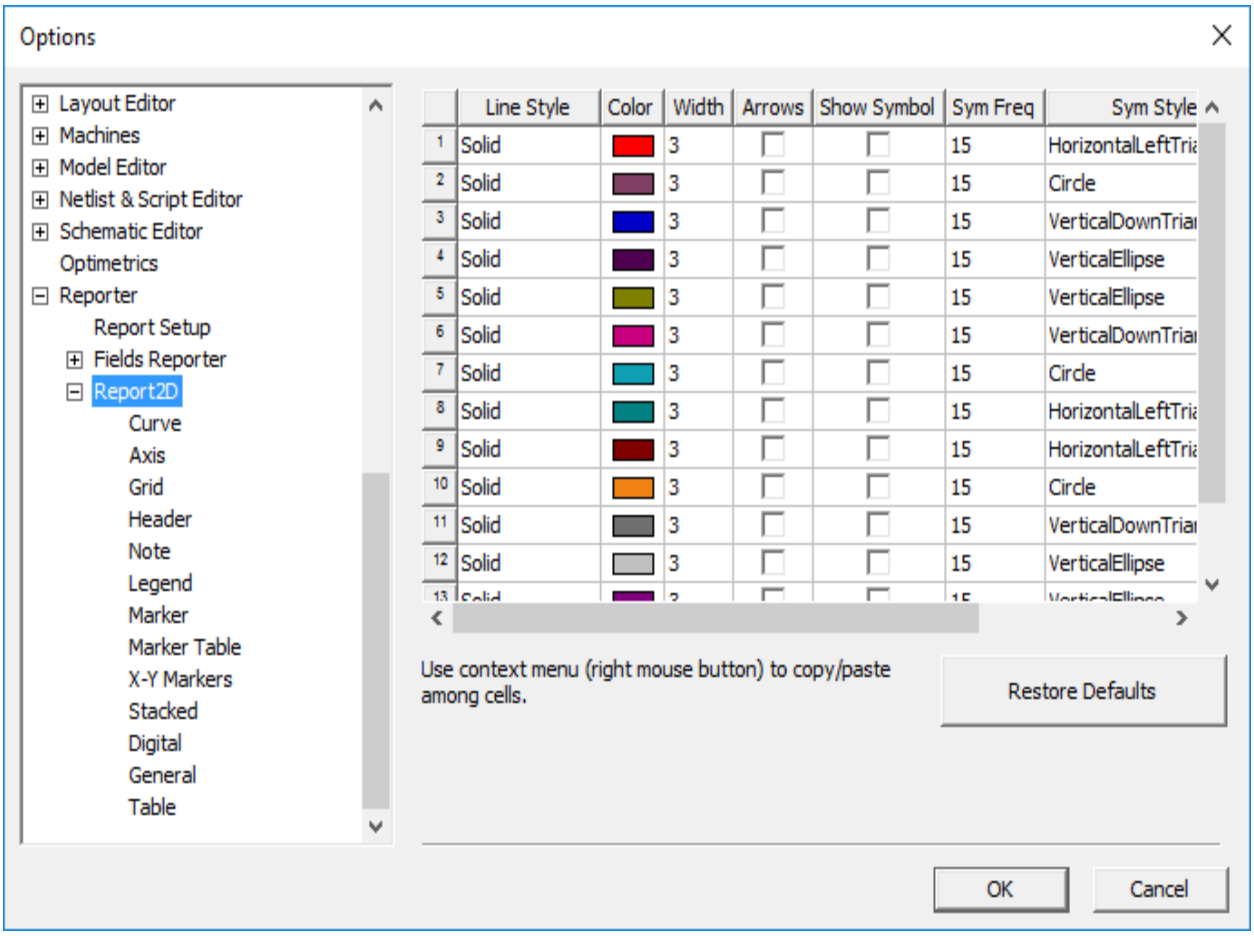
  Range selection: Select quantity nodes between the last mouse clicked quantity node and the newly clicked on quantity node. Folder nodes in between won't be selected but their children are selected. Those folder nodes display as bold.

Setting Report2D Options

Follow this procedure to set Report2D options in Twin Builder:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Tools** > **Options** > **General Options** > **Reporter** > **Report2D**.



2. The **Report2D** section has the following options:
   - **Curve**
   - **Axis**
   - **Grid**
   - **Header**
   - **Note**
   - **Legend**
   - **Marker**
   - **Marker Table**
   - **X/Y Markers**
   - **Stacked**
   - **Digital**
   - **General**
   - **Table**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. For properties controlled by check boxes, you can set values for all curves. Click the column header cell that contains the property title. Right-click a text field cell to cut, copy, and paste values. Right-click a menu cell to copy and paste entire rows.
4. Click **Restore Defaults** to revert your changes to their default values.
5. Click **OK** to save your changes and close the **Options** dialog box.

Report 2D Options: Curve Tab

Set these options on the **Curve** tab of the **Report2D Options** dialog box.

- **Line Style** – Select a combination of solid, dotted, or dashed lines.
- **Color** – Double-click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Width** – Set the line width by editing the real value in the text field.
- **Arrows** – Select this check box to use arrows on the curve ends.
- **Symbol** – Select this check box to have symbols mark the locations of data points on the curve.
- **Sym Freq** – Set the symbol frequency by editing the integer value in the text field.
- **Sym Style** – Select the symbol to display for the designated data points.
- **Fill Sym** – Select this check box to set the symbol display as a solid.
- **Sym Color** – Set the symbol color. Double-click to open the **Color** dialog box, select a default or custom color, and click **OK**.

Report2D Options: Axis Tab

You can set these options on the **Axis** tab of the **Report2D Options** dialog box.

- **Axis Name** – The axis to which the following options refer. Read-only.
- **Color** – Double-click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Font Color** – Set the font color of the axis. Double-click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Edit Font** – Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.
- **Min Gutter %** – X-axis default is **0**; all the Y-axes get a default of **5%**.
- **Font Description** – Display the font style you selected in the **Edit Font** column.

Report2D Options: Grid Tab

You can set these options on the **Grid** tab of the **Report2D Options** dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Grid Name** – Lists the name or letter of the grid. Read-only.
- **Line Style** – Select a combination of solid, dotted, or dashed lines.
- **Line Color** – Set the line color. Double-click to open the **Color** dialog box, select a default or custom color, and click **OK**.

Report2D Options: Header Tab

You can set these options on the **Header** tab of the **Report2D Options** dialog box.

- **Color** – Title and subtitle. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Font** – Title and subtitle. Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.
- **Company Name** – Subtitle only. Click the cell and enter a company name. The company name appears in the upper-left corner of 2D plots.

Report2D Options: Note Tab

Set these options on the **Note** tab of the **Report2D Options** dialog box.

- **Note Color** – Set the note color. Open the **Color** dialog box, select a default or custom color, then click **OK**.
- **Note Font** – Click the cell to display the **Font** dialog box. Use this dialog box to select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.
- **Background Color** – Set the background color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Background Visibility** – Use this check box to toggle the background display for the note.
- **Border Line Color** – Set the border color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Border Visibility** – Use this check box to toggle display of the note border.
- **Border Line Width** – Set the line width by editing the real value in the text field.

Report2D Options: Legend Tab

Set these options on the **Legend** tab of the **Report2D Options** dialog box.

- **Show Trace Name** – Use this check box to toggle display of the trace name.
- **Show Solution Name** – Use this check box to toggle display of the solution name.
- **Show Variation Key** – Use this check box to toggle display of the variation key.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Highlight Curve on Hover** – Select this check box if you want a curve to be highlighted when you hover your mouse over it.
- **Text Color** – Set the text color. Click to open the **Color** dialog box, select a default or custom color, and click **OK**.
- **Text Font** – Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.
- **Background Color** – Set the background color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Border Line Color** – Set the border color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Border Line Width** – Set the border line width by editing the value in the text field.
- **Grid Color** – Set the grid color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.

Header row (in stacked plots) properties include:

- **Text Color** – Set the text color. Click to open the **Color** dialog box, select a default or custom color, and click **OK**.
- **Text Font** – Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.

Report2D Options: Marker tab

You can set these options on the **Marker** tab of the **Report2D Options** dialog box.

- **Marker color** – Set the marker color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Marker Font** – Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.
- **Arrow Direction** – Choose an arrow direction from the drop-down menu.

Report2D Options: Marker Table Tab

You can set these options on the **Marker Table** tab of the **Report2D Options** dialog box.

- **Text Color** – Set the text color. Open the **Color** dialog box, select a default or custom color, then click **OK**.
- **Text Font** – Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Background Color** – Set the background color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Border Line Color** – Set the border color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Border Line Width** – Set the line width by editing the real value in the text field.
- **Grid Color** – Set the grid color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Grid Line Width** – Set the grid line width by editing the real value in the text field.

Header row properties include:

- **Text Color** – Set the text color. Click to open the **Color** dialog box, select a default or custom color, and click **OK**.
- **Text Font** – Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.

Report 2D Options: X-Y Markers Tab

Set these options on the **X-Y Markers** tab in the **Report2D Options** dialog box.

- **Background Color** – Set the background color for the various markers. Open the **Color** dialog box, select a default or custom color, then click **OK**.
- **On-screen Intersection** – Select this check box to activate on-screen intersection.
- **Marker Font** – Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.
- **Text Color** – Set the text color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Line Color** – Set the line color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Line Style** – Select a combination of solid, dotted, or dashed lines.
- **Line Width** – Set the line width by editing the real value in the text field.
- **Show Name** – Select this check box to display marker names.
- **Snap to Vertex** – Select this check box to snap elements to a grid.

Inter marker delta properties include:

- **Show Delta** – Select this check box to display delta values.
- **Delta Font** – Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Delta Text Color** – Set the text color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Line Color** – Set the line color. Click to open the **Color** dialog box, select a default or custom color, and click **OK**.
- **Line Style** – Select a combination of solid, dotted, or dashed lines.
- **Line Width** – Set the line width by editing the real value in the text field.

Report2D Options: Stacked

Set these options on the **Stacked** tab on the **Report2D Options** dialog box.

- **Auto Fit Mode** – Use this check box to toggle auto fit mode.
- **Stack Height In Pixels** – Click the cell and enter your desired default stack height.
- **Curve Grouping Strategy** – Can be **Single**, **By Trace**, or **By Units**. **Single** means that a new stacked plot shows a single curve per stack. **By Trace** means that all curves are grouped by their trace. **By Units** means that all curves are grouped by their unit type.

  If you change the **Curve Grouping Strategy**, existing stacked plots remain unaffected. The new default will apply only to new stacked plots.

  When a project saved in versions before 19.0, and stacked plots is opened in version 19.0, **Curve Grouping Strategy** will default to **Single** and stacked plots remain unaffected.

Related Topics

Create 2D Rectangular Stacked Plots

Report2D Options: Digital Tab

Set these options on the **Digital** tab in the **Report2D Options** dialog box.

- **Digital Literal Foreground** – Set the foreground color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Expand Arrays/Records** – Use this check box to toggle full display of arrays and records.
- **Digital Stack Height in Pixels** – Click a field and edit the pixel height value for these elements:
  - **Analog**
  - **Digital**
  - **Enum**
  - **Event**
  - **Literal**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Report2D Options: General Tab

Set these options on the **General** tab of the **Report2D Options** dialog box.

- **Background Color** – Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Plot Area Color** – Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Highlight Color** – Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Accumulate Depth** – Edit the value in this field to your preferred depth.
- **Enable Y Axis Stripes** – Use this check box to toggle Y-axis display indicators.
- **Curve Tooltip Option** – Use this check box to toggle these properties:
  - **Show Trace Name**
  - **Show Variation Key**
  - **Show Solution Name**
- **Clipboard Option** - Use the drop-down menus to specify these properties:
  - **Capture Aspect Size Ratio** – This can be **As Shown** or **Full Screen**.
  - **Capture Background Color** – This can be **As Shown** or **White**.

Report2D Options: Table Tab

Set these options on the **Axis** tab of the **Report2D Options** dialog box.

- **Rows Per Page** – Click the cell and enter a value.
- **Text Color** – Set the text color. Open the **Color** dialog box, select a default or custom color, then click **OK**.
- **Text Font** – Click the cell to display the **Font** dialog box. Use this dialog box to select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.
- **Border Width** – Set the border line width by editing the real value in the text field.
- **Border Color** – Set the border color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Grid Width** – Set the grid width by editing the real value in the text field.
- **Grid Color** – Set the grid color. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Background Color** – Click to open the **Color** dialog box. Select a default or custom color and click **OK**.
- **Page Link Color** – Set the color of links. Click to open the **Color** dialog box. Select a default or custom color and click **OK**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Arrow Color** – Set the color of arrows. Click to open the **Color** dialog box, select a default or custom color, and click **OK**.

Header row properties include:

- **Text Color** – Set the text color. Click to open the **Color** dialog box, select a default or custom color, and click **OK**.
- **Text Font** – Click the cell to display the **Font** dialog box. The dialog box lets you select from a list of available fonts, styles, sizes, effects, colors, and scripts. Also contains a preview field.
- **Background Color** – Click to open the **Color** dialog box. Select a default or custom color and click **OK**.

Formatting properties include:

- **Field Width** – Click the cell and enter your desired field width, in pixels.
- **Precision** – Click the cell and enter your desired level of precision.
- **Use Scientific Notation** – Use this check box to toggle the use of scientific notation.

Copy to clipboard properties include:

- **With Header** – Select this check box to include the table header when copying the table.
- **With Tab Separator** – Select this check box to include tab separator characters when copying the table.

Working with Traces

A trace in a 2D or 3D report defines one or more curves on a graph. A trace in a data table defines part of the displayed matrix of text values.

> **Note:**
>
> Boolean Outputs that are (enum-type) can only be plotted in digital plots or in data tables.

The values used for a plot's axes (which may be X, Y, Z, phi, theta, or R, depending on the display type) can be variables in the design, such as frequency, or functions and expressions based on the design's solutions. If you have solved one or more variables at several values, you can "sweep" over some or all of those values, resulting in a curve in 2D or 3D space.

A report can include any number of traces, and for 2D rectangular and rectangular contour plots up to 20 independent y-axes. Traces appear in the Project tree under their report. They can be selected, copied and pasted. When you move the cursor over a trace in a report, it changes to the color of the trace to show that it can be selected. Similarly, when you move the cursor close to a trace, the quantity name of that trace displays in a box that appears next to the cursor. When

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

you move the mouse cursor over a trace listed in the **Curve Info** box, the corresponding trace changes color.

In general, to add a trace to a report:

1. Right-click a report in the Project window and select **Modify Report**.
2. In the **Report** dialog box, specify the Y component information.
   a. Specify the category of information you want to plot from the drop-down menu.

      The **Category** drop-down menu lists the available categories for the solution type and the current design. Selecting a category changes the quantity and function lists to represent what is available for that category.

   b. Specify the quantity you want to plot by selecting from the **Quantity** list.

      The selected quantity appears in the **Value** field, operated on any selected function.

   c. Select the function to apply to the specified quantity.
   d. The **Value** field shows the trace being readied for plotting on the Y-axis. This field is editable when the text cursor is present. You can modify the information to be plotted by typing the name of the quantity or sweep variable to plot along an axis directly in the text boxes.

      > **Note:**
      >
      > Color shows valid expression: blue for valid, red for invalid.

3. In the **Report** dialog box, specify the X-axis information (for example, primary sweep).
4. Click **Add Trace**.

A trace appears in the traces list under its report icon in the Project tree. The trace represents the function of the quantity you selected and will be plotted against other quantities or swept variable values. Selecting a trace in the Project tree displays the **Properties** window for that trace. Selecting a trace in the report or legend displays the **Properties** window for that trace.

Trace icons can be selected, copied, and pasted for their definitions or their data. They can be selected and deleted from the Project tree.

By default, the trace name is the definition (the category, quantity, and function). The trace will be visible in the report when you click **Add Trace**.

You can edit trace properties in their **Properties** windows or the **Report** dialog box. To change the name or definition of a trace, see ***Editing Trace Properties***. To edit other display properties of a trace, see ***Editing the Display Properties of Traces***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Removing Traces

Editing Trace Properties

Editing the Display Properties of Traces

Adding Characteristics to a Trace

Adding Data Markers to Traces

Copy and Paste of Report and Trace Definitions

Copy and Paste of Report and Trace Data

Delta Markers in 2D Reports

Editing Trace Properties

To edit trace properties such as the name, Y-axis association, the component definition, or the context, or the variables, select the trace in the Project tree.

To edit a trace name:

1. Select the trace in the Project tree. A docked **Properties** window for the trace appears.
2. Select the **Specify Name** box.

   This enables editing of either the **Name** field in the docked properties dialog box, or the trace label text in the Project tree. Edit this name to change the display in the legend and in the Project tree, but not the underlying Y-component definition.

To edit the Y-axis associated with the trace (2D rectangular and rectangular contour plots):

1. Select the trace in the Project tree.
2. In the docked **Properties** window for the trace, select the Y-axis to be associated with the trace from the drop-down list. Up to 20 independent Y-axes can be added to a plot.

To edit a trace component definition:

1. Select the trace in the Project tree.
2. In the docked **Properties** window for the trace, select the component field of interest, and select **Edit** from the drop-down list.

   This displays an edit component field from which you can edit the category, quantity, and function.
3. Click **OK** to apply the changes and close the dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To edit a trace Context:

1. Select the trace in the Project tree to display the docked **Properties** window.
2. In the **Properties** window, click the **Solution** field or the **Domain** field. If other selections are possible, you can select them from the drop-down list.

To edit a variable for a trace:

1. Select the trace in the Project tree to display the docked **Properties** window.
2. Under the -Variables category, on the Families line, click **Edit** to display the **Edit families** dialog box.

   From this dialog box, you can select the **Sweeps** or **Variations** option buttons.

   If other nominal values are available, click to select from a list.

Related Topics

Removing Traces

Editing the Display Properties of Traces

Adding Characteristics to a Trace

Adding Data Markers to Traces

Discarding Report Values Below a Specified Threshold

Setting Report2D options

Copy and Paste of Report and Trace Definitions

Copy and Paste of Report and Trace Data

Delta Markers in 2D Reports

Editing the Display Properties of Traces

To edit the display properties of a trace:

1. Select a trace in an open **Report** window.
2. Click the trace to view a docked **Properties** window, or double-click to open the **Properties** dialog box. The display properties window for a trace includes a **General** tab and an **Attributes** tab.

   The **General** tab properties apply to the general appearance of the plot. They include the background color, contrast color, field width, and whether to use scientific notation for

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

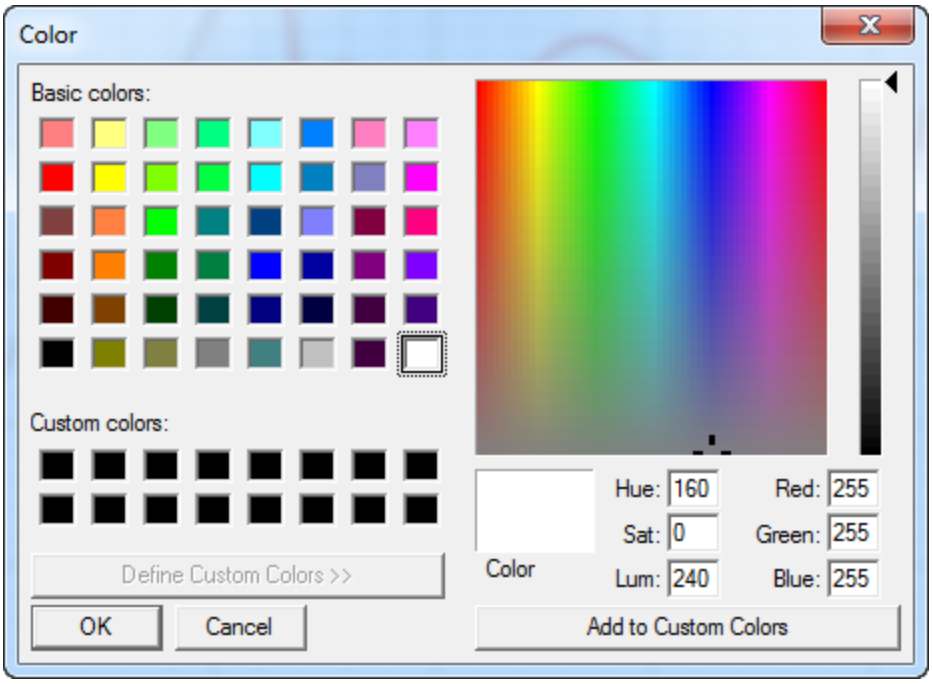
marker and delta marker displays. (X and Y notation display is set separately, in the **Axis** property tabs.)

The **Attributes** tab properties include the color, line style, line width, trace type, whether to show a symbol, symbol frequency, symbol style, whether to fill symbol, symbol color, and whether to show arrows.

> **Note:**
>
> **Box** is the default symbol; it ensures that curves with single points always appear.

The **Attributes** tab applies specifically to the trace. The defaults are set in the Report2D options. They include:

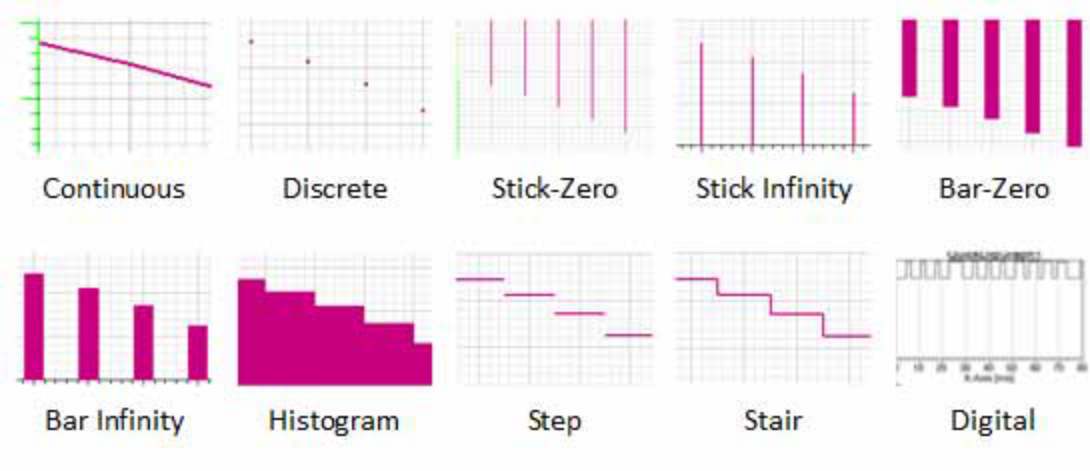
- **Name** – Not editable by selecting the trace from the report. It shows the characteristics of the trace as defined in the **Report** dialog box. To edit a trace name, see ***Editing Trace Properties***.
- **Color** – Shows the trace color. Double-click to open a **Color** dialog box. You can select from basic colors or custom colors. You can define up to 16 custom colors by selecting or by editing the hue, saturation, luminosity, and the red, green, and blue values.



- **Line style** – Use this drop-down list to select **Solid**, **Dot**, **Dash**, or **Dot-dash**.

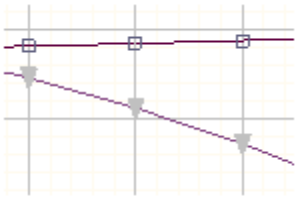
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Line width** – Edit the numeric value in this text box.
- **Trace type** – The drop-down list contains entries for **Continuous**, **Discrete**, **Bar-zero**, **Bar Infinity**, **Stick Zero**, **Stick Infinity**, **Histogram**, **Step**, **Stair**, and **Digital**.



Notice the difference between **Stair** and **Digital** is that each stair centers on a data point with transitions halfway between points, and **Digital** transitions from each data point to the next value.

The next four properties work together to define whether to show a symbol on data points, the symbol frequency, the symbol style, and whether to display the symbol as solid or hollow.



- **Show Symbol** – Whether to show a symbol at the data points on the line.
- **Symbol Frequency** – How often to show symbols on the trace.
- **Symbol Style** – Use a drop-down list to select from **Box**, **Circle**, **Vertical Ellipse**, **Horizontal Ellipse**, **Vertical Up Triangle**, **Vertical Down Triangle**, **Horizontal Left Triangle**, **Horizontal Right Triangle**.
- **Fill Symbol** – Use this check box to set the symbol display as solid or hollow.
- **Symbol Color** – Shows the symbol color. Double-click to open a **Color** dialog box. You can select from basic or custom colors. You can define up to 16 custom colors by selecting or by editing the hue, saturation, luminosity, and the red, green, and blue values.
- **Symbol Arrows** – Use this check box to show or hide arrows on the curve.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Edit the properties if needed, and click **OK** to apply the changes and close the window.

Related Topics

Editing Trace Properties

Adding Data Markers to Traces

Select **Report 2D** > **Marker** to add markers to traces.

A marker appears as **m***N* at the marked point, where *N* increments from **1** as you place additional markers. Each marker can be selected and has editable properties including name, font, background, and color.

As you place markers, one or more marker legends may appear, depending on the **View** > **Active View Visibility** settings for the legends. The main marker legend appears in the upper-left of the plot, and lists the marker names and their X and Y values in a table.

You can control the number format for the table values via the **Properties** dialog box, **General** tab. Under **Marker/Other Number** format, you can specify field width, precision, and whether to use scientific notation. This value is independent of the **Axis** tab number properties. A separate marker legend appears for delta markers, as described for the **Delta Marker** command.

When you enter marker mode, the cursor arrow is accompanied by an **m** while a circle on the selected trace shows the current position for a potential marker.

To end marker mode, right-click and select **End Marker Mode** or press Esc.

The available marker mode commands and associated icons are:

- **Add Marker** – Place a marker at an arbitrary point on a selected trace.
- **Add Delta Marker** – Enter delta marker mode, placing a circle on the selected trace. Click the trace to set an initial point; subsequent clicks on arbitrary points on the trace place additional markers until you leave marker mode. These markers have their own legend, which includes the following information for each pair of markers specified:

Name	Delta(X)	Delta(Y)	Slope(Y)	InvSlope(Y)
d(m2,m3)	0.4700	1.8319	3.8976	0.2566

- **Export Marker Table** – Export the marker table data as a CSV or TAB file.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Export Delta Marker Table** – Export the delta marker table data as a CSV or TAB file.
- **Add Maximum** – Place a marker at the maximum value on the selected trace.
- **Add Minimum** – Place a marker at the minimum value on the selected trace.
- **X Marker** – Add up to 10 movable markers at the origin of the plot with a vertical line rising from the X-axis. Each added marker has its own color and editable properties. You can change the X marker value either by dragging or editing the X marker property **Xvalue**. To move an X marker, click the X label and drag it to the desired location. The label at the bottom of the line gives the X coordinate, and flag on the vertical line identifies the Y coordinate on the trace. A **Lock Drag** trace property lets you lock the drag feature to leave the marker in place. If more than one X marker is present, marker properties include a dimension line between adjacent markers that displays the magnitude of the delta X value between the markers. Select **Clear All** to remove the X markers. **Bring X Marker into view** – Select from a list of existing X markers to bring into view. This command is enabled if an X marker is not visible in the plot.
- **Y Marker** – Add up to 10 movable markers at the origin of the plot with a horizontal line extending from the Y-axis. Each added marker has its own color and editable properties. You can change the Y marker value either by dragging or editing the Y marker property **Y value**. To move a Y marker, click the Y label and drag it to the desired location. The label at the left of the line gives the Y coordinate, and a flag on the horizontal line identifies the Y coordinate on the trace. A **Lock Drag** trace property lets you lock the drag feature to leave the marker in place. If more than one Y marker is present, marker properties include a connecting line between adjacent markers and the delta Y value between the markers. If more than one Y marker is present, marker properties include a dimension line between adjacent markers that displays the magnitude of the delta Y value between the markers. The Y markers are cleared by the **Clear All** command. For more detail on Y markers and their use, see ***Y Markers in stacked XY plots***.
- **Bring Y Marker into view** – Select from a list of existing Y markers to bring into view. This command is enabled if a Y marker is not visible in the plot.
- **Next Peak** – Move a selected marker on the next peak on a trace. You must exit marker mode and select a marker to enable this command.
- **Next Minimum** – Move a selected marker to the next minimum on a selected trace. You must exit marker mode and select a marker to enable this command.
- **Previous Peak** – Move a selected marker on the previous peak on a selected trace. You must exit marker mode and select a marker to enable this command.
- **Previous Minimum** – Place a marker on the previous minimum on a selected trace. You must exit marker mode and select a marker to enable this command.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Go to Start** (Right arrow) – Move a selected trace marker to the first data point. Enabled by leaving marker mode and selecting a marker.
- **Go to Previous** (Left arrow) – Move a selected trace marker to the previous data point.
- **Go to Next** – Move a selected trace marker to the next data point.
- **Go to End** – Move a selected trace marker to the last data point.
- **Next Curve** – Select the next curve in the report, based on the order in the trace legend.
- **Previous Curve** – Select the previous curve in the report, based on the order in the trace legend.
- **Clear All** – Clear all markers on a report.

Related Topics

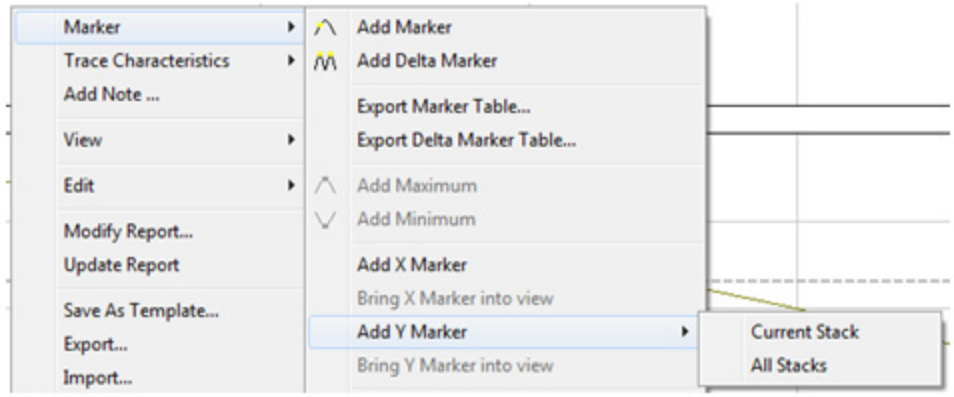
Y Markers in stacked XY plots

Y Markers in stacked XY plots

Y markers allow for easy analysis and comparison of curves at a particular y-coordinate. Y markers can be used to compare stacked curves.

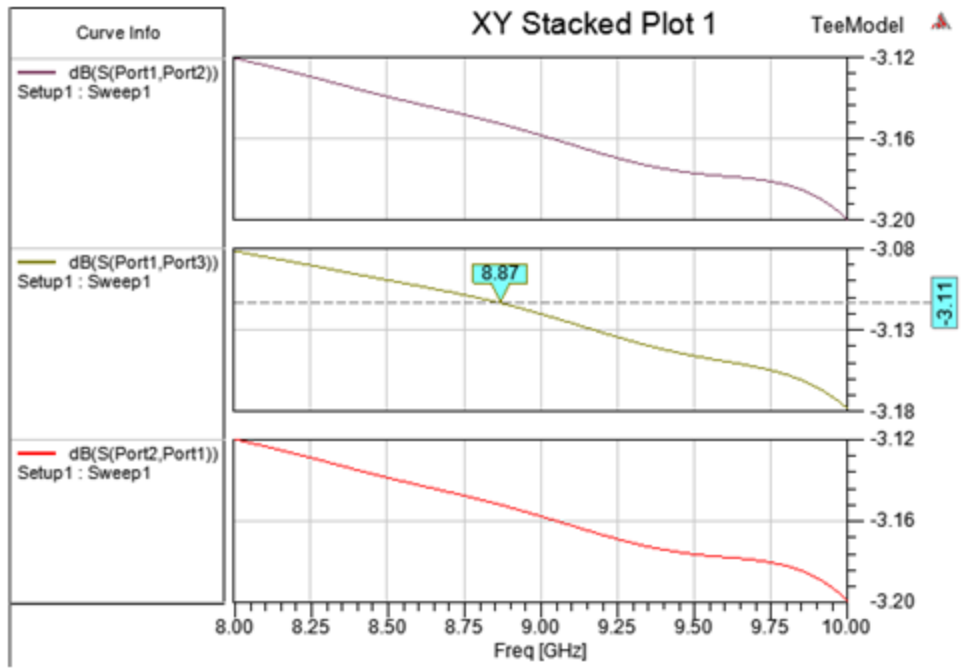
Creating Y Markers in Stacked Plots

There are two ways to create Y markers in stacked plots. You can create a Y marker for a particular stack or for all stacks. Right-click any stack to open the following menu:

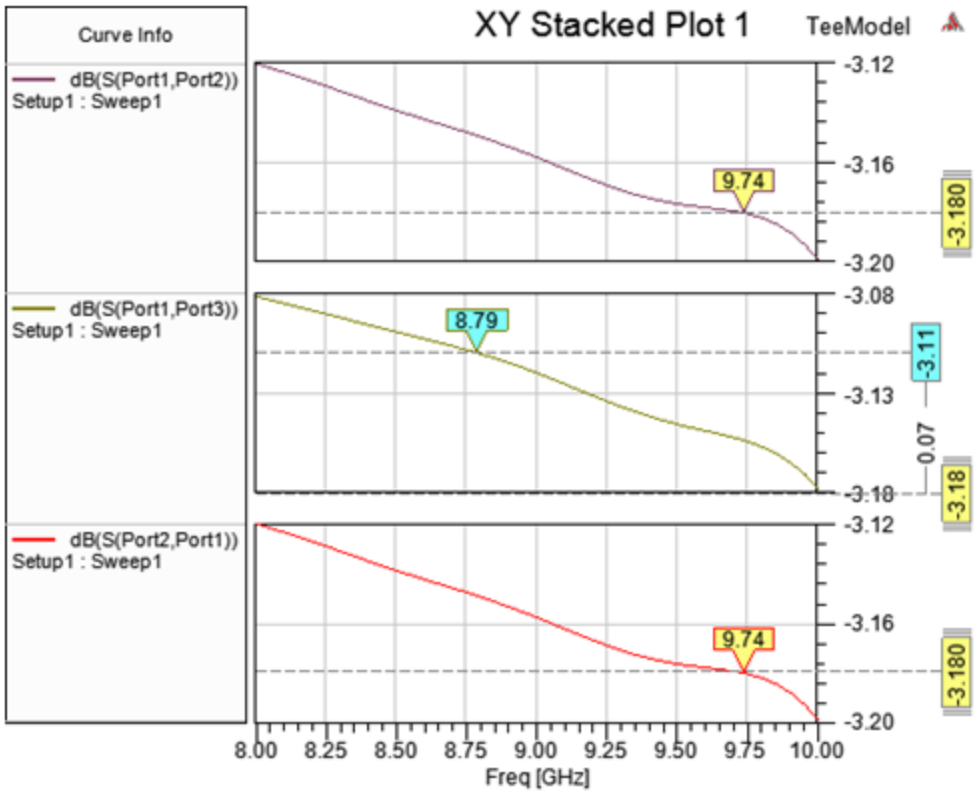


Add Y Marker > **Current Stack** creates a Y marker for the selected stack. The following figure shows that a Y marker was added to the second stack only:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Add Y Marker > **All Stacks** creates one Y marker in each stack with the same value. Initially this value is the minimum Y value of the Y ranges in all the stacks. This is shown is the figure below:

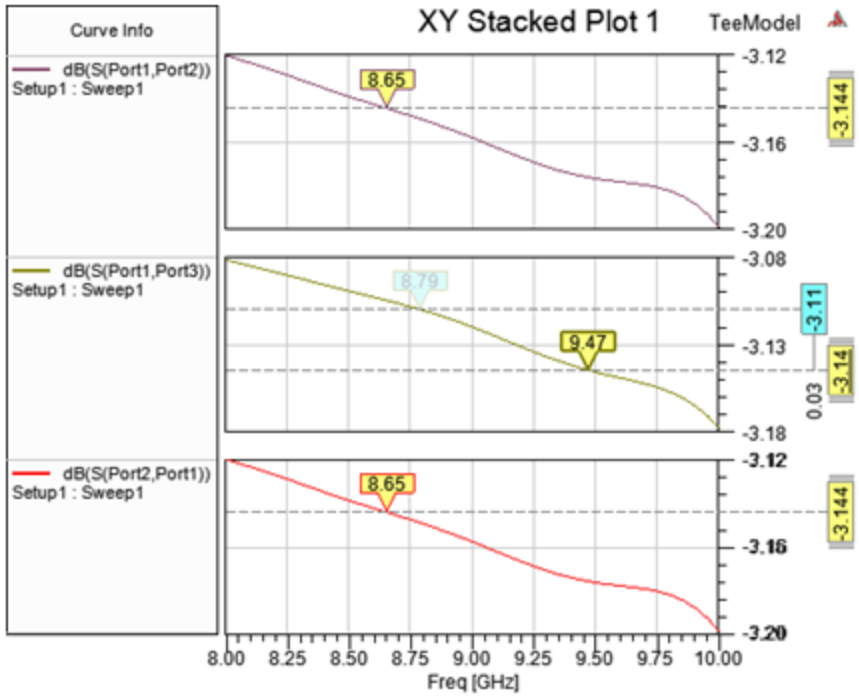


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Notice that the Y marker for all stacks has a different appearance than the Y marker for a particular stack; it has double parallel lines above and below the Y marker text box.

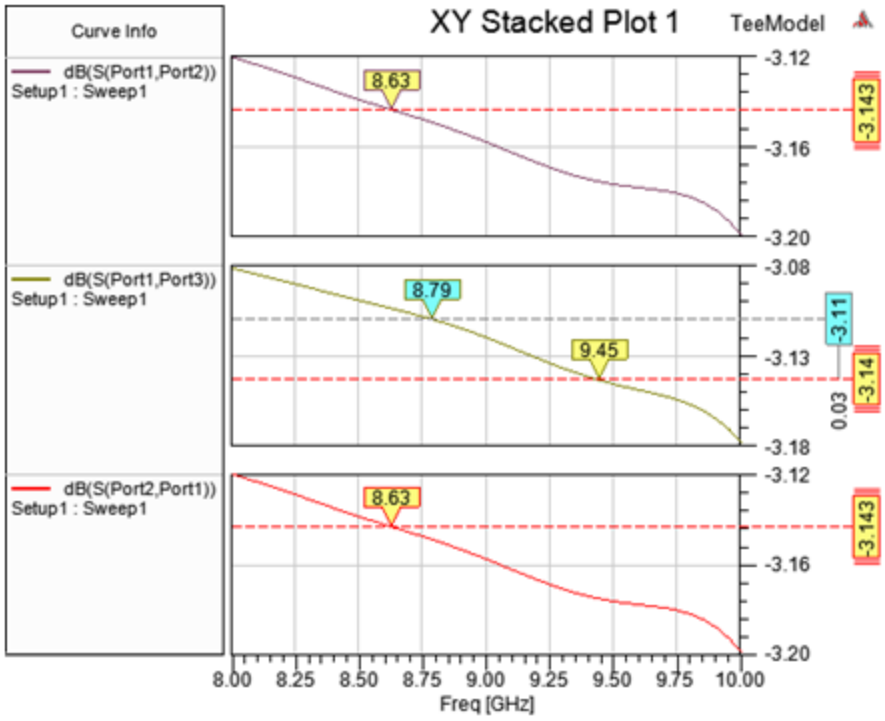
Synchronized Y Markers

All the "same" Y markers for all stacks are synchronized; that is, if one Y marker is dragged or its value is changed, all the "same" Y markers in all the stacks will also change their position. The following figure shows that when a Y marker in the bottom stack was dragged, the Y marker in top stacks moved as well:



Also, if a property of any one Y marker is changed, all the "same" Y markers show the change in property as well. For example, the following figure shows that when the line color of a Y marker in the top stack was changed to a red color, a Y marker in the bottom stack shows the same line color as well:

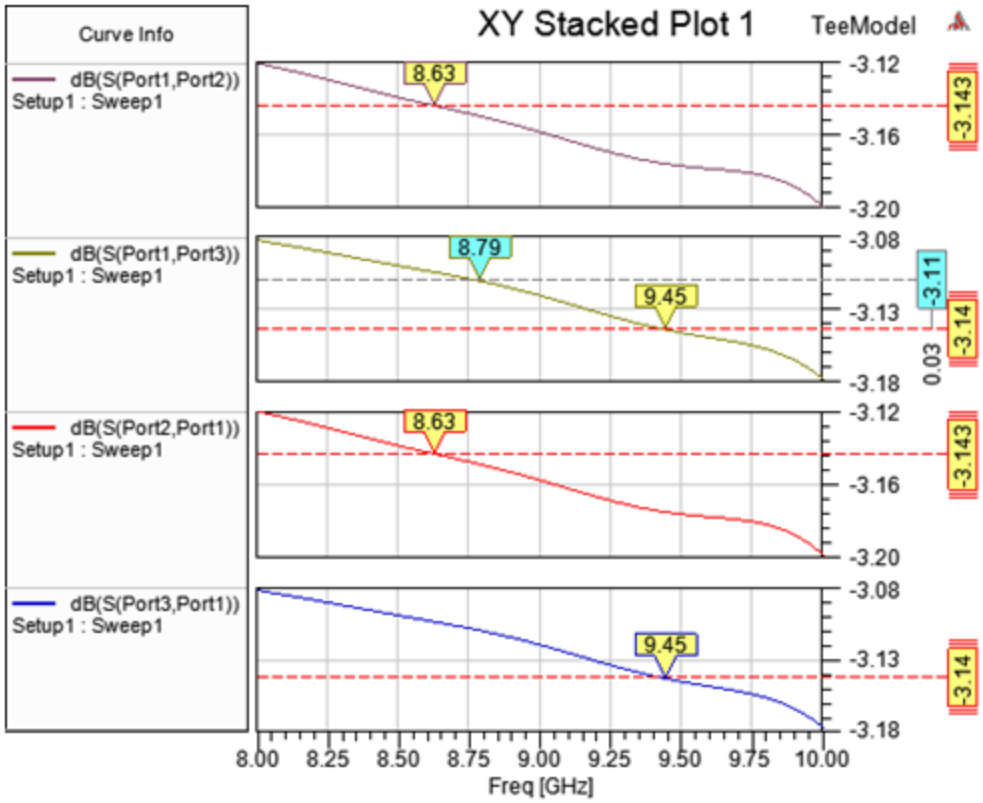
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Automatic Y Markers for the new stack

When a new curve is added to the plot, it gets all the Y markers for all stacks in other stacks, excluding the Y marker for particular stacks. The following figure shows that when the new curve "dB(S(Port3, Port1)" was added, a Y marker was added to it with value -3.14 and it has all the same properties as other "same" Y markers in other stacks:

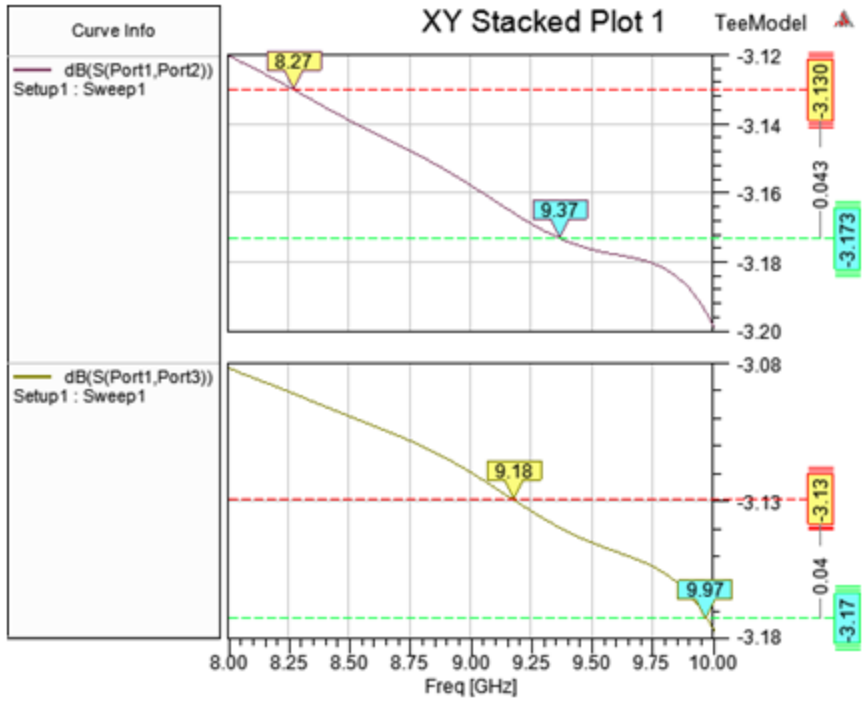
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Y Marker Delta Annotations

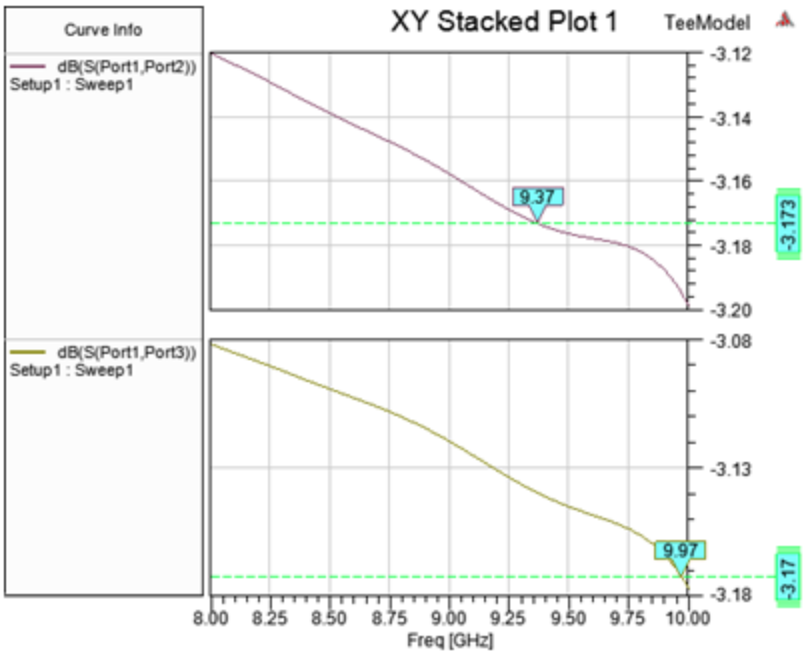
When two more Y markers are present in a stacked eye diagram, then delta annotations are shown between a pair of adjacent Y markers in all the stacks, as shown in the following figure:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Deleting a Y Marker

To delete a Y marker, select a Y marker in any stack and press Delete. This action will also delete all the corresponding Y markers in all the stacks. For example, when the Y marker with value -3.13 (red Y marker) was deleted from the bottom-most stack, all of the corresponding Y markers were also deleted:

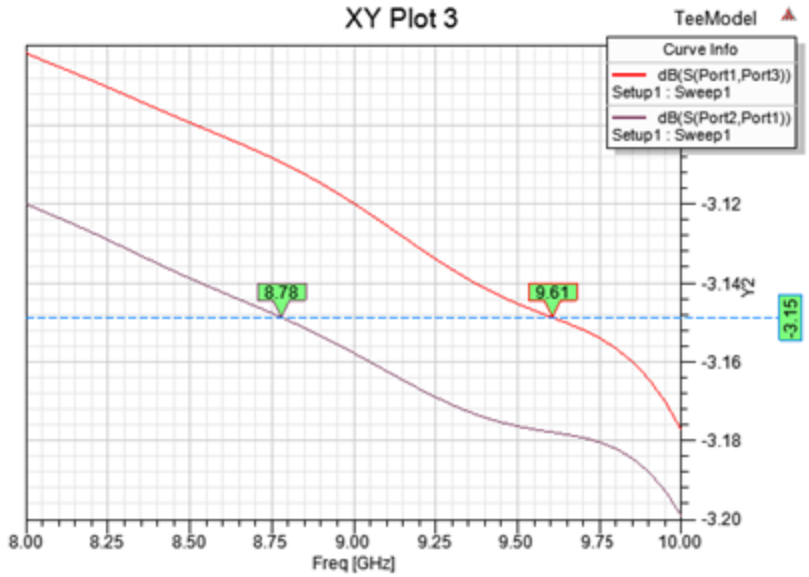


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

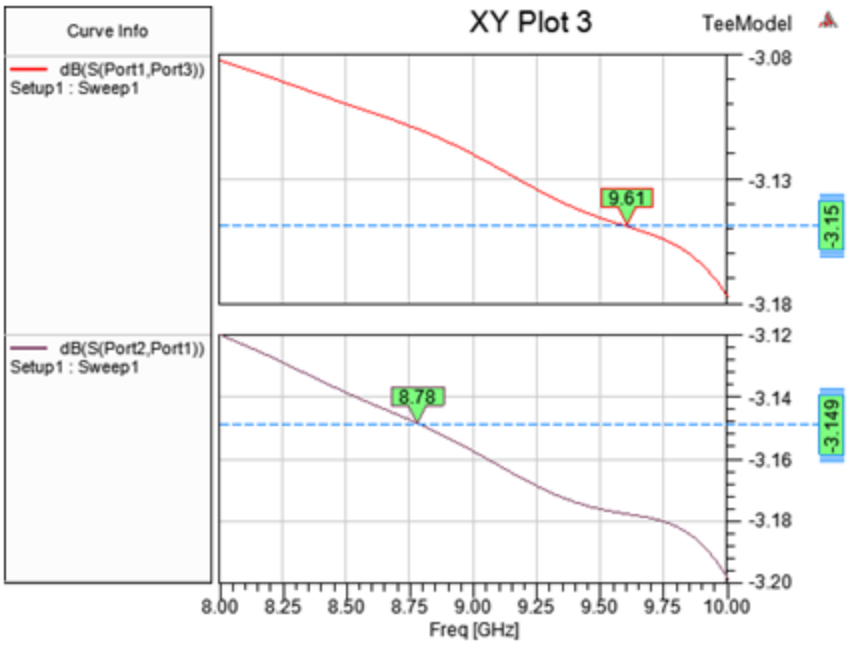
Note that on deleting a stack, Y markers in other stacks are not affected.

Converting Rectangular XY Plot to Rectangular Stacked XY Plot

The following figure shows a rectangular XY plot with two curves and a Y marker with value -3.15 (blue Y marker):

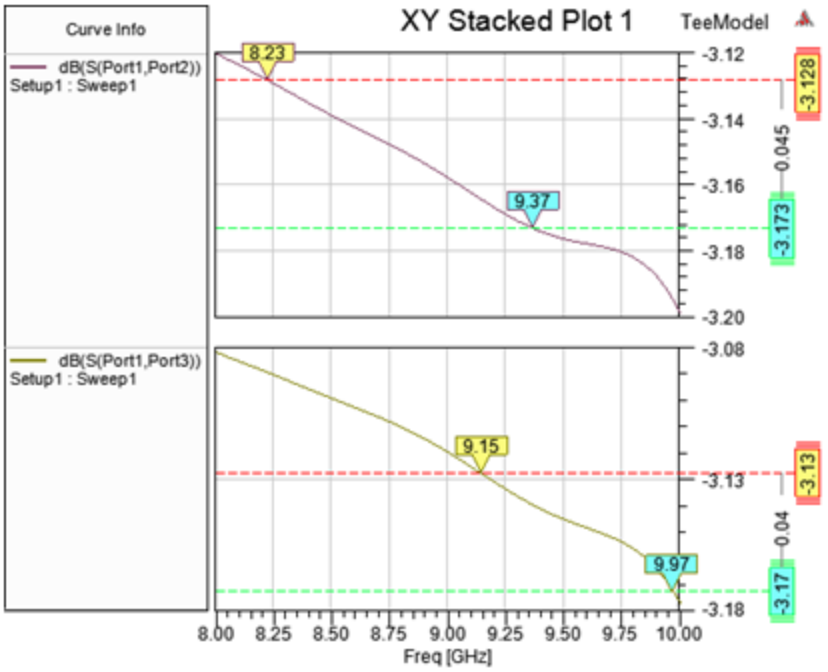


If you change the **Display Type** property of this plot to **Rectangular Stacked Plot**, then a rectangular stacked XY plot is created with each curve in its own stack and a Y marker is shown in each stack with value -3.15 (blue Y marker):

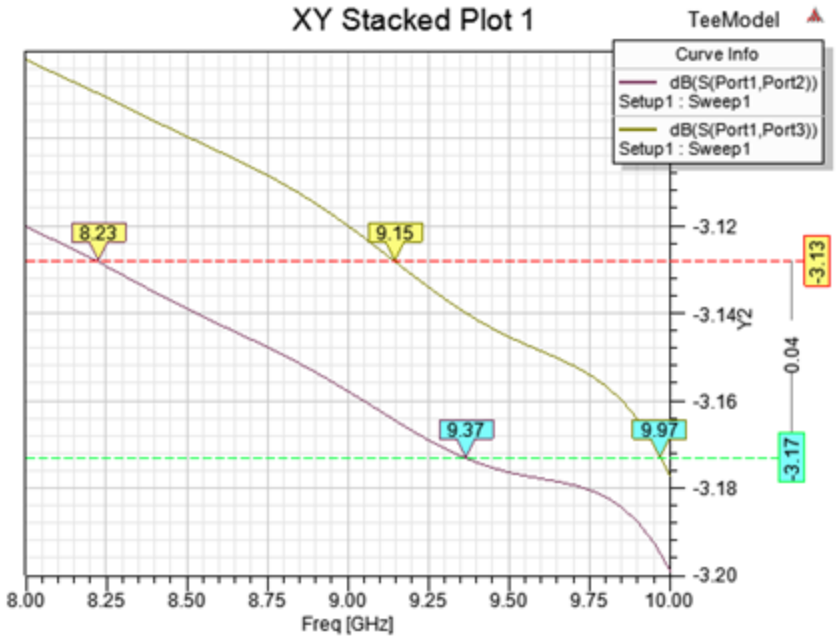


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Similarly when you change a rectangular stacked XY plot to a rectangular XY plot, then all the "same" Y markers in all the stacks are shown as a single Y marker in the rectangular XY plot as shown in following figures:



The rectangular stacked XY plot in the previous figure, when converted to rectangular XY plot, looks like the following figure:



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Adding Data Markers to Traces

Rectangular Stacked Plot

Delta Markers in 2D Reports

To view the difference between any two marker points in a report:

1. Click and hold the mouse button to set the first marker.
2. Drag the mouse to another position, then release the button to create the second marker.

In the marker text window, you see the difference between the two markers instead of the X, Y value of marker.

Discarding Report Values Below a Specified Threshold

To prevent real small numbers from skewing a plot, you can discard small values below a specifiable threshold. Follow this procedure.

1. Double-click the X- or Y-axis of interest on an open plot display to open the **Properties** window for the selected axis.
2. Under the **Axis** tab, find the **Specify Discard Values** property.
3. Select the check box to enable the property.
4. Enter a value in the **Discard Below** field. Units specified elsewhere in the axis property are applied to this value. The **Discard Below** text box is inactive if the **Specify Discard Values** check box is cleared.
5. Click **OK** to apply the discard values to the report.

Adding Characteristics to a Trace

See these topics for detailed information on adding characteristics to a trace:

Adding a Recently Used Trace Characteristic

Adding a Trace Characteristic from Favorites

Adding Trace Characteristics to your Favorites

Adding Characteristics using the Add Trace Characteristics Dialog

Removing All Trace Characteristics

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding a Recently Used Trace Characteristic

If you recently used a characteristic, you can add it to a selected trace by selecting from a list of recently used characteristics. A maximum of 10 is displayed in the menu, and they are sorted alphabetically.

To add a recently used characteristic to a selected trace:

1. Select a trace in a report plot or legend.
2. Click **Report 2D** > **Trace Characteristics**, or right-click a trace.
3. Select **Recent**, then select the function you want. The specified characteristic is added to the trace.

Related Topics

Adding a Characteristic to a Trace

Adding a Trace Characteristic from Favorites

Adding Trace Characteristics to your Favorites

Adding Characteristics using the Add Trace Characteristics Dialog

Removing All Trace Characteristics

Adding a Trace Characteristic from Favorites

You can add a trace characteristic to a selected trace by selecting from a list of favorites. A maximum of 10 is displayed in the menu, and they are sorted alphabetically.

To add a favorite characteristic to a selected trace:

1. Select a trace in a report plot or legend.
2. Click **Report 2D** > **Trace Characteristics**, or right-click a trace.
3. Select **Favorites**, then select the function you want. The specified characteristic is added to the trace.

Related Topics

Adding a Characteristic to a Trace

Adding Trace Characteristics to your Favorites

Adding Characteristics using the Add Trace Characteristics Dialog

Removing All Trace Characteristics

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

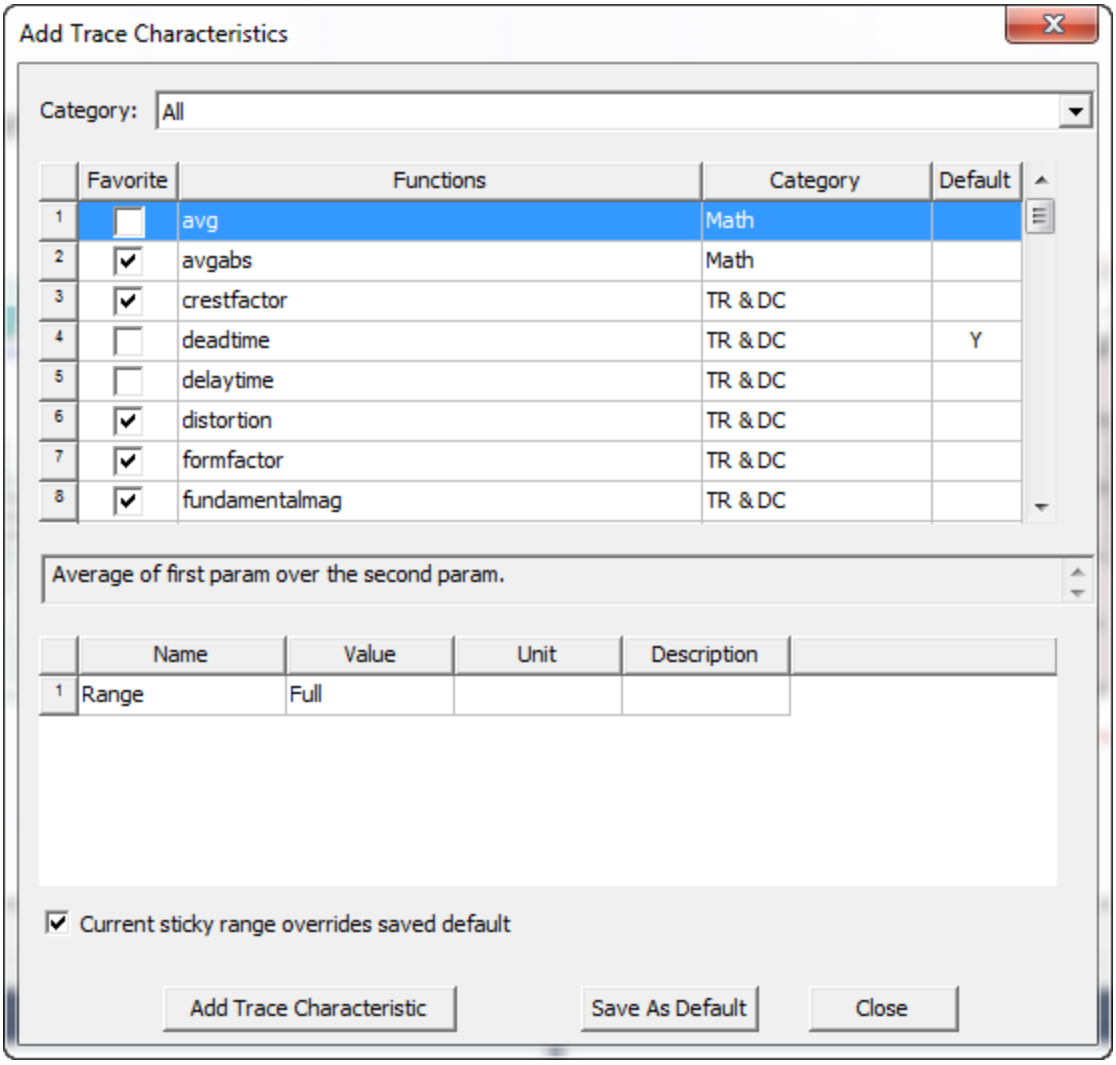
Adding a Recently Used Trace Characteristic

Adding Trace Characteristics to your Favorites

You can add trace characteristics to your list of favorites.

To add characteristics to your list of favorites:

1. Select a trace in a report plot or legend.
2. Click **Report 2D** > **Trace Characteristics**, or right-click a trace.
3. Select **All**. The **Add Trace Characteristics** dialog box appears.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Select the **Favorite** check box in front of any function you want to add to your favorites. You can define as many favorites as you need, but no more than 10 are displayed in the menu, and they are displayed in alphabetical order.
5. Click **Close**. You can view the current favorites by selecting **Favorites** in the **Category** drop-down list.

> **Note:**
>
> Clear the **Favorite** check box for one or more functions and click **Close** to remove favorites.

Related Topics

Adding a Characteristic to a Trace

Adding Characteristics using the Add Trace Characteristics Dialog

Removing All Trace Characteristics

Adding a Recently Used Trace Characteristic

Adding a Trace Characteristic from Favorites

Adding Characteristics using the Add Trace Characteristics Dialog Box

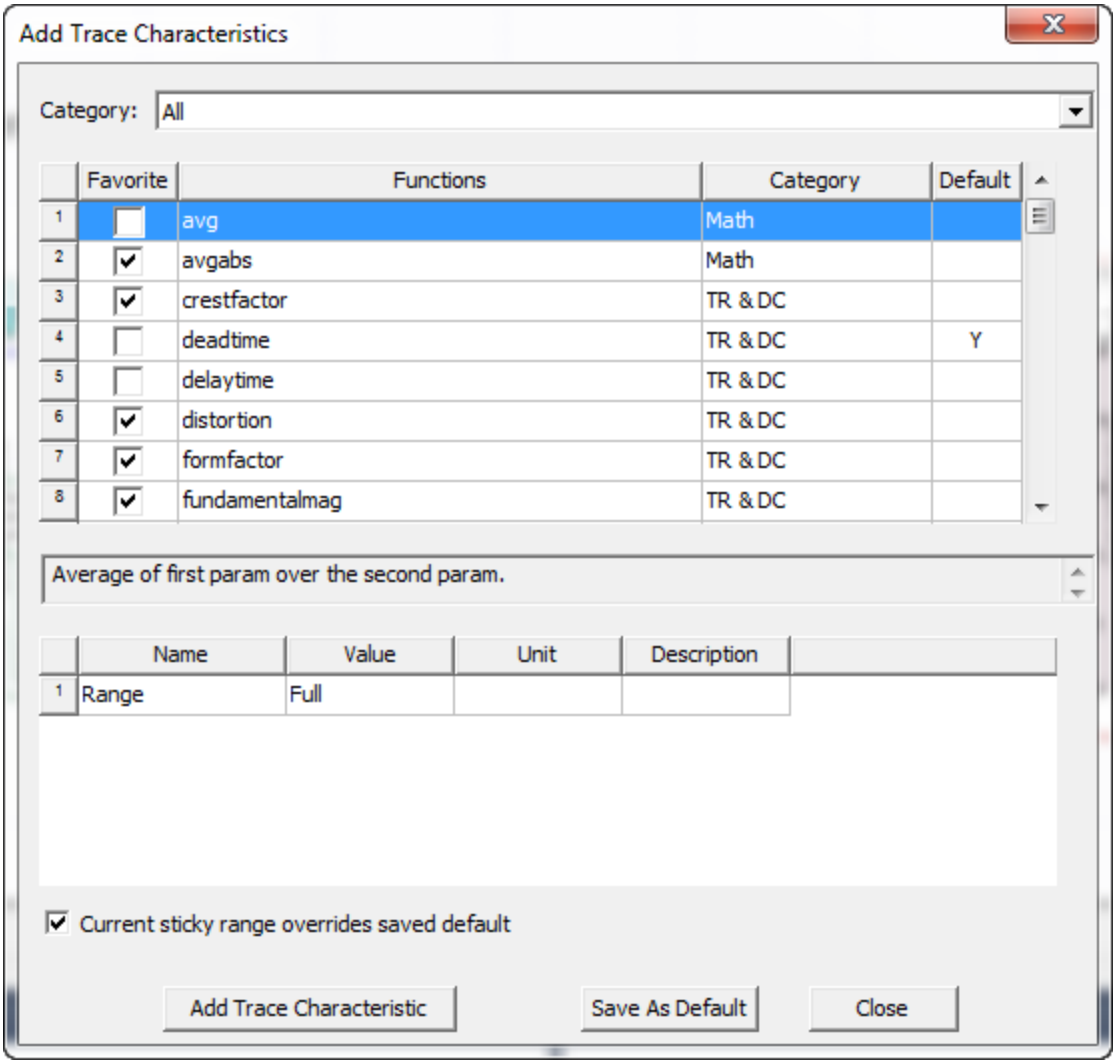
You can add characteristics to a selected trace by selecting from the **Add Trace Characteristics** dialog box.

To add additional characteristics to a selected trace:

1. Select a trace in a report plot or legend.
2. Select **Report 2D** > **Trace Characteristics**, or right-click a trace.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Select **All**. The **Add Trace Characteristics** dialog box appears.



4. Select the **Category**. The available categories depend on the plot, and the selecting of a category displays its associated functions.

Category	Functions
Recent	Displays the most recent functions used, sorted by the time they were added.
Favorites	Displays all favorites. The defaults are max, min, and pk2pk.
All	Displays all available functions.
Math	avgabs, integ, integabs, max, mean, min, pk2pk, pkavg, ripple, rms, rmsAC, stddev, sum, variance, XatYMax, XatYMin, XatYVal, XWidthAtYVal, YatXMax, YatXMin, YatXVal

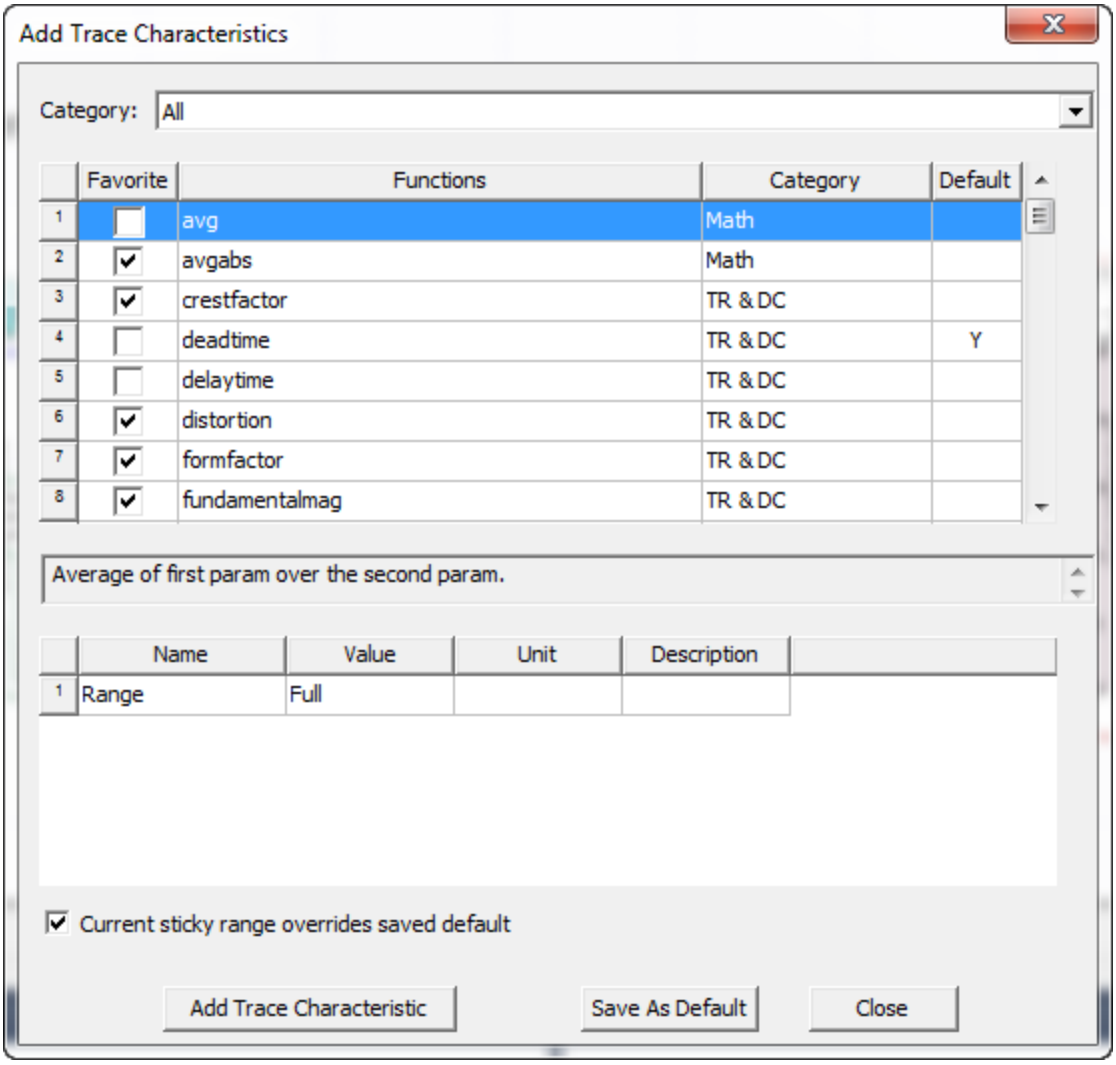
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Category	Functions
PulseWidth	pulsefall9010, pulsefront1090, pulsefront3090, pulsemax, pulsemaxtime, pulsemin, pulsemintime, pulsetail50, pulsewidth5050, pw_minus, pw_minus_avg, pw_minus_max, pw_minus_min, pw_minus_rms, pw_plus, pw_plus_avg, pw_plus_max, pw_plus_min, pw_plus_rms
Overshoot/ Undershoot	overshoot, overshootAbs, undershoot, undershootAbs
TR & DC	crestfactor, deadtime, delaytime, distortion, formfactor, fundamentalmag, risetime, settlingtime
Error	iae, ise, itae, itse
Period	per, pmax, pmin, prms
AC	gainmargin, phasemargin, gaincrossover, phasecrossover, lowercutoff, uppercutoff, bandwidth, peakgain, peakgainfreq
Radiation	lSidelobeY, rSidelobeY, lSidelobeX, rSidelobeX, xdb10Beamwidth, xdb20Beamwidth
Eye Measurements	EyeLevelZero, EyeLevelOne, EyeAmplitude, EyeHeight, EyeSignalToNoise, EyeOpeningFactor, EyeWidth, EyeJitterP2P, EyeJitterRMS, EyeRiseTime, EyeFallTime, MinEyeWidth, MinEyeHeight
TDR	Shunt_C_in_pF, Series_L_in_nH

For a selected function, the **Add Trace Characteristics** dialog box displays the function's purpose in a text field. For a list of functions and their definitions, see the table in ***Defining Traces Using Range Functions***.

5. Some categories and functions call for you to specify one or two additional values in a table. Click **Save as Default** to save these values. The **Default** column shows a **Y** if there is a saved default value for the function.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



6. Select the **Current sticky range overrides saved default** check box if you do not want the range value in the table to change when the function selection is changed; in this case, the current range value becomes the sticky range. If the check box iscleared, the range value is updated from the saved default values and becomes a new sticky range.
7. Click **Add Trace Characteristic** to add the specified characteristics to the trace.
8. Click **Close**.

Related Topics

Adding a Characteristic to a Trace

Adding Trace Characteristics to your Favorites

Removing All Trace Characteristics

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Adding a Recently Used Trace Characteristic

Adding a Trace Characteristic from Favorites

Removing All Trace Characteristics

1. Select a trace in a report plot or legend.
2. Click **Report 2D** > **Trace Characteristics**, or right-click a trace.
3. Select **Trace Characteristics** > **Clear All**.

Trace characteristics are cleared from the selected trace.

Related Topics

Adding a Characteristic to a Trace

Adding Trace Characteristics to your Favorites

Adding Characteristics using the Add Trace Characteristics Dialog

Adding a Recently Used Trace Characteristic

Adding a Trace Characteristic from Favorites

Defining Traces Using Range Functions

Range functions use a 2D dataset as input, along with zero or more additional parameters. Use range functions to produce a 2D report that displays a collection of traces and their attributes. The trace collection and attributes generate a portion of a report definition that generates a family of curves in the report window.

Apply range functions to a range (subset) of points on an X-Y plot; they will then calculate a single-number representation of the specified range, and display that number directly over a plotted wave-form. Use range functions to extract trace characteristics (such as maximum, minimum, and overshoot) from a plot and use those values for additional plotting. You can also use the values to export to a file, where the data table formats are supported. Range functions appear in the **Range Function** dialog box. Extracted trace characteristics may also be used in Optimetrics.

For example, with a wave-form in a transient plot that contains a square pulse, a range function can calculate and display a single value that represents the High-to-Low/Low-to-High transition states. Or, with a plot that shows a pulse, the **PulseWidth** and **RiseTime** range functions use the entire plotted curve to calculate a single number that represents the width or rise-time, and display that single-number representation directly over the specified range of plotted points.

- Range functions trace characteristics appear on the report window as a column in legend window.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Range functions are available from reporter and optimetrics.
- Numerous range functions are available in these categories:
  - Math functions
  - Pulse width functions
  - Overshoot/Undershoot functions
  - TR & DC functions
  - Error functions
  - Period functions
  - Radiation functions
  - Eye Measurements functions

This table lists the available range functions:

Range Function	Category	Description	Parameters
***avg**	Math	Returns the average of the values of the selected quantity. avg = (Area between the curve and the X-axis) / (X length of the curve)	N/A
avgabs	Math	Returns the mean of the absolute value of the selected quantity.	N/A
crestfactor	TR & DC	Returns the crest factor (peak/RMS) for the selected quantity.	N/A
deadtime	TR & DC	Obtains the latest time when the qtyl is within a tolerance of zero.	Tolerance: The +/- bandwidth around zero .
delaytime	TR & DC	Obtains the time from zero to 50% of the target point.	Target: The target value for input.
distortion	TR & DC	Returns the total	Frequency: Freq in Hz at which to calculate

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Range Function	Category	Description	Parameters
		distortion for the selected simulation quantity and an additional argument frequency, which is the frequency in Hz at which to calculate the fundamental RMS of the simulation quantity.	the RMS value of the selected quantity.
formfactor	TR & DC	Returns the form factor (RMS/Mean Absolute Value) for the selected quantity.	N/A
fundamentalmag	TR & DC	Returns the RMS value of the fundamental frequency for the selected quantity, and an additional argument, Frequency, which specifies the fundamental frequency.	Frequency: Freq in Hz at which to calculate the RMS value of the selected quantity.
iae	Error	Returns the integral of the absolute deviation of the selected quantity from a target value that is entered via the additional argument.	Target: Target value.
integ	Math	Integral of the selected quantity.	N/A

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Range Function	Category	Description	Parameters
		Uses trapezoidal area.	
integabs	Math	Returns the integral of the absolute value of the selected qty.	N/A
ise	Error	Returns the integral of the squared deviation of the selected quantity from a target value that is entered via an additional argument.	Target: Target value.
itae	Error	Returns the time-weighted absolute deviation of the selected quantity from a target value that is entered via an additional argument.	Target: Target value.
itse	Error	Returns the time-weighted squared deviation of the selected qty from a target value that is entered via an additional argument.	Target: Target value.
max	Math	Maximum (of magnitudes).	N/A
mean	Math	Returns the average in the set of quantities selected. mean = sum(all	N/A

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Range Function	Category	Description	Parameters
		y-value) / (number of y-values)	
min	Math	Minimum (of magnitudes).	N/A
overshoot	Overshoot/Undershoot (Overridable)	Calculates peak overshoot given a threshold value and number of evenly spaced points over entire time range.	Threshold: The reference value from where the overshoot/undershoot is calculated. Number of Points: Number of evenly spaced time points.
per	Period (Overridable)	Calculates period.	Threshold: Y transition value which determines the period. Number of Points: Number of evenly spaced time points.
pk2pk	Math	Difference between maximum and minimum of the first parameter over the second parameter. Returns the peak-to-peak value for the selected simulation quantity.	N/A
pkavg	Math	Returns the ratio of the peak to peak-to-average for the selected quantity.	N/A
pmax	Period (Overridable)	Maximum period of input stream.	Threshold: Y transition value which determines the period. Number of Points: Number of evenly

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Range Function	Category	Description	Parameters
			spaced time points.
pmin	Period (Overridable)	Minimum period of input stream.	Threshold: Y transition value which determines the period. Number of Points: Number of evenly spaced time points.
prms	Period (Overridable)	Rms of period of input stream.	Threshold: Y transition value which determines the period. Number of Points: Number of evenly spaced time points.
pulsefall9010	Pulse Width	Returns the pulse fall time of the selected quantity according to the 90%-10% estimate.	N/A
pulsefront1090	Pulse Width	Returns the pulse front time of the selected quantity according to the 10%-90% estimate.	N/A
pulsefront3090	Pulse Width	Returns the pulse front time of the selected quantity according to the 30%-90% estimate.	N/A
pulsemax	Pulse Width	Returns the pulse maximum from the front and tail estimates for the selected quantity.	N/A
pulsemaxtime	Pulse Width	Returns the time at which the maximum pulse value of the	N/A

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Range Function	Category	Description	Parameters
		selected quantity is reached.	
pulsemin	Pulse Width	Returns the pulse minimum from the front and tail estimates for the selected quantity.	N/A
pulsemintime	Pulse Width	Returns the time at which the minimum pulse value of the selected quantity is reached.	N/A
pulsetail50	Pulse Width	Returns the pulse tail time of the selected quantity from the virtual peak to 50%.	N/A
pulsewidth5050	Pulse Width	Returns the pulse width of the selected quantity as measured from the 50% points on the pulse front and pulse tail.	N/A
pw_minus	Pulse Width (Overridable)	Pulse width of the first negative pulse.	Threshold: Y transition value which determines the pulse width. Number of Points: Number of evenly spaced time points.
pw_minus_avg	Pulse Width (Overridable)	Average of the negative pulse width input stream.	Threshold: Y transition value which determines the pulse width. Number of Points: Number of evenly spaced time points.
pw_minus_max	Pulse Width	Maximum pulse	Threshold: Y transition

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Range Function	Category	Description	Parameters
	(Overridable)	width of the negative pulse of input stream.	value which determines the pulse width. Number of Points: Number of evenly spaced time points.
pw_minus_min	Pulse Width (Overridable)	Minimum pulse width of the negative pulse of input stream.	Threshold: Y transition value which determines the pulse width. Number of Points: Number of evenly spaced time points.
pw_minus_rms	Pulse Width (Overridable)	RMS of the negative pulse width input stream.	Threshold: Y transition value which determines the pulse width. Number of Points: Number of evenly spaced time points.
pw_plus	Pulse Width (Overridable)	Pulse width of first positive pulse.	Threshold: Y transition value which determines the pulse width. Number of Points: Number of evenly spaced time points.
pw_plus_avg	Pulse Width (Overridable)	Average of the positive pulse width input stream.	Threshold: Y transition value which determines the pulse width. Number of Points: Number of evenly spaced time points.
pw_plus_max	Pulse Width (Overridable)	Maximum pulse width of the positive pulse of input stream.	Threshold: Y transition value which determines the pulse width. Number of Points: Number of evenly

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Range Function	Category	Description	Parameters
			spaced time points.
pw_plus_min	Pulse Width (Overridable)	Minimum pulse width of the positive pulse of input stream.	Threshold: Y transition value which determines the pulse width. Number of Points: Number of evenly spaced time points.
pw_plus_rms	Pulse Width (Overridable)	RMS of the positive pulse width input stream.	Threshold: Y transition value which determines the pulse width. Number of Points: Number of evenly spaced time points.
ripple	Math	Returns the ripple factor (AC RMS/Mean) for the selected quantity.	N/A
risetime	TR & DC	Obtains the time taken to go from 10% to 90% of target point.	Target: The target value for input.
rms	Math	Returns total RMS of the selected quantity.	N/A
rmsAC	Math	Returns the AC RMS for the selected quantity.	N/A
settlingtime	TR & DC	Returns the latest time at which the value of the selected simulation quantity fell outside its tolerance band. The target value of the quantity	Target: Tolerance.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Range Function	Category	Description	Parameters
		and the +/- bandwidth of the tolerance band are the additional args.	
stddev	Math (Overridable)	Returns the standard deviation of given values.	N/A
sum	Math (Overridable)	Returns the sum of given values.	N/A
undershoot	Overshoot/Undershoot (Overridable)	Calculates peak undershoot given a threshold value and number of evenly spaced points over entire time range.	Threshold: The reference value from where the overshoot/undershoot is calculated. Number of Points: Number of evenly spaced time points.
variance	Math (Overridable)	Returns the variance of given values.	N/A
XAtYMax	Math	Returns the X value at maximum Y.	N/A
XAtYMin	Math	Returns the X value at minimum Y.	N/A
XAtYVal	Math	Returns the X value at the first occurrence of Y value.	Y Value: Y value at which we need to find X.
YAtXMax	Math	Returns the Y value at maximum X.	N/A
YAtXMin	Math	Returns the Y value at minimum X.	N/A
YAtXVal	Math	Returns the Y	X Value: X value at

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Range Function	Category	Description	Parameters
		value at the first occurrence of X value.	which we need to find Y.

Copy and Paste of Report and Trace Definitions

You can copy and paste report and individual trace definitions within a single design or across designs. The report or trace definition will be evaluated within the context of the target design or report.

> **Note:**
>
> If the report or trace definition contains properties that do not exist in the target design (for example, a port name) an error will be posted that indicates a solution does not exist for this trace.

> **Note:**
>
> You must copy and paste trace definitions between the same report types. For example, you cannot copy a trace from a Modal Solution Data report and paste it in a Far Fields report.

To copy a Report Definition

Right-click the report name in the Project tree and select **Copy Definition**.

To paste the Report Definition

Right-click **Results** in the Project tree of the target design and select **Paste**.

A new report is created and it contains the copied definitions.

To copy individual Trace Definitions

Right-click the trace or traces under a report name in the Project tree and select **Copy Definition**.

To paste the Trace Definitions

Right-click the report in the target design to which you would like to copy the trace or traces and select **Paste**.

New traces are added to the report and it contains the copied trace definitions.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> If you copy and paste a report or trace definition to a design which contains a definition with the same name, an incremented number is appended to the pasted report or trace name.

Related Topics

Copying Plots to the Clipboard as Images

Copy and Paste of Report and Trace Data

Copy and Paste of Report and Trace Data

You can copy and paste report and individual trace data within a single design or across designs. The report and trace definitions and all underlying data within the report or trace are copied and pasted to the target design or report.

To copy all data from a report

- Right-click the report name in the Project tree and select **Copy Data**,
- Select **Edit** > **Copy Data**, or
- Right-click a plot and select **Copy Data**.

To paste copied report data

Right-click **Results** in the Project tree of the target design and select **Paste**.

To copy data from individual traces in a report

Right-click the trace or traces under a report name in the Project tree and select **Copy Data**.

To paste copied trace data

Right-click the report in the target design to which you would like to copy the trace data and select **Paste**.

> **Note:**
>
> If you copy and paste report or trace data which contains the same name definition as a report or trace in the target design, the system appends an incremented number to the pasted name.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Copying Plots to the Clipboard as Images

Copy and Paste of Report and Trace Definitions

Removing Traces

In the Project tree, select the traces you want to remove and click **Delete**.

Related Topics

Setting Report2D options

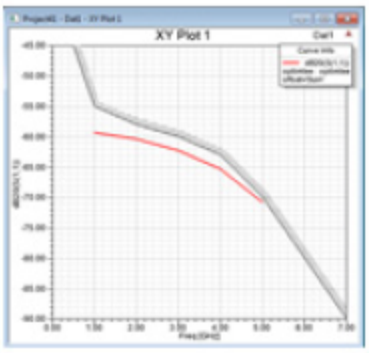
Working with Traces

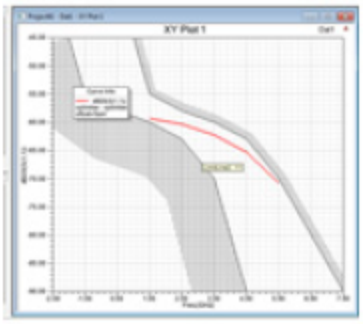
Adding Characteristics to a Trace

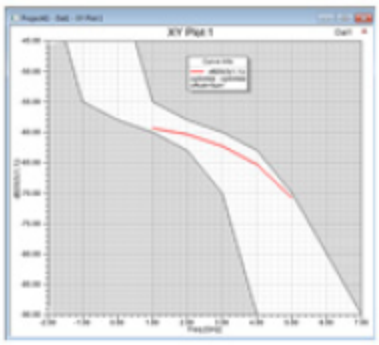
Removing Traces

Limit Lines in Cartesian Plots

Limit lines are simple graphical representation of constraints on XY plots. These are modeled as a sequence of XY point pairs. For example, you can designate a single limit line to delineate either an upper or lower limit, or two limit lines to delineate upper and lower limits. You can add as many limit lines as you want. You can control the display properties of the line including color and hatch width.







Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> Limit lines are available only on rectangular (XY) plots, not on XY-like plots such as Bode or rectangular stacked.

You can create limit lines in the following ways:

- By specifying points to define the lines.
- From a selected curve on the plot.

Related Topics

Specifying Points to Create Limit Lines

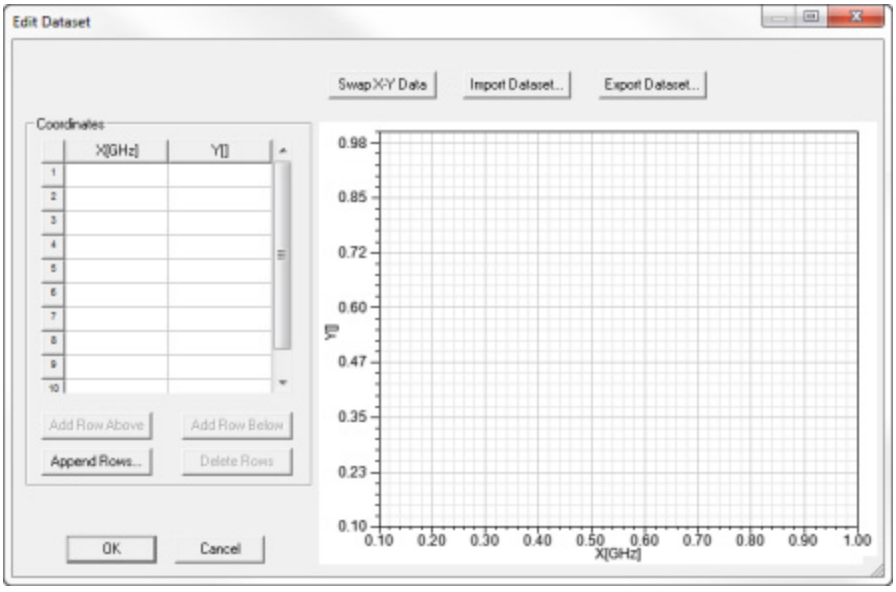
Creating Limit Lines from a Selected Curve

Editing Limit Lines

Modifying the Background Properties of a Report

Specifying Points to Create Limit Lines

1. Select **Report2D** > **Add Limit Line** > **Specify Points**, or right-click an XY plot and select **Add Limit Line** > **Specify Points**. The **Edit Dataset** dialog box appears.



You can use this dialog box to:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

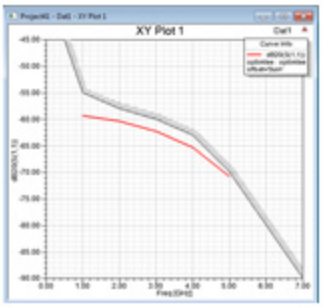
- Enter the XY coordinate values directly.
- Import XY values from a tab-delimited TAB file.
- Export a dataset to a file.

If you require additional data points, use the buttons to append rows to the **Coordinates** table. If you select a row in the **Coordinates** table, you can then use the buttons to add a row above or add a row below the selected rows, or delete rows.

Press Shift+click to select multiple adjacent rows, or Ctrl+click to select any combination of rows for deletion.

> **Note:**
>
> Each limit line is associated with a particular Y-axis (since it has to be scaled the same way as all the curves associated with the axis, follow its log/linear scale, and so on). This Y-axis association defaults to the first available Y-axis when the limit line is created. However, if the plot contains multiple Y-axes, it can be associated with a different Y-axis later via its properties tab.

2. When you click **OK**, the limit line you defined is added to the plot. The line divides the plot into regions within the context of its length. By default, the region above the limit line is hatched.



Related Topics

Creating Limit Lines From a Selected Curve

Editing Limit Lines

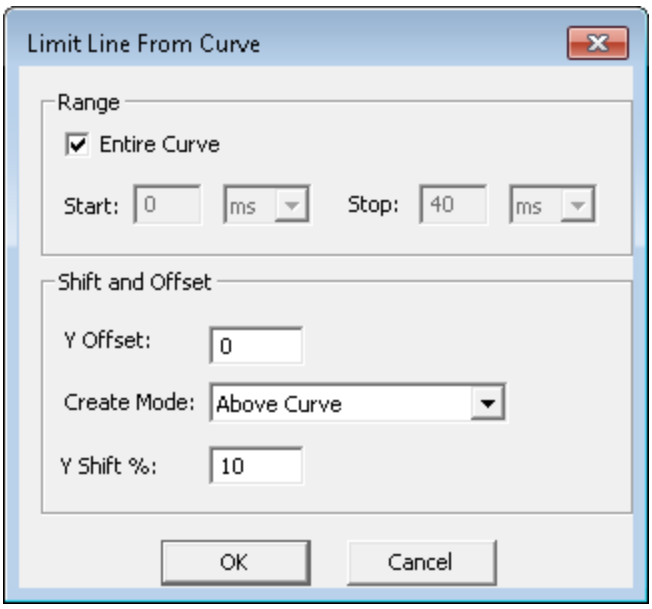
Modifying the Background Properties of a Report

Creating Limit Lines from a Selected Curve

Follow this procedure to create limit lines from a selected curve.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

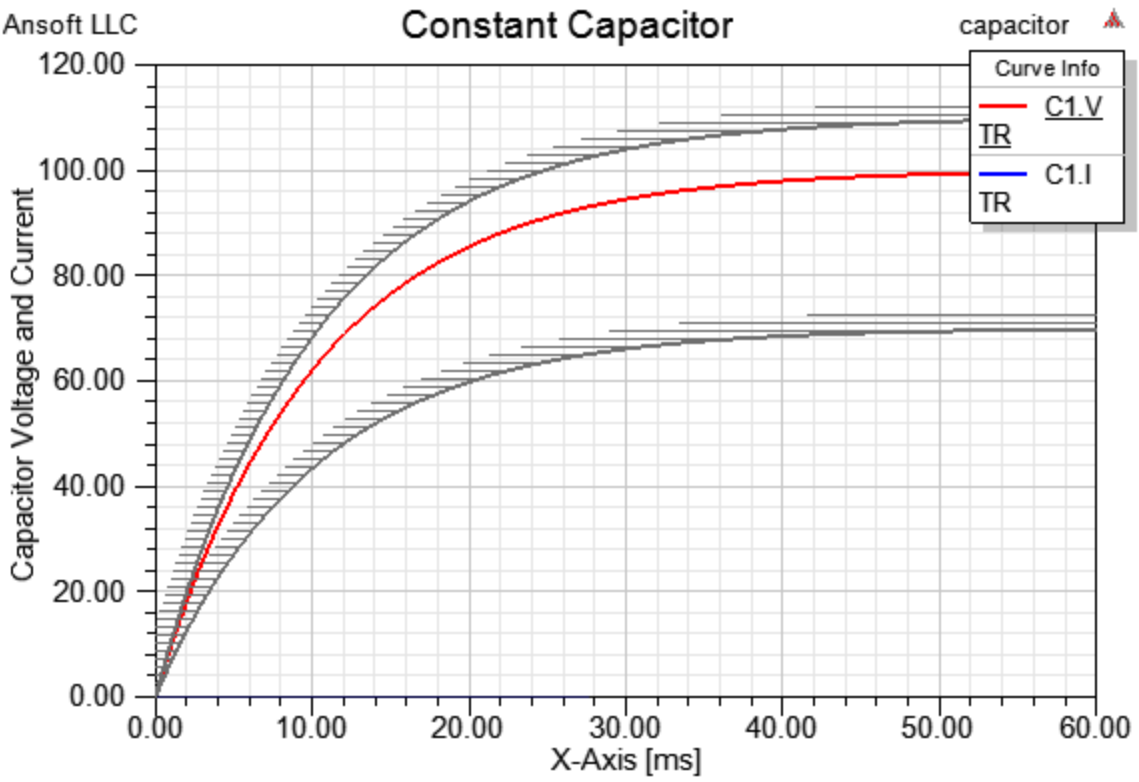
1. Click the desired plot curve and select **From Selected Curve**. The **Limit Line From Curve** dialog box appears.



2. The **Limit Line From Curve** dialog box lets you create limit lines over the range of the entire curve or between specified X-axis start and stop points. **Create Mode** controls whether the limit line is above the curve, below the curve, or above and below the curve. You can also specify a Y offset value, which sets a constant distance between the curve and limit lines, and a Y shift %, which shifts the limit lines by the specified percentage of the Y-axis values.

   In the following example, the upper limit line has a Y offset value of 0.01 and a Y shift % of 10, while the lower limit line has no Y offset and a Y shift % of 30,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



3. When you click **OK**, the limit lines you defined are added to the plot. The lines divide the plot into regions within the context of its length. By default, the region above a limit line is hatched.

Related Topics

Specifying Points to Create Limit Lines

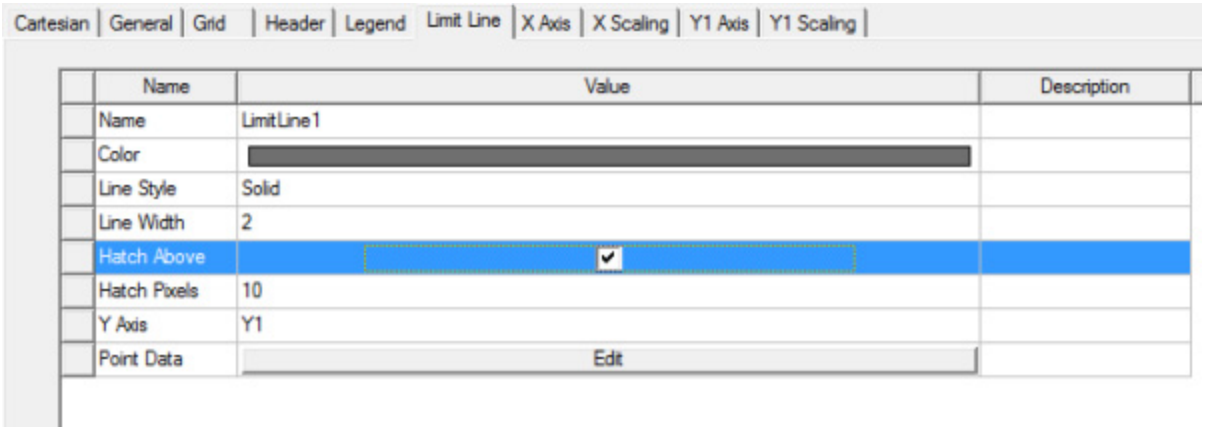
Editing Limit Lines

Modifying the Background Properties of a Report

Editing Limit Lines

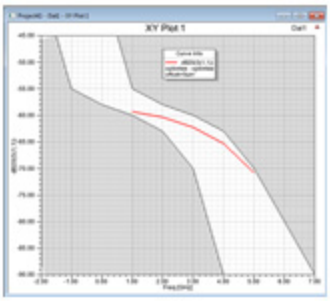
You can select a limit line in a plot to edit the following properties on the **Limit Line** tab of the plot properties dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



- **Color** – Affects both line and hatching.
- **Style and Width** – Affects only the line, not the hatching.
- **Y Axis Y**– Axis associated with the limit line.
- **Hatch Pixels** – Sets the length of the hatch lines.
- **Point Data** – Edit the data that defines the limit line.
- **Hatch Above** – Sets the direction of the hatch lines (above or below) the limit line.

For example, if you add a second limit line, you could designate it as hatch below by clearing **Hatch Above** to produce a “tunnel” marking the upper and lower constraints.



Related Topics

Specifying Points to Create Limit Lines

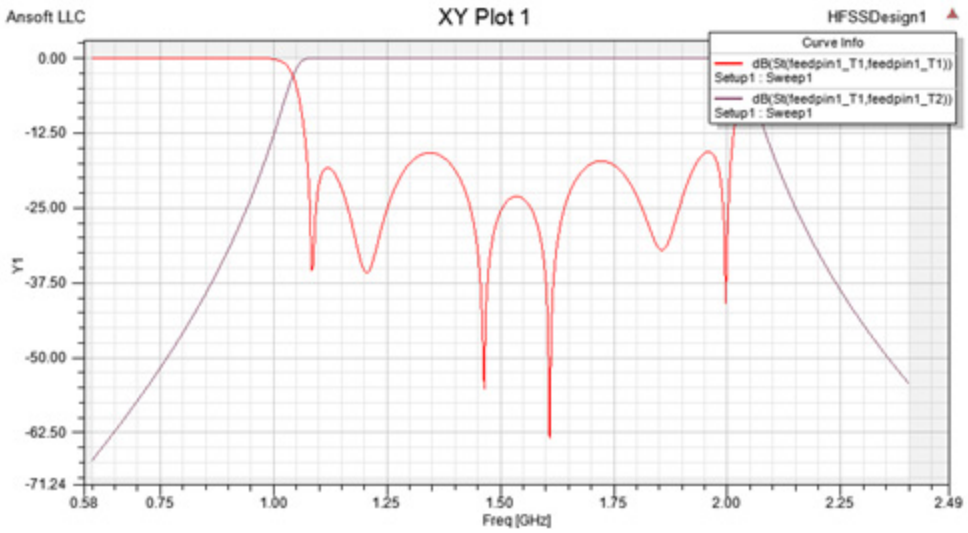
Creating Limit Lines From a Selected Curve

Modifying the Background Properties of a Report

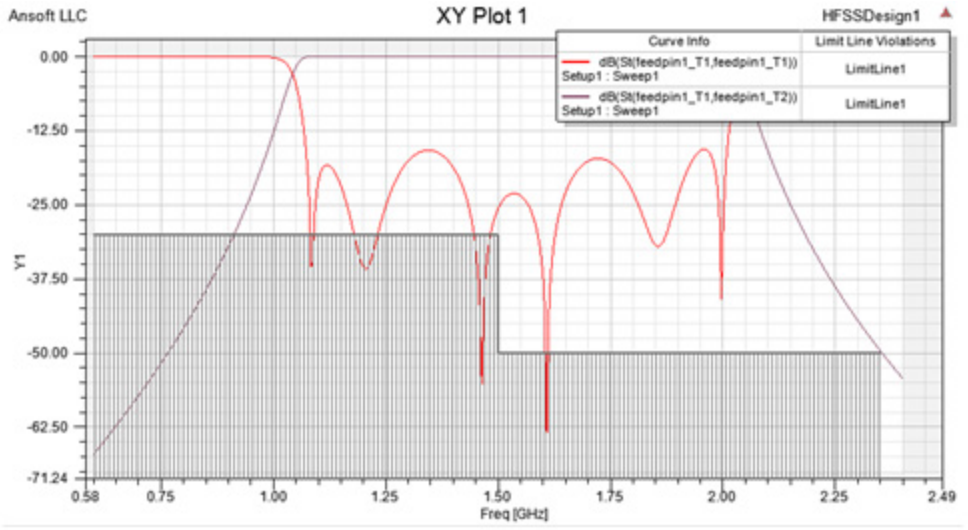
Limit Line Violations

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Use a plotting feature to help you discern whether a curve violates a limit line. Consider the following plot which shows two curves:



Suppose that the response cannot be below -30 dB till 1.5 GHz and below -50 dB above 1.5 GHz. You add a limit line for this requirement in the plot. The plot calculates whether a curve violates this requirement for the limit line and shows it in the legends window, as shown in following figure:

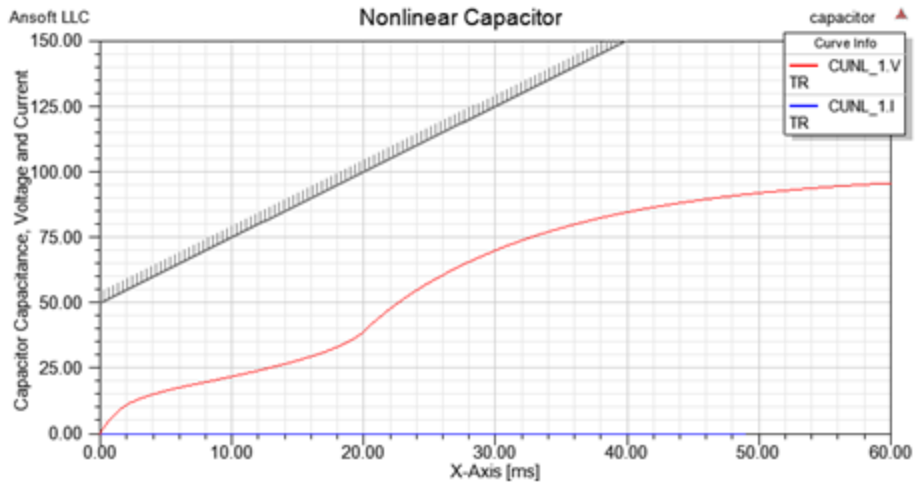


If a curve is selected, then the plot shows the region of the curve that violates the limit line (shaded with slanted red lines), as shown in following figures:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

If no curve violates a limit line, then limit line hatching is limited to 10 pixels; otherwise, limit line hatching is limited to infinity as shown in following figure. This is done to retain focus on the curves instead of the limit lines.



