Ansys Electromagnetics HPC Administrator's Guide

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 2025 R2
July 2025

Conventions Used in this Guide

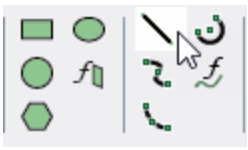
Please take a moment to review how instructions and other useful information are presented in this documentation.

- 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**, then type a space, and 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, "click **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** *filename*" means you must type the word **copy**, then type a space, and 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, and 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 **Draw** > **Line**"



This instruction means that you should click the **Line** command on the **Draw** 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 then click **Open Examples** to launch the dialog box.

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

  “Right-click and select **Assign Excitation** > **Wave Port**” means when you click the right-mouse button with an object face selected, 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.

Table of Contents

1 - Introduction

Ansys Electromagnetic products have a variety of ways to use multiple threads, multiple cores or multiple hosts to improve productivity. These additional resources may be used to solve a given problem faster, or to solve larger problems, or to improve the quality of the solution (using more stringent convergence criteria, for example). To make effective use of the resources allocated to the analysis, the job settings must match the job requirements. In addition, the cluster configuration and the resources must be allocated to the job in a way that the products are able to make effective use of the resources.

For example, consider a case in which a single host with 8 cores and 128 GB of memory is available to run the analysis. For this example, the analysis is a parametric analysis with 16 variations. Suppose that when the analysis may is run sequentially on this host using a single core, it finishes in one hour, but it requires all 128 GB of memory available on the host. Changing the configuration of the analysis to run 8 variations in parallel is unlikely to speed up the job. Although there may be sufficient CPU resources to effectively run 8 variations in parallel, there is unlikely to be sufficient memory for the job to run without severe paging and/or swapping. This may result in the job running more slowly than for the sequential analysis.

2 - General Considerations

Suspending and Resuming Jobs

In general, Ansys Electromagnetics products do not support suspending and resuming a job. However, in most cases, if an analysis is aborted and restarted, then portions of the analysis that have been completed and written to the results folder/directory will not be rerun. If the analysis does not terminate gracefully, then it may need to be restarted from the beginning. For example, if the scheduler terminates the job because a resource exceeds the limits for the job, then it may not terminate gracefully.

Changing Resources over Time

The job submission GUI for Ansys Electromagnetics does not attempt to determine how the resources (such as CPU usage or memory usage) vary over time. Instead, the user specifies the overall resource limits, and the software attempts to effectively utilize the resources allocated to the job. In many cases, a portion of the analysis may not be parallelized. During this part of the analysis, only a portion of the resources allocated to the job may be used. For example, if a frequency sweep is analyzed, a portion of the analysis may run adaptive passes for a single nominal frequency serially. The adaptive passes may only use the resources allocated to the job on a single host. After convergence, multiple frequencies may be solved in parallel, using all resources allocated to the job.

It is not easy to predict in advance how resource utilization will vary over time. As a result, it is not generally useful to specify resources that increase or decrease as the job runs.

Distribution Types

There are a number of ways that an analysis may be distributed across multiple threads, cores, or hosts. These vary depending on the specific product, the details of the project, the design and setups, and the configuration of the job. The type of distribution may have an effect on how resources may be effectively allocated to a job.

Ansys Electromagnetics products support using multiple distribution types at the same time, with some limitations. For any given phase of the analysis, only two distribution types may be used at

the same time. The top level distribution type may be Frequencies or Optimetrics Variations. The second level distribution type may be any distribution type allowed for the product other than Frequencies or Optimetrics Variations.

Multithreading

Ansys Electromagnetic products are able to take advantage of multithreading (MT), provided that sufficient licenses are available. The speedup achieved using multithreading may drop off if the number of parallel threads is large, due to contention for locks or communication overhead between threads. Specifying more threads than there are physical or logical cores available to the job may be counterproductive.

Multithreading may only be applied if the threads are running on the same host. Therefore, a job configured for distribution using multithreading will not be able to effectively use the cores allocated to the job unless the cores are on a single host.

Frequencies

Frequency sweeps represent one of the most common distribution types. After analysis is finished at one frequency, analyses at different frequency points are essentially independent. If the resources allocated to the job have sufficient processing power and memory, multiple frequency points may be analyzed in parallel tasks. The frequencies may be distributed to different hosts, with little communication required between the tasks.

Optimetrics Variations

Optimetrics variations are another type of distribution where the analyses are essentially independent. After the nominal analysis is finished, analyses of different variations may be distributed to different tasks, with little communication required between the tasks. The variations may be may be distributed to different hosts, because little communication is required between the tasks.

Other Distribution Types

There are a number of other distribution types. The allowed distribution types depend on the product and on the design type. Some of these distribution types (e.g., Domain Solver or DDM) may allow an analysis that is too large to solve on a single host to be distributed to multiple hosts. For other distribution types, distribution may allow more resources to be applied to a given problem to achieve the solution faster. For most of these distribution types, MPI must be configured to support distributed analysis. More communication may be required between the distributed tasks than for Frequencies or Optimetrics Variations. As a result, inter-node communication may become significant if the tasks are distributed across many nodes.

Two-level Distribution

Ansys Electromagnetics products attempt to minimize communication between processes in order to maximize performance. For two level distributed analysis, an attempt is made to group the processes distributed at the second level on the same hosts, as much as possible. This is to reduce the overhead due to communication between these processes. The processes distributed at the first level (Frequencies or Optimetrics Variations) require very little communication between processes, while processes at the second level may require more communication between processes. As a result, each group of second level processes analyzing the same frequency or variation are grouped on the same host or hosts, as much as possible.

Specifying Tasks Manually

Generally, a task is a single process that may run in parallel with other tasks. The cores allocated to a task may be used for multithreading. There are two options for specifying tasks manually; the second option may not be available for all schedulers. For the first option, the user may specify the overall number of tasks and the number of cores per task. For the second option, the user may specify the total number of tasks and total number of cores for each specific host.

For the first case, each task may only run on a single host, and the number of concurrent threads used by the task will be limited to the specified number of cores. Multiple tasks may run on a single host, if there are sufficient cores available on the host. If the analysis is memory bound, rather than CPU bound, performance may be improved by limiting the number tasks per node. If the user specifies a limit on the number of tasks per node, then cores required per task are scaled up to enforce the limit. For example, if the job requests 4 cores per task, and the limit on the number of tasks per node is set to 2, and at least one of the nodes available to the job has 12 cores, then each task will require 6 cores, because 12 cores divided by 6 cores per task, results in a maximum of two tasks per host. For this case, the resources allocated to the job should be the same on each host. Some schedulers limit how resources may be requested for a job.

For the second case, the user specifies the specific nodes to allocate to the job, including the total number of tasks and total number of cores for each node. The total cores for each node should be equal to or greater than the total tasks for that node. For this case, there may be different numbers of core specified for tasks running on different nodes. If the number of cores allocated to a node is not a multiple of the number of tasks allocated to the node, then the tasks running on that node may use different numbers of cores.

Specifying Tasks Automatically

For these cases, the user specifies the resources for the job, and the Ansys Electromagnetics software automatically determines how the tasks should be distributed to the available resources. There are three options for specifying tasks automatically. For the first option, the

user may specify the total number of cores for the job. For the second option, the user may specify both the total number of nodes and the total number of cores for the job. For the third option, the user may specify the total number of cores to allocate to the job for each specific host. The third option may not be available for all schedulers.

For the first case, the user specifies the total number of cores for the job. For some schedulers, the user may optionally specify the RAM per core for the job.

Multiple tasks may run on a single host, if there are sufficient cores available on the host. If the analysis is memory bound, rather than CPU bound, performance may be improved by limiting the number tasks per node. If the user specifies a limit on the number of tasks per node, then cores required per task are scaled up to enforce the limit. For example, if the job requests 4 cores per task, and the limit on the number of tasks per node is set to 2, and at least one of the nodes available to the job has 12 cores, then each task will require 6 cores, because 12 cores divided by 6 cores per task, results in a maximum of two tasks per host. For this case, the resources allocated to the job should be the same on each host. Some schedulers limit how resources may be requested for a job.

For the third case, the user specifies the specific nodes to allocate to the job, including the total number of cores for each node. For this case, the software will determine how many tasks to create, how the tasks are distributed to the available nodes, and how the available cores on each node are allocated to the tasks on that node. There may be different number of tasks on each node and/or different numbers of cores allocated to each task.

GPU Acceleration

Several Ansys Electronics Desktop solvers can benefit from GPU acceleration:

- HFSS frequency domain, transient, and SBR+ solvers.
- HFSS 3D Layout for matrix solves and eye analysis.
- Maxwell 3D AC Magnetic matrix solves.
- Circuit Design for eye analysis.
- Circuit Netlist for eye analysis.
- EMIT

For details on the requirements for GPU use, see "GPU Acceleration" above.

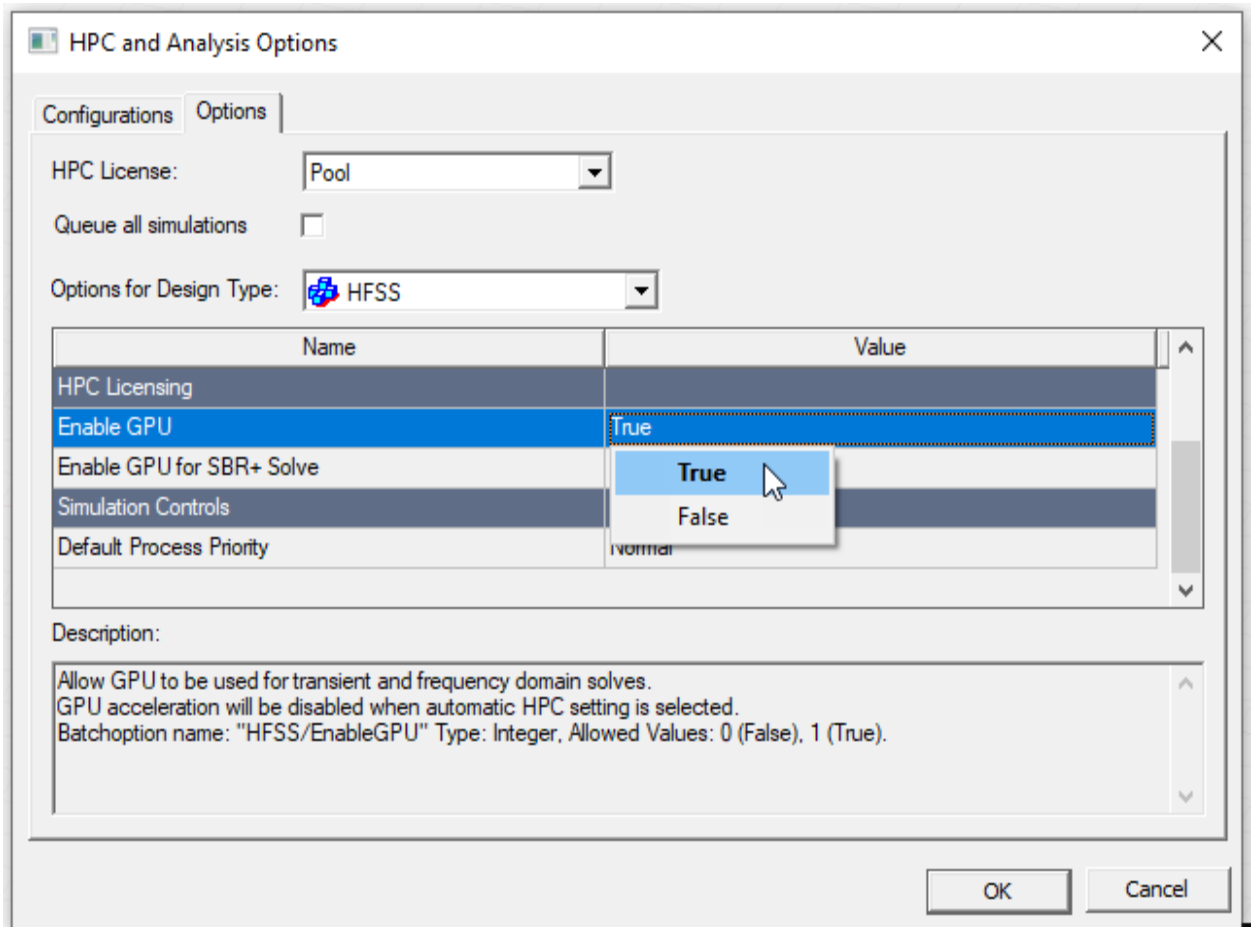
License Options

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. For details, see the discussion of licensing under Options in Setting HPC and Analysis Options.

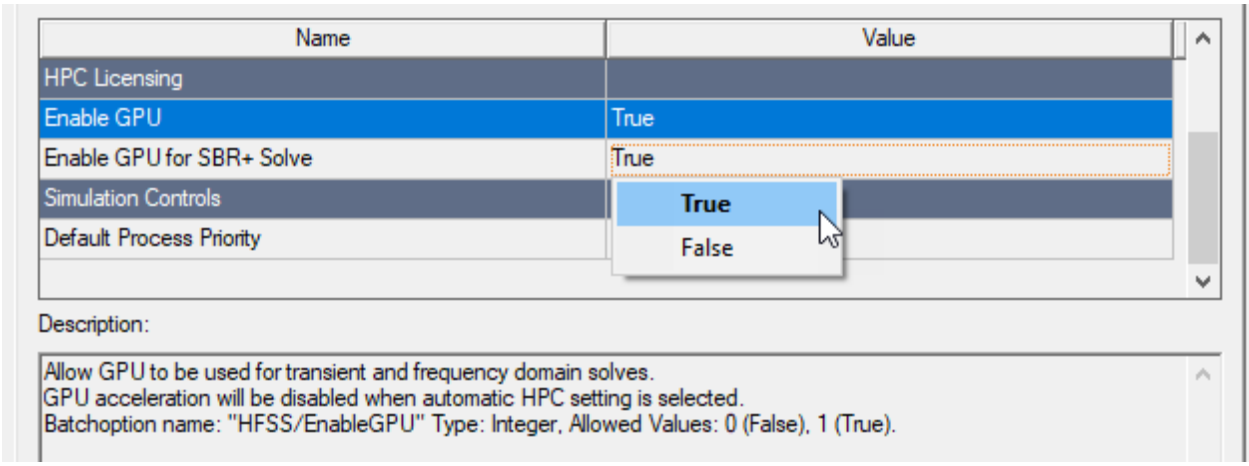
If the number of simulation jobs exceeds the number of GPUs in a system, the excessive jobs will fall back to CPUs and will be accelerated by up to 8 CPU cores for each job.

Enable/Disable GPU Acceleration from the Ansys Electronics Desktop Solver User Interface

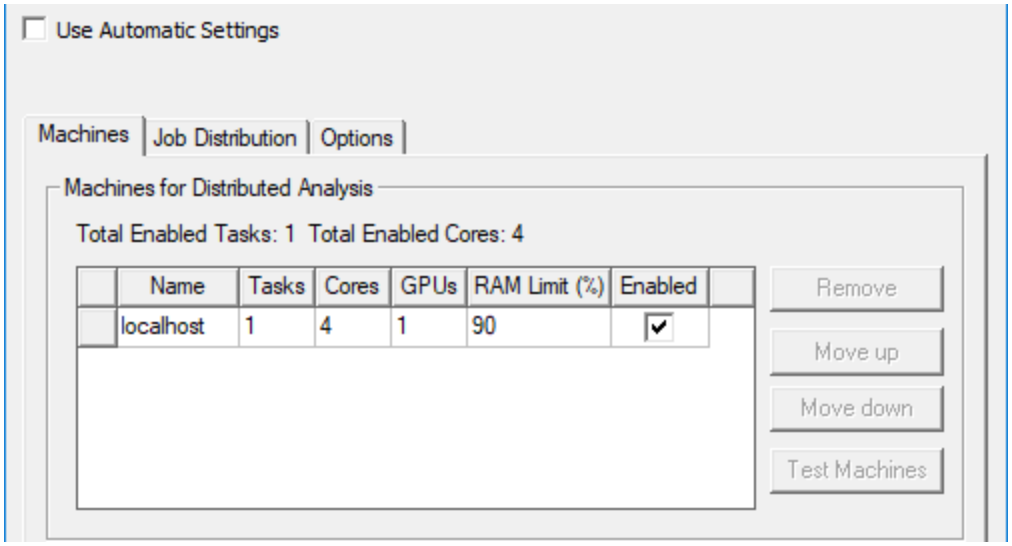
To turn on or off GPU acceleration for supported solvers, click **Tools>Options>HPC and Analysis Options** and select the **Options** tab. Click on the value of Enable GPU and toggle it to either True or False. HFSS includes a separate toggle for Enable GPU for SBR+ Solve.



For HFSS SBR+ projects, you can also Enable GPU for SBR+ solves.



To enable use of GPU or MIC acceleration, you can set **Enable GPU** to **True** and set **Use Automatic Settings** to **off** in the Analysis Configuration dialog box. The matrix solver automatically determines if all cores should be used, or if one GPU should be used to give the best performance. For example, for a Maxwell AC Magnetic Solution Matrix, if you specify 4 cores for the simulation, the 3D AC Magnetic solver will use 4 cores in parallel during matrix assembly while the matrix solver will use either 4 cores or 1 GPU.



For HFSS, **Use Automatic Settings** supports SBR+ Solution types only. The GPU column is always shown, but only applies when solving the SBR+ solution type. For all other solution types, the GPU count is ignored, and treated as if set to zero.

For HFSS SBR+ in auto mode, each machine has an allocated number GPUs, but no preset number of tasks. In this case, since HFSS SBR+ acceleration is mainly through GPUs, the

variation distribution prioritizes use of GPUs. In other words, the distribution divides the GPUs equally among the variations, although it may not be possible for every variation to have the same number of GPUs. Below the level of variations, the allocated GPUs per machine will be passed to the solver's auto algorithm, which then allocate these GPUs to the distributed solver instances.

For solvers that can use GPUs, when you uncheck Use Automatic Settings, the analysis configuration lets you specify the number of GPUs to be used by each machine, along with the existing settings for cores and RAM Limit. This makes sense, because GPUs, cores, and RAM are all resources that the user is managing, and GPUs and cores both contribute to HPC count. Each machinehas tasks and GPUs. The GPUs are distributed among the tasks, similarly to how cores are distributed. For example, if a machine has 3 tasks and 2 GPUs, then the first two tasks will get one GPU each, and the third task will get no GPU.

For design types like HFSS that support GPU for different types of solve (frequency, transient, SBR+), the GPU setting when Use Automatic settings is unchecked applies to all types. To use GPU for some solves and not for others, you can create multiple configurations.