Error Handling

If the plot encounters an error while calculating limit line violations, the legend window shows NaN (limit line name) in front of the curve under the **Limit Line Violations** column. This ensures that you do not get misleading information about no-violations when the issue is actually a limitation in our code.

Related Topics

Modifying the Background Properties of a Report

Variables, Quantities and Functions

The follow sections describe how to sweep variables, and select various quantities and functions for use in specifying values to display in plots and reports.

Sweeping a Variable in a Report

In Twin Builder, a swept variable is a variable that typically has more than one value. You can plot any calculated or derived quantity against one or more of the swept variable's values.

To specify the swept variable values to plot a selected quantity against:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. In the **Report** dialog box, select the variable from the **X (Primary Sweep)** drop-down menu.
2. To modify the values that will be plotted for a variable:

   a. Click on the **X (Primary Sweep)** line to display a list of all the possible values.
   b. Select **All Values** or click **Edited** to open a dialog box that lets you specify the sweeps to use.

      All of the selected variable's values will be plotted.

Selecting a Function

The value of a quantity being plotted depends upon its mathematical function, which you select from the **Function** list in the **Report** dialog box. The available, valid functions depend on the type of quantity (real or complex) being plotted. The function is applied to the quantity which is implicitly defined by all the swept and current variables.

Some of these functions can operate along an entire curve. These are min, max, integ, rms, pk2pk, cang_deg and cang_rad.

You can select from the following functions in the **Trace** tab **Function** list:

abs	Absolute value.
acos	Arc cosine.
acosh	Hyperbolic arc cosine.
ang_deg	Angle (phase) of a complex number, cut at +/-180.
ang_deg_val	Angle (phase of a complex number in unitless degree values). Returns simple numbers.
ang_rad	Angle in radians.
arg	Argument of a complex number. It is the angle the complex number makes with the positive X-axis. Same as **ang_deg**.
asin	Arc sine.
asinh	Hyperbolic arc sine.
atan	Arc tangent.
atanh	Hyperbolic arc tangent.
cos	Cosine.
cosh	Hyperbolic cosine.
cum_integ	The cumulative integral function returns a set of values that have the same length as the original set of points (the first element will always be zero). Element *I* of the set returned by cum_integ is the integral of elements 1 through *I* of the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	original data set.
cum_sum	The cumulative sum function returns a data set that has the same length as the original set of points. Element *I* of the set returned by cum_sum is the sum of elements 1 through *I* of the original data set.
dB(x)	20*log10(|x|).
dBc	Decibels relative to the carrier. It is the power ratio of the signal to a carrier signal. Gives the relative signal strength.
degel	Degrees electrical. degel(x, y) = x / 2*pi*y.
deriv	Derivative of a given parameter.
even	Returns **1** if integer part of the number is even; returns **0** otherwise.
exp	Exponential function (the natural anti-logarithm).
int	Truncated integer function.
ln	Natural logarithm.
log10	Logarithm base 10.
mag	Magnitude of the complex number.
nint	Nearest integer.
normalize	Divides each value within a trace by the maximum value of the trace. ex. normalize(mag(x)).
odd	Returns **1** if integer part of the number is odd; returns **0** otherwise.
rem	Fractional part.
sgn	Sign extraction.
sin	Sine.
sinh	Hyperbolic sine.
sqr	Square of the argument.
sqrt	Square root.
tan	Tangent.
tanh	Hyperbolic tangent.

Additionally, the following functions are not in the **Function** list, but are supported and can be entered manually.

db10normalize (x)	10*log [normalize(mag(x))].
db20normalize (x)	20*log [normalize(mag(x))].

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

j0	Bessel function of the first kind (0th order).
j1	Bessel function of the first kind (1st order).
y0	Bessel function of the second kind (0th order).
y1	Bessel function of the second kind (1st order).

Selecting Solution Quantities to Plot

When you create a report of solution data, each trace in the report includes a quantity that is plotted along an axis. This quantity can be a value calculated by Twin Builder, a value from a calculated expression, or an intrinsic (inherent) variable value such as frequency or theta. The valid categories available depend on the type of quantity (real or complex) being plotted, the setup, the solution type, and the plot domain.

To select a quantity to plot:

1. In the **Report** dialog box, select one of the following categories. The selected category provides the default name for the plot, such as S Parameter Plot *n*. You can edit the plot names in the Project tree and the plot header text in the report synchronizes.

<various>	Categories vary based on user-defined outputs, for example: Voltage, Current, Flux, Torque, Frequency.
Variables	Intrinsic variables, such as frequency or theta, or user-defined project variables, such as the length of a quarter-wave transformer.
Output Variables	User-defined expressions applied to derive quantities from the original field solution.

2. Select a quantity to plot from the **Quantity** list. The available quantities depend upon the selected category and the setup of the design.

Plotting Imported Solution Data

1. Select **Twin Builder** > **Import SDB File**. A file **Open** dialog box appears.
2. Locate the Twin Builder solution SBD file containing the solution to import and click **OK**.
3. If the imported data is in table format, click **Table Data** on the **Import Data** dialog box. The **Data Table Import** dialog box opens.
4. Locate the data table file and click **OK**.

To plot imported solution data:

1. Expand **Analysis** in the Project tree. Verify the presence of the imported solution data.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Right-click **Results** in the Project tree and select **Create Standard Report** > *<report type>*.
3. On the **Report** dialog box, select the imported solution from the **Solutions** drop-down menu.
4. Select the desired data in the **Traces** tab and click **New Report**. The **Report** dialog box opens to display the imported solution.
5. Select and add traces or other reports based on the imported solution data as needed. When finished, click **Close** to dismiss the **Report** dialog box.

Setting a Range Function

To apply a range function to the Y, Z, or Mag component of a trace:

1. Click **Range Function** in the **Reports** dialog box. This opens the **Set Range Function** dialog box. The functions available are listed in the table in ***Defining Traces Using Range Functions***.
2. Click **Specified** on the **Range function** line to enable the **Range Function** fields.
3. Select the **Category**, then an associated function to apply. The available categories depend on the plot, and the category enables the display of associated functions.

Category	Functions for the Category
Math	max, min, pk2pk, rms, integ, integabs, avgabs, rmsAC, ripple, pkavg, XatYMin, XatYMax, XatYVal
PulseWidth	pulsefall9010, pulsefront9010, pulsefront3090, pulsemax, pulsemaxtime, pulsemin, pulsemintime, pulsetail50, pulsewidth5050, pw_plus, pw_plus_max, pw_plus_min, pw_plus_avg, pw_plus_rms, pw_minus_max, pw_minus_min, pw_minus_avg, pw_minus_rms
Overshoot, Undershoot	overshoot, undershoot
TR & DC	crestfactor, formfactor, distortion, fundamentalmag, delaytime, risetime, deadtime, settlingtime
Error	iae, ise, itae, itse
Period	per, pmax, pmin, prms
Radiation	xdb10bandwidth, xdb20bandwidth, lSidelobeX, lSidelobeY, rSidelobeX, rSidelobeY

Given a selected function and category, the **Set Range Function** dialog box displays a text field that explains the purpose of the function. For a list of functions and their definitions, see the table in ***Defining Traces Using Range Functions***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Selecting a function causes the display of a description in the **Purpose** field. If the function requires a value (such as the **XatYVal Math** function or the **pw_minus_max Pulse Width** function), the **Function** field displays its name, editable value field, unit, and description.

4. Use the **Over Sweep** drop-down menu to select from available sweeps.
5. To select from available sweeps, or to edit them, click and clear the **Use All Sweeps** check box. This enables a list of the sweeps. The sweeps you select appear on the **Over Sweep** line. Use the buttons to **Clear All Selections** or **Select All** sweeps.
6. Select the sweeps **Default** or **Edited** buttons to specify whether to accept the default or edited sweeps.
7. To edit the sweeps further, click to display an **Edit Sweep** dialog box.

   For frequency variables, this lets you specify a single value, linear step, linear count, decade count, octave count, or exponential count. You can **Add** legal values to the list of sweep values, **Update** the list for changes, or **Delete** selected entries.
8. Click **OK** to apply the range function.

Related Topics

Selecting a Function

Specifying Output Variables

The **Output Variables** dialog box contains four sections:

- **Context** – Specify the report type, the solution, and for appropriate report types, the domain. Changing the report type affects whether the **Domain** menu appears, and may affect the functions listed in the **Calculation** section.
- **Output Variables** – Specify the name and expression for a new output variable.
- **Quantities** – Insert quantities into the **Expression** area of the **Output Variables** section.
- **Function section** – Insert completed expressions into the **Expression** area of the **Output Variables** section.

Adding a New Output Variable

Follow this procedure to add an output variable:

1. Open the **Output Variables** dialog box in one of these ways:
   - Select **Twin Builder** > **Results** > **Output Variables**.
   - In the Project tree, right-click **Results** and select **Output Variables**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- In the **Report** dialog box, click **Output Variables**.

2. The **Output Variables** dialog box appears. Variables defined using **Twin Builder** > **Results** > **Output Variables** appear in the list at the top of the dialog box.
3. In the **Output Variables** section, enter a name for the new variable in the **Name** box.
4. To enter an expression, do one or both of the following:
   - Type part or all of the expression directly in the **Expression** area. Valid functions appear in blue. Invalid functions appear in red.
   - Insert part or all of the expression using the options in the ***Building an Expression*** section.
5. Click **Add** to add the new variable to the list.
6. Repeat steps 3 through 5 to add additional variables.
7. When you are finished adding output variables, click **Done** to close the **Output Variables** dialog box.

Building an Expression Using Existing Quantities

When you are entering an expression for a new output variable, you can insert part or all of the expression using the options in the **Quantities** and **Function** sections of the **Output Variables** dialog box.

Follow this procedure to add an input variable by inserting part or all of the expression:

1. Open the **Output Variables** dialog box in one of these ways:
   - Click **Twin Builder** > **Results** > **Output Variables**.
   - In the Project tree, right-click **Results** and select **Output Variables**.
   - In the **Report** dialog box, click **Output Variables**.

   The **Output Variables** dialog box appears.
2. In the **Output Variables** section, enter a name for the new variable in the **Name** box.
3. To insert a quantity:
   a. From the **Report Type** drop-down list, select the type of report from which you want to select the quantity.
   b. From the **Solution** drop-down list, select the solution from which you want to select the quantity.
   c. From the **Category** list, select the type of quantity to enter.
   d. From the **Quantity** list, select the quantity.
   e. From the **Function** list, select a ready-made function (this option is the same as inserting the function from the **Function** section).
   f. If applicable, from the **Domain** list, select the solution domain.
   g. Click **Insert Into Expression**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The selected quantity is entered into the **Expression** area of the **Output Variables** section.

4. To insert a function:
   a. In the **Function** section, select a ready-made function from the drop-down list.
   b. Click **Insert Function into Expression**.

   The function appears in the **Expression** area of the **Output Variables** section.
5. When you are finished defining the variable in the **Expression** area, click **Add** to add the new variable to the list.
6. Repeat steps 2 through 5 to add additional variables.
7. When you are finished adding output variables, click **Done** to close the **Output Variables** dialog box.

> **Note:**
>
> Remember the evaluated value of an expression is always interpreted as SI units. However, when a quantity is plotted in a report, you have the option to plot values in units other than SI. For example, the expression “1+ang_deg(S11)” represents an “angle” quantity evaluated in radians though plotted in degrees units. To represent an angle quantity in degrees, you would specify units as “1 deg + ang_deg(S11)”.

Deleting Output Variables

Follow this procedure to delete output variables:

1. Remove all references to the output variable in the project.
2. Save the project to erase the command history.
3. Open the **Output Variables** dialog box in one of these ways:
   - Select **Twin Builder** > **Results** > **Output Variables**.
   - In the Project tree, right-click **Results** and select **Output Variables**.
   - In the **Report** dialog box, click **Output Variables**.
4. Select the variable and click **Delete**.
5. Click **OK** to close the dialog box.

Report Data

The following sections describe how you can view, update, export, import, override, and delete various values used for display in plots and reports.

Updating Reports (Post-Processing Data)

Deleting Reports

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Exporting Plot Data

Importing 2D Plot Data

Report File Formats

Exporting Graphics Files from a Plot

Plotting Imported Solution Data

Setting a Range Function

Updating Reports (Post-Processing Data)

To manually update the active report in the design area using the latest simulation results, right-click the report and select **Update Report**, right-click the report under **Results** in the **Project Manager** pane and select **Update Report**.

If you want to update all reports in the design, select **Twin Builder** > **Results** > **Update All Reports**, or right-click **Results** in the **Project Manager** pane and select **Update All Reports**.

> **Note:**
>
> If the simulation associated with a report has been paused via the progress bar menu, select **Update Report** or **Update All Reports** to update the report with the SDB data current at the time the simulation paused.

In a schematic design, an option is available to control all schematic-based analyses. Select **Tools** > **Options** > **General Options** > **General** > **Desktop Performance**. The **Dynamically update postprocessing data during edits** check box controls the version of solution data to use in reports.

Twin Builder solution data is versioned. You might have solution data for a particular state of a design, then edit the design and re-simulate. Solution data is now available for *both* states of the design.

Suppose the following sequence occurs with the **Dynamically update postprocessing data during edits** check box cleared:

- Simulate and create a report. The report displays the current solution data (*state 1*).
- Next, edit the design. The report is marked **Invalid** (a large **X** appears in the legend of the report window and on the report icon in the **Results** section of the Project tree).
- Rerun the simulation. The report is updated with the latest solution data (*state 2*), and the report is marked as valid (the **X** disappears).
- Now click **Undo**. Twin Builder still has the old solution data (*state 1*).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If you clear the **Dynamically update postprocessing data during edits** check box, the *state 2* report data becomes invalid again.

If you select the **Dynamically update postprocessing data during edits** check box, Twin Builder reloads the now valid *state 1* solution data into the plot and marks it valid.

- Upon selecting **Redo**, the result again depends on the setting of the **Dynamically update postprocessing data during edits** check box. If the check box is cleared, the report becomes valid again (the *state 1 data* was not reloaded by the **Undo** operation, so the report still displays *state 2*). If the check box is selected, the *state 2* data is reloaded dynamically and the plot is also valid.

Deleting Reports

To delete a single report, right-click the report under **Results** in the **Project Manager** pane and select **Delete**.

To delete all reports, either select **Twin Builder** > **Results** > **Delete All Reports**, or right-click **Results** in the **Project Manager** pane and select **Delete All Reports**.

Opening Reports

To open a single report, right-click a report under **Results** in the **Project Manager** pane and select **Open Report**.

To open all reports, either select **Twin Builder** > **Results** > **Open All Reports**, or right-click **Results** in the **Project Manager** pane and select **Open All Reports**.

Exporting Plot Data

To export plot results data to a file:

1. Select **Report2D** or **Report 3D** > **Export**. The **Export Report** dialog box opens.
2. Browse to the directory where the data is to be saved.
3. Select the data file format from the **Save as type** drop-down list. Options include:
   - ***.csv** – Comma delimited data files.
   - ***.tab** – Tab delimited data files.
   - ***.dat** – Ansoft PlotData files (2D reports only).
   - ***.txt** – Post processor format files (2D reports only).
   - ***.rdat** – Report data format files.

> **Note:**
>
> You can also export graphic image files of the plot in several formats.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4. Enter the file name.
5. Optionally, for 2D plots, you can choose to **Export Uniform Points**, which enables fields in which you can specify the start, stop, and step values and units to output uniformly spaced point data.
6. Click **Save** to save the data, or click **Cancel** to close the dialog box without saving any data.

Related Topics

Importing 2D Plot Data

Creating a Report from a Report Data File

Report File Formats

Importing 2D Plot Data

To import 2D plot data to a report:

1. Select **Report2D** > **Import Data**. The **Open** dialog box appears.
2. Browse to the directory where the data file is located.
3. Select the file format from the **Files of type** drop-down list. Options include:
   - ***.csv** – Comma delimited data files.
   - ***.tab** – Tab delimited data files.
   - ***.dat** – Ansoft PlotData files.
   - ***.txt** – Post processor format files.
   - ***.rdat** – Report data files.

For more information, see ***Report File Formats***.

4. Select the file name.
5. Click **Open** to import and plot the data, or **Cancel** to close the dialog box without importing any data.

Related Topics

Exporting Plot Data

Creating a Report from a Report Data File

Report File Formats

Report File Formats

This table provides information about the file formats that you can import into plot data reports.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

File Extension	Information
CSV	• Uses a comma as the separation character. • The X-axis value is in the first column. Each curve's Y-values make up one column, and the curves are in the same order as in the plot legend. • The first row is the X-axis name [unit] and the information for each curve, which is the same as the plot curve legend. The name must be **Time** or **F**. • Other rows are the X-axis value and the Y-value for each curve.
TAB	• Uses the tab character as the separation character. • The X-axis value is in the first column. Each curve's Y-values make up one column, and the curves are in the same order as in the plot legend. • The first row is the X-axis name [unit] and the information for each curve, which is the same as the plot curve legend. • Other rows are the X-axis value and the Y-value for each curve.
TXT	• The TXT file must begin with a header that contains: ◦ A first line made of 92 equals signs. ◦ A second line that begins with the company name and ends with the date in MM/DD/YY format. ◦ A third line that begins with a plot name and ends with the time in hh:mm:ss format. ◦ A fourth line, which is empty. ◦ A fifth line made of 92 hyphens. **Note:** Each header line must be 92 characters in length, except the empty line. • Each column has a fixed width, separated by white space. • The X-axis value is in the first column. Each curve's Y-values make up one column, and the curves are in the same order as in the plot legend. • The first row is the X-axis name [unit] and the trace name [unit] for each curve, which is the same as the plot curve legend. • The second row is an empty column and the variable values for each curve, which is the same as the plot curve legend. • Other rows are the X-axis value and the Y-value for each curve.
DAT	DAT is an Ansys-specific format. Ansys recommends that you do not generate files using this format. Files imported with these formats should be exported only from Ansys products.
RDAT	RDAT is an Ansys-specific format. Ansys recommends that you do not generate files using this format. Files imported with these formats should be exported only from Ansys products.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Importing 2D Plot Data

Exporting Plot Data

Creating Custom Report Templates and Defaults

You can edit properties from any report type and save them as a template or as the default. This can save repeated editing of properties (for example, your company's name, color schemes, or plot attribute settings) when you create other reports. You can prepare a template by copying and pasting plot settings from one plot to another of the same display type. Once you create templates, you can access them from the **Results** > **Report Templates** menu and the **Report2D** > **Report Templates** > **Apply Settings** menu.

See ***Modifying the Background Properties of a Report*** for a discussion of format changes you can make to any report.

To save an edited report as a template:

1. In the Project tree, right-click a report and select **Report Templates** > **Save**, **Report2D** > **Report Templates** > **Save**, or **Report3D** > **Save As Template**.

   The **Save As Report** dialog box appears. By default, the directory is your **AnsysEM***version***Win64\userlib\ReportTemplates** directory.

2. Accept the directory, or save to the **syslib** directory.
3. Enter a file name, which is given an RPT extension.

   It is good practice to give the template a descriptive name, showing both the kind of format you begin with (such as rectangular plot or digital plot) and a description of the distinguishing edits (such as for company name, or color scheme). Once saved, this name appears on the **PersonalLib** menu.

   The **Save As Type** field currently supports the Ansoft Report Format (RPT) format.

4. Click **Save** to save the template to the **PersonalLib** menu.

All RPT templates in the directory appear on the **Results** > **Report Templates** > **PersonalLib** menu. Select a report from the **PersonalLib** menu to open a report that you can then modify to add traces or perform other edits. Templates in the **syslib** directory appear on the **Report Templates** menu.

To save an edited report as a default:

In the Project tree, right-click a report and select **Report Templates** > **Save Settings** as default. You can also select **Report2D** > **Report Templates** > **Save Settings** as default or **Report3D** > **Save Settings** as default.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Modifying Reports

Modifying the Background Properties of a Report

Modifying the Legend in a Report

Working with Traces

Editing the Display Properties of Traces

Discarding Report Values Below a Specified Threshold

Exporting Graphics Files from a Plot

You can export plot images in any of the following graphics formats from a plot:

- **JPG** – Joint Photographics Experts Group files.
- **GIF** – Graphics Interchange Format files.
- **BMP** – Bitmap files.
- **TIFF** – Tagged Image File Format files.
- **WRL** – Virtual Reality Modeling Language (VRML) files.

To export a graphics file:

1. Select **Report 2D** or **Report3D** > **Export**. The **Export Report** dialog box appears.
2. Use the file browser to find the directory where you want to save the file.
3. Type the name of the file in the **File name** box.
4. Select the desired graphics file format from the **Save as type** drop-down list.
5. Click **Save**. The file is exported to the specified location as a graphics file.

Related Topics

Exporting Plot Data

Copying Reports to the Clipboard as Images

You can copy images of reports—including plots-on-schematics—to the clipboard, so you can paste them into other applications as follows:

1. Make the report the active window. For a plot-on-schematic, click the plot to select it.
2. Select **Edit** > **Copy Image**, or right-click a report or plot-on-schematic and select **Copy Image** to copy the report to the clipboard as an image.
3. Open the application into which you want to paste the image, and paste the image.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Copy and Paste of Report and Trace Data

Copy and Paste of Report and Trace Definitions

Plotting Imported Solution Data

1. Select **Twin Builder** > **Import SDB File**. A file **Open** dialog box appears.
2. Locate the Twin Builder solution SBD file containing the solution to import and click **OK**.
3. If the imported data is in table format, click **Table Data** on the **Import Data** dialog box. The **Data Table Import** dialog box opens.
4. Locate the data table file and click **OK**.

To plot imported solution data:

1. Expand **Analysis** in the Project tree. Verify the presence of the imported solution data.
2. Right-click **Results** in the Project tree and select **Create Standard Report** > *<report type>*.
3. On the **Report** dialog box, select the imported solution from the **Solutions** drop-down menu.
4. Select the desired data in the **Traces** tab and click **New Report**. The **Report** dialog box opens to display the imported solution.
5. Select and add traces or other reports based on the imported solution data as needed. When finished, click **Close** to dismiss the **Report** dialog box.

Using Animation

The following sections describe how to postprocess and view various animated reports.

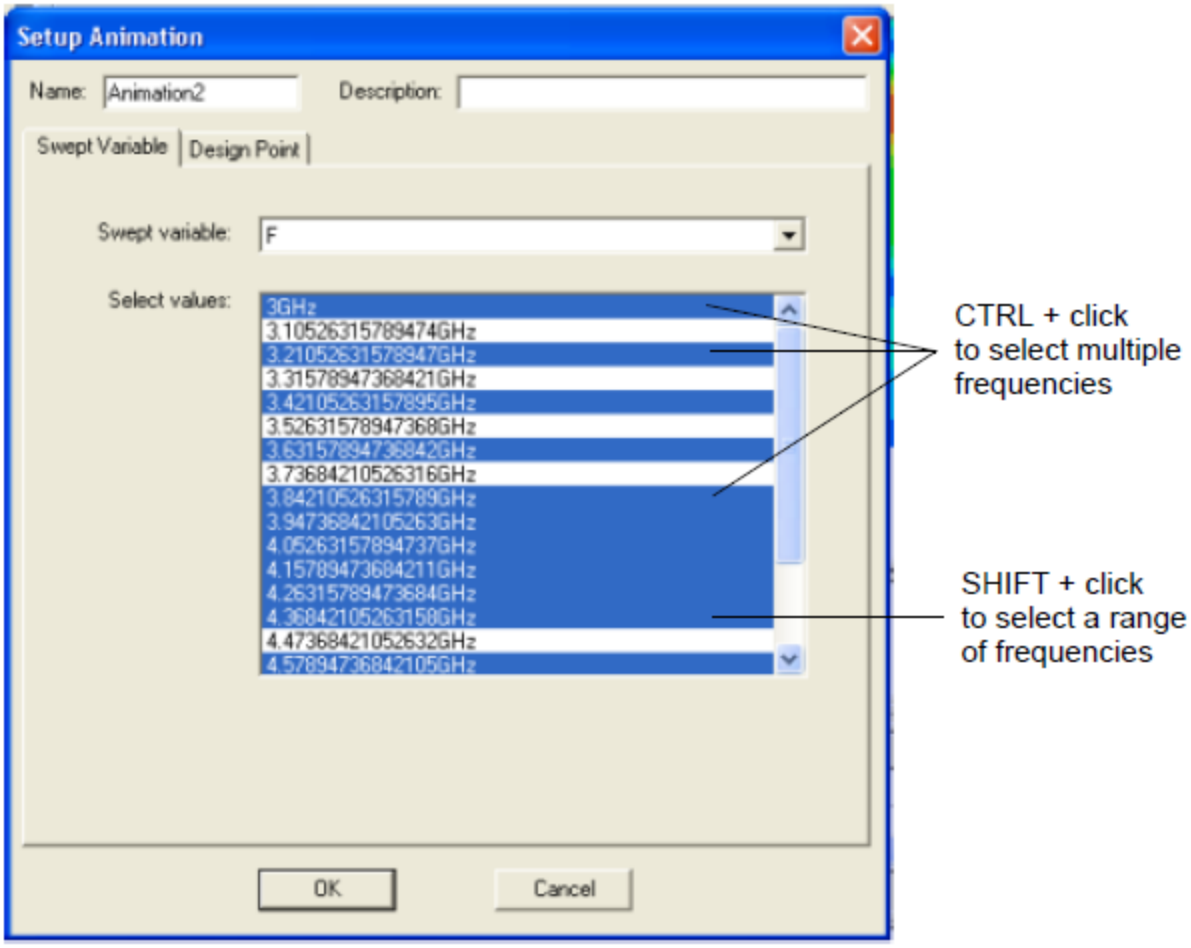
- Frequency Animation
- Changing the Design Point

Frequency Animation

1. To initiate the animation of the plot that is currently displayed:
   - Select **Animate** from the **View** menu, or
   - Expand the **Results** icon in the **Project Manager** pane, right-click the plot entry, and select **Animate**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. If no animations have been defined previously, the **Setup Animation** dialog box opens:



- Specify a name in the **Name** field (or accept the default, **Animation***n*, where *n* is a numeral). Optionally, enter a description.
- For a frequency animation, select **F** as the **Swept Variable**.
- Click **OK**.

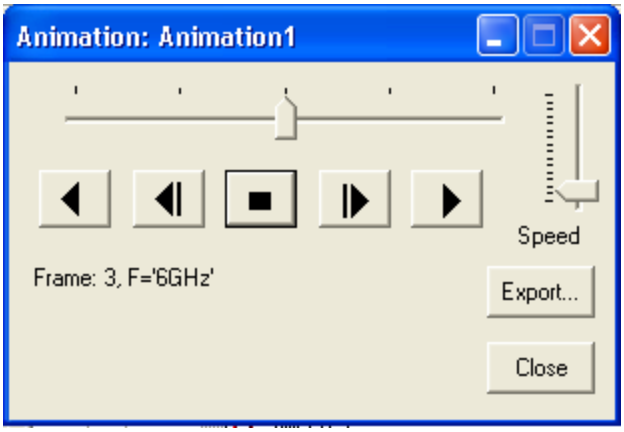
If one or more animations are defined, select **Animate** from one of the menus to open the **Select Animation** dialog box.

The **Select Animation** dialog box provides the following operations:

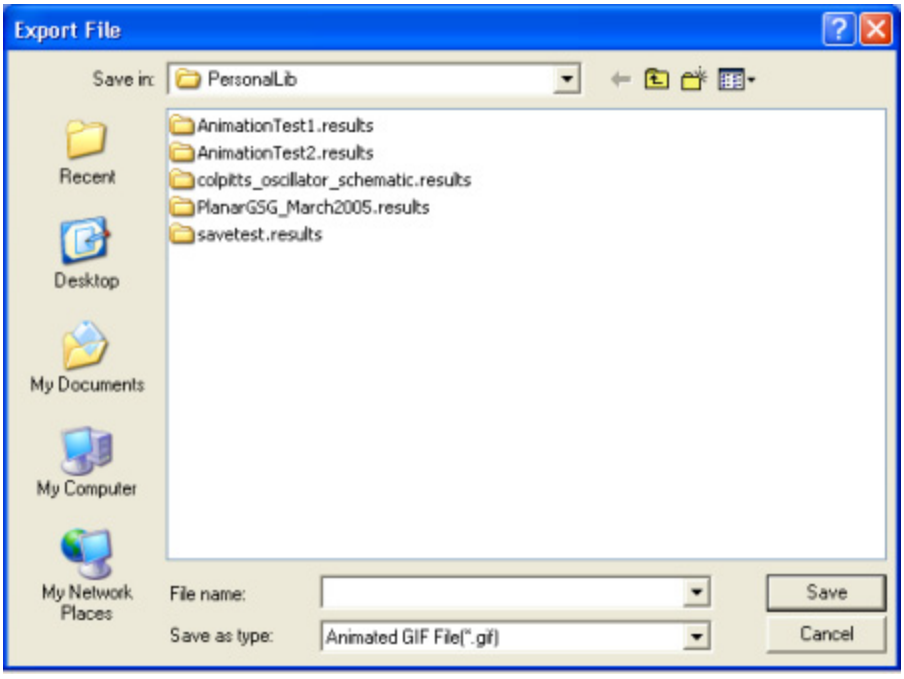
- Select an animation and click **OK** to start it.
- Click **New** to open the **Setup Animation** dialog box described above and close the **Select Animation** dialog box.
- Select an animation and click **Delete** to delete the definition.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3. Twin Builder calculates the frame data. If the **Progress** window is active, you can monitor the progress of the calculation. When the frames have been calculated, the animation begins and the **Animation** control panel opens:



- Use the buttons to play the animation. Use the **Speed** slider to control the speed of the animation.
- To export the frame data to a file, click **Export**. The **Export File** dialog box opens.



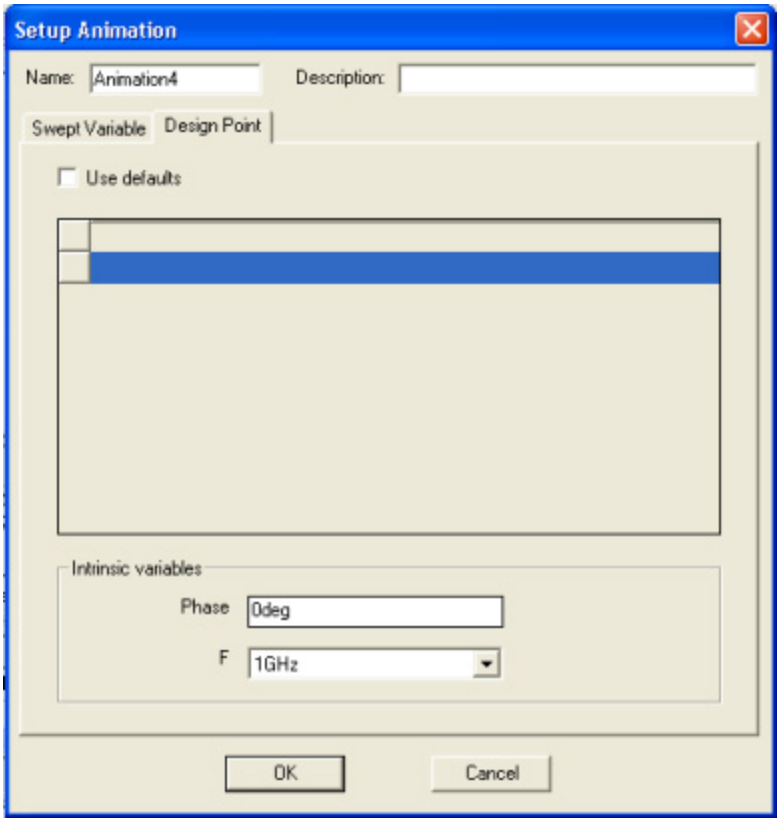
Specify the directory and file name. Use the **Save as type** menu to select the file format (Animated GIF or AVI). Click **Save** to save the data and close the dialog box.

- Click **Close** on the **Animation** control panel to stop the animation and close the panel.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Changing the Design Point

Calculating frames for an animation is equivalent to re-simulating the design. You can specify design point parameters for the animation calculations that are different from the ones used in the original simulation. On the **Setup Animation** dialog box, select the **Design Point** tab and clear the **Use defaults** check box. The following fields display:



Make any desired changes, then click **OK** to apply the changes and close the dialog box. Click **Cancel** to close the dialog box without making any changes.

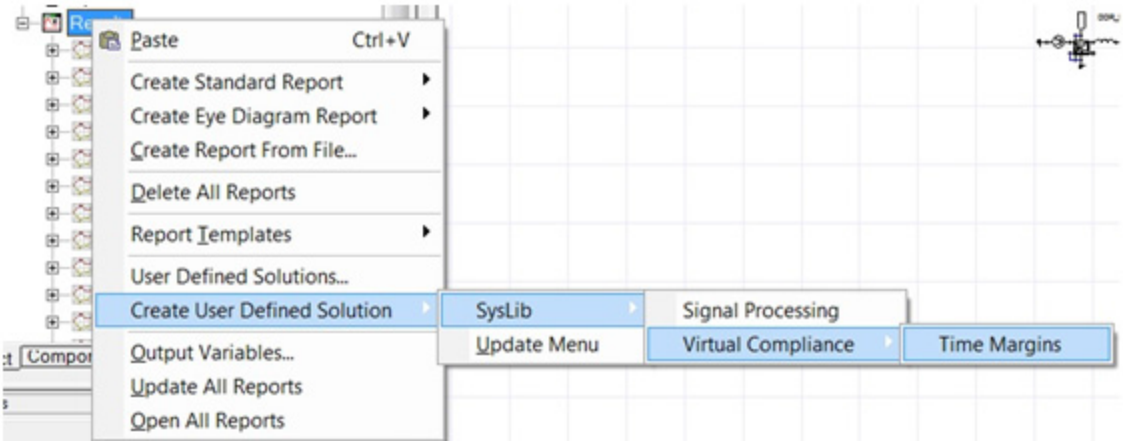
User Defined Outputs: Introduction

User defined outputs (UDOs) let you define calculations through IronPython scripts or any .NET language (and used by the IronPython script). The UDO scripts must be in the **UserDefinedOutputs** directory under either of **syslib**, **userlib**, or **Personallib** with any directory structure needed for organization. (The **Lib** directory name is special and its purpose will be explained in a subsequent section.)

The UDO scripts in **syslib/UserDefinedOutputs**, **userlib/UserDefinedOutputs**, or **Personallib/UserDefinedOutputs** become available to you to create user-defined solutions

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

through the **Results** > **Create User Defined Solution** menu. Use **Results** > **Create User Defined Solution** > **Update Menu** to refresh the menu to include the new UDO scripts that might have been copied to **syslib**, **userlib**, or **Personallib**, or exclude them if they have been deleted, after the launch of desktop. Once the user-defined solution is created, the solution and the calculations defined by UDO become available in Reporter as any other quantities in a new User Defined report type.



Related Topics

User Defined Outputs: Python Script API

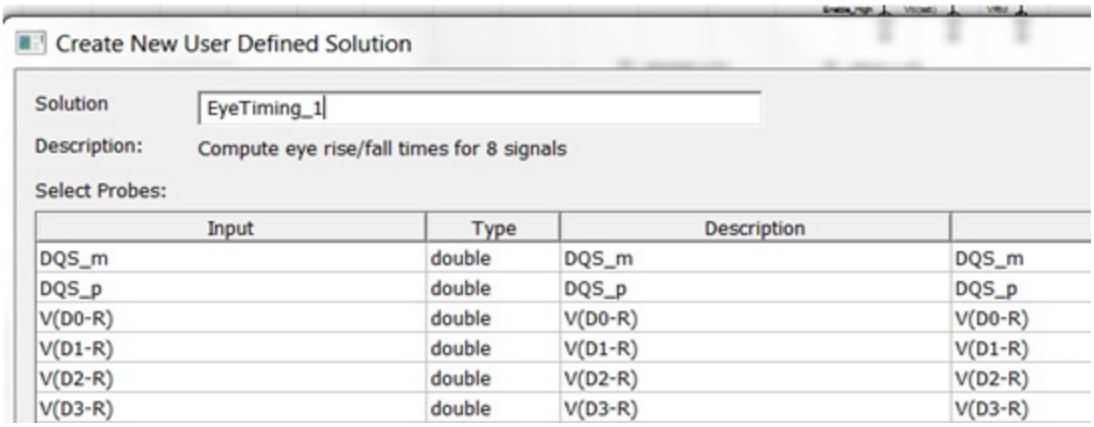
User Defined Outputs: Script Organization

Named Probes and Properties in User Defined Outputs

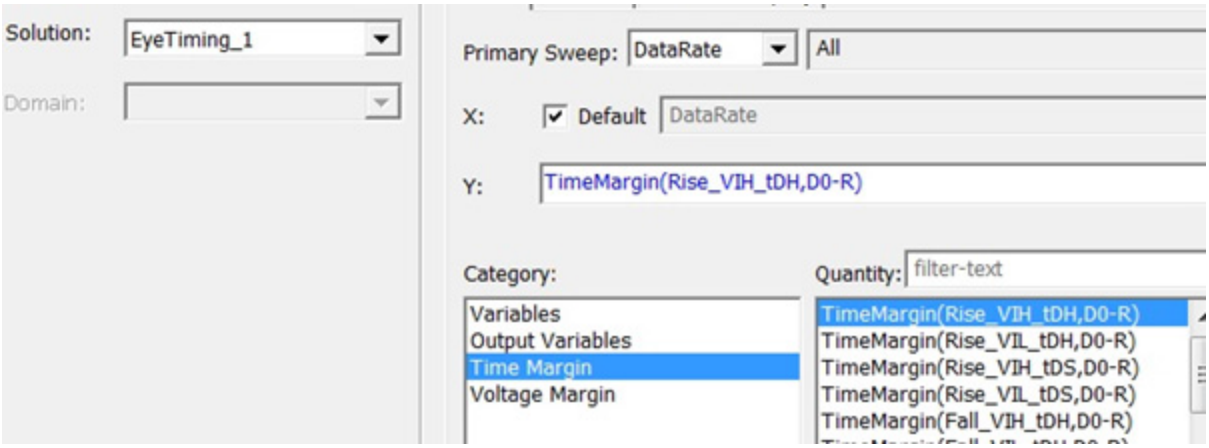
UDOs allow processing data across traces, solutions, and report types. A UDO specifies the named probes and properties for which you select/enter the values at the time of creation of user-defined solution. Probes are very similar to traces except that you select the values of only intrinsic variables for probes. The values of design/project variables are selected when a trace is created based upon the user-defined solution in reporter.

For example, you could create a user defined solution called **EyeTiming_1**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



You can then access this solution in the Reporter.



Computation of Traces Based UDO Calculations

When traces based upon UDO outputs are computed, the data for probes is computed and passed to the UDO script for each design variation. Along with the probe data, the values of properties you entered are also passed. The information about the UDO calculations that need to be computed is also made available. The UDO then performs the computation and passes the results to reporter. Note that UDOs can compute and pass back more calculations than have been requested at that point of time. This lets UDOs compute a set of calculations that take almost same amount of computational resources as any one calculation in that set and cache that with reporter.

> **Note:**
>
> When you plot those calculations, reporter will use the cached results instead of invoking the computation on UDO.

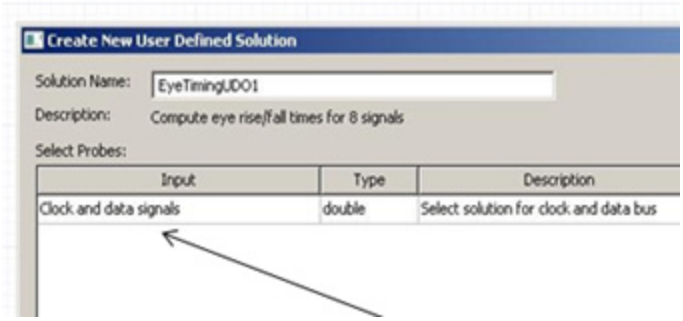
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Dimensions Reduction by UDO Calculations

The probes in a UDO can have heterogeneous dimensions of data. For example, one probe in a UDO can have data that is function of **n** intrinsic variable, while another probe in same UDO can have data that is a function of **m** intrinsic variables, with **n** and **m** potentially being different. UDOs allow reducing any number of these intrinsic variables. For example, in the above example, UDO calculations can be a function of any number of intrinsic variables including not being a function of any intrinsic variable at all. UDO calculations can also be a function of an intrinsic variable that none of the probes is function of. **The only restriction is that Freq cannot be reduced if any of the probes are on a Fields report type.**

Dynamic Probes

In addition to named probes and properties, UDOs can specify named dynamic probes. The difference between probes and dynamic probes is that while the end user of a UDO specifies the complete trace definition for probe, the expression for dynamic probe is specified by UDO code itself and not by the end user. This lets UDOs access the data for probes without having the end user enter each individual probe. For example, a UDO can access data for a huge S matrix for a 100 port design without requiring the end user to enter the probe information for each of those 10,000 quantities. Each dynamic probe is associated with a named probe that you enter; information about solution, context, and intrinsic variables is used from user-selected probe. However, multiple dynamic probes can be associated with the same user-selected probe. The dynamic probes are enquired from UDOs at the time of trace computation and not at the time of creation of user-defined solution.



This means that you select solution, context, and values of intrinsic variables just once; the same information is used (in this case) for all clock and data signals. The expression for those signals comes from the UDO code.

User Defined Outputs: Python Script API

A user defined output (UDO) extension is implemented as an IronPython script that defines a class with a specific name: **UDOExtension**, which derives from a specific base class **IUDOPluginExtension** and implements its abstract methods.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

User Defined Outputs: An Introduction

User Defined Outputs: Script Organization

UDO Extension IMPLEMENTATION

The purpose, argument list and expected return types for each of the **IUDOPluginExtension** abstract methods—which the UDO author is expected to implement—are described below.

Import Statements

UDOExtension Class

IUDOPluginExtension Abstract Class

Import Statements

The base class to be used and the types it uses in turn are contained in .NET assemblies. The use of these requires that the assemblies be imported into the UDO script. These import statements should be added to the top of the python script:

```
from Ansys.Ansoft.ModulePluginDotNet.Common.API import *

from Ansys.Ansoft.ModulePluginDotNet.Common.API.Interfaces import *

from Ansys.Ansoft.ModulePluginDotNet.UDO.API.Interfaces import *

from Ansys.Ansoft.ModulePluginDotNet.UDO.API.Data import *
```

DOExtension Class

The UDO should be implemented as an IronPython class called ***UDOExtension*** which **must** derive from the ***IUDOPluginExtension*** abstract base class (from the **Ansys.Ansoft.ModulePluginDotNet.UDO.API.Interfaces** namespace).

Note that power users could derive a class hierarchy tuned toward a specific type of UDO and that they can derive from their own base classes. The only requirement is that directly or indirectly, the UDO class must derive from ***IUDOPluginExtension.***

Example:

```
  def BaseClassUDO ((IUDOPluginExtension):

  #base class implementation

          …
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
def UDOExtension ((BaseClassUDO):

#UDO class implementation

…
```

IUDOPluginExtension Abstract Class

The implementation of the **IUDOPluginExtension** class is described in this section using a simple UDO example that expects a single probe and reduces its dimension returning as its outputs, the maximum, minimum, and average of its input probe data. The script in its entirety will also be listed later on.

Required functions:

The **IUDOPluginExtension** abstract class declares the following abstract methods that must be implemented in the **UDOExtension** class or one of its base classes. Not implementing any of these methods will result in a run-time error and a non-functioning UDO.

GetUDSName()

GetUDSDescription()

GetUDSSweepNames()

GetCategoryNames()

GetQuantityNames(string categoryName)

GetQuantityInfo(string quantityName)

GetInputUDSParams(List<UDSProbeParams> udsParams,

GetDynamicProbes(List<UDSDynamicProbes> dynamicProbes);

Compute(IUDSInputData inData,

GetUDSName()

- **Purpose** – Return a string used as a prefix for all solution instances created using this UDO.
- **Returns** – String.

Example:

```
def GetUDSName(self):

  return "MinMaxAvg
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetUDSDescription()

- **Purpose** – Returns a description for the UDO, including its purpose. Used in multiple UDO-related dialog boxes in the application to describe the UDO.
- **Returns** – String.

Example:

```
def GetUDSDescription(self):

   return "Sample UDO for dimension reducing quantities
```

GetUDSSweepNames()

- **Purpose** – Returns a list of sweep names used for the solution generated by the UDO. These will appear in the sweeps list displayed in the standard reporter dialog box when used to create reports from the solution generated by the UDO.
- **Returns** – List of strings. If the UDO outputs have no sweeps, return the empty list [].

Example:

```
# Returns list of sweeps names

# We have no sweeps as we reduce them.

def GetUDSSweepNames(self):

   return []
```

GetCategoryNames()

- **Purpose** – The outputs that the UDO solution provides/generates can be classified into multiple categories (like how the application does as displayed in the report creation dialog box). These appear in the categories box in the dialog box when creating reports from the UDO-generated solution data.
- **Returns** – List of strings.

Example:

```
def GetCategoryNames(self):

   return ["UDOOutputs"]
```

GetQuantityNames(string categoryName)

- **Purpose** – For each of the category names returned from the **GetCategoryNames** method, this function is called to return a list of quantities to be organized under that category name. **Note that the quantity names must be unique across the**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

categories; that is, no two categories can have quantities with the same name.

- **Parameters**
  - **categoryName** (input python string) – Category name.
- **Returns** – Python list of strings.

Example:

```
# returns a list of quantity names for the supplied category name
def GetQuantityNames(self, catName):
   if catName == "UDOOutputs":
        return ["min_val", "max_val", "avg_val"]
    else:
         return []
```

GetQuantityInfo(string quantityName)

- **Purpose** – For each quantity that the UDO creates, it must also describe the quantity (unit and other details). This method is called for each quantity name (across all categories) as returned from an earlier call of the **GetQuantityNames** method.
- **Parameters**
  - **quantityName** (input string) – Quantity name.
- **Returns** – Object of type QuantityInfo.

Example:

```
# Returns an instance of QuantityInfo for the qtyName supplied or
None if such a
# quantity could not be found
def GetQuantityInfo(self, qtyName):
   # All the quantities we have are simple doubles
   # we can leave them unitless
   return QuantityInfo(Constants.kDoubleParamStr)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetInputUDSParams(List<UDSProbeParams> udsParams,

IPropertyList propList,

List<UDSProbeParams> userSelectionForDynamicProbes)

- **Purpose** – This is the main definition part of the UDO. The supplied arguments populate details of the parameters to which the UDO user will specify value, specify the probe names and their types, as well as the dynamic probe selections.
- **Parameters**
  - **udsParams** – .NET list of **UDSProbeParams** objects. The UDO script is expected to add one instance of **UDSProbeParams** for each probe definition it wants displayed. The UDO user will, when creating the UDO solution, assign a matching quantity to each such probe.
  - **propList** – **IPropertyList** object. The **propList** object adds properties that should be displayed to the user for data collection. These properties with the user-supplied values are returned to the UDO script in the **Compute** methods.
  - **userSelectionForDynamicProbes** – .NET list of **UDSProbeParam** objects.
- **Returns** – Boolean. True on success, false on failure.

Example:

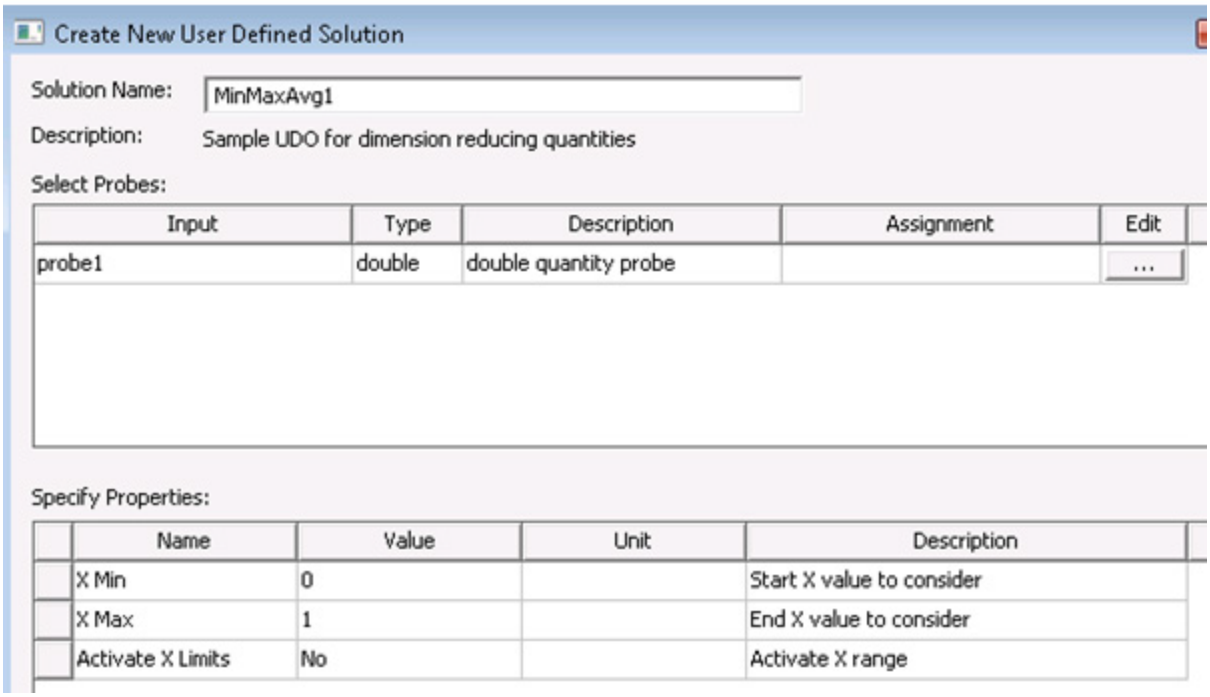
```
  # Returns list of UDSParams and list of dynamic properties

  # Adds setup time properties to the propList

  def GetInputUDSParams(self, udsParams, propList,
 userSelectedDynamicProbes):


    # Add the probes. We need only one double quantity

    param1 = UDSProbeParams("probe1",

"double quantity probe",

Constants.kDoubleParamStr,

"", "")

    udsParams.Add(param1)


    # Add the properties we want the user to supply

    # In this case, we will ask for a start/end range for
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

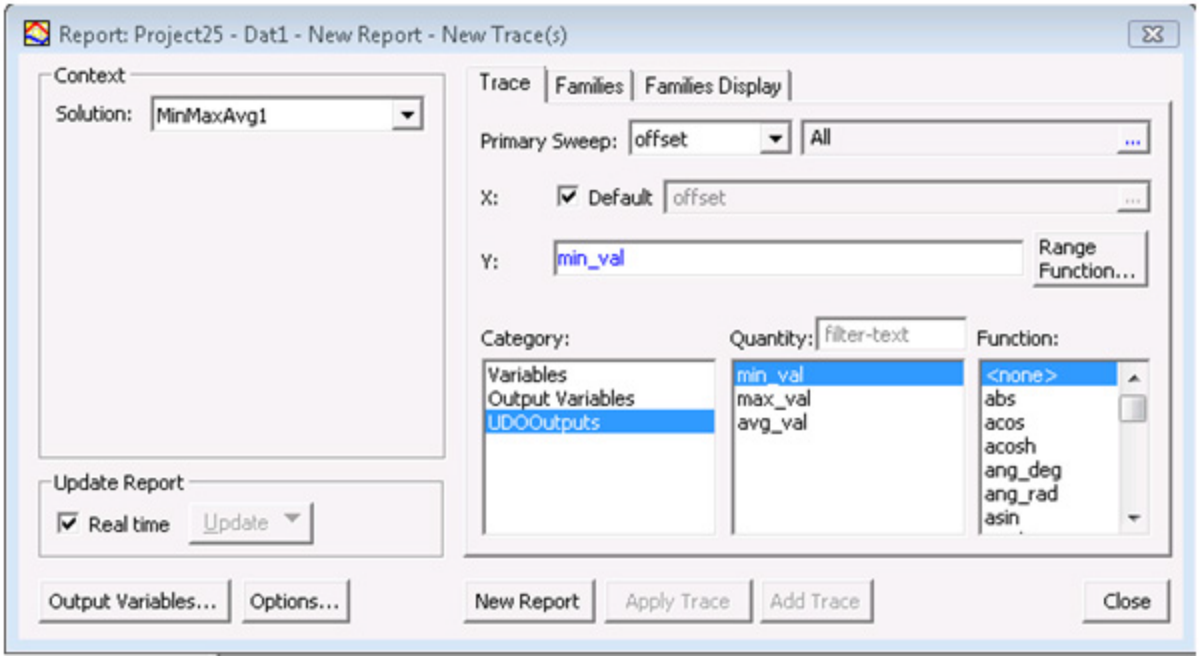
```
    # X parameters. Since we cannot reasonably provide defaults
    # as we have no idea what the sweep limits will be, we will
    # also ask if the limits are to be activated.
    prop = propList.AddNumberProperty("X Min", "0")
    prop.Description = "Start X value to consider"

    prop = propList.AddNumberProperty("X Max", "1")
    prop.Description = "End X value to consider"

    # For menus, the first option is the default.
    prop = propList.AddMenuProperty("Activate X Limits", ["No",
"Yes"])
    prop.Description = "Activate X range"

    return True
```

The above function results in the following dialog box when you select **Reports** > **Create User Defined Solution**. The mapping from the **UDSParams** and the properties to the GUI elements should be unambiguous. The name and description of the UDS also appear in this dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



When a report is created from the UDO dialog box, the category/quantity names specified by the UDO are used as shown below:



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetDynamicProbes(List<UDSDynamicProbes> dynamicProbes);

- **Parameters** – **dynamicProbes**: .Net list of **UDSDynamicProbes objects**. Output parameter.
- **Returns** – True on success, false on failure.

Example:

```
# Returns list of UDSParams and list of dynamic properties
# output UDSDynamicProbeCollection probes
def GetDynamicProbes(self, probes):
   pass
```

Compute(IUDSInputData inData

IUDSOutputData outData,

IPropertyList propList,

IProgressMonitor progressMonitor)

- **Purpose** – This is the main computation method which generates the data for the quantities that make up the UDO solution.
- **Parameters**
  - ***inData*** – **UDSInputData** object. Used to get the input probe data.
  - ***outData*** – **UDSOutputData** object. Used to set the UDO solution quantity and sweep data.
  - ***propList*** – **IPropertyList** object. Used to get the user-entered values for each of the properties defined during the **GetInputUDSParams** call.
  - ***progressMonitor*** – **IProgressMonitor** object. This can be used to set progress for long running calculations, check for user initiated abort, and so on.
- **Returns** – True on success, false on failure.

The data is received from the UI using the ***IUDSInputData*** API. It is processed and the result data is sent to the UI using the ***IUDSOutputData*** API.

Example:

```
# IUserDefinedSolutionHandle API implementation.
# Calculates output values and sets them using
IUDSInputData/IUDSOutputData API.
def Compute(self, inData, outData, propList, progMon):
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
# Get the sweeps associated with the probe and validate

# use the probe name that we had defined earlier

sweeps = inData.GetSweepNamesForProbe("probe1")

if( sweeps == None or sweeps.Count > 1):

  AddErrorMessage(self.GetName() + "Unexpected sweep count 0 or > 1
  in Compute")

   return False



# Get the data associated with our probe

probeData = inData.GetDoubleProbeData("probe1")

sweepData = inData.GetSweepsDataForProbe("probe1", sweeps[0])


# Get the user-specified properties.

# Note that ideally, these "X Min" etc names should be written as

# constant membets and referred to in both the GetInputUDSParams

# and in Compute to reduce the change of typos.

useXRangeProp = propList.GetMenuProperty("Activate X
Limits").SelectedMenuChoice

xRangeStart = propList.GetNumberProperty("X Min").ValueSI

xRangeEnd = propList.GetNumberProperty("X Max").ValueSI


# At this stage, one can look at the RequestedQuantities and create

# a dictionary to later check against. However, I am simply
computing

# all the quantities.

minVal = 0

maxVal = 0
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential

information of ANSYS, Inc. and its subsidiaries and affiliates.

```
   avgVal = 0

   # Check if we need to perform range computation
   if useXRangeProp == "Yes":
           seenAny = False
          avgSum = 0
          count = 0

     # zip is used since we also need to pull in sweep data
     # an index and the array notation could also have been used
     for probeVal, sweepVal in zip(probeData, sweepData):
     if sweepVal < xRangeStart or sweepVal > xRangeEnd:
pass

     # Note that in a better written script, this code would be
     # refactored into it's own function to avoid code
     # duplication
     if not seenAny:
minVal = probeVal
maxVal = probeVal
avgSum = probeVal
seenAny = True
count = 1
     else:
if probeVal < minVal:
minVal = probeVal

if probeVal > maxVal:
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
maxVal = probeVal


avgSum += probeVal
count += 1


          if seenAny:
     avgVal = avgSum/count


     else:
          seenAny = False
          avgSum = 0
          for probeVal in probeData:
      if not seenAny:
minVal = probeVal
maxVal = probeVal
avgSum = probeVal
seenAny = True
     else:
if probeVal < minVal:
minVal = probeVal


if probeVal > maxVal:
maxVal = probeVal


avgSum += probeVal


          if seenAny:
     avgVal = avgSum/probeData.Count
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
# Finally set the output values. Note that these are always set
as
# lists even if we have just one item.
outData.SetDoubleQuantityData("min_val", [minVal])
outData.SetDoubleQuantityData("max_val", [maxVal])
outData.SetDoubleQuantityData("avg_val", [avgVal])

# And we are done.
return True
```

Optional Functions in IDO Extension Abstract Class

The following functions, while a part of the IUDOExtension abstract class, have meaningful default implementations and are optional. However, you can overridde them to take advantage of advanced functionality.

Validate(List<string> errorStringList,

```
List<UDSProbeParams> udsProbParams,
IPropertyList propList,
List<UDSProbeParams> userSelectionForDynamicProbes)
```

- **Purpose** – This method validates your choices. You can check the values of the properties entered, the probes, and so on for suitability.
- **Parameters**
  - **udsProbParams** – C# list of **UDSProbeParams** objects.
  - **propList** – _IPropertyList object.
  - **userSelectionForDynamicProbes** – C# list of **UDSProbeParams** objects.
  - **errorStringList** – C# list of python strings. Output parameter. Should be set only if validation failed; ignored if validation is successful. One error string should be set per each validation error.
- **Returns** – True on validation success, False on failure.
- **Default implementation** – Always returns true.

Example:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
def Validate(self, errorStringList,probeList,propList,
dynamicProbes):
  if probeList == None or probeList.Count == 0:
        errorStringList.Add("Empty probe list")
       return False
  return True
```

Data Types Used in Python Script

There are several types you must use while authoring the Python script. Some of them pass data from the UI to the Python script and to provide an interface for working with this data. Some pass data from the Python script to the UI.

To pass data from the Python script to the UI, the objects of the C# class must be created in Python script using their C# constructors. Then they can be set as functions return values or set to the output parameters using their API.

Constants class

kTraceTypeStr – String constant used to specify an input of trace type.

kSolutionTypeStr – String constant used to specify an input of solution type.

kNumberTypeStr – String constant used to specify an input of number type.

kTextTypeStr – String constant used to specify an input of text type.

kBoolTypeStr – String constant used to specify an input of Boolean type.

kStandardReportStr – String constant to specify a standard report.

kEyeDiagramReportStr – String constant to specify an eye diagram report.

kUserDefinedReportStr – String constant to specify a user-defined report.

kSweepDomainStr – String constant to specify the sweep domain.

kTimeDomainStr – String constant to specify the time domain.

UDDInputParams class

The objects of this class must be created in Python script in the **GetUDDInputParams()** function and the **SetUDDInputParams()** function.

Attributes

Input Name (string)

Input Description (string)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Input Type (Can be Boolean, Number, Text, Trace, or Solution) (string)

BoolData (Boolean)

DoubleData (double)

TextData (string)

ReportType (string)

SolutionName (string)

DomainName (string)

Constructors

UDDInputParams (string name, string description, string type)

UDDInputParams (string name, string description, string type, bool data)

UDDInputParams (string name, string description, string type, double data)

UDDInputParams (string name, string description, string type, string data)

UDDInputParams (string name, string description, string type, string reportType, string solutionName, string domainName)

Property Accessors

Name – Get/Set the name of an input.

Description – Get/Set the description of an input.

Type – Get/Set the type of an input.

BoolData – Get/Set the data of a Boolean input.

DoubleData – Get/Set the data of a number input.

TextData – Get/Set the data of a text input.

ReportType – Get/Set the report type.

SolutionName – Get/Set the name of the solution.

DomainName – Get/Set the name of the domain.

IProgressMonitor Abstract Class

The object of this class is a progress monitor. It is used to display calculations progress in the UI and check if you have requested an abort of the computation.

When displayed in the application, each progress message has four items:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- A task name.
- A sub-task name.
- The progress amount.
- A button to abort the task in progress.

All of this functionality and abort interaction is achieved using these functions:

SetTaskName (string taskName):

SetSubTaskName (string subTaskName)

BeginTask (string name)

SetTaskProgressPercentage(int progressPercent)

CheckForAbort() – If the generated quantities are computationally expensive, the UDO author can periodically call this method, then call EndTask with Fail and return False.

EndTask (bool passFail)

Example:

```
progMon.BeginTask("Process DQS")

progMon.SetSubTaskName("Compute UI segments")

progMon.SetTaskProgressPercentage(33)

progMon.SetSubTaskName("Compute the rest")

progMon.SetTaskProgressPercentage(100)

progMon.EndTask(True)
```

IUDSInputData

The purpose of this class is to get data (probe and sweep) from Ansys Electronics Desktop.

Examples in this section are just to show proper syntax of the function calls. For actual usage of the class, see Compute function example.

GetDoubleProbeData(probeName)

GetSweepsDataForProbe(probeName, sweepName)

GetComplexProbeData(probeName)

GetSweepNamesForProbe(probeName)

GetRequiredQuantities()

GetVariableValues()

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetInterpolationOrdersData(probeName);

GetDoubleProbeData(probeName)

- **Purpose** – This is the primary mechanism by which the UDO script obtains the probe data (as double precision values) for its compute process.
- **Parameters**
  - **probeName** – String representing the probe name for which data is requested. This has to be one of the many probes supplied during a call to the UDO's **GetInputUDSParams** method.
- **Returns** – .NET double array of data for the specified probe if the probe exists or null if the probe is unknown.

Example:

```
# doubleData is a list of floats

doubleData = inData.GetDoubleProbeData("probe1")
```

GetSweepsDataForProbe(probeName, sweepName)

- **Purpose** – All supplied probe data is associated with one or more sweep quantites (that is, intrinsic quantities like time, frequency, Theta, Phi, and so on).
- **Parameters**
  - **probeName** – Probe name for which want the sweep data.
  - **sweepName** – Sweep name.
- **Returns** – .NET double array of data for the specified probe and sweep.

Example:

```
# sweepData is C# Array of doubles (floats in python)

sweepData = inData.GetSweepsDataForProbe("FarFieldsProbe","Freq"])
```

GetComplexProbeData(probeName)

- **Purpose** – The primary mechanism by which the UDO retrieves data for its input probes (if it expects complex data for the probe).
- **Parameters**
  - **probeName** – Probe name for which complex data is requested.
- **Returns** – .NET double array (float in Python) of data for the specified probe. Each pair of floats represent one complex number; the first value is for the real part, the second value is for the imaginary part. For instance, array [10.0, 0, 5.1, 2.1] represents 2 complex numbers: (10.0, 0) and (5.1, 2.1).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Example:

```
# complexDataAsDouble is C# Array of doubles (floats in python)
# each pair of floats represents one complex number
complexDataAsDouble = inData.GetComplexProbeData("FarFieldsProbe")
# creating a list of complex numbers from complexDataAsDouble array
complexData = []
if complexDataAsDouble != None:
  for i in xrange(0,complexDataAsDouble.Count , 2):
    complexData.append(complex(complexDataAsDouble
    [i],complexDataAsDouble[i+1]))
```

GetSweepNamesForProbe(probeName)

- **Purpose** – To obtain the list of sweep quantity names associated with a given probe. This also indicates the dimensionality of the data. One name implies that the probe-data is 2D (probe quantity vs sweep quantity) and two names implies 3D data (probe quantity vs Sweep 1 X Sweep 2).
- **Parameters**
  - **probeName** – Probe name.
- **Returns** – .NET IList<*string*> - list of sweep names for the current probe name.

Example:

```
# sweepNames is C# Array of strings
sweepNames = inData.GetSweepNamesForProbe("FarFieldsProbe")
```

GetRequiredQuantities()

- **Purpose** – A given UDO can specify that it provides one or more computed quantities. You might choose to create a report from only a few among the various available UDO outputs. This function returns that list of the UDO output quantities. Only these need be computed in the UDO's compute method.
- **Returns** – .NET IList<*string*> - List of required quantity names.

Example:

```
# quantities is C# Array of strings
quantities= inData.GetRequiredQuantities()
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetVariableValues()

- **Purpose** – This allows the UDO to obtain the names and values of all the design variables for which the UDO quantities are being requested.
- **Returns** – .NET IDictionary<*string,string*> of key-value pairs for variables. Both key and value are strings.

Example:

```
# theDict is C# Dictionary<string, string>
theDict = inData.GetVariableValues()
if theDict != None:
  #varPair is of .Net KeyValuePair type
  for varPair in theDict:
       varName = varPair.Key #string
       varValue = varPair.Value #string
```

GetInterpolationOrdersData(probeName);

- **Purpose** – Returns the interpolation orders that are associated with the probe-data. The probe data is specified at each value of the various sweeps. Any value in between the sweep data points, can use the interpolation data to get a possibly more accurate (compared to linear interpolation) inter-sweep value.
- **Parameters**
  - **probeName** (input Python string) – Probe name.
- **Returns** – NET byte array of interpolation order for the specified probe. These are to be treated as 8-bit signed integers (that is, their values range from 0-127).

Example:

```
# interData is C# Array of bytes (integers in python)
interData = inData.GetInterpolationOrdersData(kProbeNames[0])
  for interValue in theDict:
       order = interValue # interValue and order are integers
```

IUDSOutputData

This type is a twin of the IUDSInputData; it is used to store the values computed by the UDO's compute method.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Examples in this section show proper syntaxis function calls. For actual usage of the class, see the Compute function example.

SetSweepsData(sweepName, sweepData)

SetDoubleQuantityData(qtyName,qtyData)

SetComplexQuantityData(qtyName, qtyData)

SetSweepsData(sweepName, sweepData)

- **Purpose** – Each quantity that is computed by the UDO can be associated with a sweep. If it is, the values that make up the sweep's data points must be specified using this call.
- **Parameters**
  - **sweepName** (string) – Sweep name.
  - **sweepData** (Python list of floats) – Sweep data for the specified sweep.
- **Returns** – True on success, False on failure.

Example:

```
sweepList = [12.3, 14.5, 16.7]
outData.SetSweepsData("Freq", sweepList)
```

SetDoubleQuantityData(qtyName,qtyData)

- **Purpose** – This method is used to record the computed quantity data for each output that is computed. Unless all the sweeps are reduced, this should be used in conjunction with **SetSweepsData**.
- **Parameters**
  - **qtyName** (string) – Quantity name.
  - **qtyData** (Python list of floats) – Quantity data for the specified quantity.
- **Returns** – True on success, False on failure.

Example:

```
doubleList = [12.3, 14.5, 16.7]
outData.SetDoubleQuantityData("V1PlusV2", doubleList)
```

SetComplexQuantityData(qtyName, qtyData)

- **Purpose** – If the quantity computed is complex, use this method to set the quantity values. Any sweep values must be set separately via the **SetSweepsData** method.
- **Parameters**
  - **qtyName** (string) – Quantity name.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **qtyData** (Python list of floats) – Quantity data for the specified quantity. Complex numbers are passed as pairs of floats.
- **Returns** – True on success; False on failure.

Example:

```
doubleFromComlexList=[]
comlexList = [(1+1j), (2+4j), (9.1+3.2j)]
for aComplex in comlexList:
  doubleFromComlexList.append(aComplex.imag)
  doubleFromComlexList.append(aComplex.real)
outData. SetComplexQuantityData ("V1PlusV2", doubleFromComlexList)
```

Working With Properties for UDO

A property is the unit for collecting user input to influence the UDO's Compute. These are initially set up when the UDOs **GetInputUDSParams** method is called and are retrieved in the UDO's Compute method.

There are three supported property types that can be used in the UDO script:

- **INumberProperty** – Specify number properties (with unit support).
- **IMenuProperty** – Select from a list of options.
- **ITextProperty** – Enter text.

The IPropertyList type implements a collection for these properties.

IPropertyList Abstract class

IProperty Abstract class

INumberProperty Abstract class

ITextProperty Abstract class

IMenuProperty Abstract class

IPropertyList Abstract class

Attributes

- **AllProperties** (**IEnumerable**<*IProperty*> - see **IProperty**)
- **NumProperties** (int)

Functions

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **GetProperty(string propName)** – Returns a named property as an **IProperty**.
- **GetMenuProperty (string propName)** – Returns the named property as an **IMenuProperty**.
- **GetTextProperty (string propName)** – Returns the named property as an **ITextProperty**
- **GetNumberProperty (string propName)** – Returns the named property as an **INumberProperty**.
- **DeleteProperty (string propName)** – Deletes an already added named property.
- **AddNumberProperty(string name, string numberWithUnits)** – Adds a new number property. The system overwrites any existing property with the same name.
- **AddTextProperty(string name, string textValue)** – Adds a new named text property with the supplied value. The system overwrites any existing property with the same name.

- **AddMenuProperty(string name, IList**<*string*>**menuChoices)** – Creates a new named menu property with the supplied list of choices. The default selection is set to item 0 (the first item). The system overwrites any existing property with the same name.

IProperty Abstract class

Attributes

- **Name** (string)
- **Description** (string)
- **PropType** (read-only **EPropType** – see Constants)

Constructor

- **IProperty**(string name, **EPropType** type)

The class is a base class for **INumberProperty**, **IMenuProperty**, and **ITextProperty**.

INumberProperty Abstract class

Base class

- abstract class **IProperty**

Attributes

- **ValueSI** (read-only double)
- **ValueInUnits** (read-only double)
- **Units** (read-only string)
- **HasUnits** (read-only bool)

Constructor

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **INumberProperty**(string name)

Functions

- **Set**(string **numberWithUnits**)
- **SetDouble**(double number, string **unitString**)

ITextProperty Abstract class

Base class

- abstract class **IProperty**

Attributes

- **Text** (string)

Constructor

- **ITextProperty**(string name)

IMenuProperty Abstract class

Base class

- abstract class **IProperty**

Attributes

- **MenuSelection** (int) – This represents the index into the **MenuChoices** list.
- **SelectedMenuChoice** (string) – This is the item in the **MenuChoices** list corresponding to the **MenuSelection** index.
- **MenuChoices** (**IList**<*string*>)

Constructor

- **IMenuProperty** (string name)

Example:

```
 # adding data to IPropertyList propList; used in Compute function
prop = propList.AddNumberProperty('Offset 1', '0')
 prop.Description = 'Trace 1 Offset'
prop = propList.AddNumberProperty("TRATE", "800 MHz")
prop.Description = "Frequency"
 prop = propList.AddTextProperty("Text", "The Text")
 prop.Description = "Text Property"
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
prop = propList.AddMenuProperty('Operation', ['Add', 'Subtract',
'Max' ,'Min','Mean'])

prop.Description = 'Operation menu'


 # reading data from IPropertyList propList; used in Validate
function

numOfNumberProperties = 0

   if propList != None and propList.AllProperties != None:

        for prop in propList.AllProperties:

  if prop.PropType == Constants.EPropType.PT_NUMBER:

numOfNumberProperties ++
```

Other Application Specific Classes Used in Python Scripts

This section describes other classes used in Python scripts.

Constants Class

The constants used in Python script are defined in the **Constants** class.

Attributes

- **kDoubleParamStr** – String constant used to specify *double* as the type of a quantity.
- **kComplexParamStr** – String constant used to specify *complex* as the type of a quantity.

Enum EPropType (used to set property type)

EPropType.PT_NUMBER

EPropType.PT_TEXT

EPropType.PT_MENU

Example:

```
paramType = Constants.kDoubleParamStr

propType = Constants.EPropType.PT_NUMBER
```

UDSProbeParams Class

The objects of this class must be created in Python script in the **GetInputUDSParams** function. They are supplied to the **Validate** function, if implemented.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Attributes

- **ProbeName** (read-only string)
- **ProbeDescription** (read-only string)
- **ParamType** (read-only string)
- **ReportTypeName** (read-only string)
- **ComponentExpression** (read-only string)

Constructor: **UDSProbeParams**(*string probeName*, *string probeDescription*, *string paramType*, *string reportTypeName*, *string componentExpression*);

- **probeName** – Required.
- **probeDescription** – Optional (can be empty string).
- **paramType** – Required (can be one of the Constants).
  - **kDoubleParamStr**
  - **kComplexParamStr**
- **reportTypeName** – Optional (can be empty string).
- **ComponentExpression** – Optional (can be empty string).

Example:

```
udsProbParam = UDSProbeParams("probe1","",
Constants.kDoubleParamStr, "", "",)
```

UDSDynamicProbes Class

Attributes

- **UDSParam** (read-only **UDSProbeParams**)
- **UserSelectedProbeName** (read-only string)

Constructor: **UDSDynamicProbes** (**UDSProbeParams udsParam**, *string userSelectedProbeName*.

- **udsParam** – Required.
- **userSelectedProbeName** – Required.

Example:

```
udsProbParam = UDSProbeParams("probe1","",
Constants.kDoubleParamStr, "", "",)

selectedName = “probe1”

udsDynamicProbParam = UDSDynamicProbes(udsProbParam , selectedName
)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

QuantityInfo Class

Attributes

- **ParamType** (read-only string)
- **FullUnitType** (read-only string)

Constructors

- **QuantityInfo**(*string paramType*)
- **QuantityInfo**(*string paramType*, *string fullUnitType*)
- **Parameters**:
  - **paramType** can be one of the **Constants**.

    kDoubleParamStr

    kComplexParamStr
- **fullUnitType** is a case-insensitive string representing full unit type. It is not defined in **Constants**. Instead, you can use any of the units in string representation (for example, **mm** or **ghz**).

Example:

```
quantityInfo1 = QuantityInfo(Constants.kDoubleParamStr)

quantityInfo2 = QuantityInfo(Constants.kDoubleParamStr,”ghz”)
```

IProgressMonitor Abstract Class

The object of this class is a progress monitor. It displays calculations progress and checks if you have requested an abort of the computation.

When displayed in the application, each progress message has four items:

- A task name.
- A sub-task name.
- The progress amount.
- A button to abort the task in progress.

All of this functionality and abort interaction is achieved using the following functions.

- **SetTaskName** (*string taskName*):
- **SetSubTaskName** (*string subTaskName*)
- **BeginTask** (*string name*)
- **SetTaskProgressPercentage(int progressPercent)**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **CheckForAbort()** – If the generated quantities are computationally expensive, the UDO author can periodically call this method, then call **EndTask** with Fail and return False.
- **EndTask (bool passFail)**

Example:

```
progMon.BeginTask("Process DQS")

progMon.SetSubTaskName("Compute UI segments")

progMon.SetTaskProgressPercentage(33)

progMon.SetSubTaskName("Compute the rest")

progMon.SetTaskProgressPercentage(100)

progMon.EndTask(True)
```

Using .NET Collection Classes and Interfaces

Some of the API functions specified above use .NET collection classes and interfaces, such as **Array** class, **IList** interface, **IEnumerable** interface, and **IDictionary** interface. This topic describes how to work with the .NET collection objects in Python scripts.

.NET **Array**, **IEnumerable**, and **IList** objects can be indexed and iterated over as if they were Python lists. You can also check for membership using **in**. To get .NET **Array** and **IList** sizes, you can use Python's **len** or .NET's **Count**.

Example:

Getting size:

```
arraySize = doubleDataArray.Count

arraySize = len(doubleDataArray)

 listSize = sweepsNamesList.Count

 listSize = len(sweepsNamesList)
```

Iterating:

```
 for sweep in sweepsNamesList:

     print sweep

for in in xrange(listSize)

     print sweepsNamesList[i]
```

Checking for membership:

```
if 'Time' in sweepsNamesList:
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
        doThis()
    else:
        doThat()
```

For .NET **IDictionary**, same as for **Array** and **IList**, you can get size with **len** or **Count** and check for membership of the keys using **in**. Getting values for the keys also works the same way as in Python **dict**.

Example

Getting size:

```
varValuesSize = varValues.Count
varValuesSize = len(varValues)
```

Checking for membership:

```
 if 'offset' in varValues:
     print varValues['offset']
```

Getting value:

```
if 'offset' in varValues:
     offsetValue = varValues['offset']
```

As for iteration .NET **Dictionary** is different from Python **dict**. While iterating, Python **dict** will return keys, .NET **Dictionary** will return .NET **KeyValuePair**.

Example

Iterating:

for .NET **IDictionary**:

```
for varPair in varValues: #varPair is of .Net KeyValuePair type
     varName = varPair.Key
     varValue = varPair.Value
```

for Python **dict**:

```
for varName in varValues:
     varValue = varValues[varName]
```

You can use Python types instead of .NET types if you prefer. Cast .NET **Array** and .NET **iList** to Python **list** type and .NET **Dictionary** to Python **dict** type.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Do not use casting for data arrays; it can be extremely costly for the memory usage, as well as time consuming.

Example

```
  aPythonList = list(dotNetArray)

  aPythonList = list(dotNetList)

  aPythonDict = dict(dotNetDictionary)
```

User Defined Outputs: Messaging Methods

Messaging methods are provided to convey additional information from any of the UDOs methods. The Compute function is one typical location where such use is anticipated. Any messages sent via these functions appear in the application's message window using the appropriate icon.

You can also use these functions for debugging purposes:

- **AddErrorMessage(string)** – Call this method to convey an error condition.
- **AddWarningMessage(string)** – Call this method to convey a warning message; it is typically used for conditions that are not ideal but can be tolerated by the script.
- **AddInfoMessage(string)** – Call this method to convey an informational message. Use this call to output messages for debugging purposes.

```
############################################################

# Imports

############################################################

from Ansys.Ansoft.ModulePluginDotNet.Common.API import *

from Ansys.Ansoft.ModulePluginDotNet.Common.API.Interfaces import *

from Ansys.Ansoft.ModulePluginDotNet.UDO.API.Interfaces import *

from Ansys.Ansoft.ModulePluginDotNet.UDO.API.Data import *

class UDOExtension(IUDOPluginExtension):

  def __init__(self):

    pass


   #--- IDA IUDOPluginExtension ------------------------

  def GetUDSName(self):
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    return "MinMaxAvg"


  #--- ISA IUDOPluginExtension ------------------------
  def GetUDSDescription(self):
    return "Sample UDO for dimension reducing quantities"



   #--- ISA IUDOPluginExtension ------------------------
  # Returns list of category names
  def GetCategoryNames(self):
    return ["UDOOutputs"]

   #--- ISA IUDOPluginExtension ------------------------
  # returns a list of quantity names for the supplied category name
  def GetQuantityNames(self, catName):
    if catName == "UDOOutputs":
          return ["min_val", "max_val", "avg_val"]
    else:
         return []


  #--- ISA IUDOPluginExtension ------------------------
  # Returns an instance of QuantityInfo for the qtyName supplied or
  None if such a
  # quantity could not be found
  def GetQuantityInfo(self, qtyName):
    # All the quantities we have are simple doubles
    # we can leave them unitless
    return QuantityInfo(Constants.kDoubleParamStr)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
#--- ISA IUDOPluginExtension ------------------------
# Returns list of UDSParams and list of dynamic properties
 # Adds setup time properties to the propList
def GetInputUDSParams(self, udsParams, propList,
userSelectedDynamicProbes):

    # Add the probes. We need only one double quantity
    param1 = UDSProbeParams("probe1",
         "double quantity probe",
         Constants.kDoubleParamStr,
         "", "")
   udsParams.Add(param1)

    # Add the properties we want the user to supply
   # In this case, we will ask for a start/end range for
   # X parameters. Since we cannot reasonably provide defaults
   # as we have no idea what the sweep limits will be, we will
   # also ask if the limits are to be activated.
   prop = propList.AddNumberProperty("X Min", "0")
   prop.Description = "Start X value to consider"

   prop = propList.AddNumberProperty("X Max", "1")
   prop.Description = "End X value to consider"

   # For menus, the first option is the default.
   prop = propList.AddMenuProperty("Activate X Limits", ["No",
   "Yes"])
   prop.Description = "Activate X range"
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
  return True

 #--- ISA IUDOPluginExtension ------------------------
# Returns list of UDSParams and list of dynamic properties
# output UDSDynamicProbeCollection probes
def GetDynamicProbes(self, probes):
   pass


 #--- ISA IUDOPluginExtension ------------------------
# Returns list of sweeps names
 # We have no sweep sas we reduce them.
def GetUDSSweepNames(self):
  return []



 #-----------------------------------------------------------------
----------------
# IUserDefinedSolutionHandle API implementation.
# Calculates output values and sets them using
IUDSInputData/IUDSOutputData API.
def Compute(self, inData, outData, propList, progMon):


   # Get the sweeps associated with the probe and validate
  # use the probe name that we had defined earlier
   sweeps = inData.GetSweepNamesForProbe("probe1")
   if( sweeps == None or sweeps.Count > 1):
        AddErrorMessage(self.GetName() + "Unexpected sweep count 0
      or > 1 in Compute")
        return False
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
     # Get the data associated with our probe
     probeData = inData.GetDoubleProbeData("probe1")
     sweepData = inData.GetSweepsDataForProbe("probe1", sweeps[0])

 # Get the user specified properties.
 # Note that ideally, these "X Min" etc names should be written as
 # constant membets and referred to in both the GetInputUDSParams
 # and in Compute to reduce the change of typos.
 useXRangeProp = propList.GetMenuProperty("Activate X
Limits").SelectedMenuChoice
 xRangeStart = propList.GetNumberProperty("X Min").ValueSI
 xRangeEnd = propList.GetNumberProperty("X Max").ValueSI

 # At this stage, one can look at the RequestedQuantities and create
 # a dictionary to later check against. However, I am simply computing
 # all the quantities.
      minVal = 0
     maxVal = 0
     avgVal = 0

  # Check if we need to perform range computation
 if useXRangeProp == "Yes":
  seenAny = False
 avgSum = 0
 count = 0

  # zip is used since we also need to pull in sweep data
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 # an index and the array notation could also have been used
for probeVal, sweepVal in zip(probeData, sweepData):
 if sweepVal < xRangeStart or sweepVal > xRangeEnd:
pass

# Note that in a better written script, this code would be
# refactored into it's own function to avoid code
 # duplication
if not seenAny:
minVal = probeVal
maxVal = probeVal
avgSum = probeVal
seenAny = True
count = 1
else:
if probeVal < minVal:
 minVal = probeVal

 if probeVal > maxVal:
maxVal = probeVal

avgSum += probeVal
 count += 1

 if seenAny:
avgVal = avgSum/count

else:
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 seenAny = False
 avgSum = 0
 for probeVal in probeData:
 if not seenAny:
minVal = probeVal
maxVal = probeVal
avgSum = probeVal
seenAny = True
else:
if probeVal < minVal:
minVal = probeVal

if probeVal > maxVal:
maxVal = probeVal

avgSum += probeVal

 if seenAny:
avgVal = avgSum/probeData.Count

# Finally set the output values. Note that these are always set as
# lists even if we have just one item.
outData.SetDoubleQuantityData("min_val", [minVal])
outData.SetDoubleQuantityData("max_val", [maxVal])
outData.SetDoubleQuantityData("avg_val", [avgVal])

 # And we are done.
 return True
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential
information of ANSYS, Inc. and its subsidiaries and affiliates.

User-Defined Outputs: Script Organization

UDO scripts should all reside under the **UserDefinedOutputs** folder under either of the three library locations (**system**, **user**, or **personal**).

Using Script Libraries

You can incorporate base classes, additional data files, and so on, to organize your UDOs better. This type of library organization allows code reuse between similar UDOs and can be very helpful. There is special support provided for this type of script-library organization:

- All script-library and other support files need to be in a **Lib** sub-directory under the **UserDefinedOutputs** directory. Any **.py** files found in such **Lib** directories are ignored and not displayed in the GUI as a valid UDO choice.
- For a UDO script at any given directory depth, the system adds all **Lib** directories in their parent directories to the system include path (and so, any support script files from any **Lib** directory until the top level **UserDefinedOutputs** directory are imported).

Using Additional .NET Assemblies

Because the UDO functionality uses IronPython, you have access to the full .NET ecosystem. If needed, any subset of the UDO functionality can be implemented in any .NET language and used by the UDO script. Follow this procedure:

1. Build your .NET assembly for .NET 2.0 runtime.
2. Drop the built assembly in any **Lib** directory upstream of the UDO script location: that is, if you have your UDO script in **C:\Users\x\PersonalLib\UserDefinedOutputs\a\b\c\myudo.py** and have a .NET assembly called **com.Acme.UDOLib** You can keep the .NET assembly under:
   - **UserDefinedOutputs\Lib**
   - **UserDefinedOutputs \a\Lib**
   - **UserDefinedOutputs \a\b\Lib**
   - **UserDefinedOutputs\a\b\c\Lib**
3. If you cannot place the .NET assemblies into a **Lib** directory under **UserDefinedOutputs**, add these lines to your Python script.