Determining When a GPU or MIC is Being Used

You can determine if a GPU or MIC is being used for acceleration by viewing the Solutions dialog box, Profile tab. If a GPU is successfully locked for use by the solver process, the profile shows the GPU's CUDA device ID and its name.

If the GPU is not used, the Solution Profile indicates that, and the CPU cores are used.

When GPU is used:

- GPU must be enabled for HFSS/Maxwell.
- Both Windows and Linux are supported.
- Only complex symmetric matrices can be solved by CUDA.
- Matrix must be large enough. By default, its dimension should be larger than 2,000,000.
- The times using CUDA and CPU only are estimated. If GPU is faster, it will be used. If not, the solver falls back to multi-core CPU.
- HFSS-SBR+: GPU is used a) whenever computing far-field radiation and b) during coupling (S-parameter) computations whenever the receiving antenna is a FEBI antenna. In particular, GPU is used to accelerate the radiation of SBR ray footprints (equivalent currents) to scattered field observers. For example, if you have two antennas, where one is FEBI and the other uses a far-field representation, and the SBR+ setup includes a configured far-field radiation sphere, GPU is used for computing the installed far-field radiation pattern of both

antennas and the coupling from the far-field antenna (source) to the FEBI antenna (receiver). However, coupling from the FEBI antenna (source) to the far-field antenna (receiver) is computed using multi-core CPU. The speed benefit of GPU over multi-core CPU is greatest when computing the SBR+ contribution to far-field radiation, especially if there are many observation angles, as in a fully populated sphere. GPU loses efficiency when computing coupling to FEBI antennas located near the scattering geometry.

Estimate Time on GPU and CPU:

- The estimation is based on the structure of the matrix, not just the dimension.
- The generation and model of GPU are considered. The newer the model is, the faster the GPU estimation is.
- The clock rate and number of cores of CPU are consider. The higher the rate is, the more the cores are, the faster the CPU estimation is.
- Finally, the faster device (GPU or CPU) is selected based on estimation.

Why GPU is Not Used:

- The estimation is based on the whole matrix, not just the size. Having larger matrices doesn't mean GPU will be used.
- The bandwidth is not useful either. It is possible to have two matrices with exactly the same size and bandwidth, one favoring GPU and the other CPU.
- For HFSS SBR+ for monostatic RCS problems, the GPU is not used for simulation. The reason is that there is only one observation point for each plane wave excitation, so the GPU does not have as much potential to enhance the processing speed.
- Other HFSS SBR+ problems where GPU is not used include antenna problems where an antenna that is designated as an Rx or Tx/Rx type is a far-field source only. GPU is most effective in antenna problems involving many current sources, many far-field observation points and multiple frequencies.
- For HFSS SBR+ GPU will not kick in at all for antenna coupling problems with Rx antennas that are far-field type only.
- If GPU is of older models while CPU is not, GPU may not be used.
- If CPU has high clock rate, GPU may not be used.
- If CPU has many cores, GPU may not be used.
- If Use Automatic settings in Analysis Configuration is checked.
- Nvidia GPU drivers are downloaded from certain repository (e.g. RedHat) instead of directly from Nvidia website.

Multiple GPUs:

When there are multiple GPU cards in the same machine, the first solver process (hf3d, 3dedy) locks the first GPU card available (not locked or used by other processes, not used for display, etc.), the second process locks the second card available, and so on. After all cards are locked, solver uses CPU only.

Supported GPUs by Solver

GPU support includes Quadro and Tesla cards and GPU generations (Kepler (K), Maxwell (M), Pascal (P), Volta (V), Turing/RTX, Ampere(A)). GPU acceleration has been developed for Nvidia cards and is officially supported with the Tesla series. We highly recommend NVIDIA Tesla cards for the best performance when using several cards on one machine to solve either multiple variations (DSO) or excitations (HPC) in parallel which is referred to in this document collectively as distributed.

- HFSS Frequency-domain and Time-domain solvers support NVIDIA Data Center GPUs of the Ampere series and Tesla GPUs of the Volta, Pascal, and Kepler generations. NVIDIA Workstation RTX and Quadro GPUs for all generations are not supported except for Quadro GV100.
- HFSS SBR+ solver supports NVIDIA Data Center GPUs of the Ampere series and Tesla GPUs of the Volta, Pascal, Maxwell, and Kepler generations. NVIDIA Workstation GPUs of the RTX and Quadro families are supported by the HFSS SBR+ solver.
- Maxwell solvers support NVIDIA Data Center GPUs of the Ampere series and Tesla GPUs of the Volta, Pascal, and Kepler generations. NVIDIA Workstation RTX and Quadro GPUs for all generations are not supported except for Quadro GV100.
- Ansys EMIT supports NVIDIA Data Center GPUs of the Ampere series and Tesla GPUs of the Volta, Pascal, Maxwell, and Kepler generations. NVIDIA Workstation GPUs of the RTX and Quadro families are supported by EMIT.
- Icepak supports NVIDIA's CUDA-enabled Tesla and Quadro series workstation and server cards.

Driver requirements:

NVIDIA GPU minimum driver requirements are dictated by the NVIDIA CUDA version. The current Ansys EM uses CUDA 10.2 which requires a minimum driver version of 441.22 for Windows x86_64 and 440.33 for Linux x86_64.

We recommend downloading standard NVIDIA drivers for the user-specific cards rather than “DCH” drivers on Windows OS.

To contact Ansys technical support staff in your geographical area, please log on to the Ansys corporate website, ansys.com/support.

Note:

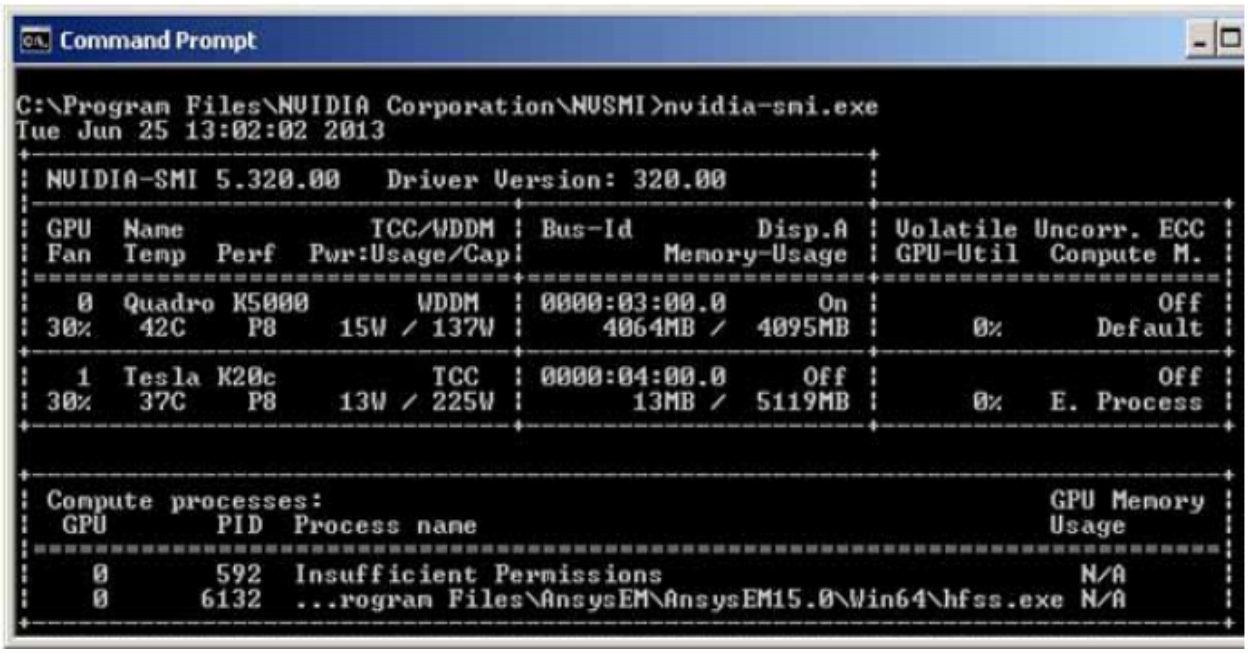
nVIDIA Tesla M2090, is a previous generation (code Fermi) GPU card and doesn't work for Workstation since it has no fan for active cooling but needs a server with GPU cooling solution (passive cooling) similar to the nVIDIA Tesla K80.

To get the best performance, the GPU used for running simulation jobs should not be attached to any display. Only GPU cards with CUDA Compute Compatibility 3.0 (Kepler) and above should be used. To improve the speedup of transient field visualization, you should install GPU cards on a system with PCI-E 3.0 slots. A mixture of interface cards with lower PCI-E versions may result in the data not being transferred from GPU to CPU at the highest speed.

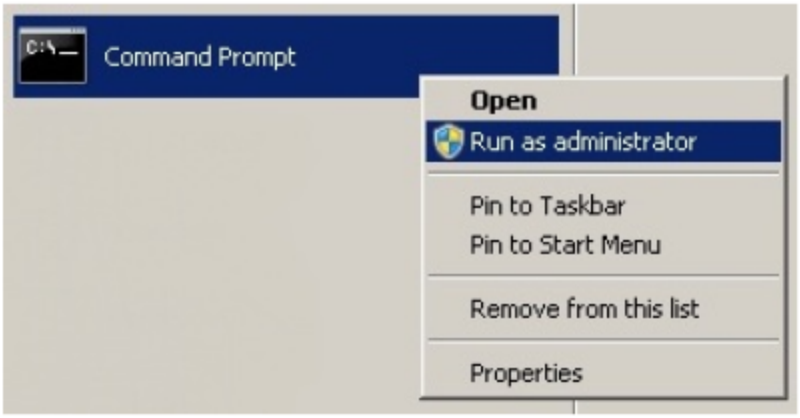
Setup for Windows

1. After you install GPU cards and Nvidia graphics drivers downloaded directly from the Nvidia website, you should be able to find the cards in Windows Display Manager.

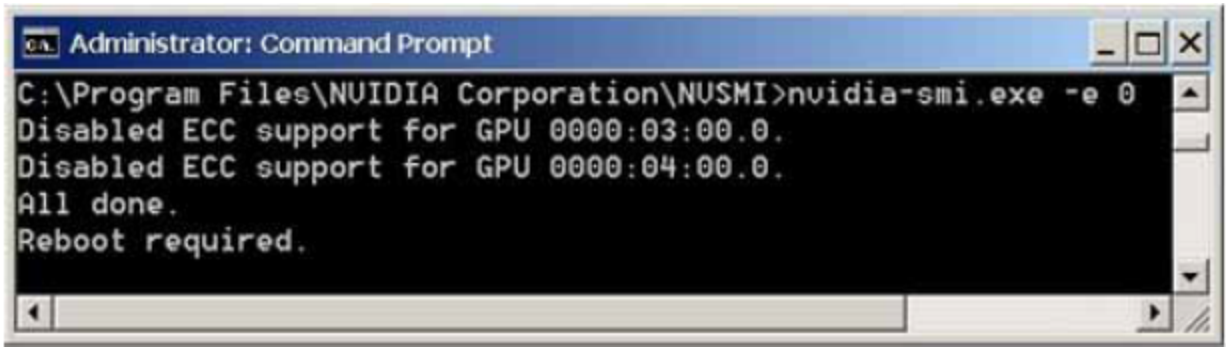
2. You should run nvidia-smi.exe at C:\Program Files\NVIDIA Corporation\NVSMI or C:\Windows\System32 to check if GPU cards are installed successfully. The executable nvidia-smi.exe should exist after the display driver is installed.



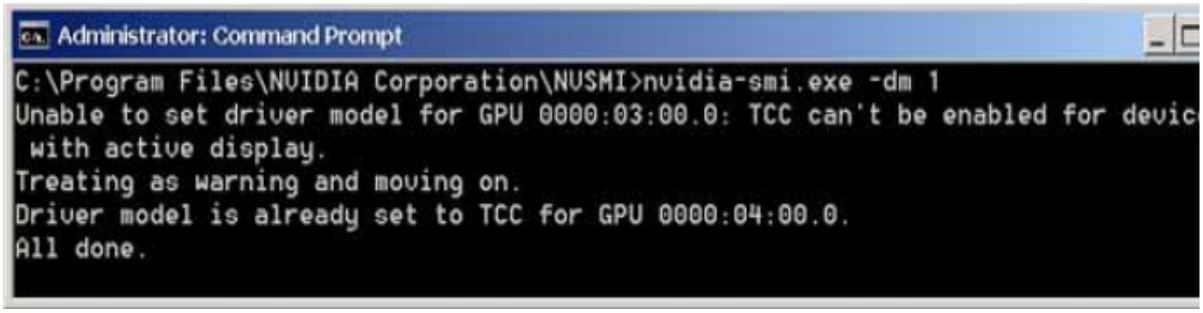
3. To further set up the configuration of GPU cards, open a command window as an administrator.



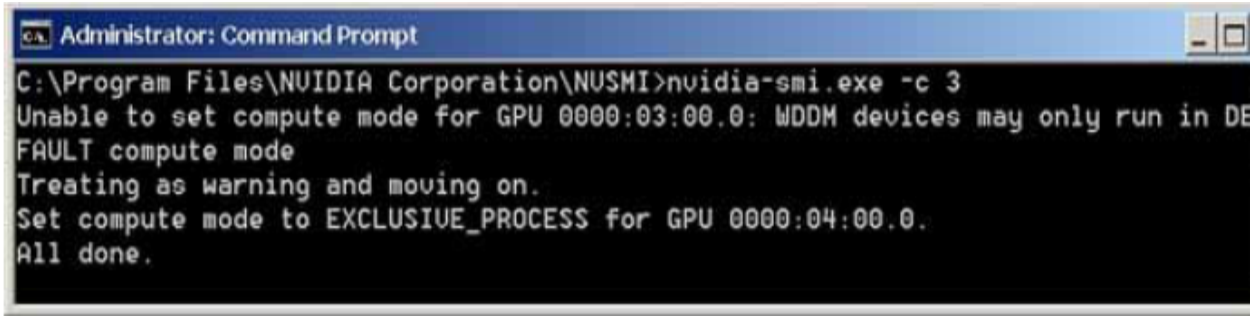
To improve the performance of GPU acceleration, it is recommended that you turn off the Error Correction Code (ECC) support by the -e 0 option of nvidia-smi. New ECC settings will be effective only after system reboot.



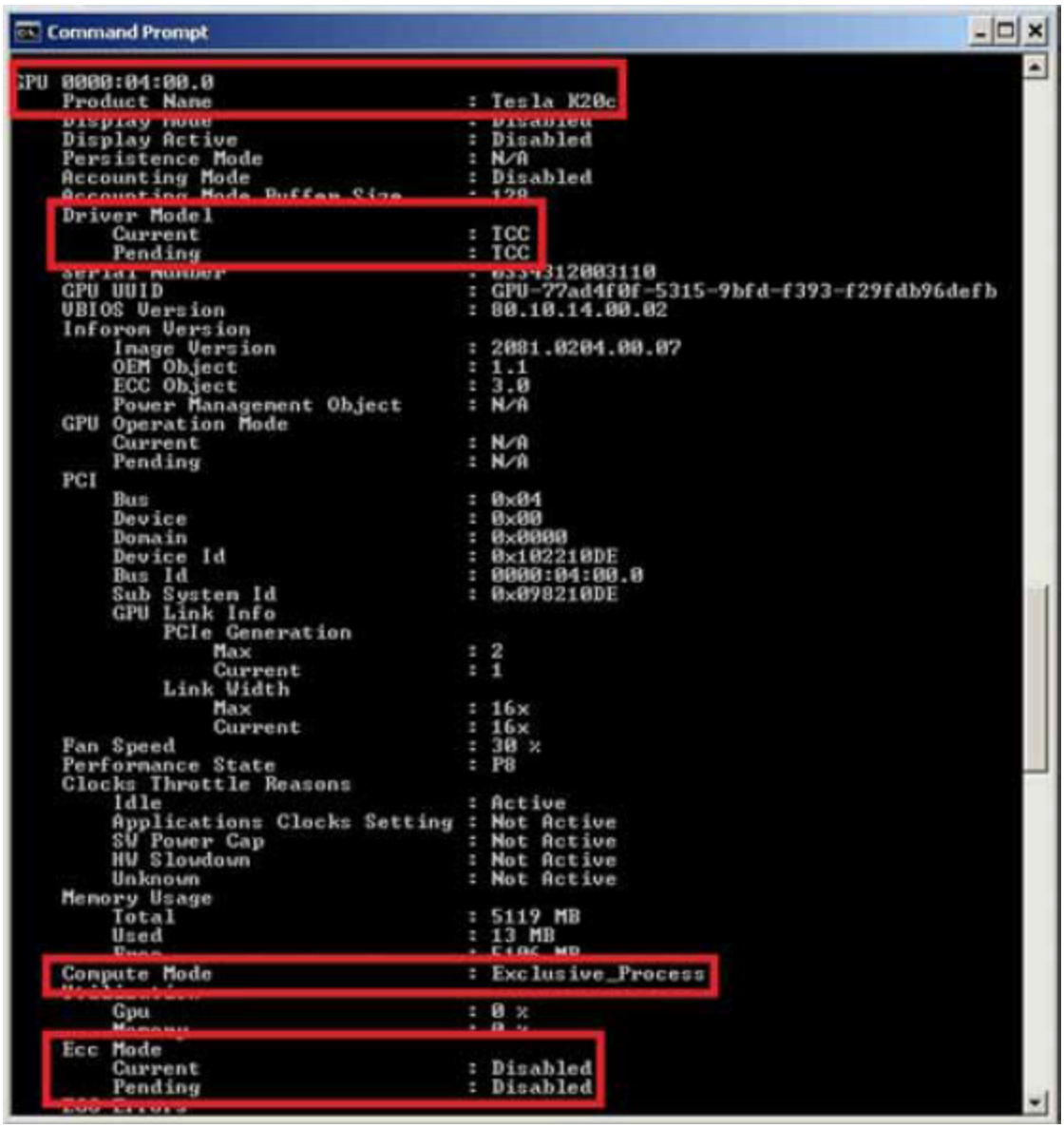
4. (Optional) For remote execution of GPU accelerated jobs (e.g., through Windows Remote Desktop Connection or RSM options in HFSS), it is necessary to turn on the Tesla Compute Cluster (TCC) mode by the -dm 1 option of nvidia-smi. New TCC settings will be effective only after system reboot. This step is unnecessary if you run HFSS Transient from a local machine. Please note that only Tesla cards support TCC. You cannot run GPU accelerated jobs on remote GeForce and Quadro cards.