   ```
   Import sys

   sys.path.append("full path to your .NET assembly location")
   ```

4. Add these lines to your Python script:

   ```
   Import clr
   ```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
clr.AddReference("com.Acme.UDOLib")
```

`import com.Acme.UDOLib` or `from com.Acme.UDOLibimport *`

Related Topics

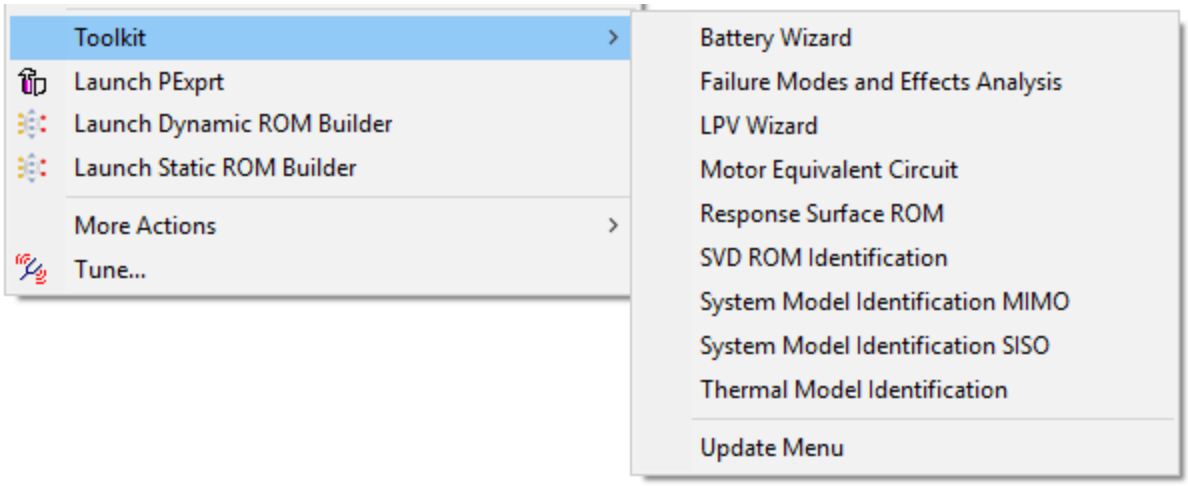
User Defined Outputs: An Introduction

User Defined Outputs: Python Script API

Toolkit

Select **Twin Builder** > **Toolkit** to access design type-specific IronPython scripts or module-specific tasks.

You can add the **Toolkits** directory and the **Twin Builder** subdirectory to your **syslib**, **userlib**, and/or **personallib** directories. IronPython toolkits (scripts) in these **Toolkits** > **Twin Builder** directories appear in the menu for use in Twin Builder.

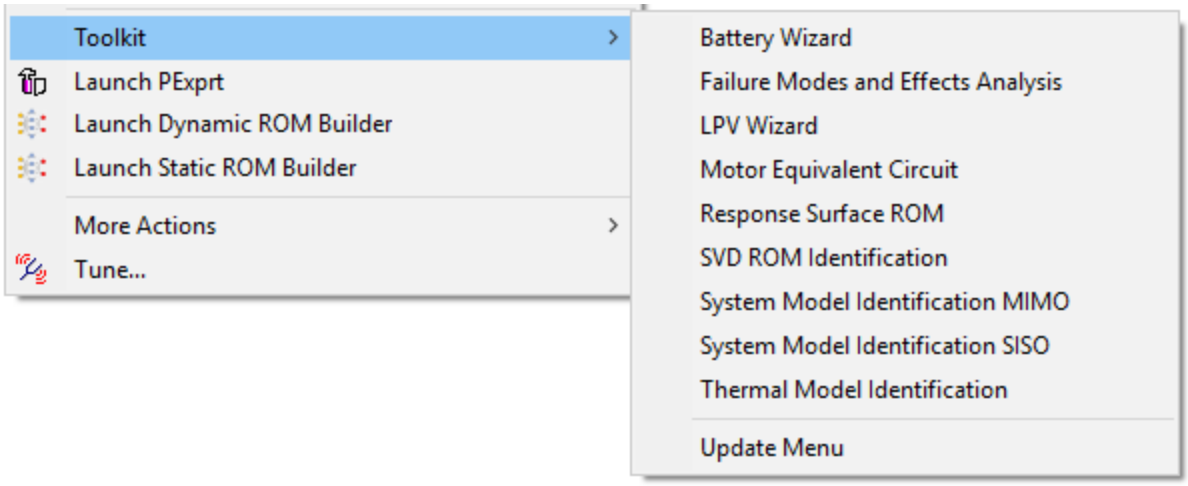


Select **Twin Builder** > **Toolkit** > **Update Menu** to add newly added toolkit scripts to the **Toolkit** menu. Menu items for files found in *<installation>*/**syslib** are inserted at the first level of the menu.

Battery Wizard

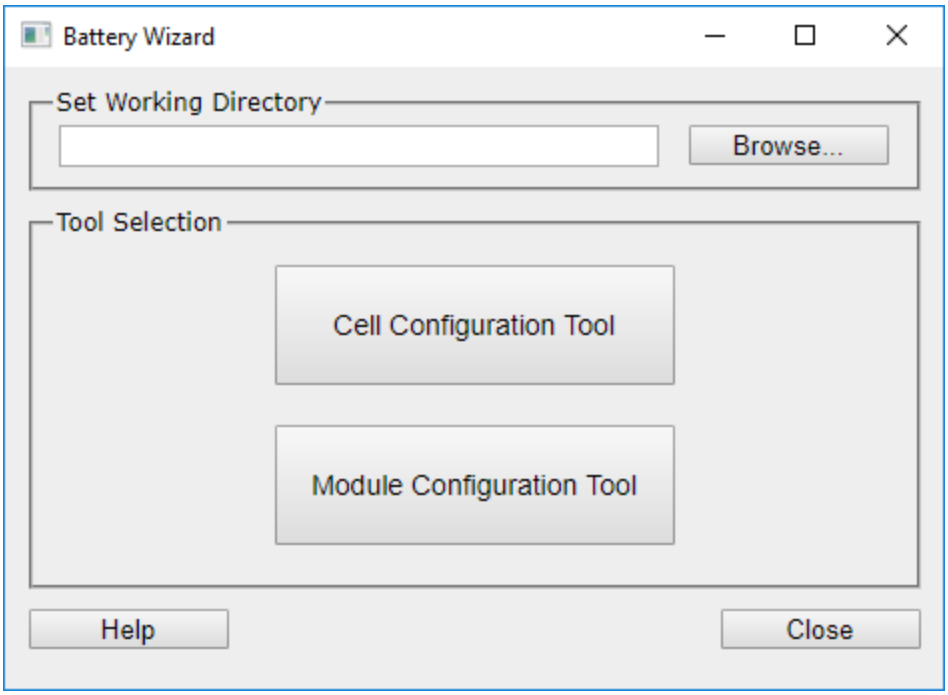
Use the **Battery Wizard** option under **Twin Builder** > **Toolkit** to build battery modules.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Procedure

1. Select **Twin Builder** > **Toolkit** > **Battery Wizard**. The **Battery Wizard** dialog box appears.



2. Click **Browse** and designate your working directory in the **Set Working Directory** field.
3. Click **Cell Configuration Tool** to construct configuration files that describe a battery cell. Use these cell configuration files to build battery modules using the **Module Configuration Tool**. You can select different cell configurations, as well as import

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

parameter characteristics using different methods. The tool also generates Modelica and FMU models of a single cell in Twin Builder, in addition to cell configuration files.

4. Click **Module Configuration Tool** to build parallel- and serial-connected battery modules. You can also generate battery modules in the output format of your choice. This tool requires cell configuration files as input. You can generate these files with the **Cell Configuration Tool** or you can manually construct them.

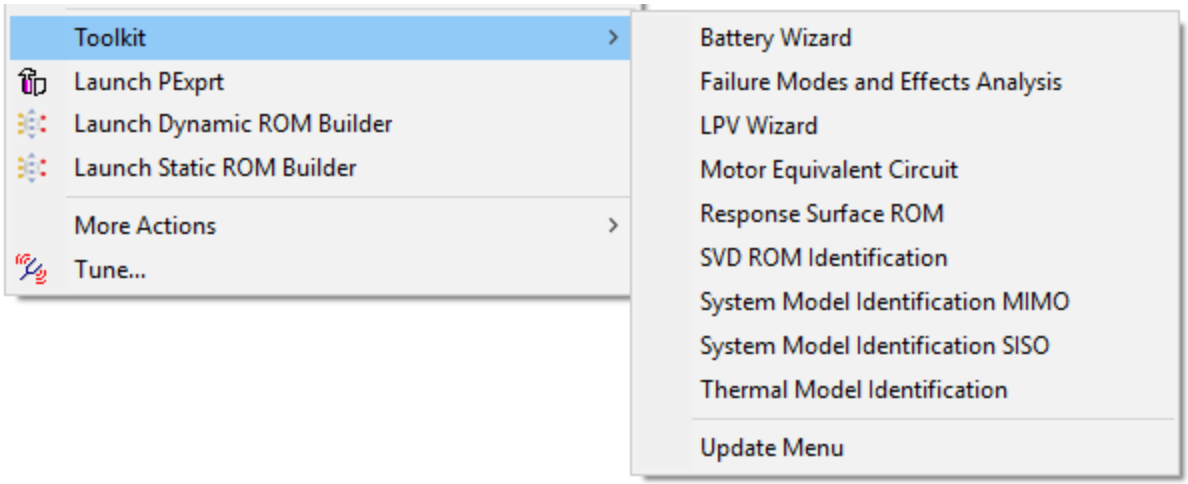
> **Note:**
>
> If you wish to build battery modules and already have the cell configuration file, you can start with the **Module Configuration Tool**. Otherwise, you may want to start with the **Cell Configuration Tool** and build a cell configuration file first.

LPV Wizard

Use the **LPV Wizard** option under **Twin Builder** > **Toolkit** to process the thermal step response training data of a local state-space (SS) Linear Parameter Varying (LPV) Reduced Order Model (ROM).

> **Note:** Click **Inspect** to confirm that your **dataInfo.txt** file is populated. You can also edit the file directly.

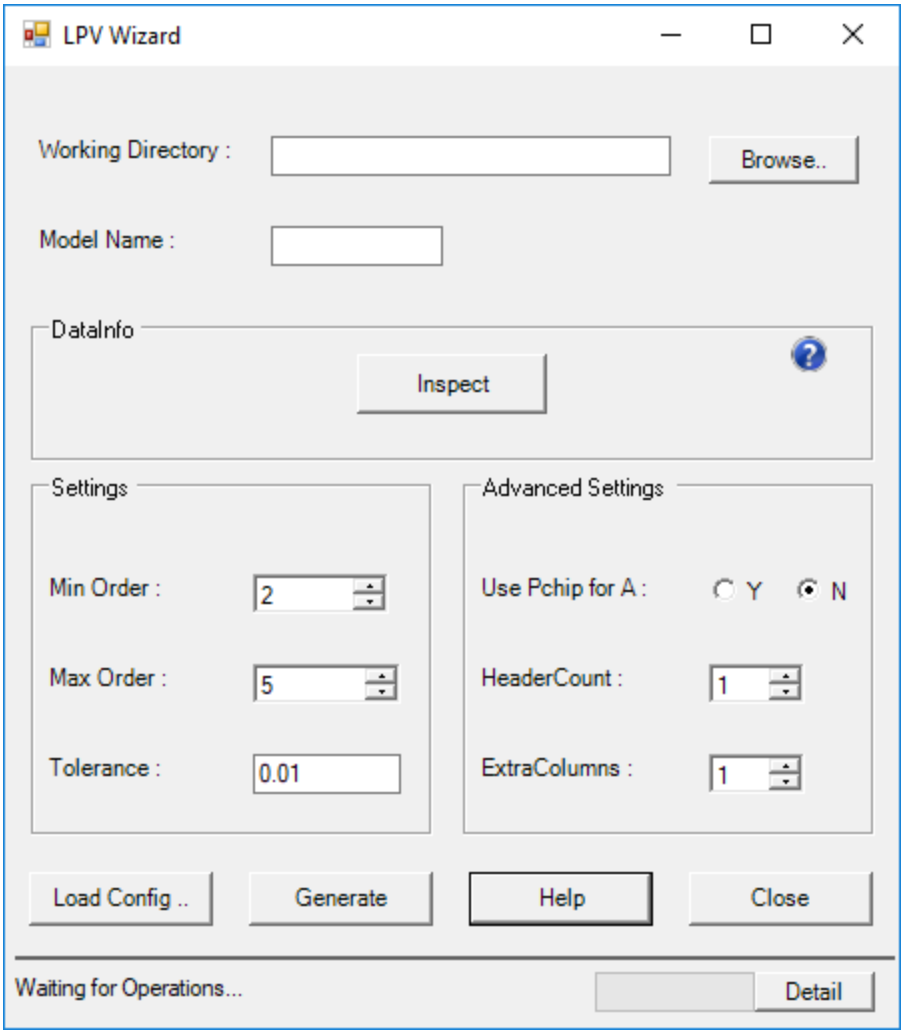
> **Note:** Make sure your input training data resides in a working directory.



Procedure

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Twin Builder** > **Toolkit** > **LPV Wizard**. The **LPV Wizard** dialog box appears.



2. Click **Browse** and designate your working directory in the **Working Directory** field.
3. Add a **Model Name**. This is the generated Modelica name. You don't have to add the **.mo** suffix; when your model generates, it will be added. Use alphanumeric characters only.
4. In the **DataInfo** section, click **Inspect** to review your **dataInfo.txt** file, including these elements:
   a. **InputNames** – The names of the inputs.
   b. **stepValues** – The ROM builder assumes that the step response training data represents a unit step response. If this is not the case, and if it will take a certain amount of (heat) step power or energy to achieve the temperature rise, you must enter that constant amount into the **stepValues** field. This value must equal the number of

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

inputs.

c. **parameterNames** – The names of the scheduling parameters.

d. **parameterValues** – The values of the scheduling parameters.

e. **outputNames** – The names of the outputs; the number of outputs should match the list of output names.

5. In the **Settings** section, populate these fields:

   a. **Min Order** – Enter the minimum order for the single scheduling parameter LPV. The value must be equal to or greater than 2.

   b. **Max Order** – Enter the maximum order for the single scheduling parameter LPV. The value must be equal to or less than 5, as a local state-space LPV ROM is best suited for low-order input-output representations.

   c. **Tolerance** – Enter the tolerance for step convergence.

6. In the **Advanced Settings** section, populate these fields:

   a. **Use Pchip for A** – Select **Y** to use Pchip interpolation for the initial interpolation step; select **N** to use positive interpolation. Using positive interpolation in the initial step can help assure stability throughout every interpolation step. All subsequent interpolation steps utilize Pchip interpolation.

   b. **HeaderCount** – Enter the number of lines to designate as header rows at the top of each input file. These rows will be ignored. Make sure that this number is consistent in all training data files, or your final results could be inaccurate.

   c. **ExtraColumns** – If you have columns in your training data files that LPV Wizard should ignore, enter that number of columns in this field. For example, when generating training data from Fluent, the first column often represents the time index, which should be ignored as this information is not needed by the ROM builder. In this case, setting **ExtraColumns** to **1** instructs the parser to ignore the first column of training data.

7. Click **Load Config** to load the configuration file, which contains the settings above in a valid configuration format.

8. Click **Generate** to create or update the configuration (**.lpvcfg**) files in the working directory, and to generate the SS-LPV ROM macromodel in Modelica format.

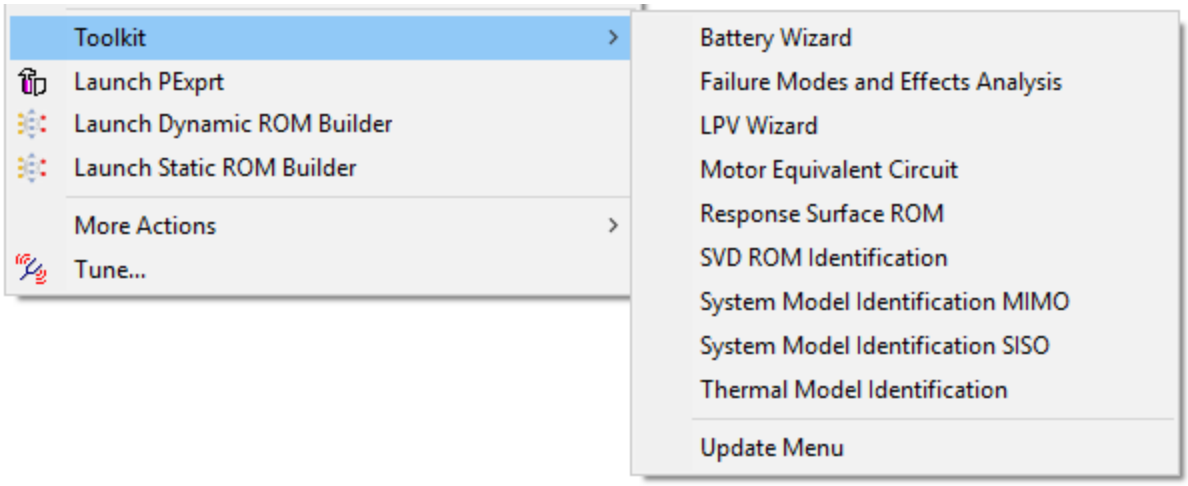
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note: As the LPV ROM is being processed, you may observe jump discontinuity in the response of the local state-space LPV ROM, specifically when the velocity profile experiences sharp transitions with fast instantaneous switching. This can occur because of the discontinuity of the parameter varying output matrix *C(p(t))* where *p(t)* is the velocity profile. When the parameter varying modes of the system are slow, varying, and smooth, then the discontinuity is a result of the interpolated parameter varying output matrix *C(p(t))*. If the system matrix modes vary slowly and smoothly, then the sudden and fast switching won't cause discontinuity.

Motor Equivalent Circuit

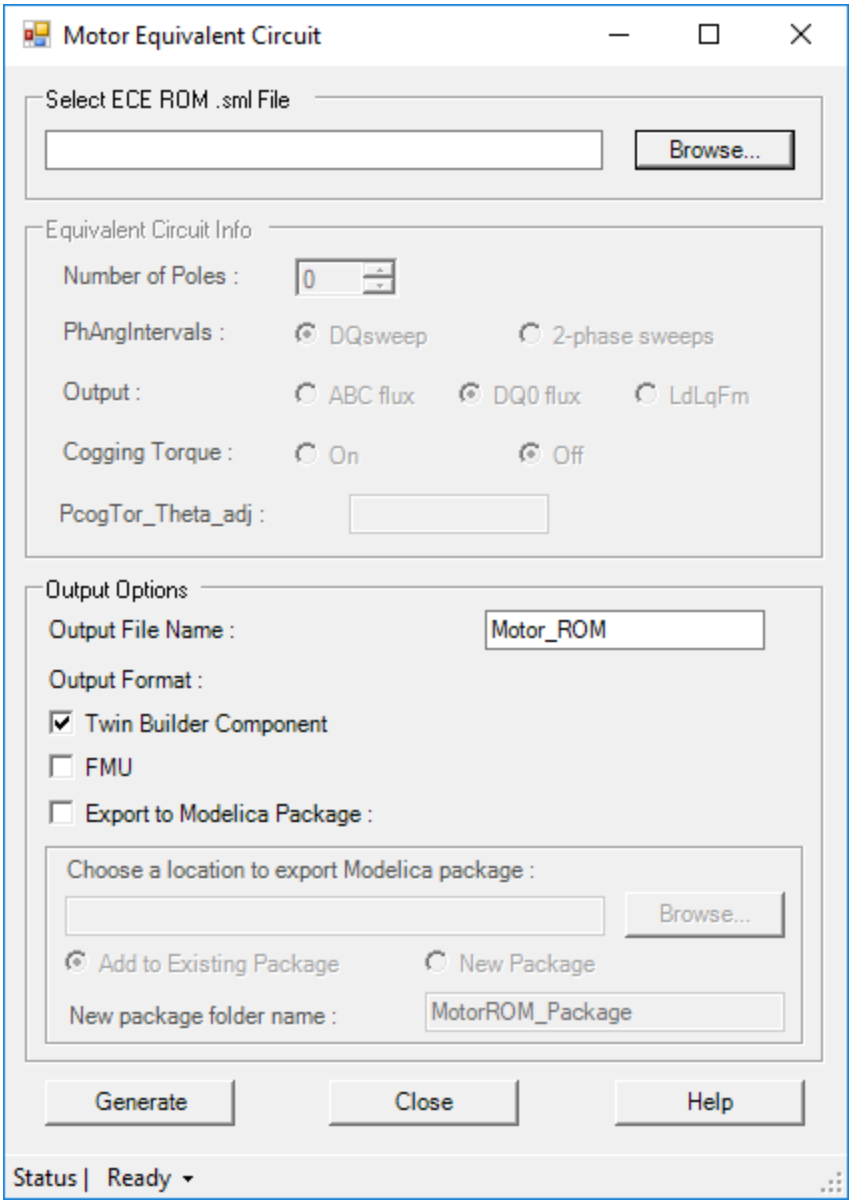
Use the **Motor Equivalent Circuit** option under **Twin Builder** > **Toolkit** to convert a Maxwell ECE ROM of an electrical motor to its equivalent Modelica model, which you can then export as an FMU in Twin Builder.

This toolkit only supports PMSM ECE models generated by Maxwell. Other Maxwell ECE models (for example, relays, transformers, and so on) or motor ROMs generated by other tools (such as an application from the Ansys store) are currently not supported.



1. Select **Twin Builder** > **Toolkit** > **Motor Equivalent Circuit**. The **Motor Equivalent Circuit** dialog box appears.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



2. Click **Browse** and select an ECE ROM .**sml** file. This is your input file.
3. In the **Output Options** section, enter an output file name and choose your output formats with the check boxes.
   - If exporting as an FMU, first complete the electrical and mechanical part of the system in a Modelica environment to avoid conversion between conservative pins to input/output pins.
   - If exporting as a Modelica package, the system does not compile the model.
4. Click **Generate** to parse the .**sml** file and generate equivalent models in user-specified forms. Temporary files and FMUs are saved in the **TBresult** folder in the input file

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

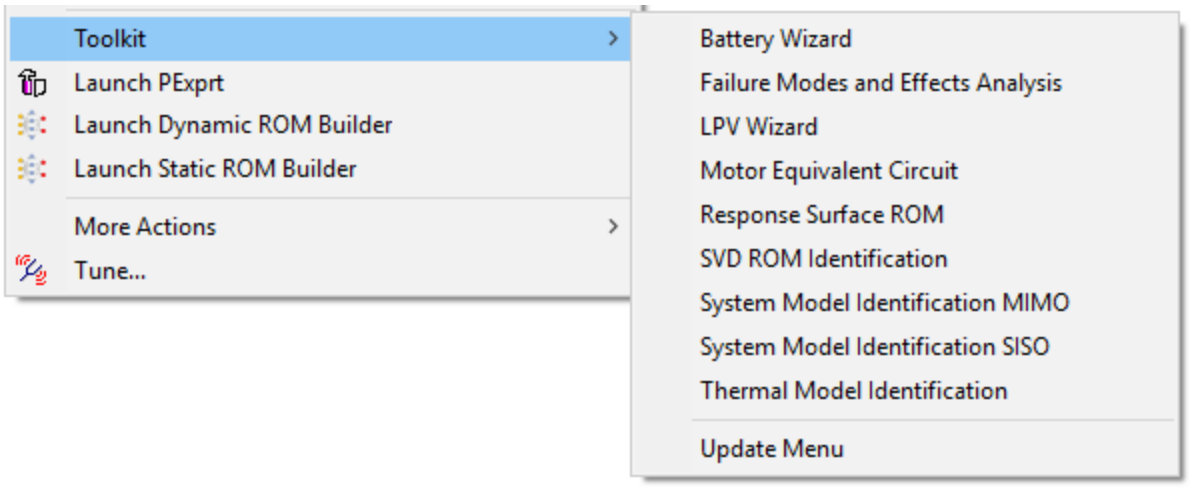
directory. Large datasets in the lookup tables can take a long time to compile.

> **Note:**
>
> - If you selected the **Twin Builder Component** or **FMU** check boxes, the models are saved in the **TBresult** folder.
> - If you selected the **Export to Modelica Package** check box, the system writes both models to the Modelica package, and you select a directory to save them in.
> - If you selected all three check boxes, the models are saved to the location you specify.

5. Due to different interface requirements in Twin Builder schematics and Modelica environment, two equivalent Modelica models are prepared for each input .**sml** file:
   - *output_file name*.**mo** – Use this model in a Modelica environment to connect with other Modelica components, such as mechanical loads in Modelica). This model cannot be compiled alone without connecting to a load.
   - *output_file name*_**RotV.mo** – Select the **Twin Builder Component** or **FMU** check box in the **Output Format** section to generate this model. The model automatically compiles and is available for use in Twin Builder schematics. If pin N0 is float in design, select **Edit Component** > **Terminals** and change **Unconnected/default behavior** for the N0 pin to **No net** to avoid an unconnected pin error.

Response Surface ROM

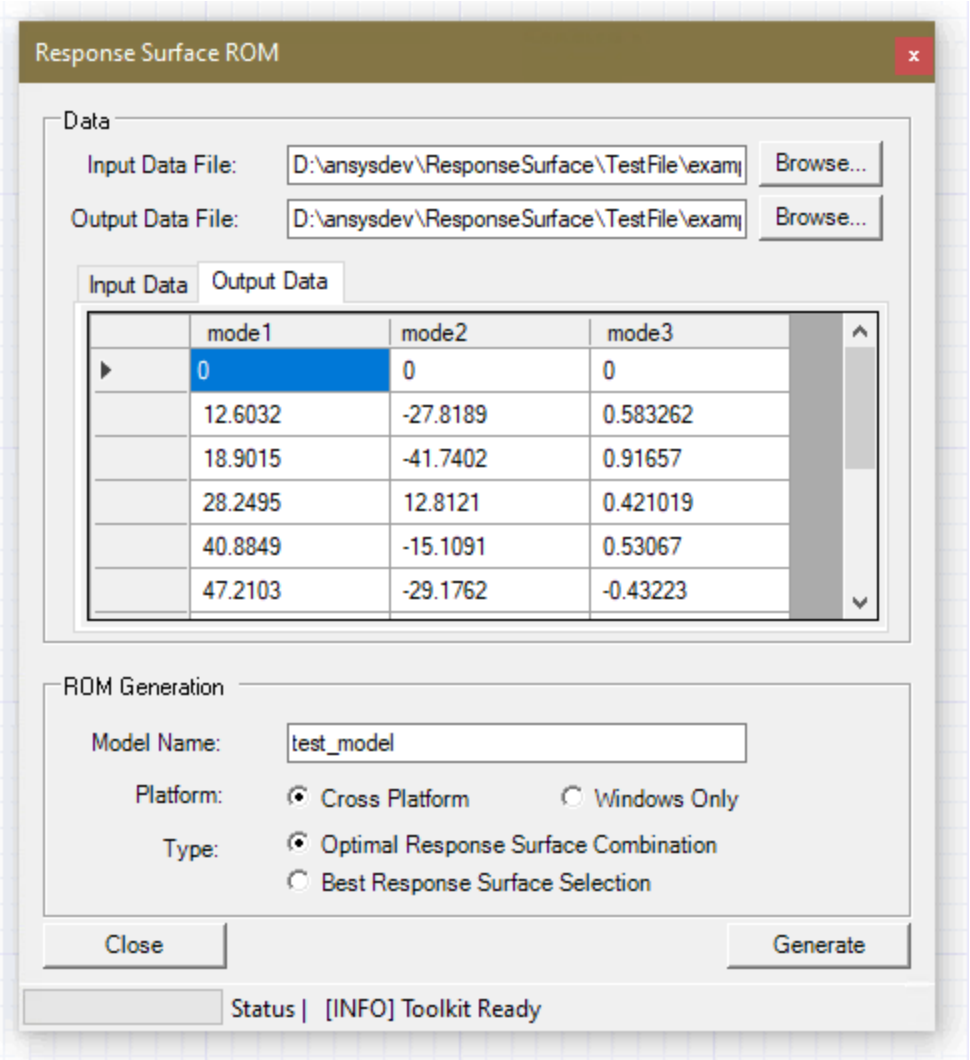
Use the **Response Surface ROM** option under **Twin Builder** > **Toolkit** to generate a reduced order model using the response surface data that you provide.



Procedure

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Select **Twin Builder** > **Toolkit** > **Response Surface ROM**. The **Response Surface ROM** dialog box appears.



2. Designate your input and output data files in the **Input Data File** and **Output Data File** fields. Click **Browse** to navigate to these files.
   - The **Input Data** and **Output Data** grids populate after choosing the files.
3. Enter a name for the output FMU file in the **Model Name** field.
4. Click **Cross Platform** or **Windows Only**, depending on the operating system you're running.
5. Select **Optimal Response Surface Combination** to generate a model with a response surface type available in previous versions of Twin Builder; select **Best Response Surface Selection** to generate a model that incorporates optiSLang functionality.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

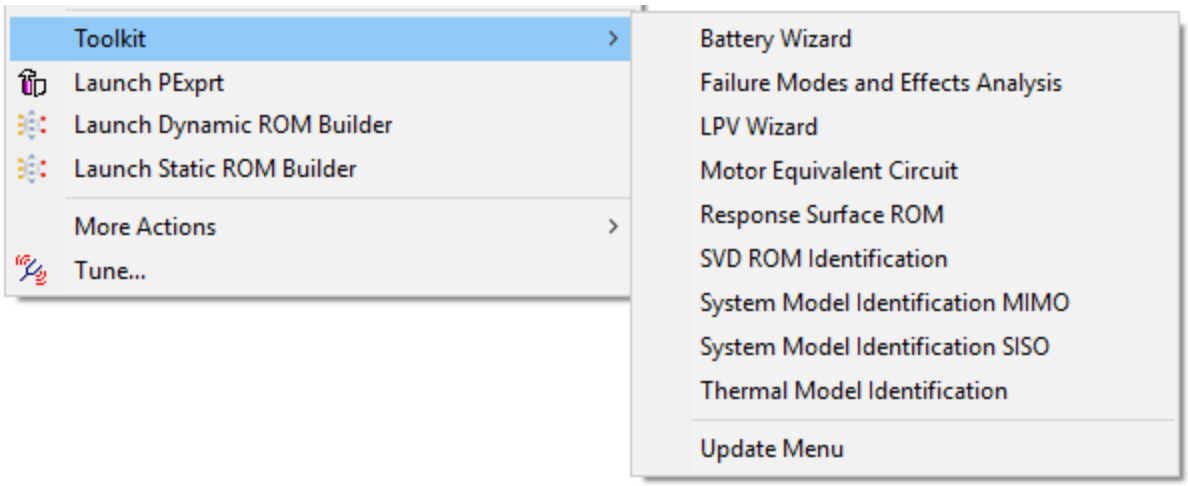
> **Note:**
>
> **Best Surface Response Selection** is supported if you chose the **Windows Only** option in the **Platform** area.

> **Note:**
>
> For more details on optimal surface ROM selection, see **Genetic Aggregation** in the DesignXplorer documentation. For more details on best response surface generation, see the optiSLang documentation on the Ansys help site.

6. Click **Generate** to create the model.

System Model Identification MIMO

Select the **System Model Identification MIMO** option under **Twin Builder** > **Toolkit** to generate a multiple-input and multiple-output (MIMO) linear system ROM, using the input and step response files that you create and provide.



Creating Files Needed for ROM Creation

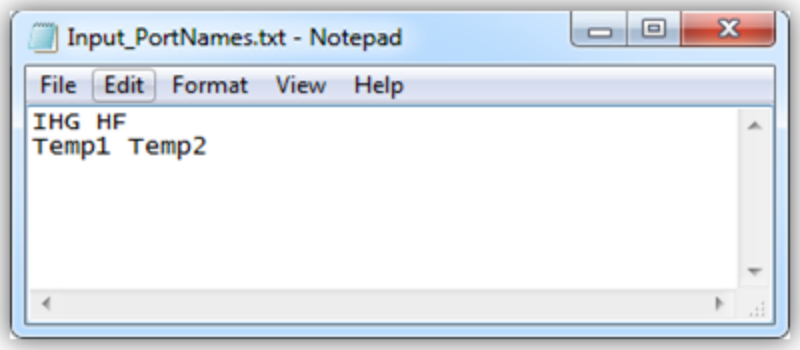
To use **System Model Identification MIMO** to create a MIMO state space model, create the following files:

- **Input_PortNames.txt**
- **Input_StepInputValues.txt**
- step response data files

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Input_PortNames.txt

The **Input_PortNames.txt** file contains port names you want to show as the Twin Builder component pin names; these names must match the step response file name. The file must contain two rows; the first row lists all the inputs, and the second row lists all the outputs as shown below.

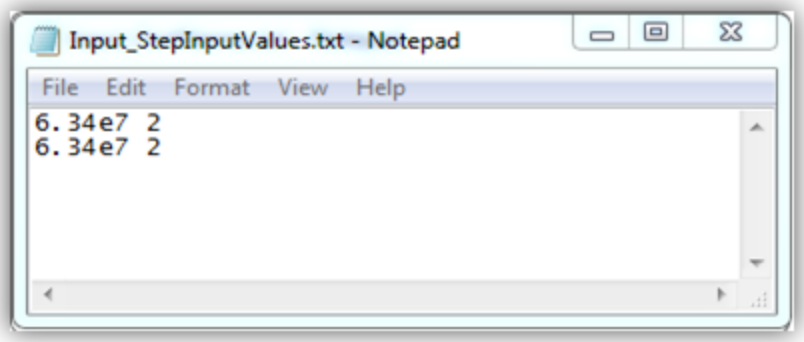


Input_StepInputValues.txt

The **Input_StepInputValues.txt** file identifies the step inputs used to generate the LTI ROM. It is formatted as a matrix, and it must be compatible with **Input_PortNames.txt**.

- The first element (1,1) is the step input used to generate **IHG.Temp1.out**.
- (2,1) is the step input used to generate **IHG.Temp2.out**.
- (1,2) is the step input used to generate **HF.Temp1.out**, and so on.

The same input is typically used for generating all the output step responses in one simulation, so each column should have the same value if that is the case. For example:



Step Response Data Files

You must generate step response data files independently for each input.

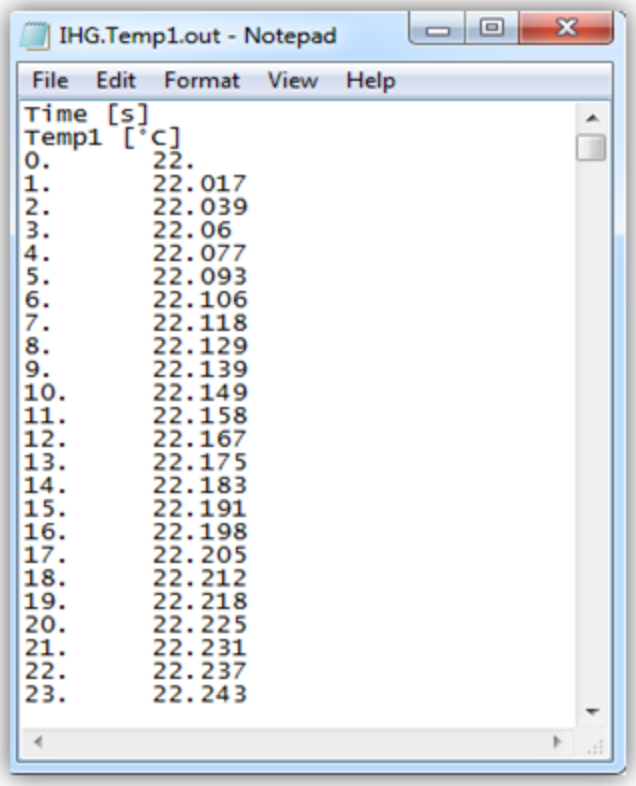
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For example, to generate step responses for a multi-input system, each time only one input should be activated with a step input. To simplify the process, you can export all the desired output step responses for each specific input. The step response tabular data must be formatted with *time unit as sec* and must be saved with the name formatted as *inputname*.*outputname*.**out**, defined in **Input_PortNames.txt**.

After saving all the step responses to the OUT files, you must manually reformat these files according to these requirements:

- The labels must be in the first line and second line of the file.
- The data section must have two columns separated by a tab or space.

Here is an example of a data file:



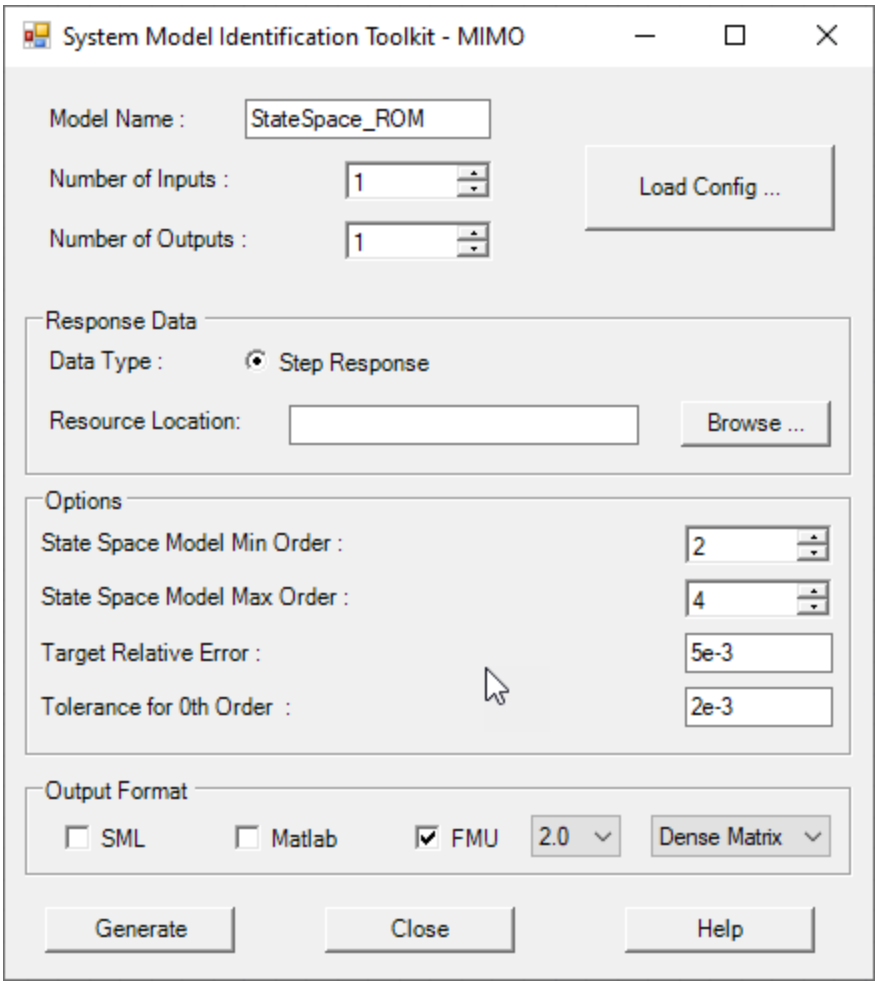
> **Note:**
>
> The step response file must have enough information to represent the system characteristics.

After creating the files, you must copy them into the same location.

Using System Model Identification MIMO

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

After you have created the files required for creating the MIMO linear state space ROM, use **System Model Identification Toolkit - MIMO** to generate an export file in an SML, MATLAB, or Modelica+FMU format. Click **Help** to access information about the options and inputs.



1. In the **Model Name** box, you can edit the default name of the generated component. You cannot use spaces, hyphens, or other special characters.
2. In the **Number of Inputs** box, you can edit the number of inputs to be used in Twin Builder. The number of inputs provided must match the data of the prepared files.
3. In the **Number of Outputs** box, you can edit the number of outputs to be used in Twin Builder. The number of outputs provided must match the data of the prepared files.
4. Click **Load Config** to select a configuration file and populate the dialog box.
5. In the **Response Data** section, browse to the location where you placed the **Input PortNames.txt** and **Input_StepInputValues.txt** data files. The **Step Response** option is selected by default.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. Edit parameters for the **Options**.

Parameter	Description
State Space Model Min Order	Minimum order used. A higher order gives more accurate results, but takes longer to simulate. If you enter **0**, then a contribution might be ignored based on the **Tolerance for 0th Order** specified.
State Space Model Max Order	Maximum order used. 40th order is set to be the maximum.
Target Relative Error	This is used to determine the order by the algorithm in the limit range given by [*minorder maxorder*].
Tolerance for 0th Order	This is used to ignore certain contributions if its value is less than the tolerance multiplied by the largest contribution.

7. For **Output Format**, select **SML**, **Matlab**, or **Modelica + FMU**. Raw state space format is always exported. When you select **Modelica + FMU**, a drop-down lets you select 1.0 or 2.0 for the FMU.
8. Click **Generate** for the toolkit to generate the chosen output format in the resource location:
   - **SML** – SML components load into Twin Builder.
   - **Matlab** – A MATLAB **m** file is exported and the raw matrices are generated and written separately.
   - **FMU** – Modelica components load into Twin Builder and the FMU version you indicated is also exported to the step response data location. When you select this check box, the next two options appear.
     - **FMU Version** – Select whether to generate and load FMU 1.0 or FMU 2.0.
     - **Sparse Matrix/Dense Matrix** – Choose if the Modelica components generate in regular dense matrix or sparse matrix. Sparse matrix is suggested for a state space model with many zero cells.

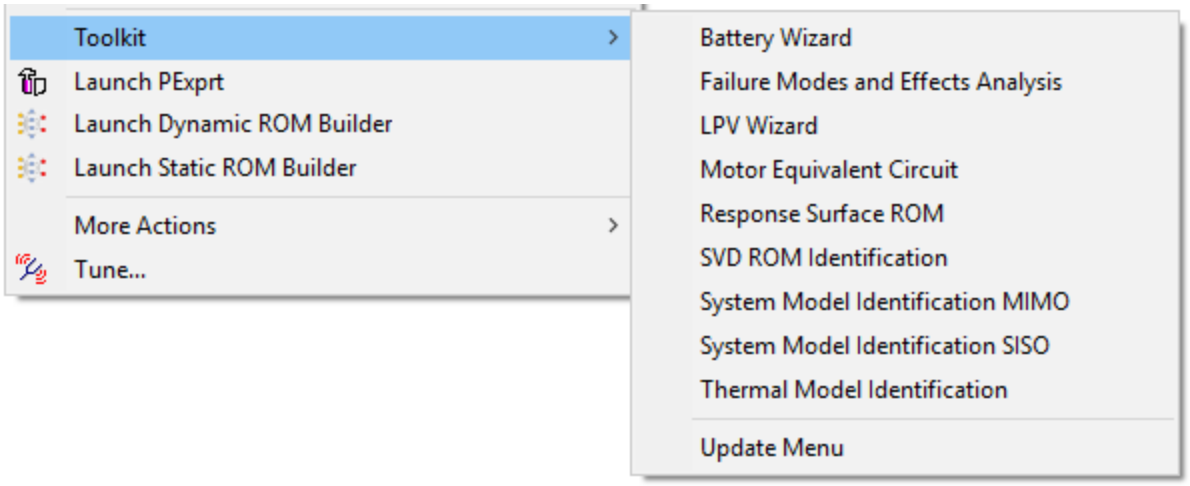
     > **Note:**
     >
     > The B matrix and D matrix expand to take reference values as extra constant inputs. The toolkit already embedded the reference values for SML model and Modelica model (as well as in the FMU version). For MATLAB file and raw matrices, you must manually assign correct reference values to output pins.

9. Click **Close**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

System Model Identification SISO

Use the **System Model Identification SISO** option under **Twin Builder** > **Toolkit** to generate an export file for use in a single-input and single-output (SISO) linear system, using the step response data file you provide. You must create the step response data file first.



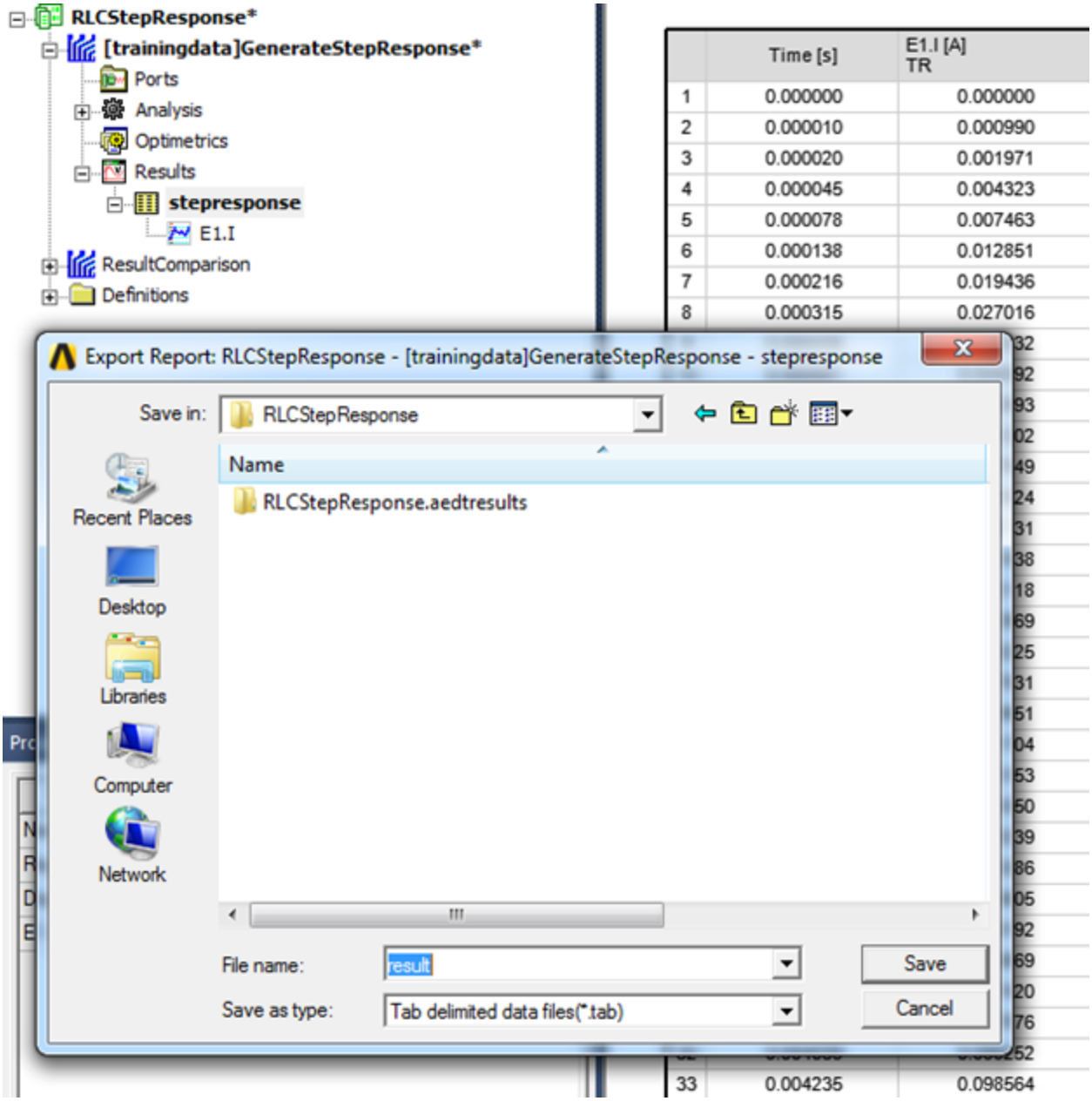
Creating a Step Response Data File

To use **System Model Identification**, first create a step response file and define the input and output characteristics.

You can access step response examples by selecting **File** > **Open Examples**, browsing to **Twin Builder\Applications\System Model Identification**, and selecting an example.

The first column must be a time step with the SI unit (in seconds). In the following example, the unit of the first column for E1.I has been changed to **s** to represent the time steps, and the **Display Type** in the report properties was defined as **Data Table** to ensure the unit is exported correctly. The data must be exported as a tab-delimited data file (***.tab**).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

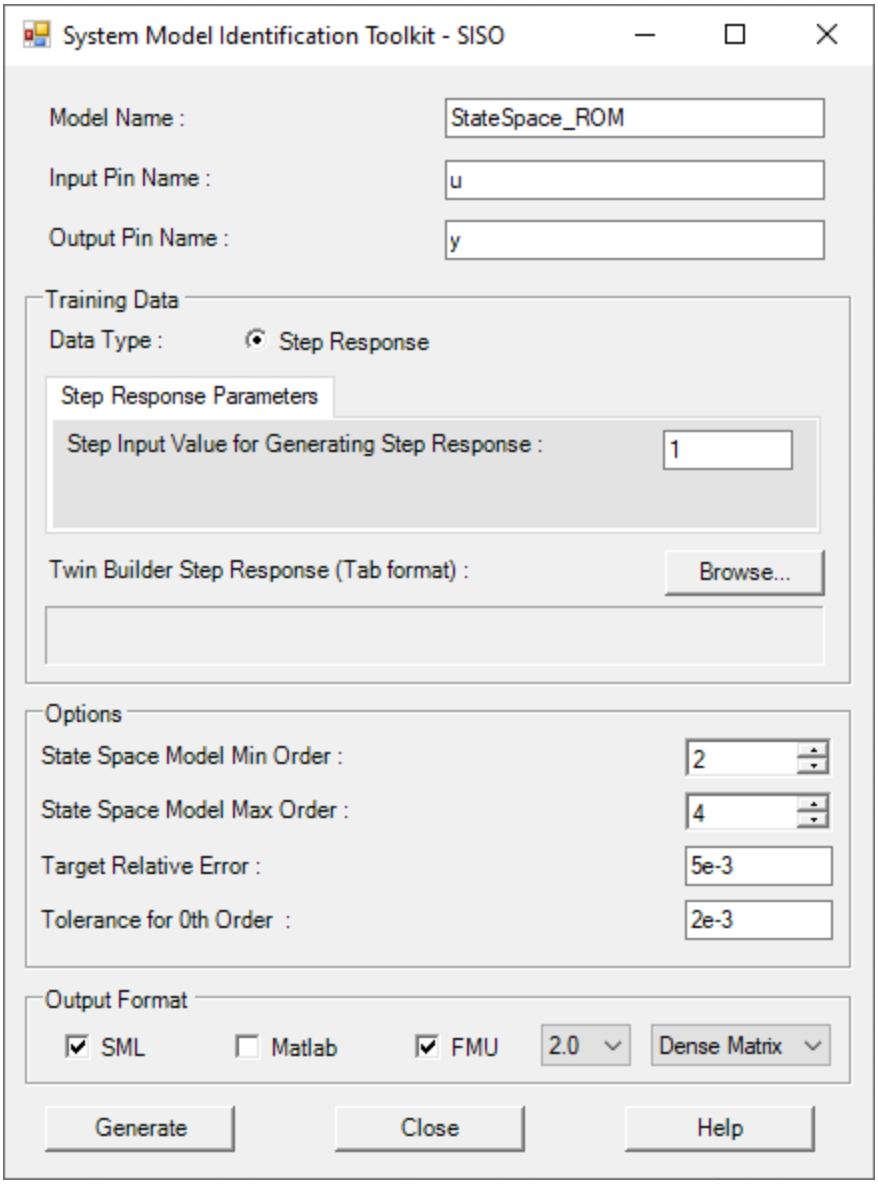


After you create the file, you can select it when using **System Model Identification**.

Using System Model Identification SISO

After you have created a step response data file, use **System Model Identification** to generate an export file in an SML, MATLAB, or Modelica + FMU format.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



1. In the **Model Name** box, edit the default name of the generated component. You cannot use spaces, hyphens, or other special characters.
2. In the **Input Pin Name** box, edit the default input pin name to be used in Twin Builder.
3. In the **Output Pin Name** box, edit the default output pin name to be used in Twin Builder.
4. For the **Step Input Value for Generating Step Response**, enter the step input value in Twin Builder for generating step response training data.
5. For **Twin Builder Step Response (Tab format)**, browse to select the data file (***.tab**)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

exported from the Twin Builder plot.

6. Edit parameters for the **Options**.

Parameter	Description
State Space Model Min Order	Minimum order used. A higher order gives more accurate results, but takes longer to simulate. If you entered 0, then a contribution might be ignored based on the **Tolerance for 0th Order** specified.
State Space Model Max Order	40th order is set to be the maximum. For thermal problems, an 8th order should work well for most applications.
Target Relative Error	This is used to determine the order.
Tolerance for 0th Order	This is used to ignore certain contributions if its value is less than the tolerance multiplied by the largest contribution.

7. For **Output Format**, select **SML**, **Matlab**, or **Modelica + FMU**. Raw state space format is always exported. When you select **Modelica + FMU**, a drop-down lets you to select 1.0 or 2.0 for FMU.
8. Click **Generate** for the toolkit to generate the chosen output format in the resource location:
   - **SML** – SML components load into Twin Builder.
   - **Matlab** – A MATLAB **m** file is exported and the raw matrices are generated and written separately.
   - **FMU** – Modelica components are loaded into Twin Builder and the FMU version you indicated is also exported to the step response data location. When you select this check box, the next two options appear.
     - **FMU Version** – Select whether to generate and load FMU 1.0 or FMU 2.0.
     - **Sparse Matrix/Dense Matrix** – Choose if the Modelica components generate in regular dense matrix or sparse matrix. Sparse matrix is suggested for a state space model with many zero cells.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

> **Note:**
>
> The B and D matrices expand to take reference values as extra constant inputs. The toolkit already embedded the reference values for the SML and Modelica models (as well as in the FMU version). For MATLAB **m** file and raw matrices, you must manually assign correct reference values to output pins.

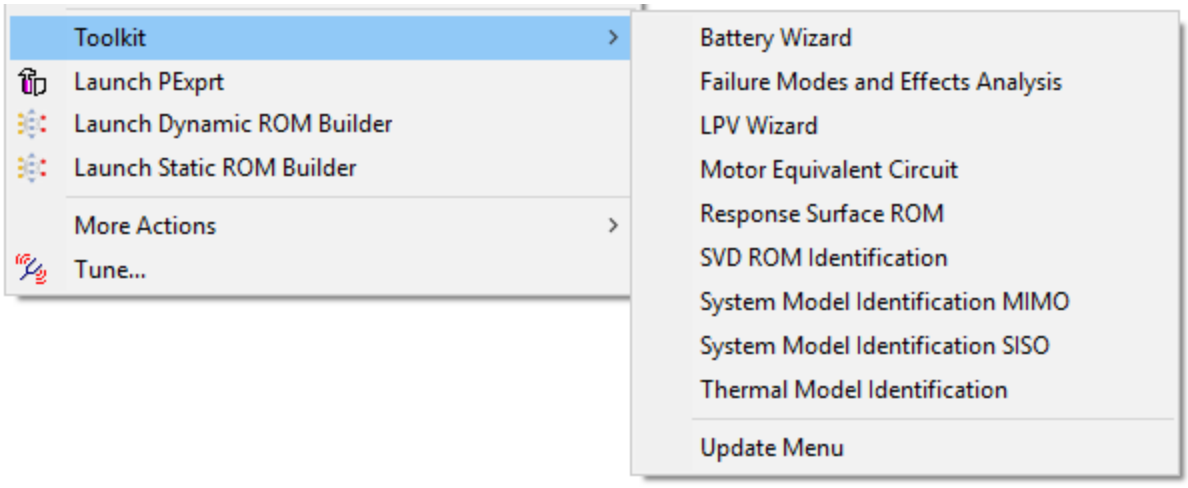
9. Click **Close**.

Related Topics

Creating a Step Response Data File

Thermal Model Identification

The **Thermal Model Identification** option under **Twin Builder** > **Toolkit** provides access to accurate reduced order models (ROMs) that you can bring into Twin Builder for system level analysis and simulation.



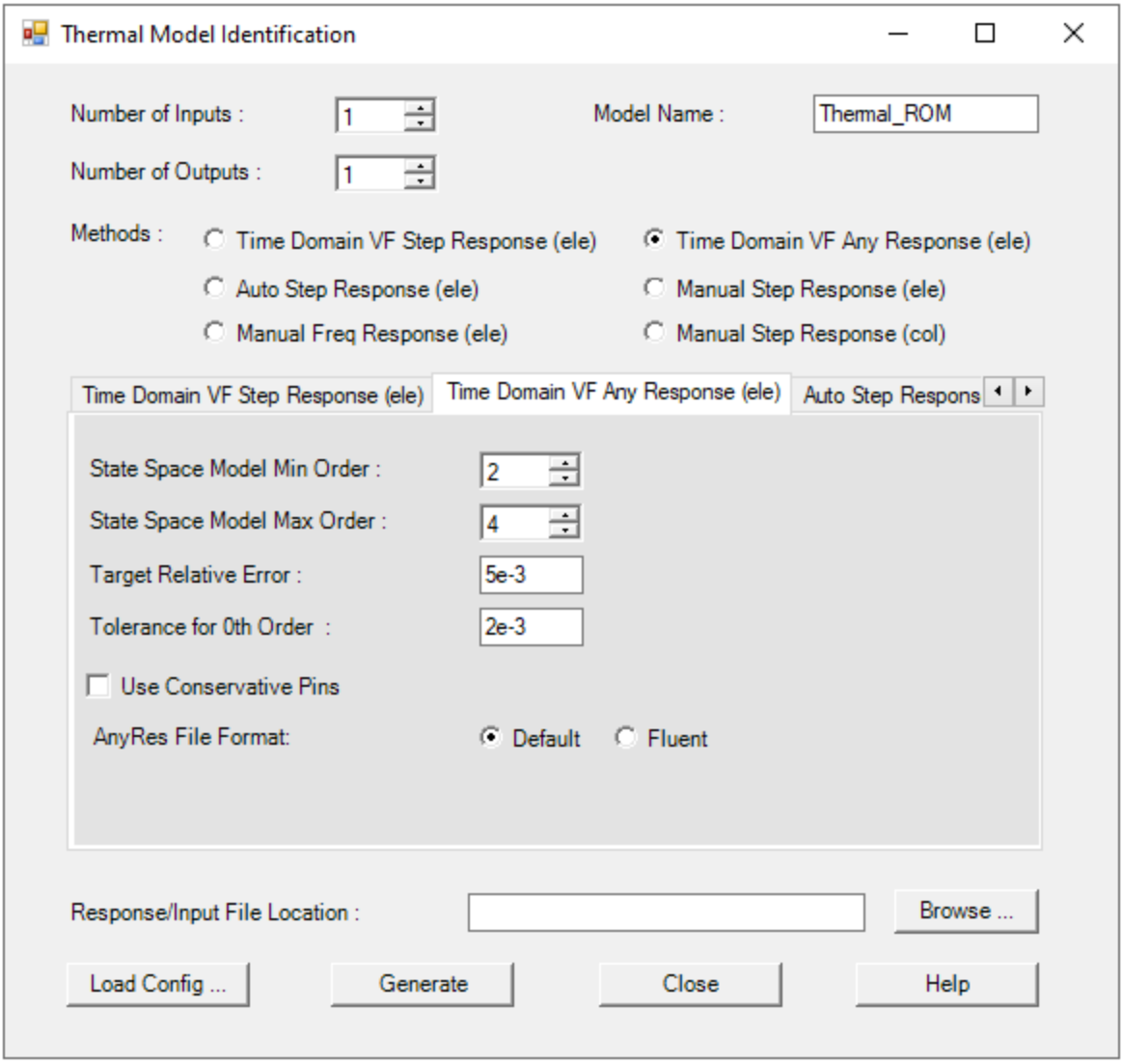
Each ROM is built from the accurate simulation results (step responses/frequency responses) from Finite Element Analysis (FEA) tools with Linear Time Invariant (LTI) vector fitting techniques.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

Access detailed help files and demo examples under *[installation directory]***\Win64\Examples\Twin Builder\Applications\Thermal Model Identification**.

Click **Help** on the **Thermal Model Identification** dialog box to view input information.



1. In the **Response/Input File Location** box, browse to the example you want to use and select the folder containing the resource data. Click **Load Config** to restore the existing

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

parameter information on the toolkit interface.

2. In the **Number of Inputs** box, enter the number of independent heat sources.
3. In the **Number of Outputs** box, enter the number of measurement points.
4. For the **Model Name**, enter the name for the model within Twin Builder. Do not use spaces, hyphens, or other special characters.
5. Select a method from the **Methods** options. You can edit parameters on the corresponding tab.

Method	Parameter	Description
Time Domain VF Step Response (ele)	**State Space Model Min Order**	Minimum order used. A higher order gives more accurate results, but takes longer to simulate. If **0** is used, then a contribution might be ignored based on the **Tolerance for 0th Order** specified.
	State Space Model Max Order	40th order is set to be the maximum. For thermal problems, an 8th order should work well for most applications.
	Target Relative Error	This is used to determine the order.
	Tolerance for 0th Order	This is used to ignore certain contributions if its value is less than the tolerance multiplied by the largest contribution.
	Use Conservative Pins	This option allows for conservative pins to be used for the pairs of inputs/outputs specified in the **Input_ConPins.txt** file.
	NonZero StepRes Slope	If your ROM is not expected to reach a steady state, select **Calculate** so the system will calculate the slope based on data in the step response files, or **Provide** if the slope is provided in a separate file. Select **No** if the step response is expected to reach a steady state.
	StepRes File Format	Choose the file format of your step response file. **Default** format includes a separate file for each input/output combination. **Fluent** format includes a separate file for each input; all outputs for a given input are included in a single file.
Time Domain VF Any Response (ele)	**State Space Model Min Order**	Minimum order used. A higher order gives more accurate results, but takes longer to simulate. If **0** is used, then a contribution might be ignored based on the **Tolerance for 0th Order** specified.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	State Space Model Max Order	40th order is set to be the maximum. For thermal problems, an 8th order should work well for most applications.
	Target Relative Error	This is used to determine the order.
	Tolerance for 0th Order	This is used to ignore certain contributions if its value is less than the tolerance multiplied by the largest contribution.
	Use Conservative Pins	This option allows for conservative pins to be used for the pairs of inputs/outputs specified in the **Input_ConPins.txt** file.
	Any File Format	Choose the file format of your response file of any type of response than step response. **Default** format includes a separate file for each input/output combination. **Fluent** format includes a separate file for each input; all outputs for a given input are included in a single file.
Auto Step Response (ele)	**State Space Model Min Order**	Minimum order used. A higher order gives more accurate results, but takes longer to simulate. If **0** is used, then a contribution might be ignored based on the **Tolerance for 0th Order** specified.
	State Space Model Max Order	40th order is set to be the maximum. For thermal problems, an 8th order should work well for most applications.
	Target Relative Error	This is used to determine the order.
	Tolerance for 0th Order	This is used to ignore certain contributions if its value is less than the tolerance multiplied by the largest contribution.
	Sampling dt Factor	This is used to increase or decrease the sampling dt, which is determined automatically.
	Keep Parameter Files	This allows the program to write out all parameters used into files. These files are useful for the manual step response (ele) method.
	Use Conservative Pins	This option allows for conservative pins to be used for the pairs of inputs/outputs specified in the **Input_ConPins.txt** file.
Manual Step Response (ele)		When this method is used, all parameters are from files, and they are not shown in

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

		the panel. There are a few additional parameters besides the ones used for other methods. When calculating the Fourier transform, FFT is the default method. For some cases, FFT becomes too time consuming. You can also use a numerical method to calculate the Fourier transform, called NFT. The flag for this is **F** for FFT and **N** for the NFT method. When using NFT, you can choose linear or quadratic interpolation. The flag for this is **l** for linear and **q** for quadratic. When using FFT, you must choose the maximum frequency for fitting along with the number of points used to represent the FT before vector fitting. Another parameter is the minimum number of iterations during vector fitting.
Manual Step Response (col)	**State Space Model Order**	The order is used for column fitting. 40th order is set to be the maximum. For thermal problems, a 15th order should be enough for most applications.
	Sampling Period	This the sampling period used to sample the step responses before FFT is performed.
	VF Number of Points	The number of frequency points for the lower frequency portion of the FFT for vector fitting. This portion becomes the sampled Fourier transform of the original system. Note that the entire FFT is not the sampled Fourier transform.
	Padding Factor	Add additional zero padding before FFT. (It is not critical to know the zero padding. Using **0** or **1** should be fine.)
	Interpolation Method	When calculating impulse response from step response, an interpolation is used first followed by derivative. **Cubic Spline** is more accurate, but the linear method is faster. For cases with many data points, it

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

		might be sufficient to use the **Linear** method without loss of accuracy. Interpolation is also needed when smoothing the given impulse responses.
	Coarsen Sampled Freq. Responses by a Factor of	The sampled frequency responses may have too many data points for vector fitting. You could reduce the number of data points before performing vector fitting. If **2** is used, only half of the data points are used.
	Use Conservative Pins	This option allows for conservative pins to be used for the pairs of inputs/outputs specified in the **Input_ConPins.txt** file.

6. Click **Generate** to create the model, which is added under **Project Components** in the **Components Libraries** dialog box. You are notified that an LTI ROM was generated and can be loaded into Twin Builder for calculation.
7. Click **Close**.

Response/Input Generation Files from Ansys 3D Products

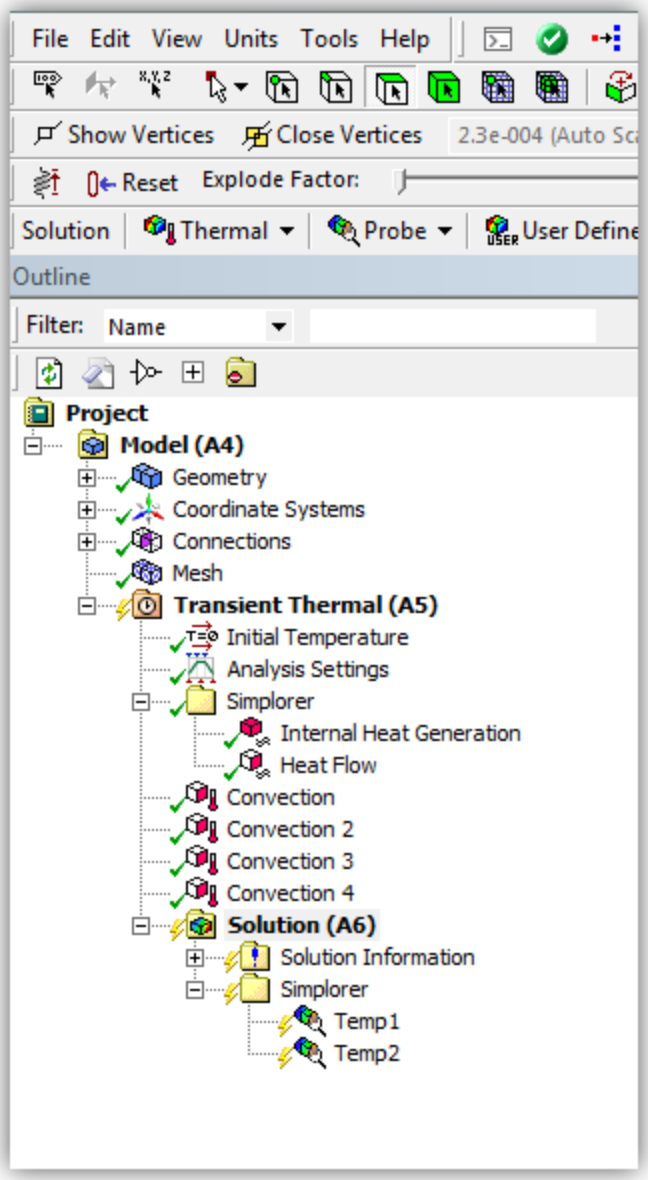
The following response/input generation files from Ansys 3D products are available for use in Thermal Model Identification.

- Icepak
  - For generating step response data files, see ***Adding an Ansys Icepak Component Subcircuit***.
  - For generating other configure files, refer to the help files in *[installation directory]***\Win64\Examples\Twin Builder\Applications\Thermal Model Identification\Help**
- Fluent 3D
  - The ROM Extract app (v1) extracts step responses from a Fluent 3D computation fluid dynamics simulation and the results can be used with Simplorer Thermal Model Extraction toolkit to create an ROM.

  The app is available in the Ansys App Store. Search for **ROM Extract**.
- Mechanical Thermal

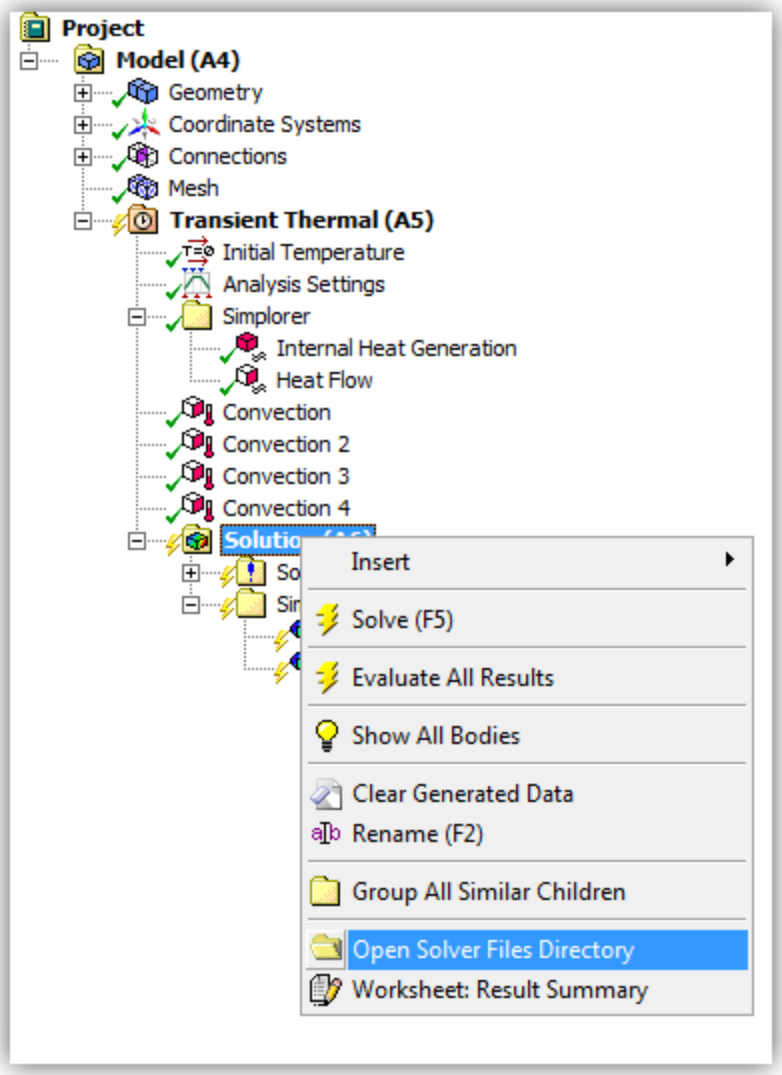
1. Open the Mechanical Thermal project.
2. Select the inputs and outputs for generating the Thermal LTI ROM. Note that the script requires that temperature probes be used for outputs.
3. Group them into a folder called **Simplorer** as shown:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



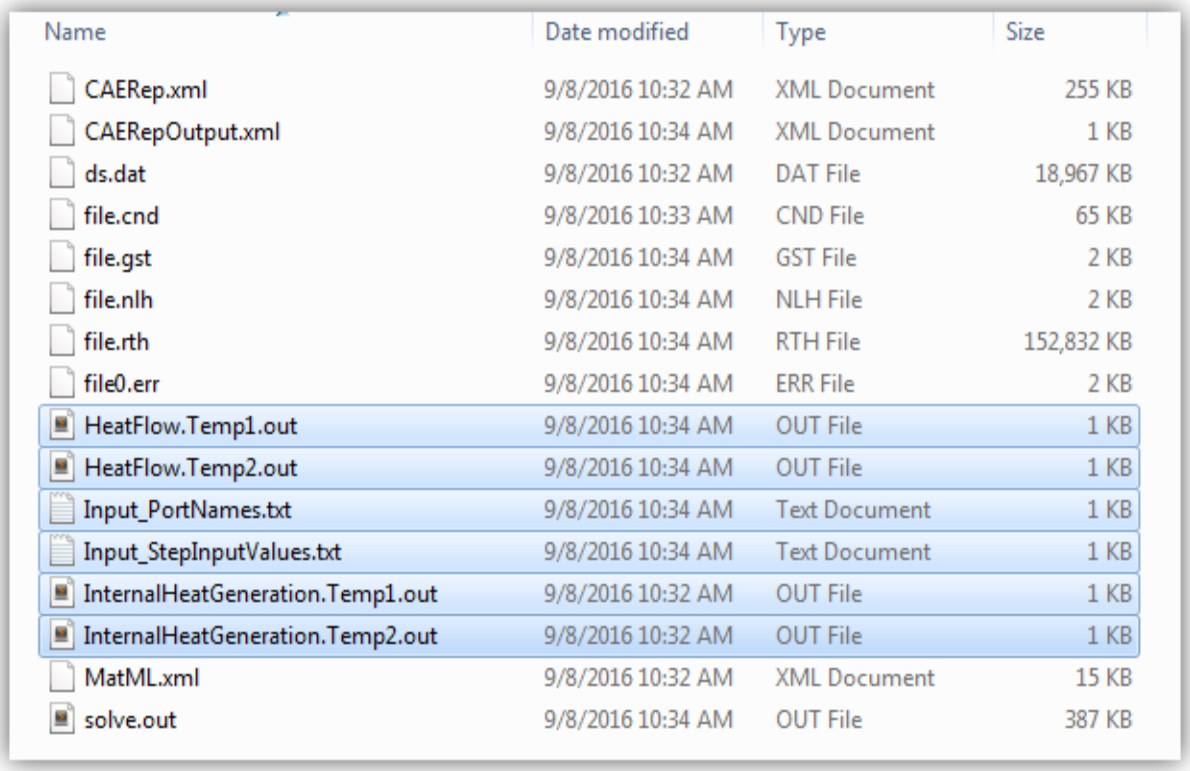
4. Select **Tools** > **Run Script**, browse to *[installation directory]***\Win64\syslib\Toolkits\Simplorer**, and select **MechanicalThermalStepResponseGeneration.js**.
5. After the script completes, click **Open Solver Files Directory**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



6. Copy the files shown in the **Solver Files Directory** to the **Resource** folder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



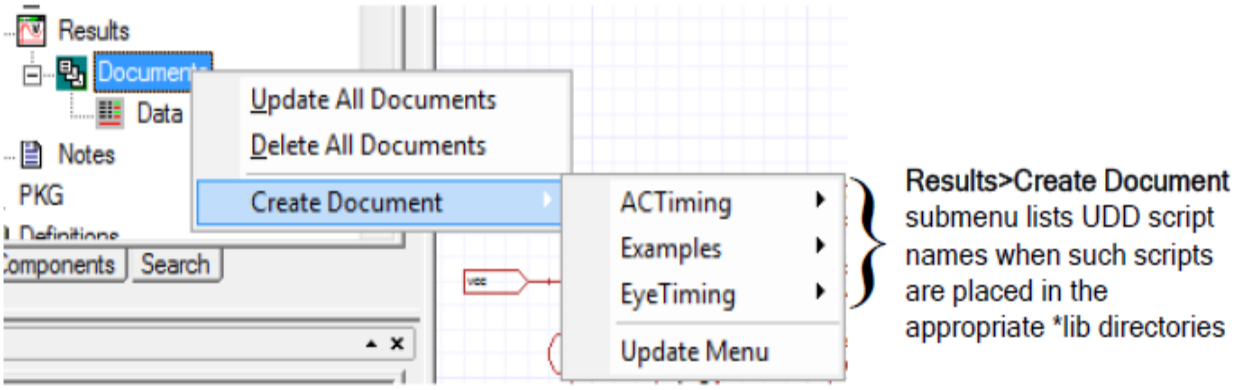
Name	Date modified	Type	Size
CAERep.xml	9/8/2016 10:32 AM	XML Document	255 KB
CAERepOutput.xml	9/8/2016 10:34 AM	XML Document	1 KB
ds.dat	9/8/2016 10:32 AM	DAT File	18,967 KB
file.cnd	9/8/2016 10:33 AM	CND File	65 KB
file.gst	9/8/2016 10:34 AM	GST File	2 KB
file.nlh	9/8/2016 10:34 AM	NLH File	2 KB
file.rth	9/8/2016 10:34 AM	RTH File	152,832 KB
file0.err	9/8/2016 10:34 AM	ERR File	2 KB
HeatFlow.Temp1.out	9/8/2016 10:34 AM	OUT File	1 KB
HeatFlow.Temp2.out	9/8/2016 10:34 AM	OUT File	1 KB
Input_PortNames.txt	9/8/2016 10:34 AM	Text Document	1 KB
Input_StepInputValues.txt	9/8/2016 10:34 AM	Text Document	1 KB
InternalHeatGeneration.Temp1.out	9/8/2016 10:32 AM	OUT File	1 KB
InternalHeatGeneration.Temp2.out	9/8/2016 10:32 AM	OUT File	1 KB
MatML.xml	9/8/2016 10:32 AM	XML Document	15 KB
solve.out	9/8/2016 10:34 AM	OUT File	387 KB

> **Note:**
>
> The first rows of the **.out** files should be the initial conditions of the outputs; by default, the script outputs only system default initial conditions. If you manually modify the initial condition, you also need to manually modify the first rows of the **.out** files.

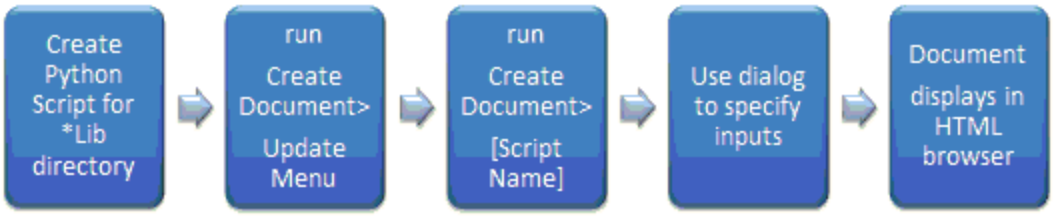
User Defined Documents (UDDs)

User defined documents (UDDs) are custom reports that you define through IronPython scripts. Once placed in a **Lib** directory, you can access the scripts with **Create Document**. The scripts describe a **Create User Defined Document** dialog box that lets you specify trace and solution inputs. After you confirm your input selections, XML, HTML, and PDF documents generate. A web browser displays the generated HTML file. The created document appears in the Project tree, under **Results** in the **Documents** folder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The general UDD process flow is as follows.



The UDD Python scripts must be placed in the **UserDefinedDocuments** directory under **syslib**, **userlib**, or **Personallib** with any subdirectory structure needed. The **Lib** directory can contain Python scripts that have common code that other scripts can use.

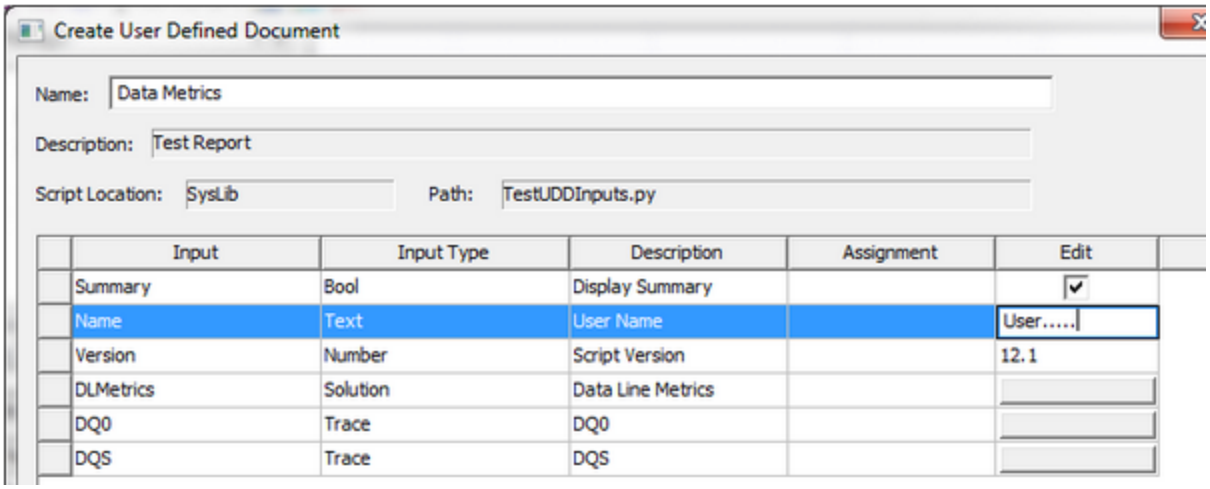
Use **Results** > **Create Document** > **Update Menu** to refresh the menu to include the new UDD scripts that have been copied to **syslib**, **userlib**, or **Personallib**, or to exclude them if they have been deleted, after the launch of desktop.

The UDD scripts that are in **syslib/UserDefinedDocuments**, **userlib/UserDefinedDocuments**, or **Personallib/UserDefinedDocuments** become available through the **Results** > **Create Document** menu.

Create User Defined Document Dialog Inputs

User defined documents allow data from traces, solutions and report types as inputs. A UDD can specify the named inputs for which you select or enter the values in the **Create User Defined Document** dialog box that displays when you run **Results** > **Create Document** > <*scriptName*>.

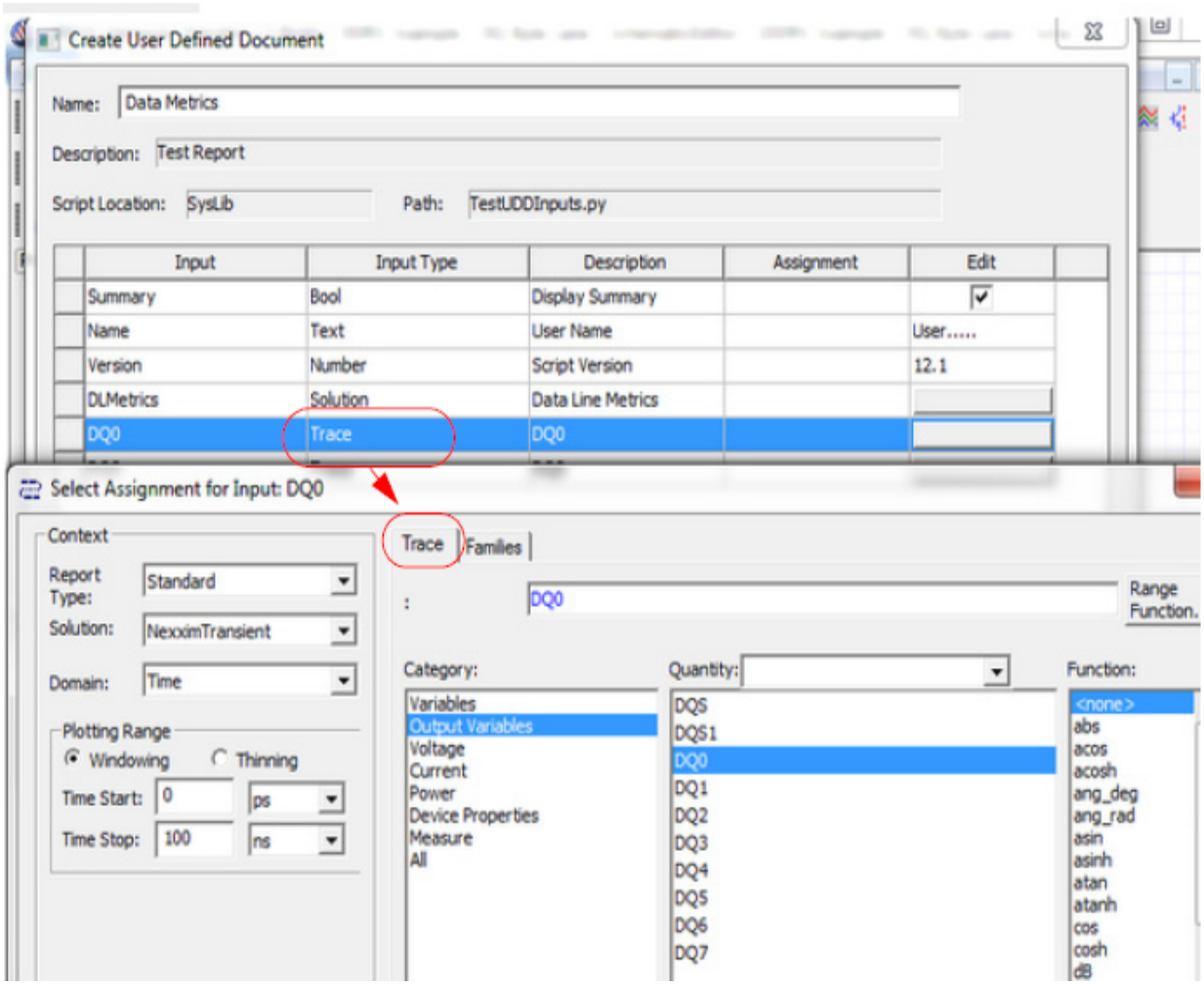
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Input types can be Boolean, number, text, trace, or solution. The Boolean, number, and text type can be given a default value that you can override when the document is created or modified. For example, you can select a trace when you create or modify a UDD document. The trace data is available to you and can be accessed from the Python script.

At the time of selection you can choose from the **Reporter** dialog box, the report type (Standard, Eye Diagram, User Defined), solution name, context and the quantity for which you want the trace data.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

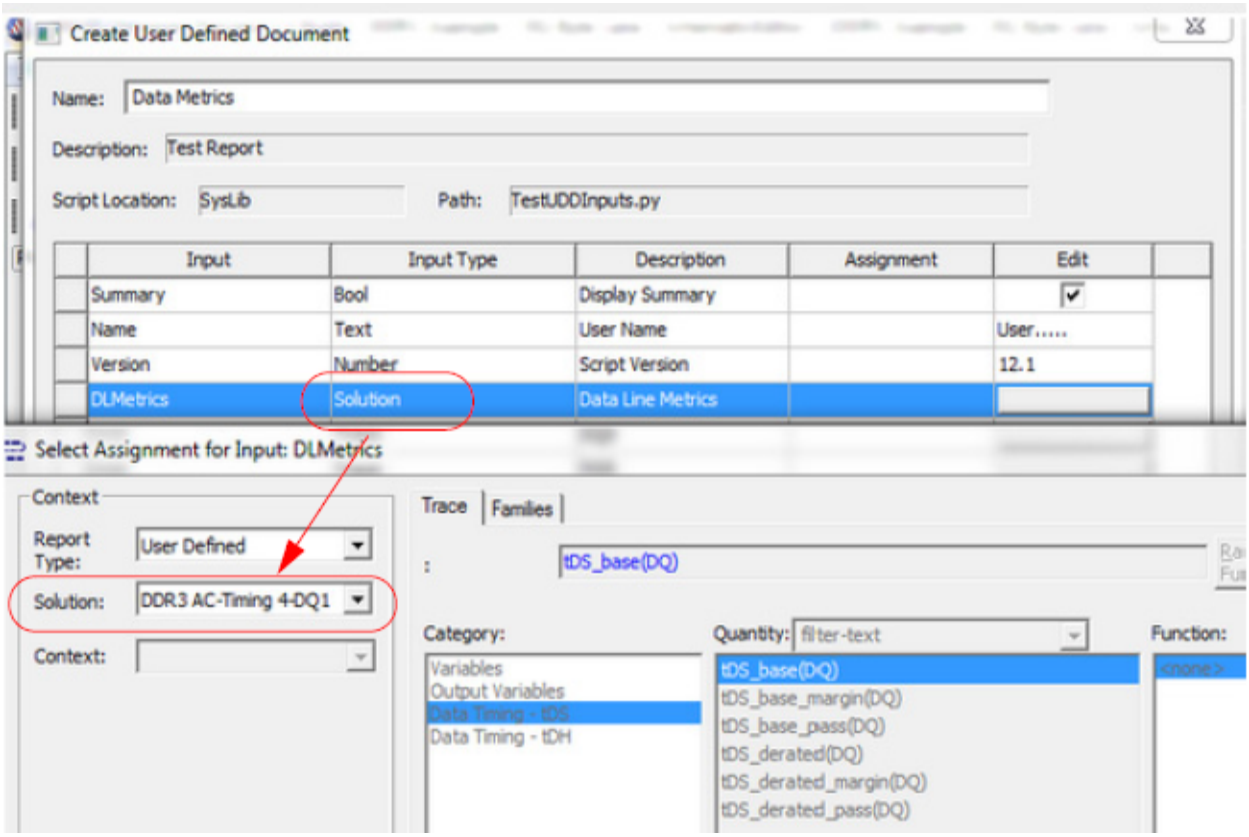


Input Type can also be Solution. You can select an entire solution when the document is created or modified. The solution data is now available and can be accessed from the Python script.

At the time of selection you can choose from the reporter dialog box, the report type (Standard, Eye Diagram, User Defined), solution name and context. A specific quantity cannot be selected since data for all quantities in the solution are available.

Note: The **Category**, **Quantity**, and **Function** fields are read-only.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



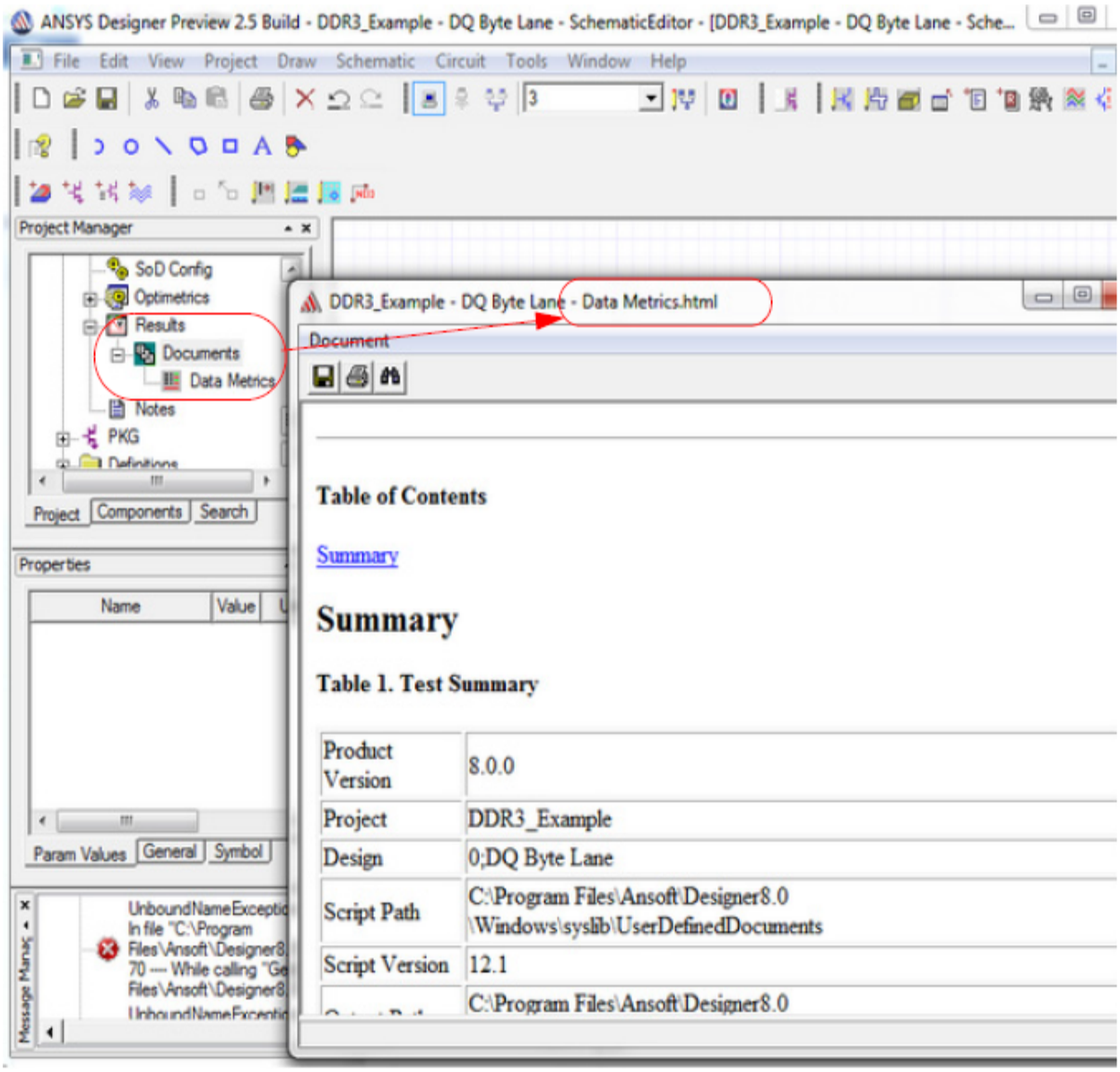
UDD Document Creation and Display

After all the input selections for a UDD are confirmed, based on the script, XML, HTML, and PDF documents generate based on the inputs you provided. The XML, HTML, and PDF generation is based on specific calls in the Python script, which are explained in a following section. A web browser displays the generated HTML file.

The created document appears in a new folder named **Documents** in the **Results** folder. All documents that you create for the design are placed in this folder.

Note that generated PDF documents do not display certain font styles, such as Cyrillic.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Related Topics

Managing Documents Listed in the Project Window Under Results

Viewing UDDs with an Html Web Browser

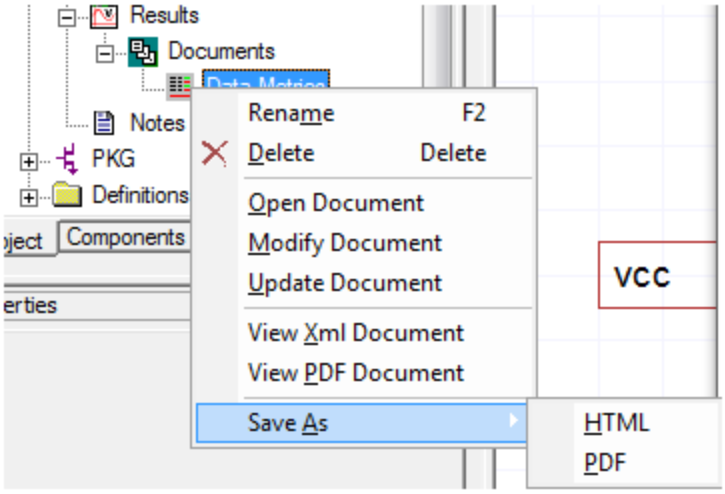
UDD Script Libraries

User Defined Definitions: Python Script API

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

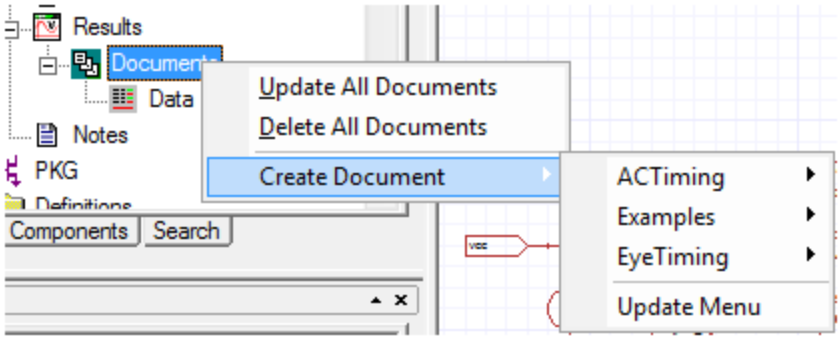
Managing Documents Listed in the Project Window under Results

Right-click a user-defined document displayed in the **Project Manager** tree to bring up a menu where you can rename or delete the document. **Open document** opens the web browser with the HTML document. **Modify document** opens a setup dialog box where you can change the selections for the input. To view the XML and PDF documents, choose the appropriate menu items. There is also a menu item to save the document in a different location.



Documents folder shortcut menu

Right-click the **Documents** folder to **Update All Documents** or **Delete All Documents**. It also provides the option of creating a document from here.

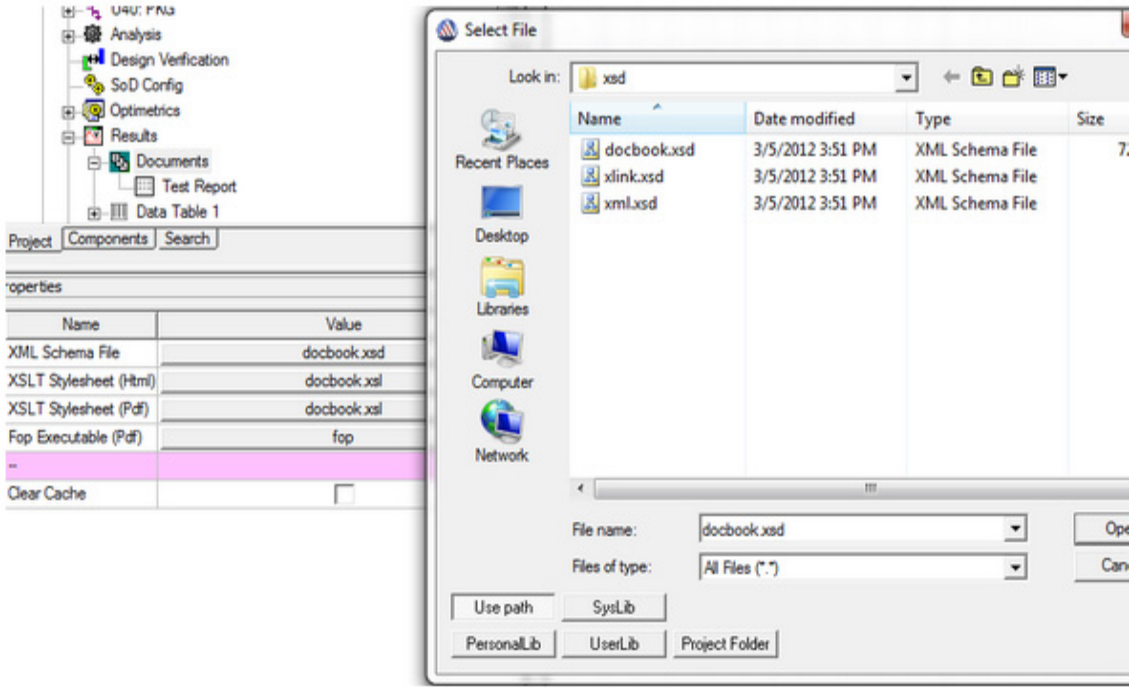


Document folder Property window

When the documents folder is selected, the Property window shows these properties:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **XML Schema File** – File path to the XML schema file.
- **XSLT StyleSheet (Html)** – File path to the XSLT stylesheet file used for HTML generation.
- **XSLT StyleSheet (Fo)** – File path to the XSLT stylesheet file used for PDF generation.
- **Fop Executable (Pdf)** – File path the FOP executable used for PDF generation.
- **Clear Cache** – Clears the cached XSL transform object and forces creation of a new one. The caching is done to save time during document generation, so subsequent generation or update of the document can use the cached transform object. But sometimes you may want to force a recompile of the document if you change the style sheet.

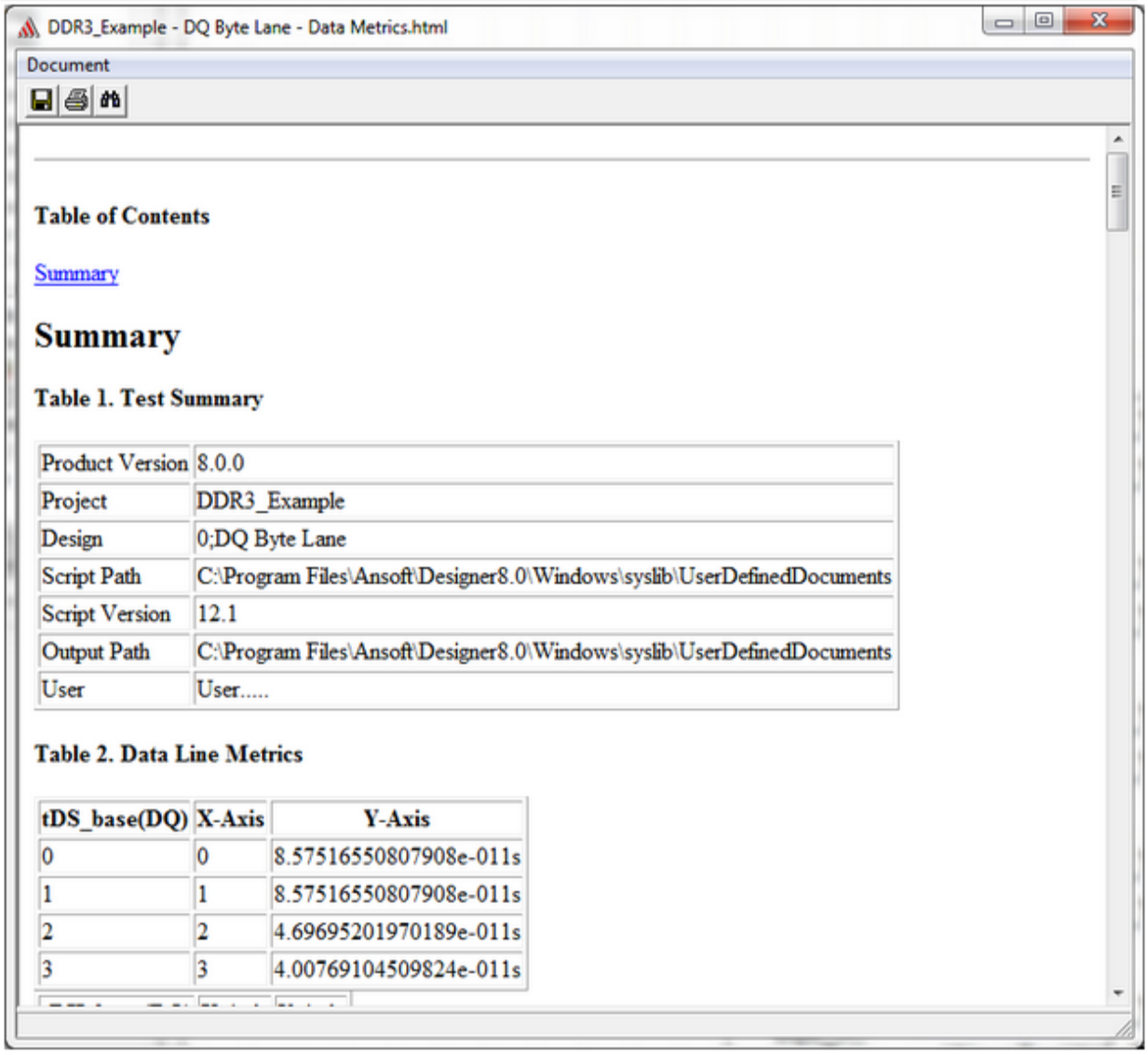


The XML, HTML, and PDF generation require the XML schema file and XSLT style sheets to generate proper output. In addition, the PDF generation requires a FOP executable. You can use the defaults provided in the installation or provide the file paths of your own preferred style sheets and FOP executable installed in his machine.

Viewing UDDs with an HTML Web Browser

View the XML and HTML documents in a web browser with some basic functionality like printing, searching in, and saving the document.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



UDD Script Libraries

You can organize base classes and data files shared between similar UDDs to reuse the code in a better way. All script-library and other support files need to be in a **Lib** subdirectory in the **UserDefinedDefinitions** directory. Any **.py** files found in such **Lib** directories are ignored and not displayed in the GUI as a valid UDD choice. For a UDD script at any given directory depth, all **Lib** directories in their parent directories are added to the system include path (and so, any support script files from any **Lib** directory untill the top level **UserDefinedDefintions** directory can be imported).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The UDD functionality uses IronPython so we have access to all the .NET assembles. If needed, any subset of the UDD functionality can be implemented in any .NET language and used by the UDD script. There are simple rules to follow to achieve this.

1. Build your .NET assembly for .NET 2.0 runtime.
2. Drop the built assembly in any **Lib** directory upstream of the UDD script location: that is, if you have your UDD script in **C:\Users\x\PersonalLib\UserDefinedDefintions\a\b\c\myudd.py** and have a .NET assembly called **com.Acme.UDDLib**. You can keep the .NET assembly under:
   - **UserDefinedDefintions\Lib**
   - **UserDefinedDefintions\a\Lib**
   - **UserDefinedDefintions\a\b\Lib**
   - **UserDefinedDefintions\a\b\c\Lib**
3. If you cannot place the .NET assemblies into a **Lib** directory under **UserDefinedDefintions**, add these lines to your Python script.

   ```
   Import sys

   sys.path.append("full path to your .NET assembly location")
   ```

4. Add these lines to your Python script:

   ```
   Import clr

   clr.AddReference("com.Acme.UDDLib")

   import com.Acme.UDDLib
   ```
   or
   ```
   com.Acme.UDDLib import *
   ```

Related Topics

User Defined Definitions: Python Script API

User Defined Definitions: Python Script API

A User Defined Definition (UDD) extension is implemented as an IronPython script that defines a class with a specific name: **UDDExtension**, which derives from a specific base class **IUDDPluginExtension** and implements its abstract methods.

Import Statements

The base class to use and the types it uses in turn are contained in .NET assemblies. Their use requires that the assemblies are imported into the UDD script. Add these import statements to the top of the Python script:

```
from Ansys.Ansoft.DocGeneratorPluginDotNet.DocGenerator.API.Data
import *
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
from
Ansys.Ansoft.DocGeneratorPluginDotNet.DocGenerator.API.Interfaces
import *
```

UDDExtension Class

The UDD itself should be implemented as an IronPython class called **UDDExtension** which must derive from the **IUDDPluginExtension** abstract base class (from the **Ansys.Ansoft.DocGeneratorPluginDotNet.DocGenerator.API.Interfaces** namespace).

Power users could derive a class hierarchy tuned toward a specific type of UDDs and that they can derive from their own base classes. The only requirement is that directly or indirectly, the UDD class must derive from **IUDDPluginExtension**.

Example

```
def BaseClassUDD ((IUDDPluginExtension):
#base class implementation
…
def UDDExtension ((BaseClassUDD):
#UDD class implementation
…
```

> **Note:**
>
> UDDs are modeled after UDOs, and as such, the usage is similar.

IUDDPluginExtension Abstract Class

Required functions

The **IUDDPluginExtension** abstract class declares the following abstract methods that must be implemented in the **UDDExtension** class or one of its base classes. Not implementing any of these methods will result in a run-time error and a non functioning UDD.

GetUDDName() – Returns a string that is used as a prefix for all solution instances created using this UDD.

Example

```
def GetUDDName(self):
return "MinMaxAvg
```

GetUDDDescription() – Returns a description for the UDD, its purpose, and so on.

Example

```
def GetUDDDescription(self):
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
return "Sample UDD”
```

ShowDefaultSetupDialog() – Returns True if the default dialog box is to be shown. Returns False if you do not want the default dialog box. In this case, you might want to implement/show a customized setup dialog box.

Example

```
def ShowDefaultSetupDialog(self):
return True
```

GetUDDInputParams(List<UDDInputParams> uddInputs) – Returns the list of inputs parameters for the User Defined Document. Returns Boolean: True on success, False on failure.

The supplied input parameters are used to populate details of the parameters to which the UDD user will specify value, specify the input names and their types.

uddInputs: .NET list of **UDDInputParams** objects. The UDD script is expected to add one instance of **UDDInputParams** for each input definition to display. The UDD user will, when creating the UDD, assign a matching value to each such input.

Example

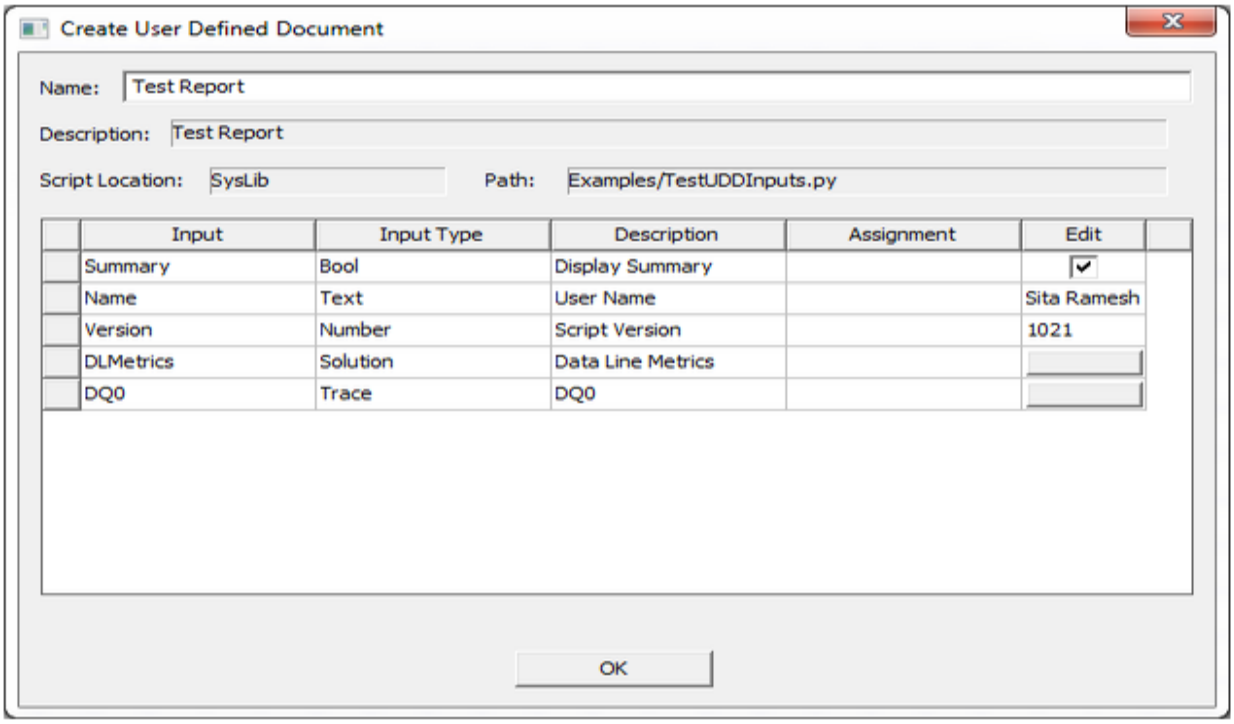
```
def GetUDDInputParams(self, uddInputs)
# Boolean input
param1 = UDDInputParams("Summary","Display Summary",
Constants.kBoolTypeStr, True)
uddInputs.Add(param1)
# Text input
param2 = UDDInputParams("Name","User Name", Constants.kTextTypeStr ,
"Sita Ramesh")
uddInputs.Add(param2)
# Number input
param3 = UDDInputParams("Version","Script Version",
Constants.kNumberTypeStr, 1021)
uddInputs.Add(param3)
# Solution input
param5 = UDDInputParams("DLMetrics","Data Line
Metrics",Constants.kSolutionTypeStr)
uddInputs.Add(param4)
# Trace input
param5 = UDDInputParams("DQ0","DQ0",Constants.kTraceTypeStr)
uddInputs.Add(param5)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential

information of ANSYS, Inc. and its subsidiaries and affiliates.

```
return True
```

Based on the input parameters, the following dialog box appears when you select **Reports** > **Create Document**. The name and description of the UDD also appear in this dialog box.



Generate(List<UDDInputData> uddInputs, IUDDGenerator generator, IProgressMonitor progressMonitor) – This is the main method which accesses the data from the **uddInputs** and generates the document.

uddInputs – The list of inputs that you set up in the dialog box. They are now available to query for data.

generator – This is the document generator object which we use to create different elements of the document like titles, sections, tables, images and write the data too. This interface is explained in the Document Generator Interface document.

progressMonitor – **IProgressMonitor** object. This can be used to set progress for long running calculations, check for user initiated abort, and so on.