5. (Optional) For users who want to run multiple GPU-accelerated jobs on one machine through distributed mode, it is required to install Nvidia Tesla cards with EXCLUSIVE_ PROCESS support. Using the -c 3 option of nvidia-smi, one can set GPUs in a system to be Exclusive_Process. HFSS Transient relies on this compute mode to assign each simulation job to a dedicated GPU card. Please note that GeForce and Quadro cards do not support EXCLUSIVE_PROCESS. Therefore, they should not be used for GPU-acceleration of HFSS Transient in distributed mode.



6. (Optional) Using -q option of nvidia-smi, one can check if the compute mode is set properly.



7. (Optional) For HFSS Transient to run in GPU-distributed, it is necessary to check if the dynamically linked library nvml.dll exists in its default directory C:\Program Files\NVIDIA Corporation\NVSMI. If not, its path should be added to the Windows environment variable Path.
8. The setup for a Windows GPU system is complete through Steps 1 to 7.

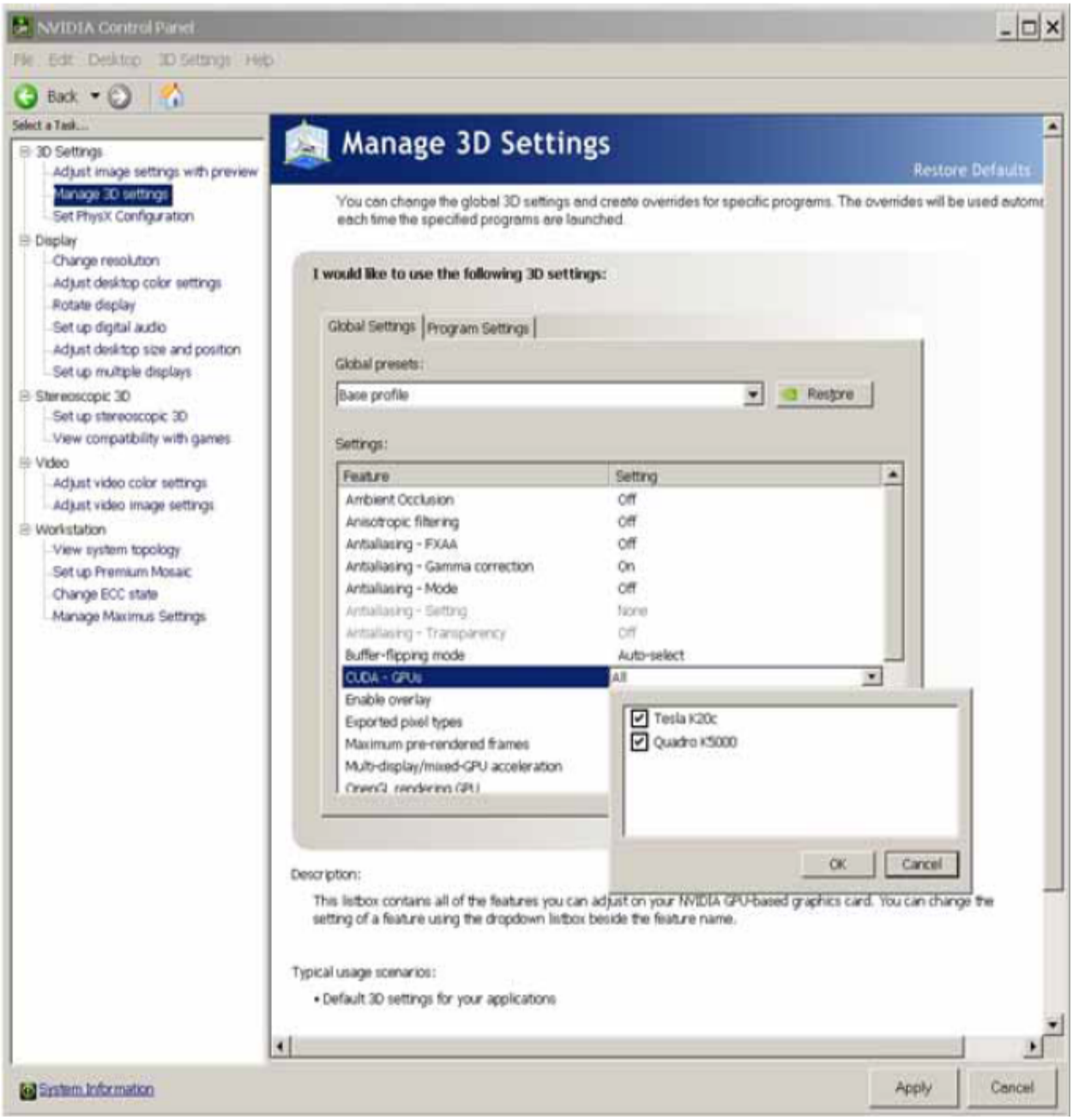
   If the computer system has both Quadro and Tesla cards, please read carefully the next section for setting up a Maximus platform.

Setup for an Nvidia Maximus Platform

When both Quadro (600, 2000, 4000, 5000, or 6000) and Tesla (C2075 and above) cards exist in a machine, the system is an Nvidia Maximus system and GPUs can be dedicated to graphics or compute tasks.

1. Once Nvidia graphics drivers are installed for a Maximus System, the icon of Nvidia Control Panel will be available on the Windows Taskbar. Click the icon to launch Nvidia Control Panel.

2. It is necessary to ensure there are GPUs visible for HFSS Transient. After the following setting to have both cards visible, HFSS Transient will be able to automatically grab Tesla K20c for GPU acceleration instead of Quadro K5000, because the latter is attached to a display and other processes in the system may use it for graphics acceleration. If only Tesla K20c is visible (Quadro K5000 unchecked), it will be used for GPU acceleration. If only Quadro K5000 is visible (Tesla K20c unchecked), it will still be used for GPU acceleration even with a display attached.



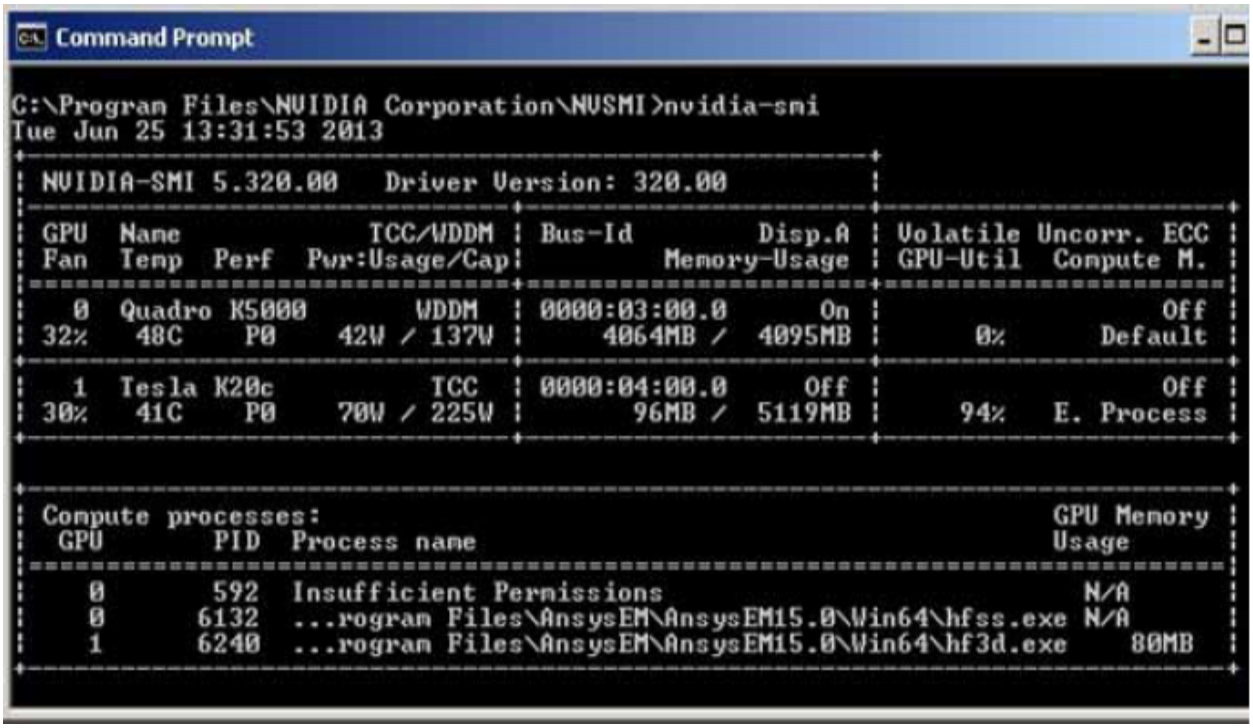
3. Furthermore, you must check to see if an environment variable CUDA_VISIBLE_ DEVICES exists in Windows. To be consistent with the above Maximus settings, the variable should be set to make both GPUs visible.

   CUDA_VISIBLE_DEVICES=0,1

   If the variable does not exist, all CUDA devices are visible in a system by default. HFSS Transient will only run on visible GPUs not attached to displays. However, if only one GPU

exists in a system and it is used for both compute and display, HFSS Transient will still grab it for acceleration.

4. The Maximus setup is complete through Steps 1 to 3. As an example, the following figure illustrates one hf3d process running on Tesla K20c and one hfss process using Quadro K5000 for graphics acceleration.