Example

```
def Generate(self, input, docgen, progMon):

# Gather data from inputs
boolinput = input[0].Data()
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
textinput = input[1].Data()
dblinput = input[2].Data()

# Get document root
docroot = docgen.GetDocumentRoot()

# Add Section
section1 = docroot.AddSection("Summary", "Overall Results ")

# Add a table
table1 = section1.AddTable("Test Summary")

#Add a table group with 2 columns
tgroup1 = table1.AddTableGroup(2)

# get desktop application
oApp = self.GetUDDAppContext()
if oApp != None:
   oDesktop = oApp.GetAppDesktop()
   if oDesktop != None:
   # version number
   version = oDesktop.GetVersion()
   text1 = tgroup1.AddContent()
   text1 .Add(0, "Product Version")
   text1 .Add(1, version)

   oProject = oDesktop.GetActiveProject()
   if oProject != None:
     projectname= oProject.GetName()
     text1 = tgroup1.AddContent()
     text1 .Add(0, "Project")
     text1 1.Add(1, projectname)

   oDesign = self.GetUDDDesignContext()
   if oDesign != None:
     designname = oDesign.GetName()
     text1 = tgroup1.AddContent()
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
    text1 .Add(0, "Design")
    text1 .Add(1, designname)

  # Provides a script path
  scriptpath = docgen.GetScriptPath()
  text1 = tgroup1.AddContent()
  text1 .Add(0, "Script Path")
  text1 .Add(1, scriptpath )

  #Provides the script version
  text1 = tgroup1.AddContent()
  text1 .Add(0, "Script Version")
    text1 .Add(1, str(dblinput ))

  #Provides the output xml path
  outputpath = docgen.GetOutputFilePath()
  text1 = tgroup1.AddContent()
  text1 .Add(0, "Output Path")
  text1 .Add(1, outputpath )

    #Provides the user information
    text1 = tgroup1.AddContent()
    text1 .Add(0, "User")
    text1 .Add(1, textinput)

# Generate Xml output
docgen.Write(False)

# Generate Html output
docgen.WriteHTML()

# Generate PDF output
docgen.WritePDF()

return True
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Optional functions

SetupUDDInputParams(List<UDDInputParams> uddInputs) – Displays a customized dialog box and returns the user choices for the input params.

uddInputs– .NET list of **UDDInputParams** objects with values for each of them. These can be the user choice for each input obtained through a custom dialog box or some other non graphical assignment.

We cannot process trace and solution types of input with a custom dialog box because there is no way of assigning solution data to the input without the invocation of the reporter dialog box.

Example

```
def SetupUDDInputParams(self, uddInputs)
udddialog = BaseExampleUDDDialog()
if udddialog.ShowDialog() == Forms.DialogResult.OK:
  # Boolean input
  param1 = udddialog.GetInput("Summary")
  uddInputs.Add(param1)

  # Text input
  param2 = udddialog.GetInput("Name")
  uddInputs.Add(param2)

  # Number input
  param3 = udddialog.GetInput("Version")
  uddInputs.Add(param3)
```

HandleUDDEvents(List<string> eventTags) – The tags associated with the event is received by plugin using this abstract class.

This method is the event handler for all link events set by the **SetEventLink()** method on a **IUDDText**. Refer to the definition of the **IUDDText** object in the Document Generator Interface document.

Example

```
def HandleUDDEvents(self, uddLinks):
if uddLinks[0] == "Open" Report":
  # Get Design Name
  oDesign = self.GetUDDDesignContext()
   if oDesign != None:
    oDesign.OpenReport(uddLinks[1])
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 return True
```

GetUDDSchema() – Returns the file path of the schema to validate the XML. This will override the default schema used. Return string containing the full file path of the schema.

```
def GetUDDSchema(self):
return "C:\\Program
Files\\AnsysEM\\v242\\Win64\\common\\docbook\\schema\\xsd\\docbook.xs
d"
```

GetUDDStyleSheetForHtml() – Returns the file path of the style sheet used to generate the HTML document. This will override the default style sheet for HTML. Returns string containing the full file path of the style sheet.

```
def GetUDDStyleSheetForHtml(self):
return "C:\\Program Files\\AnsysEM\\v242\\Win64\\common\\docbook\\"
```

GetUDDStyleSheetForPdf() – Returns the file path of the style sheet used to generate the PDF document. This will override the default style sheet for PDF. Returns string containing the full file path of the style sheet.

Example

```
def GetUDDStyleSheetForPdf(self):
return "C:\\Program
Files\\AnsysEM\\v242\\Win64\\common\\docbook\\xsl\\fo\\docbook.xsl"
```

GetFopExecutable() – Returns the file path of the fop executable used to generate the PDF document. This will override the default style sheet for PDF. Returns string containing the full file path of the fop executable.

Example

```
def GetFopExecutable(self):
return "C:\\Program
Files\\AnsysEM\\v242\\Win64\\common\\ApacheFOP\\fop-1.0\\fop"
```

GetUDDAppContext() – Returns the UDD Owner (if set). This is a Dispatch wrapper that is essentially a COM IDispatch implementation and corresponds to the IDispatch pointing to the desktop app.

GetUDDDesignContext() – Returns the UDD Owner (if set). This is a Dispatch wrapper that is essentially a COM IDispatch implementation and corresponds to the IDispatch pointing to the Design.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Data Types Used in Python Script

There are several types you must use while authoring the Python script. Some of them pass data from the UI to the Python script and to provide an interface for working with this data. Some pass data from the Python script to the UI.

To pass data from the Python script to the UI, the objects of the C# class must be created in Python script using their C# constructors. Then they can be set as functions return values or set to the output parameters using their API.

Constants class

kTraceTypeStr – String constant used to specify an input of trace type.

kSolutionTypeStr – String constant used to specify an input of solution type.

kNumberTypeStr – String constant used to specify an input of number type.

kTextTypeStr – String constant used to specify an input of text type.

kBoolTypeStr – String constant used to specify an input of Boolean type.

kStandardReportStr – String constant to specify a standard report.

kEyeDiagramReportStr – String constant to specify an eye diagram report.

kUserDefinedReportStr – String constant to specify a user-defined report.

kSweepDomainStr – String constant to specify the sweep domain.

kTimeDomainStr – String constant to specify the time domain.

UDDInputParams class

The objects of this class must be created in Python script in the **GetUDDInputParams()** function and the **SetUDDInputParams()** function.

Attributes

Input Name (string)

Input Description (string)

Input Type (Can be Boolean, Number, Text, Trace, or Solution) (string)

BoolData (Boolean)

DoubleData (double)

TextData (string)

ReportType (string)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SolutionName (string)

DomainName (string)

Constructors

UDDInputParams (string name, string description, string type)

UDDInputParams (string name, string description, string type, bool data)

UDDInputParams (string name, string description, string type, double data)

UDDInputParams (string name, string description, string type, string data)

UDDInputParams (string name, string description, string type, string reportType, string solutionName, string domainName)

Property Accessors

Name – Get/Set the name of an input.

Description – Get/Set the description of an input.

Type – Get/Set the type of an input.

BoolData – Get/Set the data of a Boolean input.

DoubleData – Get/Set the data of a number input.

TextData – Get/Set the data of a text input.

ReportType – Get/Set the report type.

SolutionName – Get/Set the name of the solution.

DomainName – Get/Set the name of the domain.

IProgressMonitor Abstract Class

The object of this class is a progress monitor. It is used to display calculations progress in the UI and check if you have requested an abort of the computation.

When displayed in the application, each progress message has four items:

- A task name.
- A sub-task name.
- The progress amount.
- A button to abort the task in progress.

All of this functionality and abort interaction is achieved using these functions:

SetTaskName (string taskName):

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SetSubTaskName (string subTaskName)

BeginTask (string name)

SetTaskProgressPercentage(int progressPercent)

CheckForAbort() – If the generated quantities are computationally expensive, the UDO author can periodically call this method, then call EndTask with Fail and return False.

EndTask (bool passFail)

Example:

```
progMon.BeginTask("Process DQS")
progMon.SetSubTaskName("Compute UI segments")
progMon.SetTaskProgressPercentage(33)
progMon.SetSubTaskName("Compute the rest")
progMon.SetTaskProgressPercentage(100)
progMon.EndTask(True)
```

UDD Input interfaces

The Generate function takes in a list of inputs. These input interfaces let you access data from the design.

IUDDInputBool: This interface exposes three methods.

- **Name()** – Gets the input name.
- **Type()** – Gets the input type.
- **Data()** – Gets the Boolean data, set in the setup dialog box.

IUDDInputDouble: This interface exposes three methods.

- **Name()** – Gets the input name.
- **Type()** – Gets the input type.
- **Data()** – Gets the double data, set in the setup dialog box.

IUDDInputText: This interface exposes three methods.

- **Name()** – Gets the input name.
- **Type()** – Gets the input type.
- **Data()** – Gets the text data, set in the setup dialog box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

IUDDInputTrace: This interface exposes two methods.

- **Name()** – Gets the input name.
- **Type()** – Gets the input type.

DoubleData(): Method used to return x and y double data as a IDictionary<double, double>.

DoubleData(IDictionary<string, string> variation): Method used to return x and y double data as a IDictionary<double, double>, given a variation.

ComplexData(): Method used to return x data and y complex data as a IDictionary<double, double[]>

ComplexData(IDictionary<string, string> variation): Method used to return x data and y complex data as a IDictionary<double, double[]>, given a variation.

TextData(): Method used to return x data and y data as a IDictionary<string, string>.

TextData(IDictionary<string, string> variation) : Method used to return x data and y data as a IDictionary<string, string>, given a variation.

VariableValues(): Method used to get a list of variations as a IList<Dictionary<string, string>>.

IUDDInputSolution: This interface exposes two methods.

- **Name()** – Gets the input name.
- **Type()** – Gets the input type.

DoubleData(string name): Method used to return x and y double data as a IDictionary<double, double>, given a quantity name.

DoubleData(string name, IDictionary<string, string> variation): Method used to return x and y double data as a IDictionary<double, double>, given a quantity name and a variation.

ComplexData(string name): Method used to return x data and y complex data as a IDictionary<double, double[]>, given a quantity name.

ComplexData(string name, IDictionary<string, string> variation): Method used to return x data and y complex data as a IDictionary<double, double[]>, given a quantity name and a variation.

TextData(string name): Method used to return x data and y data as a IDictionary<string, string> given a quantity name.

TextData(string name, IDictionary<string, string> variation): Method used to return x data and y data as a IDictionary<string, string>, given a quantity name and a variation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

CategoryNames(): Method to return a list of category names in the solution as an IList<string>.

QuantityNames(string category): Method to return a list of quantity names in the solution as an IList<string>, given a category.

VariableValues(): Method used to get alist of variations as a IList<Dictionary<string, string>>.

Examples

```
def Generate(self, input, docgen, progMon):
# Getting the boolean data set by the user
boolinput = input[0].Data()
# Getting the double data set by the user
dblinput = input[1].Data()
# Getting the text data set by the user
textinput = input[2].Data()
# Getting the category names in a solution
categories = input[3].CategoryNames()
# Getting the quantity names based on a category
quantities = input[3].QuantityNames(categories[0])
# Getting the XY data from the trace
xydata = input[4].DoubleData()
```

User Defined Document Scripting Interface

To access the **UserDefineddocuments** scripting object, use:

```
Set oModule = oDesign.GetModule("UserDefinedDocuments")
```

Once you have the scripting object, you can use these methods:

```
AddDocument([in] VARIANT data, [in] VARIANT traces, [out, retval]
BSTR* uniqueName)
```

- Takes a VARIANT data which defines the document.
- Takes a VARIANT trace data for the inputs in the document.
- Returns a unique name.

```
EditDocument( [in] BSTR originalName, [in] VARIANT modifiedData, [in]
VARIANT modifiedTraces, [out, retval] BSTR* uniqueName)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Takes the name of the original document.
- Takes a VARIANT data which defines the edited document.
- Takes a VARIANT trace data for the inputs in the document.
- Returns a unique name.

```
RenameDocument( [in] BSTR oldName, [in] BSTR newName)
```

- Takes the name of the original document.
- Takes the new name of the document.

```
DeleteDocument( [in] BSTR name)
```

- Takes the name of the document to delete.

```
UpdateDocument( [in] BSTR name)
```

- Takes the name of the document to update.

```
ViewHtmlDocument( [in] BSTR name)
```

- Takes the name of the document to view in HTML.
- The document must have an existing, generated HTML or PDF.

```
ViewPdfDocument( [in] BSTR name)
```

- Takes the name of the document to view as a PDF.
- The document must have an existing, generated HTML or PDF.

```
SaveHtmlDocumentAs( [in] BSTR name, [in] BSTR saveTo)
```

- Takes the name of the document to save.
- Takes the file path to save the document as.
- The document must have an existing, generated HTML or PDF.

```
SavePdfDocumentAs( [in] BSTR name, [in] BSTR saveTo)
```

- Takes the name of the document to save.
- Takes the file path to save the document as.
- The document must have an existing, generated HTML or PDF.

```
GetDocumentDefinitionNames( [in] BSTR separator, [out, retval]
VARIANT* names)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- *separator* conveys the directory level.
- Returns the file names of doc definitions according to the files in various installation directories.

```
DeleteAllDocuments()

UpdateAllDocuments()
```

The UserDefinedDocument Data Format in the Script

To define a document in VB script:

```
Array("NAME:Test Report", (Name of the document)

"Test Report", (Description of the document)

"SysLib", (Location of the python script(Syslib, Userlib, PeronalLib
etc)

"TestUDDReport", (Relative path of the script in the
UserDefinedDocuments folder)


// Start of input definition //

Array("NAME:Inputs", (Document Inputs keyword)

// Solution input //

Array("NAME:DLMetrics", (Input name)

"Solution", (Solution Input Type)

"Data Line Metrics", (Input Description)

-1, (Solution ID)

-1), (Report ID)

// Trace input //

Array("NAME:DQ0", (Input name)

"Trace", (Trace Input Type)

"DQ0", (Input Description)

-1, (Solution ID)

-1), (Report ID)

// Text input //
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential

information of ANSYS, Inc. and its subsidiaries and affiliates.

```
Array("NAME:Name", (Input name)

"Text", (Text Input Type)

"User Name", (Input Description)

Array("Sita Ramesh")), (Default Value)

// Bool input //

Array("NAME:Summary", (Input name)

"Bool", (Boolean Input Type)

"Display Summary", (Input Description)

Array(true)), (Default Value)

// Number input //

Array("NAME:Version", (Input name)

"Number", (Number Input Type)

"Script Version", (Input Description)

Array(1021)))), (Default Value)

// Trace selection for the solution and trace inputs //

Array("NAME:DocTraces", (Document traces keyword)

// For input “DLMetrics” //

Array("NAME:DLMetrics", (Input name)

// Trace defintion similar to the UDO. This trace definition is a
User defined solution //

Array("User Defined",

"", "DDR3 AC-Timing 4-DQ1", Array("Context:=", ""), Array("Index:=",
Array("All"), "Trise:=", Array("Nominal"), "Tfall:=", Array
("Nominal"), "Pulse_Width:=", Array("Nominal"), "Data_Rate:=", Array
("Nominal"), "Length:=", Array("Nominal")), Array("Probe
Component:=", Array("")), Array())),

// For input “DQ0” //

Array("NAME:DQ0",

// Trace defintion similar to the UDO. This trace definiton is a
Standard solution //
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential

information of ANSYS, Inc. and its subsidiaries and affiliates.

```
Array("Standard", "DQ0", "NexximTransient", Array("NAME:Context",
"SimValueContext:=", Array(1, 0, 2, 0, false, false, -1, 1, 0, 1, 1,
"", 0, 0, "DE", false, "0", "DP", _

false, "20000000", "DT", false, "0.001", "WE", false, "100ns", "WM",
false, _

"100ns", "WN", false, "0ps", "WS", false, "0ps")), Array("Time:=",
Array("All"), "Trise:=", Array( _

"Nominal"), "Tfall:=", Array("Nominal"), "Pulse_Width:=", Array
("Nominal"), "Data_Rate:=", Array("Nominal"), "Length:=", Array
("Nominal")), Array("Probe Component:=", Array( _

 "DQ0")), Array())))
```

Python Script to Define Document

To define a document in Python script:

```
 [
 "NAME:Test Report", "Test Report", "SysLib",
"Examples/TestUDDInputs",
[
"NAME:Inputs",
[
"NAME:DLMetrics","Solution", "Data Line Metrics", -1, -1
],
[
"NAME:DQ0", "Trace", "DQ0", -1, -1
],
[
 "NAME:DQS", "Trace", "DQS", -1, -1
],
[
"NAME:Name", "Text", "User Name", ["Sita Ramesh"]
],
[
 "NAME:Summary", "Bool", "Display Summary", [True]
],
[
 "NAME:Version", "Number", "Script Version" [1021]
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
]
]
 ],
 [
"NAME:DocTraces",
[
 "NAME:DLMetrics",
[
"User Defined", "", "DDR3 AC-Timing 4-DQ1",
[
"Context:=" , ""
],
[
 "Index:=" , ["All"], "Trise:=" , ["Nominal"], "Tfall:=" ,
["Nominal"], "Pulse_Width:=" ,
["Nominal"], "Data_Rate:=" , ["Nominal"], "Length:=" , ["Nominal"]
],
[
"Probe Component:=" , [""]
],
[]
]
],
[
"NAME:DQ0",
[
 "Standard", "DQ0", "NexximTransient",
[
"NAME:Context", "SimValueContext:=" ,
[1,0,2,0,False,False,-
1,1,0,1,1,"",0,0,"DE",False,"0","DP",False,"20000000","DT",False,"0.0
01","WE",False,"100ns","WM",False,"100ns","WN",False,"0ps","WS",False
,"0ps"]
],
[
 "Time:=" , ["All"],"Trise:=" , ["Nominal"],"Tfall:=" ,
["Nominal"],"Pulse_Width:=" ,["Nominal"],
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
"Data_Rate:=" , ["Nominal"], "Length:=" , ["Nominal"]
],
[
"Probe Component:=" , ["DQ0"]
],
[]
]
],
 ]
```

Sample Script

This script adds, edits, renames, and deletes a document.

```
Set oModule = oDesign.GetModule("UserDefinedDocuments")
' Add a UDD
oModule.AddDocument Array("NAME:Test Report1", "Test Report",
"SysLib", _
"Examples/TestUDDInputs", Array("NAME:Inputs", Array
("NAME:DLMetrics", "Solution", _
"Data Line Metrics", -1, -1), Array("NAME:DQ0", "Trace", "DQ0", -1, -
1), Array("NAME:DQS",
"Trace", "DQS", -1, -1), Array("NAME:Name", "Text", "User Name",
Array("Sita Ramesh")), Array("NAME:Summary", "Bool", "Display
Summary", Array(true)), Array("NAME:Version", "Number", "Script
Version"))), Array("NAME:DocTraces", Array("NAME:DLMetrics", Array
("User Defined", "", "DDR3 AC-Timing 4-DQ1", Array("Context:=", ""),
Array("Index:=", Array("All"), "Trise:=", Array( "Nominal"),
"Tfall:=", Array("Nominal"), "Pulse_Width:=", Array("Nominal"),
"Data_Rate:=", Array( "Nominal"), "Length:=", Array("Nominal")),
Array("Probe Component:=", Array("")), Array())), Array("NAME:DQ0",
Array( _
"Standard", "DQ0", "NexximTransient", Array("NAME:Context",
"SimValueContext:=", Array( _
1, 0, 2, 0, false, false, -1, 1, 0, 1, 1, "", 0, 0, "DE", false, "0",
"DP", false, "20000000", "DT", false, "0.001", "WE", false, "100ns",
"WM", false, "100ns", "WN", false, "0ps", "WS", false, "0ps")), Array
("Time:=", Array("All"), "Trise:=", Array("Nominal"), "Tfall:=",
Array("Nominal"), "Pulse_Width:=", Array("Nominal"), "Data_Rate:=",
Array("Nominal"), "Length:=", Array("Nominal")), Array("Probe
Component:=", Array("DQ0")), Array())))
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
' Edit Document
oModule.EditDocument "Test Report1", Array("NAME:Test Report", "Test
Report", _
"SysLib", "Examples/TestUDDInputs", Array("NAME:Inputs", Array
("NAME:DLMetrics", _
"Solution", "Data Line Metrics", 1000001, 0), Array("NAME:DQ0",
"Trace", "DQ0", 32, _
 2), Array("NAME:DQS", "Trace", "DQS", 32, 4), Array("NAME:Name",
"Text", "User Name", Array( "Sita Ramesh")), Array("NAME:Summary",
"Bool", "Display Summary", Array(true)), Array("NAME:Version",
"Number", "Script Version"))), Array("NAME:DocTraces", Array
("NAME:DLMetrics", Array( "User Defined", "Solution", "DDR3 AC-Timing
4-DQ1", Array("Context:=", ""), Array("Index:=", Array( "All"),
"Trise:=", Array("Nominal"), "Tfall:=", Array("Nominal"), "Pulse_
Width:=", Array( "Nominal"), "Data_Rate:=", Array("Nominal"),
"Length:=", Array("Nominal")), Array("Probe Component:=", Array("")),
Array())), Array("NAME:DQ0", Array("Standard", "DQ1",
"NexximTransient", Array("NAME:Context", "SimValueContext:=", Array
(1, 0, 2, 0, false, false, -1, 1, 0, 1, 1, "", 0, 0, "DE", false,
"0", "DP", _
 false, "20000000", "DT", false, "0.001", "WE", false, "100ns", "WM",
false, "100ns", "WN", false, "0ps", "WS", false, "0ps")), Array
("Time:=", Array("All"), "Trise:=", Array("Nominal"), "Tfall:=",
Array("Nominal"), "Pulse_Width:=", Array("Nominal"), "Data_Rate:=",
Array("Nominal"), "Length:=", Array("Nominal")), Array("Probe
Component:=", Array("DQ1")), Array())))

' Rename a UDD
oModule.RenameDocument "Test Report", "Test UDD Report"

' Update UDD
oModule.UpdateDocument "Test UDD Report"

' View Html
oModule.ViewHtmlDocument "Test UDD Report"

' View Pdf
oModule.ViewPdfDocument "Test UDD Report"

' Save Html
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential
information of ANSYS, Inc. and its subsidiaries and affiliates.

```
oModule.SaveHtmlDocumentAs "Test UDD Report",
"c:/AnsysProjects/Test.html"

' Save pdf
oModule.SavePdfDocumentAs "Test UDD Report",
"c:/AnsysProjects/Test.pdf"

‘ Delete UDD
oModule.DeleteDocument "Test UDD Report"
```

> **Note:**
>
> The product implements the **GetModule** call to create the **UserDefinedDocument** scripting object. For example, **Check AltraSimDesign.cpp** (function **GetMgrIDispatch()**).

Document Generator Interfaces

This document briefly describes the API interfaces available in the document generator plugin.

(Ansys.Ansoft.DocGeneratorPluginDotNet.dll)

Scripting objects available in the script for the **Generate** function:

oApp = self.GetUDDAppContext(

- Gets the application context.
- Usage – Gets the active project ad the version of the product.

```
oDesktop = oApp.GetAppDesktop()
 if oDesktop != None:
vr = oDesktop.GetVersion()
oProject = oDesktop.GetActiveProject()
```

oDesign = self.GetUDDDesignContext()

- Gets the design context.
- Usage – Gets the design name.

```
oDesign = self.GetUDDDesignContext()
if oDesign != None:
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
nm = oDesign.GetName()
```

IUDDGenerator interface

This interface is available in the **Generate** method of the **UDDPluginExtension**.

This interface can be used to:

- Set the document output file path.

```
docgen.SetOutput("C:\\Examples\\DocumentOutput.xml")
```

- Get the document root.

```
docroot = docgen.GetDocumentRoot()
```

- Write out to the output file.

```
docgen.Write()
```

- Write an HTML document.

```
 void WriteHTML();
```

- Write a PDF document.

```
void WritePDF();
```

- Load the HTML transform object.

```
void LoadHTMLTransform();
```

- Load the cached PDF transform object.

```
 void LoadPDFTransform();
```

- Get the script path.

```
string GetScriptPath();
```

- Get the output file path.

```
string GetOutputFilePath();
```

IUDDRoot interface

Calling **GetDocumentRoot()** on the **IUDDGenerator** interface provides this interface. This interface can be used to:

- Add a new section to the document. Provide a section title.

```
section1 = docroot.AddSection("Section title")
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Add a new section to the document. Provide a section title and subtitle.

```
section1 = docroot.AddSection("Section title", "Section subtitle")
```

- Add a new title.

```
section1 = docroot.AddTitle("Title")
```

- Add a new subtitle.

```
section1 = docroot.AddSubtitle("Subtitle")
```

IUDDSection interface

Calling **AddSection()** on the **IUDDRoot** interface provides you with the this interface. This interface can be used to:

- Set an ID for the section for internal links.

```
 section1.SetID("id")
```

- Add a new table to the document. Provide a table title.

```
table1 = section1.AddTable("Table title")
```

- Add a new image to the document. Provide an image title and a file path to the image file.

```
image1 = section1.AddImage("Image title")
```

- Add text to the document.

```
text1 = section1.AddText("Random text........")
```

IUDDImage interface

Calling **AddImage()** on the **IUDDSection** interface provides you with the this interface. On this interface you can call the following methods:

- Set an ID for the image for internal links.

```
image1.SetID("id")
```

- Set alignment information. Can be **center**, **left**, and **right**.

```
image1.SetAligment("center")
```

- Set the file path of the image file. Not necessary if image file path is set through the **AddImage()** method.

```
image1.SetFileRef("Image path")
```

- Set the format of the image file. Can be **BMP**, **PNG**, **JPEG**, **JPG**, **DVI**, and so on.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
image1.SetFormat("format")
```

IUDDText interface

Calling **AddText()** on the **IUDDSection** interface provides you with the this interface. On this interface you can call the following methods:

- Set an ID for the text for internal links.

```
text1.SetID("id")
```

- Set the emphasis attribute on the text.

```
text1.SetEmphasis()
```

- Set the quotes attribute on the text.

```
Text1.SetQuotes()
```

- Set the block quotes attribute on the text.

```
text1.SetBlockquotes()
```

- Set quotes on the text.

```
Text1.SetQuotes()
```

- Set the wordsize attribute on the text.

```
text1.SetSize(size as an integer)
```

- Set a link to an ID of any element to provide internal links.

```
text1.SetLink("linkname")
```

- Set an event link to handle an event. The **HandleUDDEvents** method must be implemented in the script to handle the event.

```
text1.SetEventLink("linkname")
```

IUDDTable interface

Calling **AddTable()** on the **IUDDSection** interface provides you with the this interface. On this interface you can call the following methods:

- Set an ID for the table for internal links.

```
table1.SetID("id")
```

- Set alignment information. Can be **center**, **left**, and **right**.

```
table1.SetAlignment("center")
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Set the background color of the table.

```
table1.SetBgColor(string bgcolor)
```

- Set the frame type. Can be **all**, **bottom**, **top**, **sides**, **topbot**.

```
table1.SetFrame(string frame)
```

- Add a table group and specify the number of columns. A table can have multiple table groups.

```
IUDDTableGroup table1.SetTableGroup(int columns)
```

IUDDTableGroup interface

Calling **AddTableGroup()** on the **IUDDTable** interface provides you with the this interface. On this interface you can call the following methods.

- Set an ID for the table group for internal links.

```
tgroup1.SetID("id")
```

- Set alignment information. Can be **center**, **left**, and **right**.

```
tgroup1.SetAlignment("center")
```

- Set the column width of a column given the index of the column and the required width. Width can be set in two ways.
    - Width can be set relative to 1. For example, setting it to **2** makes the column width double the width of the others.
    - If the entire table width is considered to be 99.99 units, width can be a number relative to this.

```
tgroup1.SetColumnWidth(int index, string width)
```

- Add a header to the table group

```
IUDDTableRow tgroup1.AddHeader()
```

- Add a header with multiple rows to the table group. Takes number of sub rows.

```
IUDDTableRow tgroup1.AddHeader(int rows)
```

- Add a row of content to the table group.

```
IUDDTableRow tgroup1.AddContent()
```

- Add content with multiple rows to the table group. Takes number of sub rows.

```
IUDDTableRow tgroup1.AddContent(int rows)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

IUDDTableRow interface

Calling **AddHeader()** & **AddContent()** on the **IUDDTableGroup** interface provides you with the this interface. On this interface you can call the following methods:

- Set an ID for the table row for internal links.

```
trow1.SetID("id")
```

- Set alignment information. Can be **center**, **left**, and **right**.

```
trow1.SetAlignment("center")
```

- Set cell text. Can be cell content or header text. Takes a column index and a text string. It is added to the first row.

```
IUDDTextElement trow1.Add(int column, string text)
```

- Set cell text. Can be cell content or header text. Takes a column index, row index and a text string. Takes in a row number because a table row can have multiple sub rows.

```
IUDDTextElement trow1.Add(int column, int subrow, string text)
```

- Set cell content. Takes a column index and an int value. It is added to the first row.

```
IUDDTextElement trow1.Add(int column, int value)
```

- Set cell content. Takes a column index, row index, and an int value.

```
IUDDTextElement trow1.Add(int column, int subrow, string text)
```

- Set cell text. Takes a column index and a double value. It is added to the first row.

```
IUDDTextElement trow1.Add(int column, double value)
```

- Set cell text. Takes a column index, row index and a double value.

```
IUDDTextElement trow1.Add(int column, int subrow, double value)
```

- Set cell text spanning two columns. Can be cell content or header text. Takes a sub row index, starting column index, ending column index, and a text string.

```
IUDDTextElement trow1.AddSpanningcolumnst(int subrow, int
columnstart, int columnend, string text)
```

- Set cell text. Can be cell content or header text. Takes a column index, starting sub row index, ending sub row index, and a text string.

```
IUDDTextElement trow1.AddpanningRows(int column, int subrowstart,
int subrowend, string text)
```

IUDDTableRow interface

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Calling **Add()** on the **IUDDTableGroup** interface provides you with the this interface. On this interface you can call the following methods:

- Set an ID for the table row for internal links.

```
trow1.SetID("id")
```

- Set alignment information. Can be **center**, **left**, and **right**.

```
trow1.SetAlignment("center")
```

- Includes all the methods exposed by the **IUDDText** interface.

Chip Model Analyzer (CMA)

Ansys Chip Model Analyzer (CMA) is an advanced Chip Power Model (CPM) creation tool for performance checking and failure diagnosis of the Power Delivery Network (PDN). CMA can use either HSpice or Nexxim simulators. HSpice and external Nexxim require licenses, but the Nexxim Internal simulator does not.

CMA does not currently interact with Twin Builder. However, you can launch CMA from within Twin Builder. Select **Tools** > **Chip Model Analyzer (CMA)**.

From there, you can import CPM files for analysis or create a pseudo CPM.

For more information, launch CMA and select **Help** > **Content**to access the CMA training documentation.

PinToPin Utility

The PinToPin utility can quickly extract HFSS or SIwave models for specified nets of package and PCB geometries. It establishes a repeatable process for robust geometry extraction, port configuration, and passive model assignment. Execute the process with the user interface or in non-graphical mode.

For more information, consult the SIwave and HFSS 3D Layout help.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

21 - C-Models in Twin Builder

Twin Builder C-Models provide a powerful method for describing components and equations using C or C++. You can use Twin Builder C-Models like any other Twin Builder components in network simulations or block diagrams. Twin Builder's support for the creation and use of C-Models provides the model developer with functions to control and influence the simulator behavior and internal solving algorithms.

Introduction to the Twin Builder C Interface

Use the Twin Builder C interface to program arbitrary nonlinear algebraic and ordinary differential equations to be solved by the Twin Builder simulator. This modeling approach is suitable for following applications:

- Model description, such as:
  - Loops and iterations.
  - Extensive mathematical operations.
  - Multiple coupled ordinary differential equations.
- Reuse of already present C/C++ code.
- Inclusion and development of micro-controller source code.

The Twin Builder C Interface also allows the modeling of characteristics used to establish nonlinear relationships of one input and one output quantity. Various Twin Builder basic models can be connected to such blocks.

C-Models show high simulation speed, accuracy, and numerical stability in Twin Builder simulations. They can be used in DC, AC, and transient simulations. It is possible to program variable step-size models as well as fixed step-size models. These latter allow for example to run micro-controller code at exactly the same sample times like on the real hardware. This is essential for realistic simulation of this kind of systems.

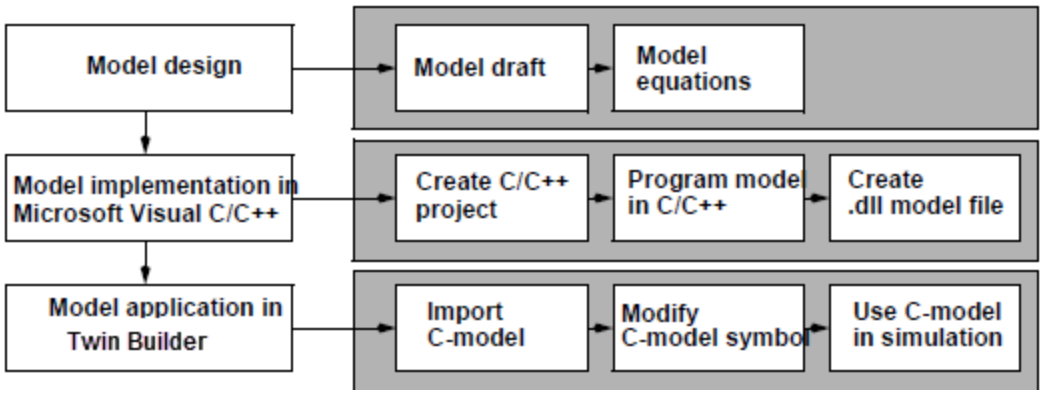
Twin Builder C Interface Overview

Developing a C-Model consists essentially of three steps:

- Designing the C-Model.
- Implementing (programming) the model in C/C++.
- Using the model in Twin Builder.

To assist in model development, Twin Builder provides a C-Model Editor.The following figure shows these steps, from creating the model draft to using the model in Twin Builder. If the problem is complex and comprehensive, individual steps may need to be repeated.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



The following steps outline the general methodology for defining, programming, and using a C-Model:

1. Define the model draft that you want to implement as a C-Model.
2. Define the model equations.
3. Create a C/C++ project.
4. Program the model in C/C++ code.
5. Create the dynamic link library (DLL) model file.
6. Import the C-Model into Twin Builder.
7. Modify the C-Model symbol.
8. Use the C-Model in simulation models.

Program Requirements

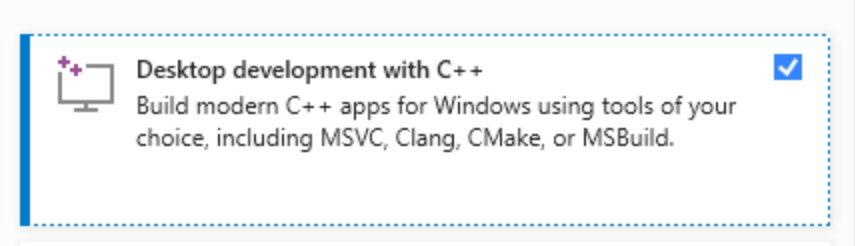
Developing C-Models requires that one of these C++ compilers is installed:

- Microsoft® Visual Studio Professional or Community Version
  - Visual Studio or Visual Studio Community 2022
  - Visual Studio or Visual Studio Community 2019
  - Visual Studio or Visual Studio Community 2017
  - Visual Studio or Visual Studio Community 2015

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

- If no professional version is available, we recommend that you use the Visual Studio Community version.
- You must install the workload package **Desktop development with C++** before using Visual Studio 2017 (or later) C++ Compiler.



- By default, the compilation of C-models in the C-model editor does not require the MFC package to be installed. If the C-model does not require MFC functionality, set **Use MFC** to **No** in the C-model editor build settings. If needed, you must manually add the MFC package to the list of packages in the C++ development package, since it is not installed automatically by the VS installer.

Installation

Developing C-Models requires that the Microsoft compiler is installed. During installation of the Twin Builder C Interface, the required preparations in the Microsoft Developer Studio environment are carried out.

Designing a C-Model

Before you can program and use C-Models in Twin Builder, first develop the model equations. The help topics in this section provide an overview of the design considerations to observe while defining equations for your models.

Defining Model Equations

All components in a Twin Builder simulation model exchange data with other components using conservative and/or non-conservative nodes. Conservative nodes are nodes where laws of conservation (for example, the sum of currents = 0) are applicable whereas non-conservative nodes are used as signal ports without a flow quantity. Every conservative node has an assigned potential (v) and flow (i) quantity (across and through quantity). For electrical domain

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

components these variables are electrical voltage and current. In addition to across and through quantities, components can contain internal state variables x_j which are linked together and to the flows and potentials through algebraic or ordinary differential equations. For every conservative node a model has, an equation linking the flow through the pin as a function to the potentials and the internal state variables is needed:

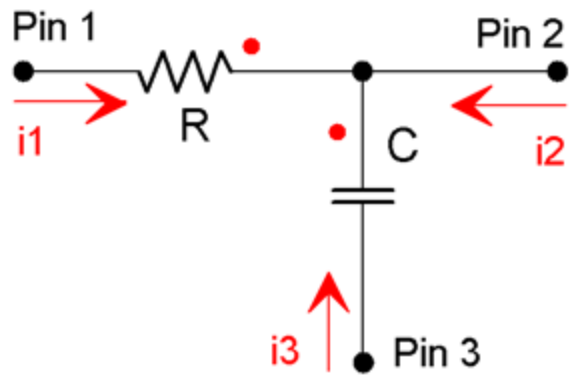
$i_{1,...k}=i_{1,...k}(v_{1,...k},x_{1,...n})$

In addition, every internal state variable requires an equation linking its value to the other variables:

$F_{1,...n}(v_{1,...k},x_{1,...n})=0$

RC Example

The RC combination has three electrical pins and one internal state variable, the charge Q (= x) on the capacitor. The system needs three equations describing the currents in and out of the three conservative pins $i_1=i_1(v_1,v_2,v_3,Q)$, $i_2=i_2(v_1,v_2,v_3,Q)$, $i_3=i_3(v_1,v_2,v_3,Q)$ and one equation describing the link of the internal state variable to the potentials $F(Q,v_1,v_2,v_3)=0$.



If it is impossible to explicitly solve the flow equation for an input current (for example i_j), the input current must be used as internal variable: $x_l=i_j$. So Twin Builder will numerically solve for the flow variable.

Twin Builder handles the mathematical description of components using the Jacobian matrix description, where **J** is the Jacobian matrix, **s** is the solution vector and **r** the right side vector of the equation system.

For this purpose, the node potentials and the internal quantities are to be lined up in a solution vector. Due to this definition of the solution vector the corresponding Jacobi matrix and right hand side vector of the system is defined.

You must enter the Jacobian matrix elements and right side entries. In the case of linear equations, only the constant terms $i^0_l=i_l(v_{1,...k}=0,x_{1,...n}=0)$ and $F^0_j=F_j(v_{1,...k}=0,x_{1,...n}=0)$ in the right side vector r_{lin} are necessary.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The simulator adds the variable parts. This feature can also be switched off to fill the right side manually. When it comes to nonlinear equations, you must fill the complete corresponding right side entry. Ordinary differential equations fit into this scheme by using the exchangeability of differential operators.

$$\frac{\partial}{\partial x}\left(\frac{d}{dt}\right) = \frac{d}{dt}\left(\frac{\partial}{\partial x}\right)$$

Twin Builder uses the same Jacobian matrix scheme in transient, DC and AC simulations. You must select and establish the suited equations for the different simulation domains.

Example

The node behavior of a simple ohmic resistor is described by the following two equations:

$$i_1(v_1, v_2) = \frac{v_1 - v_2}{R} \qquad i_2(v_1, v_2) = -\frac{v_1 - v_2}{R}$$

Manually filled right side: $\underline{r_{manual}} = \begin{pmatrix} -i_1 \\ -i_2 \end{pmatrix} = \begin{pmatrix} -\frac{v_1 - v_2}{R} \\ \frac{v_1 - v_2}{R} \end{pmatrix}$

Automatically filled right side: $\underline{r_{automatic}} = \begin{pmatrix} -i_1^0 \\ -i_2^0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

If the resistor is a linear system, its right side is filled. Since $i_1=Fj(v_{1,\ldots k}=0,x_{1,\ldots n}=0)$, there are no constant terms to be filled on the right side.

Transient (TR) Equations

In transient simulations, the Twin Builder simulator calculates the temporal course of potential and flow quantities and component internal quantities. For this purpose, you must establish equations describing the links of instantaneous signals and their temporal development. These equations can generally be reduced to 1st order ordinary differential equations well suited for use in transient Twin Builder simulations. The Jacobian scheme for transient simulations is:

$$J_{TR} \cdot \underline{s}(t) = \underline{r}(t)$$

RC Example

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The transient behavior of the RC combination is described by the following equations:

$$i_1 = i_R = \frac{v_1 - v_2}{R}$$

$$i_2 = -i_R + i_C = -\frac{v_1 - v_2}{R} + \frac{dQ}{dt}$$

$$i_3 = -i_c = -\frac{dQ}{dt}$$

$$F(Q, v_1, v_2, v_3) = v_2 - v_3 - \frac{Q}{C} = 0$$

Using the symbolic operation the equation system results to (E1):

$$\begin{pmatrix} \frac{1}{R} & -\frac{1}{R} & 0 & 0 \\ -\frac{1}{R} & \frac{1}{R} & 0 & \frac{d}{dt} \\ 0 & 0 & 0 & -\frac{d}{dt} \\ 0 & 1 & -1 & -\frac{1}{C} \end{pmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ Q \end{bmatrix} = \begin{pmatrix} -\left(\frac{v_1 - v_2}{R}\right) \\ -\left(-\frac{v_1 - v_2}{R}\right) \\ 0 \\ -\left(v_2 - v_3 - \frac{Q}{C}\right) \end{pmatrix}$$

DC Equations

DC Analysis results in steady-state values of the potential, flow, and internal quantities. This means the DC-equations contain no dynamic effects. DC behavior can be described by algebraic equations. In general, they can be derived from the TR equations by setting the temporal derivatives to zero. Expressed in the Jacobian scheme the equation system can be reduced to:

$$J_{DC} \cdot \underline{s}_{DC} = \underline{r}_{DC}$$

RC Example

The DC behavior of the RC-combination can be described by the following equations, which were extracted from the TR-equations:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$i_1 = i_R = \frac{v_1 - v_2}{R}$$

$$i_2 = -i_R + i_C = -\frac{v_1 - v_2}{R}$$

$$i_3 = -i_c = 0$$

$$F(Q, v_1, v_2, v_3) = v_2 - v_3 - \frac{Q}{C} = 0$$

Using the symbolic operation the equation system results to (E2):

$$\begin{pmatrix} \frac{1}{R} & -\frac{1}{R} & 0 & 0 \\ -\frac{1}{R} & \frac{1}{R} & 0 & \frac{d}{dt} \\ 0 & 0 & 0 & -\frac{d}{dt} \\ 0 & 1 & -1 & -\frac{1}{C} \end{pmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ Q \end{bmatrix} = \begin{pmatrix} -\left(\frac{v_1 - v_2}{R}\right) \\ -\left(-\frac{v_1 - v_2}{R}\right) \\ 0 \\ -\left(v_2 - v_3 - \frac{Q}{C}\right) \end{pmatrix}$$

AC Equations

In contrast to the TR analysis which deals with the temporal course of signals, the AC analysis calculates system behavior in the conjugated frequency domain. The complex description of signals in this domain is generally applied:

- $s(\omega)=s_r(\omega)+j \cdot s_i(\omega)$, where $\omega=2\pi \cdot f$ is the angular frequency.
- $s_r(\omega)$ is the real part.
- $s_i(\omega)$ is the imaginary part of the complex signal $s(\omega)$.

For the complex calculation of the signals, the same Jacobian formalism is used, as in the case of TR- and DC- analysis:

$$\hat{J}_{AC} \cdot \hat{\underline{x}}(\omega) = \hat{\underline{r}}(\omega)$$

Jacobian matrix entries, right side entries and the solution vector are complex numbers. The AC analysis is performed on a fixed operating point. Such an operating point is a solution of the TR-equation-system at a fixed time step. This means the transient equations are almost literally applicable for the AC analysis. Present temporal derivatives are replaced by their formal Laplace transformation:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$\frac{d}{dt} \rightarrow j\omega$$

RC Example

In the case of the RC combination, the AC equations appear in the following manner:

$$i_1 = \hat{i}_R = \frac{\hat{v}_1 - \hat{v}_2}{R}$$
$$i_2 = -\hat{i}_R + \hat{i}_C = -\frac{\hat{v}_1 - \hat{v}_2}{R} + j\omega \cdot Q$$
$$i_3 = -\hat{i}_c = -j\omega \cdot Q$$
$$\hat{F}(Q, \hat{v}_1, \hat{v}_2, \hat{v}_3) = \hat{v}_2 - \hat{v}_3 - \frac{Q}{C} = 0$$

Using the symbolic operation, the equation system results to (E3):

$$\begin{pmatrix} \frac{1}{R} & -\frac{1}{R} & 0 & 0 \\ -\frac{1}{R} & \frac{1}{R} & 0 & j\omega \\ 0 & 0 & 0 & -j\omega \\ 0 & 1 & -1 & -\frac{1}{C} \end{pmatrix} \cdot \begin{bmatrix} \hat{v}_1 \\ \hat{v}_2 \\ \hat{v}_3 \\ Q \end{bmatrix} = \begin{pmatrix} -\left(\frac{\hat{v}_1 - \hat{v}_2}{R}\right) \\ -\left(-\frac{\hat{v}_1 - \hat{v}_2}{R} + j\omega\right) \\ j\omega \\ -\left(\hat{v}_2 - \hat{v}_3 - \frac{Q}{C}\right) \end{pmatrix}$$

Implementing C-Models

Once you have drafted a model design and developed the model's equations, you can implement them in the following environment:

- Microsoft Visual Studio Project Environment
- C-Model Editor

It is within this environment that you program C-Models, build the model DLL files, and also modify and debug your model programs.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Microsoft Visual Studio Project Environment

The topics in this section describe operations facilitated in a Microsoft® Visual Studio project environment.

Defining C-Models in Microsoft® Visual Studio

Follow this procedure to program a new C-Model in Microsoft Visual Studio:

1. Create a C-Model project for Microsoft Visual Studio.

   > **Note:**
   >
   > Twin Builder does not provide a Microsoft Visual Studio project wizard to create a new C-Model project. However, you can use the internal C-Model editor to create the base structure of the project and export the project to Microsoft Visual Studio.

   a. Add a C-Model using Twin Builder‘s C-Model editor.

      The C-Model editor creates all necessary files and project settings for a basic Twin Builder C-Model. It installs two different configurations: **Release** and **Debug**. Debug is the default configuration. Building the project results in a Windows DLL model file for use with Twin Builder.

   b. Select **C-Model Editor** > **File Settings** > **Model Source** and set it to **Path to Project**.
   c. Export the C-Model project to Microsoft Visual Studio.
   d. Load the exported project into Microsoft Visual Studio.
2. Check the search path setting for header files.
   a. Choose **Project** > **Properties** and select the **C/C++** and **General** folders.
   b. Select **All Configurations** from the configuration list.
   c. Check the entry in the **Additional include directories** box and click **OK**.The path must point to the location of the files **Sim2000User.h**, **Complex.h**, and **DataTypes_Basic.h**. These files are usually located in **…\Twin Builder\syslib\bin\Template\C_Interface** below the Twin Builder installation path.
3. Adapt location of DLL model files.

   The location of the DLL model file depends on the used configuration. The default paths are **…\Release** and **…\Debug**. To use the C-Model in Twin Builder, place the corresponding DLL model file in the Twin Builder directory or in the current simulation

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

project folder. You can change the C/C++ project paths to one of these folders to avoid the copy operation.

   a. Choose **Project** > **Settings** and select the **Linker** and **General** folders.
   b. Select **a configuration** (debug or release) from the configuration list.
   c. Adapt the entry in the **Output file name** box for the selected configuration and click **OK**.

4. Check the search path setting for library files.
   a. Choose **Project** > **Properties** and select the **Linker** and **General** folders.
   b. Select **All Configurations** from the configuration list.
   c. Check the entry in the **Additional library directories** box and click **OK**. The path must point to the location of the files **Sim2000User.lib** and **Complex.lib**. These files are usually located in **…\Twin Builder\syslib\bin\Template\C_Interface\Lib** below the Twin Builder installation path.
5. Choose **File** > **Open** to open the source file *project_name*.**cpp** in the **Source Files** folder. It represents a template for programming a Twin Builder C-Model.
6. Add your modifications to the source file to define the model behavior.
7. Choose **Build** > **Configuration Manager** to set the desired configuration for the created DLL model file. Use the **Debug** configuration during the model development and debugging phase, use the **Release** configuration for the final model build.
8. Choose **Build** > **Build Solution** to build the Dynamic Link Library of the model. After successful compilation and linkage, the DLL model file is ready to be introduced to Twin Builder.
9. Start Twin Builder and select **Tools** > **Edit Libraries** > **Models** to include the C-Model in a model library.

Modifying C-Models in Microsoft Visual Studio

During the development and debug phase of a C-Model, it is necessary to modify the source code and test its effects in Twin Builder. Follow this procedure to modify and test C-Models:

1. Open Microsoft® Visual Studio.
2. Load the Microsoft Visual Studio workspace file (SLN or VCXPROJ) of the C/C++ project.
3. Include and add changes to the model source code file (CPP).

> **Warning:**
>
> Make sure there is no Twin Builder simulation running that uses the open C-Model. If the C-Model DLL file is already in use, the linker cannot overwrite it.

4. Choose **Build** > **Build Solution** to compile and link the model source code.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5. Make sure the linker stored the file on the right location accessible to Twin Builder. If necessary, copy the new DLL model file to the right location.
6. If changes were made that require the C-Model to be updated, use **Tools** > **Import Simulation Model** and select the changed DLL.
7. The model is now ready to use. Start the simulations from Twin Builder Schematic.
8. For further model source code changes repeat Steps 3 through 7 as needed.

Related Topics

Updating C-Models

Debugging C-Models in Microsoft Visual Studio

Easy debugging of the model source code is very important in the model development phase. For this purpose, you can use the Microsoft debugger which comes with the Developer Studio suite. For details about this software, see its manuals.

Follow this procedure to debug Twin Builder C-Models:

1. Open Microsoft® Visual Studio.
2. Load the Microsoft Visual Studio workspace file (SLN or VCXPROJ) of the C/C++ project.
3. Choose **Build** > **Configuration Manager** and set the active configuration to **Debug**.
4. Make sure that you build your DLL in a library directory (either *PersonalLib***bin** or *UserLib***bin**).
   a. To find out the current location of the C-Model DLL open the **Edit Component** dialog for the C-Model component and switch to the **Simulation Model** tab. If the location says **Stored with Model**, then the DLL was extracted to the **.aedtresults\bin** directory of the project.
   b. In the Visual Studio project settings, choose **Project** > **Properties** and select the **Linker**. Set the path of the PDB file in the **Debugging** subfolder in the **Generate Program Database File** box and the path of the DLL in the **General** subfolder in the **Output File** box.
5. If not already done, import your DLL model into Twin Builder.
6. Add an ASSERT(FALSE) statement into the **Initialize** function and set at least one break point at the statement after the ASSERT.
7. Run the simulation until the ASSERT message box comes up.
8. Switch to the Microsoft Visual Studio workspace and attach to the **SimplorerSolver.exe** process (**Debug** > **Attach to Process**).
9. Switch back to the ASSERT message box and click **Ignore**.
10. The debugger will stop at the break point you set.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

11. Set breakpoints and other debug related settings in Microsoft Visual Studio (see its manual for help).
12. After the end of the simulation, the simulator shuts down and the debugger stops.

Programming C-Models

Using the Twin Builder project template or the C-Model editor, programming Twin Builder C-Models is an easy and straightforward task.

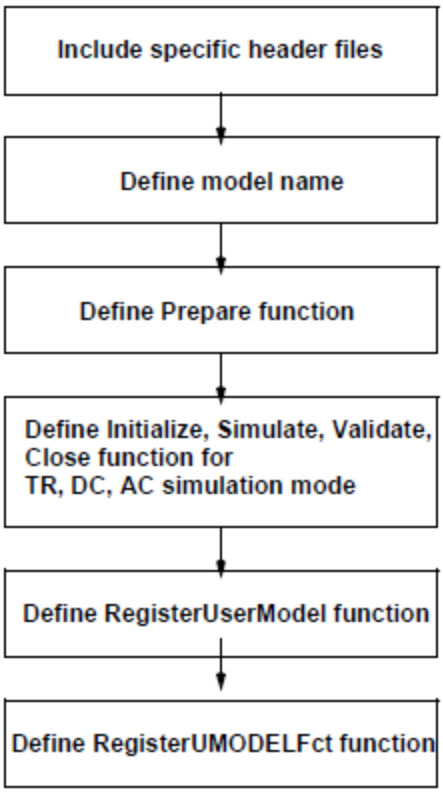
Refer to the **RC3pin** example in the Twin Builder path **...\Examples\Applications\C_Interface**. It is used to describe C-Model functions in this section.

> **Note:**
>
> A reference of Twin Builder-specific C/C++ functions appears at the end of this chapter.

Basic Steps

There are a few basic constructions needed in every model source file to ensure a proper link to the Twin Builder simulator. These statements are included in the template file and can be extended and adapted to model specific needs.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Include Twin Builder-specific Header Files

The following Twin Builder-specific header files must be included:

```
#include "SIMPLORERAfx.h"

#include "Sim2000User.h"

#include "Complex.h"
```

Their purpose is mainly to define standardized data types for data exchange between the model and simulator. The most important of these data types is the C++ object type **CModUser**, which is the basic model object type for storing all model-dependent information, such as pins, parameters, outputs, equations, and so on.

Define a Model Name

Ansys recommends that you define a model name. For example:

```
#define   STRG_MODELNAME   "RC3Pin"
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

That eases the later change of the name without the risk of skipping some occurrences. Since it is possible to include more than one model in a model source file – and consequently in the same DLL model file – the other model names should also be defined here.

Define Inputs/Outputs Using the Prepare Function

Define the basic input and output ports of the model using the Prepare function. This function includes conservative pins like electrical nodes, as well as non-conservative inputs or outputs. The assignment of the different nodes is done through specialized methods provided by the **CModUser** object. In most cases, you can use the Prepare function for all analysis types (TR, DC, AC).

RC Example

The RC-combination has three electrical pins and two parameters (resistance R and capacitance C) and an initial value (the initial voltage across the capacitor). In addition, the electrical charge on the capacitor shall be provided as an output. This is programmed in the following Prepare function:

```
FCTDECL PREP_FCN( CModUser *pMod )
{
 // Parameters (conservative nodes)
 pMod->AddNode__c( "n1" );
 pMod->AddNode__c( "n2" );
 pMod->AddNode__c( "n3" );
 // Parameters (non-conservative nodes)
 pMod->AddNode_nc( STRG_NCNAME_U0, DEFAULT_U0, DIRIN );
 pMod->AddNode_nc( STRG_NCNAME_RESISTANCE, DEFAULT_RESISTANCE, DIRIN
);
 pMod->AddNode_nc( STRG_NCNAME_CAPACITANCE, DEFAULT_CAPACITANCE,
DIRIN );
 // Parameter Units
 pMod->SetUnitNameNode_nc( STRG_NCNAME_U0, "V" );
 pMod->SetUnitNameNode_nc( STRG_NCNAME_RESISTANCE, "Ohm" );
 pMod->SetUnitNameNode_nc( STRG_NCNAME_CAPACITANCE, "F" );
 // Parameter Info
 // English
 pMod->SetInfoNode_nc( STRG_NCNAME_U0, "Initial capacitor voltage" );
 pMod->SetInfoNode_nc( STRG_NCNAME_RESISTANCE, "Resistance" );
 pMod->SetInfoNode_nc( STRG_NCNAME_CAPACITANCE, "Capacitance" );
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 // German
 pMod->SetInfoNode_nc( STRG_NCNAME_U0, "Anfangsspannung am
Kondensator", MAKELANGID( LANG_GERMAN, SUBLANG_GERMAN ) );
 pMod->SetInfoNode_nc( STRG_NCNAME_RESISTANCE, "Widerstand",
MAKELANGID( LANG_GERMAN, SUBLANG_GERMAN ) );
 pMod->SetInfoNode_nc( STRG_NCNAME_CAPACITANCE, "Kapazitaet",
MAKELANGID( LANG_GERMAN, SUBLANG_GERMAN ) );
 // Outputs
 pMod->AddNode_nc( STRG_NCNAME_CHARGE, DEFAULT_CHARGE );
 // Output Units
 pMod->SetUnitNameNode_nc( STRG_NCNAME_CHARGE, "C ");
 // Output Info
 pMod->SetInfoNode_nc( STRG_NCNAME_CHARGE, "Capacitor charge" );
 pMod->SetInfoNode_nc( STRG_NCNAME_CHARGE, "Ladung auf Kondensator",
MAKELANGID( LANG_GERMAN, SUBLANG_GERMAN ) );
 return 1L;
}
```

The appearance of the definitions in the Prepare function is shown in the property dialog box of the corresponding model in the schematic.

Define the PISVC Functions

You must define the PISVC functions for the several simulation modes (TR, AC, and DC) of the model. If a simulation mode is not implemented, the model cannot be used for the corresponding analysis.

Every **CModUser** object must provide four model-function pointers for every simulation mode (TR, AC, and DC) for which it is designed, and also one Prepare-function pointer. These model functions are stored in the object using specialized methods.

The five model functions and their assignment are listed in the following table. These five model functions are later called PISVC functions (**Prepare**, **Initialize**, **Simulate**, **Validate**, **Close**).

Function name	**Task**	**CModUser-method**
Prepare function	Assign number, names, and info-lines of external conservative and non-conservative nodes, inputs, outputs, and model internal states. This function is called by the compiler (**comp.exe**) for model creation or interface scanning.	SetUMODPrepFct()

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Function name	Task	CModUser-method
Initialize function	Set symbolic and static Jacobi matrix entries and flags, initialize internal variables and states, and allocate dynamic memory if needed. This function is called by the Simulator (**SimplorerSolver.exe**) to initialize the model before the simulation starts.	SetUMODInitFct()
Simulate function	Set static and dynamic Jacobi matrix elements. Calculate and set outputs or inner states. Feedback to simulator if needed. This function is called by the Simulator (**SimplorerSolver.exe**) for every iteration in DC, for every iteration of every time step in TR, and for every frequency-step in AC.	SetUMODSimFct()
Validate function	Perform tasks after simulation time steps. Feedback to simulator if needed. This function is called by the Simulator (**SimplorerSolver.exe**).	SetUMODValidFct()
Close function	Final cleaning up after simulation run. Free possibly allocated memory. This function is called by the Simulator (**SimplorerSolver.exe**)	SetUMODCloseFct()

There is no Twin Builder demanded restriction on the model function names used for these tasks. But we suggest that you store the names in define-variables for later easy change or extension of the file.

> **Note:**
>
> Model functions assigned to the various simulator modes are not required to be different.

As shown in the source code example of **RegisterUMODELFct**, every DLL-included object must provide its own model functions. The proper allocation is done through the model's name by using the **CModUser** method **GetDefName**.

> **Note:**
>
> Different models can share the same model functions.

Even though there are no Twin Builder restrictions on the names of the model functions, the function definitions must adhere strictly to the following data types.

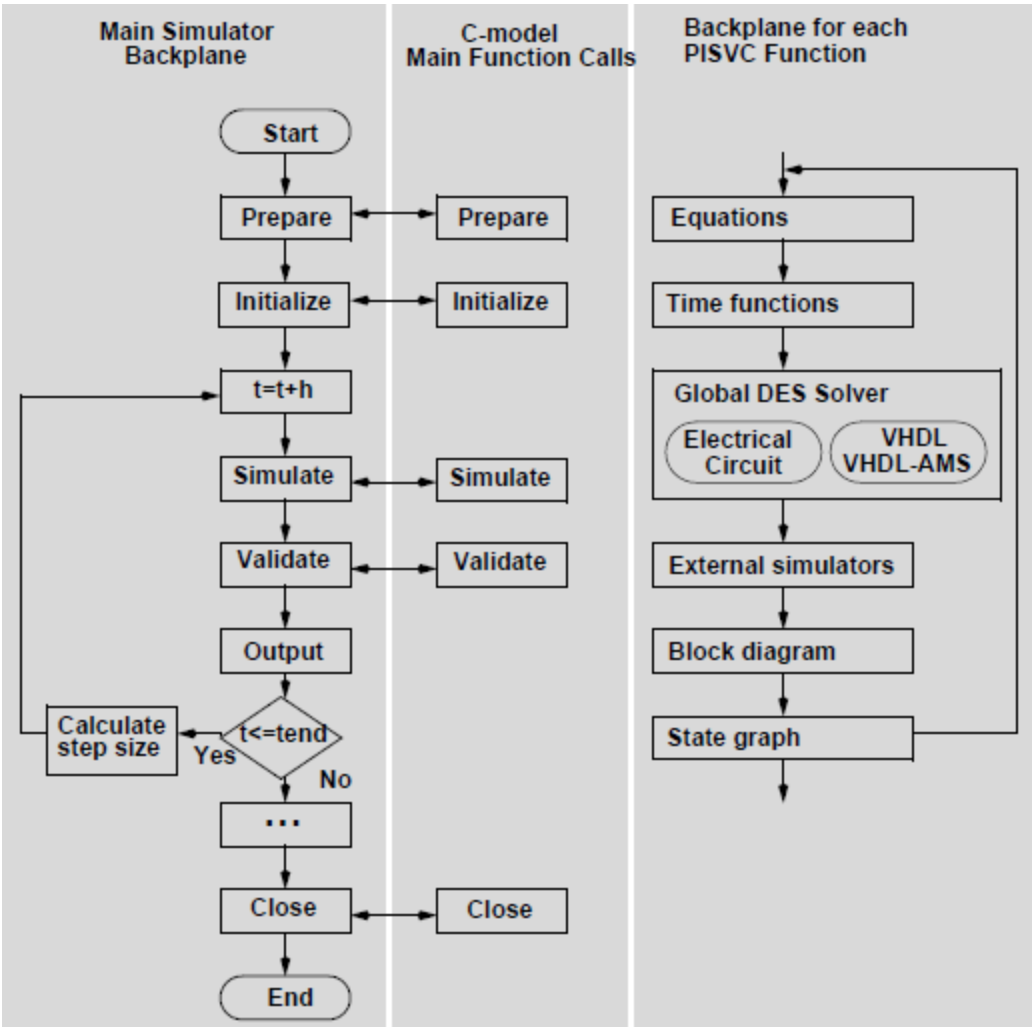
```
FCTDECL modelfct( CModUser *pMod )
{
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
...
return TRUE; // or FALSE...
}
```

The model functions must be of type FCTDECL, which means that they can return the predefined values TRUE or FALSE to signal a successful or erroneous execution to the simulator. The model functions use as the only argument a pointer to a **CModUser** object. Such an object facilitates extensive access of the model's functions to the simulator, its equation system, and model I/O.

The sequence of the PISVC functions is shown below in connection with the Twin Builder simulator.



Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Define Model Kind and Number Using `RegisterUserModel`

The **RegisterUserModel** function is called by the Twin Builder simulator to learn about the number and kind of models wrapped in the DLL model file. Therefore, every model present in the model source file requires a separate entry in the switch loop. The example shows a register function for two models. A predefined model registration function **RegisterUserModel** according to the following scheme must be included:

```
FCTDECL RegisterUserModel( long lIndex, CUModDecl *pUModDecl )
{
switch( lIndex )
 {
 case 0: pUModDecl->SetDefName( STRG_MODELNAME1 );
 pUModDecl->SetModelType( UMODEL );
 pUModDecl->SetSimPref( ECM );
 break;
 case 1: pUModDecl->SetDefName( STRG_MODELNAME2 );
 pUModDecl->SetModelType( UMODEL );
 pUModDecl->SetSimPref( BDM );
 break;
 default:
 break;
 }
 return TRUE;
}
```

Twin Builder needs a name, model type, and preferred subsimulator for each model. This information is provided by the **CUModDecl**'s methods **SetDefName**, **SetModelType**, and **SetSimPref**. Since Twin Builder distinguishes different models in a DLL model file by their names, you must use a unique name for each model. See ***CModUser Object Methods***. Twin Builder distinguishes two types of models: UMODEL and UDCHAR. UMODEL models whole dynamic systems, while UDCHAR is used in programming characteristics for the description of non-linear I/O-relationships and may be used in combination with a number of Twin Builder's basic components and C-Models of UMODEL type.

See ***CUmodDecl Object*** and ***C-Models used as Characteristics***.

The simulator preference method **SetSimPref** controls the accessibility to Twin Builder's internal equation system. Full access to the equation system is enabled using the ECM (Electric Circuit Module) simulator. You can then introduce arbitrary nonlinear ordinary differential equations to Twin Builder and an extensive access to its step size algorithm.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Select the BDM (Block Diagram Module) simulator as the preferred simulator to allow the inclusion of algebraic input-output relations only. No conservative nodes are possible with this selection. BDM models are provided with a node for explicit specification of the block's sample time. See also ***C-Models with Specified Sample Time***. The TFM (Time Function Module) is used for modeling time-dependent signals like sine waves.

Define Model-dependent Functions Using RegisterUMODELFct

The last essential function to be implemented in the model source code is **RegisterUMODELFct**. This function registers the model-dependent functions of all included models. A syntax example is shown below:

```
FCTDECL RegisterUMODELFct( CModUser *pMod )
{
 if( !strcmp( pMod->GetDefName(), STRG_MODELNAME1 ) )
 {
 pMod->SetUMODPrepFct( PREP_FCN1 ); // ModelInterface
 switch( pMod->GetAnalysisType() )
 {
 case TR_:
 pMod->SetUMODInitFct( INIT_FCN1_T ); // Initialize Funct. TR
 pMod->SetUMODSimFct( SIMU_FCN1_T ); // Simulate Funct. TR
 pMod->SetUMODValidFct( VALI_FCN1_T ); // Validate Funct. TR
 pMod->SetUMODCloseFct( CLOSE_FCN1_T ); // Close Function TR
 pMod->AddAvailableAnalysisType(TR_);
 break;
 case DC_:
 pMod->SetUMODInitFct( INIT_FCN1_D ); // Initialize Funct. DC
 pMod->SetUMODSimFct( SIMU_FCN1_D ); // Simulate Funct. DC
 pMod->SetUMODValidFct( VALI_FCN1_D ); // Validate Funct. DC
 pMod->SetUMODCloseFct( CLOSE_FCN1_D ); // Close Function DC
 pMod->AddAvailableAnalysisType(DC_);
 break;
 case AC_:
 pMod->SetUMODInitFct( INIT_FCN1_A ); // Initialize Funct. AC
 pMod->SetUMODSimFct( SIMU_FCN1_A ); // Simulate Funct. AC
 pMod->SetUMODValidFct( VALI_FCN1_A ); // Validate Funct. AC
 pMod->SetUMODCloseFct( CLOSE_FCN1_A ); // Close Function AC
 pMod->AddAvailableAnalysisType(AC_);
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 break;
 default:
 break;
 }
 }
 if( !strcmp( pMod->GetDefName(), STRG_MODELNAME2 ) )
 {
 pMod->SetUMODPrepFct( PREP_FCN2 ); // ModelInterface
 switch( pMod->GetAnalysisType() )
 {
 case TR_:
 pMod->SetUMODInitFct( INIT_FCN2_T ); // Initialize Funct. TR
 pMod->SetUMODSimFct( SIMU_FCN2_T ); // Simulate Funct. TR
 ......
 }
 }
 return TRUE;
}
```

The argument of this function is a C++ object of type **CModUser**. This central object stores all model related information needed for its simulation. See also ***CModUser Object Methods***. As shown in the example, the main task of the required function **RegisterUMODELFct** is to store model-specific function pointers in the different **CModUser** objects.

Implementing for Transient Simulations (TR)

The topics in this section describe the following PISVC functions for the TR simulation mode.

TR Initialize Function

The primary task of the Initialize function is to mark all non-vanishing entries in the Jacobian matrix to make it ready for filling with static and dynamic entries. Initial conditions must then be set and static entries filled. In the case of dynamic models, these functions differ for TR-, DC-, and AC-modes because of the existence of temporal derivatives.

Make symbolic entries by using the `AddSymbolicGSEntry` method of the `CModUser` object.

```
pMod->AddSymbolicGSEntry( 1, 3 )
// Informs the simulator about a non-vanishing entry at (1,3)
```

In the case of a filled right side vector and additive terms in specific entries, it is necessary to separate these entries in several terms.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Example (E4)

Let $F_1=3\cdot x_1+0.5\cdot x'_1$. Then $\delta F_1/\delta x_1=3+0.5\cdot d/dt$ results. Therefore, the corresponding Jacobian entry (i,j) consists of two separate entries and results in the following program code:

```
pMod->AddSymbolicGSEntry( i, j, 0 );
pMod->AddSymbolicGSEntry( i, j, 1, D_DT_OPERAT );
```

The third parameter of the `AddSymbolicGSEntry` method is the index of the subentry; the optional fourth parameter is used as a flag to signalize a temporal derivative to the simulator. See ***CUmodDecl Object***.

In case of an error, the initialize function returns the FALSE value to inform the simulator about this problem. Apart from this the model, you can be informed about details of the problem by using the `Report2Sim` callback function. See ***Callback Functions***.

RC Example

According to equation (E1), the Jacobian matrix for transient analysis consists of nine non-vanishing entries. Only two of them are constant. The other seven are potentially varying during a simulation run. Apart from this, the following example shows the use of the `Report2Sim()` function.

```
FCTDECL Init_RC_TR( CModUser *pMod )
{
 // Get values from model interface
double cap = pMod->GetValNode_nc("C");
double uc0 = pMod->GetValNode_nc("UC0");
double q;
char message[100];
strcpy( message, pMod->GetUseName() );
strcat( message, ": " );
// Check input parameters
 if (cap<=0)
{
strcat( message,
"Capacitance must be positive! Please check model parameters" );
Report2Sim( pMod, UMOD_ERR, message);
return FALSE;
}
 if (cap>1000)
{
strcat( message, "Capacitance is unusual high! Are you sure? " );
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 return !Report2Sim( pMod, UMOD_WRN, message );
}
 // Set symbolic Jacobian entries according to (E1)
pMod->SetSymbolicGSEntry( 0, 0 );
 pMod->SetSymbolicGSEntry( 0, 1 );
 pMod->SetSymbolicGSEntry( 1, 0 );
pMod->SetSymbolicGSEntry( 1, 1 );
 pMod->SetSymbolicGSEntry( 1, 3, 0, D_DT_OPERAT );
 pMod->SetSymbolicGSEntry( 2, 3, 0, D_DT_OPERAT );
 pMod->SetSymbolicGSEntry( 3, 1 );
 pMod->SetSymbolicGSEntry( 3, 2 );
pMod->SetSymbolicGSEntry( 3, 3 );
 // Set constant Jacobian entries
pMod->SetRealGSEntry( 3, 1, 1.0 );
pMod->SetRealGSEntry( 3, 2, -1.0 );
// Set initial condition electric charge on capacitor
q = cap * uc0;
pMod->SetSVVal( 3, q ); // Entry #3 in solution vector
return TRUE;
}
```

TR Simulate Function

After setting up the Jacobian matrix in the Initialize function, the whole mathematical model description must be provided for the simulator. This is done through the Simulate function. In almost every case three different simulate functions for the different simulator modes (TR, DC, AC) must be programmed.

The frequently used temporal derivative operator **d/dt** is replaced by the specialized callback function `ISIM_ECM_D_DT`. See ***Callback Functions***.

Example

Let $F_1=2\cdot(v1-v2)+3\cdot x'_1$. Then $\delta F_1/\delta x_1=3\cdot d/dt$ which results in the following program code:

```
pMod->SetRealGSEntry( 2, 2, 3*ISIM_ECM_D_DT( pMod ) );
```

If the right side vector is filled (which is only possible in the case of linear equations) and the partial derivatives are composed of additive terms with and without temporal derivatives, several subentries at the corresponding Jacobian entry must be filled.

Example

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Let $F_1=3 \cdot x_1+0.5 \cdot x'_1$. Then $\delta F_1/\delta x_1=3+0.5 \cdot d/dt$ results. If the Prepare function sets the symbolic Jacobian entries corresponding to example (E4) results in the following program code.

```
pMod->SetRealGSEntry( i, j, 0, 3.0 );
pMod->SetRealGSEntry( i, j, 1, 0.5*ISIM_ECM_D_DT( pMod ) );
```

Apart from setting the Jacobian entries, the Simulate function also sets output quantities to the model interface using the SetValNode_nc() method.

RC Example

The transient equations of the RC combination contain several dynamic terms. These must be set to the Jacobian matrix in the simulate model function:

```
FCTDECL Simu_RC_TR( CModUser *pMod )
{
 // Getting parameters from the model interface
 double cap = pMod->GetValNode_nc("C");
 double res = pMod->GetValNode_nc("R");
 // Getting Values from the solution vector
 double q = pMod->GetSVVal( 3 ); //Charge on capacitor
 // Example for parameter check
 if (res <= 0.0)
 {
 char message[100];
 strcpy( message, pMod->GetUseName() );
 strcat( message, ": R must be positive" );
 Report2Sim( pMod, UMOD_ERR, message );
 return FALSE;
 }
 // Setting variable Jacobian matrix entries
 pMod->SetRealGSEntry( 0, 0, 1.0/res );
 pMod->SetRealGSEntry( 0, 1, -1.0/res );
 pMod->SetRealGSEntry( 1, 0, -1.0/res );
 pMod->SetRealGSEntry( 1, 1, 1.0/res );
 pMod->SetRealGSEntry( 1, 3, ISIM_ECM_D_DT( pMod ) );
 pMod->SetRealGSEntry( 2, 3, -ISIM_ECM_D_DT( pMod ) );
 pMod->SetRealGSEntry( 3, 3, -1.0/cap );
 // Because of the linearity of the equations the right side
 // vector is filled automatically!
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 // Sending output values to model interface
 pMod->SetValNode_nc( "Q", q );
 return TRUE;
}
```

Accessing Characteristics from C-Models in TR Analysis

C-Models for transient analysis can access characteristics in their Simulate function similar to internal Twin Builder models. Therefore, the model must provide a non-conservative node for the connection to a characteristic block. The type of the node shall be set to **CHAR_** to signalize its behavior to the schematic.

```
FCTDECL Prepare( CModUser *pMod )
{
 pMod->AddNode_nc( "CH", 0.0, DIRIN );
 pMod->SetDataTypeNode_nc( "CH", CHAR_ );
 ...
}
```

In a Twin Builder schematic, this pin must be connected to a characteristic component (>>Basiscs>Tools>Characteristics) or a C-Model characteristic. The characteristic is accessed in the Simulate function using the GetValNode_nc method:

```
FCTDECL Simulate( CModUser *pMod )
{
 double x, y, dy_dx;
 // Let for example
 x = 5;
 y = pMod->GetValNode_nc( "CH", x, &dy_dx );
 // Retrieves y = y(x=5) and dy/dx at x=5
 ...
}
```

TR Validate Function

After every successful time step, Twin Builder calls the **Validate** function for all models in a system. In this section, the model can perform tasks unique in every time step. For example, it is possible to store data in an internal state for further use in the next time step.

TR Close Function

The **Close** function is called after finishing a transient analysis. It is the proper place for some final cleanup procedures such as:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Freeing dynamically allocated memory.
- Storing documentation information on the hard disk.
- Closing all opened files.

Using Internal States

Use the Twin Builder C Interface to define an arbitrary number of internal states for each model. Using the **AddNode_State** method of the **CModUser** object in the **Prepare** function adds internal states to the component. Since these states are identified by their name, the name of the state must be unique in every model.

States are mainly used for storing instance-specific data which can be used in subsequent time steps for further processing. The data does not interfere with the corresponding data of other model instances.

Another very important application of internal states is their possible link to non-conservative output nodes. To associate an internal state to a non-conservative node, you must create a state with an identical name. Then, every setting of the internal state directly affects the output node of the component.

This method of setting output nodes is favorable because internal states are restored by Twin Builder in the case of back-step processes. Such processes can be initiated by almost every part of the Twin Builder simulator in cases where the equations have not finally converged. In such cases, the entire simulation step is disregarded and a new attempt with a smaller step size will be performed.

A direct setting of a linked output node will break the link to the internal state and initiates the independent use of the two nodes.

Example

Initiating a link between an output node and an internal state in the **Prepare** function and following use of the link in the simulate function.

```
FCTDECL Prepare( CModUser *pMod )
{
 pMod->AddNode_nc( "velocity", 0.0 );
 pMod->AddNode_State("velocity", 0.0 );
 // Initiates a link to equal named output
 ...
}
FCTDECL Simulate( CModUser *pMod )
{
 ...
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 pMod->SetValNode_State( "velocity", v ); //Set State AND Output!
 //pMod->SetValNode_nc("velocity", v ); would break the link
 ...
}
```

Implementing for Operating Point Analysis (DC)

The topics in this section describe the following PISVC functions for the DC simulation mode.

DC Initialize Function

The DC Initialize function has similar purposes like the corresponding model function for TR analysis. Moreover, for static models (components without temporal derivatives), they are almost identical. Then a corresponding implementation (similar to the DC Prepare function) is applicable.

RC Example

According to equation (E2) the DC Jacobian is very similar to the TR case with the exception of two temporal derivatives which vanish in the DC case. This results in the following source code fragment:

```
FCTDECL Init_RC_DC( CModUser *pMod )
{
 // Get values from model interface
 double cap = pMod->GetValNode_nc("C");
 double uc0 = pMod->GetValNode_nc("UC0");
 double q;
 // Set symbolic Jacobian entries according to (E1)
 pMod->SetSymbolicGSEntry( 0, 0 );
 pMod->SetSymbolicGSEntry( 0, 1 );
 pMod->SetSymbolicGSEntry( 1, 0 );
 pMod->SetSymbolicGSEntry( 1, 1 );
 pMod->SetSymbolicGSEntry( 3, 1 );
 pMod->SetSymbolicGSEntry( 3, 2 );
 pMod->SetSymbolicGSEntry( 3, 3 );
 // Set constant Jacobian entries
 pMod->SetRealGSEntry( 3, 1, 1.0 );
 pMod->SetRealGSEntry( 3, 2, -1.0 );
 // Set initial condition electric charge on capacitor
 q = cap * uc0;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 pMod->SetSVVal( 3, q ); // Entry #3 in solution vector
 return TRUE;
}
```

DC Simulate Function

The DC Simulate function provides the mathematical DC description of the model's static behavior.

The equations can be derived from the transient equations by dropping all dynamic parts. That means that all Jacobian entries containing temporal derivatives vanish for DC analysis. All explicit time dependencies must be extrapolated to infinite time.

RC Example

The DC equations of the RC combination are found in equation (E2). They are similar to the equations of the TR version:

```
FCTDECL Simu_RC_DC( CModUser *pMod )
{
 // Getting parameters from the model interface
 double cap = pMod->GetValNode_nc("C");
 double res = pMod->GetValNode_nc("R");
 // Getting Values from the solution vector
 double q = pMod->GetSVVal( 3 ); //Charge on capacitor
 // Setting variable Jacobian matrix entries
 pMod->SetRealGSEntry( 0, 0, 1.0/res );
 pMod->SetRealGSEntry( 0, 1, -1.0/res );
 pMod->SetRealGSEntry( 1, 0, -1.0/res );
 pMod->SetRealGSEntry( 1, 1, 1.0/res );
 pMod->SetRealGSEntry( 3, 3, -1.0/cap );
 // Because of the linearity of the equations the right side
 // vector is filled automatically!
 // Sending output values to model interface
 pMod->SetValNode_nc( "Q", q );
 return TRUE;
}
```

DC Validate Function

In most cases, the DC Validate function is only required to signal a successful DC simulation step to the simulator by returning the TRUE variable. It can be omitted if it is not used.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

DC Close Function

Because the results of DC analyses are used as the operating point for subsequent AC simulations, the DC Close function is very important. The DC Close function must store all information necessary to reconstruct the operating point for a (possible) subsequent AC analysis. Therefore UserData is used. The UserData linked via the `SetUserData` method in DC can be queried in AC analyses by `GetUserDataDC`.

RC Example

For the RC example, three values shall be stored for further use in the AC analysis. These are the capacity and resistance values, and the voltage at the capacitor. For this purpose, an adequate memory structure (sDCOP) must be established.

```
struct sDCOP
{
 double cap, res, Vc;
};
FCTDECL Close_RC_DC( CModUser *pMod )
{
 double C = pMod->GetValNode_nc("C");
 double R = pMod->GetValNode_nc("R");
 double Vc = pMod->GetSVVal(1) - pMod->GetSVVal(2);
 sDCOP *pUserData = (sDCOP*)pMod->AllocateUserDataMemory( sizeof
(sDCOP) );
 pMod->SetUserData( pUserData, sizeof(sDCOP) );
 pUserData->cap = C;
 pUserData->res = R;
 pUserData->Vc = Vc;
 return TRUE;
}
```

Implementing for Spectral Analysis (AC)

The AC implementation differs from the other TR and DC simulation modes because of the use of complex numbers and calculations. Every signal can be described by its magnitude A and phase angle φ: $A \cdot e^{j \cdot \varphi}$. That requires the use of complex numbers in the Jacobian matrix as well as in the solution vector and the right side.

The AC Simulate function according to EQ(3) makes use of complex calculations. The solution vector elements as well as the Jacobian entries are generally complex numbers.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

AC Initialize Function

Due to the complex algebra used in the description of the AC behavior of dynamic systems, the AC Initialize function often differs from the TR or DC version.

RC Example

Comparing the AC Jacobian matrix of equation (E3) to the one of the TR example (E1) shows many similarities, except the temporal derivatives which are replaced by complex numbers $j\omega$. Therefore, the symbolic AC entries have no derivative flag (D_DT_OPERAT). Furthermore, the solution vector of the AC problem consists of complex entries.

```
FCTDECL Init_RC_AC( CModUser *pMod )
{
 // Get values from DC analysis, sDCOP is defined below
 long DCsize;
 sDCOP *pUserData = (sDCOP*)pMod->GetDCUserData( &DCsize );
 double cap = sDCOP->cap;
 double res = sDCOP->res;
 double Vc = sDCOP->Vc;
 // Set symbolic Jacobian entries according to (E1)
 pMod->SetSymbolicGSEntry( 0, 0 );
 pMod->SetSymbolicGSEntry( 0, 1 );
 pMod->SetSymbolicGSEntry( 1, 0 );
 pMod->SetSymbolicGSEntry( 1, 1 );
 pMod->SetSymbolicGSEntry( 1, 3 );
 pMod->SetSymbolicGSEntry( 2, 3 );
 pMod->SetSymbolicGSEntry( 3, 1 );
 pMod->SetSymbolicGSEntry( 3, 2 );
 pMod->SetSymbolicGSEntry( 3, 3 );
 // Set constant Jacobian entries
 pMod->SetRealGSEntry( 0, 0, 1.0/res );
 pMod->SetRealGSEntry( 0, 1, -1.0/res );
 pMod->SetRealGSEntry( 1, 0, -1.0/res );
 pMod->SetRealGSEntry( 1, 1, 1.0/res );
 pMod->SetRealGSEntry( 3, 1, 1.0 );
 pMod->SetRealGSEntry( 3, 2, -1.0 );
 pMod->SetRealGSEntry( 3, 3, -1.0/cap );
 // Set initial condition electric charge on capacitor
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 double q = cap * Vc;
 pMod->SetSVVal( 3, q ); // Entry #3 in solution vector
 return TRUE;
}
```

AC Simulate Function

The AC Simulate function provides the mathematical description of the model's AC behavior. Generally, this contains complex algebra and differentiation. From Laplacian transformation, the temporal derivative operator **d/dt** is replaced by the complex term jω. This factor is represented in the source code by the callback function `CComplex ISIM_ECM_P`. See ***Callback Functions***.

RC Example

From the AC equations of the RC combination example (E3), the following AC Simulate function results:

```
FCTDECL Simu_RC_AC( CModUser *pMod )
{
 long DCsize;
 // Get values from DC analysis, sDCOP is defined below
 // (they are not needed in the Simulate-function of this example)
 sDCOP *pUserData = (sDCOP*)pMod->GetDCUserData( &DCsize );
 double cap = pUserData->cap;
 double res = pUserData->res;
 // Variable Jacobian entries
 pMod->SetCplxGSEntry( 0, 0, ISIM_ECM_P( pMod ) );
 pMod->SetCplxGSEntry( 0, 1, -ISIM_ECM_P( pMod ) );
 // Because of the linearity of the equations the right side
 // vector is filled automatically!
 return TRUE;
}
```

AC Validate function

The AC Validate function is called by the simulator between subsequent frequency steps. It can be omitted if not used.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

AC Close function

The AC Close function is called after a AC simulation. It is the right place for releasing dynamically allocated memory or for other cleanup.

C-Models with Specified Sample Time

The Twin Builder C interface provides the possibility of programming time-discrete models, too. This is particularly useful in the simulation of digital systems (for example, digital controllers). These components consist of a microprocessor system running the controller program code at a uniformly clocked sample rate. When simulating the behavior of such a controller, it is essential to consider its clock speed or sample rate to achieve reasonable results.

For the implementation of these kind of systems, the Twin Builder C interface provides the BDM subsimulator. It is selected in the `RegisterUserModel` function. See ***`CUmodDecl` Object***.

```
FCTDECL RegisterUserModel( long lIndex, CUModDecl *pUModDecl )
{
switch( lIndex )
 {
 case 0: pUModDecl->SetDefName( "PID_Controller" );
 pUModDecl->SetModelType( MODEL );
 pUModDecl->SetSimPref( BDM );
 break;
 default:
 break;
 }
 return TRUE;
}
```

When selecting BDM as the preferred simulator, a non-conservative node named TS is created. Your input for this parameter is taken as sample time of the block. This ensures the sampling of the block at these sample times independently of all other elements on a simulation sheet. All block outputs are held constant until the block is sampled again.

Enter **0** to disable the fixed sample time and force the Twin Builder simulator to calculate the block at every simulator step.

To ensure proper operation in all circumstances, it is necessary to access block outputs via associated internal states only. See ***Using Internal States***.

Direct setting of model outputs shall be prevented. Due to this technique, the simulator can access the output nodes of the component in case of a back step event.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

You can access or modify the sample time of a model using the callback functions `GET_SAMPLETIME` and `SET_SAMPLETIME`. Ansys recommends that you set a default sample time in the prepare function of the model. Otherwise, Twin Builder uses the system sample time. In case of a zero sample time input, the `GET_SAMPLETIME` function returns the current simulator time step width. This facilitates the use of this function in temporal numerical integration procedures.

```
FCTDECL PREP( CModUser *pMod )
{
 …
 SET_SAMPLETIME( pMod, DEFAULT_TS );
 …
}
```

C-Models Used as Characteristics

Apart from the programming of dynamic systems, the Twin Builder C Interface enables the programming of characteristics. These characteristics usually describe nonlinear relationships between an input variable *x* and an output variable *y* in the form y=f(x). You can find several predefined characteristics in the Twin Builder library.

Some Twin Builder internal components have a node for connecting characteristics. This facilitates a quick and easy way of considering variable nonlinear behavior in standard components. Dependent on the Twin Builder component, the input and output signal types are fixed. The following table lists the used signal types.

Twin Builder Model	**SML Model Name**	**Input**	**Output**
Nonlinear resistor	RNL	V	I
Nonlinear conductor	GVNL	V	G
Nonlinear dual capacitor	CVNL	V	C
Nonlinear capacitor	CNL	Q	V
Nonlinear dual inductor	LINL	I	L
Nonlinear inductor	LNL	Ψ	I
Nonlinear voltage controlled voltage source	EVNL	V	V
Nonlinear current controlled voltage source	EINL	I	V
Nonlinear voltage controlled current source	IVNL	V	I
Nonlinear current controlled current source	IINL	I	I
DC machine with non-linear excitation inductance characteristics	DCMENL-CHLE	Ie	Le

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder Model	SML Model Name	Input	Output
DC machine with non-linear magnetization characteristics	DCMENL-CHM	ie	Ψe
Non-linear block	NL	INPUT	VAL
C-Models	–	arbitrary	arbitrary

The programming of a C-Model characteristics must register the model as a **UDCHAR** in the `RegisterUserModel` section:

```
FCTDECL RegisterUserModel( long lIndex, CUModDecl *pUModDecl )
{
 switch( lIndex )
 {
 case 0: pUModDecl->SetDefName( "Charactermodel" );
 pUModDecl->SetModelType( UDCHAR );
 break;
 default:
 break;
 }
}
```

You don't need to set a simulator preference for characteristics.

The registration as a **UDCHAR** model adds a node for the bidirectional data exchange with other Twin Builder components to the model. Further non-conservative nodes can be added in the same way, like in dynamic TR-models in the prepare section of the source code.

The calculation of the output quantity *y* from the independent input value *x* is done in the simulate section. The input variable is accessed by the callback function `CHAR_IN`, the output is sent back to the calling element by using `CHAR_OUT`. Apart from this, the partial derivative δy/δx is set through `CHAR_OUT_DERIVE`.

Example

In the following example a simple quadratic relation between an input and an output variable of a characteristics is programmed.

```
FCTDECL Prepare_Q ( CModUser *pMod )
{
 return TRUE;
}
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
FCTDECL Initialize_Q( CModUser *pMod )
{
 return TRUE;
}
FCTDECL Simulate_Q( CModUser *pMod )
{
 double x = CHAR_IN(pMod);
 double y, dy_dx;
 y = x*x;
 dy_dx = 2*x;
 CHAR_OUT(pMod, y);
 CHAR_OUT_DERIVE(pMod, dy_dx);
 return TRUE;
}
FCTDECL Close_Q( CModUser *pMod )
{
 return TRUE;
}
FCTDECL RegisterUserModel( long lIndex, CUModDecl *pUModDecl )
{
 switch( lIndex )
 {
 case 0: pUModDecl->SetDefName( "Q_character" );
 pUModDecl->SetModelType( UDCHAR );
 break;
 default:
 break;
 }
 return TRUE;
}
FCTDECL RegisterUMODELFct( CModUser *pMod )
{
 if( !strcmp( pMod->GetDefName(), "Q_character" ) )
 {
 pMod->SetUMODPrepFct( Prepare_Q );
 pMod->SetUMODInitFct( Initialize_Q );
 pMod->SetUMODSimFct( Simulate_Q );
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 pMod->SetUMODCloseFct( Close_Q );
 }
 return TRUE;
}
```

Using C-Models in Twin Builder

To use C-Models in a schematic, you must include them either in the project library or in an existing library. If you've created a new C-Model (and corresponding component) and want it available for use in other projects, export the component (and its referenced model) to a new or existing library. Use the **Tools** > **Edit Libraries** menu to arrange the component, symbol, and model within the libraries.

To import a new C-Model from a DLL into the project, use **Tools** > **Project Tools** > **Import Simulation Model**. The handling of the C-Model binaries is defined by the general default setting for the binary location (**Tools** > **Options** > **General Options** > **Twin Builder** > **Import Options**). You can change this location later for any individual C-Model.

If the C-Model is placed from a registered library, and the binary location setting for that C-Model was set to store them together with the model, the binaries are extracted into a temporary directory to be used at runtime. Otherwise, the C-Model DLL must be available in the **bin** subdirectory of either the **PersonalLib** or **UserLib** directories. To see the current location of the C-Model binaries, open the **Edit Component** dialog box for the C-Model. On the **Simulation Model** tab, the location of the binaries is listed either as **Stored with model** or the actual path of the model DLL is listed.

Related Topics

Importing Models from a C-Model DLL into Twin Builder

Exporting models from the project to a model library

Twin Builder Options: C-Model Options

Updating C-Models

After a C-Model has been changed by an external compiler, the information of the C-Model in the project must be updated. Depending on the type of changes and the storage settings, the model, symbol, and/or component of the C-Model may need to be updated.

If the model binary is stored with the model, any time the model DLL has changed the model needs to be updated. Otherwise, if the model DLL is kept on disc, the DLL can be overwritten by the external compiler. In order to update the model binary in the project, use **Tools** > **Import Simulation Model** and select the updated DLL.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If the interface of the C-Model changed, both the C-Model definition and the symbol must be updated. Use the same procedure for updating the component or symbol as for updating the model.

C-Models in Schematics

Components for C-Models appear in the **Project Manager** pane under **Definitions** > **Components**. You can place them on a schematic sheet and used in a simulation model like any other Twin Builder component. The **Properties** window of a C-Model lists the model's parameters and nodes, as defined in the DLL model file. Like any other Twin Builder model, you can design a customized component dialog box with the Component Editor.

C-Models in SML Description

In the SML description of simulation models, user-defined C-Models are defined with a UMODEL model instance. The statement contains the model name, the instance name, the list of parameters, and the name of the DLL model file containing the C-Model definition. See ***Twin Builder Modeling Language***.

UMODEL	Keyword for internal simulator models.
Type name:	Name of the model in the model library.
Instance name:	User-defined name for using the model.
Conservative nodes:	List of conservative nodes (if existing) separated by a comma.
Non-conservative nodes:	List of non-conservative nodes (if existing) separated by a comma.
Attributes:	The SRC attribute defines the DLL model file.
UMODEL VOL_CONST VOL_CONST1 H1:=N0002 (VOL:=100u ,B:=700meg) SRC: DLL(File:="Hydraulic.dll") ;	

C/C++ Function Reference

This section gives an overview about Twin Builder-specific C/C++ functions, as well as callback functions and data types.

Data Types

The data exchange of user-defined C-Models with Twin Builder via the `CModUser` object requires the use of some predefined data types. These data types are listed below:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Data Type	Usage	Example
BOOL	Logical values.	TRUE, FALSE

CComplex	Storage of complex numbers.	CComplex a(2.0,3.0); //2.0+3.0i
		CComplex b(-1.5,-4); //-1.5-4.0i
		CComplex c; // 0.0+0.0i
		c = a * b; // =9.0-12.5i

eAnlsType	Simulator analysis type.	TR_, DC_, AC_
eDataType	Type of non-conservative nodes.	INTEGER_, REAL_, ENUMERATION_, PHYSICAL_, ARRAY_, RECORD_, FILE_, STRING_, BOOL_, COMPLEX_, CHAR_
eNature	Domain name of conservative nodes.	ELECTRICAL_, MAGNETIC_, THERMAL_, TRANSLATIONAL _, TRANSLATIONAL_V, ROTATIONAL_, ROTATIONAL_V, FLUIDIC_
eSolverType	Solving algorithm of the electrical network simulator.	EULER_, TRAPEZOIDAL_
eUModType	Model type.	UMODEL, UDCHAR
FCTDECL	Model function declaration.	FCTDECL simu(CModUser *pMod) FCTDECL init_AC(CModUser *pMod)
LPCTSTR	Pointer to a constant character string.	LPCTSTR name = "Resistor"
SIMFCT_ UMODEL	Pointer to a model function.	
Flag	Simulator type.	ECM, BDM, TFM
Flag	Filling type for RS entries.	D_DT_OPERAT, RS_DONTFILL

CUModDecl Object

The Twin Builder C Interface provides a specialized C++ class to perform basic link operations of Twin Builder with C-Models. This class is named `CUModDecl`. It and its methods is only used in the RegisterUserModel section of the model source code. There are three public methods to be used by the programmer:

SetDefName

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Assigns a name to the model.	
`void SetDefName( char *DefName )`	
DefName	Character string containing the model name. In the `RegisterUserModel` section of the model's source code, you must assign a name to the model. This name must be unique in the DLL model file. The model is distinguished from the other models in the DLL model file by Twin Builder using this name.

```
FCTDECL RegisterUserModel( long lIndex, CUModDecl *pUModDecl )
{
switch( lIndex )
 {
 case 0: pUModDecl->SetDefName( "IGBT_TA1" );
 pUModDecl->SetModelType( UMODEL );
 pUModDecl->SetSimPref( ECM );
 break;
 default:
 break;
 }
 return TRUE;
}
```

SetModelType

Assigns a model type to the component.	
`void SetModelType( eUModType Type )`	
Type	Model type. This method is used to inform Twin Builder about the model type. Twin Builder distinguishes between UMODEL and UDCHAR type C-Models. The first are components with an arbitrary number of conservative and non-conservative pins and internal equations. The latter are used to represent characteristics relating an output to an input value. These kinds of models are used as a parameter input for Twin Builder internal components or other C-Models.
`FCTDECL RegisterUserModel( long lIndex, CUModDecl *pUModDecl )`	

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
{
switch( lIndex )
 {
 case 0: pUModDecl->SetDefName( "XPO_LIM");
 pUModDecl->SetModelType( UDCHAR );
 break;
 default:
 break;
 }
 return TRUE;
}
```

SetSimPref

Assigns a simulator to the model.	
`SetSimPref( ULONG Sim)`	
Sim	Simulator: ECM, BDM, TFM. With this method, one of the Twin Builder internal simulators is assigned to a component. There are three predefined simulators: • **ECM** – The electrical circuit simulator (for conservative systems). Full access to simulator equation system. • **BDM** – Block diagram simulator (for non-conservative systems). No access to simulator equation system. • **TFM** – Time Function Module of the simulator. No access to equation system.

```
FCTDECL RegisterUserModel( long lIndex, CUModDecl *pUModDecl )
{
switch( lIndex )
 {
 case 0: pUModDecl->SetDefName( "IGBT_TA1" );
 pUModDecl->SetModelType( UMODEL );
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 pUModDecl->SetSimPref( ECM );
 break;
 case 1: pUModDecl->SetDefName( "EXPO_LIM" );
 pUModDecl->SetModelType( UDCHAR );
 break;
 default:
 break;
 }
 return TRUE;
}
```

CModUser Object Methods

The `CModUser` object is a C++ class provided by the Twin Builder C Interface to enable communication between the C-Models and the simulator. There is an instance of the `CModUser` object associated with every instance of a C-Model. This object carries all necessary information about the component instance. Number and kind of component nodes are stored, as well as parameters, states, and internal equations.

All model functions use `CModUser` objects as argument. A well-defined set of object methods let the programmer manipulate and change the model's properties. These methods are described in the following sections. See ***Technical Notes*** for information on time trigger method extensions.

CModUser object methods grouped by application area

The `CModUser` object methods can be grouped after their application area. For detailed information about the different methods, see ***Alphabetical List of CModUser object methods***.

DATA I/O	
AddNode_c	Adds a conservative node to a model.
AddNode_nc	Adds a non-conservative node to a model.
AddSubNode_nc	Adds subnodes to an already defined non-conservative node. This feature is used to vectorize nc-nodes.
GetDataTypeNode_nc	Returns the data type of a non-conservative node.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetNode_ncParam	Returns the parameter string of a non-conservative node.
GetValNode_nc	Access the real value of a non-conservative node.
GetValNode_ncCplx	Access the complex value of a non-conservative node.
GetValNode_ncFile	Get the file name through a non-conservative node.
GetValNode_ncStrg	Gets a character string through a non-conservative node.
GetValSubNode_nc	Gets a value from a non-conservative subnode (input vector component).
IsCharConn	Checks if a characteristic block is connected to a specified node.
IsParamSetFlag	Checks if the parameter has been set by the user or the default value is used.
SetDataTypeNode_nc	Sets the data type for a non-conservative node.
SetInfoNode__c	Assigns an information line to a conservative node.
SetInfoNode_nc	Assigns an information line to a non-conservative node.
SetNatureTypeNode__c	Assigns a nature to a conservative node.
SetUnitNameNode_nc	Assigns a unit to a non-conservative nodes.
SetValNode_nc	Assigns a value to a non-conservative node.
SetValNode_ncFile	Assigns a file name to a non-conservative node.
SetValPtrNode_nc	Assign a double pointer to a non-conservative node.
SetValSubNode_nc	Assigns a value to a non-conservative subnode (output vector component).
Internal states	
AddNode_State	Adds an internal state to a component model.
GetValNode_State	Returns the real value of an internal state value.
GetValNode_StateCplx	Returns the value of the complex internal state.
SetValNode_State	Assigns a value to an internal state.
Equation system	
GetCplxSVVal	Returns the value of a complex solution vector element.
GetDSVVal	Returns the real temporal derivative value of a solution vector element.
GetSVVal	Returns the real value of a solution vector element.
SetCplxGSEntry	Sets the complex value of an entry in the Jacobian matrix.
SetCplxRSEntry	Sets the complex value of an entry in the right side entries vector.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SetRealGSEntry	Sets the real value of an entry in the Jacobian matrix elements.
SetRealRSEntry	Sets the real value of an entry in the right side entries vector.
SetSVVal	Assigns a value to the solution vector.
SetSymbolicGSEntry	Marks a non-vanishing Jacobian matrix element.
Model identification	
GetDefName	Returns the definition name of the model.
GetHierName	Returns the hierarchical name of the model.
GetUseName	Returns the model instance name.
Miscellaneous	
AllocateUserDataMemory	Allocates memory for the storage of instance specific data.
GetAnalysisType	Returns the current analysis type.
GetDCUserData	Retrieves a DC operating point for use in AC analysis.
GetUserData	Access of instance specific user defined data block.
IsTransientOP	Checks if the simulator calculates the transient operating point. This is done before a transient simulation.
SetAsThreadSafe	Sets this model eligible to be processed by the multithreaded solver for parallel model evaluation.
SetUModCloseFct	Assigns a close function to the model.
SetUModInitFct	Assigns an initialize function to the model.
SetUModPrepFct	Assigns a prepare function to the model.
SetUModSimFct	Assigns a simulate function to the model.
SetUModValidFct	Assigns a validate function to the model.
SetUserData	Storage of arbitrary instance-specific user data memory.