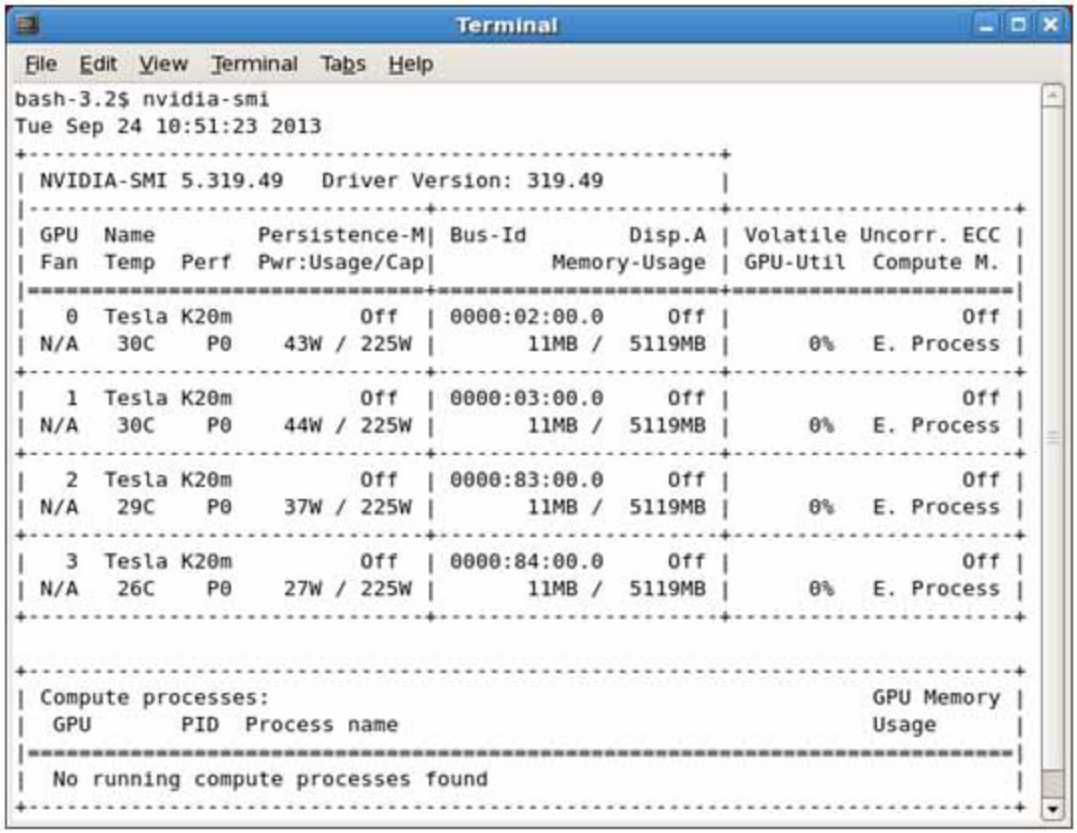
Setup for Linux

1. After you install Nvidia GPU cards and graphics drivers downloaded directly from the Nvidia website, you should be able to find the cards by the command.

   ```
   /sbin/lspci | grep -i nvidia
   ```

   You can also use the following command to check if GPU cards can be recognized by the system.

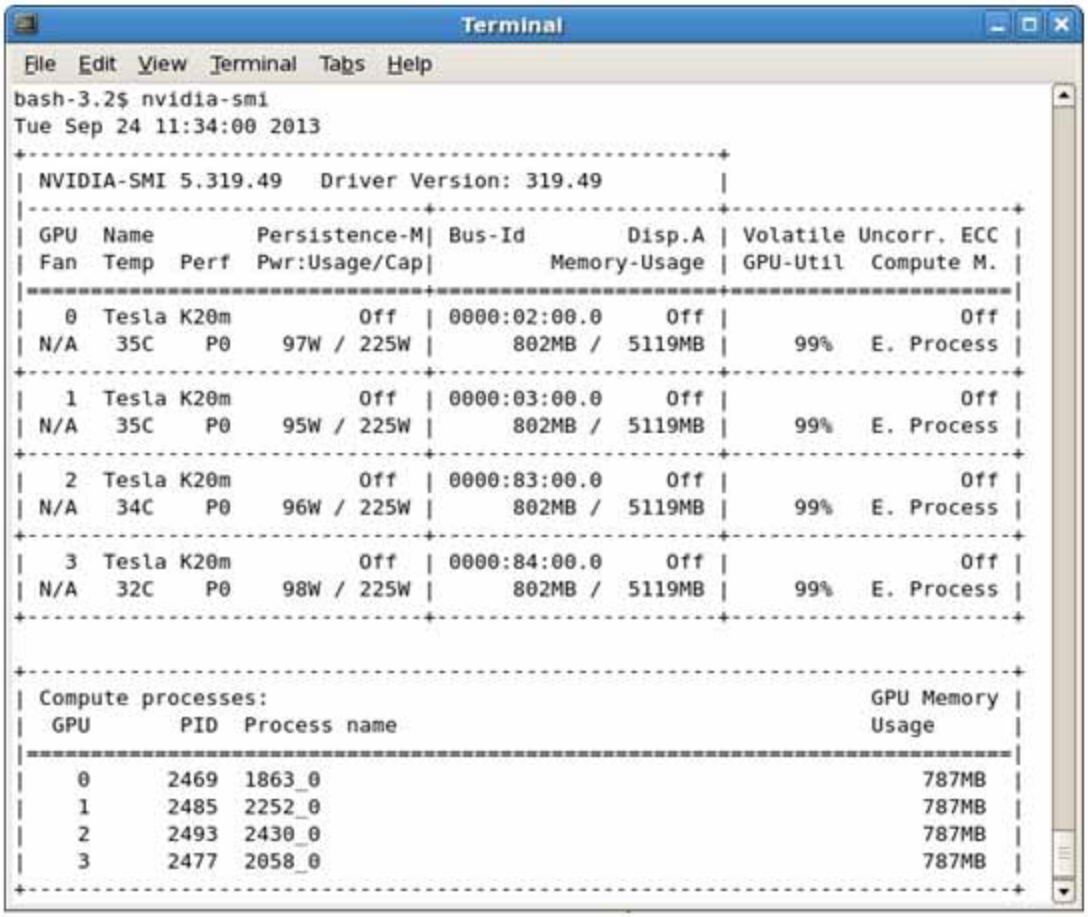
   ```
   /usr/bin/nvidia-smi
   ```



2. (Optional) The setting of GPUs to disable ECC (for performance), enable TCC (for remote execution), and enable Exclusive_Process (for GPU-distributed) are similar to Windows. You need the administrative right to make such changes.

   ```
   sudo nvidia-smi -e 0
   sudo nvidia-smi -dm 1
   sudo nvidia-smi -c 3
   ```

3. (Optional) On Linux Maximus platforms, the environment variable CUDA_VISIBLE_DEVICES can be set in the shell file ~/.bash_profile to toggle the visibility of GPU cards as CUDA devices. If the variable does not exist, all CUDA devices are visible in a system by default.
4. (Optional) In order to let HFSS Transient access the dynamically linked library libnvidia-ml.so in GPU-distributed, one must check if the library exists in its default directory /usr/lib64. If not, its path should be added to the Linux environment variable PATH through a shell startup file.
5. The setup for a Linux GPU system is complete through Steps 1 to 4. As an example, the following figure illustrates four distributed hf3d processes running on four Tesla K20m cards.



License Options

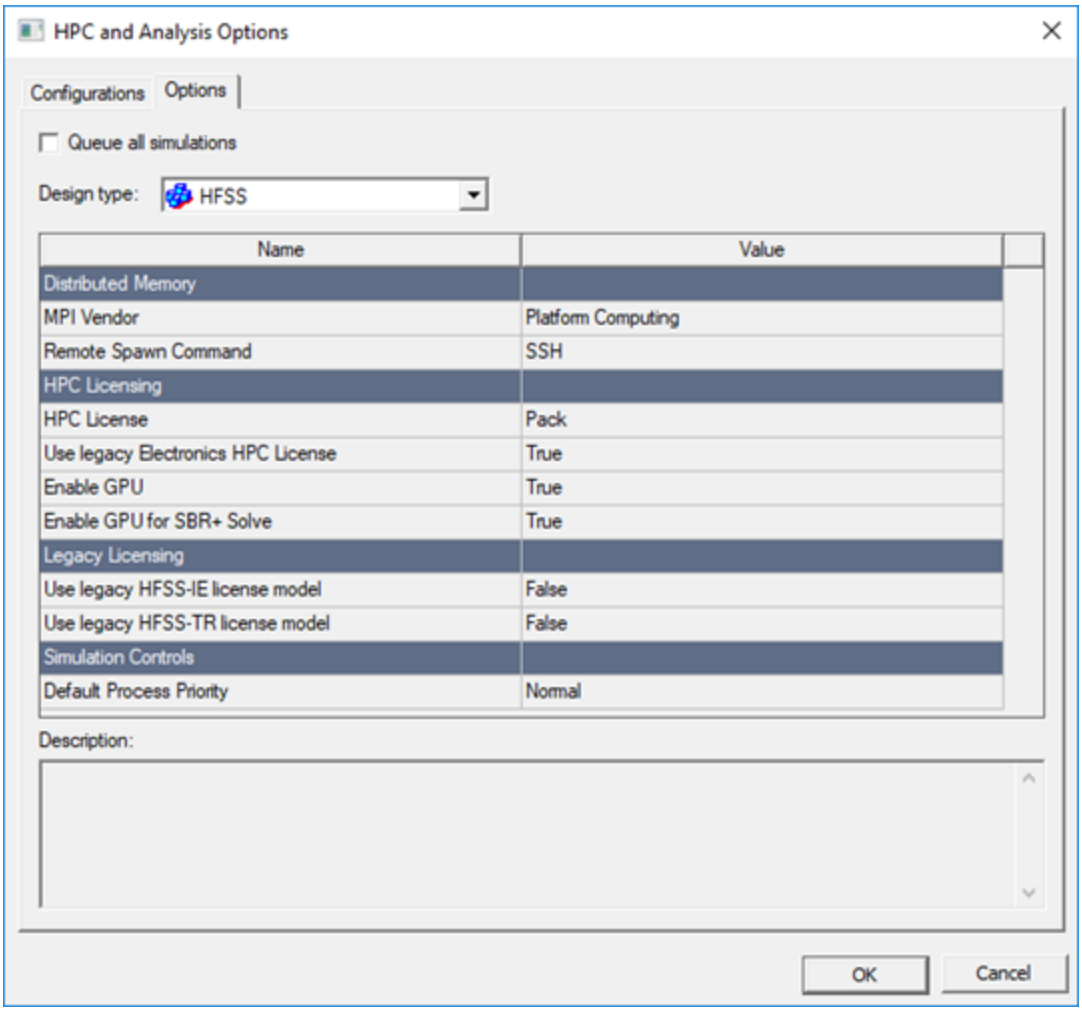
Users with HPC packs can use GPU acceleration for HFSS Transient. The maximum number of GPUs to be used on a standalone machine or for all machines in a cluster is limited by the number of HPC packs.