Alphabetical List of CModUser object methods

The section provides a complete alphabetical list of `CModUser` object methods, including parameter explanations.

AddAvailableAnalysisType

Adds an analysis type to a component. This method informs the Twin Builder simulator about the analysis types supported by the component. It is generally used in conjunction with the definition of model functions during model registration
`BOOL AddAvailableAnalysisType( eAnlsType Type, char *pszFreeName =`

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

NULL);	
Type	Analysis type. Predefined values are TR_, AC_ and DC_.
pszFreeName	If FREE_ANLS is declared, a string that defines the analysis type per string.
CModUser *pMod; pMod->AddAvailableAnalysisType(TR_);	
Sections	RegisterUMODELFct
See also GetAnalysisType, SetUMODPrepFct, SetUMODInitFct, SetUMODSimFct, SetUMODValidFct, SetUMODCloseFct, AddNode_nc.	

AddNode_c

Adds conservative nodes to component. Components in conservative systems must have conservative nodes to allow connection with other systems of the same domain. Assigned with such a node is in every domain a flow and a potential variable. For example, in electrical systems (NatureName=NATURE_NAME_ELECTRICAL_) these are electrical current or voltage; in the case of translational mechanical systems (NatureName=NATURE_NAME_TRANSLATIONAL_), these are force or velocity.	
void AddNode__c(LPCTSTR Name, enum eNature Nat=ELECTRICAL_) void AddNode__c(LPCTSTR Name, LPCTSTR NatureName)	
Name	Unique name of the node within the component.
Nat	Nature of the node: ELECTRICAL_ is default. See Definition eNature.
NatureName	String representation of nature name. Predefined values are: NATURE_NAME_ELECTRICAL_, NATURE_NAME_MAGNETIC_, NATURE_NAME_THERMAL_, NATURE_NAME_TRANSLATIONAL_, NATURE_NAME_TRANSLATIONAL_V_, NATURE_NAME_ROTATIONAL_, NATURE_NAME_ROTATIONAL_V_, NATURE_NAME_FLUIDIC_

CModUser *pMod;
 	
pMod->AddNode__c("np"); //Electrical is default
 	
pMod->AddNode__c("hipres", FLUIDIC_);
 	
pMod->AddNode__c("flange1", NATURE_NAME_ROTATIONAL_);	

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Sections	Prepare
See also AddNode_nc, AddNode_State, SetInfoNode_c, SetNatureTypeNode_c.	

AddNode_nc

Adds non-conservative nodes to component. Non-conservative nodes pass values in and out of a model. They can represent parameters, initial conditions, or even time-dependent signals.	
int AddNode_nc(LPCTSTR Name, double Val=0, long Dir=DIROUT, BOOL Vec=FALSE) int AddNode_nc(LPCTSTR Name, LPCTSTR Use, long Dir=DIROUT, BOOL Vec=FALSE)	
Name	Unique name of the node.
Val	Default value.
Use	Default string value for the **Node_nc**, for example **data.mdx**.
Dir	Signal direction of port. Output direction is default. (DIRIN, DIROUT, DIRINOUT)
Vec	Specifies whether the **Node_nc** is a vector, and can have subnodes.
Return value	Index of the non-conservative node within the model.

CModUser *pMod;pMod->AddNode_nc("V0", 12.9, DIRIN);
pMod->AddNode_nc("filename", "data.out", DIRIN);
pMod->AddNode_nc("Power", 0.0); // DIROUT is default	

Sections	Prepare
See also AddNode_State, GetValNode_nc, SetDataTypeNode_nc, SetValNode_nc, SetInfoNode_nc.	

AddNode_State

Adds subnodes to an appropriate non-conservative node. You can vectorize input and output nodes of C-Models by extending an existing non-conservative node with this function. You must add the node using the “Vec”-flag in its AddNode_nc call.	
int AddNode_State(LPCTSTR Name, double dVal=0)	
Name	Unique name of the internal state within the model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

dVal	Default value.
Return value	Index of the internal state within the model.

CModUser *pMod;
pMod->AddNode_State("Heatshieldtemperature");
pMod->AddNode_State("Warpfactor", 7.3);
pMod->AddNode_State("On_Off", 0.0);	

Sections	Prepare
See also AddNode_c, AddNode_nc, GetValNode_State, GetValNode_StateCplx, SetValNode_State, SetInfoNode_nc.	

AddSubNode_nc

Adds non-conservative nodes to component. Non-conservative nodes pass values in and out of a model. They can represent parameters, initial conditions, or even time-dependent signals.	
long AddSubNode_nc(LPCTSTR Name, long Index, double Val, long Dir); long AddSubNode_nc(LPCTSTR Name, long Index, LPCTSTR Use, long Dir);	
Name	Name of the nonconservative base node.
Val	Default value.
Use	Default string value for the **Node_nc**, for example **data.mdx**.
Dir	Signal direction of port. Output direction is default. (DIRIN, DIROUT, DIRINOUT)
pMod->AddNode_nc("InputVec", 0, DIRIN, TRUE); for (i=0; i<4; i++) pMod->AddSubNode_nc("InputVec", i, 0, DIRIN);	
Sections	Prepare
See also AddNode_nc, AddNode_State, AddSubNode_nc, GetValNode_nc, GetValSubNode_nc, SetDataTypeNode_nc, SetValNode_nc, SetValSubNode_nc.	

AllocateUserDataMemory

Allocates memory for the storage of instance-specific data. Use this method to allocate a memory block to store instance specific data. You can access the allocated memory block after allocating from every model function in every analysis type.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

It is for example used to store the DC operating point after DC analysis for further use in AC analysis.	
void *AllocateUserDataMemory(long size)	
size	Size of the user data in bytes.
Return value	Pointer to the memory block.

CModUser *pMod;
struct UD { double d1; int cnt; };
UD *pData = (UD*)pMod->AllocateUserDataMemory(sizeof(UD));
pMod->SetUserData(pData, sizeof(UD));
pData->d1 = 1.5;
pData->cnt = 30;	

Sections	Init, Validate, Close
See also GetDCUserData, GetUserData, SetUserData.	

GetAnalysisType

Returns current analysis type. Returns the analysis type to inform the model about currently applied analysis type. Can be used for example when introducing analysis dependent functionality or equation sets.	
eAnlsType GetAnalysisType()	
Return value	Analysis type TR_, DC_ or AC_.

CModUser *pMod;
if (pMod->GetAnalysisType() == TR_)
pMod->SetUMODInitFct(INIT_FCN_TR);	

Sections	RegisterUMODELFct(*CModUser), (or anywhere)
See also AddAvailableAnalysisType, SetUMODPrepFct, SetUMODInitFct, SetUMODSimFct, SetUMODValidFct, SetUMODCloseFct.	

GetCplxSVVal

Returns complex solution vector elements. In general, the solution vector in AC analysis consists of complex elements. The method returns such a complex vector element.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

CComplex GetCplxSVVal(int LineNo)	
LineNo	Number of solution vector element (starting at 0).
Return value	Complex solution vector element.
CModUser *pMod; CComplex V1 = pMod->GetCplxSVVal(0); CComplex V2 = pMod->GetCplxSVVal(1);	
Sections	Simulate (AC), Validate (AC), Close (AC), Init(AC)
See also GetSVVal, SetCplxGSEntry, SetCplxRSEntry, SetSVVal.	

GetDataTypeNode_nc

Returns data type of non-conservative nodes. Returns the data type of non-conservative nodes. Predefined values are INTEGER_, REAL_, ENUMERATION_, PHYSICAL_, ARRAY_, RECORD_, FILE_, STRING_, BOOL_, COMPLEX_, and CHAR_.	
eDataType GetDataTypeNode_nc(LPCTSTR Name) eDataType GetDataTypeNode_nc(int NodeNo)	
Name	Name of non-conservative node.
NodeNo	Number of non-conservative node (starting at 0). Can be determined by the return value of **AddNode_nc**.
Return value	Data type of non-conservative node.

CModUser *pMod;
char sn[100];
double dn;
if (pMod->GetDataTypeNode_nc("Pin1") == STRING_)
strcpy(sn, pMod->GetValNode_ncStrg("Pin1"))
else
double dn = pMod->GetValNode_nc("Pin1");	

Sections	Init, Simulate, Validate, Close
See also GetValNode_nc, SetDataTypeNode_nc, SetValNode_nc.	

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

GetDefName

Returns definition name of the model. Returns the name given in the **RegisterUserModel** section with **SetDefName**.
LPCTSTR GetDefName()
Return value: Definition name of model.
CModUser *pMod; char modelname[100]; strcpy(modelname, pMod->GetDefName());
Sections: Prepare, Init, Simulate, Validate, Close.
See also `GetHierName`, `GetUseName`.

GetDCUserData

Retrieves a DC operating point for use in AC analysis. Use this method to access the operating point data in an AC analysis already stored in a preceding DC analysis. Since the AC version of a model can use its own user data, this function allows a non-interfering access to the separate DC data block. You can use the simultaneously returned size information to double-check the validity of the data.
void *GetDCUserData(long *size)
size: Returned value of the DC user data memory size in bytes.
Return value: Pointer to the DC user data block.

```
struct sDCOP { double cap, res; };
//...
FCTDECL SIMU_AC( pMod )
{
 long DCOPsize;
 sDCOP *pUserData = (sDCOP*)pMod->GetDCUserData( &DCOPsize );
 double cap = pUserData->cap;
 double res = pUserData->res;
 //...
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

}	
Sections	Init (AC), Simulate (AC), Validate (AC), Close (AC).
See also AllocateUserDataMemory, GetUserData, SetUserData.	

GetDSVVal

Returns the temporal derivative of a solution vector element. Returns the temporal derivative of a solution vector element in transient analysis.	
double GetDSVVal(int LineNo)	
LineNo	Number of solution vector element (starting at 0).
Return value	Temporal derivative of a real value of a solution vector element, specified by the LineNo parameter.

CModUser *pMod;
double position = pMod->GetSVVal(3);
double velocity = pMod->GetDSVVal(3);	

Sections	Init, Simulate (TR), Validate (TR), Close (TR).
See also GetSVVal, GetValNode_nc.	

GetHierName

Returns the hierarchical name of the model. When the model is situated within subsheets or macros, this function returns the path to the model, for example **subsheet1**.	
LPCTSTR GetHierName()	
Return value	Hierarchical name without model instance name.
CModUser *pMod; char hname[100]; strcpy(hname, pMod->GetHierName());	
Sections	Prepare, Init, Simulate, Validate, Close.
See also GetDefName, GetUseName.	

GetNode_ncParam

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Returns the parameter string of a non-conservative node. If you set a number, the function returns an empty string; otherwise, it returns the name of the connected value.		
LPCTSTR GetNode_ncParam(LPCTSTR lpszName); LPCTSTR GetNode_ncParam(int iNr);		
lpszName	Name of non-conservative node.	
iNr	Number of non-conservative node (starting at 0) within the model.	
Return value	Parameter string or ""	

CModUser *pMod;
LPCTSTR strParam1, strParam2;
strParam1 = pMod->GetNode_ncParam ("Parameter_1");
strParam2 = pMod->GetNode_ncParam (7);		

Sections	Prepare, Init, Simulate, Validate, Close.	

GetSVVal

Returns a real solution vector element. In every analysis type, Twin Builder provides solution vector elements for further calculation and evaluation. The method is used to access the real value of a solution vector elements.	
double GetSVVal(int LineNo)	
LineNo	Number of solution vector element (starting with 0).
Return value	Real solution vector element.
CModUser *pMod; V1 = pMod->GetSVVal(0); position = pMod->GetSVVal(3);	
Sections	Simulate, Validate, Close, Init.
See also GetDSVVal, SetSVVal, GetValNode_nc, GetValNode_State.	

GetUseName

Returns model instance name. Several instances of one model in a simulation model (sheet) can be distinguished by their

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

user-defined instance name. Use this method to access the actual instance name of the model.
LPCTSTR GetUseName()
Return value: Model instance name.

CModUser *pMod;
char message[100];
strcpy(message, pMod->GetUseName());
strcat(message, " says hello! ");
Report2Sim(pMod, UMOD_INF, message);

Sections: Prepare, Init, Simulate, Validate, Close.
See also GetDefName, GetHierName.

GetUserData

Accesses an instance-specific user-defined data block. For the storage of instance specific model data, you can use the user data structure. For every simulation mode (TR, DC, AC) a separate memory data block can be used and accessed by this method.
void *GetUserData(long *size)
size: Returned value of the memory size of the user data block.
Return value: Pointer to the user data block.

struct UD { double d1; int ii; };
//....
long UDsize;
UD *pUD = (UD*)pMod->GetUserData(&UDsize);
double d1new = pUD->d1;
int cnt = pUD->ii;

Sections: Init, Simulate, Validate, Close.
See also AllocateUserDataMemory, GetDCUserData, SetUserData.

GetValNode_nc

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Accesses a value of a non-conservative node. Use this method to access signal values at non-conservative nodes like initial values or parameters. The use with characteristics is explained below.	
double GetValNode_nc(LPCTSTR Name) double GetValNode_nc(intNo)	
Name	Name of non-conservative node.
No	Number of non-conservative node (starting at 0) within the model.
Return value	Real value of signal at non-conservative node or y=f(x) when connected to a characteristics.

CModUser *pMod;
double v = pMod->GetSVVal(0) - pMod->GetSVVal(1);
double torque1 = pMod->GetValNode_nc("Flange1");
double input3 = pMod->GetValNode_nc(3);	

Sections	Init, Simulate, Validate, Close.
See also GetValNode_ncCplx, GetValNode_ncFile, GetValNode_ncStrg.	

GetValNode_nc (for use with characteristics)

Accesses the value of a non-conservative node connected to characteristic. The method enables the communication with characteristics, used to describe nonlinear relations (for example, electrical current as a function of voltage). The characteristic also returns the partial derivative most likely needed as an Jacobian entry.	
double GetValNode_nc(LPCTSTR Name, double x, double* dy_dx = NULL,) double GetValNode_nc(int No double x, double* dy_dx = NULL,)	
Name	Name of non-conservative node.
No	Number of non-conservative node (starting at 0) within the model.
x	X value to send to a characteristic which calculates y=f(x).
dy_dx	Partial derivative δy/δx returned by a characteristic.
Return value	y=f(x) when connected to a characteristics.
CModUser *pMod; double i, v, di_dv;	

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

i = pMod->GetValNode_nc("CH", v, &di_dv);	
Sections	Init, Simulate, Validate, Close.
See also GetValNode_ncCplx, GetValNode_ncFile, GetValNode_ncStrg.	

GetValNode_ncCplx

Accesses the complex value of a non-conservative node. Especially in the case of AC simulations, the exchange of complex signals between components is frequently used. This method accesses complex signals fed to non-conservative nodes.	
CComplex GetValNode_ncCplx(LPCTSTR Name) CComplex GetValNode_ncCplx(int No)	
Name	Name of non-conservative node.
No	Number of non-conservative node (starting at 0) within the model.
Return value	Complex node value.

CModUser *pMod;
CComplex impedance, shift(1, pi);
impedance = pMod->GetValNodeCplx("Zport");
impedance = impedance * shift;	

Sections	Init (AC), Simulate (AC), Validate (AC), Close (AC).
See also GetValNode_nc, GetValNode_ncFile, GetValNode_ncStrg.	

GetValNode_ncFile

Gets a file name through a non-conservative node. Models with included file input/output can use file names as input parameters. These names are passed to the model via non-conservative nodes. The file name is accessed by using the **GetValNode_ncFile** method.	
LPCTSTR GetValNode_ncFile(LPCTSTR Name) LPCTSTR GetValNode_ncFile(int No)	
Name	Name of non-conservative node.
No	Number of non-conservative node (starting at 0) within the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	model.
Return value	File name.

CModUser *pMod;
char parafil[20], logfile[25];
strcpy(parafil, pMod->GetValNode_ncFile("P_f"));
strcpy(logfile, pMod->GetValNode_ncFile(7));	

Sections	Prepare, Init, Simulate, Validate, Close.
See also GetValNode_nc, GetValNode_ncCplx, GetValNode_ncStrg.	

GetValNode_ncStrg

Gets a character string through a non-conservative node. In addition to numerical values, it might be necessary to pass character strings such as file names, messages, or expressions to components. These strings are passes to the model via non-conservative nodes.	
LPCTSTR GetValNode_ncStrg(LPCTSTR Name) LPCTSTR GetValNode_ncStrg(int No)	
Name	Name of non-conservative node.
No	Number of non-conservative node (starting at 0) within the model.
Return value	Character string.

CModUser *pMod;
char messg[20], formula[250];
strcpy(messg, pMod->GetValNode_ncFile("Signal"));
strcpy(formula, pMod->GetValNode_ncFile(2));
Report2Sim(pMod, UMOD_INF, messg);	

Sections	Prepare, Init, Simulate, Validate, Close.
See also GetValNode_nc, GetValNode_ncCplx, GetValNode_ncFile.	

GetValNode_State

Retrieves real internal state value.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Internal states store instance-specific values or signals. You can access the real values of these internal states with this method. Internal states are preferable compared to global variables whenever applicable.	
double GetValNode_State(LPCTSTR Name) double GetValNode_State(int No)	
Name	Name of state used in **AddNode_State**.
No	Number of state (starting at 0) within the model.
Return value	Real value of internal state.

CModUser *pMod;
double stateofcharge = pMod->GetValNode_State("SOC");
double T = pMod->GetValNode_State(3);	

Sections	Simulate, Validate, Close.
See also `AddNode_State`, `GetValNode_StateCplx`, `SetValNode_State`.	

GetValNode_StateCplx

Retrieves complex internal state value. Internal states store instance-specific values or signals. You can access the complex values of these internal states with this method. Internal states are preferable compared to global variables whenever applicable.	
CComplex GetValNode_StateCplx(LPCTSTR Name) double GetValNode_StateCplx(int No)	
Name	Name of state used in **AddNode_State**.
No	Number of state (starting at 0) within the model.
Return value	Complex value of internal state.
CModUser *pMod; CComplex CSignal = pMod->GetValNode_StateCplx("CS");	
Sections	Simulate, Validate, Close.
See also `AddNode_State`, `GetValNode_State`, `SetValNode_State`.	

GetValSubNode_nc

Retrieves the value of an input vector element.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Components of vectorized non-conservative input nodes must be accessed element-wise using this function.	
double GetValSubNode_nc(LPCTSTR Name, long Index);	
Name	Name of the non-conservative base node.
Index	Index of the vector element, starting at 0.

double x = pMod->GetValSubNode_nc("Space_time_Vector", 0);
double y = pMod->GetValSubNode_nc("Space_time_Vector", 1);
double z = pMod->GetValSubNode_nc("Space_time_Vector", 2);
double t = pMod->GetValSubNode_nc("Space_time_Vector", 3);	

Sections	Simulate, Validate, Close.
See also AddNode_nc, AddSubNode_nc, GetValNode_nc.	

IsCharConn

Checks if a characteristic is connected to a non-conservative node. When using a node for connecting an external characteristics block to the C-Model, it is very important to really connect such a special block to this nc-node. When connecting other blocks to this node, the results may be completely wrong. This function returns TRUE in case of a characteristics is connected to the node and FALSE in all other cases.	
BOOL IsCharConn(LPCTSTR Name);	
Name	Name of the non-conservative node.

if (!pMod->IsCharConn("ChNode")){
Report2Sim(pMod, UMOD_ERR, "There is no Char-Block connected
at node 'ChNode'!");
return FALSE;
}	

Sections	Initialize.
See also GetValNode_nc.	

IsParamSetFlag

Checks that you have changed the default parameter values. Non-conservative model inputs usually have a default value. Use this method to sense the

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

change of these parameters from default values.	
BOOL IsParamSetFlag(LPCTSTR Name)	
Name	Name of non-conservative node.
Return value	Flag indicating parameter change. TRUE, if you have explicitly set this parameter, FALSE if you have not made changes to the parameter value, and the default value is used.

CModUser *pMod;
double pp;
if (pMod->IsParamSetFlag("P1"))
pp = pMod->GetValNode_nc("P1")
else
pp = DEFAULT_P1;	

Sections	Init, Simulate.
See also GetValNode_nc.	

IsTransientOP

Checks if simulator is calculating transient operating point. Before every transient simulation, the simulator performs an operating point analysis. For example, initial values of capacitors and inductors are set, if they are specified. If they are not specified, the simulator calculates reasonable starting values for the transient simulation. For this calculation, most models must have a special equation system. This special system can be selected by monitoring the function return value. It is TRUE in case the simulator is calculating the transient operating point and FALSE all other times.	
BOOL IsTransientOP();	

if (pMod->IsTransientOP()) {
// Use special Transient OP equation set
...
} else {
// Normal TR equation set
...
};	

Sections	Simulate.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SetAsThreadSafe

Sets this model eligible to be processed by the multithreaded solver for parallel model evaluation. If this is not called, the C-Model is assumed to be non-thread-safe; therefore, the model is processed in serial mode. Call is only successful when called from Initialize function. **Note:** Currently, multi-threading is only applicable for ECM (circuit simulator) models.
BOOL SetAsThreadSafe ();

Return value	TRUE if successful, FALSE if not.

CModUser *pMod; pMod->SetAsThreadSafe();

Sections	Initialize

SetCplxGSEntry

Sets a complex entry in the Jacobian matrix. For AC analysis, the coefficients of the equation system to be solved are complex. Use this method to enter these complex values into the Jacobian matrix. It also lets you enter multiple additive values at the entry sites. This might be necessary when filling the right side of the equation system.
BOOL SetCplxGSEntry(long Row, long Col, const CComplex &Val) BOOL SetCplxGSEntry(long Row, long Col, const CComplex &Val, long sub)

Row	Row number of Jacobian entry (starting at 0).
Col	Column number of Jacobian entry (starting at 0).
Val	Complex value to be entered into matrix.
sub	Index of subentry in case of overloaded Jacobian entries.
Return value	TRUE if successful, FALSE if not.

CModUser *pMod;
CComplex z1(1.3,-2.2), z2(2.0,3.3);
CComplex jw = ISIM_ECM_P(pMod);
pMod->SetCplxGSEntry(0, 2, z1);
pMod->SetCplxGSEntry(1, 3, z2 , 0);

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

pMod->SetCplxGSEntry(1, 3, 2.0 * jw, 1);	
Sections	Init (AC), Simulate (AC).
See also SetCplxRSEntry, SetRealGSEntry.	

SetCplxRSEntry

Sets a complex right side entry. The automatic filling of the right side is only possible for linear equations. All nonlinear systems are required to fill the right side of the equations manually. In AC analysis, the right side entries are complex numbers. They are set using the **SetCplxRSEntry** method.	
BOOL SetCplxRSEntry(long No, const CComplex &Val)	
No	Number of right side entry (starting at 0) within the model.
Val	Complex value to be set.
Return value	TRUE if successful, FALSE if not.

CModUser *pMod;
CComplex z(1.2, 3.14)
pMod->SetSymbolicGSEntry(1, 3, 0, RS_DONTFILL);
pMod->SetCplxRSEntry(1, z);	

Sections	Init (AC), Simulate (AC).
See also SetCplxGSEntry, SetRealGSEntry, SetRealRSEntry, SetSymbolicGSEntry.	

SetDataTypeNode_nc

Sets a data type for a non-conservative node. To every non-conservative node, a specific data type can be assigned. This can be done after introducing the model to Twin Builder, or in the model itself using this method.	
BOOL SetDataTypeNode_nc(LPCTSTR Name, eDataType DataType) BOOL SetDataTypeNode_nc(int No, eDataType DataType)	
Name	Name of a non-conservative node as set in **AddNode_nc**.
No	Index of a non-conservative node (starting at 0) within the model.
DataType	Data type to be set. Valid entries are INTEGER_, REAL_, ENUMERATION_, PHYSICAL_, ARRAY_, RECORD_, FILE_,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	STRING_, BOOL_, COMPLEX_ and CHAR_.
Return value	TRUE if successful, FALSE if not.

CModUser *pMod;
pMod->AddNode_nc("Z", 0.0, DIRIN);
pMod->AddNode_nc("name", "Ralf", DIRIN);
pMod->SetDataTypeNode_nc("Z", COMPLEX_);
pMod->SetDataTypeNode_nc(1, STRING_);	

Sections	Prepare.
See also Addnode_nc, GetValNode_nc, SetInfoNode_nc, SetNatureTypeNode_c.	

SetInfoNode__c

Assigns an information line to conservative pins. To every conservative node, an information line can be assigned. This line describes the meaning or function of the node. The text line can be assigned in the model source code. Information lines can be assigned in various languages (English is the default). According to the language selected in Twin Builder, the correct line is displayed.	
void SetInfoNode__c(LPCTSTR Name, LPCTSTR Info, LANGID lLangID = MAKELANGID (LANG_ENGLISH,SUBLANG_ENGLISH_US)) void SetInfoNode__c(int No, LPCTSTR Info, LANGID lLangID = MAKELANGID(LANG_ ENGLISH,SUBLANG_ENGLISH_US))	
Name	Name of the conservative node as set in **AddNode__c**.
No	Index of the conservative node (starting at 0) within the model.
Info	Info line to be displayed in Twin Builder Schematic.
lLangID	Language ID as obtained from the MAKELANGID function (see Visual C++ Manual).

CModUser *pMod;
pMod->SetInfoNode__c("V1", "Voltage");
pMod->SetInfoNode__c("V1", "Spannung", MAKELANGID(LANG_GERMAN, SUBLANG_ GERMAN));	

Sections	Prepare.
See also Addnode_c, SetInfoNode_nc, SetNatureTypeNode_c, SetUnitNameNode.	

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SetInfoNode_nc

Assigns an information line to non-conservative nodes. To every non-conservative node, an info line can be assigned. This line describes the meaning or function of the node. The text line is assigned in the model source code. Info lines are assigned in various languages (English is default). According to the language selected in Twin Builder, the correct line is displayed.	
void SetInfoNode_nc(LPCTSTR Name, LPCTSTR Info, LANGID lLangID = MAKELANGID (LANG_ENGLISH,SUBLANG_ENGLISH_US)) void SetInfoNode_nc(int No, LPCTSTR Info, LANGID lLangID = MAKELANGID(LANG_ ENGLISH,SUBLANG_ENGLISH_US))	
Name	Name of the non-conservative node as set in **AddNode_nc**.
No	Index of the non-conservative node (starting at 0) within the model.
Info	Info line to be displayed in Twin Builder Schematic.
lLangID	Language ID as obtained from the MAKELANGID function (see Visual C++ Manual).

CModUser *pMod;
pMod->SetInfoNode_nc("R", "Resistance");
pMod->SetInfoNode_nc("R", "Widerstand", MAKELANGID(LANG_GERMAN, SUBLANG_ GERMAN));	

Sections	Prepare.
See also Addnode_nc, SetInfoNode_c, SetUnitNameNode.	

SetNatureTypeNode__c

Assigns a nature to a conservative node. Use this method to assign a nature type to a conservative node. Twin Builder can simulate dynamic systems in multiple physical domains. In every domain, there are potential and flow variables assigned to conservative nodes. Such an assignment is called the nature of the node. Twin Builder prevents the unintentional connection of nodes of different domains. Therefore, you must assign a nature to every conservative node. When creating conservative nodes using **AddNode__c**, the electrical nature is default.	
BOOL SetNatureTypeNode__c(LPCTSTR Name, LPCTSTR NatureName)	
Name	Name of the conservative node as assigned in **AddNode__ c**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

NatureName	Name of the Nature. Valid predefined entries are: NATURE_NAME_ELECTRICAL_, NATURE_NAME_MAGNETIC_, NATURE_NAME_THERMAL_, NATURE_NAME_TRANSLATIONAL_, NATURE_NAME_TRANSLATIONAL_V_, NATURE_NAME_ROTATIONAL_, NATURE_NAME_ROTATIONAL_V_, NATURE_NAME_FLUIDIC_
Return value	TRUE if successful, FALSE if not.

```
CModUser *pMod;
pMod->SetNatureTypeNode__c( "V1",
 NATURE_NAME_ELECTRICAL_ );
pMod->SetNatureTypeNode__c( "p2",
 NATURE_NAME_FLUIDIC_ );
```

Sections	Prepare.
See also `AddNode_c`, `SetInfoNode_c`.	

SetRealGSEntry

Sets a Jacobian matrix element.

Use this method to enter real values into the Jacobian matrix. It also lets you enter multiple additive values at the entry sites. This might be necessary when filling the right side of the equation system.

```
BOOL SetRealGSEntry( long Row, long Col, double Val )
BOOL SetRealGSEntry( long Row, long Col, double Val, long sub )
```

Row	Row number of Jacobian entry (starting at 0).
Col	Column number of Jacobian entry (starting at 0).
Val	Real value to be entered into the matrix.
Sub	Index of the subentry for overloaded Jacobian entries.
Return value	TRUE if successful; FALSE if not.

```
CModUser *pMod;
double r1 = 10.0;
double r2 = 2.3;
double dh = ISIM_ECM_D_DT( pMod );
pMod->SetRealGSEntry( 0, 2, r1 );
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

pMod->SetRealGSEntry(1, 3, r2, 0); pMod->SetRealGSEntry(1, 3, 2.0*dh, 1);	
Sections	Init, Simulate.
See also SetCplxGSEntry, SetRealRSEntry.	

SetRealRSEntry

Sets a right side entry at a defined position. The automatic filling of the right side is only possible for linear equations. All nonlinear systems must fill the right side of the equations manually. For real entries, this is done using the **SetRealRSEntry** method.	
BOOL SetRealRSEntry(long No, double Val))	
No	Index of right side entry (starting at 0) within the model.
Val	Real value to be set.
Return value	TRUE if successful, FALSE if not.

CModUser *pMod;
pMod->SetSymbolicGSEntry(1, 3, 0, RS_DONTFILL);
pMod->SetRealRSEntry(1, -2.7);	

Sections	Init, Simulate.
See also SetCplxGSEntry, SetCplxRSEntry, SetRealGSEntry, SetSymbolicGSEntry.	

SetSVVal

Assigns a value to the solution vector. Use this method to assign initial conditions to the elements of the solution vector. It assigns real as well as complex quantities to the solution vector. Complex quantities are only used in AC analysis.	
BOOL SetSVVal(int No, double dVal) BOOL SetSVVal(int No, const CComplex &cVal)	
No	Number of the solution vector element (starting at 0) within the model.
dVal	Real value to be assigned.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

cVal	Complex value to be assigned.
Return value	TRUE if successful, FALSE if not.

CModUser *pMod;
CComplex z1(1.7, -3.1);
pMod->SetSVVal(0, 230.0);
pMod->SetSVVal(3, z1);	

Sections	Init.
See also GetDSVVal, GetSVVal, SetValNode_nc.	

SetSymbolicGSEntry

Marks non-vanishing Jacobian matrix elements. Before filling the Jacobian matrix with real or complex entries, the simulator must be informed about the non-vanishing entries of the matrix. Several additive terms can be filled at one entry site using the subentry functionality. The method is mainly used when filling the right side vector. Several predefined flag values can be combined using the '|' (OR-) operator: D_DT_OPERAT|RS_DONTFILL.	
BOOL SetSymbolicGSEntry(long Row, long Col) BOOL SetSymbolicGSEntry(long Row, long Col, long sub, long Flag=0L)	
Row	Row number of the Jacobian entry (starting at 0).
Col	Column number of the Jacobian entry (starting at 0).
Sub	Index of the subentry in case of an overloaded Jacobian entries.
Flag	Flag to announce special entries to the simulator. Predefined are D_DT_OPERAT for entries containing a temporal derivative and RS_DONTFILL to prevent the simulator from automatic filling of the right side vector for this particular entry.
Return Value	TRUE if successful, FALSE if not.
CModUser *pMod; SetSymbolicGSEntry(0, 0); SetSymbolicGSEntry(1, 2, 0, RS_DONTFILL); SetSymbolicGSEntry(1, 2, 1,RS_DONTFILL|D_DT_OPERAT);	
Sections	Prepare.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

See also SetCplxGSEntry, SetRealGSEntry.

SetUModCloseFct

Assigns a close function to the model. In the **RegisterUMODELFct** function of the source code, the model functions must be announced to the Twin Builder simulator. The method is used to announce the Close function, which is used for final clean up after simulation run. **Note:** The Close function should not block the simulator indefinitely, as the simulator needs to forcefully terminate the solver thread after a certain timeout at the end of simulation run. Therefore, operations like user interactions that could block the simulator a long period must not be used in this function.	
void SetUMODCloseFct(SIMFCT_UMODEL pFct)	
pFct	Pointer to a model function of type **FCTDECL function (*CModUser)**.

FCTDECL RegisterUMODELFct(CModUser *pMod)
{
pMod->SetUMODCloseFct(modelclose);
return TRUE;
}	

Sections	RegisterUMODELFct.
See also SetUMODInitFct, SetUMODPrepFct, SetUMODSimFct, SetUMODValidFct.	

SetUModInitFct

Assigns an initialize function to the model. In the **RegisterUMODELFct** function of the source code, the model functions must be announced to the Twin Builder simulator. The method announces the Initialize function, where the component initialization is performed.	
void SetUMODInitFct(SIMFCT_UMODEL pFct)	
pFct	Pointer to a model function of type FCTDECL function (*CModUser).
FCTDECL RegisterUMODELFct(CModUser *pMod)	

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
{
 pMod->SetUMODInitFct( ini_func );
 return TRUE;
}
```

Sections	RegisterUMODELFct.
See also SetUMODCloseFct, SetUMODPrepFct, SetUMODSimFct, SetUMODValidFct.	

SetUModPrepFct

Assigns a prepare function to the model.

In the **RegisterUMODELFct** function of the source code, the model functions must be announced to the Twin Builder simulator. The method announces the Prepare function, where all the nodes and internal states of the model are defined.

```
void SetUMODPrepFct( SIMFCT_UMODEL pFct )
```

pFct	Pointer to a model function of type FCTDECL function (*CModUser).

```
FCTDECL RegisterUMODELFct( CModUser *pMod )
{
 pMod->SetUMODPrepFct( model_basics );
 return TRUE;
}
```

Sections	RegisterUMODELFct.
See also SetUMODCloseFct, SetUMODInitFct, SetUMODSimFct, SetUMODValidFct.	

SetUModSimFct

Assigns a simulate function to the model.

In the **RegisterUMODELFct** function of the source code, the model functions must be announced to the Twin Builder simulator. The method announces the Simulate function, where the mathematical model description is computed.

```
void SetUMODSimFct( SIMFCT_UMODEL pFct )
```

pFct	Pointer to a model function of type FCTDECL function (*CModUser).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
FCTDECL RegisterUMODELFct( CModUser *pMod )
{
 switch( pMod->GetAnalysisType() )
 {
 case DC_:
 pMod->SetUModSimFct( SIM_DC );
 break;
 case TR_:
 pMod->SetUModSimFct( SIM_TR );
 break;
 case AC_:
 pMod->SetUModSimFct( SIM_AC );
 break;
 default:
 break;
 }
}
```

Sections	RegisterUMODELFct.

See also SetUMODCloseFct, SetUMODInitFct, SetUMODPrepFct, SetUMODValidFct.

SetUModValidFct

Assigns a simulate function to the model.

In the **RegisterUMODELFct** function in the source code, the model functions must be announced to the Twin Builder simulator. The method announces the Validate function, which is called after every complete time step.

```
void SetUMODValidFct( SIMFCT_UMODEL pFct )
```

pFct	Pointer to a model function of type FCTDECL function (*CModUser).

```
FCTDECL RegisterUMODELFct( CModUser *pMod )
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
{
 pMod->SetUMODValidFct( model_basics );
 return TRUE;
}
```

Sections	RegisterUMODELFct.
See also SetUMODCloseFct, SetUMODInitFct, SetUMODPrepFct, SetUMODSimFct.	

SetUnitNameNode

Assigns units to non-conservative nodes.

Twin Builder components can provide the units of its non-conservative nodes in the schematic. The method assigns the units. Since these values inform the user only, there is no restriction in unit names.

```
void SetUnitNameNode_nc( LPCTSTR Name, LPCTSTR Unit )
void SetUnitNameNode_nc( int No, LPCTSTR Unit )
```

Name	Name of the non-conservative node as assigned in AddNode_nc.
No	Index of the non-conservative node (starting at 0) within the model.
Unit	String representing the unit of the quantity in or out of the model.

```
CModUser *pMod;
pMod->SetUnitNameNode_nc( 0, "Volts" );
pMod->SetUnitNameNode_nc( "i_dens", "A/sqfeet" );
pMod->SetUnitNameNode_nc( "speed", "Warp" );
```

Sections	Prepare.
See also AddNode_nc, SetInfoNode_nc.	

SetUserData

Stores arbitrary instance-specific user data memory.

It is often useful to reuse or exchange data in several model functions. The user data structure lets you avoid the use of global variables which would cause interference with other

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

instances of the same model. This method connects the user data to the model instance for further access using the **GetUserData** function.

BOOL SetUserData(void *userdata, long size);	
userdata	Pointer to the user data memory block.
size	Size of the user data memory in bytes.
Return value	TRUE if successful, FALSE if not.

```
struct UD { double d1; int ii; };

//...

UD *pUD = (UD*)pMod->AllocateUserDataMemory( sizeof( UD ) );

pMod->SetUserData( pUD, sizeof( UD ) );

pUD->d1 = 17.5;

pUD->ii = 125;
```

Sections	Prepare, Init, Simulate, Validate, Close.

See also `AllocateUserDataMemory`, `GetDCUserData`, `GetUserData`.

SetValNode_nc

Assigns a value to the non-conservative node. Non-conservative nodes provide a very versatile interface to pass signals between different components on a Twin Builder schematic. Use this method to assign data of nearly arbitrary type to these nodes. For a complex value, the real and imaginary parts of the node_nc are set. For a real value, only the real part is set. That means the imaginary part is not influenced by setting a real value. The previous imaginary value still exists.

```
BOOL SetValNode_nc( LPCTSTR Name, double dVal )

BOOL SetValNode_nc( LPCTSTR Name, int iVal )

BOOL SetValNode_nc( int No, double dVal )

BOOL SetValNode_nc( LPCTSTR Name, const CComplex &cVal )

BOOL SetValNode_nc( int No, const CComplex &cVal )

BOOL SetValNode_nc( LPCTSTR Name, LPCTSTR Strg )

BOOL SetValNode_nc( int No, LPCTSTR Strg )
```

Name	Name of the non-conservative node as defined in **AddNode_nc**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

No	Index of the non-conservative node (starting at 0) within the model.
dVal	A value of type double to be assigned.
iVal	A value of type integer to be assigned.
cVal	A complex value to be assigned.
Strg	A character string to be assigned.
Return value	TRUE if successful, FALSE if not.

```
CModUser *pMod;
double r = 27.5;
int i = 30;
CComplex z(2.3, -3.14);
pMod->SetValNode_nc( "speed", r );
pMod->SetValNode_nc( "step", i );
pMod->SetValNode_nc( 4, z );
pMod->SetValNode_nc( 5, "data.txt" );
pMod->SetValNode_nc( "displacement", 1380.0 );
```

Sections	Init, Simulate, Validate.

See also `AddNode_nc`, `GetValNode_nc`, `GetValNode_ncCplx`, `GetValNode_ncStrg`, `SetDataTypeNode_nc`, `SetInfoNode_nc`, `SetUnitNameNode`.

SetValNode_ncFile

Assigns a file name to a non-conservative node.
Use this method to pass file names from the model to the simulator or the user.

```
BOOL SetValNode_ncFile( LPCTSTR Name, LPCTSTR Strg )
BOOL SetValNode_ncFile( int No, LPCTSTR Strg )
```

Name	Name of the non-conservative node as defined in **AddNode_nc**.
No	Index of the non-conservative node (starting at 0) within the model.
Strg	A file name to be assigned.
Return value	TRUE if successful, FALSE if not.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

CModUser *pMod;
pMod->SetValNode_ncFile("file1", "autoexec.bat");
pMod->SetValNode_ncFile(3, "manual.pdf");

Sections | Init, Simulate, Validate.
See also AddNode_nc, GetValNode_nc, GetValNode_ncFile, GetValNode_ncStrg, SetDataTypeNode_nc.

SetValNode_State

Assigns a value to an internal state. Use this method to assign a value to an internal state. The internal states are created by the **AddNode_State** function. This kind of storage must be preferred over global variables whenever applicable. Every model instance has its own memory for state variables. The simulator stores and restores state variables in case of a backstep event. Complex states are mainly used in AC analysis.
BOOL SetValNode_State(LPCTSTR Name, double dVal) BOOL SetValNode_State(int No, double dVal) BOOL SetValNode_State(LPCTSTR Name, CComplex &cVal) BOOL SetValNode_State(int No, const CComplex &cVal)
Name | Name of the internal state as defined in **AddNode_State**.
No | Index of the internal state (starting at 0) within the model.
dVal | A value of type double to be assigned.
cVal | A complex value to be assigned.
Return value | TRUE if successful, FALSE if not.

CModUser *pMod;
CComplex z(3.5, -1.7);
pMod->SetValNode_State("temperature", 36.5);
pMod->SetValNode_State(4, z);

Sections | Init, Simulate, Validate.
See also AddNode_State, GetValNode_State, GetValNode_StateCplx.

SetValPtrNode_nc

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Links a non-conservative node to a double pointer. A non-conservative node can be linked to a static variable in the Dynamic Link Libraries memory, so that every change of the value of the double pointer affects the value of the node. This is useful for variables defined within the **UserData** Memory of a model.
BOOL SetValPtrNode_nc(LPCTSTR Name, double *dVal) BOOL SetValPtrNode_nc(int No, double *dVal)

Name	Name of the non-conservative node as assigned in **AddNode_nc**.
No	Number of the non-conservative node (starting at 0) within the model.
*dVal	Pointer to a double value.
Return value	TRUE if successful, FALSE if not.

```
CModUser *pMod;
double r[5];
int i;
for( i=0; i<5; i++ )
 r[i] = sqrt(i);
double* p1 = &((CUserData*)pData)->m_dVal;
pMod->SetValPtrNode_nc( "roots", p1 );
```

Sections	Init, Simulate, Validate.
See also `GetValNode_nc`, `SetInfoNode_nc`, `SetValNode_nc`.	

SetValSubNode_nc

Assigns a value to an output vector element. Non-conservative output vectors of C-Models must be assigned by element with this function.
long SetValSubNode_nc(LPCTSTR Name, long Index, double Val); long SetValSubNode_nc(LPCTSTR Name, long Index, LPCTSTR strVal);

Name	Name of the non-conservative base node.	
Index	Index of the vector element, starting at 0.	
Val	Double value to be assigned to the output vector.	

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

*Val	String to be assigned to the vector element.
Return value	TRUE if successful, FALSE if not.
pMod->SetValSubNode_nc("OutVec", 0, 1.23); pMod->SetValSubNode_nc("OutVec", 1, 2.44);	
Sections	Init, Simulate, Validate.
See also AddNode_nc, AddSubNode_nc, SetValNode_nc.	

Callback Functions

The Twin Builder C interface provides a number of callback functions to access simulator-specific data or to influence the simulator behavior. These functions can be used in nearly all model functions. The calling **CModUser** object is always a parameter of these functions.

CHAR_IN

Access the input variable in case of a characteristics model. The Twin Builder C interface enables the programming of arbitrary nonlinear relationships in a characteristic model. See ***C-Models Used as Characteristics***. You can connect this kind of model to any characteristics input node of Twin Builder basic components, as well as to other C-Models. The characteristics model uses the **CHAR_IN** function to access the independent variable *x* set by the connected element.	
`double CHAR_IN(CModUser *pModU )`	
Return Value	Value of the input variable of a characteristics block.

CHAR_OUT

Sets the output variable in case of a characteristics model. The Twin Builder C interface enables the programming of arbitrary nonlinear relationships y=f(x) in a characteristics model. See ***C-Models Used as Characteristics***. You can connect this kind of model to any characteristics input node of Twin Builder basic components, as well as to other C-Models. The characteristics model uses the **CHAR_OUT** function to set the *y* value to be returned to the connected element.	
`void CHAR_OUT(CModUser *pModU, double y )`	
y	Output y=f(x) of a characteristics model.

CHAR_OUT_DERIVE

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Sets the partial derivative δy/δx of a characteristics model. The Twin Builder C interface enables the programming of arbitrary nonlinear relationships in a characteristic model. See ***C-Models Used as Characteristics***. You can connect this kind of model to any characteristics input node of Twin Builder basic components. as well as to other C-Models. Since in most cases the partial derivative δy/δx is also needed for the Jacobian matrix, this function sets its values to be accessed by the connected component.
`void CHAR_OUT_DERIVE(CModUser *pModU, double dx_dy )`

dx_dy	The partial derivative of the output quantity *y* with respect to the input quantity *x*.

GET_SAMPLETIME

Gets the sample time of a component. The Twin Builder C interface enables the definition of components with arbitrary sample times to be defined at initialization. These components are executed at this sample rate independently of all other present components. This callback function returns the user input at the corresponding node. In case of a zero input, it returns the current sample time. See ***C-Models with Specified Sample Time***.
`double GET_SAMPLETIME(CModUser *pModU)`

Return Value	Current sample time of a component.

GetInfo

Returns sheet-related path information. This callback function returns the file path to the currently simulated sheet or the sheet name. The output is controlled by the first function parameter, which can be set to the predefined values **PATH2MODELDESCRIPTION** or **MODELFILENAME**.
`void GetInfo( long lOption, char *pszString, long lLen );`

Return Value	Path information.
lOption	Used to select the function's output: **SYSLIB_DIR**, **USERLIB_DIR**, **PERSLIB_DIR**, **SIMPL_PRJ_DIR**, **SIMPL_TMP_DIR**, **SIMPL_INST_DIR**, **DSN_NAME**, **SETUP_NAME**, **PRJ_PATH**, **PRJ_FILENAME**, **NETLIST_PATH**, **NETLIST_FILENAME**.
pszString	Character string containing the function's output.
lLen	Maximum length of output. Shall be set to **_MAX_PATH**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Return Value	lOption	Path information.
	SYSLIB_DIR	Path to system libraries.
	USERLIB_DIR	Path to user libraries.
	PERSLIB_DIR	Path to personal libraries.
	SIMPL_PRJ_DIR	Path to Twin Builder project directory.
	SIMPL_TMP_DIR	Path to Twin Builder temp directory.
	SIMPL_INST_DIR	Path to Twin Builder installation directory.
	DSN_NAME	Name of the simulated design.
	SETUP_NAME	Name of the simulated setup.
	PRJ_PATH	Path to current project file.
	PRJ_FILENAME	Name of the current project file.
	NETLIST_PATH	Path to current netlist (SML) file.
	NETLIST_FILENAME	Name of the current netlist (SML) file.

getPATH

Returns sheet related path information.

This callback function returns the file path to the currently simulated sheet or the sheet name. The output is controlled by the first function parameter, which can be set to the pre-defined values **PATH2MODELDESCRIPTION** or **MODELFILENAME**.

```
void getPATH( long lOption, char *pszString, long lLen );
```

Return Value	Path information.
lOption	Used to select the function's output. • **PATH2MODELDESCRIPTION** returns the file path. • **MODELFILENAME** returns the sheet name without an extension.
pszString	Character string containing the function's output.
lLen	Maximum length of output. Shall be set to **_MAX_PATH**.

ISIM_BASE

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Gets the current simulation time in TR simulation mode (T). Components with explicit time dependency can use this callback function to access simulation time.	
`double ISIM_BASE(CModUser *pModU)`	
Return Value	Simulation time.

ISIM_BASE_MAX

Returns maximum time step in TR simulation mode (HMAX).	
`double ISIM_BASE_MAX(CModUser *pModU)`	
Return Value	Maximum time step.

ISIM_BASE_MIN

Returns minimum time step in TR simulation mode (HMIN).	
`double ISIM_BASE_MIN(CModUser *pModU)`	
Return Value	Minimum time step.

ISIM_BASE_STEP

Returns current time step in TR simulation mode (H).	
`double ISIM_BASE_STEP(CModUser *pModU)`	
Return Value	Current step size.

ISIM_BASE_TEMP

Returns current ambient temperature in °C (THETA).
`double ISIM_BASE_TEMP(CModUser *pModU)`

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Return Value	Current ambient temperature in °C.

ISIM_BASE_XEND

Returns simulation end time (TR) or ending frequency (AC).	
`double ISIM_BASE_XEND(CModUser *pModU)`	
Return Value	Simulation end time (TR) or ending frequency (AC).

ISIM_BASE_XSTART

Returns simulation start time (TR) or starting frequency (AC).	
`double ISIM_BASE_XSTART(CModUser *pModU)`	
Return Value	Simulation start time (TR) or starting frequency (AC).

ISIM_ECM_D_DT

Returns temporal derivative operator in TR analysis. Use the callback function to access the numerical equivalent to the d/dt-operator in TR analysis. Since Twin Builder uses variable step size algorithms to simulate the system, this value is not constant. The function must be used every time the temporal derivative is needed.	
`double ISIM_ECM_D_DT(CModUser *pModU)`	
Return Value	1/dt

ISIM_ECM_IEMAX

Returns maximum current error (IEMAX) of the Newton-Raphson iterations during TR calculation.	
`double ISIM_ECM_IEMAX(CModUser *pModU)`	
Return Value	Maximum current error.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

ISIM_ECM_ITERAT

Returns the maximum number of iterations (ITERATMAX).	
`double ISIM_ECM_ITERAT(CModUser *pModU)`	
Return Value	Maximum iteration number.

ISIM_ECM_LDF

Returns the local discretisation error (LDF).	
`double ISIM_ECM_LDF(CModUser *pModU)`	
Return Value	Local discretization error.

ISIM_ECM_NEW

Returns step information. The callback function returns TRUE in normal smooth operation of the simulator when it progresses continuously with time. Several events can force the simulator to step back in time. During such a step it returns FALSE.	
`DOUBLE ISIM_ECM_NEW(CModUser *pModU)`	
Return Value	Step information: 1.0, if current step is a new step, 0.0 if step is repeated.

ISIM_ECM_P

Returns complex frequency operator in AC analysis. Use the callback function to access the complex frequency (the numerical equivalent to the d/dt -operator) in AC analysis. Obtained through Laplacian transformation this results to the complex term j∗ω. This requires the update of the operator in every frequency step of an AC analysis. Therefore, this callback function must be used every time in the simulate section to access the correct value.	
`CComplex ISIM_ECM_P(CModUser *pModU)`	
Return Value	j∗ω.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

ISIM_ECM_RELTOL

Returns relative tolerance value during TR calculation.	
`double ISIM_ECM_RELTOL (pMod)`	
Return Value	Relative tolerance value.

ISIM_ECM_SOLVER

Returns the solver type of the network simulator.	
`eSolverType ISIM_ECM_SOLVER(CModUser *pModU)`	
Return Value	EULER_ or TRAPEZOIDAL_.

ISIM_ECM_UEMAX

Returns maximum voltage error (VEMAX) during TR calculation.	
`ISIM_ECM_UEMAX(pMod)`	
Return Value	Maximum voltage error.

OSIM_ECM_REJECT

Requests back step. Use the callback function to force the simulator to step back and repeat the last time step with a smaller (simulator-defined) step size. You can use it when numerical instabilities occur which are related to a too large step size or to approach switching events.
`void OSIM_ECM_REJECT(CModUser *pModU)`

OSIM_ECM_SYNC

Sets new user-defined step size for ECM models. Use the callback function to force the simulator to step back and repeat the last time step with

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

a user-defined smaller step size. You can use it when numerical instabilities occur which are related to a too large step size or to approach switching events. The step size must be greater than or equal to the minimum time step (HMIN).
`void OSIM_ECM_SYNC(CModUser *pModU, double dt)`

dt	New step size.

OSIM_SYNC

Requests back step and setting new user-defined step size for TFM models. Use the callback function to force the simulator to step back and repeat the last time step with a user-defined smaller step size. You can use it when numerical instabilities occur which are related to a too large step size or to approach switching events. The step size must be greater than or equal to the minimum time step (HMIN).
`void OSIM_SYNC(CModUser *pModU, double dt)`

dt	New step size.

Report2Sim

Displays messages in the simulator output window or opens pop-up windows. The callback function displays important information. You can use it to display information or errors. These messages help to identify problems and avoid unexpected simulation results. In case of an error, this routine does not stop the simulator. This has to be done in returning a 'FALSE' value when leaving the model function where the error occurred.
`BOOL Report2Sim( CModUser *pModU, long Flag, LPCTSTR Message )`

Flag	Used to force various messaging behavior. Predefined values are: • **UMOD_ERR** – Prints an error message in the simulator output window and activates the simulator output window. To stop the simulation the model function (Initialize, simulate,...) must return FALSE. • **UMOD_WRN** – Opens a pop-up window with a user input request. You can cancel or continue the simulation. In the first case, the callback function returns TRUE; in the second, it returns FALSE. • **UMOD_WRN_OK** – Displays the warning text in a pop-up window with an **OK** button only. • **UMOD_INF** – Prints information in the simulator

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	output window without affecting the simulation. • **UMOD_INF_AW** – Prints information in the simulator output window without affecting the simulation and activates the simulator output window.
Message	The message text to be displayed.
Return Value	User response in case of a warning. TRUE: Cancel, FALSE: continue.

SET_SAMPLETIME

Sets the sample time of a component. Use the callback function to set a user-defined sample time for a component. The component is calculated at the defined sample time. Twin Builder can handle components with fixed sample time parallel to continuous components. See ***Callback Functions***.
`void SET_SAMPLETIME(CModUser *pModU, double DTS)`

DTS	Sample time.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

22 - VHDL-AMS Models in Twin Builder

VHDL-AMS (**V**ery high-speed integrated circuit **H**ardware **D**escription **L**anguage – **A**nalog **M**ixed **S**ignal) is a standardized language used for describing digital, analog, and mixed-signal systems.

The Institute of Electrical and Electronics Engineers (IEEE) standardized the VHDL-1076 language as a Hardware Description Language (HDL) for digital models. The VHDL standard from 1993 was extended in 1999 for the description of analog and mixed-signal models in the form of the IEEE 1076.1 standard for VHDL-AMS (hereafter referred to as VHDL).

See ***VHDL-AMS Language Fundamentals*** for detailed information on the VHDL-AMS language in Twin Builder.

You can access the *VHDL-AMS Tutorial* documentation within the help system.

In addition to the functionality provided by the **VHDL-AMS Model Editor**, Twin Builder supports the development and simulation of VHDL-AMS analog, digital, and mixed-signal models in the following ways:

- Import existing VHDL-AMS models in ASCII text into a library or a subsheet on a schematic.
- Export VHDL-AMS models in libraries, as well as text or graphical subsheets, to ASCII text files.
- Export schematics containing VHDL-AMS models as netlists to VHDL ASCII files.
- VHDL-AMS models can instantiate Twin Builder models as foreign models, so you can take advantage of the large number and variety of highly optimized and fast Twin Builder models.

Understanding VHDL-AMS Models

Twin Builder provides several libraries of VHDL-AMS components developed according to IEEE 1076.1 (VHDL Analog and Mixed Signal Extensions Standard) and IEEE 1076 (VHDL standard). These libraries are accessible via the Component Libraries window Components tab.

- The **Basic_VHDLAMS** library (**Basic_VHDLAMS.asmd**) contains basic circuit components, blocks, and measurement devices, as well as fluidic, magnetic, mechanical, and thermal components. It also contains a number of time functions.
- The **Digital_Elements** library (**Digital_Elements.asmd**) contains basic components commonly used for simple digital circuits (A-to-D and D-to-A converters, counters, flip-flops, latches, gates, and so on).
- The **Transformations** library (**Transformations.asmd**) contains auxiliary components (called **OmniCasters**) that easily connect different data types and natures.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The models provided are open and can be used to derive more advanced models by copying the description to a user library and editing the text to modify the model. You can use and distribute the files if the copyright statement, included in each model description, is not removed. To access the model description, right-click the component name in the **Project Manager** pane and select **View Component Help**.

The functionality of all VHDL-AMS models is a subset of the equivalent Twin Builder models available in the **Basic_Elements** library (**Basic_Elements.asmd**). You can use VHDL-AMS models in parallel with Twin Builder models.

The digital models operate with digital signals and can be characterized with rise time/fall time/propagation delays. They do not have any conservative nodes but can be connected to analog quantities using **OmniCasters** (in the **Transformations** library).

Across and Through Quantities of Natures

VHDL-AMS models can support nature types for several physical domains. Nature types are properties of conservative nodes (also referred to as ports or terminals) of models. At least one specific nature exists for each domain. *Across* and *through* quantities are associated with each nature. The following table links the *across* and *through* quantities for each nature type:

Nature	Across	Through	Circuit
ELECTRICAL	Voltage [V]	Current [A]	
FLUIDIC	Pressure [Pa]	Flow Rate [m^3/s]	
MAGNETIC	Magneto Motive Force [A]	Magnetic Flux [Vs]	
TRANSLATIONAL	Displacement [m]	Force [N]	
TRANSLATIONAL_V	Velocity [m/s]	Force [N]	
ROTATIONAL	Angle [rad]	Torque [Nm]	
ROTATIONAL_V	Angular Velocity [rad/s]	Torque [Nm]	
THERMAL	Temperature [K]	Heat Flow [J/s]	

The circuit graphic illustrates how *across* and *through* quantities are measured. The measuring direction is marked by the red dot on the model symbol. The red dot is always at pin 1 of a model.

- **Across quantities** – Value is calculated by subtracting the value at Pin2 from the value at Pin1.
- **Through quantities** – Value is positive if the quantity flows into the model at the pin marked with the red dot.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Packages and Models in Libraries

Packages are collections of reusable declarations and definitions such as types, constants, functions, procedures, and natures.

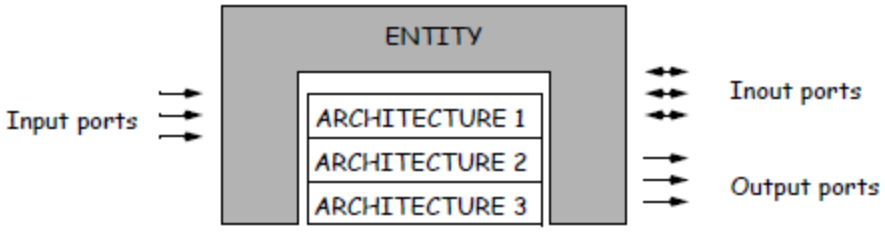
Packages appear under green package folder symbols in the tree and have a green rectangle in front of their names. Standardized packages from IEEE (such as **math_real** and **textio**) and proposed packages from IEEE (such as **electrical_systems** and **thermal_systems**) are available in the **Ieee.apkg** and **Std.apkg** libraries in the VHDL Model Editor **Edit Model** dialog box.

To use declarations from a package, the corresponding package must be included in the model description, and the package must be available in an installed model library within the current Twin Builder configuration. In the following example the resistor model (res) uses the **ELECTRICAL** nature for its conservative pins. Consequently, the **ELECTRICAL_SYSTEMS** package from the **Ieee** library must be included in the model description.

```
LIBRARY IEEE;
USE IEEE.ELECTRICAL_SYSTEMS.ALL;
ENTITY res IS
  PORT (QUANTITY r_nom : REAL := 1000;
        TERMINAL p,m : ELECTRICAL);
END ENTITY res;
```

Entities and Architectures of VHDL-AMS Models

VHDL-AMS models consist of two parts: an **ENTITY** declaration and one or more **ARCHITECTURE** descriptions. The **ENTITY** describes the interface of the model and declares inputs, outputs, constant value parameters, conservative pins, and so on. The **ARCHITECTURE** defines the behavior of the model, and several modeling styles may be used for this description such as behavioral, dataflow, and structural. It is possible to associate multiple architectures with an entity declaration, and only the selected architecture is used during simulation.



The model description for a passive resistor model included with the system follows. Explanations of the statements used in the model description are included.

VHDL-AMS Resistor Model Example

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To view the model description, select **Tools** > **Edit Libraries** > **Models**, and click to select the model from the list in the **Edit Libraries** dialog box. The model description appears in the **Model text** pane to the right of the list. The example resistor model has an interface with two electrical terminals and one non-conservative input for the static resistance value.

```
LIBRARY IEEE;
USE IEEE.ELECTRICAL_SYSTEMS.ALL;
ENTITY R IS
 PORT (QUANTITY R : RESISTANCE := 1.0e+3;
 TERMINAL p,m : ELECTRICAL);
END ENTITY R;
ARCHITECTURE behav OF R IS
 QUANTITY v ACROSS i THROUGH p TO m;
BEGIN
 v == i*R;
END ARCHITECTURE behav;
```

ENTITY	Interface description of the model *r*.
PORT	Conservative and non-conservative pins of a circuit. Here, there are two electrical **TERMINAL**s that represent the plus (+) and minus (–) pins of the model, and one non-conservative **QUANTITY** input for the resistance value of the model.
TERMINAL	Conservative pin associated with a domain. Here, there are two conservative pins *p* and *m* declared for the **ELECTRICAL** domain.
ARCHITECTURE	Defines the behavior *behav* of the model *r*. **QUANTITY** voltage **ACROSS** current **THROUGH** p **TO** m; This statement defines voltage as an **ACROSS** quantity and current as a **THROUGH** quantity between the pins *p* and *m* for the model. v == i * R; This statement specifies the model behavior v = i*R in VHDL-AMS equation form.

Creating and Editing VHDL-AMS Models

Twin Builder includes the VHDL-AMS Model Editor to facilitate creating and editing VHDL-AMS models. See ***Using the VHDL-AMS Model Editor*** for details.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Placing and Connecting VHDL-AMS Models

You can place VHDL-AMS models on a schematic in the same way that you would place any component. To place a component, select a component in the **Project Manager Components** tree and drag it onto the sheet.

Connections between VHDL-AMS components and other components on the schematic are made in wire mode. To enter wire mode, select **Draw** > **Wire**, press Ctrl+W, or click two connection points to create a link between them. Click within the schematic while not on a connection point to set corners to change the direction of the wire. To exit wire mode, press Esc. You can also connect components by overlapping their pins, or you can connect one to a wire by placing a component's pin over the wire. Unconnected wires are shown as broken red lines.

Related Topics

- Placing Components
- Wiring Components

Using Transformation Models

Complex simulation models usually need simple interface models to connect different data types and natures, so that the real subject of simulation can be investigated easily. VHDL-AMS models need transformation models that perform data type conversions; VHDL-AMS models from different domains need transformation models that connect conservative nodes of different natures.

OmniCasters are interface models used to interface analog quantities with digital signals, or to connect digital signals of different data types.

A "flexible" **OmniCaster** model that performs conversions depending on the connected data types — as well as "fixed" **OmniCaster** models with predefined data types — are available for use. The flexible **OmniCaster** model uses a fixed **OmniCaster** model based on the data types of the pins connected to it. The fixed **OmniCaster** models have an **ENTITY** description according to the nature of the conversion and an **ARCHITECTURE** description that may use function calls to perform the conversion.

The different data types considered for signals are **REAL**, **INTEGER**, **BIT**, **BOOLEAN**, **BIT_VECTOR**, **STD_LOGIC**, and **STD_LOGIC_VECTOR**. The only data type considered for analog quantities is **REAL**. Some transformations can be specified with propagation delay, rise time, fall time, threshold, and output value parameters. The functions used for the type conversions are available in the **omnicaster_package**.

Connect conservative nodes of different domains using a Domain-to-Domain (D2D) model available in the **Nature Transformations** folder of the **transformations.asmd** library. It is also possible to connect a conservative node to a non-conservative node using a C2NC connection

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

model. In this case, the across value from the conservative node is transferred to the non-conservative node.

> **Note:**
>
> Unlike the **OmniCasters** described in VHDL-AMS, the D2D and C2NC are not VHDL-AMS models. Consequently, if schematics that use D2D or C2NC models are exported to an ASCII netlist, the exported description may not simulate in a third-party VHDL-AMS simulator.

Defining Model Properties

Every VHDL-AMS model placed on the sheet has a **Properties** dialog box where you can modify its parameters, as well as select simulation languages and architectures. Unlike SML models, VHDL-AMS models do not use component dialog boxes for parameterization of the components. All parameters are listed and changed in the **Parameters** tab.

To open the **Properties** dialog box, do one of the following:

- Double-click the component.
- Right-click the component and select **Properties**.
- Select **Edit** > **Properties** for the selected component.

The number of parameters in the **Parameters** and **Output/Display** dialog boxes varies depending on the selected model. To enter a parameter value, click in the input field of the corresponding parameter and enter a numerical value (with or without suffix), a variable, or an expression. Default values are used if no other value is defined for the parameter

Property expressions may contain variables that are generic, quantity, or signal properties of VHDL-AMS components, such as **r1.r**. Only properties defined in the model's entity are available for this use, consistent with the VHDL-AMS language fundamentals. In previous versions of Twin Builder, properties from a model architecture were available for use, but this is no longer the case.

VHDL-AMS Language Fundamentals

The VHDL-AMS language fundamentals described in this section provide a quick reference guide to find a specific statement, or the syntax for a specific statement, needed to write VHDL-AMS code.

For more information, visit the IEEE 1076.1 Working Group at http://www.eda.org/vhdl-ams.

The VHDL-AMS language is case insensitive; upper case letters are equivalent to lower case letters. In this document, reserved words are in UPPER case and shown in **BOLD**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Identifiers are simple names starting with a letter and may have letters and digits. The underscore character is allowed but not as the first or last character of an identifier.

A comment starts with two consecutive hyphens, “--”, and continues until the end of the line.

The following table shows the syntax used in the documentation.

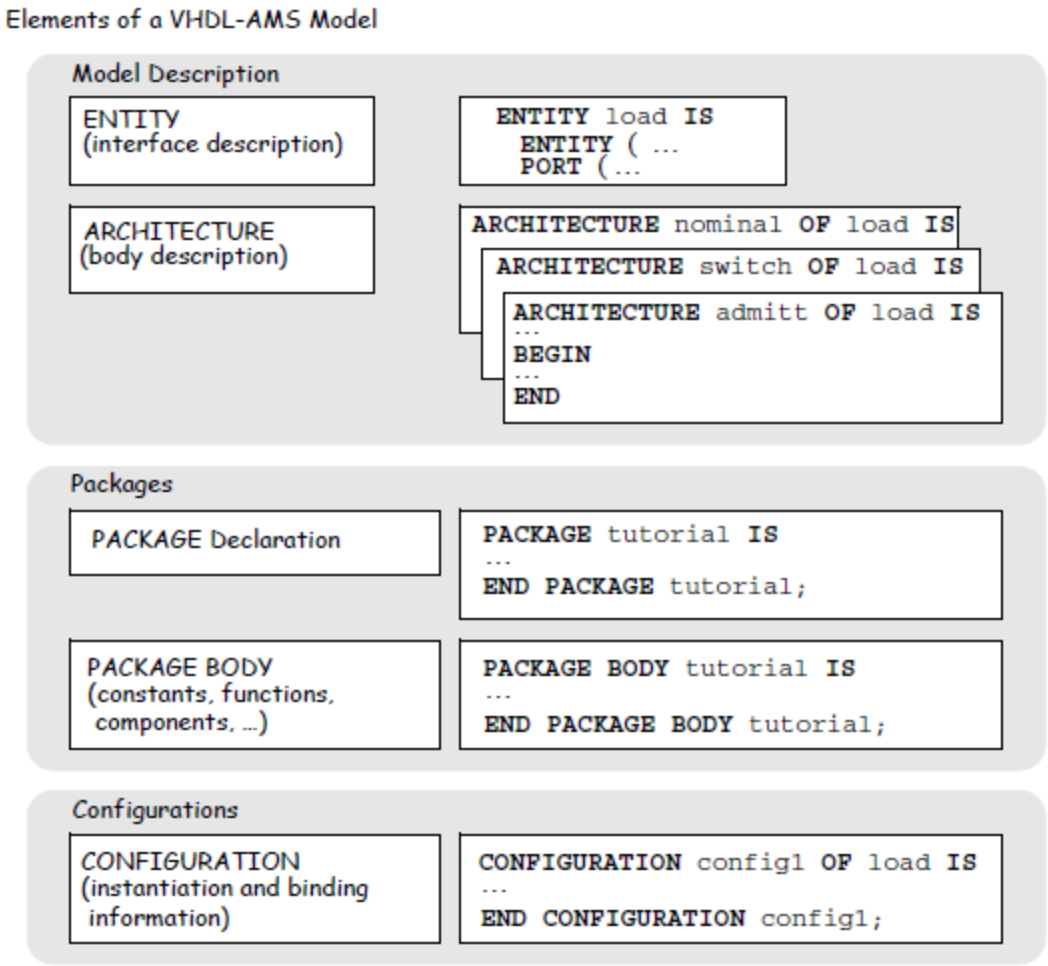
Syntax	Description
ENTITY	VHDL-AMS keyword
[expression]	optional entry
[name | string]	alternative selection
identifier {,…}	repeated entries
=> <= :=	assignment operators
==	simultaneous statement

Design Units

The VHDL-AMS language allows the definition of models for analog, digital, and mixed signal circuits and systems in a standardized language. Design units (also known as library units) are segments of VHDL-AMS code that can be compiled separately and stored in a library.

An entity normally consists of five basic elements, or design units: entities, architectures, packages, package bodies, and configurations. Entities and architectures are the only two design units that must exist in any VHDL-AMS design description. Packages and configurations are optional.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Entities and Architectures

Each VHDL-AMS design description consists of an **ENTITY** declaration and one or more architectures. The **ENTITY** declaration defines the inputs to and outputs from the model, as well as any **GENERIC** parameters used by the different implementations. Each **ARCHITECTURE** defines a different implementation or behavior of a given design unit.

Entity Declaration

An entity declaration defines input to a model, outputs supported by the model, and generic parameters used by the different implementations of the model.
ENTITY entity_name **IS** **GENERIC** (generic_list); -- optional generic list **PORT** (port_list); -- input/output signal ports **END** [**ENTITY**] name;

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

generic_ list	Specifies static information to be communicated to a model from its environment for all architectures. These include timing information (setup, hold, delay times), ambient information (temperature), and other parameters.
port_list	Specifies dynamic information to be communicated between a model and its environment for all architectures. A port can be represented by a quantity, terminal, or signal. The mode of a port defines the directions of the signals on that port, and is one of **IN**, **OUT**, **INOUT**, **BUFFER**, or **LINKAGE**.
Port Modes	An **IN** port can be read but not updated within the module. An **IN** port cannot appear on the left side of a signal assignment. An **OUT** port can be updated but not read within the module. An **OUT** port cannot appear on the right side of a signal assignment. An **INOUT** port is bidirectional and can be both read and updated, with multiple update sources possible. Signal objects can use the mode **IN**, **OUT**, and **INOUT**; quantity objects can use **IN** and **OUT** whereas terminals have no direction mode. Ports of type **BUFFER** and **LINKAGE** are transformed to **INOUT** types in Twin Builder.

ENTITY spring_tr **IS**
GENERIC(s0: DISPLACEMENT:= 0.0); -- list of generic parameters
PORT(**QUANTITY** c: **IN** STIFFNESS:= 100.0; -- list of quantity ports
TERMINAL tr1, tr2 : TRANSLATIONAL_V; -- list of terminal ports
SIGNAL ctrl: **IN** BIT); -- list of signal ports
END ENTITY spring_tr;

Architecture

An architecture defines one particular implementation of a design unit (model behavior), at some desired level of abstraction.

ARCHITECTURE architecture_name **OF** entity_name **IS**
…declarations
BEGIN
…concurrent/sequential/simultaneous statements
END [**ARCHITECTURE**][architecture_name]

declarations	Information used in the model description. Declarations include data types, constants, signals, files, components, attributes, subprograms, and others.
concurrent statements	Digital statements that are executed asynchronously with respect to each other. They describe a design unit at one or more levels of modeling

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	abstraction, including dataflow, structural, and/or behavior.
sequential statements	Statements that are executed in the order in which they appear. They define algorithms for the execution of a subprogram or process.
simultaneous statements	Statements that are executed at the same time with respect to each other. They describe analog Differential Algebraic Equations (DAE).

```
ARCHITECTURE behav of spring_tr IS
QUANTITY v ACROSS f THROUGH tr1 TO tr2; -- branch quantity declaration
QUANTITY s: DISPLACEMENT; -- parameter declaration
BEGIN
 BREAK s => s0; -- value assignment
 f == c*s; -- model equation
 v == s'DOT; -- model equation
END ARCHITECTURE behav;
```

Packages

A VHDL-AMS package contains subprograms, constant definitions, and/or type definitions used in one or more design units. Each package comprises a package declaration and a package body. The declaration represents the portion of the package that is visible outside of that package.

Package Declaration

Package declarations define the available types, constants, natures, subprograms, and attributes.

```
PACKAGE package_name IS
 …constant/type/subprogram/nature/attribute declarations
END package_name;
```

declarations	Information used in the model description. Declarations include data types, constants, signals, files, components, attributes, subprograms, and others.

```
PACKAGE omnicaster_package IS
 -- subprogram declarations
 FUNCTION b2bl (b: BIT) RETURN BOOLEAN;
 FUNCTION b2i (b: BIT; zv : INTEGER; ov : INTEGER) RETURN INTEGER;
END omnicaster_package;

PACKAGE misc IS
 TYPE short_integer IS range -100 TO 100; -- type declaration
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
CONSTANT K : REAL := 1.3806503e-23; -- constant declaration
--subtype declarations
SUBTYPE TEMPERATURE IS REAL TOLERANCE "DEFAULT_TEMPERATURE";
SUBTYPE HEAT_FLOW IS REAL TOLERANCE "DEFAULT_HEAT_FLOW";
-- nature declaration
NATURE THERMAL IS TEMPERATURE ACROSS HEAT_FLOW THROUGH THERM_
REF REFERENCE;
END misc;
```

Package Body

The package body defines the subprograms along with any internally-used constants and types.

```
PACKAGE BODY package_name IS
…subprogram bodies
END package_name;
```

subprogram_bodies	Specifies the subprograms declared in the package declaration.

```
PACKAGE BODY omnicaster_package IS
FUNCTION b2bl (b : BIT) RETURN BOOLEAN IS
BEGIN
IF (b = '1') THEN
RETURN TRUE;
ELSE
RETURN FALSE;
END IF;
END b2bl;
FUNCTION b2i (b : BIT; zv : INTEGER; ov : INTEGER) RETURN INTEGER IS
BEGIN
IF (b = '1') THEN
RETURN ov;
ELSE
RETURN zv;
END IF;
END b2i;
END omnicaster_package;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Package Visibility

The **LIBRARY** statement is used to make specified libraries such as the **IEEE** library visible in a model description. If the library file name is not consistent with VHDL-AMS naming conventions, you must define an alias. See ***Alias for File Names***. A **USE** statement can precede the declaration of any entity or architecture which is to utilize items from the package. If the **USE** statement precedes the **ENTITY** declaration, the package is also visible to the architecture. To make all items of a package visible to a design unit, precede the desired design unit with a **USE** statement accompanied by the **ALL** keyword. To make single items of a package visible, only the corresponding item is defined in the **USE** statement. This saves simulation (compilation) time, since all visible items in a **USE** statement must be loaded before simulation. See ***WORK Library***.
LIBRARY library_name; **USE** library_name.package_name.**ALL**; -- all items are visible **USE** library_name.package_name.item_name; -- one item is visible

library_name	Library name, for example **transformations**. You must omit the .**smd** extension appended to Twin Builder library files.
package_name	Name of the package in a library, for example **omnicaster_package**.
item_name	Name of a type, subprogram, or constant in the package.

LIBRARY TRANSFORMATIONS; -- make library transformations.smd visible **LIBRARY** VHDLAMS_TUTORIAL; -- make library vhdlams_turorial.smd visible **USE** TRANSFORMATIONS.omnicaster_package.**ALL**; -- all information available **USE** VHDLAMS_TUTORIAL.octal_conversions.int2oct; -- function int2oct available

VHDL-AMS Standard Packages and Types

Several standard packages are installed along with Twin Builder in the STD and IEEE libraries. These libraries can be found in the **Basic_VHDLAMS** folder in the **Project Manager** pane. Open the **Properties** dialog box for a package to view the VHDL-AMS source code in the VHDL Model Editor. The source code for the **MATH_REAL**, **NUMERIC_STD**, and **NUMERIC_BIT** packages are protected by an IEEE copyright and consequently cannot be viewed in the VHDL Model Editor.

To use any of the packages, the **LIBRARY** statement and the **USE** statement must be specified as follows:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
LIBRARY library_name;
USE library_name.package_name.ALL;
```

STD Library

The STD library has two packages: **STANDARD** and **TEXTIO**.

- **STANDARD** – Provides a number of types, subtypes, and functions. This package is different from all other packages in that it is always visible and need not be explicitly included within a model.
- **TEXTIO** – Provides data types and subprograms required for reading and writing ASCII files.

The TEXTIO package can be made visible within a model description in the following manner:

```
LIBRARY STD;
USE STD.TEXTIO.ALL;
```

The IEEE Library

The IEEE library houses a number of packages that can be classified into three categories:

- **Digital** – Packages for the simulation of digital designs.
- **Physical Domains** – Packages for the simulation of multidomain systems.
- **Math** – Packages for mathematical operations.

Packages for the Simulation of Digital Designs

The following table lists the packages available for use in digital designs:

Package Name	Functionality	Usage
STD_ LOGIC_ 1164	Defines the multi-value logic data type (**STD_LOGIC**) and associated operations.	**USE** IEEE.STD_ LOGIC_ 1164.**ALL**;
STD_ LOGIC_ ARITH	Synopsys package based on multi-value logic that defines types and basic arithmetic operations for representing integers.	**USE** IEEE.STD_ LOGIC_ ARITH.**ALL**;
STD_	Synopsys extension of the **STD_LOGIC_ARITH**	**USE** IEEE.STD_

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Package Name	Functionality	Usage
LOGIC_ SIGNED	package to handle multi-value logic values as signed integers.	LOGIC_ SIGNED.**ALL**;
STD_ LOGIC_ UNSIGNED	Synopsys extension of the **STD_LOGIC_ARITH** library to handle multi-value logic values as unsigned integers.	**USE** IEEE.STD_ LOGIC_ UNSIGNED.**ALL**;
NUMERIC_ STD	IEEE package based on multi-value logic that defines types and basic arithmetic operations for representing integers. This is similar to **STD_LOGIC_ARITH** and consequently should not be used together.	**USE** IEEE.NUMERIC_ STD.**ALL**;
NUMERIC_ BIT	IEEE package based on binary (BIT) data type that defines types and basic arithmetic operations for representing integers.	**USE** IEEE.NUMERIC_ BIT.**ALL**;

Note:

The definitions for signed and unsigned data types are available in the **NUMERIC_ STD** package, and in the Synopsys **SIGNED** and **UNSIGNED** packages. Consequently, the Synopsys packages cannot be used with the **NUMERIC_STD** package in any VHDL-AMS design.

Packages for the Simulation of Multidomain Systems

The following table lists the physical domain packages available for use in multidomain simulations:

Domain	Usage
Electrical	**USE** IEEE.ELECTRICAL_ SYSTEMS.**ALL**;
Mechanical • Translational • Rotational	**USE** IEEE.MECHANICAL_ SYSTEMS.**ALL**;
Thermal	**USE** IEEE.THERMAL_SYSTEMS.**ALL**;
Fluidic	**USE** IEEE.FLUIDIC_SYSTEMS.**ALL**;
Radiant	**USE** IEEE.RADIANT_SYSTEMS.**ALL**;

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The following table lists the common declarations and constants that are valid in all physical domains:

Package Name	Usage
Energy Systems	**USE** IEEE.ENERGY_SYSTEMS.**ALL**;
Fundamental Constants	**USE** IEEE.FUNDAMENTAL_ CONSTANTS.**ALL**;
Material Constants	**USE** IEEE.MATERIAL_CONSTANTS.**ALL**;

Packages for Mathematical Operations

Package Name	Usage
Math_Real (1076.2)	**USE** IEEE.MATH_ REAL.**ALL**;

Subprograms

Subprograms define algorithms for calculating particular functions of a model. They divide complex model descriptions into smaller sections.

There are two forms of subprograms: procedures and functions. A procedure call is a statement; a function call is an expression and returns a value. A subprogram has two parts:

- Declaration statements
- Sequential statements defining the behavior

Subprograms can use constants, variables, and signals as parameters. The parameters used within a subprogram are called the formal parameters, while the parameters passed into the function or procedure are called the actual parameters.

Note that a simultaneous procedural statement is different from a subprogram. A procedural statement describes analog behavior that occurs sequentially (rather than be solved simultaneously).

Procedures

A procedure defines a group of sequential statements executed when the procedure is called. A procedure can return any number of values (or no values) via its parameter list. A procedure call invokes the execution of the procedure body.
PROCEDURE procedure_name [(formal_parameters)] **IS**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 …declarations
BEGIN
 …sequential statements
END [PROCEDURE] [procedure_name];
…
 procedure_name[actual_parameters] -- procedure call
```

formal_parameters	Specifies a list of parameters (constants, signals, or variables with the mode in, out, or inout).
declarations	Declarations include data types, constants, signals, files, variables, attributes, and subprograms.
sequential statements	Statements that are executed in the order in which they appear, that define algorithms for the execution of a subprogram or process.

```
-- declaration with formal parameters int and bin
PROCEDURE int2bin (VARIABLE int: IN INTEGER; VARIABLE bin: OUT BIT_VECTOR) IS
 VARIABLE temp: INTEGER;
BEGIN -- start of sequential procedure statements
 temp := int;
 FOR i IN 0 TO (bin'LENGTH -1) LOOP
 IF (temp mod 2 = 1) THEN
 bin(i) := '1';
 ELSE
 bin(i) := '0';
 END IF;
 temp := temp/2;
 END LOOP;
END int2bin; -- end of sequential procedure statements
…
PROCESS -- continued model description
 VARIABLE in_var: INTEGER:=10;
 VARIABLE out_vec: BIT_VECTOR (1 to 8);
BEGIN
 -- procedure call with actual parameters int_var and out_vec
 int2bin (in_var, out_vec);
 WAIT;
END PROCESS;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Functions

A function defines a group of sequential statements executed when the function is called. A function returns a single value. Unlike procedures, functions are primarily used in expressions and only have inputs in their argument list. A function call invokes the execution of the function body.

```
[PURE | IMPURE] function function_name RETURN type_name
 …declarations
BEGIN
 …sequential statements
END [FUNCTION] [function_name];
…
 function_name[actual_parameter] -- funtion call
```

PURE | **IMPURE**	Pure functions return the same value each time they are called with the same values as actual parameters (default). Impure functions can return a different value each time they are called, even when multiple calls have the same actual parameter values.
formal_parameters	Specifies a list of parameters (constants, signals, or variables with the mode in, out, or inout).
type_name	Data type or data subtype name, for example **BIT**, **INTEGER**, **REAL**, … of the return value.
declarations	Declarations include data types, constants, signals, files, components, attributes, subprograms, and other information used in the model description.
sequential statements	Statements that are executed in the order in which they appear, that define algorithms for the execution of a subprogram or process.

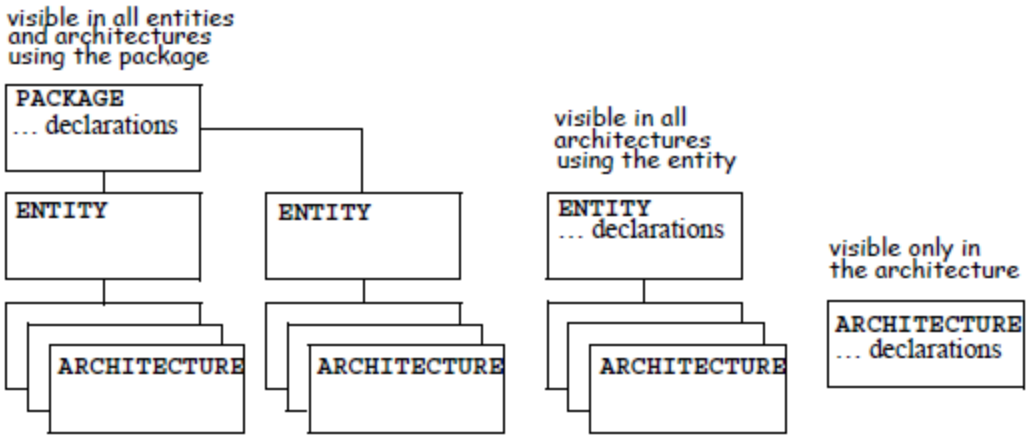
```
-- declaration with formal parameters i and length and data type of return value
FUNCTION i2bv (i: INTEGER; length: INTEGER) RETURN BIT_VECTOR IS
 VARIABLE bv: BIT_VECTOR(length-1 DOWNTO 0);
 VARIABLE temp: INTEGER;
BEGIN -- start of sequential procedure statements
 temp := i;
 FOR j IN 0 TO (bv'LENGTH-1) LOOP
 IF (temp mod 2 = 1) THEN
 bv(j) := '1';
 ELSE
 bv(j) := '0';
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 END IF;
 temp := temp/2;
 END LOOP;
 return bv;
END i2bv; -- end of sequential procedure statements
…
ARCHITECTURE bench OF entity_name IS
 SIGNAL in_var: INTEGER;
 SIGNAL out_vec: BIT_VECTOR (1 to 8);
 …
BEGIN
 in_var <= 10;
 -- function call with actual parameters in_var and value 8
 out_vec <= i2bv (in_var,8) AFTER 1ms;
END ARCHITECTURE bench;
```

Declarations

Declarations specify types, data objects, attributes, and components used in design units. Declarations are located in packages, entities, and architectures. The scope or visibility of a declaration depends on where they are used.



VHDL-AMS data types include scalar, composite, access, and file. VHDL-AMS supports different kinds of data objects for each data type. These data objects include constants, signals, variables, files, quantities, and terminals. VHDL-AMS includes a number of predefined data types, and allows user-defined data types as needed. A **TYPE** statement is used to declare a new data type. A **SUBTYPE** statement is used to constrain an existing type.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

TYPE Declarations

A type declaration defines a new data type.

- **Scalar** – Declares a type used to create enumeration, integer, physical, and floating point elements.
- **Composite** – Declares a type for creating array or record elements.
- **File** – Declares a type for creating file handles.

TYPE type_name **IS** scalar_type_definition; -- scalar type definition
TYPE type_name **IS** composite_type_definition; -- composite type definition
TYPE type_name **IS FILE OF** subtype_name; -- file type declaration

type_name	Data type or data subtype name, for example **BIT**, **INTEGER**, **REAL**, …

```
TYPE scalar_int1 IS RANGE -5 TO 5;
TYPE scalar_int2 IS RANGE 31 DOWNTO 0;
TYPE scalar_bit IS ('0', '1');
TYPE scalar_enum IS (red, green, blue);
TYPE scalar_boolean IS (TRUE, FALSE);
TYPE composite1 IS ARRAY (0 TO 31) OF BIT;
TYPE composite2 IS ARRAY (natural RANGE <>) OF INTEGER;
TYPE composite3 IS RECORD
 RE: REAL; IM: REAL;
END RECORD composite3;
TYPE composite4 IS ARRAY (INTEGER RANGE <>, INTEGER RANGE <>) OF REAL;
TYPE file_txt IS FILE OF INTEGER;
```

SUBTYPE Declaration

A subtype declaration defines a type derived from an existing type. **TYPE** creates a new type while **SUBTYPE** creates a type that is a constraint of an existing type. It is possible to assign values of objects belonging to subtypes to objects belonging to the base type.

SUBTYPE name **IS** type_name [constraint] [tolerance];

constraint	Specifies a range constraint for a scalar type with a **…** **RANGE … TO**(**DOWNTO**) **…** statement.

```
SUBTYPE electrical_units IS STRING (1 TO 20);
 VARIABLE var_names: electrical_units := "voltage, current";
SUBTYPE small_int IS INTEGER RANGE 0 TO 10; -- subtype with constraint definition
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

VARIABLE little : small_int := 4;

NATURE Declaration

A nature declaration defines a nature and its *across* and *through* quantities you can access through the model terminals. See ***Across and Through Quantities of Natures***.	
NATURE nature_name **IS** across_type_name **ACROSS** through_type_name **THROUGH** reference_terminal reference;	
nature_name	Name of a nature, for example ELECTRICAL, MECHANICAL, THERMAL, …
across_type_name through_type_name	Specifies the branch types of the nature, which define the type of branch quantities. Only terminals can represent a nature.
reference_terminal	Specifies the ground (zero) of the across type of a nature.
NATURE ELECTRICAL **IS** -- nature declaration of an electrical system voltage **ACROSS** current **THROUGH** electrical_ref **REFERENCE**; **NATURE** THERMAL **IS** -- nature declaration of a thermal system temperature **ACROSS** heat_flow **THROUGH** thermal_ref **REFERENCE**;	

Data Object Declarations

A data object holds a value of a specified type. Every data object belongs to one of six different classes, described in this section.

CONSTANT Declaration

A constant declaration assigns a value to an identifier of a given data type. The use of constants can improve the readability of VHDL-AMS code. The value of a **CONSTANT** object cannot be changed by any executable code after declaration.	
CONSTANT constant_name: type_name [expression];	
constant_name	List of constant names separated by a comma.
type_name	Data type or data subtype name, for example, **BIT**, **INTEGER**, **REAL**, …

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

expression	Expression that performs an arithmetic or logical computation by applying an operator to one or more operands (constant value).

```
CONSTANT Pi: REAL := 3.14159;
CONSTANT Half_Pi: REAL := Pi/2.0;
CONSTANT cycle_time: TIME := 11 ns;
CONSTANT N, N5: INTEGER := 5;
CONSTANT ctrl: BIT:= '1'; -- logic 1 constant
CONSTANT zero4: BIT_VECTOR(0 to 3) := "0000";
```

SIGNAL Declaration

A signal declaration defines an identifier as a signal object. A **SIGNAL** object holds a list of values, including the previous value, current value, and a set of possible future values that appear on the signal. A signal object has “digital properties”; this means that any change in a signal causes an event which can start a process or value assignment specified in the model description.

SIGNAL name_list: type_name [expression];

name_list	List of signal names separated by a comma.
type_name	Data type or data subtype name, for example, **BIT**, **INTEGER**, **REAL**, …
expression	Expression that performs an arithmetic or logical computation by applying an operator to one or more operands (default value).

```
SIGNAL a_bit : BIT := '0';
a_bit <= b_bit XOR '1';
SIGNAL clock : BIT := '0';
clock <= NOT clock AFTER 1 ms;
SIGNAL bool_sig : BOOLEAN := FALSE;
bool_sig <= TRUE WHEN clock = '1' ELSE FALSE;
SIGNAL gate_delay : TIME := 10 ns;
```

VARIABLE Declaration

A variable declaration defines an identifier as a variable object. A variable can be of any scalar or aggregate data type and is updated immediately when an assignment statement is executed. Variables can be declared only within subprograms, procedural statements, or

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

process statements.	
VARIABLE variable_name: type_name [expression];	
variable_name	List of variable names separated by a comma.
type_name	Data type or data subtype name, for example, **BIT**, **INTEGER**, **REAL**, …
expression	Expression that performs an arithmetic or logical computation by applying an operator to one or more operands (default value).

```
clock <= NOT clock AFTER 10 us;
PROCESS (clock)
 VARIABLE num_events : INTEGER := 0;
BEGIN
 num_events := num_events + 1;
END PROCESS;
```

FILE Declaration

A file declaration defines an identifier as a file object.	
FILE file_name: type_name [[**OPEN** file_open_kind] **IS** file_logical_name];	
file_ name	Name of the created file.
type_ name	Data type name, for example **TEXT**, …
file_ open_ kind	Specifies the access mode for the file object: **READ_MODE**, **WRITE_MODE**, **APPEND_MODE**. These operation modes are defined in the **STD.TEXTIO** package.
file_ logical_ name	Defines the file name. The name is a string in quotes and its syntax must conform to the operating system where the VHDL-AMS model will be simulated.

```
FILE cycle: TEXT OPEN READ_MODE IS "cyc_mph_manhattan_tab.txt";
…
WHILE NOT ENDFILE (cycle) LOOP
 READLINE (cycle,buf);
END LOOP;
```

QUANTITY Declaration

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

A quantity declaration defines one or more identifiers as quantity objects. A quantity object is specified by its type and a default value. Quantities can be declared in both entity and architecture declaration placeholders. Quantities can be classified as PORT, FREE and BRANCH quantities depending on where they are declared. PORT quantities appear in the **ENTITY** declaration, BRANCH quantities appear in the **ARCHITECTURE** declaration and specify the across and through values for terminals of a particular nature, and FREE quantities are declared as analog values in the **ARCHITECTURE** declaration.

QUANTITY name_list : real_type_name [expression];
QUANTITY [across_aspect] [through_aspect] terminal_aspect;

name_list	List of quantities separated by a comma.
real_type_name	Real type or real subtype name, for example, **REAL**, **VOLTAGE**, **TORQUE**...
across_aspect	Identifier that serves as the across quantity for the specified terminals.
through_aspect	Identifier that serves as the through quantity for the specified terminals.
terminal_aspect	Positive and negative terminal names.

```
ARCHITECTURE admittance OF load IS
 QUANTITY v ACROSS i THROUGH p TO m;
 QUANTITY ctrl_ramp: REAL := 0.0;
 QUANTITY ctrl_qty: REAL := 0.0;
BEGIN
-- …
END ARCHITECTURE admittance;
```

TERMINAL Declaration

A terminal declaration declares a terminal, as well as the reference quantity and contribution quantity of the terminal.

TERMINAL name_list: nature_name;

name_list	List of terminals separated by a comma.
nature_name	Name of the conserved physical domain.

TERMINAL terminal_e1 : ELECTRICAL; --Local Electrical Terminal
TERMINAL terminal_t1 : THERMAL; --Local Thermal Terminal

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Other Declarations

An attribute declaration defines an attribute name and its type. An attribute specification assigns a value to the attribute.

A component declaration defines a component's interface and is typically placed in an **ARCHITECTURE** or **PACKAGE** declaration. The component or instances of the component are related to a design entity in a library using a configuration.

ATTRIBUTE Declaration

An attribute declaration defines an attribute name and its type. An attribute specification assigns a value to the attribute. See ***Predefined Attributes***.	
ATTRIBUTE attribute_name: type_name; -- declaration **ATTRIBUTE** attribute_name **OF** name: entity_class **IS** expression; -- specification	
attribute_name	Name of the attribute.
type_name	Data type name, for example, **BIT**, **INTEGER**, **REAL**, …
name	Identifier of the existing object.
entity_class	Specifies the name of the entity class, for example, **TYPE**, **SIGNAL**, **FILE**, …
expression	Expressions to perform an arithmetic or logical computation by applying an operator to one or more operands.
ATTRIBUTE UNIT **OF** VOLTAGE: **SUBTYPE IS** "Volt"; -- attribute declaration **ATTRIBUTE** SYMBOL **OF** RESISTANCE: **SUBTYPE IS** "OHM";	

COMPONENT Declaration

A component declaration defines a component's interface and is typically placed in an **ARCHITECTURE** or **PACKAGE** declaration. The component or instances of the component are related to a design entity in a library using a configuration.	
COMPONENT component_name [**IS**] [local_generic_clause] [local_port_clause] **END COMPONENT** [component_name];	
component_name	Name of the component.
local_generic_clause	Specifies static information to be communicated to a model from its environment in this form: *variable_name* **:** *variable_type* **:=** *value***;**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

local_port_clause	Specifies dynamic information communicated to a model from its environment in this form: *variable_name* **:** *port_mode variable_type***;**

```
ENTITY inv4 IS
 GENERIC (TP_LH, TP_HL: TIME);
 PORT (I_4 : IN BIT_VECTOR(3 DOWNTO 0);
 Y_4 : OUT BIT_VECTOR(3 DOWNTO 0));
END ENTITY inv4;

ARCHITECTURE inv4_struct OF inv4 IS
 COMPONENT inv
 GENERIC (TP_LH, TP_HL: TIME);
 PORT (I1 : IN BIT;
 Y1 : OUT BIT);
 END COMPONENT;
 SIGNAL INV_OUT : BIT_VECTOR(3 DOWNTO 0);
BEGIN
-- instantiation of the inv component
 INV_1 : inv GENERIC MAP (TP_LH,TP_HL) PORT MAP (I_4(0),Y_4(0));
 INV_2 : inv GENERIC MAP (TP_LH,TP_HL) PORT MAP (I_4(1),Y_4(1));
 INV_3 : inv GENERIC MAP (TP_LH,TP_HL) PORT MAP (I_4(2),Y_4(2));
 INV_4 : inv GENERIC MAP (TP_LH,TP_HL) PORT MAP (I_4(3),Y_4(3));
END inv4_struct;
```

Concurrent Statements

Concurrent statements are included within **ARCHITECTURE** bodies and **BLOCK** statements, representing concurrent digital behavior within the modeled design unit. These statements are executed in an asynchronous manner, with no defined order, modeling the overall behavior or structure of a design.

BLOCK Statement

A block statement groups related concurrent statements. The order of the concurrent statements does not matter, because all statements are always executed together.

If a guard expression appears after the reserved word **BLOCK**, a Boolean variable *GUARD* is defined and set to the Boolean value of the guard expression. *GUARD* can then be tested within the block, to perform selected signal assignments or other statements only when the guard condition evaluates to *TRUE*.

```
[label_name:]
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

BLOCK [(guard expression)]
…local declarations
BEGIN
…concurrent statements
END BLOCK [label_name];

guard expression	Defines the value of the signal. The type of the expression must be BOOLEAN.
local declarations	Declarations include data types, constants, signals, files, components, attributes, subprograms, and other information used in the model description.
concurrent statements	Digital statements that are executed asynchronously with respect to each other, that describe a design unit in one or more modeling styles such as dataflow, structural, and behavior styles.

```
-- D Latch: Transfer D input to Q output when Enable = '1'
B1:
BLOCK (Enable = '1')
BEGIN
 Q <= guarded D AFTER 5ns;
END BLOCK B1;
```

PROCESS Statement

A **PROCESS** statement contains sequential statements but is itself a concurrent statement within an architecture. An independent sequential process represents the behavior of some portion of a design. The body of a process is a list of sequential statements. See ***Processes***.