- 1 HPC pack = 1 GPU
- 2 HPC packs = 4 GPUs
- 3 HPC packs = 16 GPUs
- 4 HPC packs = 64 GPUs

If the number of simulation jobs exceeds the number of GPUs in a system, the excessive jobs will fall back to CPUs and will be accelerated by up to 8 CPU cores for each job.

Enable/Disable GPU Acceleration from the HFSS User Interface

To turn on or off GPU acceleration, click **Tools>Options>HPC and Analysis Options** and select the **Options** tab. Click on the value of Enable GPU and toggle it to either True or False.



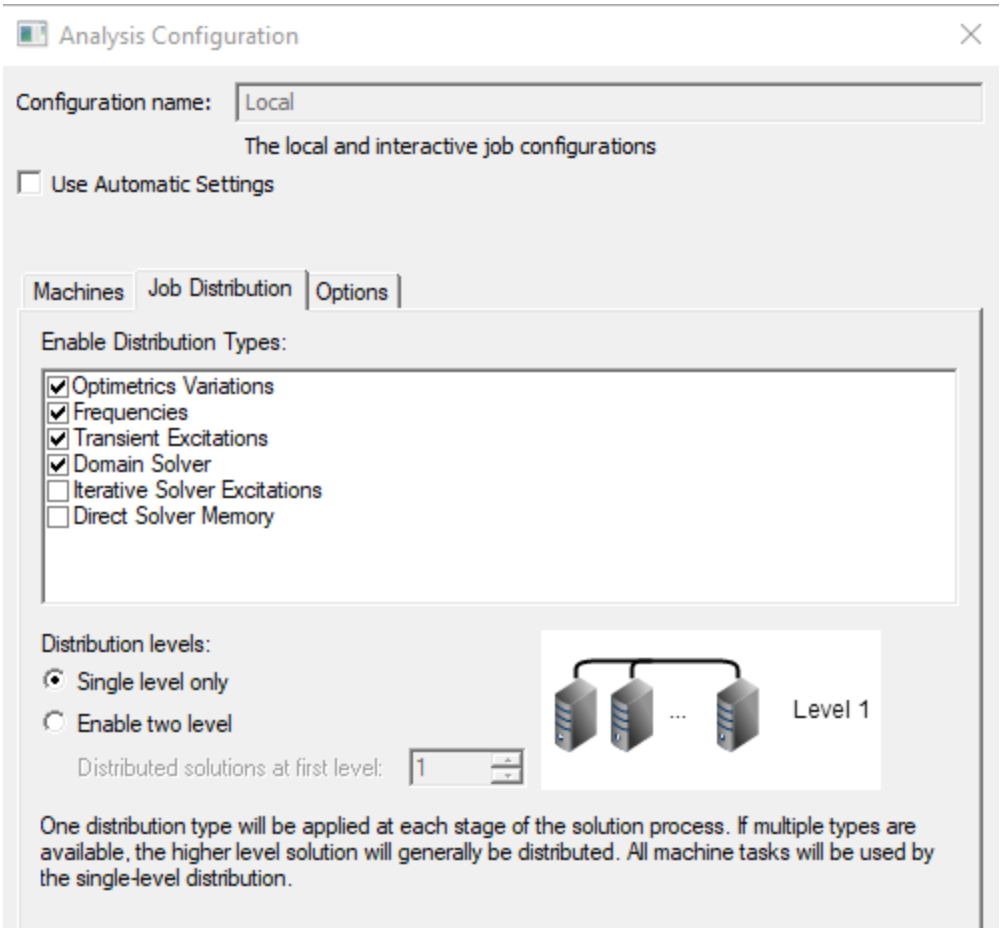
Enabling GPU-Distributed

There are two steps for setting up GPU-distributed.

1. Go to **Tools>Options>HPC and Analysis Options** and click **Edit** in the dialog box, or select the **Simulation** tab of the ribbon, and click **Analysis Config** to edit Analysis Configurations.

   Uncheck **Use Automatic Settings**, select the **Job DIstribution** tab, and select Transient Excitations. GPU-DSO in HFSS Transient can be used either for parametric sweep

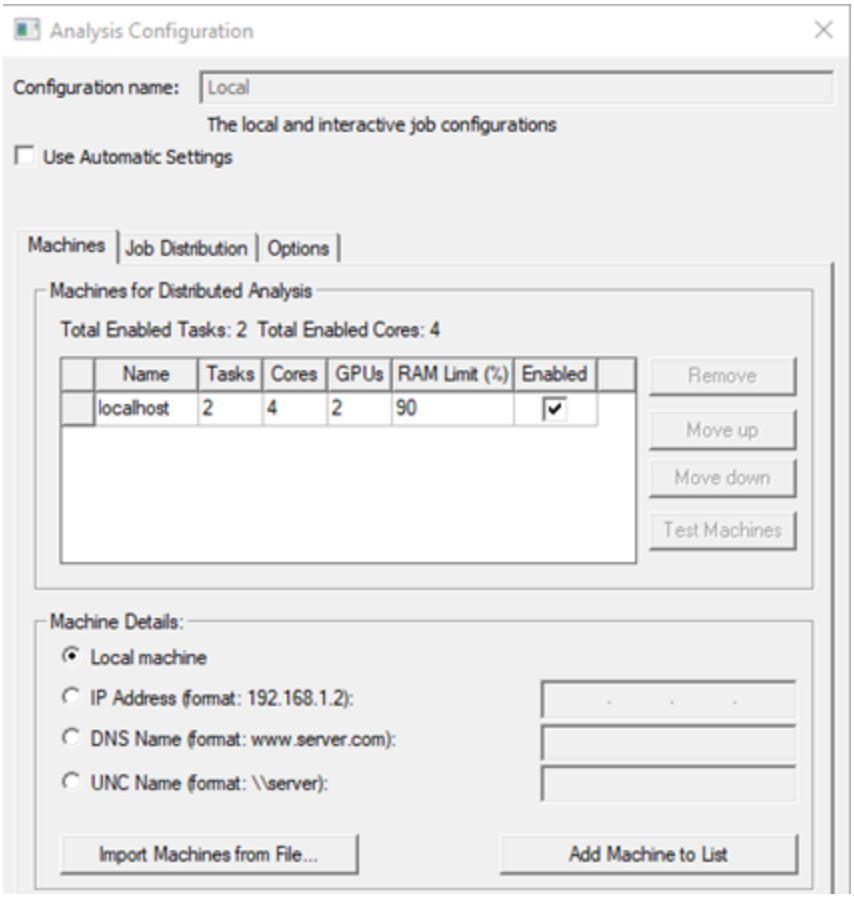
(Variations) or Transient Network analysis of multiport networks (Transient Excitations).



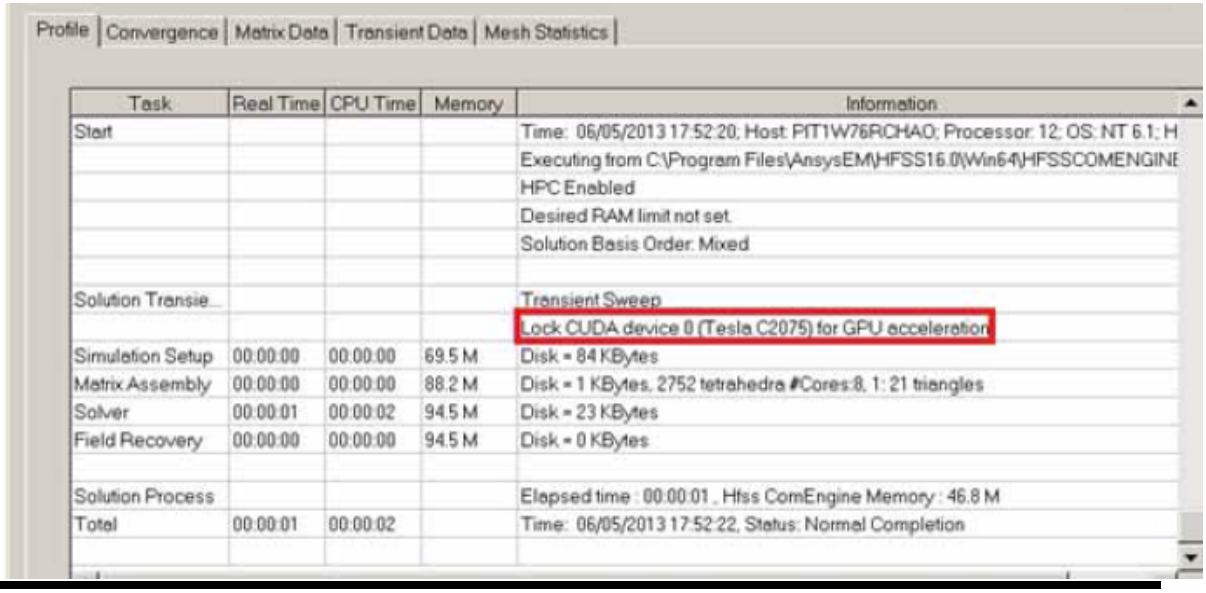
2. Set up the Machines for GPU-Distributed.