The sequential statements in the process are executed in order, commencing with the beginning of simulation. After the last statement of a process has been executed, the process repeats from the first statement, and continues to repeat until suspended. Processes can be suspended with a **WAIT** statement. The **WAIT** statement is a versatile statement that lets you suspend a process for a specific period of time, Boolean condition to occur, or/and an event to occur on a signal. The optional sensitivity list is equivalent to providing a **WAIT ON** statement and causes the process to be suspended. A process cannot have both a sensitivity list as well as a **WAIT ON** statement. See ***Wait Statement***.

```
[label_name:]
PROCESS [(sensitivity_list)]
 …local declarations
BEGIN
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

…sequential statements **END PROCESS** [label_name];	
sensitivity_list	List of signal names separated by a comma. The list represents a set of signals to which the **WAIT** statement within the process is sensitive. If no sensitivity list is defined, a **WAIT** statement within the process must be specified.
local declarations	Declarations declare data types, constants, signals, files, components, attributes, subprograms, and other information used in the model description.
sequential statements	Statements that are executed in the order in which they appear, that define algorithms for the execution of a subprogram or process.

clock <= **NOT** clock **AFTER** 10 us;
PROCESS (clock) -- sensitivity list consists of clock
VARIABLE num_events : INTEGER := 0;
BEGIN
num_events := num_events + 1;
END PROCESS;	

Concurrent Procedure Call Statement

A concurrent procedure call used in an **ARCHITECTURE** or a **BLOCK** statement, invokes an externally defined subprogram with parameters passed to it as necessary. See ***Procedures***.	
[label_name:] procedure_name [actual_parameter_list];	
actual_parameters	Values of these parameters are transferred to the procedure for the formal parameters (the parameters specified in the procedure declaration).
-- procedure ReadMem procedure can be defined in a package and used in many places ReadMem (DataIn, DataOut, RW, Clk);	

Concurrent ASSERT Statement

A concurrent **ASSERT** statement checks a condition (occurrence of an event) and provides a report if the condition is *FALSE*. A severity level can be specified to generate several types

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

of messages.	
ASSERT (condition) [**REPORT** string] [**SEVERITY** severity_level];	
condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*). If the condition expression is *FALSE* (indicating the assertion failed), the text specified in the string expression displays.
string	Defines the text string which appears in the message. String literals are one-dimensional arrays of characters enclosed in double quotation marks (” ”).
severity_level	Defines how the message appears and the effect on simulation. • **NOTE** – Message output in the message window. • **WARNING** – Message output in a dialog box and a request for user input to break or continue simulation. • **ERROR/FAILURE** – Message output in the message window and termination of simulation.
ASSERT (TS > 0.0) **REPORT** "Sample Time must be specified" **SEVERITY** ERROR; **ASSERT** (UL >= LL) **REPORT** "Upper Limit must be greater than Lower limit" **SEVERITY** WARNING; **ASSERT** (arg > 100.0) **REPORT** "Invalid Function argument " & REAL'IMAGE(arg) & " at time " & TIME'IMAGE (now) & "." **SEVERITY** INFO;	

Concurrent SIGNAL Assignment Statement

A concurrent signal assignment statement, one of the most common signal assignments, represents a process that assigns values to signals and specifies logical relationship between different signals. There is no significance to the order in which the assignments appear in the description. There are two forms: conditional signal assignment and selected signal assignment statements. See ***SIGNAL Assignment Statement***.	
[label_name:] target_signal <= var; -- signal assignment; target_signal <= var1 WHEN condition ELSE var2; -- conditional signal assignment target_signal <= var2 WHEN condition var3 WHEN OTHERS; -- selected signal assignment target_signal <= var4 AFTER 5ns; -- signal assignment with delay	
target_signal	Name of the signal which obtains the value.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

conditional_signal_ assignment	Special form of signal assignment that assigns values only if a sequence of related conditions are true.
selected_signal_assignment	Special form of signal assignment that assigns values only if a sequence of related conditions are true but differs in that the input conditions specified have no implied priority.
condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*).

```
ARCHITECTURE behav OF test IS
SIGNAL A: BIT :='1';
SIGNAL B: BIT :='0';
BEGIN
 A <= B;
 B <= '1'; -- value of B=1 value of A=1
END;
```

Component Instantiation Statement

The component instantiation statement instantiates (creates an instance of) predefined components within an architecture. Each such component is first defined in the declaration section of that architecture, then instantiated one or more times in the body of the architecture. See ***WORK Library***.

```
label_name: ENTITY [library_name.]entity_name [(architecture_name)]
 [GENERIC MAP (generic_map_list)]
 [PORT MAP (port_map_list)];
```

generic_map_list	Specifies a list of generics (separated by a comma) which associate values with the formal generics in the corresponding component declaration or entity interface.
port_map_list	Specifies a list of ports (separated by a comma) which associate signals, quantities, and/or terminals with the formal ports in the corresponding component declaration or entity interface.

```
LIBRARY vhdlams_tutorial;
LIBRARY IEEE;
USE vhdlams_tutorial.res; -- model res of the tutorial_vhdlams library is declared
USE IEEE.ELECTRICAL_SYSTEMS.ALL;
ENTITY bat_multi IS
 GENERIC(
 factor : REAL := 1.0;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 v_init : VOLTAGE := 12.0);
 PORT(
 TERMINAL p,m : ELECTRICAL;
 QUANTITY v_out : OUT VOLTAGE := 0.0);
END ENTITY bat_multi;
ARCHITECTURE struct OF bat_multi IS
 CONSTANT ri: RESISTANCE := 1.0e-2;
 CONSTANT fc: CAPACITANCE := 60.0;
 CONSTANT rd: RESISTANCE := 4.0e-2;
 CONSTANT sc: CAPACITANCE := 2.0e4;
 TERMINAL t1,t2 : ELECTRICAL;
 QUANTITY v ACROSS p TO m;
BEGIN
 fc1: ENTITY WORK.cap(behav) -- instantiation of a component in the WORK lib
 GENERIC MAP (c_nom => fc*factor, v_init => v_init)
 PORT MAP (p => t1, m => m);
 sc1: ENTITY WORK.cap(behav) -- instantiation of a component in the WORK lib
 GENERIC MAP (c_nom => sc*factor, v_init => v_init)
 PORT MAP (p => t2, m => m);
 ri1: ENTITY res(behav) -- instantiation of a component declared in the entity
 GENERIC MAP (r_nom => ri)
 PORT MAP (p => p, m => t1);
 rd1 : ENTITY res(behav) -- instantiation of a component declared in the entity
 GENERIC MAP (r_nom => rd)
 PORT MAP (p => t1, m => t2);
 v_out == v;
END ARCHITECTURE struct;
```

Concurrent BREAK Statement

A **BREAK** statement explicitly indicates the occurrence of discontinuities in a VHDL-AMS description. The concurrent **BREAK** statement represents a process containing a **BREAK** statement.

[label_name:]
BREAK [break_list] [sensitivity_clause] **WHEN** condition;

break_list	Specifies a list of break elements (separated by a comma) which can cause a discontinuity.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

sensitivity_clause	Specifies a list of signals to which the **BREAK** statement is sensitive.
condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*).

```
ARCHITECTURE behav OF LMT IS
 SIGNAL lower_crossing : BOOLEAN := FALSE;
 SIGNAL upper_crossing : BOOLEAN := FALSE;
BEGIN
 BREAK ON lower_crossing;
 BREAK ON upper_crossing;
…
END ARCHITECTURE;
```

Sequential Statements

Sequential statements appear in process statements and in subprograms (procedures and functions), representing sequential behavior within the modeled design unit. These statements are executed in the order in which they appear in the model.

WAIT Statement

A **WAIT** statement suspends subprogram execution until a signal changes, a condition becomes *TRUE*, or a defined time period has elapsed. You can also use combinations of these.

WAIT [**ON** sensitivity_list] [**UNTIL** condition] [**FOR** time_expression];

sensititvity_list	List of signals which can cause an event.
condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*).
time_expression	Specifies a time expression which evaluates to a *TIME* value.

```
WAIT ON INPUT;
WAIT UNTIL ctrl > 1.5;
WAIT FOR TDELAY*unit_time;
```

ASSERT Statement

A sequential **ASSERT** statement checks a condition and provides a report if the condition is

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

not *TRUE*. You can specify a severity level to generate several types of messages.	
ASSERT (condition) [**REPORT** string] [**SEVERITY** severity_level];	
condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*). If the condition expression is *FALSE* (indicating the assertion failed), the text specified in the string expression displays.
string	Defines the text string which appears in the message. String literals are one-dimensional arrays of characters enclosed in double quotation marks (” ”).
severity_level	Defines how the message appears and the effect for simulation. • **NOTE** – Message output in the **Message Manager** pane. • **WARNING** – Message output in a dialog box and a request for user input to break or continue simulation. • **ERROR/FAILURE** – Message output in the **Message Manager** pane and termination of simulation.

```
ENTITY sequential_assert IS
END;
ARCHITECTURE behav OF sequential_assert IS
SIGNAL clk:bit;
BEGIN
 clk<=not clk AFTER 1 ns;
 PROCESS(clk)
 variable a:integer := 0;
 BEGIN
 a:=a+1;
 ASSERT a/=1 REPORT "a!=1" SEVERITY FAILURE;
 ASSERT a/=2 REPORT "a!=2" SEVERITY WARNING;
 ASSERT a/=3 REPORT "a!=3" SEVERITY ERROR;
 ASSERT a<10 REPORT "a==" & INTEGER'IMAGE(a) SEVERITY NOTE;
 END PROCESS;
END;
```

SIGNAL Assignment Statement

A signal assignment statement assigns a waveform to one signal driver (edits the event queue). Signal assignments are always performed at the end of a process. See ***Concurrent SIGNAL Assignment Statement*** and ***Signal Assignments with Delay***.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

[label_name:] target_signal <= [delay_mechanism] source_signal	
target_signal	Name of the signal which obtains the value.
delay_mechanism	Specifies a delay for signal assignment after one of the reserved words: **TRANPORT**, **REJECT**, or **INERTIAL**.
source_signal	Name of the signal which provides the value.

```
SIGNAL A: BIT :='1';
SIGNAL B: BIT :='0';
PROCESS
 BEGIN
 A <= B;
 B <= '1'; -- value of B=1 value of A=0
 WAIT;
END PROCESS;
```

VARIABLE Assignment Statement

A variable assignment updates a process, procedure, or function variable with the value of an expression. The update takes effect immediately, unlike signal assignments where there is at least a delta delay before the update is reflected.

[label_name:] variable := expression;	
variable_name	Name of the variable.
expression	Expression that performs an arithmetic or logical computation by applying an operator to one or more operands.

```
clock <= NOT clock AFTER 10 us;
PROCESS (clock)
 VARIABLE num_events : INTEGER := 0;
BEGIN
 num_events := num_events + 1;
END PROCESS;
```

Procedure Call Statement

A procedure call statement invokes an externally-defined subprogram in the same manner

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

as a concurrent procedure call. A sequential procedure call statement differs from a concurrent procedure call in that it is placed in a process, procedure, or function, and is executed in the order in which it appears. See ***Subprograms***.
[label_name:] procedure_name [(actual_parameter)]

formal_parameters	Specifies a list of parameters (constants, signals, or variables with the mode **in**, **out**, or **inout**).

```
-- procedure call with actual parameters int_var and out_vec
 int2bin (in_var, out_vec);
```

IF Statement

An **IF** … **THEN** … **ELSE** statement performs a sequence of statements depending on the defined condition. **ELSIF** and **ELSE** clauses are optional.

```
[label_name:]
IF condition THEN
 …sequential statements
[{ELSIF condition THEN
 …sequential statements}]
[ELSE
 …sequential statements]
END IF [label_name];
```

condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*). If the condition expression is *TRUE*, the corresponding statements after the condition are executed.
sequential statements	Statements that are executed in the order in which they appear; they define algorithms for the execution of a subprogram or process.

```
pwm_ctrl:
PROCESS (cond1,cond2)
BEGIN
 IF (cond1) THEN
 ctrl_sig <= 0.0;
 ELSIF (cond2 AND (NOT cond1)) THEN
 ctrl_sig <= 1.0;
 END IF;
END PROCESS pwm_ctrl;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

CASE Statement

A **CASE** statement selects one of a number of alternative sequences of statements for execution based on the value of an expression. The choices must be constants of the same discrete type as the expression. Case choices can be expressions or ranges. Case statements must also include all possible values of the control expression. Use the **OTHERS** expression to guarantee that all conditions are covered.

```
[label_name:]
CASE control_expression IS
 WHEN choice1 =>
 …sequential statements
 WHEN choice2 =>
 …sequential statements
 WHEN OTHERS =>
 …sequential statements
END CASE [LABEL];
```

control_expression	Specifies a value that selects one statement sequence among the list of alternatives. The expression must be of a discrete type, or of a one-dimensional array.
choice	A choice specifies the value of the control expression for which the alternative is chosen. Each choice in a case statement alternative must be of the same type as the expression.
sequential statements	Statements that are executed in the order in which they appear, that define algorithms for the execution of a subprogram or process.

```
CASE J_K IS
 WHEN "11" =>
 state <= NOT state;
 WHEN "10" =>
 state <= '1';
 WHEN "01" =>
 state <= '0';
 WHEN OTHERS =>
 NULL;
END CASE;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

LOOP Statements

The **WHILE** and **FOR** loop statements control the repetition of sequentially executed statements. Loop termination statements allow termination of one iteration, loop, or procedure.

NEXT [**WHEN** condition]; -- end current loop iteration

EXIT [**WHEN** condition]; -- exit innermost loop entirely

```
[label_name:]
LOOP
 …sequential statements -- use exit statement to leave the loop
END LOOP [label];

[label_name:]
FOR variable IN start_val TO end_val LOOP
 …sequential statements
END LOOP [LABEL];

[label_name:]
WHILE condition LOOP
 …sequential statements
END LOOP [LABEL];
```

sequential statements	Statements that are executed in the order in which they appear, that define algorithms for the execution of a subprogram or process.
variable/start_ val/end_val	Implicit index variables of the loop. Index variables need not be declared separately.
condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*). If the condition is *TRUE*, the corresponding statements after the condition execute.

```
LOOP
 input_something;
 exit when end_file;
END LOOP;

FOR I IN 1 TO 10 LOOP
 AA(I) := 0;
END LOOP;

WHILE NOT end_file LOOP
 input_something;
END LOOP;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
PROCESS
 VARIABLE buf : LINE;
 VARIABLE t_val_old, t_val_new, t_val, s_val : REAL := 0.0;
BEGIN
 WHILE NOT ENDFILE(cyc) LOOP
 READLINE(cyc,buf);
 WAIT FOR 4ms;
 END LOOP;
 WAIT;
END PROCESS;
```

NEXT Statement

A **NEXT** statement causes the next iteration in a loop.

[label_name1:]
NEXT [label_name2] [when condition];

condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*). If the condition is *TRUE*, the next iteration in the loop executes.

NEXT;
NEXT outer_loop;
NEXT WHEN A>B;
NEXT this_loop **WHEN** C=D **OR** done; -- done is a BOOLEAN variable

EXIT Statement

An **EXIT** statement causes the immediate termination of the loop.

[label_name:]
EXIT [label2] [**WHEN** condition];

condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*). If the condition is *TRUE*, the loop terminates.

EXIT;
EXIT outer_loop;
EXIT when A>B;
EXIT this_loop **WHEN** C=D **OR** done; -- done is a BOOLEAN variable

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

RETURN Statement

A **RETURN** statement terminates a subprogram. A function definition requires a return statement to specify the return value. In a procedure definition, a **RETURN** statement is optional.

[label_name:]
RETURN [expressions];

expression	Expression that performs an arithmetic or logical computation by applying an operator to one or more operands.

```
RETURN; -- from somewhere in a procedure
RETURN a+b; -- returned value in a function
```

NULL Statement

A **NULL** statement explicitly states that no action is required. It is often used in **CASE** statements because all choices must be covered, even if some choices are ignored.

[label_name:]
NULL;

```
CASE J_K IS
 WHEN "11" =>
 state <= NOT state;
 WHEN "10" =>
 state <= '1';
 WHEN "01" =>
 state <= '0';
 WHEN OTHERS =>
 NULL;
END CASE;
```

BREAK Statement

A **BREAK** statement explicitly indicates discontinuities in a VHDL-AMS description. The sequential **BREAK** statement represents a process containing a **BREAK** statement.

The execution of a **BREAK** statement notifies the analog solver that it must determine the discontinuity augmentation set for the next analog solution point. It can also specify reset and

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

initial values. The effect is conditional if the statement includes a condition.	
[label_name:] **BREAK** [break_list] [when condition];	
break_list	Specifies a list of break elements (separated by a comma) which can cause a discontinuity.
condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*).

```
LIBRARY IEEE;
USE IEEE.MATH_REAL.ALL;
ENTITY sequential_break1 IS
END ENTITY sequential_break1;
ARCHITECTURE behav OF sequential_break1 IS
 SIGNAL xs1,xs2 : BOOLEAN := FALSE;
 SIGNAL aVal : REAL := 0.0;
BEGIN
 xs1 <= NOT xs1 AFTER 1 ms;
 xs2 <= NOT xs2 AFTER 2.5 ms;
 PROCESS (xs1, xs2)
 BEGIN
 BREAK aVal => 4.0 WHEN xs2;
 BREAK aVal => 2.0 WHEN xs1;
 END PROCESS;
END ARCHITECTURE behav;
```

Simultaneous Statements

Simultaneous statements appear in the architecture body of a model and are placed in the same parts of a VHDL-AMS description as concurrent statements. Simultaneous statements are used to express Differential Algebraic Equations (DAE) that together with implicit equations describe the analog behavior of a model. This section describes the five types of simultaneous statements.

Simple Simultaneous Statement

A simple simultaneous statement specifies expressions that constrain the values of quantities by the analog solver.
[label_name:] expression == expression;

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

expressions	Expressions that perform an arithmetic or logical computation by applying an operator to one or more operands.

TERMINAL plus, minus : ELECTRICAL;
QUANTITY voltage **ACROSS** current **THROUGH** plus **TO** minus;
QUANTITY power : REAL;
power == voltage * current; -- power calculation

Simultaneous IF Statement

A simultaneous **IF** statement specifies analog behavior of a system based on a set of conditions. A simultaneous **IF** statement differs from a sequential **IF** statement in that the simultaneous statement syntax is "**IF-USE-END USE**" while the sequential statement syntax is "**IF-THEN-END IF**".

[label_name:]
IF condition **USE**
…simultaneous_statements
[{ **ELSIF** condition **USE**
…simultaneous_statements }]
[**ELSE**
…simultaneous_statements]
END USE [label_name];

condition	Specifies an expression which evaluates to a BOOLEAN value (*TRUE* or *FALSE*). If the condition expression is *TRUE*, the corresponding statements after the condition execute.
simultaneous statements	Statements that execute at the same time with respect to each other; they describe Analog Differential Algebraic Equations (DAE).

IF (sw_on) **AND** (CTRL > 0.0)) **USE**
v == 0.0;
ELSE
i == 0.0;
END USE;

Simultaneous CASE Statement

A simultaneous **CASE** statement specifies analog behavior by selecting one of a number of

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

alternatives based on the value of an expression. A simultaneous **CASE** statement differs from a sequential **CASE** statement in that the simultaneous statement syntax is "**CASE-USE-END CASE**" while the sequential statement syntax is "**CASE-IS-END CASE**".

[label_name:]
CASE control_expression **USE**
WHEN choices =>
…simultaneous_statements
{ when choices =>
…simultaneous_statements}
END CASE [**LABEL**];

control_expression	Specifies a value that selects one statement sequence among the list of alternatives. The expression must be of a discrete type, or of a one-dimensional array.
choice	A choice specifies the value of the control expression for which the alternative is chosen. Each choice in a case statement alternative must be of the same type as the expression.
simultaneous statements	Statements that execute at the same time with respect to each other; they describe Analog Differential Algebraic Equations (DAE).

```
LIBRARY IEEE;
USE IEEE.MATH_REAL.ALL;
USE IEEE.ELECTRICAL_SYSTEMS.ALL;

ENTITY simultaneous_case IS
END ENTITY simultaneous_case;

ARCHITECTURE behav OF simultaneous_case IS
 TERMINAL n1, n2 : ELECTRICAL;
 QUANTITY vin ACROSS iin THROUGH n1 TO electrical_ref;
 QUANTITY vout ACROSS iout THROUGH n2 TO electrical_ref;
 CONSTANT Amp : REAL := 1.0;
 CONSTANT a : REAL := 1.0E3;
 CONSTANT f : REAL := 1.0E3;
 SIGNAL clk : INTEGER := 0;
BEGIN
 clk <= clk+1 AFTER 1ms;
 vout == vin'SLEW(1.0e38,-1.0e38);
 CASE clk USE
 WHEN 0 => vin == Amp*1.0*sin(2.0*math_pi*f*15.0*NOW);
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 WHEN 1 => vin == Amp*2.0*sin(2.0*math_pi*f*15.0*NOW);
 WHEN 2 => vin == Amp*3.0*sin(2.0*math_pi*f*15.0*NOW);
 WHEN 3 => vin == Amp*4.0*sin(2.0*math_pi*f*15.0*NOW);
 WHEN 4 => vin == Amp*5.0*sin(2.0*math_pi*f*15.0*NOW);
 WHEN others => vin == Amp*6.0*sin(2.0*math_pi*f*15.0*NOW);
 END CASE;
END ARCHITECTURE simple;
```

Simultaneous PROCEDURAL Statement

A simultaneous procedural statement provides a sequential notation for expressing differential and algebraic equations. Procedural statements are included within the architecture of a model and are not invoked with a calling mechanism.

```
[label_name:]
PROCEDURAL [IS]
 …declarations
BEGIN
 …sequential_statements
END PROCEDURAL [label_name];
```

declarations	Declarations include data types, constants, signals, files, variables, attributes, subprograms, and other information used in the model description.
sequential statements	Statements that execute in the order in which they appear; they define algorithms for the execution of a subprogram or process.

```
ENTITY fktarccos IS
 GENERIC (TS : REAL := 0.0);
 PORT (QUANTITY INPUT : IN REAL;
 QUANTITY VAL : OUT REAL);
END ENTITY fktarccos;
ARCHITECTURE behav OF fktarccos IS
 QUANTITY temp_val : REAL := 0.0;
BEGIN
 PROCEDURAL
 BEGIN
 IF (INPUT>-1.0) AND (INPUT<1.0) THEN
 temp_val := arccos(INPUT);
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 ELSIF (INPUT = -1.0) THEN
 temp_val := MATH_PI;
 ELSE
 temp_val := 0.0;
 END IF;
 END PROCEDURAL;
 VAL == temp_val'ZOH(TS);
END behav;
```

Simultaneous NULL Statement

A simultaneous **NULL** statement explicitly specifies that no action be performed.

```
[label_name:]
NULL;
```

```
LIBRARY IEEE;
USE IEEE.ELECTRICAL_SYSTEMS.ALL;
ENTITY two_port IS
 GENERIC
 (param : REAL := 1.0;
 mod_type : INTEGER := 0);
 -- RESISTOR(0); CAPACITOR(1);
 -- INDUCTOR(2)
 PORT
 (TERMINAL p,m : ELECTRICAL);
END ENTITY two_port;
ARCHITECTURE behav OF two_port IS
 QUANTITY v ACROSS i THROUGH p TO m;
BEGIN
 CASE mod_type USE
 WHEN 0 => v == i*param;
 WHEN 1 => i == param*v'dot;
 WHEN 2 => v == param*i'dot;
 WHEN OTHERS => NULL;
 END CASE;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

END ARCHITECTURE behav;

Identifiers, Literals, and Expressions

Identifiers and literals are operands representing values of constants, variables, functions, and so on in expressions. Expressions perform arithmetic or logic computations by applying an operator to one or more operands.

Identifiers

An identifier is the name of a constant, variable, function, signal, entity, port, subprogram, or parameter and returns that object's value to an operand. Identifiers in VHDL-AMS must begin with a letter, and can comprise any combination of letters, digits, and underscores. Identifiers must not end with an underscore and must not include two successive underscore character. Also no space is allowed within an identifier since a space is a separator. An indexed identifier is the name of one element of an array variable or signal. The expression must return a value within the array's index range. The value returned to an operator is the specified array element.
letter { [underscore] letter_or_digit } -- identifier letter { [underscore] letter_or_digit } (expressions) -- indexed identifier
Voltage1, power_dissipated, product_of_sums voltage(2), current(3+1)

Literals

A literal (constant) operand can be a numeric, a character, an enumeration, or a string literal.
There are two forms of numeric literals: integer and real literals. Integer literals represent a whole number. Real literals can represent fractional numbers and always include a decimal point preceded and followed by at least one digit. Both types of numeric literals can use exponential notation and can be expressed in a base. A decimal literal is written in base 10 and a based literal is written in a base from 2 to 16 and is composed of the base number, a number sign (#), the value in the given base, and another number sign (#).
base#digits# -- base must be a decimal number 2#101# -- decimal 5 16#AA# 16#f.1f#E+2 -- floating, exponent is decimal

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Character literals are single characters enclosed in single quotation marks, for example 'a'. Character literals are used both as values for operators and in defining enumerated types, such as CHARACTER and BIT.

Enumeration literals are values of enumerated types. The two kinds of enumeration literals are character literals and identifiers. Character literals are described earlier. Enumeration identifiers are those listed in an enumeration type definition. If two enumerated types use the same literals, those literals are overloaded.

TYPE ENUM_1 **IS** (AAA, BBB, 'A', 'B', ZZZ);
TYPE ENUM_2 **IS** (CCC, DDD, 'C', 'D', ZZZ);
AAA -- Enumeration identifier of type ENUM_1
'B' -- Character literal of type ENUM_1
CCC -- Enumeration identifier of type ENUM_2
'D' -- Character literal of type ENUM_2
ENUM_1'(ZZZ) -- Qualified because overloaded

String literals are one-dimensional arrays of characters enclosed in double quotation marks (” ”). The two kinds are:

- Character strings, which are sequences of characters in double quotation marks, for example, ”ABCD”.
- Bit strings, which are similar to character strings but represent binary (B), octal (O), or hexadecimal (X) values. For example, B”1101”. The digits in a bit string literal value can be separated with underscores (_) for readability.

x"ffe"
-- 12-bit hexadecimal value, digits 0 to 9 and A to F
-- each value represents four BITs in the generated bit vector (array)
o"777"
-- 9-bit octal value, digits 0 to 7
-- each octal digit represents three BITs in the generated bit vector (array)
b"1111_1101_1101"
-- 12-bit binary value, digits 0 or 1, each bit in the string represents
-- one BIT in the generated bit vector (array)

Arithmetic and Logical Expressions

Expressions in VHDL-AMS are similar to those of most high-level languages. Data elements must be of the same type, or subtypes of the same base type. Expressions perform arithmetic or logical computations by applying an operator to one or more operands. Operators specify the computation to perform, operands are the data for the computation.

Logical	• Predefined operators for types BIT and BOOLEAN.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	AND | OR | NAND | NOR | XOR | XNOR | NOT
Relational	• Predefined operators for all types except files. = | /= • Predefined operators for scalar and discrete array types. < | <= | > | >= Relational operators include tests for equality (=), inequality (/=), and ordering of operands (<, <=, >, >=). The operands of each relational operator must be of the same type. The result type of each relational operator is the predefined type BOOLEAN.
Shift	sll | srl Shift left/right logical. sla | sra Shift left/right arithmethic. rol | ror Rotate left/right logical. Operators for any one-dimensional array type whose element type is either of the predefined types BIT or BOOLEAN (left operant) and predefined type *INTEGER* (right operand). The result type of each shift operator is the same as the left operand.
Adding	• Predefined operators for any numeric type. + | – • Predefined operator for any one-dimensional array type. & Concatenate, a & b makes one array For each of these adding operators, the operands and the result are of the same type.
Multiplying	• Predefined for any integer and any floating point type. * | / • Predefined for any integer type. mod | rem a mod b takes sign of b, a rem b takes sign of a For each of these multiplying operators, the operands and the result are of the same type.
Miscellaneous	• Predefined for any numeric type. abs absolute value ** a** is a^2

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	The exponential operator ** is predefined for each integer type and for each floating point type.

Predefined Data Types

The following table shows predefined types from the package *STANDARD*. The data type names are not technically reserved words but they represent a common standard. To avoid misunderstandings, do not re-define them. The standard package file is provided on the tutorial CD.

Data Type	Values	Example
BIT	'1', '0'	Q <= '1';
BIT_VECTOR	Array of bits.	DataOut <= "00010101";
BOOLEAN	TRUE, FALSE	EQ <= True;
INTEGER	-2, -1, 0, 1, 2, 3, 4 …	Count <= Count + 2;
REAL	1.0, -1.0E5	V1 = V2 / 5.3
TIME	1 ua, 7 ns, 100 ps	Q <= '1' after 6 ns;
CHARACTER	'a', 'b', '2', '$' …	CharData <= 'X';
STRING	Array of characters.	Msg <= "MEM: " & Addr

Predefined Type Declarations

```
TYPE INTEGER IS RANGE -2147483647 TO -2147483648;
TYPE BOOLEAN IS (FALSE,TRUE);
TYPE BIT IS ('0','1');
TYPE CHARACTER IS ( character_set_used_by_the_system );
TYPE SEVERITY_LEVEL IS (note,warning,error,failure);
TYPE REAL IS RANGE -9.9e99 TO 9.9e99;
TYPE TIME IS RANGE -9.9e99 TO 9.9e99
 UNITS fs;
 ps = 1000 fs;
 ns = 1000 ps;
 us = 1000 ns;
 ms = 1000 us;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 sec= 1000 ms;
 min= 60 sec;
 hr = 60 min;
 END UNITS;
TYPE DOMAIN_TYPE IS (QUIESCENT_DOMAIN, TIME_DOMAIN, FREQUENCY_
DOMAIN);
TYPE STRING IS ARRAY( 1 TO 2147483647 ) OF CHARACTER;
TYPE BIT_VECTOR IS ARRAY( 0 TO 2147483647 ) OF BIT;
```

Predefined Attributes

An object attribute returns information about a signal or data type. Predefined attributes denote values, functions, types, and ranges associated with various kinds of named entities.

The syntax of an attribute is some named entity followed by an apostrophe and one of the following attribute names. A parameter list is used with some attributes. The following abbreviations are used in the tables:

- **Q** – represents a quantity
- **S** – represents a signal
- **X** – represents a signal or variable
- **T** – represents a type
- **A** – represents an array or constrained array type
- **t** – represents an expression for time
- **e** – represent a static expression

Quantity Attributes

Q'DOT -- value of the derivative with respect to time of Q at the time the attribute -- is evaluated
Q'INTEG -- value of the time integral of Q from time 0 to the time the attribute is evaluated
Q'DELAYED[(t)] -- value of quantity Q delayed by t; if t is omitted, it defaults to 0.0
Q'ABOVE(e) -- true if Q-e is sufficiently larger than 0.0, false if Q-e is sufficiently smaller -- than 0.0. This attribute always returns a Boolean signal (TRUE/FALSE).
Q'ZOH(e[,INITIAL_DELAY])

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

-- constant value of Q at the sample times INITIAL_DELAY+k*e until the next sample -- time, with e as sample frequency and INITIAL_DELAY as the time of the first sampling -- (k is any non-negative integer); if INITIAL_DELAY is omitted, it defaults to 0.0
Q'LTF(NUM,DEN) -- Laplace transfer function of Q with NUM as the numerator and NEN as -- denominator polynomials
Q'ZTF(NUM,DEN,e[,INITIAL_DELAY]) -- Z transfer function of Q with NUM as the numerator and NEN as denominator -- polynomials, with e as sample frequency and INITIAL_DELAY as the time of the -- first sampling; if INITIAL_DELAY is omitted, it defaults to 0.0
S'RAMP[(TRISE[,TFALL])] -- quantity which follows the corresponding value of S with delay of rise time and -- fall time; if TRISE or TFALL is greater than 0.0, the corresponding value change -- is linear from the current value of S to its new value, whenever S has an event
S'SLEW[(RISING_SLOPE[,FALLING_SLOPE])] -- quantity which follows the corresponding value of S with rising slope and falling -- slope; if RISING_SLOPE is less than REAL'HIGH, or if the value or FALLING_SLOPE is -- greater than REAL'LOW, the corresponding value change is linear from the current -- value of S to its new value, whenever S has an event
Q'SLEW[(MAX_RISING_SLOPE[,MAX_FALLING_SLOPE])] -- quantity which follows the corresponding value of Q, but its derivative time -- is limited by MAX_RISING_SLOPE and MAX_FALLING_SLOPE

Signal Attributes

S'DELAYED[(t)] -- value of signal S at time now-t; if t is omitted, it defaults to 0.0 S'STABLE -- true if no event is occurring on signal S S'STABLE(t) -- true if no event has occurred on signal S for t time units S'QUIET -- true if signal S is quiet (no event this simulation cycle) S'QUIET(t) -- true if signal S has been quiet for t time units S'TRANSACTION -- bit value which toggles each time when signal S changes S'EVENT -- true if an event has occurred on signal S in the current cycle S'ACTIVE -- true if signal S is active in the current cycle S'LAST_EVENT -- time since the last event on signal S

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

S'LAST_ACTIVE -- time since signal S was last active S'LAST_VALUE -- value of signal S prior to latest change S'DRIVING -- false if the current driver of signal S is a null transaction S'DRIVING_VALUE --current driving value of signal S

Data Type Bounds

T'BASE -- base type of data type T T'LEFT -- left bound of data type T (largest if downto) T'RIGHT -- right bound of data type T (smallest if downto) T'HIGH -- upper bound of data type T (may differ from left bound) T'LOW -- lower bound of data type T T'ASCENDING -- true if range of T defined with to

Enumeration Data Types

T'IMAGE(X) -- string representation of X that is of discrete type T T'VALUE(X) -- value of discrete type T converted from the string X T'POS(X) -- integer position number of value of X of discrete type T T'VAL(X) -- value of discrete type T at position number X T'SUCC(X) -- value of discrete type T at position number X-1 T'PRED(X) -- value of discrete type T at position number X+1 T'LEFTOF(X) -- value of discrete type T at position left of X T'RIGHTOF(X) is the value of discrete type T at position right of X

Array Indexes for an Array A

For multi-dimensional array, N^{th} index must be indicated in the attribute specifier. N can be omitted for a one-dimensional array.
A'LEFT -- leftmost subscript of array A or constrained array type A'LEFT(N) -- leftmost subscript of dimension N of array A A'RIGHT -- rightmost subscript of array A or constrained array type A'RIGHT(N) -- rightmost subscript of dimension N of array A

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

A'HIGH -- highest subscript of array A or constrained array type A'HIGH(N) -- highest subscript of dimension N of array A A'LOW -- lowest subscript of array A or constrained array type A'LOW(N) -- lowest subscript of dimension N of array A A'RANGE -- range A'LEFT to A'RIGHT or A'LEFT downto A'RIGHT A'RANGE(N) -- RANGE of dimension N of array A A'REVERSE_RANGE -- range of array A with to and downto reversed A'REVERSE_RANGE(N) -- REVERSE_RANGE of dimension N of array A A'LENGTH -- integer value of the number of elements in array A A'LENGTH(N) -- number of elements of dimension N of array A A'ASCENDING -- Boolean True if range of array A is defined with to A'ASCENDING(N) -- Boolean True if dimension N of array A is defined with to

Reserved Words

The following words are reserved for the VHDL-AMS language, so they cannot be used as identifiers. Reserved appear in bold.

Word		**Description**
ABS		Operator, absolute value of right operand.
ACCESS		Declares a type for creating access objects, pointers.
ACROSS		Declares the across quantity of a particular **NATURE** type.
AFTER		Defines a delay value (time after *NOW*) in signal assignment statements.
ALIAS	*	Defines an alternate name for an object.
ALL		Makes all items in a package visible, refers to all names of a class, or refers to all instances of a component.
AND		Operator, logical **AND** of left and right operands.
ARCHITECTURE		Primary design unit. Defines an architecture.
ARRAY		Declares an array data type containing a collection of elements that are all of the same type (array, vector, or matrix).
ASSERT		Checks a condition (occurrence of an event) and provides a report if the condition is not *TRUE*.
ATTRIBUTE		Defines an attribute name and its type.
BEGIN		Defines the begin of a **BLOCK**, **ENTITY**, **ARCHITECTURE**, **IF**, or **PROCESS** statement.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Word		Description
BLOCK		Starts the description of a block structure.
BODY		Starts the description of various subprograms that are declared by the package body's associated package declaration.
BUFFER		Port mode. Indicates a port which can be used for both input and output, and it can have only one source.
BUS		Signal mode. Defines a bus that floats to a user-specified value when all of its drivers are turned off.
CASE		Sequential statement. Executes one of several sequences of statements within a **PROCESS**, **PROCEDURE**, or **FUNCTION** structure, depending on the value of a single expression.
COMPONENT		Starts a component declaration.
CONFIGURATION	*	Primary design unit. Defines a configuration for an **ENTITY**.
CONSTANT		Declares an identifier name for a constant value (read only).
DISCONNECT	**	Signal driver condition. Defines the time delay to disconnect the guarded feature of a signal which is part of a guarded signal statement.
DOWNTO		Defines a descending range in a **RANGE** statement or other statement which includes a range.
ELSE		Defines the final alternative in an **IF** or **WHEN** statement.
ELSIF		Defines an interim alternative in an **IF** statement.
END		Defines the end of an **ENTITY**, **ARCHITECTURE**, **CONFIGURATION**, **PACKAGE**, **PACKAGE** body, or **PROCESS** statement.
ENTITY		Primary design unit. Declares the input and output ports of a design.
EXIT		Sequential statement. Causes the immediate termination of a **LOOP**.
FILE		Declares a type for creating file handles and an identifier as a file object.
FOR		Identifies a parameter specification in a **LOOP** statement or time expression in a **WAIT** statement.
FUNCTION		Defines a group of sequential statements that are executed when the function is called.
GENERATE		Copies a set of concurrent statements, or executes a set of concurrent statements selectively if a specified condition is met.
GENERIC		Specifies static information to be communicated to a model from its environment for all architectures.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Word		Description
GROUP	**	Defines a group template or specific group that can get an attribute.
GUARDED		Limits the execution of a **SIGNAL** statement. Causes a **WAIT** until a signal changes from *FALSE* to *TRUE*.
IF		Sequential statement. Performs a sequence of statements dependent on a defined condition.
IMPURE		Declares a function that is assumed to have side effects (return a different value given the same actual parameters).
IN		Port mode. Indicates a port which can be used for input.
INERTIAL		Clause in delay mechanism, followed from a time. Signals smaller than delay time are suppressed.
INOUT		Port mode. Indicates a port which can be used for both input and output.
IS		Used as a connective in various statements.
LABEL		Defines a label name in attribute statements.
LIBRARY		Designates a simple library name to identify libraries from which design units can be referenced.
LINKAGE		Port mode. Indicates a port which can be used for both input and output, and it can only correspond to a signal.
LITERAL		Entity class. Specifies an **ENTITY** in **ATTRIBUTE** statements.
LOOP		Sequential statement. Executes a series of sequential statements multiple times.
MAP		Associates values of constants or ports within a structure to constants and ports outside the structure.
MOD		Operator, left operand modulo right operand.
NAND		Operator, logical **NAND** of left and right operands.
NATURE		Defines the **ACROSS** and **THROUGH** types of an object with a particular energy domain (nature).
NEW	**	Creates an object of a specified type and returns an access value that refers to the created object.
NEXT		Sequential statement. Cause the next iteration in a **LOOP**.
NOISE	**	Declares the source aspect of a noise source quantity (serves as a source in a frequency domain model) in a quantity declaration.
NOR		Operator, logical **NOR** of left and right operands.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Word		Description
NOT		Operator, complement of right operand.
NULL		Sequential statement. Specifies explicitly that no action is needed.
OF		Used as a connective in various statements.
ON		Used as a connective in various statements.
OPEN		Designates the initial file characteristic or indicates a port that is not connected to any signal.
OR		Operator, logical **OR** of left and right operands.
OTHERS		Defines all remaining elements for example in a **CASE** statement, a selected assignment, and **ATTRIBUTE** specification.
OUT		Port mode. Indicates a **PORT** which can be used for output.
PACKAGE		Primary design unit. Defines a package and package body.
PORT		Specifies dynamic information to be communicated to a model from its environment for all architectures.
POSTPONED		Declares a **PROCESS** as a postponed process. Postponed processes do not execute until the final simulation cycle at the currently modeled time.
PROCEDURE		Starts the description of a group of sequential statements that are to be executed when the procedure is called.
PROCESS		Starts the description of a group of sequential statements and is considered to be a single concurrent statement within a VHDL-AMS architecture.
PURE		Declares a **FUNCTION** that is assumed to have no side effects.
QUANTITY		Declares a **PORT** in an **ENTITY** declaration or **ACROSS**, **THROUGH**, and **REFERENCE** types in a **NATURE** declaration.
RANGE		Defines a range constraint for a scalar type.
RECORD	*	Defines a record type and its corresponding element types.
REFERENCE		Declares the reference quantity of a particular **NATURE** type.
REGISTER		Signal mode. Defines a storage register that retains its last driven value when all of its drivers are turned off.
REJECT		Clause in delay mechanism, followed from a time. Signals smaller than reject time are suppressed.
REM		Operator, remainder of left operand divided by right operand.
REPORT		Defines a text string that is displayed when the condition in an **ASSERT** statement is not *TRUE*.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Word		Description
RETURN		Sequential statement. Terminates a subprogram (**PROCEDURE** or **FUNCTION**) and returns control to the calling object.
ROL		Operator, left operand rotated left by right operand.
ROR		Operator, left operand rotated right by right operand.
SELECT		Selects and assigns a value to a target signal from among a list of alternatives, based on the value of a given expression.
SEVERITY		Defines the type of the message in an **ASSERT** statement.
SHARED		Declares shared objects.
SIGNAL		Declares an identifier as a signal object.
SLA		Operator, left operand shifted left arithmetic by right operand.
SLL		Operator, left operand shifted left logical by right operand.
SPECTRUM		Declares the source aspect of a spectral source quantity (serves as a source in a noise domain model) in a **QUANTITY** declaration.
SRA		Operator, left operand shifted right arithmetic by right operand.
SRL		Operator, left operand shifted right logical by right operand.
SUBNATURE		Declares a nature that is a subnature of an existing **NATURE**.
SUBTYPE		Declares a type that is a subtype of an existing **TYPE**.
TERMINAL		Declares a terminal object of a particular **NATURE**.
THEN		Defines the first choice in an **IF** statement when the condition is *TRUE*.
THROUGH		Declares the through quantity of a particular **NATURE** type.
TO		Defines an ascending range in a **RANGE** statement or other statement which includes a range.
TOLERANCE		Declares a tolerance that can be applied to scalar quantities.
TRANSPORT		Clause in delay mechanism, followed from a time. Defines a non-inertial delay in a signal assignment statement.
TYPE		Declares a type.
UNAFFECTED		Indicates in a conditional or selected signal assignment when the signal is not to be given a new value.
UNITS		Declares physical types.
UNTIL		Defines a condition to terminate a **WAIT** statement.
USE		Makes a package available to this design unit.
VARIABLE		Declares an identifier as a variable object.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Word		Description
WAIT		Suspends temporarily a process until a specified time has passed, a specified condition is met, or an event occurs which affects one or more signals.
WHEN		Defines a condition during which an **EXIT** or **NEXT** statement will be executed or defines a choice (or choices) within a **CASE** statement.
WHILE		Defines a condition during which a **LOOP** will be executed.
WITH		Defines an expression in a selected signal assignment statement.
XNOR		Operator, logical exclusive **NOR** of left and right operands.
XOR		Operator, logical exclusive **OR** of left and right operands.

***** Partially supported in Simplorer 8.0.

****** Not supported in Simplorer 8.0.

Modeling Aspects in Twin Builder

"Processes " below

"Quantities, Signals, and Variables " on the facing page

"Signal Assignments with Delay" on the facing page

"Data Exchange in Mixed-Signal Models " on page 22-59

"Signal to Quantity Assignment (Digital to Analog)" on page 22-59

"Quantity to Signal Assignment (Analog to DIgital)" on page 22-60

"Solvability " on page 22-61

"WORK Library" on page 22-62

"Alias for File Names" on page 22-62

"Values on Sheet " on page 22-63

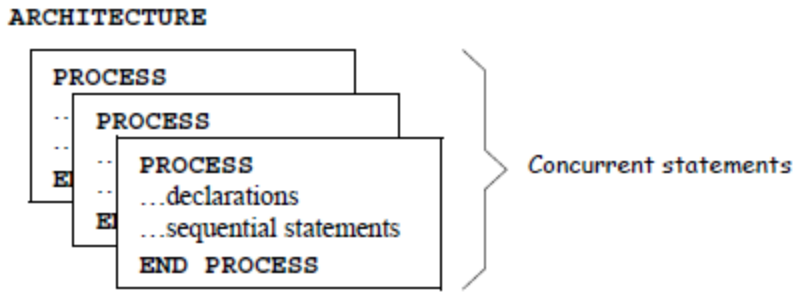
"Vector Inputs on Sheet " on page 22-64

Processes

A process statement defines an independent sequential process and is considered to be a single concurrent statement within a VHDL-AMS architecture.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Process statements contain sequential statements but are themselves concurrent statements. They are the primary means of defining sequential statements. A process statement can include all declarations and sequential statements.



Quantities, Signals, and Variables

	Description	Direction	Assignment
Quantities	Analog objects	**IN OUT**	== =>
Signals	Digital objects	**IN OUT INOUT**	<=
Variables	Auxiliary objects to store intermediate values	None	:=

Signal Assignments with Delay

VHDL-AMS offers several variants to perform signal assignments. The following example shows three delay forms:

```
ENTITY sequential_sig_assign IS
 PORT(SIGNAL output: out bit);
END;
ARCHITECTURE behav OF sequential_sig_assign IS
 SIGNAL sig_s,sig_t,sig_i,sig_r : bit;
BEGIN
 sig_s <= '1' AFTER 1 ms, '0' AFTER 5 ms,
 '1' AFTER 10 ms, '0' AFTER 13 ms,
 '1' AFTER 18 ms, '0' AFTER 20 ms,
 '1' AFTER 25 ms, '0' AFTER 26 ms;
PROCESS (sig_s)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

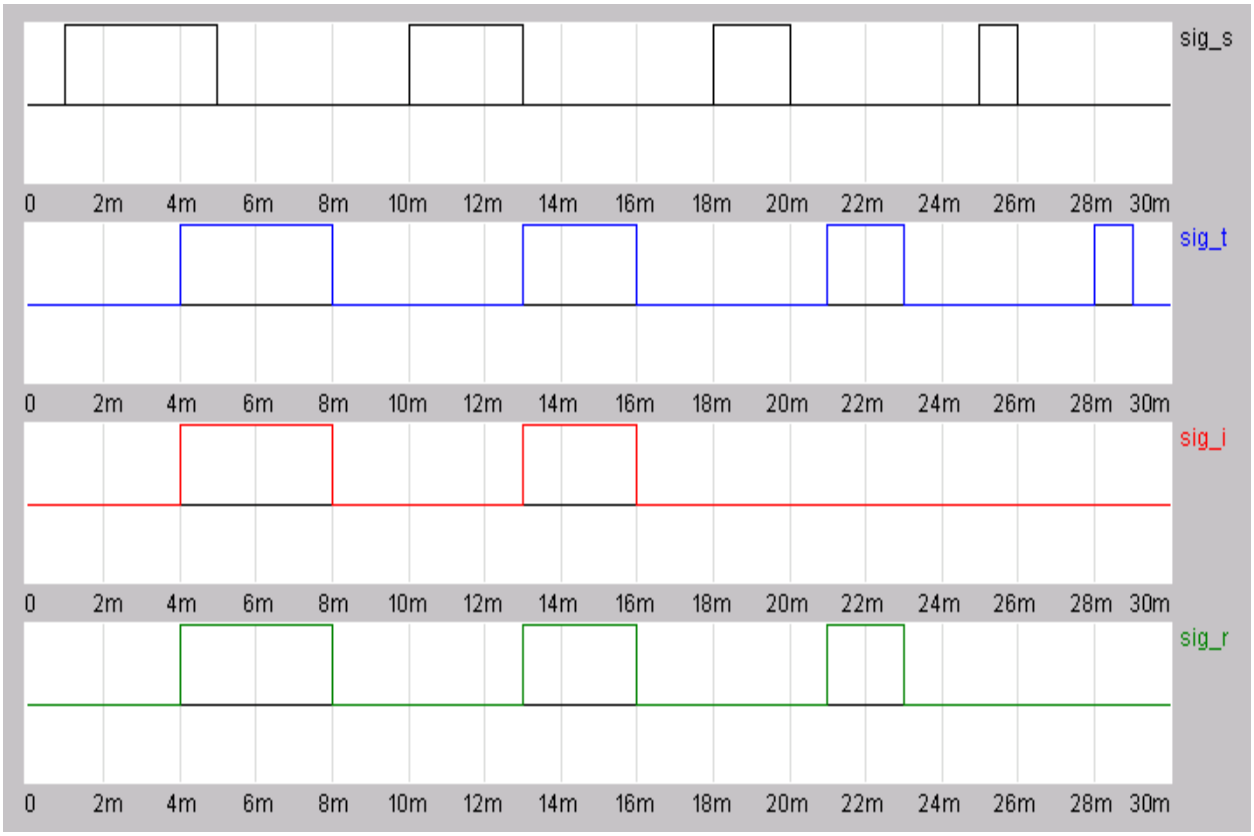
```
BEGIN
 sig_t <= TRANSPORT sig_s AFTER 3 ms; -- signal assignment 1:1
 sig_i <= INERTIAL sig_s AFTER 3 ms;
 -- signal assignment without signals smaller than delay time are
supressed
 sig_r <= REJECT 2 ms INERTIAL sig_s AFTER 3 ms;
 -- signal assignment without signals smaller than reject time are
suppressed
END PROCESS;
END;
```

The following figure shows the results of the previous modeling code. The signal *sig_s* is delayed and assigned to signals in three different ways:

- The signal *sig_t* represents *sig_s* with a delay of 3ms.
- The signal *sig_i* represents *sig_s* with a delay of 3ms without signal changes occurring in an interval smaller than delay time.
- The signal *sig_r* represents *sig_s* with a delay of 3ms without signal changes occurring in

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

an interval smaller than reject time.



Data Exchange in Mixed-Signal Models

Mixed-signal models use analog and digital statements in their model description. Mixed-signal assignments should be avoided when other modeling variants can be used. Compiler errors or simulator instabilities can occur when data assignments are not performed in the correct way.

The following sections show examples of assignments from digital to analog values and analog to digital values. In the examples, *quantity_name* stands for an analog value, *signal_name* for a digital value.

Signal to Quantity Assignment (Digital to Analog)

The first example shows a value assigned in the correct syntax but without any support for the solver to find a good solution.

```
quantity_name == signal_name; syntax correct but solver problems can occur
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To improve the solver stability, use a '*RAMP* or '*SLEW* attribute or a **BREAK** statement. The '*RAMP* and '*SLEW* attributes, as well as the **BREAK** statement, force a synchronization on the minimum simulator time step HMIN and thus minimize the error deviation of the solver.

The following examples show value assignments in connection with variants to define rise/fall time and positive/negative slew rate of a digital signal. The rise and fall time as well as positive and negative slew rate should be chosen so as to avoid infinite values of the first derivative for these quantities since this can cause simulator instabilities.

```
quantity_name == signal_name'RAMP(1.0e-9, 2.0e-9); -- define rise and fall time
quantity_name == signal_name'SLEW(100.0, 200.0); -- limit pos and neg slew rate
```

The following examples show value assignments dependent on events applied to a signal.

```
BREAK ON signal_name; -- assignment when event on signal_name
BREAK quantity_name'DOT => 1000.0 ON signal_name
BREAK qv => -qv when not quantity_'Above(0.0);
quantity_name==signal_name;
```

Quantity to Signal Assignment (Analog to DIgital)

The first example shows an incorrect way of using a value assignment. The statement is performed only during initialization, and the signal value is unchanged during rest of simulation. Signal assignments are characterized by events, and in this case quantities do not create any events.

```
signal_name <= quantity_name; -- only at init step, wrong assignment
```

The following example shows a value assignment if the quantity crosses a threshold value of 3.0. The signal value is of type BOOLEAN since the 'ABOVE attribute returns a *TRUE* or *FALSE*. In the next process, the actual value of the quantity is assigned to the signal. The next value assignment follows only after the quantity crosses the threshold again.

```
signal_name <= quantity_name'ABOVE(3.0); -- value of signal_name is true or false
PROCESS(signal_name)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
BEGIN
 signal_name <= quantity_name; -- value of signal_name is actual value + maximum
hmin
END;
```

The following example shows a value assignment at each time interval specified as delay:

```
clock <= NOT clock AFTER 1ms; -- generates an event at each time step defined as
delay
PROCESS(clock)
BEGIN
 signal_name <= quantity_name;
END;
```

Solvability

The analog solver needs a specific set of equations when it evaluates the statements in the model.

- The number of equations specified in the model must be equal to the sum of the number of free quantities, through quantities, and port quantities of mode **OUT**.
- All free quantities and port quantities of mode **OUT** must appear in a simultaneous equation within the architecture of the model.

The following example uses two relevant quantities: *I* (port quantity of mode **OUT**) and *ct* through quantity. Both quantities occur in the equations of the model.

```
LIBRARY IEEE;
USE IEEE.ELECTRICAL_SYSTEMS.ALL;
ENTITY AM IS
 PORT (QUANTITY I : OUT REAL; -- port quantity of mode out
 TERMINAL p,m : ELECTRICAL);
END ENTITY AM;
ARCHITECTURE behav OF AM IS
 QUANTITY v ACROSS ct THROUGH p TO m; -- through quantity
ct
BEGIN
 I == ct; -- first equation with port and through quantity
 v == 0.0; -- second equation
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential

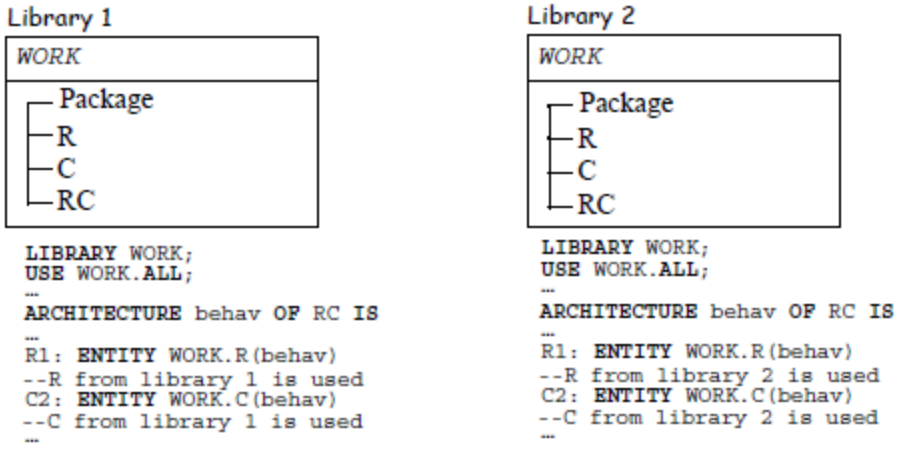
information of ANSYS, Inc. and its subsidiaries and affiliates.

```
END behav;
```

The digital solver needs a minimum simulation step HMIN that is less than or equal to the least delay specified with an **AFTER** statement.

WORK Library

Packages and models of a library are always compiled in the current WORK library. All packages and models of that library are visible to other models of the same library and can be instantiated as components in these models. If a model that instantiates models of the WORK library is copied to another library, the WORK library of this other library is used. That means, the same instantiation uses different models, because the WORK library was changed.



Models defined in VHDL-AMS text subsheets that are placed on a model sheet are also compiled in the current sheet's WORK library. All models of these subsheets are visible to other subsheets of the same sheet at any hierarchy level.

Entity names defined in subsheets must be unique within the entire sheet, because an entity with a duplicate name will overwrite the previously defined one.

The VHDL-AMS working directory of Twin Builder is

…\Ansoft\Simplorer8\CPL\VHDLA\VHDL_WORK.

To make certain that the correct model is used, delete the subdirectories in this folder from time to time.

Alias for File Names

A file alias name can be defined for a library name in Twin Builder. Select **Tools** > **LibraryTools** > **Manage Twin Builder Aliases**. Create a new file alias and define alias

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

name and path of the corresponding file. To use files with names that do not conform to VHDL-AMS identifier rules, such as a file name with spaces, specify a file alias.

Values on Sheet

In the Twin Builder Schematic, parameter values for VHDL-AMS models are entered in component dialogs. Values can be assigned to each parameter corresponding to the inputs defined in the model entity. If no value is entered, the default value specified in the model is used.

> **Note:**
>
> The syntax of user-defined values must conform to the SML conventions. VHDL-AMS attributes and syntax cannot be used.

Each value assigned to a parameter is from type *REAL*. The following table lists possible value assignments for VHDL-AMS data types and their effects.

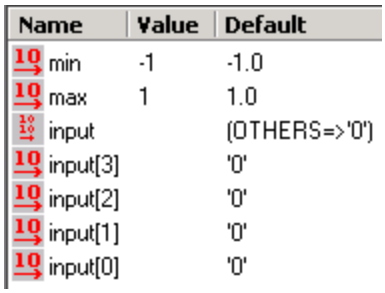
Name	Value	Default	Data Type	Object	Description
inp1	-1	-1.0	REAL	SIGNAL	All values are possible. If no default value is specified, name'LEFT is used (-1.7e-308).
inp2	var1	0.0	REAL	QUANTITY	
inp3	ctrl	'1'	BIT	SIGNAL	Values between 0 and 2 are possible, otherwise an error message occurs during simulation. if *inp*<1 then use value 0 if *inp*>=1 the use value 1
inp4	0	inp3'left	BIT	SIGNAL GENERIC	
inp5	2.5	10	INTEGER	SIGNAL	All values are possible. If *inp* is not integer value, the integer part of the value is used.
inp6	var2	0	INTEGER	GENERIC	
inp7	7	'H'	STD_ LOGIC	SIGNAL	Values between 0 and 8 are possible, otherwise an error message occurs during simulation.
inp8	var3	'0'	STD_ LOGIC	SIGNAL	

In addition to value specifications in the model dialog box, *OmniCaster* models can be used to perform value assignments. *OmniCasters* convert different data types correctly if a conversion is possible, otherwise they reject a connection between the parameters.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Vector Inputs on Sheet

Vector inputs appear in the component input dialog as parameters with index. Values for dynamic vectors can only be assigned with a wire.



Name	Value	Default
min	-1	-1.0
max	1	1.0
input		(OTHERS=>'0')
input[3]		'0'
input[2]		'0'
input[1]		'0'
input[0]		'0'

```
ENTITY DAC IS
 GENERIC (MIN: REAL := -1.0;
 MAX: REAL := 1.0);
 PORT (
 SIGNAL INPUT: IN BIT_VECTOR(3 DOWNTO 0):=
(OTHERS =>'0');
 QUANTITY VAL: OUT REAL := 0.0);
END ENTITY DAC;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

23 - Twin Builder Modeling Language

The Twin Builder Modeling Language (SML) is the general internal description language used in Twin Builder. Using the Schematic editor, components are positioned on the sheet and connected to form a graphical representation of a model. The SML description is generated from this graphical model.

Using the SML text editor, the model description in SML notation can be edited, and its syntax checked for validity.

The current version of the SML description language used by Twin Builder is SML v2.0. Select Tools>Project Tools>Import Simulation Models to convert SML files created in previous versions of Simplorer into the 2.0 format.

This chapter explains the common SML statements. Complete parameter lists of internal Twin Builder models are listed with the corresponding models.

This chapter contains information on:

- Common SML Conventions
- SML Model Descriptions
- Structural Models
- Configuration Controls

Common SML Conventions

This section gives an overview of the following conventions used in SML descriptions.

- Comments
- Names
- Separators
- Continuation Sign
- Keywords, Pre-Defined Names, and Constants
- Nodes
- Model Attributes

Comments

In SML source code comments can be inserted to:

- Document the text (author, date, and so on)
- Provide general description of a modeling task
- Remove temporary text lines.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder supports two types of comments:

- **Single Line Comments**: Text after a double slash "//" is interpreted as comment up to the line end.
- **Multi-Line Comments**: The symbols "/*" and "*/" separate comments from the other text. All characters within these marks are ignored by the simulator.

Example:

```
SMLDEF MacroRC
{
 /* electrical pins */
 PORT ELECTRICAL : N1;
 PORT ELECTRICAL : N2;
 /* parameter pins */
 PORT REAL in RESISTANCE[Ohm] : ValueR = 1k; // value of the
resistance
 PORT REAL in CAPACITANCE[F] : ValueC = 1u; // value of the
capacitance
 /* Output values - unused
 PORT REAL out : VR = R1.V;
 PORT REAL out : VC = C1.V;
 */
 INTERN R R1 N1:=N2, N2:=GND ( R := ValueR ) ;
 INTERN C C1 N1:=N2, N2:=N1 ( C := ValueC ) ;
}
```

Names

SML supports the following kinds of names.

- File Names
- Define Names
- Type Names
- Instance Names

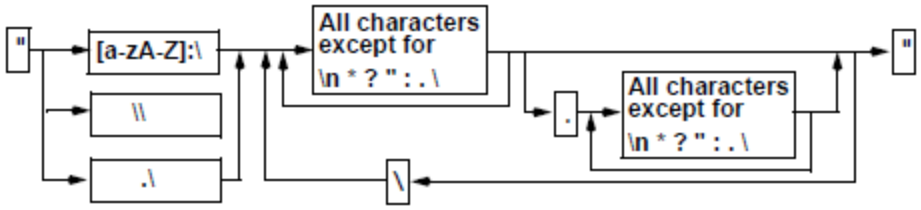
File Names

File names specify a file name and path in a definition.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

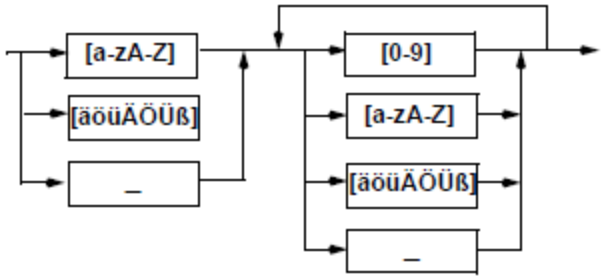
Note:

File names must be surrounded by quotation marks.



Define Name and Value

A define value set in a preprocessor instruction replaces a define name without attention to the context. The syntax rules of the context are true.



Type Names

Type names specify a model (element) type (such as R, GS, ST, EXP), or definition names of user-defined C-Models, structural models, and models of sub-simulators. Type names must be clear. You cannot use an type name twice unless other parameters (source, index, simulator) make the definition clear. The names may consist of letters, digits and underscores and can have a maximum of 50 characters.

Note:

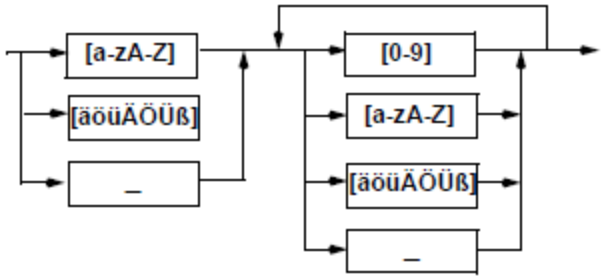
All names are case sensitive. The first character must always be a letter or underscore.

The following are *not* allowed as type names:

- SML notation keywords
- Names of standard mathematical functions

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Names of variable types (for example, real)



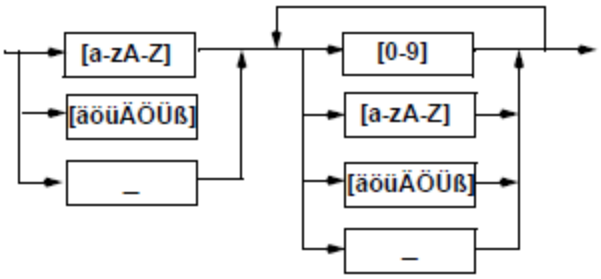
Instance Names

Instance names specify a model (element) in a modeling level, such as circuit components, blocks, states, time functions, characteristics, standard variables, subsheets, and macros. Instance names in a modeling level must be clear. You cannot use an instance name twice in the same modeling level. The names may consist of letters, digits and underscores and can have a maximum of 50 characters.

> **Note:**
>
> All names are case sensitive. The first character must always be a letter or underscore.

The following are not allowed as instance names:

- SML notation keywords
- Names of standard mathematical functions
- Names of variable types (for example, real)



Separators

Except for file names, which are surrounded by quotation marks, blank spaces, tabs, and carriage returns will be considered as separators. File names can be contain blank spaces. In

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

some cases special separators are defined.

- **Comma**

  Separator between nonconservative nodes, dimension statements of multidimensional quantities, and index statements for access to multidimensional nodes.

- **Assignment (:=)**

  Separator between name and value of a nonconservative node respectively separator between left and right side of an equation.

- **Semicolon**

  Separator between model instances.

- **Brackets**

  ( ) Start and end of a nonconservative node list, parameters of math functions, or for grouping of math operations.

  { } Start and end of a model definition or a group of similar elements.

  [ ] Start and end of a dimension statement of multidimensional quantities or the index list for access to multidimensional nodes.

Continuation Sign

If an input must be carried over to the next line, use the back slash (\ ) as a continuation sign.

Keywords, Pre-defined Functions and Operators, Constants

- **Keywords**

  INTERN, MODEL, SMLDEF, *MODELDEF, UMODEL, COUPL, SIMCTRL, SIMCFG, OUTCTRL, OUTCFG, RESULT, RUN

> **Note:**
>
> ***SMLDEF** has replaced **MODELDEF** in Simplorer 8. However, the **MODELDEF** keyword is still recognized to preserve backward compatibility with Simplorer 7.

- **Pre-defined mathematical functions**

  These functions include Trigonometric, arithmetic, exponential, complex, conversion, rounding, conditional functions. Examples of notations are sin, cos, tan, sinh, cosh, tanh, asin, acos, atan, squ, sqrt, log, and ln. For a full list, see Standard Mathematical Functions.

- **Pre-defined mathematical operators**

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

These operators include arithmetic operators (+ - * / **), logic operators (AND, NOT, OR, NAND), and comparison operators (<, >, =, <=, >=, !=).

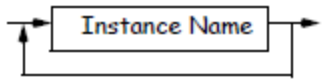
For a full list, see Operators.

- **Mathematical constants**

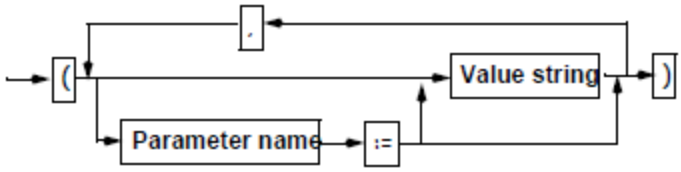
  PI, NULL, TRUE, FALSE

Nodes

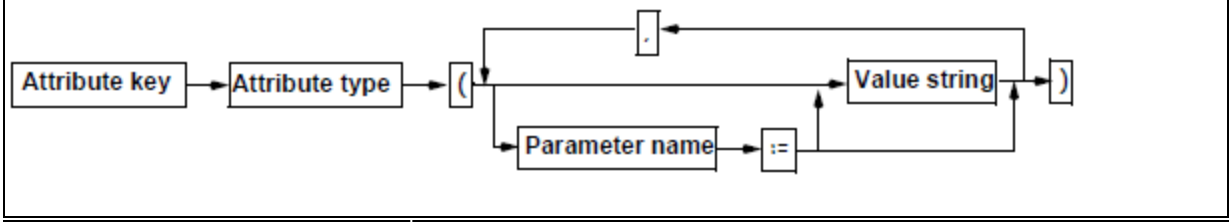
Conservative Nodes



Nonconservative Nodes



Model Attributes



Attribute key:	**SRC:** and **DST:**
Attribute types with SRC:	**DLL**: Name of DLL file, which contains the model (UMODEL).
	DB: Name of data base source file, which contains the model text (MODEL, COUPL) or model DLL (UMODEL).
Attribute types with DST:	**SIM**: Name of the external subsimulator, which calculates the models and, for immanent models, those also provide. Name of internal subsimulators for equation calculation (FML models).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

MODEL TFR1P2W TFR1P2W1 N1_A:=N0002, N1_B:=GND, N2_A:=N0005, N2_B:=N0006 (LM:=0.1 ,LS1:=1m ,LS2:=1m ,RFE:=1.0e+018 ,RCU1:=1u ,RCU2:=1u ,KTR:=1) **SRC: DB** (LIB**:="**Basic Elements\Basic Elements"**);**

UMODEL VOL_CONST VOL_CONST1 H1:=N0002 (VOL:=100u ,B:=700meg) **SRC: DLL**(File:**="**Hydraulic.dll**") ;**

INTERN EQU {variable1 := A+B ; } **DST: SIM**(Type:=SFML, Sequ:=INIT);

SML Model Descriptions

The essential parts of an SML model description are:

- Preprocessor instructions (# statements)
- Structural models (subsheets)
- Model instances
- Simulator configurations
- Output configurations

Keywords separate the SML source code into different sections. Within a section every line ends with a semicolon. For easier legibility, spaces can be inserted and lines indented in the lines of code.

Preprocessor Instructions

Preprocessor instructions begin with a number sign (“#”) and must be ended with a newline character (enter). You can arrange as many as you like in any order.

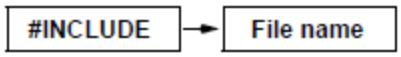
Instruction	Description
#DEFINE	Inserts equivalent names.
#UNDEF	Removes the equivalent name.
#INCLUDE	Includes an SML file.
#IFDEF	Includes SML statements dependent on #DEFINE instructions.
#IFNDEF	Includes SML statements dependent on #DEFINE instructions.
#ELSE	Includes SML statements dependent on #DEFINE instructions.
#ENDIF	Closes an #IFDEF/#IFNDEF instruction.
#SET	Sets initial values in a simulation description.
#EXTSIM	Includes models of external subsimulators.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Instruction	Description
#ENDEXTSIM	Closes an #EXTSIM instruction.

#INCLUDE Instruction

The #INCLUDE instruction includes the defined SML file at the position of the instruction and forces the compiler to continue the process with the include file. This file contains a part of the model description, which can consist of any SML text, as SML source code, model definitions, source code of models of external simulators, definitions or comments,.



File name:	File name that must be enclosed in quotation marks.

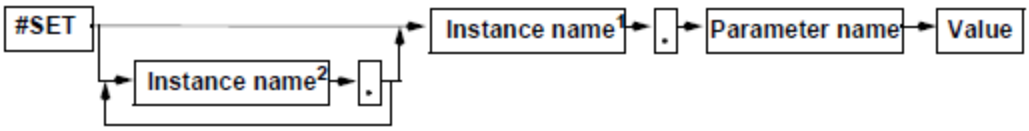
#INCLUDE "Models\my submodel.sml"

Thus parts of models can be used in several problems. The include instruction can be nested and is limited only to the maximum number of open files allowed by the operating system. However, an SML file cannot be included recursively.

SML components or models are arranged according to the sequence of the include assignments. This sequence of components is important in the block diagram and for formulas and value assignments in the state graph, where the sequence of processing can be decisive for the whole system.

#SET Instruction

The #SET instruction defines values of initial parameters and states (initial values of capacitors, inductors, integrators) before the simulation start, independent of the assignments in the SML description. You can set the values in a separate file or at the beginning of the SML source text. A separate file must be included with the #INCLUDE command in the model description.



Instance name[1]:	Name of the model, which has initial parameters.
Instance name[2]:	Additional instance names of structural models if the model with initial parameters is placed in a sub structure. To set initial parameters in structural models, all instance names must named before the parameter name beginning from the outside inwards.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Parameter name:	Name of the initial parameter.
Value:	Value of the initial parameter. Numerical value with or without unit suffix.

#SET C1.V0 -6.59909u

#SET Sym1.N0 9894.02

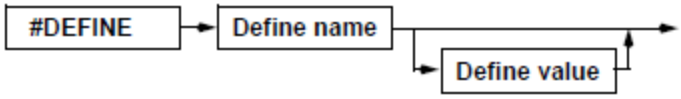
#SET State1.Z 1

#SET Sheet1.Sheet2.Lr1.I0 1.4121e-3

Thus a simulation can be started with values of a steady-state condition. If an initial value is defined with a #SET instruction twice, the last assignment is used.

#DEFINE Instruction

The #DEFINE instruction assigns the defined value to a name, independent of the assignments in the SML file. All following parameters with the define name are set to the value in the #DEFINE instruction. A define name can also be used in an #IFDEF and #IFNDEF instruction as a condition. You can place the definitions at the beginning of the SML source text or in a separate file. A separate file must be included with the #INCLUDE command in the model description.



Define name:	Name of an user-defined parameter of which value will be replaced (no parameter names of models possible).
Define value:	Character string which replaces the value of the parameter in the model description.

#DEFINE INIT

#DEFINE Initvalue 100

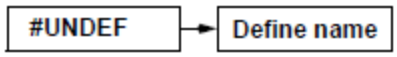
An empty character string as define value removes the value of the define name in the model description. In an #IFDEF and #IFNDEF condition, however, the define name can be used.

#UNDEF Instruction

The #UNDEF instruction removes the definition of the defined name. All following values of the define name will not be replaced. The define name will be considered of #IFDEF and #IFNDEF conditions as undefined. You can place the definitions at the beginning of the SML source text or

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

in a separate file. A separate file must be included with the #INCLUDE command in the source text.



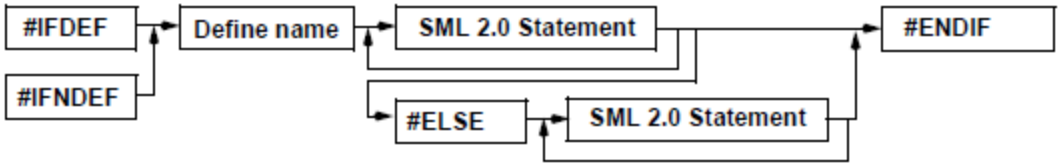
Define name:	Name of a user-defined parameter which was defined in an #DEFINE instruction.

#UNDEF MIT_INIT

#UNDEF Init_value

#IF Compile Instruction

The #IFDEF and #IFNDEF instructions control the compilation of a SML file, dependent on existing (with #DEFINE created) or not existing define names. If the define name exists (#IFDEF), or the define name not exists (#IFNDEF), the compiler continues the process directly after the statement to the next #ENDIF - or #ELSE - instruction. If no condition is true, the compilation is interrupted to the next #ENDIF - instruction. You can place the definitions at the beginning of the SML source text or in a separate file. A separate file must be included with the #INCLUDE command in the source text.



Define name:	Name, which is test of #IFDEF or #IFNDEF instructions.

#IFDEF MIT_INIT

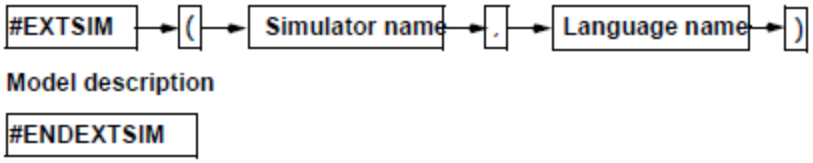
#SET C1.U0 -6.59909

#ENDIF

#EXTSIM Instruction

The instructions #EXTSIM and #ENDEXTSIM includes directions and model descriptions of an external subsimulator (for example, SimVHDL). These directions and model descriptions will be write in a separate text file and transferred to the compiler of the external subsimulator. The models defined in an #EXTSIM instruction can be used with COUPL in the SML description.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Simulator name:	Name of the external registered subsimulator of the Twin Builder simulator SIM2000 corresponding to the entry in the registration file Register.CFG.
Language name:	Name of the registered language name of the external subsimulator (for example, "VHDLA" for subsimulator SimVHDL) corresponding to the entry in the registration file Register.CFG.
Model description:	Model description in the syntax of the external subsimulator.

#EXTSIM (VHDLSim,VHDLA)

entity RC is

end;

architecture behav of RC is

terminal k1,k2,gnd: electrical;

…

begin

vout==1.0; --direct voltage

i_r==u_r/1000.0;

i_c==1.0e-6 * u_c'dot;

end;

#ENDEXTSIM

Model Instances

The structure of a simulation description is defined in general by:

- different components (circuit components, blocks, states, equations, macros, C-Models, …)
- the nodes (conservative, non-conservative) of the components
- component parameters.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

A definition for a component consist of the model instance, the component type and instance name, followed by the conservative nodes and nonconservative nodes (parameters), which must be separated by at least one blank space. In some cases additional attributes can be defined.

Types of model instances in the SML:

- INTERN
- MODEL
- UMODEL
- COUPL

INTERN Model Instance

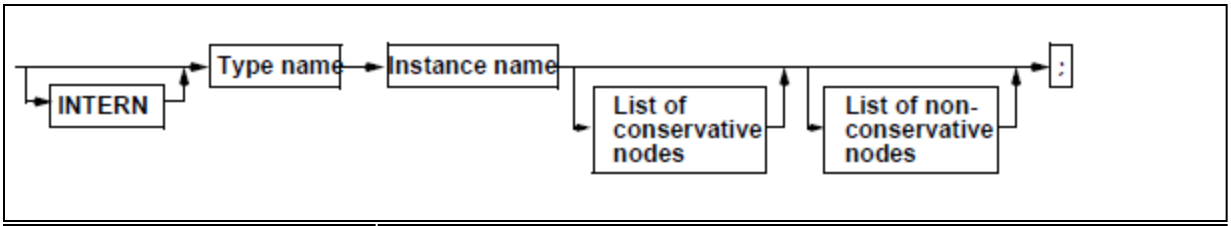
The INTERN model instance is used for internal simulator models, which are provided of the BASIC ELEMENTS model library. The keyword INTERN can be omitted. Internal models in the Basics library are:

- Circuit components to model electrical circuits.
- Block diagrams to model control and cybernetic systems consisting of different blocks with typical transfer behaviors.
- State graphs to model discontinuous processes as event oriented, based on Petri Net theory. This type of model divides a system into: significant states and events, transfers from one state to the other.
- Time functions to provide time behavior for different components. The time functions can be divided into predefined and user-defined time functions.
- Characteristic functions to model nonlinear characteristics in the form of y=f(x).

Internal models can be divided into models with fixed behavior and models with user-defined behavior.

Models with Fixed Behavior

Models with fixed behavior have a defined number of conservative and nonconservative nodes.



INTERN	Keyword for internal simulator models.
Type name:	Name of the internal model, such as RNL, IPULSE, or ASM.
Instance name:	User-defined name for using the model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Conservative nodes:	List of conservative nodes (if existing) separated by a comma.
Nonconservative nodes:	List of nonconservative nodes (if existing) separated by a comma.

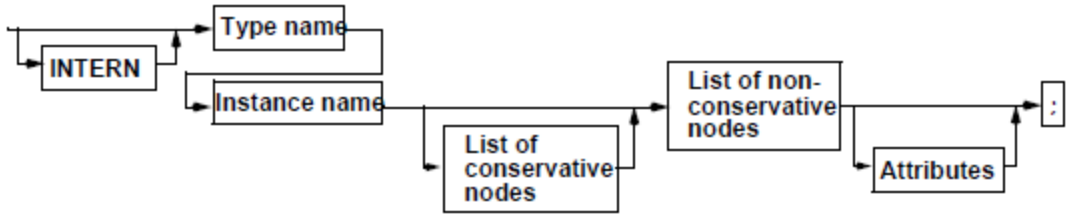
INTERN RNL R1 N1:=N0001, N2:=N0002 (CH := XY_R1_CH.VAL);

INTERN XY XY_R1_CH (FILE := "Sheet2__R1.mdx");

INTERN E E1 N1:=GND, N2:=N0001 (EMF := 1 ,PARTDERIV := 1 **);**

Models with User-Defined Behavior

Models with user-defined behavior have a different number of conservative and nonconservative nodes. Internal components with user-defined behavior are initial conditions assignments, equations, states, and equation blocks.



INTERN	Keyword for internal simulator models.
Type name	Name of the internal model, such as EQU, STATE, or EQUBL.
Instance name	User-defined name for using the model.
Conservative nodes	List of conservative nodes (if existing) separated by a comma. Only for the block equations (EQUBL) used.
Nonconservative nodes	List of nonconservative nodes (if existing) separated by a comma.
Attributes	Attribute statements (SRC: and DST:). Only for equations (EQU) used.

INTERN EQU {variable1 := A+B ; } DST: SIM(Type:=SFML, Sequ:=INIT);

INTERN EQU {variable2 := t*100 ;}

INTERN EQU {variable3 := sin(A) ;} DST: SIM(Type:=SSGM, Sequ:=PRE);

INTERN EQU {variable4 := cos(B) ;} DST: SIM(Type:=SSGM, Sequ:=POST);

INTERN STATE

STATE1 {

MARK: 0;

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
CATI:C:=10;
SET:D:=D+1;
}
INTERN EQUBL EQUBL1 ( TS := 0, INPUT[0] := E1.EMF )
{
PORT real out : VAL[2] = 0;
Z := SQU(E1.EMF);
VAL[0] := INPUT[0]+Z;
VAL[1] := Z;
}
```

Initial Condition Assignments and Equations

Each system variable provided by Twin Builder can be used within an expression on the right side. Thus it is possible to establish any context between system variables.

Initial Value Assignment

The initial value assignment is a list of value assignments to variables and constants, which are calculated only **once** at the simulation start. The process sequence is equal to the sequence of input.

```
INTERN EQU {A := PI ; B:= 0.5 } DST: SIM(Type:=SFML, Sequ:=INIT);
```

Equations

The equation assignment is a list of value assignments to variables and constants, which are calculated at each valid step in the sequence of input.

- **before all** other modules

```
INTERN EQU {variable1 := t*100; variable2 := t*200 ;}
```

- **before** the state graph analysis is processed.

```
INTERN EQU {variable3 := sin(A);} DST: SIM(Type:=SSGM, Sequ:=PRE );
```

- **after** the state graph analysis is completed.

```
INTERN EQU {variable4 := cos(B);} DST: SIM(Type:=SSGM, Sequ:=POST );
```

Defining Actions in States

Through actions a state graph model influences either the rest of the simulation model or the simulation parameters. The function is defined by the action type.

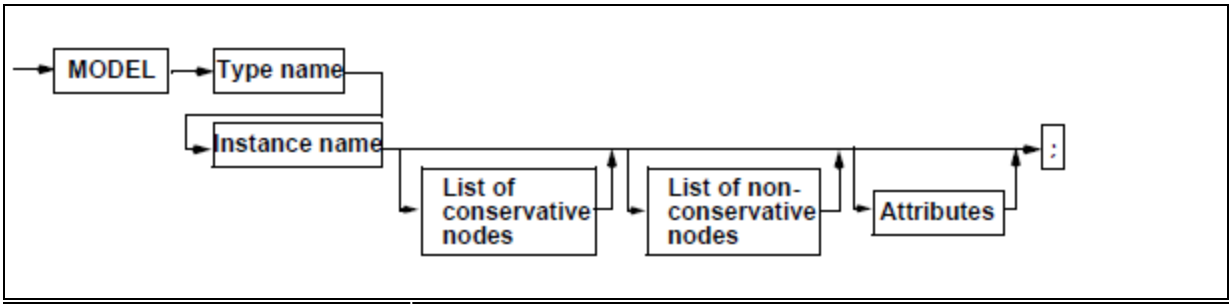
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

INTERN STATE

```
STATE1 {
 MARK: 0;
 CATI:C:=10;
 SET:D:=D+1;
}
```

MODEL Instance

The MODEL model instance is used for user-defined structural models (all kind of macros), which are included in a model library or defined with a SMLDEF statement.



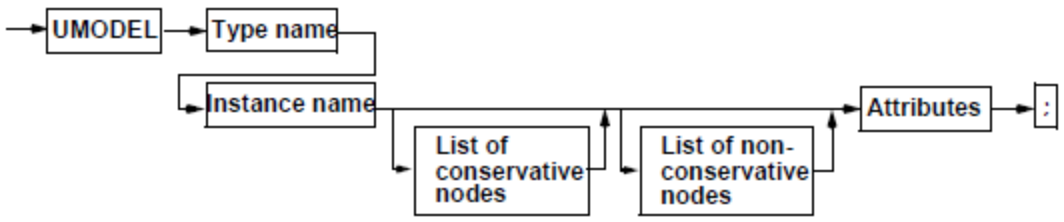
MODEL	Keyword for internal simulator models.
Type name:	Name of the model in the model library.
Instance name:	User-defined name for using the model.
Conservative nodes:	List of conservative nodes (if existing) separated by a comma.
Nonconservative nodes:	List of nonconservative nodes (if existing) separated by a comma.
Attributes:	The SRC attribute defines the source model library and identifier. Both can be omitted, the simulation speed, however, is slacked because of that.

MODEL TFR1P2W TFR1P2W1 N1_A:=N0002, N1_B:=GND, N2_A:=N0005, N2_B:=N0006 (LM:=0.1 ,LS1:=1m ,LS2:=1m ,RFE:=1.0e+018 ,RCU1:=1u ,RCU2:=1u ,KTR:=1) **SRC: DB (LIB:="**Basic Elements\Basic Elements"**);**

UMODEL Instance

The UMODEL model instance is used for user-defined C-Models, which are included in a model library.

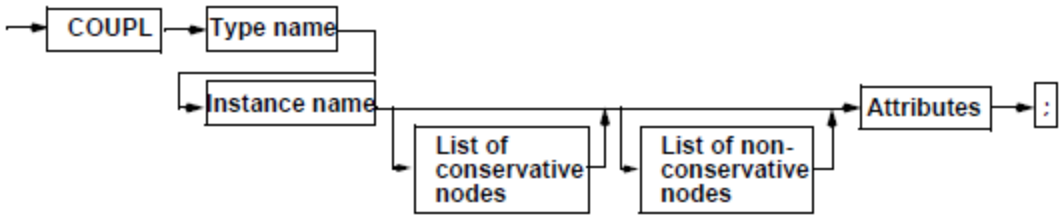
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



UMODEL	Keyword for internal simulator models.
Type name:	Name of the model in the model library.
Instance name:	User-defined name for using the model.
Conservative nodes:	List of conservative nodes (if existing) separated by a comma.
Nonconservative nodes:	List of nonconservative nodes (if existing) separated by a comma.
Attributes:	The SRC attribute defines the DLL file of the model.
UMODEL VOL_CONST VOL_CONST1 H1:=N0002 (VOL:=100u ,B:=700meg) **SRC: DLL(LIB:=**"Multiphysics\Hydraulic"**) ;**	

COUPL Instance

The COUPL model instance is used for models of external simulators.



COUPL	Keyword for internal simulator models.
Type name:	Name of the model in the model library.
Instance name:	User-defined name for using the model.
Conservative nodes:	List of conservative nodes (if existing) separated by a comma.
Nonconservative nodes:	List of non-conservative nodes (if existing) separated by a comma.
Attributes:	The SRC attribute defines the .dll file of the model.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Structural Models (Subsheets)

Structural models consist of the model definition, with header and model description. The ports in the header must match in number and type according to their utilization in the model call with the MODEL instance.

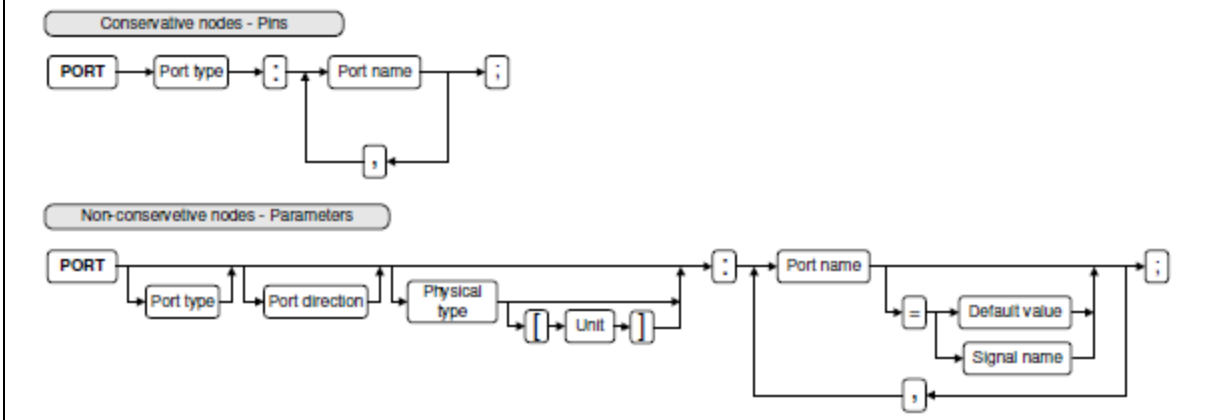
```
// Model definition
SMLDEF Name
{
```

- Header of structural model
- Model description of structural model

```
}
//Model Usage
MODEL model_name instance_name conservative_nodes (parameters)
```

Header Information of Structural Models

The header of structural models (subsheets) contain type, direction, and name of ports. There are two types of ports: electrical and real. In addition, the header may contain the assignment of default values for the variables defined in the port list.



Port type:	For conservative nodes: ELECTRICAL, FLUIDIC, MAGNETIC, TRANSLATIONAL, TRANSLATION_V, ROTATIONAL, ROTATIONAL_V, THERMAL For nonconservative nodes: REAL, FILE, STRING Default port type = REAL
Port direction:	Direction of data exchange over the nodes = { IN | OUT | INOUT }; Default = IN: The port direction is supported only for

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	real ports (nonconservative nodes).
Physical type	Specifies a special data type for non-conservative nodes (for example, RESISTANCE, CAPACITANCE, ANGLE).
Unit	The value transferred by this parameter will be converted to this unit/subunit of the given physical type (for example, Ohm, kOhm, MOhm for RESISTANCE). The default unit is always the SI unit for the given physical type.
Port name:	Definition name of the node. The syntax rules for type names are valid.
Default value	Default value for non-conservative nodes with IN as port direction. This value will be used, if this parameter is not used in the model instance call. The initial value must be a number or arithmetical expression, which is calculable without access to other quantities.
Signal name	Name of the signal which is connected to a non-conservative nodes with IN or INOUT as port direction. This name must be a name of a variable or model parameter inside the structure definition. It cannot be used as default value.

PORT electrical : T1;

PORT translational : T2;

PORT real in : Value = 2k;

PORT real in RESISTANCE[kOhm] : R_value = 2;

PORT real out : R1_I = R1.I;

PORT real inout : count = count_var;

Model Description of Structural Models

After the header, the actual model description is defined. All model instances (INTERN, MODEL, UMODEL, COUPL) may be used as model text. The description is enclosed in braces { }.

Structural models itself can be used within a SMLDEF (MODELDEF) section with the MODEL instance. However, structural models must not be called recursively.

Usage of Structural Models

With the MODEL instance, a structural model can be included in the SML description.

After the compiler finds the definition of the specified structural model/macro and checks for completeness of the parameter lists, it replaces the call with the contents of the structural model. Finally, the compiler creates a flat representation of the entire system resolving all structural

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

models/macros. This way, the formal nodes and parameters are substituted for the actual or default ones, respectively.

Also, all other nominators used in the macro are replaced. Therefore, it is impossible to get outputs or references that directly access topology elements or nominators inside the macro model. They have to be made accessible to the outside through the pins or the parameter list.

```
// Definition of structural model
SMLDEF Macro2
{
PORT electrical : T1;
PORT electrical : T2;
PORT real in : Value = 2k;
PORT real out : R1_I = R1.I;

INTERN C C1 N1:=N0002, N2:=T2 ( C := 1u ,V0 := 0 ) ;
INTERN R R1 N1:=T1, N2:=N0002 ( R := Value ) ;
}
//Model description
INTERN E E1 N1:=GND, N2:=N0001 ( EMF := 1 ,PARTDERIV := 1 );
INTERN R R1 N1:=N0001, N2:=N0002 ( R := 1k ) ;
INTERN EQU { R := 1k ;}
//Usage of structural model
MODEL Macro2 Subsheet1 T1:=GND,T2:=N0002 ( Value:=R );
```

Configuration Controls

SML has two different configuration controls: SIMCTL for simulator configurations and OUTCTL for configuration of output devices.

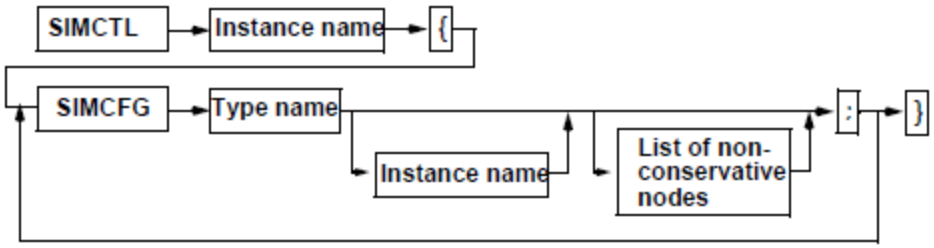
> **Note:**
>
> The configuration settings described in this section are made via Twin Builder's user interface menus and dialog boxes.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Simulator Configurations

Simulation control parameters define the process control of the simulation. These parameters must be handled very carefully, because varying them improperly may result in unpredictable numeric effects.

See also Twin Builder Analyses.



Instance name:	Name of the simulator control instance.
Type name:	Name of the used simulator.
Instance name:	Instance of the used simulator.
Nonconservative nodes:	List of nonconservative nodes (simulation parameters) separated by a comma.

```
//Simulator Configuration

SIMCTL SimCtl1 {

SIMCFG SIMPLORER_TR Simplorer1 ( Tend := 40m, Hmin := 10u, Hmax := 1m, Theta := 23 );

SIMCFG SECM SECM1 ( Solver := 1, LDF := 1, Iteratmax := 40, IEmax := 0.001, VEmax := 0.001 );

}
```

Related Topics

Standard Analysis Setup Options

Solution Options

Output Configurations

During the simulation a large amount of data will be produced. These data can be displayed directly on the screen or saved in files through output specifications.

- **Online Graphic Output**

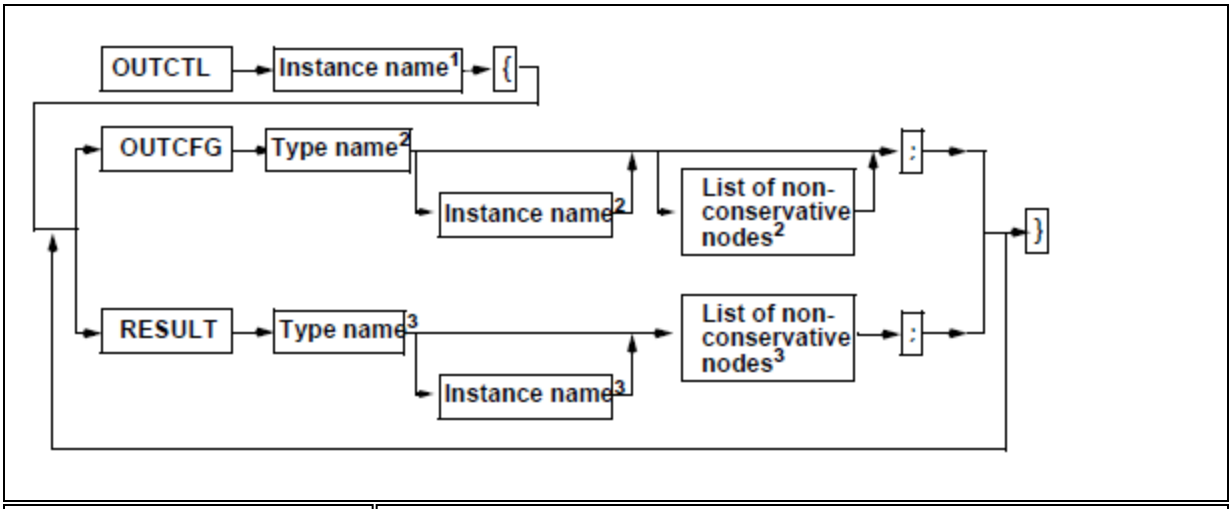
Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

To display the simulation results during the simulation, specify a View Display (VIEW) as the output destination.

- **Output in Files**

  To evaluate data with other tools, specify the Twin Builder database (SDB) as the output destination. For all output channels, the simulator creates an SDB file. The file name is formed from the name of the SML file and the extension SDB.

Every output destination can be assigned any system quantities from the electrical circuit, the block diagram, the state graph and formulas, functions and variables. It is also possible to define a system quantity for several output destinations.



Instance name [1]	Name of the output control instance.
Type name [2]	Name of the used output device = { VIEWTOOL | SimplorerDB }.
Instance name [2]	Instance of the used output device.
Nonconservative nodes [2]	List of nonconservative nodes (parameters of the output device) separated by a comma.
Type name [3]	Name of the used result format= { VIEW | SDB }.
Instance name [3]	Instance of the used result format.
Nonconservative nodes [3]	List of nonconservative nodes (output quantities) separated by a comma.

```
//Output Configuration
OUTCTL OutCtl1{
OUTCFG VIEWTOOL Out1 ( Xmin := 0, Xmax := Tend, Ymin := -400, Ymax := 400 );
RESULT SDB SDB_0( E1.EMF );
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
RESULT SDB SDB_1( STEP1.VAL );

RESULT VIEW VANALOG_2 ( E1.EMF, Type:=ANALOG );

RESULT VIEW VANALOG_3 ( C1.I, Type:=ANALOG );

OUTCFG SimplorerDB DB1 ( Xmin := 0, Xmax := Tend, Reduce := 0, StepNo := 2, StepWidth
:= 10u );

}
```

Related Topics

Generating Reports and Postprocessing

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

24 - Scripting

For information on how to create, edit and use IronPython and VB scripts in Twin Builder, select **Help** > **Twin Builder Scripting Help**.

Running VBScripts in 64-bit Windows Installations

The VBScript interpreter (**cscript.exe**/**wscript.exe**) comes in two versions on a 64-bit version of Windows: a 32-bit version and a 64-bit version. The 32-bit version can create and use 32-bit COM components only. The 64-bit version can create and use only 64-bit COM components. By default, .vbs files are associated with the 64-bit version.

Twin Builder scripting executables are 32-bit and must be created through the 32-bit VBScript interpreter.

You can find the 32-bit version of the VBScript interpreter in the **<*windows*>\SysWOW64** folder. Launching this version should give you access to all 32-bit COM components. In the scripting examples provided with Twin Builder (located in **<*installdir*>\Examples\Twin Builder\Applications\Scripting**), there are **BAT** files that run the WScript appropriate to the operating system.

In general, a VBScript can be run on either 32-bit or 64-bit Windows operating systems by using the following **BAT** file template (all on one line):

```
if exist %WINDIR%\SysWOW64 (%WINDIR%\SysWOW64\wscript.exe
<yourScriptFile>.vbs) else (%WINDIR%\System32\wscript.exe
<yourScriptFile>.vbs)
```

PyAEDT (Beta)

PyAEDT is a Python library that interacts with the AEDT API to make scripting simpler for the end user. It supports all AEDT 3D products (HFSS, Icepak, Maxwell 3D, and Q3D Extractor), 2D tools, Ansys Mechanical, EMIT, Circuit tools like Nexxim, system simulation tools like Twin Builder, and layout tools like HFSS 3D Layout and EDB. Additionally, it enables the end user to have a CPython interface with AEDT. Its class and method structures simplify operation for the end user, enabling more Pythonic code while reusing information as much as possible across the various APIs.

Documentation for PyAEDT can be found online at: https://aedt.docs.pyansys.com/version/stable/

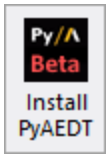
Installing PyAEDT adds three items to the **Tools** > **Toolkit** menu and to the **Automation** tab:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Console** – launches the PyAEDT console.
- **Jupyter Notebook** – launches Jupyter Notebook (a computational notebook) in an internet browser.
- **Run PyAEDT Script** – launches a file browser allowing you to select a Python script to run via PyAEDT.

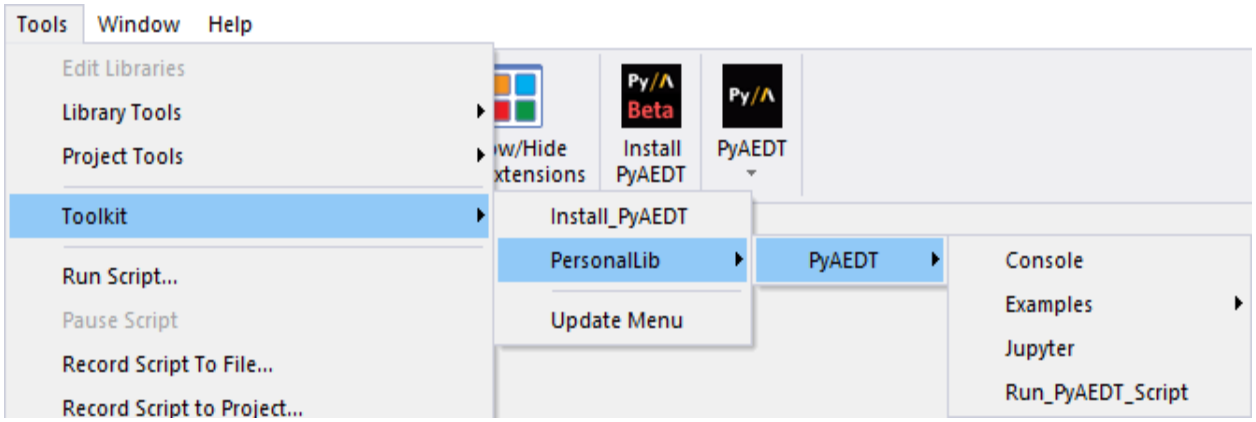
To install PyAEDT:

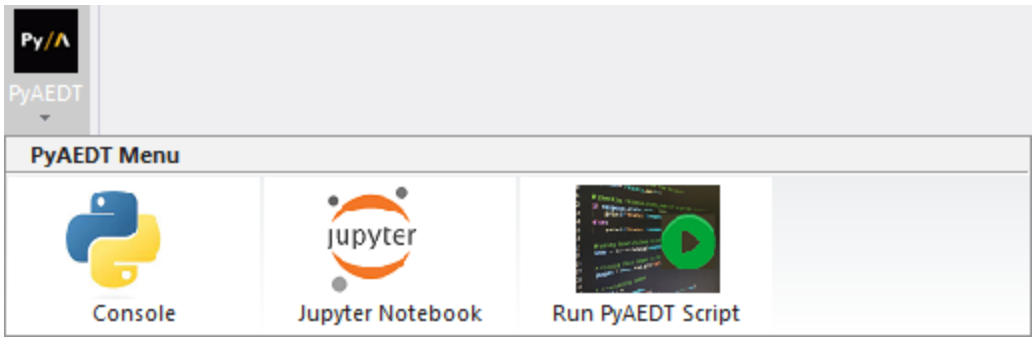
- From the **Automation** tab, click **[Beta] Install PyAEDT**.



A web browser launches, and takes you to detailed installation instructions.

When installation is complete, the **Tools** > **Toolkit** menu and the **Automation** tab update to display PyAEDT menu options:





Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

25 - Twin Builder Design Conventions

The following sections describe the conventions that you need to use when working with Twin Builder designs.

- Names of Components and Variables
- Parameter Qualifiers
- Parameter Types
- Predefined Variables
- Predefined Constants
- Equations, Expressions, and Variables
- Standard Mathematical Functions
- Unit Suffixes of Numeric Data
- SI Units
- Unit Handling
- Unit Types

Names of Components and Variables

User-defined names can be given to components, blocks, states, time functions, characteristics, nodes, ports, and standard variables. Names may contain any combination of uppercase and lowercase letters (A-Z, a-z), the numerals 0 through 9, and underscores – and can have a maximum of 50 characters.

> **Note:**
>
> - In general, user-defined names are case-sensitive. However, names of components and variables of VHDL-AMS models are case-insensitive and all uppercase letters are converted to lowercase.
> - Do not duplicate names (for example, **R1** and **r1**). Duplicate instance names result in netlist errors when attempting to compile a circuit prior to analysis.
> - The first character of a name must always be a letter.
> - Vowel mutations (for example, umlauts) are not allowed.
> - Spaces are not allowed.

The following predefined variable types are not allowed for names:

- SML notation keywords.
- Simulation parameters.
- System variables.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Predefined Variables

Parameter Qualifiers

Components are characterized by various physical quantities. A resistor, for example, is represented by current (I) and voltage (V) in the simulation. System variables may be accessed by reading (to use the actual quantity in an expression or to create an output) or writing (to influence quantities). Use the following syntax to access component variables:

ComponentName.Qualifier

Computations and outputs require access to system variables. The form and number of the qualifier depend on the corresponding component.

> **Note:**
>
> All qualifiers are case sensitive and must use capital letters (for example, **R.V**, *not***R.v**).

Qualifier Lists

The following tables display the most common Twin Builder qualifiers. The System Ouputs table lists qualifiers that are read-only. The Component Parameters table lists qualifiers that can be both read and written.

System Outputs

Notation	Description
ComponentName.V	component voltage, node potential (read)
ComponentName.I	component current (read)
ComponentName.dV	derivative of the component voltage (read)
ComponentName.dI	derivative of the component current (read)
ComponentName.VAL	block/time function/characteristic output signal (read)
ComponentName.ST	yields the currently valid marking status of a state true-1-marked-active false-0-unmarked-inactive
ComponentName.P	component pressure, node pressure

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Notation	Description
ComponentName.Q	component flow rate
ComponentName.C	component hydraulic capacitance
ComponentName .CHARGE	component charge
ComponentName.MMF	component magnetomotive force
ComponentName.FLUX	component magnetic flux
ComponentName.S	component displacement
ComponentName.F	component force
ComponentName.V	component velocity
ComponentName.PHI	component angle
ComponentName .TORQUE	component torque
ComponentName .OMEGA	component angular velocity
ComponentName.T	component temperature
ComponentName.H	component heat flow

Component Parameters

Notation	Description
ComponentName .EMF	component electromotive force
ComponentName .R	component resistance (read/write)
ComponentName .C	component capacitance (read/write)
ComponentName .L	component inductance (read/write)
ComponentName .G	component conductivity (read/write)
ComponentName .I0	component initial current (read/write only at simulation start)
ComponentName .V0	component initial voltage (read/write only at simulation start)

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Notation	Description
ComponentName.CTRL	control signal (read/write)
ComponentName.UL	upper limit (read/write)
ComponentName.LL	lower limit (read/write)
ComponentName.FREQU	frequency of a function (read/write)
ComponentName.TPERIO	cycle duration of a function (read/write)
ComponentName.AMPL	amplitude of a function (read/write)
ComponentName.INPUT	block input signal (read/write)
ComponentName.TS	sampling time of the block sampling function (read/write)
ComponentName.QUANT	control quantity (read/write)
ComponentName.CH	component characteristic (read/write)
ComponentName.FILE	file name (read/write)
ComponentName.VALUE	source pressure, flow rate, magnetomotive force, magnetic flux, displacement, force, velocity, angle, torque, angular velocity, temperature, heat flow
ComponentName.K	component hydraulic conductance
ComponentName.VOL	component fluid volume
ComponentName.B	component bulk modulus
ComponentName.P0	component initial pressure (read/write only at simulation start)
ComponentName.RHO	component fluid density
ComponentName.DIA	component diameter

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Notation	Description
ComponentName.LEN	component length
ComponentName.Q0	component initial flow rate (read/write only at simulation start)
ComponentName.K	component magnetic resistance
ComponentName.W	component number of turns
ComponentName.FLUX0	component initial magnetic flux (read/write only at simulation start)
ComponentName.C	component spring rate
ComponentName.DAMPING	component damping coefficient
ComponentName.M	component mass
ComponentName.S0	component initial position (read/write only at simulation start)
ComponentName.V0	component initial velocity (read/write only at simulation start)
ComponentName.SUL	component upper position limit
ComponentName.SLL	component lower position limit
ComponentName.J	component moment of inertia
ComponentName.PHI0	component initial angle (read/write only at simulation start)
ComponentName.OMEGA0	component initial angular velocity (read/write only at simulation start)
ComponentName.PHIUL	component upper angle limit
ComponentName.PHILL	component lower angle limit
ComponentName.K	component thermal resistance

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Notation	Description
ComponentName .C_TH	component thermal capacitance
ComponentName .T0	component initial temperature (read/write only at simulation start)

Parameter Types

The following table summarizes the parameter types used in Twin Builder.

Type	Description	Value Utilization	Accepted Value Format
Common Type (Name.X)	General parameters and quantities Example: E1.EMF	All expressions are interpreted with their actual value to define a parameter.	All numerical values (constant with or without unit suffix), simulation parameters, variables, component parameters, mathematical, or logical expressions. A logical expression provides only the value ‘1’ (TRUE) or ‘0’ (FALSE). Examples: 10k, 5K, -1E-3, 0.003, 20MEG Tend, Hmin, SECM.ITERAT var1, var2, _var3, var_4, Var_4 R23.I, C17.V, E4.EMF, GZ1.VAL 10*t+var1-INTEG(var2) delay>=2.5m*var1
Control Type (Name.CTRL)	Control input of switching devices. Example: S1.CTRL	The expression value tests if **>0**, **<0**, or **= 0**. See also **Common Type**.	All numerical values (constant with or without unit suffix), variables, component parameters, mathematical, or logical expressions. A logical expression provides only the value ‘1’ (TRUE) or ‘0’ (FALSE). Examples: See **Common Type**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Type	Description	Value Utilization	Accepted Value Format
Initial Value Type (Name.X0)	General initial parameters and quantities. Example: IM1.IA10	Initial values are set only once at simulation start. See also **Common Type**.	All numerical values (constant with or without unit suffix), simulation parameters, variables, component parameters, mathematical, or logical expressions. A logical expression provides only the value ‘1’ (TRUE) or ‘0’ (FALSE). Examples: See **Common Type**.
Logical Expression Type	Parameters for switching conditions. Example: TRANS1.TRC	Logical expressions are interpreted as **TRUE** if the provided value is ‘1’. Otherwise, the value is **FALSE**.	Boolean expressions. Examples: (V1.EMF >=0), (AM1.I = -3), (RHYD1.Q < 2.4)
Quantity Type (Name.QUANT)	Control quantity of controlled sources. Examples: ExNL.CTRL IxNL.CTRL EPOLY.CTRL	All quantities are interpreted with their actual value to define a parameter.	The type accepts only voltage and current of voltmeters, ammeters, or wattmeters. The values cannot be assigned by a variable. Examples: VM1.V, AM2.I, WM3.V, WM3.I
Characteristic Type (Name.CH)	Characteristics of non-linear elements. Examples: RNL.CH EINL.CH	For a given X value, the Y value is determined. The LOOKUP function provides characteristic values in equations.	The type accepts only the output value of a characteristic component. The values cannot be assigned by a variable. Examples: EQUL1.VAL, XY1.VAL
File Type (Name.FILE)	File name for dataset-based characteristics.	For a given X value, the Y value is	The type accepts only a name referring to a characteristic file (.mdx, .xls, xlsx, .csv, .txt). The

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Type	Description	Value Utilization	Accepted Value Format
	Example: XY1.FILE	selected.	value cannot be a variable. Example: C:\release\diode.mdx

> **Note:**
>
> This table does not apply to user-defined models (UMODEL) if you choose to integrate values directly.

Predefined Variables

The simulator uses intrinsic variables for internal computation. All predefined variables are case insensitive.

> **Warning:**
>
> Do not use predefined variables for names (including port names) in a model description. If these variables are used in a model description, unexpected effects or an error message result.

System constants (read)	F, TIME, H, PI, TRUE, FALSE, SECM.ITERAT, FSTEP
General simulation parameters (read/write)	TEND, HMIN, HMAX, TEMP, FSTART, FEND

Pre-defined Constants

The simulator provides natural and mathematical constants that can be used in mathematical expressions within component dialog boxes or SML descriptions.

The following table shows the available constants and their corresponding symbols:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Constant	Legacy Name[1]	Value	Unit	Description	Symbol
abs0	PHYS_T0	-273.15	°C	Absolute Zero	ϑ
boltz	PHYS_K	1.38066 10^{-23}	J/K	Boltzmann constant	k_B
c0	PHYS_C	299792458	m/s	Speed of light	c
e0	PHYS_E	8.85419 10^{-12}	C²•Jm	Permittivity of vacuum	ε_0
elecq	PHYS_Q	1.60217733 10^{-19}	C	Elementary charge	e
eta	N/A	376.730313461	Ohm	Impedance of vacuum	η
false		0		Boolean False	
g0	PHYS_G	9.80665	m/s²	Acceleration due to gravity	g
mathE	MATH_E	2.718281828	[/]	Euler number	E
pi	MATH_PI	3.1415926535898	[/]	Pi	π
planck	PHYS_H	6.6260755 10^{-34}	Js	Planck constant	h
true		1		Boolean True	
u0	PHYS_MU0	1.25664 10^{-06}	T²m³/J	Permeability of vacuum	μ_0

1. For backward compatibility, #defines of these names will be replaced by their values during translation of a model definition - unless the #define line is still present in the model definition.

Equations, Expressions, and Variables

Equations consist of *operands* and *operators*. An *operand* can be any number or variable name. An *operator* compares or assigns a value.

In *expressions* you can create and use *variables* as often as you want. A *variable* is defined when the *variable* name is used in an *expression* or for a parameter value within a component dialog box. You do not need to define the *variable* in a specific assignment unless you want it to have a defined initial value. For example, in the following equation:

Z:=Y+X

- **X**, **Y**, and **Z** are the *operands*
- **:=** and **+** are the *operators*.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If *operands* are complex numbers (for example, in an AC simulation), the comparison *operators* (**<**, **>**, **<=**, **>=**) consider only the real part.

Operators

Assignment operators	:=	Assignment
	##	Delay operator combined with the action type DEL
	$$	Schedule operator combined with the action type SCHED
Arithmetic operators	*	Multiplication
	/	Division
	+	Addition
	–	Subtraction
	**	Power 7**4 = 7^4 = 2401
Comparison operators without synchronization	<	Less than
	>	Greater than
	!=	Not equal to
Comparison operators with synchronization	This operator type forces the simulator to synchronize on the condition with the minimum step width.	
	<=	Less than or equal to
	>=	Greater than or equal to
	==	Equal to (compares two values for equality)
Logic operators (must use a space before and after the operator)	&&	Logical AND (conjunction)
	||	Logical OR (disjunction)
	!	Logical NOT (negation)

Note:

For backward compatibility the operators: **AND**, **OR**, **NOT**, **<>**, **><**, **=** are still recognized in pure SML text.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Standard Mathematical Functions

Mathematical functions consist of the function name and one or two arguments. An argument can be any number or variable name. A mathematical function applies the function, which it represents, to the argument(s).

r:=FCT(x,y),r:=FCT(z)

In the example above:

- **x**, **y**, and **z** are arguments.
- **z** is a complex number.
- **FCT** is the function name.
- **r** is the result.

If the argument(s) are complex numbers (for example in an AC simulation), the functions RAD, DEG, DEGEL, INT, REM, and LOOKUP consider only the real part.

> **Warning:**
>
> When entering these functions, do not leave spaces between the function arguments and the open parenthesis mark. For example: **SIN(x)** *not* **SIN (x)**.

> **Note:**
>
> For backward compatibility, the functions ARCSIN, ARCCOS, ARCTAN, SQU, LOOKUP, INTEG, ROOT(x), RAD, DEG, and MOD(x) are still recognized in pure SML text.

The following tables list the standard mathematical functions included in Twin Builder:

Trigonometric Functions

> **Note:**
>
> When defining arguments for trigonometric functions, you must consider poles to avoid potential errors during a simulation.

Notation	Description	Example
SIN(x)	Sine, x[rad]	SIN(PI/6)=0.5
COS(x)	Cosine, x[rad]	COS(2•PI/3)=-0.5

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Notation	Description	Example
TAN(x)	Tangent, x[rad]	TAN(PI/4)=1
ASIN(x)	Arc sine [rad]	ASIN(0.5)=0.524=PI/6
ACOS(x)	Arc cosine [rad]	ACOS (0.5)=1.0471=PI/3
ATAN(x)	Arc tangent [rad]	ATAN(1)=0.785=PI/4
ATAN2(x,y) ATAN2=ATAN (y/x)	Arc tangent2 [rad] r=0 if x=0 and y=0; $-\pi \leq r \leq \pi$	ATAN2(.25,1)= ATAN(4)=1.325
SINH(x)	Hyperbolic sine	SINH(1)=1.175201
COSH(x)	Hyperbolic cosine	COSH(1)=1.54308
TANH(x)	Hyperbolic tangent	TANH(1)=0.761594
ASINH	Hyperbolic Arcsine	ASINH(1.175201)=1
ACOSH	Hyperbolic Arccosine	ACOSH(1.54308)=1
ATANH	Hyperbolic Arctangent	ATANH(0.761594)=1

Arithmetic Functions

Notation	Description	Example
SQR(x)	Square.	SQR (16)=16^2=256
SQRT(x)	Square root.	SQRT (9)=$\sqrt[2]{9}$=3
ROOT(x,y)	n-th Root.	ROOT (27,3)=$\sqrt[3]{27}$=3
SDT(x,y)	Integration of a variable from the function call until to the simulation end. y=integration method NOTE: Usable in property values (during simulation) but not in postprocessing.	SDT (var1)=$\int$var1 dt **Integration methods:** 0=Euler (explicit) 1=Euler (implicit) 2=Trapezoidal (implicit) 3=Auto mode (setting taken

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Notation	Description	Example
		from transient options dialog box)
DDT(x)	Derivative of a variable in time NOTE: Usable in property values (during simulation) but not in postprocessing.	DDT (var1)=dvar1/dt
MAX(x1,x2)	Returns the greater of two values	MAX(1,5)=5
MIN(x1,x2)	Returns the lesser of two values	MIN(1,5)=1
MOD(x,y)	Modulus.	MOD (370,60)=10
LOOKUP(x,y) x=Characteristic name y=X value	Access function to a characteristic.	LOOKUP (XY1.VAL,5)= Y value of the characteristic XY1 for the X value 5
NOZ(x,y)	Not zero. Ensures that the minimum absolute value of x is restricted to the absolute value specified by y. Default for y if not specified is 1e-12. This function is useful in avoiding divide-by-zero conditions introduced either by normal expressions or the partial derivative of these expressions. For example: 1/(VM1.V) would better be written as 1/(noz(VM1.V)) to avoid division by zero when VM1.V is zero. sqrt(x.V) would be better written as sqrt(noz(x.V)) if a partial derivative of the expression is to be considered because the partial derivative of sqrt(x.V) with respect to x.V is 1/2*(sqrt(x.V)). NOTE: Usable in property values (during simulation) but not in postprocessing.	noz(x, 1e-13) = x if abs(x) > 1e-13 = 1e-13 if abs(x) < 1e-13 and x is >=0 = -1e-13 if abs (x) < 1e-13 and x is <0 If y is negative, then the absolute value of y is used. noz(x, -1e-13) = x if abs(x) > 1e-13 = 1e-13 if abs(x) < 1e-13 and x is >=0 = -1e-13 if abs (x) < 1e-13 and x is <0

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Exponential Functions

Notation	Description	Example
EXP(x)	Exponential function.	EXP(5)=e^5=148.41
LN(x)	Natural logarithm.	LN(3)=$\log_e 3$=1.099
LOG (x,y)	Common logarithm.	LOG (7,4)=$\log_4 7$=1.403

Complex Functions

Notation	Description	Example
ABS(x)	Absolute value.	ABS(-8.5)=|-8.5|=8.5
RE(z)	Real part	RE(z)=5
IM(z)	Imaginary part	IM(z)=3
ARG(z)	Returns a phase value (argument) from AC analysis results, which are complex values, in degrees. For example, given: $z=a+bi=r(\cos\varphi+i\bullet\sin\varphi)=r\bullet e^{i\varphi}$ $z=5+3i=5.83(\cos 30.96°+i\bullet\sin 30.96°)$	ARG (z)=30.96°
MAG(z)	Magnitude of a complex number.	MAG(Z)=5

Conversion Functions

Notation	Description	Example
ANG_ RAD(z)	Returns a phase value from AC analysis results in radians.	ANG_RAD (5+3i)=0.5404
ANG_ DEG(z)	Returns a phase value from AC analysis results, which are complex values, in degrees.	ANG_DEG (5+3i)=30.96°
DEGEL(x [,y]); y=1	Conversion from electrical angle (in radians) to seconds with respect to Hz.	DEGEL (PI,50)=0.010

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note:

The special conversion functions gel50, gel16, gel, mod360, mod60 have been replaced in Simplorer7 and later by generic functions degel(x,y) and mod(x,y).

Rounding Functions

Notation	Description	Example
SGN(x)	Sign dependent value (-1, 0, 1). r=0 if z=0, 1 if Re(z)>0 or (Re(z)=0 and Im(z)>0), -1 otherwise.	SGN(3)=1; SIGN(0)=0; SIGN(-3)=-1
INT(x)	Integer part of a value.	INT(2.5)=2
REM(x)	Returns the fractional part of a decimal number such that rem(x) = x-int(x) Syntax: rem(x)	REM(2.5)=0.5

Conditional Functions

Notation	Description	Example

IF (condition1)
{ var:=1; }
[ELSE IF (condition2)
{ var:=2; }
ELSE
{ var:=3; }]
	IF-ELSE returns equations (which are valid for FML and FML_INIT components).
	If-Else function to perform operations dependent on conditions.
	The ELSE IF and ELSE statements can be omitted.
		IF
		(t>=1)
		{
		var:=1;
		}
		[ELSE
		IF
		(t>=2)
		{
		var:=2;
		}
		ELSE
		{
		var:=3;
		}

IF (condition, True-expression, False-expression)	The implicit IF statement is an expression and can only be used on the right side of an equation or in value expressions.	

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Predefined Constants

Operators

Unit Suffixes of Numeric Data

Numeric data can be entered in component dialog boxes and in Twin Builder‘s editors, using the following unit extensions:

Suffix	Value		SML		Examples
tera	10^{12}	E12	t	TER	*5e12, 5t, 5ter*
giga	10^{9}	E9	g	GIG	*1.4e9, 1.4g, 1.4gig*
mega	10^{6}	E6		MEG	*-1.4E6, -0.3meg, -0.3MEG*
kilo	10^{3}	E3	k	KIL	*1000, 1e3, 1k, 1kil*
milli	10^{-3}	E-3	m	MIL	*0.0105, 1.05E-2, 10.5M, 10.5MIL*
micro	10^{-6}	E-6	u	MIC	*0.000005, 5e-6, 5u, 5mic*
nano	10^{-9}	E-9	n	NAN	*40E-9, 40n, 40nan*
pico	10^{-12}	E-12	p	PIC	*100E-12, 100P, 100PIC*
femto	10^{-15}	E-15	f	FEM	*9E-15, 9F, 9FEM*

Note:

- The **comma** is reserved for separating parameters in lists.
- The **period** (**dot**) is reserved as a decimal point.
- “**M**” is interpreted as 10^{-3}, *not* as 10^{6}.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SI Units

All units used in Twin Builder are derived from the SI system of units.

Quantity	Unit Name	Symbol
Length	Meters	m
Mass	Kilograms	kg
Time	Seconds	s
Electrical current intensity	Amperes	A
Temperature	Kelvins	K
Voltage (derived SI unit)	Volts	V

However, you can also use the unit handling feature of Twin Builder to use non-SI units for designs. Non-SI units are converted to the expected SI units.

Related Topics

Unit Handling

Unit Types

Unit Handling

Use the unit handling feature to enter component parameter values in multiples of standard SI units such as *millivolts*, *nanoamperes*, and *kilometers*, as well as in non-SI units such as *pounds-per-square-inch*, *degrees Fahrenheit*, and *feet*. This eliminates the need for error-prone unit conversions and reduces the time needed to calculate component parameters.

Each component parameter that is a physical quantity can be assigned an expected unit of measure, which is the unit of measure for the parameter value used during simulation.

Note:

- Twin Builder *internal* models have predefined expected units for each physical parameter.
- The parameters of *user-defined* models such as C-Models can be assigned expected units when the user-defined models are created.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Each parameter also is provided with a set of additional units that can be applied to the same physical quantity.

For example, Twin Builder's force source component has an expected unit of *Newtons* and an associated set of additional units that includes *milli-Newtons*, *PoundsForce*, and *dynes*.

When an instance of a component such as the force source is placed on a schematic, its parameter values can be edited in its **Properties** dialog box. The units associated with the parameters also can be changed. The units are located in combo boxes next to the text fields containing each parameter's value. Changing the unit for a parameter is as simple as selecting a new unit in the combo box for the parameter's unit.

When the component is simulated, Twin Builder converts quantities expressed in the newly chosen units to equivalent quantities of the expected units. Thus, you can express component parameters in units from different systems of measurement without affecting the accuracy of the simulation results.

Unit Types

The following table lists the supported unit types for all Ansys Electromagnetics products. Twin Builder supports a subset of the unit types shown in this list. To confirm that a particular unit is supported in Twin Builder, consult the default-value drop-down menus for the various unit specifications listed in the Default Units tab of the **General Options** dialog box.

Unit Type	String Representation	Description
Acceleration	m_per_s2	Meter per second-squared
Acceleration	cm_per_s2	Centimeter per second-squared
Acceleration	in_per_s2	Inch per second-squared
Acceleration	mm_per_s2	Milli-Meter per second-squared
Admittance	Sie	Siemens
Admittance	megSie	Mega-Siemens
Admittance	uSie	Micro-Siemens
Admittance	mSie	Milli-Siemens
Admittance	kSie	Kilo-Siemens
Admittance	mho	Mhos
AIGB	F_per_ghalfs_per_m	(Farad per gram) root second per meter
AIGB	Fs2_per_ghalf_per_m	(Farad second squared per gram) root per meter

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
AmountOfSubstance	mol	Mol
AmountOfSubstance	umol	Micro-Mol
AmountOfSubstance	nmol	Nano-Mol
AmountOfSubstance	mmol	Milli-Mol
AmountOfSubstance	kmol	Kilo-Mol
Angle	rad	Radian
Angle	deg	Degree
Angle	degsec	Second
Angle	degmin	Minute
AngleCoefficient	per_rad	Per radian
AngleCoefficient	V_per_V_per_deg	Volt per Volt per degree
AngleCoefficient	per_deg	Per degree
AngleCoefficient	V_per_V_per_rad	Volt per Volt per radian
AngularAcceleration	rad_per_s2	Radian per square-second
AngularAcceleration	deg_per_s2	Degrees per square-second
AngularAcceleration	per_s2	Revolutions per square-second
AngularDamping	Nms_per_rad	Newton-meter-second per radian
AngularDamping	dNms_per_rad	Deci-Newton-meter-second per radian
AngularDamping	kNms_per_rad	Kilo-Newton-meter-second per radian
AngularJerk	rad_per_sec2	Radian per second-cubed
AngularJerk	deg_per_sec2	Degrees per second-cubed
AngularMomentum	kgm2_per_sec	Kilo-gram square-meter per second
AngularMomentum	nms	Newton-meter-second
AngularSpeed	deg_per_sec	Degrees per second
AngularSpeed	deg_per_min	Degrees per minute
AngularSpeed	deg_per_hr	Degrees per hour
AngularSpeed	rad_per_sec	Radians per second
AngularSpeed	rad_per_min	Radians per minute

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
AngularSpeed	rad_per_hr	Radians per hour
AngularSpeed	rpm	Revolutions per minute
AngularSpeed	rev_per_min	Revolutions per minute
AngularSpeed	rev_per_sec	Revolutions per second
AngularSpeed	per_second	Revolutions per second
AngularStiffness	nm_per_rad	Newton-meter per radian
AngularStiffness	nm_per_deg	Newton-meter per degree
AngularWindage	nms2_per_rad2	Newton-meter second squared per radian squared
AngularWindage	kgm2_per_rad2	Kilogram meter squared per radian squared
Area	m2	Square-Meter
Area	km2	Square-Kilometer
Area	ft2	Square foot
Area	in2	Square Inch
Area	mm2	Square-Millimeter
Area	cm2	Square-Centimeter
Area	um2	Micro-meter squared
AreaCoefficient	per_m2	Per square-meter
AreaCoefficient	per_cm2	Per square-centimeter
ArealFlowRate	m2_per_s	Square-meter per second
ArealFlowRate	m2_per_min	Square-meter per minute
ArealFlowRate	m2_per_hour	Square-meter per hour
AreaPerPower	m2_per_W	Square-meter per Watt
AreaPerPower	m2_per_Js	Square-meter per Joule-second
AreaPerVoltage	m2_per_V	Square-meter per volt
AreaPerVoltage	Am2_per_W	Ampere-square-meter per Watt
AreaPerVoltage	Am2_per_kW	Ampere-square-meter per Kilowatt
AreaPerVoltage	m2_per_kV	Square-meter per Kilovolt
AreaPerVoltageTemperature	m2_per_Vkel	Square-meter per Volt-Kelvin
AreaPerVoltageTemperature	Am2_per_Wkel	Ampere-square-meter per Watt-Kelvin

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
BIGB	F_per_ghalfs_ per_Vm	(Farad per gram) root second per Volt-meter
BIGB	Fs2_per_ghalf_ per_Vm	(Farad second squared per gram) root per Volt-meter
Capacitance	fF	Femto-Farad
Capacitance	farad	Farad
Capacitance	pf	Pico-Farad
Capacitance	uf	Micro-Farad
Capacitance	nf	Nano-Farad
Capacitance	mf	Milli-Farad
CapacitancePerArea	F_per_m2	Farad per square-meter
CapacitancePerArea	nF_per_m2	Nano-Farad per square-meter
CapacitancePerArea	uF_per_m2	Micro-Farad per square-meter
CapacitancePerArea	F_per_cm2	Farad per square-cent-meter
CapacitancePerArea	mF_per_m2	Milli-Farad per square-meter
CapacitancePerAreaPerVoltage	F_per_Vm2	Farad per Volt and square-meter
CapacitancePerAreaPerVoltage	mF_per_Vm2	Milli-Farad per Volt-square-meter
CapacitancePerAreaPerVoltage	pF_per_Vm2	Pico-Farad per Volt-square-meter
CapacitancePerAreaPerVoltage	nF_per_Vm2	Nano-Farad per Volt-square-meter
CapacitancePerAreaPerVoltage	uF_per_Vm2	Micro-Farad per Volt-square-meter
CapacitancePerLength	F_per_m	Farad per meter
CapacitancePerLength	pF_per_m	Pico-Farad per meter
CapacitancePerLength	nF_per_m	Nano-Farad per meter
CapacitancePerLength	mF_per_m	Milli-Farad per meter
CapacitancePerLength	uF_per_m	Micro-Farad per meter
CapacitanceTemperatureCoeff	F_per_Kel	Farad per Kelvin
CapacitanceTemperatureCoeff	pF_per_Cel	Pico-Farad per degree Celsius
CapacitanceTemperatureCoeff	pF_per_Kel	Pico-Farad per Kelvin
CapacitanceTemperatureCoeff	mF_per_Cel	Milli-Farad per degree Celsius

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
CapacitanceTemperatureCoeff	pF_per_Fah	Pico=Farad per degree Farenheit
CapacitanceTemperatureCoeff	nF_per_Kel	Nano-Farad per Kelvin
CapacitanceTemperatureCoeff	mF_per_Kel	Milli-Farad per Kelvin
CapacitanceTemperatureCoeff	F_per_Cel	Farad per degree Celsius
CapacitanceTemperatureCoeff	mF_per_Fah	Milli-Farad per degree Farenheit
CapacitanceTemperatureCoeff	nF_per_Cel	Nano-Farad per degree Celsius
CapacitanceTemperatureCoeff	F_per_Fah	Farad per degree Farenheit
CapacitanceTemperatureCoeff	uF_per_Kel	Micro-Farad per Kelvin
CapacitanceTemperatureCoeff	nF_per_Fah	Nano-Farad per degree Farenheit
CapacitanceTemperatureCoeff	uF_per_Cel	Micro-Farad per degree Celsius
CapacitanceTemperatureCoeff	uF_per_Fah	Micro-Farad per degree Farenheit
Charge	Charge	Coulomb
Charge	Ahour	Ampere hour
Charge	uC	Micro-Coulomb
Charge	mC	Milli-Coulomb
Charge	nC	Nano-Coulomb
Charge	kC	Kilo-Coulomb
Charge	As	Ampere second
Compliance	m_per_N	Meter per Newton
Compliance	in_per_lbf	Inch per pound-force
Compliance	cm_per_N	Centi-meter per Newton
Conductance	fSie	Femto-Siemen
Conductance	nSie	Nano-Siemen
Conductance	pSie	Pico-Siemen
Conductance	Sie	Siemens
Conductance	megSie	Mega-Siemens
Conductance	uSie	Micro-Siemens
Conductance	mSie	Milli-Siemens
Conductance	kSie	Kilo-Siemens

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Conductance	mho	Mhos
Conductance	perohm	Reciprocal Ohm
Conductance	ApV	Ampere per Volt
ConductancePerLength	S_per_m	Siemens per meter
ConductancePerLength	mS_per_m	Milli-Siemens per meter
ConductancePerLength	kS_per_m	Kilo-Siemens per meter
ConductancePerLength	uS_per_m	Micro-Siemens per meter
ConductancePerLength	cS_per_m	Centi-Siemens per meter
Current	A	Ampere
Current	fA	Femto-Ampere
Current	kA	Kilo-Ampere
Current	mA	Milli-Ampere
Current	nA	Nano-Ampere
Current	pA	pico-Ampere
Current	uA	MIcro-Ampere
Current	a	Ampere
Current	ua	Micro-Ampere
Current	na	Nano-Ampere
Current	ma	Milli-Ampere
Current	ka	Kilo-Ampere
Current	C_per_s	Coulomb per second
CurrentChangeRate	A_per_s	Ampere per second
CurrentChangeRate	A_per_hour	Ampere per hour
CurrentChangeRate	mA_per_s	Milli-Ampere per second
CurrentChangeRate	kA_per_s	Kilo-Ampere per second
CurrentChangeRate	uA_per_s	Micro-Ampere per second
CurrentChangeRate	A_per_min	Ampere per minute
CurrentDensity	A_per_m2	Ampere per square-meter
CurrentDensity	uA_per_m2	Micro-Ampere per square-meter
CurrentDensity	mA_per_cm2	Milli-Ampere per square-centi-meter

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
CurrentDensity	mA_per_m2	Milli-Ampere per square-meter
CurrentDensity	A_per_cm2	Ampere per square-centi-meter
CurrentDensity	uA_per_cm2	Micro-Ampere per square-centi-meter
CurrentGain	A_per_A	Ampere per Ampere
CurrentGain	A_per_mA	Ampere per Milli-Ampere
CurrentGain	mA_per_A	Milli-Ampere per Ampere
CurrentLengthPerVoltage	Am_per_V	Ampere-meter per Volt
CurrentLengthPerVoltage	mAm_per_V	Milli-Ampere-meter per Volt
CurrentLengthPerVoltage	uAm_per_V	Micro-Ampere-meter per Volt
CurrentPerCharge	A_per_C	Ampere per Coulomb
CurrentPerCharge	A_per_Ahour	Ampere per Ampere-hour
CurrentPerCharge	A_per_As	Ampere per Ampere-second
CurrentPerIrradiance	A_per_W_per_m2	Ampere per Watt per meter-squared
CurrentPerIrradiance	kA_per_W_per_m2	Kilo-Ampere per Watt per meter-squared
CurrentPerIrradiance	uA_per_W_per_m2	Micro-Ampere per Watt per meter-squared
CurrentPerIrradiance	mA_per_W_per_m2	Milli-Ampere per Watt per meter-squared
CurrentPerLength	A_per_m	Ampere per meter
CurrentPerLength	mA_per_m	Milli-Ampere per meter
CurrentPerLength	kA_per_m	Kilo-Ampere per meter
CurrentPerLength	uA_per_m	Micro-Ampere per meter
CurrentPerTemperature2_half	A_per_Kel2half	Ampere per Kelvin to the power of 2.5
CurrentPerTemperatureCubed	A_per_Kel3	Ampere per Kelvin-cubed
CurrentPerTemperatureDiffCubed	A_per_diff_Kel3	Ampere per Kelvin-cubed
CurrentPerTemperatureDiffCubed	A_per_Cel3	Ampere per degree Celsius cubed
CurrentSquaredTime	A2s	Ampere-squared second
CurrentSquaredTime	mA2s	Milli-Ampere-squared second

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
CurrentTemperatureCoeff	A_per_Kel	Ampere per Kelvin
CurrentTemperatureCoeff	A_per_Cel	Ampere per degree Celsius
CurrentTemperatureCoeff	mA_per_Kel	Milli-Ampere per Kelvin
CurrentTemperatureCoeff	mA_per_Fah	Milli-Ampere per degree Farenheit
CurrentTemperatureCoeff	kA_per_Kel	Kilo-Ampere per Kelvin
CurrentTemperatureCoeff	A_per_Fah	Ampere per degree Farenheit
CurrentTemperatureCoeff	uA_per_Kel	Micro-Ampere per Kelvin
CurrentTemperatureCoeff	uA_per_Cel	Micro-Ampere per degree Celsius
CurrentTemperatureCoeff	mA_per_Cel	Milli-Ampere per degree Celsius
Damping	Ns_per_m	Newton second per meter
Damping	kNs_per_m	Kilo-Newton second per meter
Damping	mNs_per_m	Milli-Newton second per meter
Damping	cNs_per_m	Centi-Newton second per meter
Damping	dNs_per_m	Deci-Newton second per meter
Density	kg_per_m3	Kilogram per Cubic-Meter
Density	g_per_cm3	Gram per Cubic-Centimeter
Density	kg_per_l	Kilo-Gram per Liter
Density	kg_per_dm3	Kilo-gram per cubic-deci-meter
Density	g_per_l	Gram per Liter
Displacement	meter	Meter
Displacement	mm	Millimeter
Displacement	km	Kilometer
Displacement	cm	Centimeter
Displacement	um	Micrometer
Displacement	nm	Nanometer
Displacement	micron	Microns
Displacement	dm	Decimeter
Displacement	pm	Picometer
Displacement	fm	Femtometer
Displacement	mil	Milli-inch

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Displacement	in	Inch
Displacement	ft	Foot (am.)
Displacement	yd	Yard (am.)
Displacement	mile	Mile (am.)
Displacement	mileTerr	Terrestrial mile
Displacement	mileNaut	Nautical mile
Displacement	lightyear	Light year
ElectricFieldStrength	V_per_meter	Volt per meter
ElectricFieldStrength	v_per_meter	Volt per meter
ElectricFieldStrength	v_per_cm	Volt per centimeter
ElectricFlux	C	Coulomb
ElectricFlux	uC	Micro-Coulomb
ElectricFlux	nC	Nano-Coulomb
ElectricFlux	mC	Milli-Coulomb
ElectricFlux	As	Ampere-second
ElectricFluxDensity	C_per_m2	Coulomb per square-meter
ElectricFluxDensity	nC_per_m2	Nano-Coulomb per square-meter
ElectricFluxDensity	uC_per_m2	Micro-Coulomb per square-meter
ElectricFluxDensity	mC_per_m2	Milli-Coulomb per square-meter
Energy	J	Joule
Energy	Whour	Watt-hour
Energy	kJ	Kilo-Joule
Energy	eV	Electron-Volt
Energy	GJ	Giga-Joule
Energy	kWhour	kilo-Watt-hours
Energy	erg	Erg
Energy	Ws	Watt-second
Energy	megJ	Mega-Joule
Energy	uJ	Micro-Joule
Energy	mJ	Milli-Joule

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
EnergyDensity	J_per_m3	joules per meter cubed
EnergyDensity	kJ_per_m3	kilo joules per meter cubed
FluidicCapacitance	m3_per_Pa	Cubic-meter per Pascal
FluidicCapacitance	cm3_per_Pa	Centi-meter-cubed per Pascal
FluidicConductance	m3_per_Pas	Cubic-meter per Pascal-second
FluidicConductance	cm3_per_Pas	Cubic-centi-meter per Pascal-second
FluidicResistance	Pas_per_m3	Pascal-second per cubic-meter
FluidicResistance	Ns_per_m5	Newton-second per meter to the power of 5
Flux	Wb	Weber
Flux	weber	Weber
Flux	vh	Volt-hour
Flux	mx	Maxwell
Flux	vs	Volt-second
Force	fNewton	Femto-Newton
Force	pNewton	Pico-Newton
Force	nNewton	Nano-Newton
Force	uNewton	Micro-Newton
Force	mNewton	Milli-Newton
Force	kNewton	Kilo-Newton
Force	megNewton	Mega-Newton
Force	gNewton	Giga-Newton
Force	PoundsForce	Pounds-force
Force	newton	Newton
Force	megnewton	Mega-Newton
Force	unewton	Micro-Newton
Force	mnewton	Milli-Newton
Force	dyne	Dyne
Force	knewton	Kilo-Newton
Force	kp	Kilo-pound
Force	poundsForce	Pound-force

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Frequency	GHz	Gigahertz
Frequency	rps	Revolutions per second
Frequency	thz	Tera-Hertz
Frequency	Hz	Hertz
Frequency	kHz	Kilo-Hertz
Frequency	milliHz	Milli-Hertz
Frequency	MHz	Mega-Hertz
Frequency	per_sec	Per second
HeatFlow	W	Watt
HeatFlow	J_per_s	Joule per second
Illuminance	lx	Lux
Illuminance	klx	Kilo-Lux
Illuminance	meglx	Mega-Lux
Illuminance	lm_per_in2	Lumen per square-inch
Illuminance	W_per_m2	Watt per square-meter
Illuminance	W_per_cm2	Watt per square-centimeter
Illuminance	lm_per_cm2	Lumen per square-centimeter
Illuminance	lm_per_m2	Lumen per square-meter
Impedance	Ohm	Ohm
Impedance	megohm	Mega-Ohm
Impedance	mOhm	Milli-Ohm
Impedance	kOhm	Kilo-Ohm
Impedance	uOhm	Micro-Ohm
Impedance	Gohm	Giga-Ohm
Inductance	fH	Femto henri
Inductance	H	Henry
Inductance	mH	Milli-Henry
Inductance	nH	Nano-Henry
Inductance	pH	Pico-Henry
Inductance	uH	Micro-Henry
Inductance	h	Henry

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Inductance	uh	Micro-Henry
Inductance	nh	Nano-Henry
Inductance	mh	Milli-Henry
InductancePerLength	H_per_m	Henry per meter
InductancePerLength	pH_per_m	Pico-Henry per meter
InductancePerLength	nH_per_m	Nano-Henry per meter
InductancePerLength	mH_per_m	Milli-Henry per meter
InductancePerLength	uH_per_m	Micro-Henry per meter
Inertance	Pas2_per_m3	Pascal square-second per cubic-meter
Inertance	Ns2_per_m5	Newton square second per meter to power 5
Inertance	kg_per_m4	Kilo-gram per meter to the power of 4
Irradiance	irrad_W_per_m2	Watt per meter-squared
Irradiance	megW_per_m2	Mega-Watt per meter-squared
Irradiance	mW_per_m2	Milli-Watt per meter-squared
Irradiance	irrad_W_per_cm2	Watt per cent-meter-squared
Irradiance	kW_per_m2	Kilo-Watt per meter-squared
Irradiance	uW_per_m2	Micro-Watt per meter-squared
Irradiance	W_per_in2	Watt per inch-squared
Jerk	m_per_s3	Meter per second cubed
Jerk	cm_per_s3	Centi-Meter per second cubed
Jerk	nm_per_s3	Nano-Meter per second cubed
Jerk	in_per_s3	Inch per second-cubed
Jerk	um_per_s3	Micro-Meter per second cubed
Jerk	mm_per_s3	Milli-Meter per second cubed
Length	cm	Centimeter
Length	uin	Micro-inch
Length	meter	Meter
Length	mm	Milli-Meter
Length	km	Kilo-Meter

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Length	cm	Centi-Meter
Length	um	Micro-Meter
Length	nm	Nano-Meter
Length	micron	Microns
Length	dm	Deci-Meter
Length	pm	Pico-Meter
Length	fm	Femto-Meter
Length	mil	Milli-inch
Length	in	Inch
Length	ft	Foot (am.)
Length	yd	Yard (am.)
Length	mile	Mile (am.)
Length	mileTerr	Terrestrial mile
Length	mileNaut	Nautical mile
Length	lightyear	Light year
Length2PerVoltage2	m2_per_V2	Meter-squared per Volt-squared
LengthCoefficient	per_m	Per meter
LengthCoefficient	V_per_V_per_m	Volt per Volt per meter
LengthCoefficient	per_km	Per kilo-meter
LengthCoefficient	per_um	Per micro-meter
LengthCoefficient	per_in	Per inch
LengthCoefficient	per_cm	Per centi-meter
LengthCoefficient	V_per_V_per_in	Volt per Volt per inch
LengthCoefficient	per_mm	Per milli-meter
LengthPerVoltage	m_per_V	Meter per Volt
LengthPerVoltage	mm_per_V	Milli-Meter per Volt
LengthPerVoltage	km_per_V	Kilo-Meter per Volt
LengthPerVoltage	um_per_V	Micro-Meter per Volt
LengthPerVoltage	dm_per_V	Deci-Meter per Volt
LengthPerVoltage	cm_per_V	Centi-Meter per Volt
LengthPerVoltageRoot	m_per_Vhalf	Meter per Volt-root

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
LengthPerVoltageRoot	dm_per_Vhalf	Deci-Meter per Volt-root
LengthPerVoltageRoot	mm_per_Vhalf	Milli-Meter per Volt-root
LengthPerVoltageRoot	um_per_Vhalf	Micro-Meter per Volt-root
LengthPerVoltageRoot	km_per_Vhalf	Kilo-Meter per Volt-root
LengthPerVoltageRoot	cm_per_Vhalf	Centi-Meter per Volt-root
LuminousFlux	lm	Lumen
LuminousFlux	gm2_per_s3	Kilo-gram meter-squared per second-cubed
LuminousFlux	klm	Kilo-Lumen
LuminousFlux	mlm	Milli-Lumen
LuminousFlux	meglm	Mega-Lumen
LuminousIntensity	Cd	Candela
LuminousIntensity	GCd	Giga-Candela
LuminousIntensity	kCd	Kilo-Candela
LuminousIntensity	mCd	Milli-Candela
LuminousIntensity	megCd	Mega-Candela
MagFieldStrength	Oe	Oersted
MagFieldStrength	kOe	Kilo-Oersted
MagFieldStrength	A_per_meter	Ampere per meter
MagFieldStrength	kA_per_meter	Kilo-Ampere per meter
MagFieldStrength	A_per_meter	Ampere per meter
MagFieldStrength	oersted	Oersted
MagInduction	uGauss	Micro-Gauss
MagInduction	mGauss	Milli-Gauss
MagInduction	kGauss	Kilo-Gauss
MagInduction	uTesla	Micro-Tesla
MagInduction	mTesla	Milli-Tesla
MagInduction	kTesla	Kilo-Tesla
MagInduction	tesla	Tesla
MagInduction	gauss	Gauss
MagneticReluctance	A_per_Wb	Ampere per Weber

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
MagneticReluctance	A_per_Vs	Ampere per Volt-second
MassFlowRate	kg_per_s	Kilogram per second
MassFlowRate	g_per_s	Gram per second
MMF	a	Ampere
MMF	ua	Micro-Ampere
MMF	na	Nano-Ampere
MMF	ma	Milli-Ampere
MMF	ka	Kilo-Ampere
MolarDensity	mol_per_m3	Mol per cubic-meter
MolarDensity	mol_per_dm3	Mol per cubic-deci-meter
MolarDensity	mol_per_cm3	Mol per cubic-centi-meter
MolarDensity	mol_per_l	Mol per liter
MolarEnergy	J_per_mol	Joule per mole
MolarEnergy	megJ_per_mol	Mega-Joule per mole
MolarEnergy	mJ_per_mol	Milli-Joule per mole
MolarEnergy	uJ_per_mol	Micro-Joule per mole
MolarEnergy	kJ_per_mol	Kilo-Joule per mole
MolarEnergy	gJ_per_mol	Giga-Joule per mole
MolarVelocity	mol_per_s	Mole per second
MolarVelocity	cmol_per_s	Centi-Mole per second
MolarVelocity	mmol_per_s	Milli-Mole per second
MolarVelocity	kmol_per_s	Kilo-Mole per second
MolarVelocity	umol_per_s	Micro-Mole per second
MolarViscosity	Pas_per_mol	Pascal second per Mol
MomentInertia	kgm2	Kilo-gram square-meter
MomentInertia	lbin2	Pound inch-squared
MomentInertia	lbft2	Pound foot-squared
Momentum	kgm_per_s	Kilo-gram meter per second
Momentum	gm_per_s	Gram meter per second
NoiseSpectrum	dBc/Hz	
Percentage	percent	Per cent

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
PercentagePerTime	percent_per_s	Percent per second
PercentagePerTime	per_day	Per day
PercentagePerTime	percent_per_min	Percent per minute
PercentagePerTime	per_hour	Per hour
PercentagePerTime	percent_per_day	Percent per day
PercentagePerTime	percent_per_hour	Percent per hour
PercentagePerTime	per_min	Per minute
PercentagePerTime	per_s	Per second
Permeance	Vs_per_A	Volt-second per Ampere
Permeance	Wb_per_A	Weber per Ampere
Power	dBm	
Power	dBW	Decibel watt
Power	fW	Femto-Watt
Power	pW	Pico-Watt
Power	nW	nano-Watt
Power	HP	Horsepower
Power	Btu_per_hr	British thermal unit per hour
Power	W	Watt
Power	megW	Mega-Watt
Power	uW	Micro-Watt
Power	mW	Milli-Watt
Power	VA	Volt_Amps
Power	kW	Kilo-Watt
Power	gW	Giga-Watt
Power	J_per_s	Joule per second
Pressure	kn_per_meter_sq	Kilo-Newton per meter squared
Pressure	megn_per_meter_sq	Mega-Newton per meter squared
Pressure	gn_per_meter_sq	Giga-Newton per meter squared
Pressure	psi	Pounds per square inch
Pressure	kpsi	Kilo-pounds per square inch

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Pressure	megpsi	Mega-pounds per square inch
Pressure	gpsi	Giga-pounds per square inch
Pressure	pascal	Pascal
Pressure	hPascal	Hekto-Pascal
Pressure	gPascal	Giga-Pascal
Pressure	techAtm	Technical atmosphere
Pressure	kPascal	Kilo-Pascal
Pressure	mbar	Milli-bar
Pressure	stAtm	Standard atmosphere
Pressure	mPascal	Milli-Pascal
Pressure	n_per_meter_sq	Newton per square-meter
Pressure	megPascal	Mega-Pascal
Pressure	uPascal	Micro-Pascal
Pressure	torr	Torr
Pressure	bar	Bar
Pressure	cPascal	Centi-Pascal
Pressure	mmHg	Milli-meter mercury
Pressure	dPascal	Deci-Pascal
Pressure	mmh2o	Milli-meter water-column
PressureChangeRate	Pa_per_s	Pascal per second
PressureChangeRate	torr_per_hour	Torr per hour
PressureChangeRate	mmH2O_per_s	Milli-meter water-column per second
PressureChangeRate	psi_per_s	PSI per second
PressureChangeRate	techatm_per_hour	Technical atmosphere per hour
PressureChangeRate	bar_per_hour	Bar per hour
PressureChangeRate	statm_per_s	Standard atmosphere per second
PressureChangeRate	mbar_per_s	Milli-Bar per second
PressureChangeRate	mmH2O_per_ hour	Milli-meter water column per hour
PressureChangeRate	Pa_per_hour	Pascal per hour

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
PressureChangeRate	psi_per_hour	PSI per hour
PressureChangeRate	mmH2O_per_min	Milli-meter water column per minute
PressureChangeRate	techatm_per_min	Technical atmosphere per minute
PressureChangeRate	statm_per_hour	Standard atmosphere per hour
PressureChangeRate	torr_per_s	Torr per second
PressureChangeRate	mbar_per_hour	Milli-bar per hour
PressureChangeRate	techatm_per_s	Technical atmosphere per second
PressureChangeRate	bar_per_s	Bar per second
PressureChangeRate	statm_per_min	Standard atmosphere per minute
PressureChangeRate	torr_per_min	Torr per Minute
PressureChangeRate	Pa_per_min	Pascal per minute
PressureChangeRate	bar_per_min	Bar per minute
PressureChangeRate	mbar_per_min	Milli-bar per minute
PressureChangeRate	psi_per_min	PSI per minute
PressureCoefficient	per_Pa	Per Pascal
PressureCoefficient	per_statm	Per standard atmosphere
PressureCoefficient	per_psi	Per Pounds per square inch
PressureCoefficient	per_mmH2O	Per Milli-meter water-column
PressureCoefficient	per_mbar	Per Milli-bar
PressureCoefficient	V_per_Vper_Pa	Volt per Volt per Pascal
PressureCoefficient	per_mmHg	Per Milli-meter mercury
PressureCoefficient	per_techatm	Per technical atmosphere
PressureCoefficient	per_bar	Per bar
Ratio	db	Decibel
Ratio	bel	Bel
Reactance	Ohm	Ohm
Reactance	megohm	Mega-Ohm
Reactance	mOhm	Milli-Ohm

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Reactance	kOhm	Kilo-Ohm
Reactance	uOhm	Micro-Ohm
Reactance	Gohm	Giga-Ohm
ReciprocalPower	per_W	One over Watt
ReciprocalPower	per_megW	One over Mega-Watt
ReciprocalPower	per_Js	One over Joule-second
ReciprocalPower	per_mW	One over Milli-Watt
ReciprocalPower	per_kW	One over Kilo-Watt
ReciprocalPower	per_gW	One over Giga-Watt
ReciprocalResistanceCharge	per_OhmC	One over Ohm-coulomb
ReciprocalResistanceCharge	per_OhmAs	One over Ohm-Ampere-second
ReciprocalResistanceCharge	per_OhmAhour	One over Ohm-Ampere-hour
ReciprocalResistanceTime	per_Ohms	One over Ohm-second
ReciprocalResistanceTime	per_Ohmmin	One over Ohm-minute
ReciprocalResistanceTime	per_Ohmhour	One over Ohm-hour
Resistance	ohm	Ohm
Resistance	Ohm	Ohm
Resistance	megohm	Mega-Ohm
Resistance	mOhm	Milli-Ohm
Resistance	kOhm	Kilo-Ohm
Resistance	uOhm	Micro-Ohm
Resistance	Gohm	Giga-Ohm
ResistancePerCharge	Ohm_per_C	Ohm per Coulomb
ResistancePerCharge	Ohm_per_Ahour	Ohm per Ampere-hour
ResistancePerCharge	megOhm_per_C	Mega-Ohm per Coulomb
ResistancePerCharge	nOhm_per_C	Nano-Ohm per Coulomb
ResistancePerCharge	mOhm_per_C	Milli-Ohm per Coulomb
ResistancePerCharge	uOhm_per_C	Micro-Ohm per Coulomb
ResistancePerCharge	kOhm_per_C	Kilo-Ohm per Coulomb
ResistancePerCharge	Ohm_per_As	Ohm per Ampere-second
ResistancePerCharge	gOhm_per_C	Giga-Ohm per Coulomb

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
ResistancePerLength	Ohm_per_m	Ohm per meter
ResistancePerLength	megOhm_per_m	Mega-Ohm per meter
ResistancePerLength	mOhm_per_m	Milli-Ohm per meter
ResistancePerLength	uOhm_per_m	Micro-Ohm per meter
ResistancePerLength	kOhm_per_m	Kilo-Ohm per meter
ResistancePerLength	Ohm_per_um	Ohm per micro-meter
ResistanceTemperatureCoeff	Ohm_per_Kel	Ohm per Kelvin
ResistanceTemperatureCoeff	mOhm_per_Kel	Milli-Ohm per Kelvin
ResistanceTemperatureCoeff	kOhm_per_Kel	Kilo-Ohm per Kelvin
ResistanceTemperatureCoeff	Ohm_per_Cel	Ohm per degree Celsius
Resistivity	Ohmm	Ohm Meter
Resistivity	Ohmcm	Ohm Centi-Meter
Resistivity	Ohmum	Ohm micro-meter
Resistivity	Ohmmm2_per_mm	Ohm Square-Milli-Meter per Meter
SpecificHeatCapacity	J_per_Kelkg	Joule per Kelvin and Kilo-gram
SpecificHeatCapacity	mJ_per_Kelkg	Milli-Joule per Kelvin and Kilo-gram
SpecificHeatCapacity	kJ_per_Kelkg	Kilo-Joule per Kelvin and Kilo-gram
Speed	mm_per_sec	Milli-meter per second
Speed	cm_per_sec	Centi-meter per second
Speed	m_per_sec	Meter per second
Speed	m_per_hr	Meters per hour
Speed	inches_per_sec	Inches per second
Speed	feet_per_sec	Feet per second
Speed	feet_per_min	Feet per minute
Speed	miles_per_min	Miles per minute
Speed	miles_per_sec	Miles per second
Speed	km_per_min	Kilo-meters per minute
Speed	km_per_sec	Kilo-meters per second
Speed	m_per_sec	Meter per second

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Speed	km_per_hour	Kilometer per hour
Speed	miles_per_hour	Miles per hour
Stiffness	N_per_m	Newton per meter
Stiffness	lbf_per_in	Pound-force per inch
Stiffness	N_per_cm	Newton per centi-meter
Stiffness	kN_per_m	Kilo-Newton per Meter
SurfaceChargeDensity	surf_charge_C_per_m2	Coulomb per meter-squared
SurfaceChargeDensity	surf_charge_nC_per_m2	Nano-Coulomb per meter-squared
SurfaceChargeDensity	kC_per_m2	Micro-Coulomb per meter-squared
SurfaceChargeDensity	As_per_m2	Ampere-second per meter-squared
SurfaceChargeDensity	surf_charge_mC_per_m2	Milli-Coulomb per meter-squared
SurfaceMobility	m2_per_Vs	Square-meter per Volt-second
SurfaceMobility	cm2_per_Vs	Square-centi-meter per Volt-second
SurfaceMobilityPerVoltage	m2_per_V2s	Square-meter per Volt-second per Volt
SurfaceMobilityPerVoltage	cm2_per_V2s	Square-centi-meter per Volt-second per Volt
Susceptance	Sie	Siemens
Susceptance	megSie	Mega-Siemens
Susceptance	uSie	Micro-Siemens
Susceptance	mSie	Milli-Siemens
Susceptance	kSie	Kilo-Siemens
Susceptance	mho	Mhos
Temperature	kel	Kelvin
Temperature	mkel	Milli-Kelvin
Temperature	cel	Degree Celsius
Temperature	dkel	Deci-Kelvin

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Temperature	fah	Degree Farenheit
Temperature	ckel	Centi-Kelvin
TemperatureAreaPerPower	kelm2_per_w	Kelvin square-meter per Watt
TemperatureAreaPerPower	celm2_per_w	Degree Celsius square-meter per Watt
TemperatureCoefficient	per_Kel	Per Kelvin
TemperatureCoefficient	percent_per_Cel	Percent per degree Celsius
TemperatureCoefficient	per_Cel	Per degree Celsius
TemperatureCoefficient	percent_per_Fah	Percent per degree Farenheit
TemperatureCoefficient	per_Fah	Per degree Farenheit
TemperatureCoefficient	percent_per_Kel	Percentage per Kelvin
TemperatureCoefficient2	per_Kel2	Per Kelvin-squared
TemperatureCoefficient2	per_Fah2	Per degree Fahrenheit squared
TemperatureCoefficient2	per_Cel2	Per degree Celsius squared
TemperatureDifference	keldiff	Kelvin
TemperatureDifference	mkeldiff	Milli-Kelvin
TemperatureDifference	celdiff	Celsius
ThermalCapacitance	J_per_Kel	Joule per Kelvin
ThermalCapacitance	Ws_per_Kel	Watt-second per Kelvin
ThermalConductance	W_per_Kel	Watt per Kelvin
ThermalConductance	kW_per_Cel	Kilo-Watt per degree Celsius
ThermalConductance	mW_per_Kel	Milli-Watt per Kelvin
ThermalConductance	kW_per_Kel	Kilo-Watt per Kelvin
ThermalConductance	mW_per_Cel	Milli-Watt per degree Celsius
ThermalConductance	W_per_Cel	Watt per degree Celsius
ThermalConductivity	W_per_Kelm	Watt per Kelvin Meter
ThermalConductivity	mW_per_Kelm	Milli-Watt per Kelvin Meter
ThermalConvection	w_per_m2kel	Watt per square-meter Kelvin
ThermalConvection	w_per_cm2kel	Watt per square centi-meter Kelvin
ThermalRadiationCoeff	W_per_Kel4	Watt per Kelvin to the power of 4
ThermalRadiationCoeff	kW_per_Kel4	Kilo-Watt per Kelvin to the power

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
		of 4
ThermalRadiationCoeff	mW_per_Kel4	Milli-Watt per Kelvin to the power of 4
ThermalRadiationConstant	W_per_m2Kel4	Watt per square meter Kelvin to the power of 4
ThermalRadiationConstant	W_per_cm2Kel4	W per square centi-meter Kelvin to the Power of 4
ThermalResistance	Kel_per_W	Kelvin per Watt
ThermalResistance	Kels_per_J	Kelvin-second per Joule
Time	s	Second
Time	hour	Hour
Time	ps	Pico-Second
Time	us	Micro-Second
Time	ns	Nano-Second
Time	ms	Milli-Second
Time	min	Minute
Time	fs	Femto-second
Time	day	Day
TimePerAngle	s_per_rad	Second per radian
TimePerAngle	s_per_deg	Second per degree
TimePerAngle	ms_per_rad	Milli-Second per radian
TimePerAngle	s_per_rev	Second per revolution
TimeSqPerAngleSq	s2_per_rad2	Seconds squared per rad squared
TimeSqPerAngleSq	s2_per_deg2	Seconds squared per degree squared
Torque	fNewtonMeter	Femto-Newton-meter
Torque	pNewtonMeter	Pico-Newton-meter
Torque	nNewtonMeter	Nano-Newton-meter
Torque	megNewtonMeter	Mega-Newton-meter
Torque	gNewtonMeter	Giga-Newton-meter
Torque	FootPounds	Foot-pounds

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Torque	NewtonMeter	Newton-meter
Torque	kNewtonMeter	Kilo-Newton-meter
Torque	mNewtonMeter	Milli-Newton-meter
Torque	uNewtonMeter	Micro-Newton-meter
Torque	cNewtonMeter	Centi-Newton-meter
TransconductanceParameter	A_per_V	Ampere per Volt
TransconductanceParameter	mA_per_V	Milli-Ampere per Volt
TransconductanceParameter	kA_per_V	Kilo-Ampere per Volt
TransistorConstant	A_per_V2	Ampere per square Volt
TransistorConstant	mA_per_V2	Milli-Ampere per square Volt
TranslationalAcceleration	trans_accel_m_per_s2	Meter per square second
TranslationalAcceleration	trans_accel_cm_per_s2	Centi-Meter per square second
TranslationalAcceleration	dm_per_s2	Deci-Meter per square second
TranslationalAcceleration	trans_accel_in_per_s2	Inch per second-squared
VelocitySaturation	vel_sat_m_per_V	Meter per Volt
VelocitySaturation	vel_sat_mm_per_V	Milli-Meter per Volt
VelocitySaturation	vel_sat_um_per_V	Micro-Meter per Volt
VelocitySaturation	vel_sat_cm_per_V	Centi-Meter per Volt
VelocitySaturationPerVoltage	m_per_V2	Meter per Volt-squared
VelocitySaturationPerVoltage	cm_per_V2	Centi-Meter per Volt-squared
VelocitySaturationPerVoltage	um_per_V2	Micro-Meter per Volt-squared
VelocitySaturationPerVoltage	mm_per_V2	Milli-Meter per Volt-squared
Viscocity	Pas	Pascal-second
Viscocity	hPas	Hecto-Pascal-second
Viscocity	kPas	Kilo-Pascal-second
Viscocity	mPas	Milli-Pascal-second
Viscocity	poise	Poise

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
Viscocity	Ns_per_m2	Newton-second per square-meter
Viscocity	cpoise	Centi-poise
Viscocity	cPas	Centi-Pascal-second
Viscocity	dPas	Deci-Pascal-second
Viscocity	uPas	Micro-Pascal-second
ViscousFriction	vis_fric_Ns_per_m	Newton-second per meter
ViscousFriction	vis_fric_cNs_per_m	Centi-Newton-second per meter
ViscousFriction	vis_fric_mNs_per_m	Milli-Newton-second per meter
ViscousFriction	vis_fric_kNs_per_m	Kilo-Newton-second per meter
Voltage	dBV	Decibel Volt
Voltage	fV	Femto-Volt
Voltage	kV	Kilo-Volt
Voltage	megV	Mega-Volt
Voltage	mV	Milli-Volt
Voltage	nV	Nano-Volt
Voltage	pV	Pico-Volt
Voltage	V	Volt
Voltage	v	Volt
Voltage	fv	Femto-Volt
Voltage	pv	Pico-Volt
Voltage	uv	Micro-Volt
Voltage	megv	Mega-Volt
Voltage	nv	Nano-Volt
Voltage	mv	Milli-Volt
Voltage	kv	Kilo-Volt
Voltage	gv	Giga-Volt
VoltageAccelerationCoefficient	V_per_m2_per_s2	Volt per meter per second-squared

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
VoltageAccelerationCoefficient	mV_per_m2_per_s2	Milli-Volt per meter per second-squared
VoltageChangeRate	V_per_s	Volt per second
VoltageChangeRate	V_per_min	Volt per minute
VoltageChangeRate	V_per_hour	Volt per hour
VoltageChangeRate	mV_per_s	Milli-Volt per second
VoltageChangeRate	kV_per_s	Kilo-Volt per second
VoltageCoefficient	per_V	Per Volt
VoltageCoefficient	per_mV	Per Milli-Volt
VoltageCoefficient	per_kV	Per Kilo-Volt
VoltageCoefficient2	per_V2	Per Voltage-square
VoltageCubed	V3	Voltage-cubed
VoltageCubed	mV3	Milli-volt-cubed
VoltageGain	V_per_V	Volt per Volt
VoltageGain	V_per_mV	Volt per Milli-Volt
VoltageGain	mV_per_V	Milli-Volt per Volt
VoltageJerkCoefficient	V_per_m_per_s3	Volt per meter per second-cubed
VoltageJerkCoefficient	mV_per_m_per_s3	Milli-Volt per meter per second-cubed
VoltageLength	Vm	Volt meter
VoltageLength	kVm	Kilo-Volt meter
VoltageLength	mVm	Milli-Volt meter
VoltageLength	uVm	Micro-Volt meter
VoltagePerCell	V_per_cell	Volt per cell
VoltagePerCell	megV_per_cell	Mega-Volt per cell
VoltagePerCell	kV_per_cell	Kilo-Volt per cell
VoltagePerCell	uV_per_cell	Micro-Volt per cell
VoltagePerCell	mV_per_cell	Milli-Volt per cell
VoltagePerCell	nV_per_cell	Nano-Volt per cell
VoltagePerCell	pV_per_cell	Pico-Volt per cell
VoltagePerCell	gV_per_cell	Giga-Volt per cell
VoltagePerLengthRoot	V_per_mhalf	Volt per meter-root

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
VoltagePressureRootCoeff	V_per_Pahalf	Volt per Pascal-root
VoltagePressureRootCoeff	mV_per_Pahalf	Milli-Volt per Pascal-root
VoltageRoot	Vhalf	Volt to the power of 0.5
VoltageRootCoefficient	per_Vhalf	Per Volt-root
VoltageTemperature10Coeff	V_per_Kel10	Volt per Kelvin to the power of 10
VoltageTemperature10Coeff	V_per_Cel10	Volt per degree Celsius to the power of 10
VoltageTemperature10Coeff	uV_per_Kel10	Micro-Volt per Kelvin to the power of 10
VoltageTemperature10Coeff	mV_per_Kel10	Milli-Volt per Kelvin to the power of 10
VoltageTemperature10Coeff	uV_per_Cel10	Micro-Volt per degree Celsius to the power of 10
VoltageTemperature10Coeff	mV_per_Cel10	Milli-Volt per degree Celsius to the power of 10
VoltageTemperature11Coeff	V_per_Kel11	Volt per Kelvin to the power of 11
VoltageTemperature11Coeff	V_per_Cel11	Volt per degree Celsius to the power of 11
VoltageTemperature11Coeff	uV_per_Kel11	Micro-Volt per Kelvin to the power of 11
VoltageTemperature11Coeff	mV_per_Kel11	Milli-Volt per Kelvin to the power of 11
VoltageTemperature11Coeff	uV_per_Cel11	Micro-Volt per degree Celsius to the power of 11
VoltageTemperature11Coeff	mV_per_Cel11	Milli-Volt per degree Celsius to the power of 11
VoltageTemperature12Coeff	V_per_Kel12	Volt per Kelvin to the power of 12
VoltageTemperature12Coeff	V_per_Cel12	Volt per degree Celsius to the power of 12
VoltageTemperature12Coeff	uV_per_Kel12	Micro-Volt per Kelvin to the power of 12
VoltageTemperature12Coeff	mV_per_Kel12	Milli-Volt per Kelvin to the power of 12
VoltageTemperature12Coeff	uV_per_Cel12	Micro-Volt per degree Celsius to the power of 12

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
VoltageTemperature12Coeff	mV_per_Cel12	Milli-Volt per degree Celsius to the power of 12
VoltageTemperature13Coeff	V_per_Kel13	Volt per Kelvin to the power of 13
VoltageTemperature13Coeff	V_per_Cel13	Volt per degree Celsius to the power of 13
VoltageTemperature13Coeff	uV_per_Kel13	Micro-Volt per Kelvin to the power of 13
VoltageTemperature13Coeff	mV_per_Kel13	Milli-Volt per Kelvin to the power of 13
VoltageTemperature13Coeff	uV_per_Cel13	Micro-Volt per degree Celsius to the power of 13
VoltageTemperature13Coeff	mV_per_Cel13	Milli-Volt per degree Celsius to the power of 13
VoltageTemperature14Coeff	V_per_Kel14	Volt per Kelvin to the power of 14
VoltageTemperature14Coeff	V_per_Cel14	Volt per degree Celsius to the power of 14
VoltageTemperature14Coeff	uV_per_Kel14	Micro-Volt per Kelvin to the power of 14
VoltageTemperature14Coeff	mV_per_Kel14	Milli-Volt per Kelvin to the power of 14
VoltageTemperature14Coeff	uV_per_Cel14	Micro-Volt per degree Celsius to the power of 14
VoltageTemperature14Coeff	mV_per_Cel14	Milli-Volt per degree Celsius to the power of 14
VoltageTemperature15Coeff	V_per_Kel15	Volt per Kelvin to the power of 15
VoltageTemperature15Coeff	V_per_Cel15	Volt per degree Celsius to the power of 15
VoltageTemperature15Coeff	uV_per_Kel15	Micro-Volt per Kelvin to the power of 15
VoltageTemperature15Coeff	mV_per_Kel15	Milli-Volt per Kelvin to the power of 15
VoltageTemperature15Coeff	uV_per_Cel15	Micro-Volt per degree Celsius to the power of 15
VoltageTemperature15Coeff	mV_per_Cel15	Milli-Volt per degree Celsius to the power of 15

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
VoltageTemperature2Coeff	V_per_Kel2	Volt per Kelvin squared
VoltageTemperature2Coeff	V_per_Cel2	Volt per degree Celsius squared
VoltageTemperature2Coeff	uV_per_Kel2	Micro-Volt per Kelvin squared
VoltageTemperature2Coeff	mV_per_Kel2	Milli-Volt per Kelvin squared
VoltageTemperature2Coeff	uV_per_Cel2	Micro-Volt per degree Celsius squared
VoltageTemperature2Coeff	mV_per_Cel2	Milli-Volt per deree Celsius squared
VoltageTemperature3Coeff	V_per_Kel3	Volt per Kelvin cubed
VoltageTemperature3Coeff	V_per_Cel3	Volt per degree Celsius cubed
VoltageTemperature3Coeff	uV_per_Kel3	Micro-Volt per Kelvin cubed
VoltageTemperature3Coeff	mV_per_Kel3	Milli-Volt per Kelvin cubed
VoltageTemperature3Coeff	uV_per_Cel3	Micro-Volt per degree Celsius cubed
VoltageTemperature3Coeff	mV_per_Cel3	Milli-Volt per degree Celsius cubed
VoltageTemperature4Coeff	V_per_Kel4	Volt per Kelvin to the power of 4
VoltageTemperature4Coeff	V_per_Cel4	Volt per degree Celsius to the power of 4
VoltageTemperature4Coeff	uV_per_Kel4	Micro-Volt per Kelvin to the power of 4
VoltageTemperature4Coeff	mV_per_Kel4	Milli-Volt per Kelvin to the power of 4
VoltageTemperature4Coeff	uV_per_Cel4	Micro-Volt per degree Celsius to the power of 4
VoltageTemperature4Coeff	mV_per_Cel4	Milli-Volt per degree Celsius to the power of 4
VoltageTemperature5Coeff	V_per_Kel5	Volt per Kelvin to the power of 5
VoltageTemperature5Coeff	V_per_Cel5	Volt per degree Celsius to the power of 5
VoltageTemperature5Coeff	uV_per_Kel5	Micro-Volt per Kelvin to the power of 5
VoltageTemperature5Coeff	mV_per_Kel5	Milli-Volt per Kelvin to the power of 5
VoltageTemperature5Coeff	uV_per_Cel5	Micro-Volt per degree Celsius to

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
		the power of 5
VoltageTemperature5Coeff	mV_per_Cel5	Milli-Volt per degree Celsius to the power of 5
VoltageTemperature6Coeff	V_per_Kel6	Volt per Kelvin to the power of 6
VoltageTemperature6Coeff	V_per_Cel6	Volt per Kelvin to the power of 6
VoltageTemperature6Coeff	uV_per_Kel6	Micro-Volt per Kelvin to the power of 6
VoltageTemperature6Coeff	mV_per_Kel6	Milli-Volt per Kelvin to the power of 6
VoltageTemperature6Coeff	uV_per_Cel6	Micro-Volt per degree Celsius to the power of 6
VoltageTemperature6Coeff	mV_per_Cel6	Milli-Volt per degree Celsius to the power of 6
VoltageTemperature7Coeff	V_per_Kel7	Volt per Kelvin to the power of 7
VoltageTemperature7Coeff	V_per_Cel7	Volt per degree Celsius to the power of 7
VoltageTemperature7Coeff	uV_per_Kel7	Micro-Volt per Kelvin to the power of 7
VoltageTemperature7Coeff	mV_per_Kel7	Milli-Volt per Kelvin to the power of 7
VoltageTemperature7Coeff	uV_per_Cel7	Micro-Volt per degree Celsius to the power of 7
VoltageTemperature7Coeff	mV_per_Cel7	Milli-Volt per degree Celsius v 7
VoltageTemperature8Coeff	V_per_Kel8	Volt per Kelvin to the power of 8
VoltageTemperature8Coeff	V_per_Cel8	Volt per degree Celsius to the power of 8
VoltageTemperature8Coeff	uV_per_Kel8	Micro-Volt per Kelvin to the power of 8
VoltageTemperature8Coeff	mV_per_Kel8	Milli-Volt per Kelvin to the power of 8
VoltageTemperature8Coeff	uV_per_Cel8	Micro-Volt per degree Celsius to the power of 8
VoltageTemperature8Coeff	mV_per_Cel8	Milli-Volt per degree Celsius to the power of 8
VoltageTemperature9Coeff	V_per_Kel9	Volt per Kelvin to the power of 9

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
VoltageTemperature9Coeff	V_per_Cel9	Volt per degree Celsius to the power of 9
VoltageTemperature9Coeff	uV_per_Kel9	Micro-Volt per Kelvin to the power of 9
VoltageTemperature9Coeff	mV_per_Kel9	Milli-Volt per Kelvin v 9
VoltageTemperature9Coeff	uV_per_Cel9	Micro-Volt per degree Celsius to the power of 9
VoltageTemperature9Coeff	mV_per_Cel9	Milli-Volt per degree Celsius to the power of 9
VoltageTemperatureCoeff	V_per_Kel	Volt per Kelvin
VoltageTemperatureCoeff	V_per_Cel	Volt per degrees Celsius
VoltageTemperatureCoeff	uV_per_Kel	Micro-Volt per Kelvin
VoltageTemperatureCoeff	mV_per_Kel	Milli-Volt per Kelvin
VoltageTemperatureCoeff	uV_per_Cel	Micro-Volt per degree Celsius
VoltageTemperatureCoeff	mV_per_Cel	Milli-Volt per degree Celsius
Volume	l	
Volume	ml	
Volume	m3	Cubic-Meter
Volume	galUS	Gallon (am.)
Volume	cup	Cup
Volume	galUK	Gallon (br.)
Volume	l	Liter
Volume	mm3	Cubic-Centi-Meter
Volume	ml	Cubic-Centi-Meter
VolumeCoefficient	per_m3	Per cubic-meter
VolumeCoefficient	per_cm3	Per cubic-centi-meter
VolumeFlowConductance	vol_flow_con_ m3_per_Pas	Cubic-meter per Pascal-second
VolumeFlowConductance	vol_flow_con_ cm3_per_Pas	Centi-meter cubed per Pascal-second
VolumeFlowPerPressureRoot	m3_per_sPahalf	Meter-cubed per second per Pascal-root
VolumeFlowRate	m3_per_s	Cubic meter per second

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Unit Type	String Representation	Description
VolumeFlowRate	cm3_per_s	Cubic centi-meter per second
VolumeFlowRate	m3_per_hour	Cubic meter per hour
VolumeFlowRate	m3_per_min	Cubic meter per minute
VolumeFlowRateChangeRate	m3_per_s2	Cubic meter per square-second
VolumeFlowRateChangeRate	cm3_per_s2	Centi meter cubed per second squared
Weight	oz	Ounce
Weight	lb	Pound
Weight	kg	Kilogram
Weight	gram	Gram
Weight	ug	Micro-Gram
Weight	mg	Milli-Gram
Weight	ton	Tons
WireCrossSection	m2wire	Meter squared
WireCrossSection	um2wire	Micro-meter squared
WireCrossSection	ft2wire	Square foot
WireCrossSection	in2wire	Square inch
WireCrossSection	mm2wire	Milli-meter squared
WireCrossSection	cm2wire	Centi-meter squared
WireCrossSection	AWG	American wire gauge

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

26 - Technical Notes

The following technical notes provide additional information on a variety of topics.

- Fixing Non-Convergence in Simplorer
- Using TimeTrigger
- Simplorer - ModelSim Co-simulation Interface User Guide
- About the VHDL-AMS working Library in Simplorer

Fixing Non-Convergence in Twin Builder

In case of non-convergence during transient simulation (that is, the maximum number of iterations has been exceeded), using the steps below will lead to convergence and accurate simulations in most cases. Most options below (unless otherwise noted) can be found under the **TR** tab under **Solution Options**.

> **Note:**
>
> The items below are given in the recommended order of use.

- **Change the acceptable tolerance values** – The two absolute errors and the one relative tolerance value are used internally by the solver to check accuracy (convergence). Relaxing the tolerances may aid the simulation convergence.
  - It is usually safe to add a relative tolerance. Set the relative tolerance to 0.1%.
  - If acceptable, raise the values of the absolute RHS and LHS tolerances.
- **Increase the maximum allowed iterations** – By default this value is set to 40. Try setting this value to 100 or more. The solver converges very rapidly (in quadratic order) if the initial solution is close to the actual solution. If not, it can exceed the max iteration limit very quickly.
- **Reduce the minimum time step** – If increasing max iterations does not work, the system may be changing faster that the minimum time step allows the simulator to solve it. Reduce the minimum time step to allow sufficient sampling for the highest frequency signal in the simulation.

  > **Note:**
  >
  > In some cases, increasing the allowed time step value can aid convergence by allowing the simulator to step over discontinuous, difficult to solve points in the simulation.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Specify initial conditions (when available)** – When possible, such as with capacitors, inductors, and especially in VHDL-AMS models (using the break statement), specify the initial values of quantities. Since the solver relies largely on a good initial guess, it can help tremendously if a good starting point is supplied to it.
- **Adjust the slider position** – When dealing with certain systems, (possibly systems which lead to a near singular matrix) partial pivoting may not work. Try forcing full pivoting. This guarantees the same or greater accuracy and a better chance of convergence. For more information about adjusting the slider, see ***Using the Simulator Performance Slider***.
- **Use the Simulation Parameters model (located in the Component Libraries under Basic Elements > Tools > Simulator Parameters**) – A very useful model that can debug simulator issues, such as non-convergence. By monitoring the various solver parameters like the time step used by the solver, the number of iterations required, number of rollbacks, and so on, it is possible to easily diagnose and remedy the problem. The SIMPARAM parameters are defined in ***Simulator Parameters*** in the Basic Elements Library help.
- **Change the LDF** – The LDF (the integration algorithm truncation error) value can be reduced to force the solver to make smaller increases in the step size and may help convergence at the current time step. Increasing LDF may also work by allowing the solver to jump over problematic solution points, but is not a recommended practice.
- **Turn on "Apply operating-point convergence scheme"** – Achieving convergence at the transient operating point (at t = 0) can be difficult for circuits with highly nonlinear models. When such circuits provide invalid results, turning on **Apply operating-point convergence scheme** in the TR option may provide better results. This option applies the continuation methods Gmin-stepping and Source-stepping to improve convergence.
- **Improve the Model** – Some simple model improvements can save the simulator a lot of work and aid convergence. For example, smooth out any discontinuities, add damping for highly non-linear models, check and fix any inconsistencies and errors in circuit topology and connectivity, make models more realistic, and so on.

Using Time Trigger

Using the Time Trigger feature, you can specify synchronization points at which the simulator will be forced to do a calculation. A time trigger event can now be scheduled as a singular event or as a periodical one.

Related Topics

Intention

Handling

Library Extensions

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Intention

Imagine a state machine that has to perform certain operations at exactly specified time points, for example, at t1 = 1 ms, t2 = 2 ms, and so on. To trigger these events, the simulator will approach t1 with some step size h. It will ignore t1 until t > t1, then discard the last step and reduce h. The simulator does some kind of interpolation to find the new h. From then on, h will be successively increased until again t > t1 and the same procedure starts anew. This whole process continues until t1 is being stepped over by hmin.

This has many drawbacks. First, the simulation accuracy suffers considerably because t1 is only hit by an error depending on hmin. Second, simulation speed suffers extremely (especially for clocked systems and alike) because Twin Builder takes a lot of extra steps (such as many hundreds depending on h, hmin, and hmax) just to trigger this event roughly.

If the model itself knows the point where the condition goes true, it could set the step size to synchronize exactly to this point.

Handling

The backplane includes now a trigger list. It's a sorted list with all scheduled events. After a time step, when the simulator calculates the step size for the next time step, it includes this list for the step size calculation.

New functions to schedule and unschedule events are available in the system now. The functions can be called from the State graph and the C-Interface.

```
ScheduleTrigger or SCHED - will create a new trigger.
DeScheduleTrigger or DESCHED - will remove a trigger
```

Calling repeatedly the function ScheduleTrigger or SCHED with the same name will overwrite the existing time trigger with that name.

Definition of more than one time triggers for a time point is possible. Unscheduling has no effect as long as at least one trigger for that time point is active. If all triggers are removed, the simulator will not synchronize at that time point.

State Graph Extension

The state graph module has a new feature - a Schedule event action type. An associated transition will switch (that is, become TRUE) exactly at the moment of this time event. The name of this Schedule event action type is SCHED. An action of some state looks like:

```
SCHED dTEv1 $$ t + 1u
```

For periodical events, add the period length as additional parameter:

```
SCHED dTEv2 $$ t + 1u, 5u
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SCHED sets variable dTEv1 to current simulation time plus 1 microsecond and synchronize at time t + 1u. Whenever a SCHED action type is encountered, ScheduleEvent will be called by the simulator. A new trigger is created in the Backplane and sorted into the trigger list. The Simulator uses this trigger list to calculate the next time step size.

A transition condition looks like:

```
COND: dTEv1
/* transition TRUE if dTEv1 has been reached, simulator triggers
exactly on dTEv1 */
```

or

```
COND: dTEv1 OR R1.I >= I_Max
/* transition TRUE if dTEv1 has been reached (simulator triggers
exactly on dTEv1) or current through R1 greater than or equal to some
variable I_Max*/
```

There will be a DESCHED action type that executes DescheduleEvent. The according action looks like:

```
DESCHED dTEv1 /* removes the value of dTEv1 from the trigger list */
```

The $$ will set dTEv1 to the right side value and schedule the event once the state becomes active. Thus, it might also be omitted because the SCHED type seems to form a unique definition. However, the DEL type also goes along with the ##-operator although one seems to suffice. Again: This new action/transition combination corresponds to the delay operator (##), where a certain state graph transition will become TRUE after a certain time has elapsed, for example:

```
DEL lTEl1 ## t + 1.1u
/* lTel1 becomes TRUE after t + 1.1 us has elapsed */
COND: lTEl1 AND R1.I < I_Max
/* switch if time elapsed and current less than its maximum, ... */
```

C/C++ Interface Extension

There are two predefined functions that call ScheduleEvent and DescheduleEvent from inside a C-Model. Calling the Schedule or DeSchedule function are only allowed in SimInit() or SimValidate(), otherwise a error message occurs. The **Sim2000User.h** header file is extended and includes 2 new function prototypes:

```
//functions to schedule user model sync time points
void ScheduleTrigger (LPCTSTR pszName, double dAbsSyncTime, double
dPeriod = 0.0);
```

where,

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

pszName - user defined non-ambiguous name for the time trigger

dAbsSyncTime - absolute time of the next trigger time in seconds

dPeriod - period length for the trigger in seconds, 0.0 means no period

Calling the ScheduleTrigger function a second time with the same name will overwrite the previous one. For example:

```
ScheduleTrigger (“MyTrig”, 1e-3);
ScheduleTrigger (“MyTrig”, 1.5e-3, 0.5e-3);
```

Only the second trigger at time 1.5ms with a period length of 0.5ms is active. At time 1ms the simulator will not necessarily synchronize.

```
//functions to deschedule user model sync time points
void DeScheduleTrigger (LPCTSTR pszName);
```

where,

pszName – trigger name corresponding with the name used in ScheduleTrigger()

Error Messages

Messages are displayed when you call Schedule or Deschedule in SIMPREP or SIMSTEP.

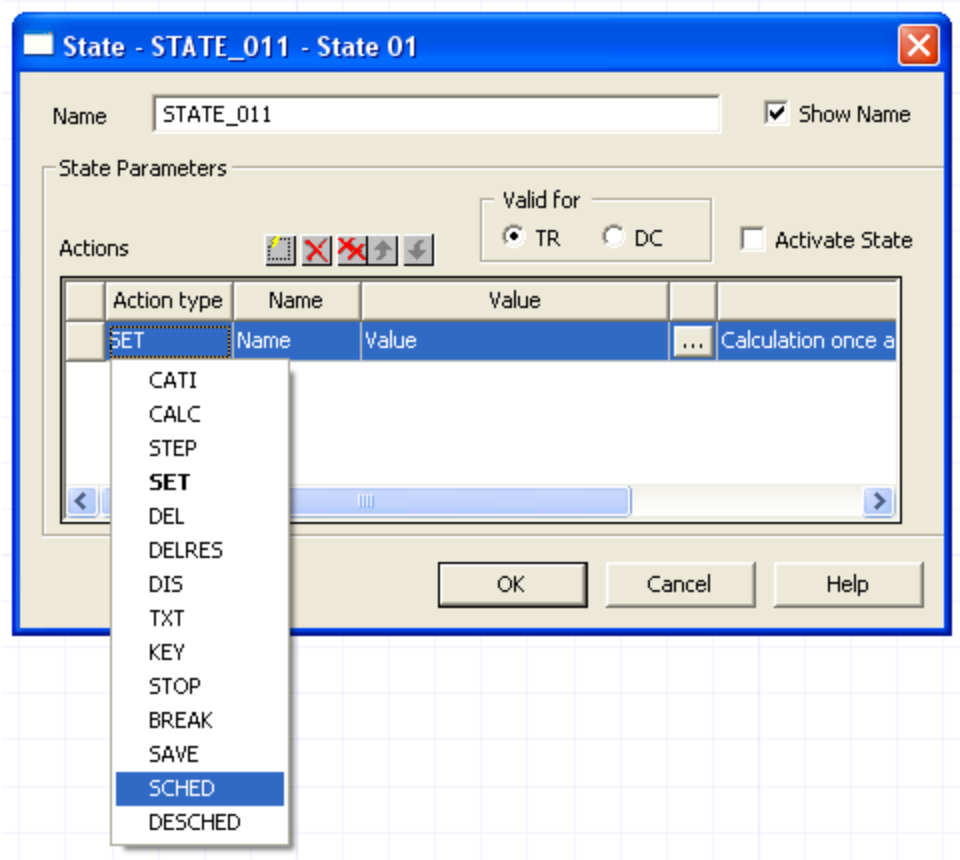
Error in deschedule the time trigger <%name>. Descheduling allowed only in SimInit() or SimValidate().

Error in schedule the time trigger <%name>. Scheduling allowed only in SimInit() or SimValidate ().

Library Extensions

New Action type is added to all State graph models in the Basic library. The wizard displays the new action type in the combo box.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Simplorer - ModelSim Co-simulation Interface User Guide

"Introduction (ModelSim)" on the facing page

"Installation/Pre-Setup (ModelSim)" on the facing page

"Setup (ModelSim)" on the facing page

"Co-simulation Setup (ModelSim)" on page 26-10

"Simulation (ModelSim)" on page 26-11

"Displaying Results and Post Processing (ModelSim)" on page 26-12

"Limitations and Known Issues (ModelSim)" on page 26-13

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Introduction (ModelSim)

This document outlines the Twin Builder co-simulation interface to *ModelSim SE 6.3* available starting with Simplorer 8.0. Through the co-simulation interface, model developers will have the opportunity to simulate and verify complex digital sub-systems in a complete system simulation environment.

Main Features

1. Supports the *ModelSim SE 6.3* license.
2. Requires that Twin Builder and ModelSim run on the same machine.

The following sections outline the procedure to setup, start, and control a co-simulation.

"Installation/Pre-Setup (ModelSim)" below

"Setup (ModelSim)" below

"Co-simulation Setup (ModelSim)" on page 26-10

"Simulation (ModelSim)" on page 26-11

"Displaying Results and Post Processing (ModelSim)" on page 26-12

"Limitations and Known Issues (ModelSim)" on page 26-13

Installation/Pre-Setup (ModelSim)

1. The co-simulation environment makes use of an environment variable "SIMPMSCoSimDLL". During installation, this is set to point to the co-simulation dll:

`<your Twin Builder installation directory>\SIMPModelSimCoSim.dll`

2. For automatic mode, Twin Builder must be able to invoke the ModelSim executables (vlib, vcom, vsim, etc). Add the ModelSim executable path to the system path to ensure this.

Setup (ModelSim)

Prior to simulation, you must first create the co-simulation models and add them to the Twin Builder library.

"Creating Co-simulation Models (ModelSim)" on the next page

"Schematic Capture (ModelSim)" on the next page

"Netlisting (ModelSim)" on page 26-10

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Creating Co-simulation Models (ModelSim)

To use a model which is to be simulated in ModelSim, you must first create a VHDL model in Twin Builder with the entity information and an empty architecture decorated with a special **foreign** attribute as in the following example.

```
--co-simulation model entity
ENTITY inv_modelsim IS
GENERIC (TP_LH : REAL := 0.0;
TP_HL : REAL := 0.0);
PORT (x : IN BIT := '0';
y : OUT BIT := '1');
END ENTITY inv_modelsim;

ARCHITECTURE behav OF inv_modelsim IS
--foreign attribute
 ATTRIBUTE foreign OF behav : ARCHITECTURE IS "ModelSim";
BEGIN
END behav;
```

The entity information is all that is needed to generate the complete interface for the foreign model. The foreign attribute is defined in package STANDARD by the VHDL-AMS Language Reference Manual and marks the architecture for “implementation-dependent elaboration”.

You can use a specific library/model/architecture from ModelSim, by extending the foreign string. The syntax is:

“ModelSim.<library name>.<entity name>.<architecture name>”

By default the co-simulation will use the work library, the same entity name as that of the interface model and the default architecture. You can specify either of the following combinations:

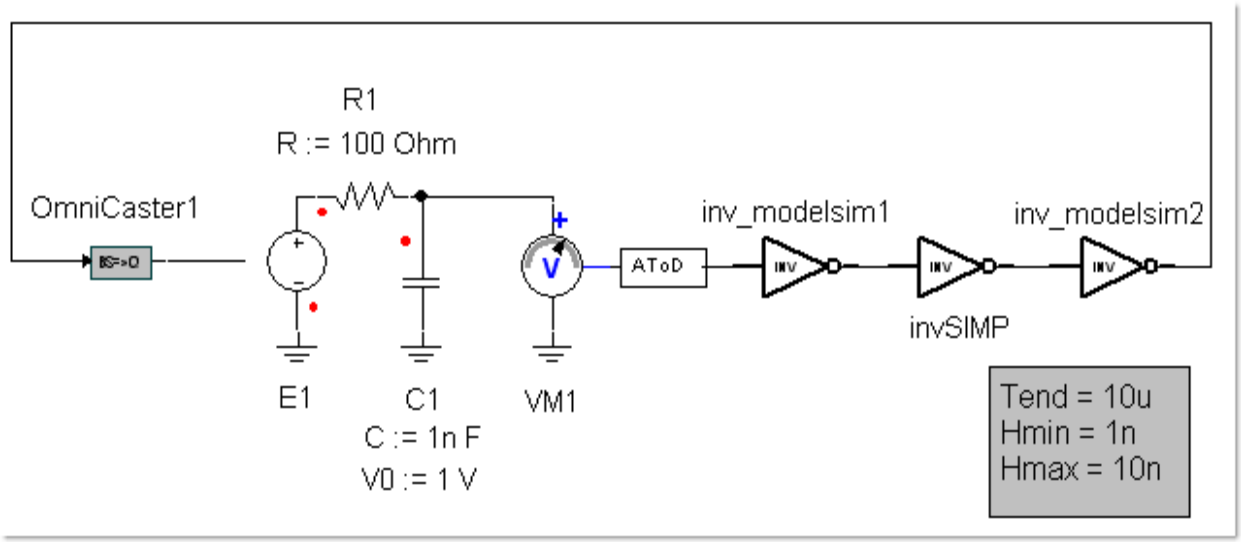
<library name>
<library name>.<entity name> or
<library name>.<entity name>.<architecture name>.

Schematic Capture (ModelSim)

Once the foreign model or models are created in Twin Builder, they can be used like any other Twin Builder model. Drop them on the sheet and connect them to other Twin Builder models to create the schematic. The following figure shows the schematic for a simple ring oscillator model

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

created using two instances (**inv_modelsim1** and **inv_modelsim2**) of the inverter model described above.



The AToD VHDL-AMS model is described below.

```
ENTITY AToD IS
  GENERIC (TS : REAL := 0.0);
  PORT (QUANTITY INPUT : IN REAL := 0.0;
    QUANTITY THRES1 : IN REAL := 0.01;
    QUANTITY THRES2 : IN REAL := 0.99;
  SIGNAL sig : OUT BIT := '0');
END ENTITY AToD;

ARCHITECTURE behav OF AToD IS
  SIGNAL upper_crossing : BOOLEAN := FALSE;
  SIGNAL lower_crossing : BOOLEAN := TRUE;
BEGIN
  upper_crossing <= input'ABOVE(THRES2);
  lower_crossing <= NOT input'ABOVE(THRES1);
  BREAK ON upper_crossing, lower_crossing;

  PROCESS (upper_crossing,lower_crossing)
  BEGIN
    IF (INPUT<THRES1) THEN
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
      sig <= '0';
    ELSIF (INPUT > THRES2) THEN
      sig <= '1';
    END IF;
  END PROCESS;
END ARCHITECTURE;
```

Netlisting (ModelSim)

Based on the schematic description, Twin Builder creates two netlists. The first is the .sml file for the Twin Builder simulator. The second is the netlist description, in VHDL, for ModelSim. The ModelSim netlist is created in the same location as the Twin Builder sml netlist and is named **<sml_name>_EDSCoSim_Netlist.vhd**.

Co-simulation Setup (ModelSim)

In this step, the top level design is compiled and elaborated in ModelSim. ModelSim is then started with the top level design (entity “top” architecture “implement”) and co-simulation can begin.

Prior to simulating, you must compile all models and sub-models to be simulated in ModelSim separately into ModelSim libraries. These libraries must be available from the same directory as the Twin Builder project through appropriate definitions in the cds.lib file. For details on how to compile models, library management and the cdl.lib file, see the Mentor Graphics ModelSim user documentation. In addition, Twin Builder uses ‘worklib’ as the default work library to compile and elaborate the ModelSim netlist.

```
-- Inverter behavioral model description
ENTITY inv_modelsim IS
  GENERIC ( TP_LH : REAL := 0.0;
    TP_HL : REAL := 0.0 );
  PORT ( x : IN BIT := '0';
    y : OUT BIT := '1');
END inv_modelsim;

ARCHITECTURE behav OF inv_modelsim IS
CONSTANT lh : TIME := TP_LH * 1 sec;
CONSTANT hl : TIME := TP_HL * 1 sec;
SIGNAL INV_OUT : BIT := '0';
BEGIN

  INV_OUT <= NOT X;
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
  Y <= INV_OUT AFTER hl WHEN INV_OUT = '0' ELSE
     INV_OUT AFTER lh;

END behav;
```

When you start a simulation (for a design with models marked for ModelSim simulation) in Twin Builder, Twin Builder generates the top-level ModelSim netlist, then compiles, elaborates, and starts ModelSim with the top level design which in turn loads the co-simulation interface.

The commands executed by Twin Builder to set up ModelSim are:

```
vlib work
vcom -bindAtCompile <sml_name>_EDSCoSim_Netlist.vhd
vsim -t 1fs work.top
```

When ModelSim successfully loads the co-simulation dll, the message “Successfully loaded the Simplorer-ModelSim Co-simulation Interface”appears. Once the ModelSim prompt appears, co-simulation setup is complete.

Note that once a setup is completed for a schematic, it remains valid until you load a new schematic. This also means that only one valid co-simulation setup can exist at one time. If an existing ModelSim setup is found by Twin Builder, it will attempt to link to it and will fail if the existing model loaded in ModelSim has a different interface that the one it is trying to link to.

Simulation (ModelSim)

"Starting Simulation (ModelSim)" below

"Controlling Simulation (ModelSim)" on the next page

"Ending Simulation (ModelSim)" on the next page

"Saving and Continuing a Simulation (ModelSim)" on the next page

Starting Simulation (ModelSim)

After a successful setup, Twin Builder begins the simulation. If the **Wait after loading external simulator** check box is selected, Twin Builder will bring up a dialog box after successful simulation and will wait for you to press **OK**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Controlling Simulation (ModelSim)

Since Twin Builder controls the simulation, all simulation parameters used for the co-simulation (such as the simulation end time) are determined by Twin Builder's simulation parameters dialog box.

During simulation, it is possible to pause or stop the simulation from Twin Builder. You may also continue the simulation at the end of a successful simulation run. To restart the simulation, start the simulation again. It is not required to follow the setup procedure if restarting a simulation for the currently setup schematic.

Ending Simulation (ModelSim)

Stop the simulation in Twin Builder to end the co-simulation. This will force ModelSim to stop as well.

Saving and Continuing a Simulation (ModelSim)

It is possible to save the state of a simulation at the end of a successful run in a **.krn** file. In case of a co-simulation, Twin Builder also prompts ModelSim to save its state in a checkpoint file. The checkpoint file name generates based on the name of the ModelSim top level source file name.

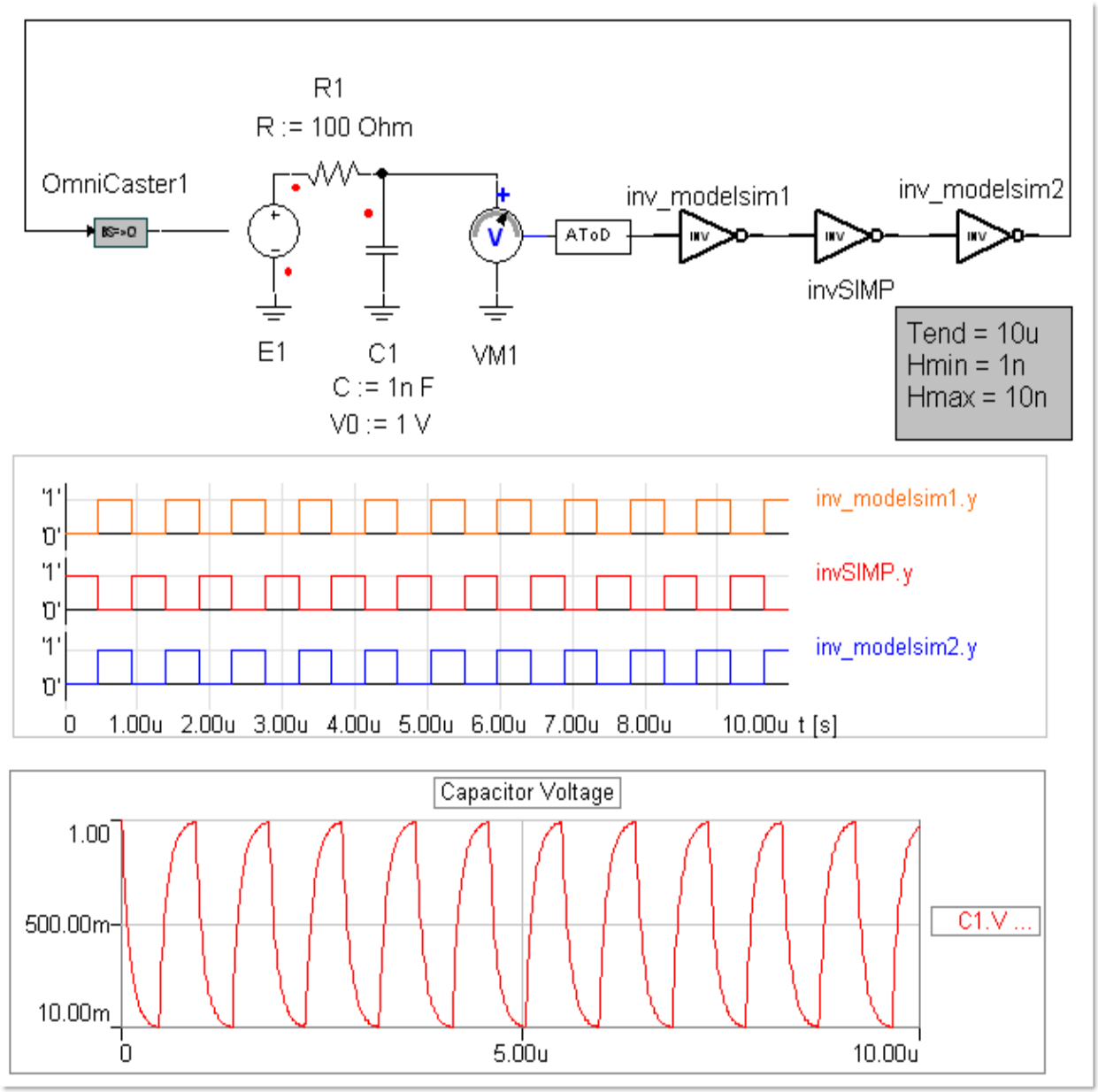
Similarly, when a previously saved **.krn** file is loaded, Twin Builder detects the presence of a co-simulation model and prompts ModelSim to load the saved checkpoint file.

Note that setup steps still must be followed before restoring from a saved simulation.

Displaying Results and Post Processing (ModelSim)

Any outputs crossing the co-simulation interface can be displayed in both Twin Builder and ModelSim. Internal ModelSim signals can be logged separately in ModelSim through the ModelSim UI at anytime after the ModelSim setup is complete. The following figure shows the results of the simulation for the ring oscillator example as seen in Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Limitations and Known Issues (ModelSim)

INOUTS are not currently supported across the co-simulation interface. Signals crossing the co-simulation interface can only be of type IN or OUT.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

About the VHDL-AMS *working* Library in Twin Builder

According to the VHDL-AMS language definition, all design units (for example, *models* or *packages* in Twin Builder) are compiled into the working library (*work*) from which all following analysis steps take the model for further processing. Generally, VHDL-AMS tools keep only the latest compiled version of a design unit in this working library. The origin of the design unit is lost in this process. That also means that in order to guarantee the correct creation of a structural design, all design units must be compiled at the same time, and the order of compilation must be according to the units' dependencies.

Twin Builder follows an approach that allows for greater control over these references. Although win Builde has a similar working library (*project*), this library can contain multiple design units that have the same definition name but were loaded from different libraries. The interfaces and contents of these models (or packages, but we'll refer to just models below for simplicity) do not have to match each other. For this reason, all models in the *project* library preserve their original library locations; and in order to access a specific model, the original library location must be specified together with the model name. In addition, *project* itself is a regular library that is used as the origin for models that were imported from text files or created manually using the VHDL-AMS Model Editor.

In other VHDL-AMS systems, if a model **def** refers to a model using **work.abc**, **def** will access the current model **abc** in the working library. When using **def**, however, the **abc** in the work library to which **work.abc** refers may not be the same **abc** that was used in the creation and testing of **def**.

References to **work.abc** are handled with more specificity in Twin Builder. The library that **abc** came from is kept with **abc** and when, as above, a model **def** references **work.abc**, then the same **abc** will be used for **def** as was used in its creation and testing.

> **Note:**
>
> Even though Twin Builder can define a library alias name **work** (this would load models specified by **work.abc** from the aliased library), it does not have the same behavior as in other VHDL tools. The library concept used in Ansys Electromagnetics tools does not allow automatic writing to any library other than **project**. You can only write to another library by manual export.

Related Topics

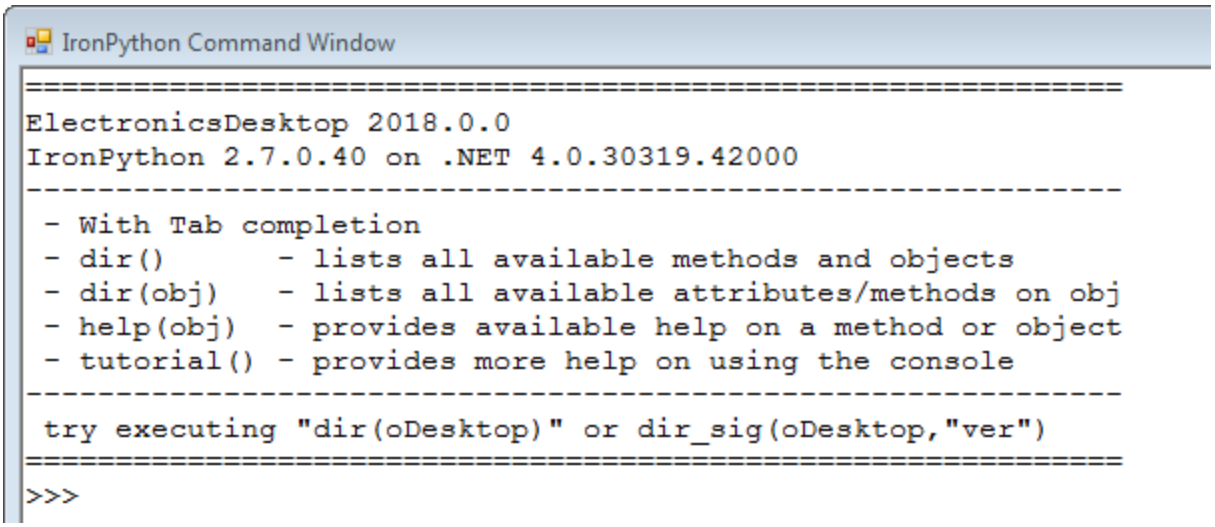
The VHDL-AMS work Library in Twin Builder

Using the VHDL-AMS Model Editor

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

27 - DesktopScripting with IronPython

This document describes IronPython briefly, then goes on to describe the desktop provided IronPython scripting console and scripting with IronPython. Select **Tools** > **Open Command Window** to open an IronPython Command Window.



The document assumes that you know how desktop scripting works using VBScript or Javascript.

Related Topics

User Defined Outputs: An Introduction

Introduction to IronPython

IronPython is an implementation of the Python programming language targeting the .NET runtime. What this means in practical terms is that IronPython uses the Python programming language syntax and standard python libraries and can additionally use .NET classes and objects to give one the best of both worlds. This usage of .NET classes is fairly seamless in that a class defined in a .NET assembly can be used as a base class of a python class.

Scope

Functioning as a tutorial on Python or IronPython is way out of the scope of this document. There are several excellent resources online that do a very good job in that regard. This document only attempts to provide a limited introduction to IronPython as used to script Ansys EM products.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

This document is also not a tutorial on the scripting of Ansys EM products. It complements the existing scripting guide (available from a product's Help menu) and provides a pythonic interpretation of that information. See either the scripting guide or recorded samples of VBScript to follow some of the sections.

Python Compatibility

The version of IronPython in use is **2.7** and built on the .NET framework version 4.0: this version targets **Python 2.7** language compatibility. While most python files will execute under IronPython with no changes, python libraries that make use of extensions written in the C programming language (NumPy or SciPy for instance), are not expected to work under IronPython. In such cases, it might be possible to locate .NET implementation of such libraries or explore the use of IronClad.

(http://code.google.com/p/ironclad/).

Advantages of IronPython

The advantages that IronPython provides are significant:

- Python has a large eco-system with plenty of supporting libraries, Visual IDEs and debuggers. It is actively developed and enhanced.
- IronPython, in addition, has access to the entire .NET ecosystem. You can use it, for instance, to create a modern GUI using the **System.Windows.Forms** assembly from IronPython code and call any other .NET assembly as well.
- The use of IronPython's technologies enables the ability to interactively script Desktop (feature in development). This allows better discovery of the scripting APIs as well as directly programming to the scripting API in python, a language more tractable and platform independent compared with VBScript.
- The Python syntax of dictionaries is somewhat easier to read and write when supplying arguments to the scripting methods.

Related Topics

IronPython Mini-cookbook

Translating Script commands from VBScript to IronPython

Scripting Using IronPython: Putting it all Together

IronPython Samples

IronPython Mini-cookbook

While a tutorial on Python syntax is beyond the scope of this document, it will present simple counterparts for the VBScript constructs that users regularly use.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Comments

Assigning/Creating variables

Create Lists/Arrays

Create Dictionaries/Maps

Boolean Values

Converting Numbers to Strings and Vice Versa

String Formatting/Concatenation

Looping over Lists

Looping over a Range

A Note About Indentation

Additional Sections:

Obtaining More Information

Discovering Methods

Help on a Method

Comments

VBScript	IronPython
' **Comments start with a single quote** ' like this line	# Comments start with a sharp or hash # symbol, like these lines

Assigning/Creating Variables

VBScript	IronPython

' Declare with a Dim
Dim oDesktop
 	
'**Assignment needs a Set instruction**
Set oDesktop = oApp.GetAppDesktop()
	# No Set syntax. Create and assign
	oDesktop = oApp.GetAppDesktop()

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Create Lists/Arrays

VBScript	IronPython

' Declare as array of String with 11
' indices from 0 through 10
Dim myArray(0 to 10) as String
myArray(0) = "Hello"
myArray(1) = "bye"
 	
' **Declare n array with no size**
Dim array2() as String
 	
' **Re-Dimension the array once size is**
' **known**
ReDim array2(0 to 2) as String
array2(0) = "this"
array2(1) = "also"
	# Declare an empty array
	myEmptyArray = []
	
	# declare an array and initialize it with 5 ints
	myInitedArray = [1, 2, 3, 4, 5]
	
	# Python lists can have items of any type
	# and there is no pre-declaration
	# declare an array and init with mixed types
	mixed = ["hello", 1 ,2 ["nested"]]
	
	# append to an array
	mixed.append(3.5)

Create Dictionaries/Maps

VBScript	IronPython

' No direct equivalent is available as
' far as the author knows
	# an IronPython dictionary is a collection of
	# name value pairs. Just like arrays, there is
	# no restriction on the keys or the values.
	# For purposes of Ansys EM scripting however,
	# all keys must be strings
	
	# delimiters are curly braces
	# use a ":" between the key and the value
	# separate key value pairs with a ","
	myDict = {
	"a" : 1,
	"b" : "hello there",

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	"c" : [1, 2, "abc"] }

Boolean Values

VBScript	IronPython
' Boolean literals are **in lower case** true false	# The first letter is capitalized True False

Converting Numbers to Strings and Vice Versa

VBScript	IronPython

' Use **CInt, CDbl, CBool, CLng**
' to convert the string representation
' to **the number representation. Use**
' **IsNumber to check before conversion**
Dim nStr = "100"
Dim n = CInt(nStr)
 	
' **Use CStr to convert a number to**
' its string representation
Dim v, vStr
v = 100
vStr = CStr(v)
	# use the integer() or float() or double()
	# functions to cast a string CONTAINING the
	# string representation of whatever you are
	# casting to.
	strInt = "3"
	intVal = int(strVal)
	floatVal = float(strVal)
	
	# invoke the str() function with the int/float
	# values as needed. You can alternately use
	# the string formatting method listed below
	strVal = str(42)
	strVal = str(42.345)

String formatting/concatenation

VBScript	IronPython

' string concatenation uses the &
' operator
Dim allStr, str1
str1 = " how are you"
allStr = "Hello " & " There" & str1
	# if you have two strings, you can always
	# concatenate then using the '+' operator
	str1 = "hello"
	str2 = "world"
	str12 = str1 + " " + str2

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

' there seems to be no direct string
' formatting function in VBScript
' using string concatenation or using
' Replace are the two **builtin options**
Dim fmt = "{1} climbs stalk {2}"
Dim str = Replace(fmt, "{1}", "jack")
str = Replace(str, "{2"}, 10)
	# if you have different types though, string
	# and int say, you must use the string
	# formatting commands. When formatting
	# multiple arguments, they must be entered
	# as a tuple (item1, item2,)
	num = 10
	str3 = "%s climbs stalk %d" % ("jack", num)
	str4 = "%d stalks" % num

Looping over lists

VBScript	IronPython

Dim myArray(0 to 2) as String
myArray(0) = "alpha"
myArray(1) = "bravo"
myArray(2) = "charlie"
 	
For Each i in myArray
Print i
Next
	vals = [1, 3, 3.456]
	
	def process(val):
	  return 2*val
	
	# is of the form
	# for variable_name in array ':'
	# < indent> statement1
	# <indent> statement2
	for i in vals:
	  print i
	  print " -> " process(i)

Looping over a range

VBScript	IronPython
' Loop over a range, specify start, end ' and step For i = 0 To 10 Step 1 Print i	# prints out values from 0 through 9 for i in range(0, 10): print i

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Next	

Related Topics

A Note About Indentation

Obtaining More Information

Discovering Methods

Help on a Method

Introduction to IronPython

Translating Script commands from VBScript to IronPython

Scripting Using IronPython: Putting it all Together

IronPython Samples

A note about indentation

Python is one of the languages where whitespace (spaces, tabs etc) are syntactically significant. You must understand the basics of indentation before scripting in python.

Any statement that introduces a block of code should be written such that every line of the block has the same indent (leading spaces or tabs) and the indent should be at least one more than the indent of the introducing statement.

```
# define a function that starts at 0 indentation.
def multInt(a,b):
# every line following the def multInt which is expected
  # to be a part of the function, must have the indent used
  # by the first line of the function (3 spaces)
  # here we introduce one more block, the if condition
  # each line that belongs to the body of this func should
  # have an indent that is more than the indent used by the
  # if statement
  If a%2 == 0:
    # I am using an indent 2 more than the parent. i.e. 5
    # spaces in total
    return (a * b) + 100
  else:
    return (a * b) + 1000
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 27-1 Sample Script 1: Describing python indentation

Related Topics

Obtaining More Information

Discovering Methods

Help on a Method

Introduction to IronPython

Translating Script commands from VBScript to IronPython

Scripting Using IronPython: Putting it all Together

IronPython Samples

Obtaining more Information on Python

Reading a book and searching online are two of the standard options. There are several very good python tutorials online and the command window is also a great way to quickly execute code and learn by doing.

Much of the official python documentation is embedded into each python distribution and the command window can also be used to get more help. Select **Tools** > **Open Command Window** to open the IronPython Command Window . You can also acces the command window executable, **Ipy.exe**, from the **IronPython** directory under the Ansys EM product installation directory. You can use this window to interactively enter Python commands (any of the pure python samples in this document)

Related Topics

Discovering Methods

Help on a Method

Introduction to IronPython

Translating Script commands from VBScript to IronPython

Scripting Using IronPython: Putting it all Together

IronPython Samples

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Discovering Methods

For instance, if you want to list all methods available in the string module, you would type in "import string" to import the module first, then type in "dir(string)". You will get a list of all the methods available (as well as some __somename__ internal names that can be ignored)

```
>>> import string
>>> dir(string)
['Formatter', 'Template', '_TemplateMetaclass', '__builtins__', '__doc__', '__file__', '__name__',
'__package__', '_float', '_idmap', '_idmapL', '_int', '_long', '_multimap', '_re', 'ascii_letters',
'ascii_lowercase', 'ascii_uppercase', 'atof', 'atof_error', 'atoi', 'atoi_error', 'atol',
'atol_error', 'capitalize', 'capwords', 'center', 'count', 'digits', 'expandtabs', 'find',
'hexdigits', 'index', 'index_error', 'join', 'joinfields', 'letters', 'ljust', 'lower', 'lowercase',
'lstrip', 'maketrans', 'octdigits', 'printable', 'punctuation', 'replace', 'rfind', 'rindex', 'rjust',
'rsplit', 'rstrip', 'split', 'splitfields', 'strip', 'swapcase', 'translate', 'upper', 'uppercase',
'whitespace', 'zfill']
```

Related Topics

Obtaining More Information

Help on a Method

Introduction to IronPython

Translating Script commands from VBScript to IronPython

Scripting Using IronPython: Putting it all Together

IronPython Samples

Help on a Method

Once you know a function name (maybe you see something you like from the dir(string) listing in ***Discovering Methods***), you can get more help on it using the builtin **help** method. For instance, executing **help(string.split)** in the **IronPython Command Window** displays the following:

```
>>> help(string.split)
Help on function split in module string:

|  split(s, sep=None, maxsplit=-1) |      split(s [,sep [,maxsplit]]) -> list of strings
|
|          Return a list of the words in the string s, using sep as the
|          delimiter string.  If maxsplit is given, splits at no more than
|          maxsplit places (resulting in at most maxsplit+1 words).  If sep
|          is not specified or is None, any whitespace string is a separator.
|
|          (split and splitfields are synonymous)
|
|
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Obtaining More Information

Discovering Methods

Introduction to IronPython

Translating Script commands from VBScript to IronPython

Scripting Using IronPython: Putting it all Together

IronPython Samples

Translating Script commands from VBScript

This section briefly describes Desktop scripting methods and arguments via VBScript samples. The distinctions made here are significant and come in use later when translating scripts written in VBScript to IronPython.

Related Topics

Introduction to IronPython

IronPython Mini-cookbook

Scripting Using IronPython: Putting it all Together

IronPython Samples

Script Method Argument

Script method calls in VBscript are described later in this document, they generally take this form:

```
objectName .methodName ( arg1, arg2, ..)
```

The function call syntax is a standard followed by several programming languages, however, the argument types in VBScript objects, when used for product scripting, are restricted to the following

Primitive types

Named Array

Named Function

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Primitive Types

Primitive types are the standard **bool, int, float, double** and **string**

Named Array

Named arrays are a special construct used very commonly and can be found in practically every recorded script sample.

It starts with **Array("NAME:someName"** and is followed by a collection of comma separated values which can be:

- A primitive value
- An array of primitive values
- Another named array
- A key, of the form "**keyName:="** followed by
  - A primitive value
  - A function (described next)

Named Function

Named functions are arrays which start with **Array(** and do not have a leading "NAME:name" item. **They are always introduced by a key** and can contain comma separated values of the following type:

- A primitive value
- A key (of the form "**keyName:="**) followed by
  - A primitive value
  - Another function (nested function)

Related Topics

Translating Script commands from VBScript to IronPython

VBscript Method Call Types

VBScript method calls fall into two categories and the distinction between the two results in syntax differences. These syntax differences are significant when converting VBScript to IronPython.

VBScript Functions

In VBScript terminology, functions return values. The syntax for this is the one shared with practically all programming languages.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
Set oDesktop = oAnsoftApp.GetAppDesktop()
Set oProject = oDesktop.NewProject
```

Figure 27-2 Sample Script 2: VBScript function call sample

Note that the significant item here is that the method name is *always* followed by an argument list enclosed in parentheses *if there are arguments*. If the argument list is empty as shown above for the *NewProject* call, the parentheses can be omitted.

VBScript Sub-Routines

VBScript Sub-Routines are those that do not have any return value. VBScript allows these to be written without any parentheses even if they have a non-empty argument list.

```
oModule.CreateReport "XY Plot1", "Standard", "XY Plot", "optimtee :
optimtee", _
  Array("Domain:=", "Sweep"), Array("Freq:=", Array("All"),
  "offset:=",_
  Array("0uin")), Array("X Component:=", "Freq", "Y Component:=", _
  Array("dB20(S(1,1))", "dB20(S(1,2))", "dB20(S(1,3))", _
     "dB20(S(2,1))", "dB20(S(2,2))", "dB20(S(2,3))", "dB20(S(3,1))",
     "dB20(S(3,2))", "dB20(S(3,3))")), Array()
```

Figure 27-3 Sample Script 3: VBScript Sub-Routine sample

Related Topics

Translating Script commands from VBScript to IronPython

Converting VBScript Function calls to IronPython Syntax

When converting functions, the important point to remember is that IronPython function names, when used for scripting, are **always** followed by parentheses. So:

- If you see a VBScript snippet that looks like a VBScript Sub-Routine, remember to add parentheses.
- If you see a VBScript function that has no arguments and no parenthesis, remember to add them around an empty argument list.

The parentheses change is the only one to keep in mind when converting VBScript function calls syntax to IronPython.

Return Values

Primitive Method Arguments

Named Array Argument

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Named Array Values with All Key Value Pairs

Named Arrays with Nested Named Arrays

Function Blocks

Return Values

VBscript return values are sometimes assigned via the **Set** declaration. IronPython return values are simple assignment (see the cookbook chapter)

Related Topics

Converting VBScript Function calls to IronPython Syntax

Primitive Method Arguments

Replace each VBScript primitive with an equivalent IronPython primitive. The main thing to notice here is that Boolean values in IronPython have their first letter capitalized.

```
True instead of true and False instead of false
```

Related Topics

Converting VBScript Function calls to IronPython Syntax

Named Array Argument

The recommended approach is to replace a VBScript array with a Python array. The mapping is:

- Change **Array(** to **[** and close with a **]** instead of the **)**
- Remove the line continuation symbols: **_**
- Map Boolean values correctly

```
oEditor.CreateCone Array("NAME:ConeParameters", "XCenter:=", "0mm",
_
"YCenter:=", "0mm", "ZCenter:=", "0mm", "WhichAxis:=", "Z",
"Height:=", "2mm",_ "BottomRadius:=", "1.56204993518133mm",
"TopRadius:=", "0mm"), Array("NAME:Attributes", "Name:=", "Cone1",
"Flags:=", "", "Color:=", _
"(132 132 193)", "Transparency:=", 0, "PartCoordinateSystem:=", _
"Global", "UDMId:=", "", "MaterialValue:=", _
"" & Chr(34) & "vacuum" & Chr(34) & "", "SolveInside:=", true)
```

Figure 27-4 Sample Script 4: Create cone recorded snippet

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For instance, method and named VBScript arrays in the snippet above are translated to

```
oEditor.CreateCone(
   [
     "NAME: ConeParameters",
     "XCenter:=" , "0mm",
     "YCenter:=" , "0mm",
     "ZCenter:=" , "0mm",
     "WhichAxis:=" , "Z",
     "Height:=" , "2mm",
     "BottomRadius:=" , "1.56204993518133mm",
     "TopRadius" , "0mm"
   ],
   [
     "NAME:Attributes",
     "Name:=" , "Cone1",
     "Flags:=" , "",
     "Color:=" , "(132 132 193)",
     "Transparency:=" , 0,
     "PartCoordinateSystem:=" , "Global",
     "UDMId:=" , "",
     "MaterialValue:=" , "\"vacuum\"",
     "SolveInside:=" , True
])
```

Figure 27-5 Sample Script 5: Create a cone in IronPython

Note that the formatting (which helps readability immensely) is not really needed. All that **had** to be done was

- Add the parentheses since the VBScript subroutine omits it
- Replace the **Array( )** delimiters with **[ ].**
- Remove the **Char(34)** function (which introduced a double quote) and replace it with the escaped double quote **\"** literal.
- Replace **true** with **True**
- Remove the line continuation symbol, **_**

Related Topics

Converting VBScript Function calls to IronPython Syntax

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Named Array Values with All Key Value Pairs

Generally, you cannot replace the arrays and nested arrays with Python dictionaries, in the case where the named array consists entirely of key value pairs (like the sample above), you can use a dictionary and avoid typing the trailing ":=" symbols after the keys. This further aids readability of the script.

- If all key value pairs
- Remove the trailing ":=" after each key
- Replace the "," after the key with a ":"
- If the named array is the top level argument, ensure that the "NAME:name" is present and is split into "NAME" : "name" as a key value pair
- Enclose the converted array in a { } pair to declare the dictionary.

```
oEditor.CreateCone(
   {
   "NAME" : "ConeParameters",
   "XCenter" : "0mm",
   "YCenter" : "0mm",
   "ZCenter" : "0mm",
   "WhichAxis" : "Z",
   "Height" : "2mm",
   "BottomRadius": "1.56204993518133mm",
   "TopRadius" : "0mm"
   },
   {
   "NAME" : "Attributes",
   "Name" : "Cone1",
   "Flags" : "",
   "Color" : "(132 132 193)",
   "Transparency" : 0,
   "PartCoordinateSystem": "Global",
   "UDMId" : "",
   "MaterialValue" : "\"vacuum\"",
   "SolveInside" : True
   }
```

Figure 27-6 Sample Script 6: CreateCone in IronPython using dictionary syntax

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

Converting VBScript Function calls to IronPython Syntax

Named Arrays with Nested Named Arrays

- Split the "NAME:name" field into a key value pair
- Translate array key value pair to a dictionary key value pair.
- Create a new key with the name of the nested array and keep the nested array (as an array or as a dictionary) as its value. If the nested array is being retained **as an array**, **the**"NAME**:name**" **field should be retained in the array**. If the nested array is being converted to a dictionary, the name is optional: if also retained in the nested array, it must match the outer key.

```
[ "NAME:name",
  "key1:=" , 1,
  "key2:=" , 2,
  ["NAME:name2", "R:=", 255]
]
```

Figure 27-7 Sample Script 7: Named array with nested named array in array syntax

The above named array with a nested named array (after conversion to IronPython as named array) can be converted to a dictionary as well. The dictionary can take any of the following forms

```
{ "NAME" : "name",
  "key1" : 1,
  "key2" : 2,
  "name2" : ["NAME:name2", "R:=", 255]
}
```

Figure 27-8 Sample Script 8: Named array with nested named array as mixed dictionary + array

```
{ "NAME" : "name",
"key1" : 1,
"key2" : 2,
"name2" : {"R" : 255}
}
```

Figure 27-9 Sample Script 9: Named array with nested named array in all dictionary syntax

```
{ "NAME" : "name",
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
"key1" : 1,
"key2" : 2,
"name2" : {
     "NAME" : "name2",
     "R" : 255
   }
}
```

Figure 27-10 Sample Script 10: Nested named array with optional "NAME:" field

Related Topics

Converting VBScript Function calls to IronPython Syntax

Function Blocks

Function blocks in VBScript argument syntax are represented as arrays without the "NAME:.." field. However, functions are always introduced by a key in a parent structure. Function blocks can therefore never exist as a top-level argument. They are only found as the value pairs inside a named array or inside another function block.

> **Note:**
>
> Function blocks and their items cannot be converted to dictionaries even though they might be composed entirely of key value pairs.

The reason for this is the need to main the user-entered order. Every item in a function block is expect to be transmitted to the script method in exactly the same order as typed out and this is impossible to achieve when a dictionary is used (as the keys get reordered according to the dictionary's internal tree/key sorting scheme).

> **Note:**
>
> When you see a function block, replace the Array( ) delimiters with the python array delimiters [ ]

Related Topics

Converting VBScript Function calls to IronPython Syntax

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Scripting Using IronPython: Putting it all Together

If you have existing VBScript/Javascript scripts use existing scripts them as much as possible by either embedding the test into the IronPython script or invoking them.

Translating a Script in VBScript to IronPython: Putting it all Together

Read the chapter on translation and study the samples in that chapter as well as those in the appendix. For python syntax and the differences, the mini-cookbook chapter will also be useful.

Writing an IronPython Script from Scratch

Read through the scripting guide available from the product's help menu and translate the VBScript methods described to IronPython using the information provided in the chapter on translation. Studying the samples in the document will also prove helpful.

For Python syntax and the differences, the mini-cookbook chapter will also be useful.

IronPython Script Execution Environment

Scripting using Embedded VBScript or JavaScript

Scripting with IronPython

Standalone IronPython and Desktop IronPython

Related Topics

Introduction to IronPython

IronPython Mini-cookbook

Translating Script commands from VBScript to IronPython

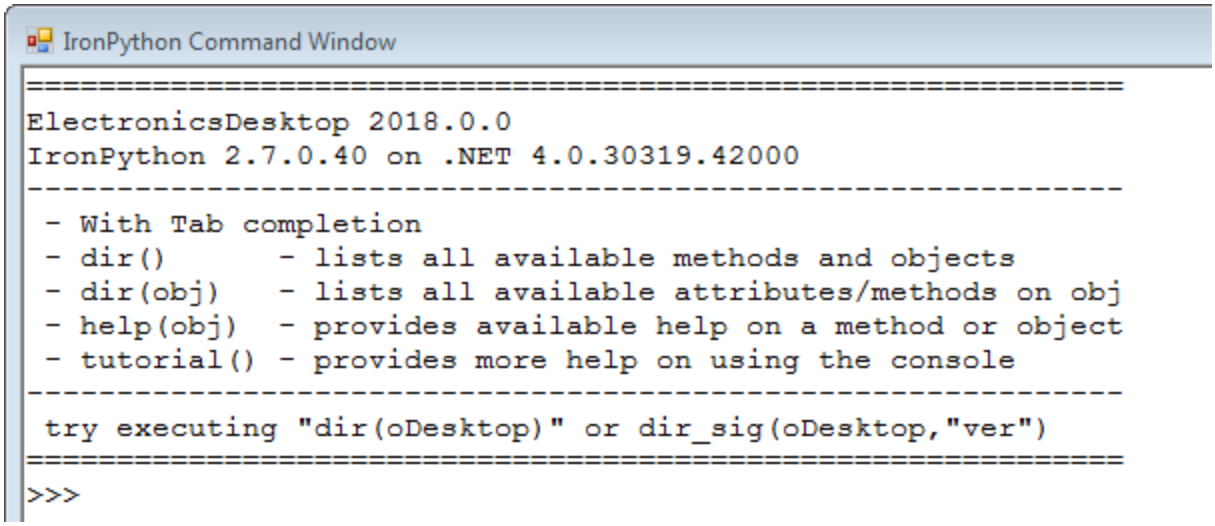
Appendix: IronPython Samples

IronPython Script Execution Environment

Scripts written in IronPython are executed by desktop in four different ways:

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- **Tools** > **Open Command Window**, to open the **IronPython Command Window**



- **Tools** > **Run Script** menu item, select **IronPython** from the file type drop down
- Launch the product with a script argument, such as **HFSS -runscript someScript.py**
- Register an IronPython script as an external tool using the **Tools > External Tools** menu item.

When Desktop executes a script, it does so in an execution environment setup with predefined variables and functions. These predefined variables and functions are how the script communicates with the desktop and they come in three flavors.

Script Argument for IronPython

Script Argument for IronPython

When scripts are launched using the **Tools > Run Script** menu item, the resulting dialog box lets you specify arguments.

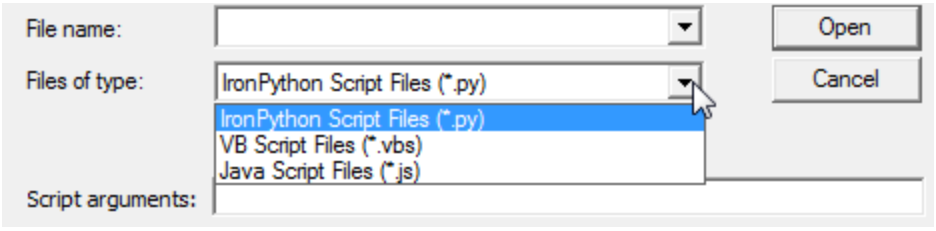


Figure 27-11 Figure 1: Run Script dialog and script arguments

Any argument specified here is communicated to the script being executed as the predefined variable **ScriptArgument**.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Related Topics

IronPython Script Execution Environment

Scripting Using Embedded VBScript or JavaScript

Since script recording is done in VBScript and you are expected to have a significant collection of VBScript or JavaScript assets, it is useful to continue to use existing script files and snippets even when scripting in IronPython. The various **Run<*>Command** methods are designed for this purpose.

For instance, you can create a parameterized cone in HFSS by executing the following IronPython script from the **Tools** > **Run Script** menu.

```
# assign the VBScript snippet obtained from a script recording from
HFSS to
# coneScript and replace the BottomRadius recorded value with
botRadius
coneScript = """Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule
Set oAnsoftApp = CreateObject("AnsoftHfss.HfssScriptInterface")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.GetActiveProject()
oProject.InsertDesign "HFSS", "HFSSPyTestDesign", "DrivenModal", ""
Set oDesign = oProject.SetActiveDesign("HFSSPyTestDesign")
Set oEditor = oDesign.SetActiveEditor("3D Modeler")
oEditor.CreateCone Array("NAME:ConeParameters", _
 "XCenter:=", "0mm", "YCenter:=", "0mm", "ZCenter:=", "0mm",_
 "WhichAxis:=", "Z", "Height:=", "2mm", _
 "BottomRadius:=", "3mm", _
 "TopRadius:=", "0mm"), Array("NAME:Attributes", "Name:=", _
 "Cone1", "Flags:=", "", "Color:=", "(132 132 193)",
"Transparency:=", 0, _
 "PartCoordinateSystem:=", "Global", "UDMId:=", "",
"MaterialValue:=", _
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
 "" & Chr(34) & "vacuum" & Chr(34) & "", "SolveInside:=", _
 true)
"""

SetScriptingLanguageToVBScript()
RunScriptCommand(coneScript)
```

Sample Script 11: Hybrid VBScript + IronPython scripting: parameterized Cone Creation

Even though recorded VBScript is used for scripting, the incremental functionality that is provided using IronPython is the ability to write a GUI using IronPython/.NET, collect user information, and modify or generate the VBScript commands to actually script the Ansys EM desktop. This GUI functionality is cross platform and a significant positive. The following example demonstrates a contrived use of a .NET window form to display the argument supplied to the IronPython script (via the **ScriptArgument** variable).

```
#import the CLR references
import clr
clr.AddReference("System.Windows.Forms")

from System.Windows.Forms import Application, Form, Label, Button,
DockStyle

# the GUI form to show some text
# the class below derives from From (System.Windows.Forms.Form)
# imported above from the .NET assembly.
class ShowPropertiesForm(Form):
  def __init__(self, name, text):
    self.Name = name
    self._label = Label()
    self._label.Text = text
    self._label.Dock = DockStyle.Fill

    _button = Button()
    _button.Text = "Close"
    _button.Dock = DockStyle.Bottom
    _button.Click += self._buttonPressed

    self.Controls.Add(self._label)
    self.Controls.Add(_button)
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
  def _buttonPressed(self,sender,args):
    self.Close()

#--------------------------------------------------
# Main script code
#--------------------------------------------------
#display the ScriptArgument variable as the text label
# in the form.
gui = ShowPropertiesForm("Sample Form", ScriptArgument)

# This makes it a modal dialog.
gui.ShowDialog()

# the following will make it a non-modal dialog
#Application.Run(gui)
```

Sample Script 12: Demonstrates the use of a .NET form from IronPython

While creating cross platform user interfaces from scripts is one of the main motivations driving the adoption of IronPython, you can use any .NET assembly.

While this hybrid approach is useful when you have existing VBScript commands that you want to reuse or when you want to quickly parameterize a recorded sample, the one significant limitation of this approach is the inability to capture return values from VBScript or JavaScript calls that do return something. Full two way communication with the product requires the use of pure IronPython to directly invoke the script objects as described below.

Related Topics

IronPython Script Execution Environment

Scripting with IronPython

While this section talks about directly interacting with the script objects, note that you can execute VBScript or Javascript at any point using any of the available Run*Command functions. using your existing script assets in this fashion and mixing with IronPython code for new functionality as needed is a viable and option.

Access to the application scripting objects is provided via the predefined **oDesktop** object (as listed in Script Objects). Interacting with the script objects is very natural, method calls are made just like in VBScript except that the argument syntax is somewhat simplified to follow natural

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Python syntax. All primitive types (string, integer, double) map to the natural primitive types in Python. The only differences from the VBScript syntax are seen when specifying array type arguments. The differences are described in earlier chapters.

> **Note:**
>
> The typical VBScript calls to:
>
> - Obtain the registered COM scripting interface via CreateObject calls and
> - Obtain the oDesktop object from it using the GetAppDesktop()
>
> are not needed (or even supported on all platforms). Since all scripting occurs in the context of a running workbench, the available Desktop object is always provided and expected to be used directly.

Scripting using the IronPython scripting API is very much like scripting with VBScript except that

- Any argument is supplied via the built in **ScriptArgument** variable
- The **oDesktop** object is always available
- The scripting method names are identical to the ones used with VBScript
- Method calls, while the name is the same have to adhere to the rule of ensuring trailing parentheses irrespective of whether the function returns anything or has any arguments.
- Any compound/block arguments should be translated to the appropriate IronPython array or dictionary syntax.

The samples section lists a collection of pure IronPython snippets; these, along with the various script snippets listed in this document should serve as a guide and reference.

Related Topics

IronPython Script Execution Environment

Standalone IronPython and Desktop IronPython

Standalone IronPython and Desktop IronPython

In general, it is easier to run a script directly from Electronics Desktop. Standalone IronPython does not implement all the functionality available when a script is run from Ansys Electronics Desktop. It only implements full support for COM functions.

Running Standalone IronPython

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Standalone IronPython uses COM to get the handle to the AnsysEDT app. To run standalone IronPython, you'll need to call the IronPython interpreter **ipy64.exe**.

It is located in:

```
<AnsysEDTInstallationPath>\<version>\<platform>\common\IronPython\ipy64.exe
```

For example, to run **myScript.py**, type the following in the command line:

```
"C:\Program Files\AnsysEM\v242\Win64\common\IronPython\ipy64.exe"
"<filePath>\myScript.py"
```

Double-click the **.py** script to set the interpreter to be the default program. You can use any recorded script as the basis for a standalone script and add an installation-internal path to the python module search path (as shown below) and end the script with a new shutdown call.

Using a Recorded Script

A python script recorded in AnsysEDT already has the required lines to be run as a standalone, except for the first two lines (path settings) and the final `Shutdown()` call. See the example script below.

Creating an External Script

When creating a script outside of Ansys Electronics Desktop, the following lines should be included at the beginning of your script:

- `import sys`

  # Imports the sys module containing system-specific functions native to IronPython.

- `sys.path.append("<InstallationPath>")`

  # Adds the Electronics Desktop installation path to the list of directories Python searches for modules and files.

- `sys.path.append("<InstallationPath>/PythonFiles/DesktopPlugin")`

  # Adds the PythonFiles/DesktopPlugin subfolder to the list of directories Python searches for modules and files.

- `import ScriptEnv`

  # This imports ScriptEnv.py from the installation path specified above. ScriptEnv.py performs an operating system check and defines functions used in Electronics Desktop scripts. See the annotations in the ScriptEnv.py file for more information.

- `ScriptEnv.Initialize("Ansoft.ElectronicsDesktop")`

  or `ScriptEnv.InitializeNew(NonGraphical=True)`

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

`Initialize` and `InitializeNew` are functions within ScriptEnv.py. Use the first option to launch Electronics Desktop. Use the second to run a script without launching Electronics Desktop. See the annotations in the ScriptEnv.py file for more information.

You must end the script with:

- `ScriptEnv.Shutdown()`

  # This stops ScriptEnv.py. If you are running multiple scripts, include this only at the end of the last script.

Example Script

```
import sys

sys.path.append(r"C:\Program Files\AnsysEM\v242\Win64")

sys.path.append(r"C:\Program
Files\AnsysEM\v242\Win64\PythonFiles\DesktopPlugin")


import ScriptEnv

ScriptEnv.Initialize("Ansoft.ElectronicsDesktop")

oDesktop.RestoreWindow()

oProject = oDesktop.NewProject()

oProject.InsertDesign("HFSS", "HFSSDesign1", "DrivenModal", "")

oDesign = oProject.SetActiveDesign("HFSSDesign1")

oEditor = oDesign.SetActiveEditor("3D Modeler")

oEditor.CreateRectangle(

[

"NAME:RectangleParameters",

"IsCovered:= ", True,

"XStart:= ", "-0.2mm",

"YStart:= ", "-3mm",

"ZStart:= ", "0mm",

"Width:= ", "0.8mm",

"Height:= ", "1.2mm",
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
"WhichAxis:= ", "Z"

],

[

"NAME:Attributes",

"Name:= ", "Rectangle1",

"Flags:= ", "",

"Color:= ", "(132 132 193)",

"Transparency:= ", 0,

"PartCoordinateSystem:=", "Global",

"UDMId:= ", "",

"MaterialValue:= ", "\"vacuum\"",

"SolveInside:= ", True

])

oDesign.SetDesignSettings(['NAME:Design Settings Data', 'Allow
Material Override:=', True, 'Calculate Lossy Dielectrics:=', True])

oEditor.SetModelUnits(['NAME:Units Parameter', 'Units:=', 'mil',
'Rescale:=' , False ])

ScriptEnv.Shutdown()
```

IronPython Samples

Change Property

The following snippets show how a change property command (in this case, to change the color of a cone) looks in VBScript and its two possible IronPython variants.

```
oEditor.ChangeProperty Array("NAME:AllTabs", Array
("NAME:Geometry3DAttributeTab",_
  Array("NAME:PropServers", "Cone1"), _
  Array("NAME:ChangedProps", _
  Array("NAME:Color", "R:=", 255, "G:=", 255, "B:=", 0))))
```

Figure 27-12 Sample Script 13: ChangeProperty command to change color of a cone in VBScript

```
oEditor.ChangeProperty(
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
["NAME:AllTabs",
  ["NAME:Geometry3DAttributeTab",
  ["NAME:PropServers", "Cone1"],
  ["NAME:ChangedProps",
  ["NAME:Color", "R:=", 0, "G:=", 0, "B:=", 64]
  ]
]
    ])
```

Figure 27-13 Sample Script 14: ChangeProperty command to change color of cone using Python arrays

Any time there are named arrays composed purely of key-value pairs, they can always be represented using a Python dictionary, irrespective of the nesting of said named array.

```
oEditor.ChangeProperty(
  ["NAME:AllTabs",
  ["NAME:Geometry3DAttributeTab",
    ["NAME:PropServers", "Cone1"],
    ["NAME:ChangedProps",
    {
    "NAME":"Color",
    "R" : 0,
    "G" : 64,
    "B" : 0
  }]]
])
```

Figure 27-14 Sample Script 15: ChangeProperty command to change the color of a cone using Python arrays and dictionaries

Create a Cone using IronPython

Most scripting tasks using IronPython should be formatted as the following example. One starts with the predefined **oDesktop** object and drills down to the design, editors, modules, and so on, then issues any required commands on the object while formatting the script command arguments in natural python syntax.

```
oProject = oDesktop.GetActiveProject()
oDesign = oProject.InsertDesign("HFSS","Random","DrivenModal","")
oEditor = oDesign.SetActiveEditor("3D Modeler")
oEditor.CreateCone(
{
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
"NAME" : "ConeParameters",
"XCenter" : "0mm",
"YCenter" : "0mm",
"ZCenter" : "0mm",
"WhichAxis" : "Z",
"Height" : "2mm",
"BottomRadius" : "1.56204993518133mm",
"TopRadius" : "0mm"
},
{
"NAME" : "Attributes",
"Name" : "Cone1",
"Flags" : "",
"Color" : "(132 132 193)",
"Transparency" : 0,
"PartCoordinateSystem": "Global",
"UDMId" : "",
"MaterialValue" : "\"vacuum\"",
"SolveInside" : True
}
)
```

Figure 27-15 Sample Script 16: IronPython script to create a cone

Create geometry then create a grid from it using copy/paste/move

The following script demonstrates slightly more advanced use of scripting and the use of return values from script methods. It creates a 5x5 grid of cones and also demonstrates the adding of information messages to the application's message window.

```
oProject = oDesktop.GetActiveProject()
oDesign = oProject.InsertDesign("HFSS","Hersheys
Kisses","DrivenModal","")
oEditor = oDesign.SetActiveEditor("3D Modeler")

# create the first cone
AddInfoMessage("Creating first cone")
firstConeName = "firstCone"
coneBotRad = "1.5mm"
oEditor.CreateCone(
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
  {
  "NAME" : "ConeParameters",
  "XCenter" : "0mm",
  "YCenter" : "0mm",
  "ZCenter" : "0mm",
  "WhichAxis" : "Z",
  "Height" : "2mm",
  "BottomRadius": coneBotRad,
  "TopRadius" : "0mm"
  },
  {
  "NAME" : "Attributes",
  "Name" : firstConeName,
  "Flags" : "",
  "Color" : "(132 132 193)",
  "Transparency" : 0,
  "PartCoordinateSystem": "Global",
  "UDMId" : "",
  "MaterialValue" : "\"vacuum\"",
  "SolveInside" : True
  }
  )

# Now replicate this a few times and create an array out of it
AddInfoMessage("Replicating it 24 times")
for x in range(5):
  for y in range(5):
       # leave the first one alone in it's created
       # position
       if x == 0 and y == 0:
                  continue

# all other grid positions, replicate from the
# first one

# copy first
oEditor.Copy(
```


Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
   {
   "NAME" : "Selections",
   "Selections" : firstConeName
   }
)

# paste it and capture the pasted name
# the pasted names come in an array as we could
# be pasting a selection cmposed of multiple objects
pasteName = oEditor.Paste()[0]

# now move the pasted item to it's final position
oEditor.Move(
   {
   "NAME" : "Selections",
   "Selections" : pasteName
   },
   {
   "NAME" : "TransalateParameters",
   "CoordinateSystemID" : -1,
   "TranslateVectorX" : "%d * 3 * %s" % (x, coneBotRad),
   "TranslateVectorY" : "%d * 3 * %s" % (y, coneBotRad),
   "TranslateVectorZ" : "0mm"
   }
)

# Now fit the display to the created grid
oEditor.FitAll()
```

Figure 27-16 Sample Script 17: Sample script to create a cone and use copy/paste/move to replicate it.

Related Topics

Introduction to IronPython

IronPython Mini-cookbook

Translating Script commands from VBScript to IronPython

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Scripting Using IronPython: Putting it all Together

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

28 - Twin Builder Terminology

This section defines the terminology used in the Twin Builder help topics. Terms are listed in alphabetical order.

Glossary: A

A/D (Analog-to-Digital)

The process of converting an analog value into its digital equivalent.

AC simulation

Harmonic analysis of a model.

AC simulator

Twin Builder simulator for AC analysis.

Action type

Defines how an action in a state is processed.

.afa file

Analytical Frequency Analysis file containing data of an analytical frequency analysis.

Analytic frequency step response

Twin Builder module to compute frequency step response information for a given transfer function.

Animated Symbol

Symbol for a component or a macro that changes to reflect changes in values assigned to it. You can modify the symbol with an interaction button or by a system value during the simulation. Animated symbols can be generated using the Symbol Editor.

.aws file

Simulator file containing initial values of capacitors and inductors.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Glossary: B

Backplane

Common communication and control structure for all Twin Builder simulators.

Basic version

Twin Builder basic version without any optional modules.

Basics

Twin Builder library containing the basic modeling elements for the Twin Builder simulators.

Behavioral model

Smallest model unit that cannot be subdivided further.

Best representation

Function for scaling all output channels of a graph window to maximum size.

Bezier

Bezier curves are used in computer graphics to produce curves which appear reasonably smooth at all scales. The curves are constructed as a sequence of cubic segments, rather than linear ones. Bezier curves use a construction in which the interpolating polynomials depend on certain control points. The mathematics of these curves is classical, but it was a French automobile engineer Pierre Bezier who introduced their use in computer graphics.

Block

Linear or nonlinear transfer function, basic element of block diagrams.

Block diagram

Combination of blocks to describe the dynamic behavior of systems.

Block Diagram Module (BDM)

Twin Builder sub simulator to analyze simulation models described using block diagrams. Uses Euler formula and distributed integration algorithms.

Bookmark

Can be used in the Twin Builder editor to mark a position and find it again.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

.brs file

External Schematic file to parametrize a model sheet.

Glossary: C

Characteristic

Function or data set to describe a nonlinear characteristic of a Twin Builder component, block or other model.

Characteristic values

Analysis method that provides a list of frequently used characteristic values of a system quantity.

C-Interface

C/C++ programming interface for the integration of user defined nonlinear models or components into the simulation model.

C-Model Editor

Twin Builder application for the definition of models in the C/C++ language.

Color scheme

Predefined or user defined color settings for screen outputs or printing.

Compiler

Program for the translation of the SML description of a simulation model into a simulator specific format.

Computation sequence

Sequence in which blocks in a simulation model are computed.

Connection rule

Rule determining which element can be connected to another under certain conditions.

Conservative node

Connection of two or more circuit component terminals.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Coordinate system

Reference system for the display of numerical values. It can be linear or logarithmic.

Crossing over

Exchange of genetic material between chromosomes in a genetic algorithm.

Cursor

Positioning element to determine the value of a quantity in a coordinate system.

Glossary: D

Data cache

Saves the data of the last simulation run.

Data channel

Simulation data for a specific quantity stored in a file or transferred to an active element.

Data filter

Function to select data by user defined criteria.

Data format

Defines how data is stored or exchanged between programs.

Data set

Simulation data of a simulation run at a given time step.

DC simulation

DC operating point analysis of a model.

DC simulator

Twin Builder simulator for DC analysis.

Differentiation

Analysis method in the Post Processor.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

DISPLAY

Twin Builder library containing display elements to display simulation data inside a schematic.

Display Element

Element for graphic or numeric online visualization of simulation data on sheet.

DLL

Dynamic link library containing program components or models. It is loaded upon request.

Dongle

Software protection device connected to the printer port of the computer.

DSDE

Dynamic Simulation Data Exchange interface used to integrate external simulators on a client-server-level.

Glossary: E

Electric circuit

Combination of electric components, connected by ideal wires.

Electric Circuit Module (ECM)

Twin Builder sub simulator to analyze simulation models described using electric circuits. Uses Euler or Trapezoid algorithm and modified nodal approach.

Element

Any item that can be placed on a Schematic sheet, including models, text elements, display elements, drawing elements, subsheets, and so on.

Entity

VHDL-AMS term to describe the interface of a behavioral or structural model.

Euler formula

Numerical integration algorithm used inside Twin Builder.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Evaluation function

Function evaluating the quality of a solution compared to the defined optimum.

Expert mode

Extended mode for the definition of an experiment. It allows fine tuning of parameters and methods.

Export filter

Special program for the export of simulation data into other applications.

Extern View

Special sheet independent oscilloscope for the display of simulation data.

Glossary: F

FFT

Fast Fourier Transformation.

File output

Saves a system quantity online during the simulation in a file.

Fitness function

Function describing the fitness of an individual (parameter constellation) in a genetic algorithm.

Formula Module (FML)

Twin Builder’s integrated expression evaluator.

Frequency step response analysis

Special mode in the Experiment tool, determining the frequency behavior of a simulation model.

Glossary: G

Genetic Algorithm

Optimization method of the experiment tool with automatic parameter variation and target function determination.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Grid lines

Grid lines in a coordinate system.

Glossary: H

HMAX

Simulation parameter, maximum step size for the integration algorithm.

HMIN

Simulation parameter, minimum step size for the integration algorithm.

Glossary: I

ID

Numeric value for the distinction of sheet elements.

Information window

Display of warnings, errors, program status for the Twin Builder modules.

Initial condition

Defines the initial value for energy storing electrical components.

Initial state

A state that is active at the beginning of the simulation.

INT

Simulation parameter, defines the integration algorithm used for the simulation of electric circuits.

Interactivity pad

Part of an animated symbol, used to change the behavior and/or shape of a symbol (model component).

Iteration

Part of the integration process for the solution of nonlinear problems.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Glossary: J

Jacobian Matrix

Coefficient matrix for the numerical integration algorithm.

Glossary: K

Keyword

Can be used to do a search in a model database.

.krn file

Simulator file containing simulation status information. It can be used as a starting point for the next simulation.

Glossary: L

Language concept

Settings to define the use of language for program (menus and dialogs) and libraries.

Library

Database containing a set of Twin Builder basic elements and/or macro models.

Local discretization error (LDF)

Simulation parameter, determines the accuracy of the computation according to the dynamics of the electric circuits.

.log file

Experiment tool file containing the protocol of an experiment.

Glossary: M

Macro

Contains one or more elements of a model description that can be used as single elements.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Main skeleton (.skl)

Special file containing basic descriptions for the generation of the model description file. Must not be modified.

Maximum current error (IEMAX)

Simulation parameter that determines the accuracy of the right side computation of the differential equation system for current lines.

Maximum number of iterations (Iteratmax)

Simulation parameter that limits the number of iteration loops for the nonlinear iteration process.

Maximum voltage error (VEMAX)

Simulation parameter that determines the accuracy of the right side computation of the differential equation system for voltage lines.

.mda/.mdk file

Twin Builder data file of simulation results and characteristics (former format).

.mdx file

Twin Builder data file of simulation results and characteristics.

Model

All elements in the libraries and subsheets (except for display elements).

Model tree

Hierarchical list box containing the available elements for the graphical modeling.

Module

Twin Builder sub simulator or program.

Monitor

Special window for the display of simulation status and progress.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Monte Carlo Analysis

Optimization method of the experiment tool with automatic parameter generation and characteristic value determination.

.mtx file

Twin Builder data file of the DES model.

Multi simulation

Analysis method of the experiment tool, batch mode computation of a simulation model with different, user defined parameter sets.

Mutation

Part of the genetic algorithm that generates new individuals (parameter sets) by randomly modified parameters.

Glossary: N

Network installation

Installation process of a Twin Builder network version.

Newton-Raphson-Algorithm

Nonlinear iteration algorithm used in Twin Builder.

Non-conservative node

Connection of two or more non-circuit component terminals.

Normal mode

Default display of the experiment tool without evaluation functions and storage options.

Glossary: O

Object

Element of the graphic model description linked to another WINDOWS application (OLE).

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Object browser

Special window for the display and browsing of the elements of a graphical model description.

Offline

When the simulation is not running in Twin Builder.

Online

During the simulation in Twin Builder.

Online graphic

Online display of simulation results during the simulation run.

Online graphic output

Display of a system quantity during the simulation in a Display element or the ViewTool.

Option

Optional Twin Builder application. It can be registered using the installation manager.

Output

Definition of a specific quantity of the simulation model to be used as an output.

Glossary: P

Password

Letter and digit combination to access Twin Builder.

Petri net

Special form of a state machine.

Pipe

Data channel for the data transmission between Twin Builder modules.

Postprocessing

Data evaluation and processing after a simulation run has finished.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Preprocess

Specialized modules for information processing and parameter determination to define Twin Builder model components.

Preprocessor directive

Special commands for the SML compiler to include files, extract macros from the model library, etc.

Print colors

Colors used to print a system quantity from an active element or the ViewTool.

Project

Organizational structure containing all files and information belonging to a simulation task.

Glossary: Q

Qualifier

References a system quantity or parameter of a Twin Builder model component.

Quality criterion

Characteristic value of a system quantity used to determine the quality of an optimization run.

Queue

Display the active and all waiting simulation runs.

Glossary: R

Recombination

Part of the genetic algorithm. It generates new individuals (parameter sets) by crossing of two parent individuals.

Roll back

Go back to a previous simulation step.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Glossary: S

Sample time

Step size for digital controller.

Schematic

Twin Builder module for the graphical model definition.

Screen colors

Colors used to display a system quantity online during the simulation in an active element or the ViewTool.

Section

Part of the SML description containing model information for the Twin Builder sub simulators.

Selection

Part of the genetic algorithm. It selects individuals of the active generation to be transferred to the next generation by specific selection criteria.

Simulation backplane technology

Twin Builder software architecture for data exchange and program control of several simulators in one environment.

Simulation model

Graphical or text description of a real system using modeling capabilities of a simulation system.

Simulation parameter

Parameter used to control the simulation process.

Simulator

Software for the analysis of the behavior of a system using a simulation model.

Simulator coupling

Direct link of one or more simulators using the Twin Builder simulation backplane technology.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Simulator interface

Special software interface for the integration of external simulators into Twin Builder.

.smd file

Legacy Twin Builder 7 ModelAgent file containing data of a model library.

SML

Twin Builder Modeling Language.

SML Editor

Twin Builder application for the definition of models in SML language.

.sml file

File containing the model description of a simulation run.

SML key word

Special terms used to determine the sections of an SML description.

Solver

Algorithm for the computation of model components without the capability to roll back steps.

.ssc file

Twin Builder project file, containing all information about a project.

.ssh file

Twin Builder schematic file, containing the graphical representation of a model and simulation data.

State

Basic element of state graphs that defines properties and activities in a certain system state.

State graph

Combination of states and transitions. A modeling language for discontinuous systems.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

State Graph Module (SGM)

Twin Builder sub simulator to analyze simulation models described using state graphs basing on the PETRI net theory.

Status line

Displays the present state of a Twin Builder application.

Subsheet

A Twin Builder sheet embedded into another Twin Builder sheet, connected via pins, creates a macro inside the **.sml** file.

Sub-simulator

Twin Builder internallycoupled simulator.

Sub-skeleton

File containing rules for the creation of a non-Twin Builder description language.

Symbol editor

Twin Builder application for the creation or modification of a symbol.

Symbol level

Group of drawing elements of an animated symbol displayed together depending on the input value or the user activity on a interaction pad.

Symbols

Graphic representations of elements on the Schematic, placed from the model library. They can be modified using the symbol editor.

Synchronization

Update of a schematic from an older Twin Builder version to the latest symbols.

Synchronization

Detection of events in a state graph.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

System quantity

Any quantity computed by the simulator.

System simulation

Simulation level, where models from different physical domains are simulated at the same time.

System variable

Predefined variables inside Twin Builder that cannot be used as a variable name or a specifier.

Glossary: T

Task

Analysis to be performed as part of an experiment. It can contain several simulation runs.

Template

Predefined structure for a Twin Builder application.

TEND

Simulation parameter that determines the simulation end time.

Time Function Module (TFM)

Twin Builder sub simulator for the computation of time dependent functions.

Time limited

A license of the simulation software for a certain (limited) time period.

Time step

Present time step size used to compute the next results vector.

Toolbar

Bar in a Twin Builder application to perform activities in the application.

Total fitness

The total fitness of an optimization run.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

TR simulation

Transient analysis of a model.

TR simulator

Twin Builder simulator for TR analysis.

Transition

Cross over condition between the input and output state(s) of a state graph, defined by a logical expression.

Transition component

Basic structure of a state machine. It comprises a transition and all of its input and output states.

Trapezoid Formula

Numerical algorithm used inside Twin Builder.

Glossary: U

UDMinit

Section of the C/C++ interface for the model initialization.

UDMMain

Section of the C/C++ interface containing the model computation algorithm.

User defined component (UDC)

Model description for electrical components using a user defined C/C++ program, with direct access to the solver matrix.

User defined model (UDM)

Model description for nonlinear characteristics using a user defined C/C++ program.

User management

Saves the workspace for individual users.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Glossary: V

Version report

Special mode inside the Twin Builder help system to create a detailed information file about the present Twin Builder installation.

VHDL

Very high-speed integrated circuit Hardware Description Language for digital systems.

VHDL Model Editor

Twin Builder application for the definition of models in the VHDL-AMS language.

VHDL-AMS

Very high-speed integrated circuit Hardware Description Language - Analog Mixed Signal. Extension of VHDL, a hardware description language for digital and analog systems.

VHDL-AMS simulator

Simulator integrated with Twin Builder's backplane that calculates simulation models described in VHDL-AMS.

ViewTool

Program to display results online during the simulation outside the Schematic.

Glossary: W

Window elements

Windows in the workspace containing various information. You can turn them on or off.

workflow

Graphic representation of a sequence of activities that are performed on a certain data set.

Worst Case Analysis

Analysis method with parameter combination of all extreme values for the parameters and characteristic value determination.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Glossary: Y

YMAX

Maximum value used for display in the online graphic using the View tool.

YMIN

Minimum value used for display in the online graphic using the View tool.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Index

*

**(function) 25-10

.

.amat file 4-118

.csv file format 16-43, 20-115

.dat file format 20-115

.mdx file format 16-39

.rdat file format 20-115

.tab file format 20-115

.txt file format 20-115

.xls file format 16-44

.xlsx file format 16-44

<

<Notation>#DEFINE 23-9

<Notation>#INCLUDE 23-8

<Notation>#SET 23-8

3

3D Report, spinning 20-20

A

aborting analyses 17-19

about symbols 6-52

ABOVE attribute 22-60

ABS (function) 25-10

Absolute Value, Function 25-10

AC analysis settings 16-12

AC analysis setup 16-10

AC analysis, adding 16-12

AC solution options 16-27

Access function to a characteristic (function) 25-10

ACOS (function) 25-10

ACOSH (function) 25-10

active analysis, setting 16-19

adding

Maxwell 2D Finite Element Model Subcircuit 13-35

adding a DC analysis 16-15

Adding a subcircuit 11-89

adding a transient analysis 16-7

Adding a VHDL-AMS Model 6-108

adding an AC analysis 16-12

Adding Datasets Manually 4-81

adding solution options 16-32

advanced analysis types (Optimetrics) 16-5

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

AFTER 22-51

Alias for file names 22-62

Alias for model library name 22-62

Aligning Component Dialog Box Elements 9-28

analyses

- changing parameter values during 17-19, 17-21
- clean stop 17-20
- monitoring 17-17
- Optimetrics 19-1
- pausing 17-19
- re-solving 17-111
- resuming 17-19
- starting 17-1
- stopping 17-19

Analysis options, setting 4-30, 4-34

analysis setup

- deleting 16-22
- editing 16-20
- enabling or disabling 16-20
- renaming 16-21

analysis types

- advanced (Optimetrics) 16-5
- standard 16-4

Analyze All command, using 17-2

AND (logic operator) 25-10

ANF file format 4-67

ANG_DEG (function) 25-10

ANG_RAD (function) 25-10

Ansys Electronics Desktop 2-3

- ribbon interface 2-55

Ansys Esterel Component Subcircuits 13-114

Ansys Fluent LTI Component Subcircuits 13-109

Ansys RBD Component Subcircuits 13-108

Arc cosine, Function 25-10

Arc sine, Function 25-10

Arc tangent, Function 25-10

Arc tangent2, Function 25-10

ARCHITECTURE statement 22-9

archiving projects 4-73

ARG (function) 25-10

Argument of complex number, Function 25-10

Arithmetic Functions 25-10

Arithmetic operators 25-10

Arranging components 13-19, 13-19, 13-20, 13-20, 13-20, 13-20

Arranging symbol elements 6-56

ASIN (function) 25-10

ASINH (function) 25-10

Assignment operators 25-10

ATAN (function) 25-10

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

ATAN2 (function) 25-10

ATANH (function) 25-10

Attribute declaration 22-24

Attributes 22-48

For quantities 22-48

For signals 22-49

auto-save file 4-72

B

backplane, simulator 16-1

Basic Dynamic IGBT

Advanced Settings 7-89

Basic Dynamic IGBT, Characterizing 7-82

batchoptions file format 4-55

BLAC Motor Drive System 13-114

block diagram simulator processing 16-2

BLOCK statement 22-25

Bookmarks 14-4

bulk conductivity 4-119, 4-123, 4-127

bus entry objects, drawing 11-42

C

calculation range

setting for a cost function 19-98

setting in a parametric setup 19-11

Calibration Wizard 18-45

category, traces 20-108

Cell Filtering 18-22

changing parameter values during simulation 17-19, 17-21

characteristics

in component dialog boxes 11-29

in simulation models 11-29

Characterization 7-1

Characterizing a Basic Dynamic IGBT, Preliminary Considerations 7-82

Check Component Dialog, using 9-47

checking connectivity 11-39

Checking Design Size 16-47

Chip Model Analyzer 20-222

circuit simulator processing 16-2

clean stop 17-20, 17-20

clearing messages 2-39

closing a component dialog box 9-48

CMA 20-222

coherent gain 20-10

command-line options 2-65

Common logarithm, Function 25-10

Comparison operators 25-10

Compile an SML Model from a Schematic 11-65

Complex Functions 25-10

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

component
placing 11-14, 11-14
wiring 11-37
COMPONENT declaration 22-24
Component Dialog Box
adding check boxes 9-15
adding ComboBoxes 9-10
adding control elements 9-2
adding groups 9-4
adding list boxes 9-23
adding radio buttons 9-21
adding static information 9-2
adding text edits 9-7
change name 9-3
Close 9-48
default behavior 9-40
display behaviors 9-37
lock behavior 9-37
relative behavior 9-39
revising 9-49
Save 9-48
setting the tab order 9-35
testing 9-46
using 9-49
Component Dialog Box elements
aligning 9-29
setting display behavior 9-37
sizing 9-29
Component Dialog Grid 9-29
Component Dialog Wizard 9-1
create 9-2
Layout Toolbar 9-52
menu bar 9-49
size controls 9-32
Standard Toolbar 9-51
using 9-2
window items 9-49
Component Editor 6-37
Component import 4-97, 6-123
Component instantiation statement 22-29
Component Libraries window 2-16
Component names 25-1
component parameters, displaying and editing 11-16
Component properties
editing 6-43
Component update 4-103
components
editor 6-37
removing unused definitions 4-79
Components Tab 2-18
Concurrent statements 22-25
ASSERT statement 22-27
BLOCK statement 22-25

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

BREAK statement 22-30

Component instantiation statement 22-29, 22-29

Concurrent procedure call statement 22-27

Concurrent signal assignment statement 22-28

PROCESS statement 22-26

configuration files, setting options 4-37

configuring AC simulations, guidelines for 16-9, 16-11

configuring DC simulations, guidelines for 16-15

Connecting components 13-19, 13-19, 13-20, 13-20, 13-20, 13-20

connectivity, checking 11-39

Connector library 6-20

CONSTANT declaration 22-20

constants, predefined 25-8

constraints, setting linear 19-203

control elements, add to Component Dialog Box 9-2

controlling simulations 17-17

Controls Menu, Component Dialog Wizard 9-51

conventions, design 25-1

Conversion

From degrees electrical to seconds (function) 25-10

From degrees to radians (function) 25-10

From radians to degrees (function) 25-10

Conversion Functions 25-10

copy command, for report and trace definitions 20-96

copy solution options 16-33

copying, subcircuits 11-87

COS (function) 25-10

COSH (function) 25-10

Cosine, Function 25-10

cost function

adding 19-94

plotting results vs. iteration 19-207

setting a goal 19-94

setting the calculation range 19-98

specifying solution quantity for 19-97

viewing results vs. iteration 19-207

count, setting for sweep definitions 19-6

coupling model files, locating 13-102

creating a new schematic 11-1

Creating Plots On Schematic 20-52

Creating Simulation Models 11-13

Creating Twin Models 11-59

creating, custom report templates 20-117

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

custom report templates, creating 20-117

D

Data Connector Component 13-138

Data object declarations 22-20

CONSTANT declaration 22-20

FILE declaration 22-22

SIGNAL declaration 22-21

VARIABLE declaration 22-21

Data table units 20-29

data tables, creating 20-27

Data types 22-47

dataset expressions, adding 4-81

Dataset preview plot properties 4-82

datasets

adding 4-81

Adding Manually 4-81

Importing 4-82

modifying 4-85

DC analysis settings 16-15

DC analysis setup 16-14

DC analysis, adding 16-15

DC Bias Values, Viewing 16-16

DC solution options 16-27

ddt (function) 25-10

decade count sweep definitions 19-6

Declarations 22-18

default variable value, overriding for a parametric setup 19-8

DEFINE command 23-9

definitions

removing unused 4-79

updating from the library 6-6

definitions folder 2-16

DEGEL (function) 25-10

Deleting

Bookmarks 6-71, 14-3

deleting an analysis setup 16-22

deleting reports 20-114

deleting solution options 16-34

delta between markers, in reports 20-79

dependent variables, definition 10-4, 10-8, 10-13, 10-14

derivative, Function 25-10

design 11-1

hierarchical 11-1

page 11-1

synchronizing incoming subcircuit 11-99

Design Area 2-41

Design Configurations 11-100

design conventions 25-1

Design of Experiments 19-145, 19-166

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Design Settings 11-103

Design units 22-7

design variables

deleting 10-12

See local variables 10-8

design variations

manually modifying points 19-7

viewing all in a parametric setup 19-4

Design Xplorer, exporting setup 19-210

designs

in project tree 2-15

inserting in project 4-63

moving between 11-98

desktop

status bar 2-47

Desktop Configuration 4-3

Device Characterization Wizard 7-1

Diagram Editor 12-1

dialog box, View/Edit Material 4-119, 4-123, 4-127

dialog element, selecting display behavior 9-41

dielectric loss tangent 4-119, 4-123, 4-127

Directories settings 4-10

disabling an analysis setup 16-20

display behaviors, component dialog box 9-37

display types, of reports 20-22

Displaying pins 11-31

Displaying properties 2-36

distributed analysis, licensing 17-11

distributed machine configurations, editing and creating 17-13

distribution criteria, setting for statistical setups 19-139

DOT attribute 22-3

dynamic coupling subcircuit

adding 13-22

Maxwell Dynamic Eddy Current 13-59

Maxwell Dynamic Electrostatic 13-55

Maxwell Dynamic Magnetostatic 13-52

Q3D SML 13-68

RMxprt 13-65

Dynamic ROM Components 13-137

E

Edit Menu, Component Dialog Wizard 9-50

editing

materials 4-118

symbol, schematic 6-52

editing an analysis setup 16-20

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Editing component properties 6-43

Editing Pin Properties 6-68, 6-68

editing solution options 16-33

editor

model 2-44

script 2-45

Electric Rule Check 11-39, 11-39

enabling an analysis setup 16-20

Encoding a Modelica Model 6-140

Encoding a VHDL-AMS Model 6-115

Encryption settings, VHDL-AMS 6-115, 6-115

ENTITY statement 22-8

Equations, operands and operators 25-9

ERC (Electric Rule Check) 11-39

Event Callbacks 4-111

EXIT statement 22-37

EXP (function) 25-10

expected units 25-17

exponential count sweep definitions 19-6

Exponential function, Function 25-10

Exponential Functions 25-10

Exporting a netlist 14-8

Exporting a schematic 11-105

Exporting a VHDL-AMS Model Description 4-107

exporting options files 4-36

Exporting Plot Data 20-114

expressions

piecewise linear functions in 10-14

Expressions

and variables 25-9

defining 10-13

intrinsic functions in 10-16

using as cost function goal 19-99

valid operators 10-13

External Tools 2-49

Extreme Value Analysis 19-126

F

Fast Calculation-Update Optimization 19-174

FILE declaration 22-22

file formats

.anf 4-63

.asmp 4-62

.asmpresults 4-63

.csv file 16-43

.mdx file 16-39

.smd 4-63

.ssc 4-63

.ssh 4-63

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

.xls file 16-44

.xlsx file 16-44

file, .amat 4-118

files

auto-save 4-72

finite element subcircuit 13-32

fixed variables, setting values during analyses 19-202

FMU Component Subcircuits 13-114, 13-136

FMU Components 13-120

Fractional part of a value (function) 25-10

free values, using for non-conservative input nodes 9-26

Functions 22-17

Arithmetic 25-12

Complex 25-11

Conversion 25-14

Exponential 25-14

for trace 20-79

Rounding 25-15

selecting for a quantity 20-106

standard mathematical 25-11

Trigonometric 25-11

valid operators 10-13

G

general solution options 16-28

goal

setting a complex value 19-99

setting a real value 19-99

setting a single value 19-99

setting as variable dependent 19-99

setting for cost function 19-98

setting weight of 19-94

using an expression for 19-99

goal weight, setting 19-94

Graphical objects, copy and paste properties 11-23

grid

lines 11-7

snapping 11-8

visibility 11-6

Guidelines for configuring AC simulations 16-9, 16-11

guidelines for configuring DC simulations 16-15

H

help

about conventions used 3-10

Help for Components 2-21

HFSS Component Subcircuits 13-104

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Hiding pins 11-31

hierarchical components, saving to libraries 6-16

hierarchical schematic 11-87, 11-98

High Performance Computing integration 17-49

HPC Analysis options, setting 4-30, 4-34, 4-57, 4-62

HPC and Analysis Options, setting 4-30, 4-34

HPC integration 17-49

HPC options, setting 4-30, 4-34

Hyperbolic arccosine, Function 25-10

Hyperbolic arcsine, Function 25-10

Hyperbolic arctangent, Function 25-10

Hyperbolic cosine, Function 25-10

Hyperbolic sine, Function 25-10

Hyperbolic tangent, Function 25-10

I

IEEE Encryption of a VHDL-AMS Model 6-116

If-Else (function) 25-10

IF (function) 25-10

IM (function) 25-10

Imaginary part, Function 25-10

Import / Update Models Options 4-25

Import Components dialog box 4-97, 6-123

Imported Solution Data, Importing 20-108, 20-119

importing 4-68

Importing Datasets 4-82

Importing Plot Data 20-115

Importing Setup Data 16-35

Importing Simulation Models 4-97

Importing Solution Data 16-34

INCLUDE command 23-8

indentation, IronPython 27-7

initial displacement, setting 19-124

initial seed value, setting for statistical analysis 19-136

inserting designs 4-63

INT (function) 25-10

Integer part of a value (function) 25-10

Integration, Function 25-10

intrinsic functions 10-16

Intrinsic Variables 10-17

introduction to Twin Builder 2-1

IronPython 27-1

indentation 27-7

J

Job management UI, RSM 17-101

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

K

keyboard shortcuts 2-54

Keywords, VHDL-AMS 22-51

L

Lande G factor 4-119, 4-123, 4-127

layout editor window 2-42

Layout Menu, Component Dialog Wizard 9-50

LayoutToolbar, Component Dialog Wizard 9-52

Legacy Libraries, Translating 6-8

legacy Simplorer projects

translation overview 6-8

legacy translation

terminal mismatch 4-67

legends in reports 20-19

libraries, search path precedence 6-5

library

precedence 6-3

structures and usage precedence 6-3

licensing, distributed analysis 17-11

Limit line violations 20-105

limit lines in plots 20-98

linear constraints

deleting 19-204

modifying 19-204

setting 19-203

linear count sweep definitions 19-6

linear step sweep definitions 19-6

Linux remote spawn 4-34

Listing schematic design elements 11-71

LN (function) 25-10

Loading component example projects 2-21

local variables

adding 10-8

units in definition 10-8

locating coupling files 13-102

lock controls, Component Dialog Wizard 9-29

LOG (function) 25-10

Logarithm 25-14, 25-14

Logical operators 25-10

LOOKUP (function) 25-10

LOOP statement 22-36

M

Macro Model Export 18-10

magnetic loss tangent 4-119, 4-123, 4-127

magnetic saturation 4-119, 4-119, 4-123, 4-123, 4-127, 4-127

Manage Files 6-28

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

markers, delta between markers 20-79

material

bulk conductivity 4-119, 4-123, 4-127

dielectric loss tangent 4-119, 4-123, 4-127

Lange G factor 4-119, 4-123, 4-127

magnetic loss tangent 4-119, 4-123, 4-127

magnetic saturation 4-119, 4-123, 4-127

name 4-120

properties, editing 4-119, 4-123, 4-127

relative permittivity 4-119, 4-123, 4-127

materials

creating 4-118

editing 4-118

materials editor, using 4-118

MATLAB Optimizer 19-29

MATLAB, use as optimizer 19-24

MATLAB/Simulink® coupling

Add Simulink Component 13-6

Creating a MATLAB/Simulink model 13-6

Simulation 13-9

MAX (function) 25-10

maximum number of iterations

setting for a sensitivity analysis 19-118

setting for an optimization 19-92

setting for statistical analysis 19-136

maximum step size, setting for optimization analysis 19-107

maximum variable value

changing for all setups 19-140

overriding for a sensitivity setup 19-123

overriding for all optimization setups 19-105

overriding for all sensitivity setups 19-123

overriding for an optimization setup 19-105

maximum variable, Optimetrics calculation of 19-102, 19-122

Maxwell 2D Finite Element Model Subcircuit 13-35

Maxwell Dynamic Eddy Current Coupling SubCircuit 13-58

Maxwell Dynamic Electrostatic Coupling SubCircuit 13-55

Maxwell Dynamic Magnetostatic Coupling SubCircuit 13-52

Maxwell Equivalent Circuit Component, Interface concept 13-41

Maxwell Transient Cosimulation subcircuit 13-32

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

menu bar
overview 2-48
message details 2-40
Message window
about 2-38
hiding 2-39
Messages
clearing 2-39
displaying details 2-39
Microsoft Excel file format 16-44
MIN (function) 25-10
Min and Max focus, SNLP optimizer 19-108
minimum step size, setting for optimization analysis 19-107
minimum variable value
changing for all setups 19-140
Optimetrics calculation of 19-102, 19-122
overriding for a sensitivity setup 19-123
overriding for all optimization setups 19-105
overriding for all sensitivity setups 19-123
overriding for an optimization setup 19-105
Minverva
remote storage environment 5-1
Mixed-mode parameters, displaying 18-24
Mixed-Signal Models 22-59
Model Editor Options 4-30
model editor window 2-44
Model libraries, Alias for library name 22-62
Modelica Compiler Options 4-28
Modelica Model Editor 6-133, 12-30
Changing Class 12-29
Diagram Object Information 12-29
Editing Properties 12-22
Modifiers 12-24
Modelica Models 6-133
Modeling, Aspects in Twin Builder 22-56
models
Creating simulation 11-13
Creating Twin 11-59
Mixed-signal 22-59
simulation 11-13
Twin 11-59
Modulus (function) 25-10
monitoring solution progress 17-21
monitoring solutions 17-17
Monte Carlo Analysis 19-129
moving between designs 11-98

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Multidomain modeling packages 22-14

N

names
- component 25-1
- variable 25-1

Natural logarithm, Function 25-10

NATURE declaration 22-20

ndExplorer 18-1
- displaying mixed-mode parameters 18-24

NdExplorer Macro Model Export 18-10

Netlist
- exporting 14-8
- netlist string property 6-48
- syntax 6-48

Netlist & Script Editor Settings 14-8

Netlist Editor window 2-43

Network Data Explorer 18-1
- Cell Filtering 18-22
- changing port properties 18-23
- color coded matrix plot 18-30
- comparing network data 18-37
- creating a statistics plot 18-36
- displaying across all frequencies 18-31
- displaying individual statistics 18-34
- displaying statistics by frequency 18-33
- loading data into 18-7
- modifying data 18-28
- Overview 18-3
- reducing matrix size 18-23
- right-click menu 18-26
- viewing data 18-28
- viewing matrix data 18-32

new projects, creating 4-63

NEXT statement 22-37

nominal design 19-1

NOT (logic operator) 25-10

notes, saving with project 4-110

numeric data, unit suffixes 25-16

O

octave count sweep definitions 19-6

On-sheet report
- adding an 11-82
- modifying an 11-83
- opening in a new window 11-84

opening
- example projects 4-66
- existing projects 4-64
- recent projects 4-65

operator precedence 10-13

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

operators
arithmetic 25-10
assignment 25-10
comparison 25-10
logic 25-10
Optimetrics
tuning a variable 19-198
types of analyses 19-1
viewing analysis results 19-205
optimization
Fast Calculation-Update 19-174
MATLAB use 19-24
norms, L1, L2, and Max 19-111
optimization analysis
choosing variables to optimize 10-20
optional settings 19-54
overview 19-14
plotting cost vs. iteration results 19-207
setting up 19-54
viewing cost vs. iteration results 19-207
Optimization Overview 19-14
optimization setups
adding 19-54
adding a cost function 19-94
procedure for defining 19-54
setting a goal 19-94
setting the max. iterations 19-92
solving 17-1
optimizers 19-14
Options
setting in Twin Builder 4-2
OR (logic operator) 25-10
output parameter
adding to sensitivity setup 19-119
plotting results 19-208
setting calculation range 19-121
specifying solution quantity for 19-120
viewing results in table format 19-208
output variables
deleting 20-112
specifying 20-110

P

Package
Body 22-11
Declaration 22-10
Visibility 22-12
Packages 22-10
For multidomain modeling 22-14
Standard 22-12

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Page Menu, Component Dialog Wizard 9-51

page setup, schematic editor 11-8

page, schematic 11-85

panning the schematic view 11-11, 12-5

Param Values Tab 2-24

Parameter Qualifiers 25-2

Parameter Types 25-6

parameterization, material properties 4-119, 4-123, 4-127

parameterizing, See variables 10-1

parameters

- assigning variables to 10-19
- reserved parameters 6-47

parametric analysis

- setting up 19-3
- solution quantity results 19-12

parametric setup

- adding 19-3
- overview 19-4

parametric setups

- adding sweep definitions 19-4
- adding to a design 19-4
- plotting solution quantity results 19-206
- setting the calculation range 19-11
- solution quantity results 19-12
- solving 17-1
- solving before optimization 19-109
- solving before sensitivity analysis 19-124
- solving during optimization 19-110
- solving during sensitivity analyses 19-125, 19-144
- specifying a solution setup 19-9
- specifying solution quantities for 19-9
- using results for optimization 19-109
- using results for sensitivity analysis 19-124

paste solution options 16-33

Paste Design Options 11-88

pattern search optimizer 19-14

pause 17-20

personallib 6-3

PExprt 4-115

Pin List dialog box 6-68

pin, properties, editing 6-68, 6-68

Pins, displaying and hiding 11-31

placing components 2-21, 11-15

Placing components 13-19, 13-19, 13-20, 13-20, 13-20, 13-20

Plot Data

- Exporting 20-114
- Importing 20-115

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

plots
adding limit lines 20-98
distribution results for statistical analyses 19-209
parametric solution quantity results 19-206
Plots on Schematic, Creating 20-52
Plotting Imported Solution Data 20-108, 20-119
polar plots
creating 2D 20-25
information displayed 20-26
Pop Up 11-98
port
interface 11-45
placing 11-45, 11-45
Port Options 4-16
Post-processing Data, updating 20-113
Power Diode
Advanced Settings 7-130
Characterizing 7-122
Power Mosfet
Advanced Settings 7-107
Power MOSFET, Characterizing 7-98, 7-116
Power, Function 25-10
precedence, library usage 6-3
Predefined Constants 25-8
Predefined variables 25-8
primary sweep
modifying the variable 20-105
specifying for 2D rectangular plots 20-23, 20-48
specifying for 3D rectangular plot 20-31
specifying for data tables 20-27
Primitive drawing elements 11-76
copy and paste properties 11-23
printing, from Twin Builder 4-109
Probe command 20-53
Probe Selection command 20-52
Procedures 22-15
PROCESS statement 22-26
Processes 22-56
profile information, for Optimetrics solutions 19-206
Progress window, monitoring solutions 17-17
Project Manager window
overview 2-12
showing 2-12
project tree
auto expanding 2-15
showing 2-12
project variables
adding 10-4

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

deleting 10-6

naming conventions 10-4

units in definition 10-4

projects

creating new 4-63

default names 4-1

deleting 4-78

managing 4-1

opening example 4-66

opening existing 4-64

opening recent 4-65

renaming 4-78

saving 4-69

saving automatically 4-71

saving copies 4-71

saving new 4-70

saving notes 4-110

Properties

copying and pasting 11-20

defining for components 6-43

displaying 11-19

Parameter Values Tab 2-29, 2-30

pin, editing 6-68, 6-68

report backgrounds 20-16

Properties Dialog Box 2-28

Properties Dialog General Tab 2-32

Properties Dialog Symbol Tab 2-33

Properties Dialog, Property Displays Tab 2-36

property window 2-22

Property Window General Tab 2-25

Property window Symbol Tab 2-25

Push Down 11-98

Q

Q3D SML Coupling SubCircuit 13-68

Qualifier List 25-2

qualifiers, parameter 25-2

Quantities 22-57

plotting S-parameter 20-108

Quantities tab, Properties window 2-26

Quantity attributes 22-48

quasi newton optimizer 19-14

queued simulations

removing 17-24

viewing 17-24

Quick Probe command 20-54

QuickFind 11-66

R

RAMP attribute 22-59

Range functions 10-14

re-solving a problem 17-111

RE (function) 25-10

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Real part, Function 25-10

rectangular plots

creating 2D 20-23

creating 3D 20-31

of parametric solution quantity results 19-206

Relational operators, VHDL-AMS 22-45

relative permittivity 4-119, 4-123, 4-127

REM (function) 25-10

remote analysis 17-8

remote spawn (Linux) 4-34

Removing Unconnected Wires 11-39

renaming an analysis setup 16-21

renaming projects 4-78

renaming solution options 16-34

Replay Analysis 17-4

Replay simulation 17-4

Report Data 20-112

Report Dialog Box 20-1

report templates, creating 20-117

report window 2-45

reporter window types 20-6

reports

adding traces 20-63

background properties 20-16

creating 20-1

creating 2D polar plots 20-25

creating 2D rectangular plots 20-23

creating 2D rectangular stacked plots 20-39

creating 3D rectangular plots 20-31

creating data tables 20-27

creating plots on schematic 20-52

display types 20-22

exporting plot data 20-114

file formats for plot data 20-115

finding delta between markers 20-79

importing plot data 20-115

modifying data in 20-15

overview 20-1

plotting imported solution data 20-108, 20-119

selecting a function 20-106

specifying time or frequency domain 20-10, 20-23, 20-25, 20-27, 20-31, 20-48, 20-50, 20-51

sweeping variables 20-105

updating 20-113

user defined outputs 20-122

Reserved Variables 10-17

Reserved words (VHDL-AMS) 22-51

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Reset command, in Tuning dialog box 19-200

result window 2-45

resume 17-20

RETURN statement 22-37

revising a component dialog box 9-49

ribbons 2-57

RMxprt Dynamic Coupling SubCircuit 13-64

ROOT (function) 25-10

Root, Function 25-10

Rounding Functions 25-10

RSM, Job Management UI 17-101

rule check (electric) 11-39

Running Simulations 17-1

S

S-parameters, plotting quantities 20-108

saving a component dialog box 9-48

saving projects 4-69

- automatically 4-71
- new projects 4-70
- saving copies 4-71

Saving Uncompiled Model Changes 4-105

saving, tuned states 19-200

scheduler, proxy interfaces 17-90

schematic 11-1

- adding a page 11-85
- adding an on-sheet report 11-82
- change page properties 11-85
- Circuit, creating and editing 11-1
- connecting pages 11-85
- grid setup 11-7
- hierarchical 11-1, 11-87, 11-98
- modifying an on-sheet report 11-83
- multipage, setting up 11-85
- opening an on-sheet report in a new window 11-84
- page 11-1, 11-85
- page setup 11-8
- placing components 2-21
- removing a page 11-85
- System, creating and editing 11-1

Schematic Editor User's Guide 11-1

schematic editor window 2-41

schematic editor, starting 11-4

Schematic export 11-105

script editor window 2-45

scripting, IronPython 27-1

sdt (function) 25-10

Search Tab 2-21

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

secondary sweep, modifying the variable 20-105

seed value 19-136

selecting solution options 16-19

sensitivity analysis

choosing variables to include 10-21

optional settings 19-117, 19-125, 19-129

setting up 19-117

sensitivity setups

adding 19-117, 19-125, 19-129

adding an output parameter 19-119

procedure for defining 19-117, 19-125, 19-129

setting initial displacement 19-124

setting the max. iterations 19-118

Sequential Mixed Integer NonLinear Programming (SMINLP) Optimizer 19-14

Sequential Nonlinear Programming (SNLP) Optimizer
optimizer 19-14

Sequential statements 22-31

ASSERT statement 22-31

BREAK statement 22-38

CASE statement 22-35

EXIT statement 22-37

IF statement 22-34

LOOP statement 22-36

NEXT statement 22-37

NULL statement 22-38

Procedure call statement 22-33

RETURN statement 22-37

Signal assignment statement 22-32

Variable assignment statement 22-33

WAIT statement 22-31

SET command 23-8

setting options, configuration files 4-37

setting the active analysis 16-19

settings

AC analysis 16-9, 16-12

AC solution options 16-27

adding solution options 16-32

DC analysis 16-13, 16-15

DC solution options 16-27

general solution options 16-28

standard analyses 16-5

transient analysis 16-5

transient solution 16-22

setup options, standard analysis 16-5

SGN (function) 25-10

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SheetScan Coordinate System, Define 4-91

SheetScan Toolbars 4-89

SheetScan, loading a datasheet 4-90

Shift operators 22-45

Shortcut Keys 2-50

shortcuts, keyboard 2-54

Show Component Dialog, using 9-46

Show Queued Simulations command, using 17-24

SI units 25-17

Sign dependent value, Function 25-10

Signal assignment 22-59

Signal attributes 22-49

SIGNAL declaration 22-21

Signals 22-57

Signals tab, Properties window 2-27

Simple simultaneous statement 22-39

Simplorer Modeling Language 23-1

Simulation

- MATLAB/Simulink® coupling 13-5, 13-6, 13-9
- VHDL-AMS 16-3

simulation models

- characteristics 11-29
- creating 11-13
- importing 4-97

Simulation Options 11-103

simulations

- changing parameter values during 17-19, 17-21
- clean stop 17-20
- controlling 17-17
- monitoring 17-17
- pausing 17-19
- re-solving 17-111
- resuming 17-19
- running Optimetrics 17-1
- starting 17-1
- stopping 17-19

simulator

- block diagram 16-2
- circuit 16-2
- Solvability 22-61
- state graph 16-3

simulator backplane 16-1

Simulator Performance slider, using 16-30

Simulink model subcircuits 13-3

Simultaneous procedural statement, Statement 22-42

Simultaneous statements 22-39

- CASE statement 22-40

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

IF statement 22-40

NULL statement 22-43

Simple simultaneous statement 22-39

Simultaneous procedural statement 22-42

SIN (function) 25-10

Sine, Function 25-10

SINH (function) 25-10

SIwave Component Subcircuits 13-103

size controls, Component Dialog Wizard 9-32

Sizing Component Dialog Box Elements 9-28

slider, Simulator Performance 16-30

SML Header Options 16-31

Smoothing, Network Data Explorer 18-20

snap to grid 11-6

SNLP optimizer, setting Min and Max focus 19-108

Solution Options 16-22

AC 16-27

adding 16-32

copy 16-33

DC 16-27

deleting 16-34

editing 16-33

General 16-28

paste 16-33

renaming 16-34

selecting 16-19

Transient 16-23

solution progress, monitoring 17-21

solution quantity

calculation range for optimization 19-98

calculation range for parametric setups 19-11

calculation range for sensitivity 19-121

calculation range for statistical 19-138

plotting parametric setup results 19-206

specifying for cost function 19-97

specifying for output parameter 19-120

specifying for parametric setups 19-9

specifying for statistical setups 19-137

Solution setup options 16-5

solution setups, choosing for a parametric analysis 19-9

solutions

after modifying the model 17-111

monitoring 17-17

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

pausing 17-19
re-solving 17-111, 17-111
resuming 17-19
starting 17-1
stopping 17-19, 17-19
Solvability 22-61
Solver Files Directory 20-186
solving 17-1
batch solution 2-60
parametric setup before optimization 19-109
parametric setup before sensitivity analysis 19-124
parametric setup during optimization 19-110
parametric setup during sensitivity analysis 19-125, 19-144
solving remotely 17-7
Sorting Block components 11-70
spectral 20-6
Spice Compiler Options 4-27
Spinning a 3D Report 20-20
SQRT (function) 25-10
SQU (function) 25-10
Square root, Function 25-10
Square, Function 25-10
standard analysis setup options 16-5
standard analysis types 16-4
Standard mathematical functions 25-11
Standard Toolbar, Component Dialog Wizard 9-51
starting variable value
overriding for optimizations 19-101
overriding for sensitivity 19-122
overriding for statistical 19-143
state graph simulator processing 16-3
Statements
Concurrent 22-25
Sequential 22-31
Simultaneous 22-39
statistical analysis
choosing variables to include 10-23
plotting distribution results 19-209
setting up 19-134
viewing distribution results 19-209
statistical setups
adding 19-134
procedure for defining 19-134
setting the initial seed value 19-136
setting the max. iterations 19-136
specifying solution quantities for 19-137
status bar, overview 2-47
STD STANDARD 22-13
Step Response Data File 20-175

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

step size

setting constraints for optimization 19-107

setting for sweep definitions 19-6

stopping an analysis 17-19

stopping criteria for optimization, maximum number of iterations 19-92

stopping criteria for sensitivity analysis, max. iterations 19-118

subcircuit

copying 11-87

creating 11-87

creating from a selection area 11-92

finite element 13-32

Maxwell Dynamic Eddy Current Coupling 13-58

Maxwell Dynamic Electrostatic Coupling 13-55

Maxwell Dynamic Magnetostatic Coupling 13-52

Maxwell Equivalent Circuit 13-41

Maxwell Transient Cosimulation 13-32

placing 11-87

Q3D SML Dynamic Coupling 13-68

Simulink model 13-3

Subprograms 22-15

Functions 22-17

Procedures 22-15

SUBTYPE declaration 22-19

sweep definitions, See variable sweep definitions 19-3

sweep variables in reports, modifying values 20-105

symbol editor, grid setup 11-6

Symbol Elements, Arranging 6-56

symbol, schematic

creating and editing 6-52

symbols, about 6-52

Sync # column 19-7

Synchronize Design dialog box 11-99

synchronizing sweep definitions 19-7

syslib 6-3

System constants 25-8

System Model Identification 20-175

System Model Identification MIMO 20-170

System Model Identification Toolkit - MIMO 20-173

T

TAN (function) 25-10

Tangent, Function 25-10

TANH (function) 25-10

templates for reports, creating 20-117

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

terminal mismatch 4-67

Test Machines, Distributed Analysis Machines dialog 17-13

testing a component dialog box 9-46

Thermal Model Identification 20-179

Time step guidelines 16-6

toolbar, Twin Builder Simulation 17-19

Toolkit 20-161

Tools, external 2-49

Top Menu Bar 2-48

trace characteristics 20-85

traces

- adding characteristics 20-79
- adding to reports 20-63
- categories 20-108
- copy and paste definitions 20-96
- display properties 20-66
- removing 20-98

Traces dialog box 20-1

transient analysis

- adding 16-7
- timestep guidelines 16-6

transient analysis settings 16-7

Transient analysis setup 16-5

transient solution settings 16-22

Translating Legacy Libraries 6-9

translating legacy projects 6-8

Trigonometric Functions 25-11

tuning analysis

- resetting variable values after 19-200
- reverting to a state 19-200
- saving a state 19-200
- setting up 19-198

tuning, choosing variables to tune 10-22

Twin Builder

- command-line options 2-65
- introduction 2-1
- launching 2-1
- setting options 4-2

Twin Builder Analyses 16-4

Twin Builder conventions 25-1

Twin Builder Design, setting up 4-63

Twin Builder Desktop 2-10

Twin Builder General Options 4-16

Twin Builder Modelica connector library 6-20

Twin Builder Simulation toolbar 17-19

Twin models

- creating 11-59

TYPE declaration 22-19

Type declarations

- NATURE 22-20

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SUBTYPE 22-19

TYPE 22-19

U

Unconnected Wires, Removing 11-39

Undoing commands 4-95

Unit Handling 25-17

unit suffixes 25-16

units

as part of variable definitions 10-4

expected 25-17

used 25-17

Update Components dialog box 4-103

UpdateRegistry command, setting or removing option values 4-37, 4-41, 4-45

Updating Modelica Library Definitions 12-34

Updating post-processing data 20-113

Updating Reports 20-113

used units 25-17

User Defined Documents (UDDs) 20-187, 20-187

User Defined Outputs 20-122

userlib 6-3

using a component dialog box 9-49

using Check Component Dialog 9-47

using Show Component Dialog 9-46

using the materials editor 4-118

Using, Bookmarks 14-4

V

Values on sheet 22-63

variable-dependent goal 19-99

Variable assignment statement 22-33

VARIABLE declaration 22-21

Variable names 25-1

variable sweep definitions

adding to parametric setups 19-4

manually modifying 19-7

overview 19-4

setting values to solve 19-6

synchronizing 19-7

tracking changes to 19-7

viewing all design variations 19-4

variables

adding local variables 10-8

adding project variables 10-4

assigning to parameters 10-19

choosing to optimize 10-20

choosing to tune 10-22

defining sweep definitions 19-4

deleting from design 10-12

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

deleting from project 10-6

dependent 10-5, 10-9, 10-13

excluding from Optimetrics analyses 19-201

in expressions 25-9

including in sensitivity analysis 10-21

including in statistical analysis 10-23

Intrinsic 10-17

min. and max values for optimization 19-102

min. and max values for sensitivity analysis 19-122

output 20-110

overriding default value for a parametric setup 19-8

overview 10-1

predefined 25-8

Reserved 10-17

setting default value 10-4

setting distribution criteria 19-139

setting fixed values 19-202

setting range of values 19-122

setting range of values for optimization 19-102

tuning 19-199

types in HFSS 10-1

updating to optimized values 19-110

Vector inputs on sheet 22-64

VHDL-AMS

Concurrent statements 22-25

Design units 22-7

Sequential statements 22-31

Simultaneous statements 22-39

Standard Packages and Types 22-12

VHDL-AMS Model Description, Exporting 4-107

VHDL-AMS Model Editor 6-108

VHDL-AMS simulation 16-3

VHDL Compiler Options 4-27

view navigation options 2-52

View/Edit Material dialog box 4-119, 4-123, 4-127

Viewing DC Bias Values 16-16

W

WAIT statement 22-31

Water Pumping System 13-114

window

property 2-24

result 2-45

ribbons 2-57

window items, Component Dialog Wizard 9-49

Window layouts 2-46

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

wire properties, displaying 11-38

wire, selecting 11-38, 11-38

wiring 11-37

Wizard Menu, Component Dialog Wizard 9-49

wizard, calibration 18-45

Wizard, Component Dialog 9-1

WORK library 22-62

Worst case analysis 19-125

Example 19-129

Y

Y Markers, Stacked plots, stacked plots, Y Markers 20-71

Z

zooming the schematic view 11-11, 12-5

Twin Builder 2024 R2 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.