   In the following figure two simulation jobs are allowed to run on the local host, or up to two GPUs can be used for acceleration depending on the availability of licenses and GPU

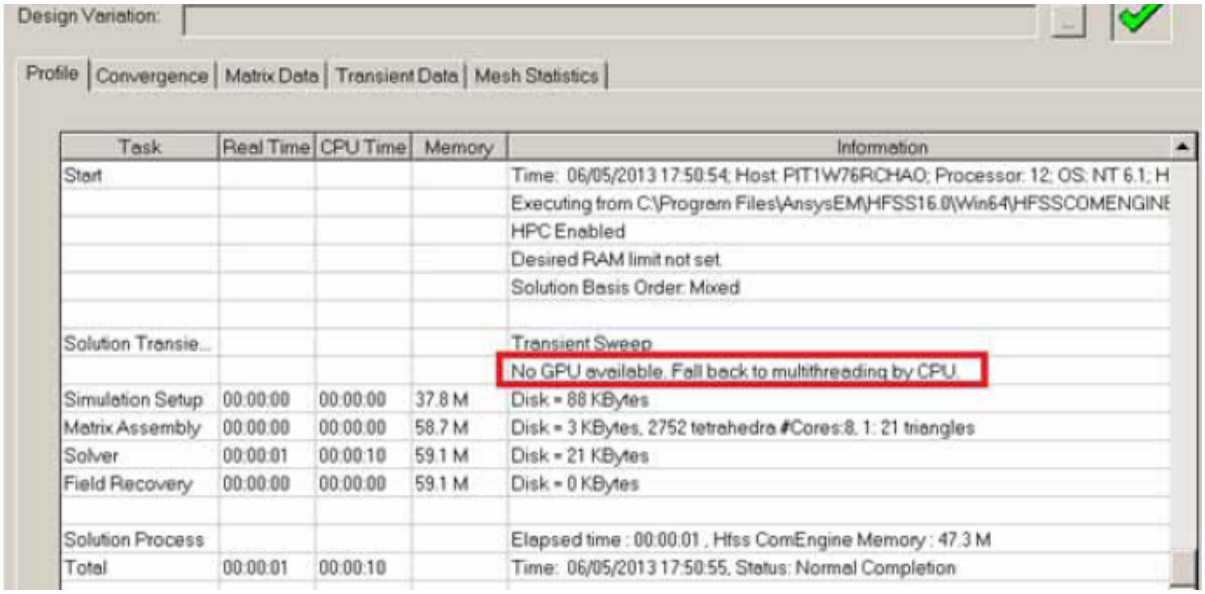
cards.



Whether GPU is used for acceleration can be checked by viewing the **Solutions** dialog box, **Profile** tab. If a GPU is successfully locked for the use by an hf3d process, the profile will show the GPU's CUDA device ID and its name.



Otherwise, the profile will indicate the fallback to CPUs. More information about why GPUs are not available for acceleration can be found in the HFSS Transient log file.



Enable/Disable GPU Acceleration from the Command Line

GPU acceleration of HFSS Transient can be toggled by the -batchoptions command line argument:

```
EnableGPU=[0/1].
```

For example, the following command turns on GPU acceleration.

```
ansysedt.exe -batchsolve -batchoptions "'EnableGPU'=1"
projectname.aedt
```

License Options:

Users with HPC packs can use MIC acceleration for HFSS. The maximum number of MICs to be used on a standalone machine or for all machines in a cluster limited by the number of HPC packs.

- 1 HPC pack = 1 MIC
- 2 HPC packs = 4 MICs
- 3 HPC packs = 16 MICs
- 4 HPC packs = 64 MICs

If the number of simulation jobs exceeds the number of MICs in a system, the excessive jobs will fall back to CPUs and will be accelerated by up to 8 CPU cores for each job.

Enable or Disable GPU or MIC Acceleration from the User Interface:

See Setting HPC and Analysis Option in the online help.

3 - Cluster Specific Considerations

Ansys Cloud Burst Computing

Job Submission Methods

- Manual: Number of Tasks and Cores: allows specification of Total number of tasks, Cores per distributed task, and RAM Limit %. Also allows manual specification of allowed Job distribution types and two-level distribution.
- Automatic: Number of Cores and (Optional) RAM: allows specification of Total number of cores, RAM Limit %, and Num variations to distribute.

Configuration Guidelines

Compute resource selection for Ansys Cloud Burst Computing resource selection parameters. (The default region is "Eastern US".)

The Resource selection parameters text field gives a summary of the compute resources by showing the region and the compute pool size (i.e. number of nodes and cores plus GB of RAM).

HPC License Count

Ansys Cloud Burst adopts a straightforward approach to license count. For each machine, the HPC count will be max(cores). The total HPC count will then be the sum of the HPC counts for all machines. If using DSO licensing, the total HPC count is divided by the number of distributed variations.

HFSS Defaults

HFSS uses Auto as its default configuration.

Windows HPC

Job Submission Methods

- Manual: Number of Tasks and Cores: OK
- Manual: Individual Nodes: OK
- Automatic: Number of Cores and (Optional) RAM: OK, but RAM not supported
- Automatic: Number of Nodes and Cores: OK
- Automatic: Individual Nodes: OK

Background

Under Windows HPC, there are limited ways to specify job parameters to ensure that cores allocated to the job are not distributed across many different hosts. To make use of multithreading, the cores for a task must be allocated on the same host. For distribution types other than frequencies and optimetrics variations, distributing the tasks to many different hosts may adversely impact performance due to the overhead of cross-host communication between tasks.

One way to ensure that all cores are allocated in groups of N cores on each host, is to specify that only hosts with exactly N cores may be allocated to the job, to specify that the job be allocated to hosts exclusively, and to specify that the total number of cores is an integral multiple of N. This approach is used in several cases, when the job is submitted using the Ansys Electromagnetics job submission GUI. All N cores on each host may be allocated to a single task, or the N cores may be divided between several tasks on each host.

Configuration Guidelines

For jobs submitted to a Windows HPC cluster, the job unit type should always be “Core.” A job may need to create tasks to start processes on some hosts allocated to the job.

If the job unit type is “Socket” or “Node,” then these tasks may be queued, resulting in a failure of the job.

To reduce the contention for resources on a single host from multiple jobs, Ansys Electromagnetics jobs should always be allocated to hosts exclusively. This allows the greatest opportunity to use multithreading. It also allows cross-host MPI communication to be minimized.

The Windows HPC cluster affinity setting should be set to **Non-Exclusive jobs** or **No jobs**. This is needed because some tasks only use significant CPU resources for a small fraction of the job. Setting the cluster affinity to “All jobs” would tie up any cores bound to these tasks for the entire duration of the job, preventing them from being used by other tasks.

LSF

Job Submission Methods

- Manual: Number of Tasks and Cores: OK
- Manual: Individual Nodes: Not supported
- Automatic: Number of Cores and (Optional) RAM: OK
- Automatic: Number of Nodes and Cores: OK
- Automatic: Individual Nodes: Not supported

Background

Ansys Electromagnetics products do not support resizable jobs. As a result, the LSF features for resizable jobs are not applicable to Ansys Electromagnetics.

Ansys Electromagnetics products to not support multi-phase resource specification. As a result, jobs submitted using the Ansys Electromagnetics job submission GUI will not include duration or decay parameters in the rusage section of the resource requirement string.

For LSF, the span string is used to specify how compute resources are distributed over the nodes allocated to the job. Jobs submitted using the Ansys Electromagnetics job submission GUI may include a hosts specification or a ptile specification. Only an integral ptile specification is used by such jobs. The special value'!' is not used by the job submission GUI, nor are multiple ptile values which vary by host type or host model. Here is a brief description of the meaning of the span strings that may be specified for a job by the Ansys Electromagnetics job submission GUI:

- span[hosts=1]: All processor cores for the job must be on the same node.
- span[hosts=-1]: There are no restrictions on how cores allocated to the job are distributed over the nodes allocated to the job.
- span[ptile=n]: Exactly n cores will be allocated to the job on each node allocated to the job, where n is an integer.

For LSF, exclusive jobs may only run in a queue configured as an exclusive queue. The Ansys Electromagnetics job submission GUI supports exclusive jobs if an exclusive queue is available. The Ansys Electromagnetics job submission GUI does not support compute unit exclusive jobs.

Configuration Guidelines

To allow users to submit exclusive jobs, they must have access to queues which are configured with the EXCLUSIVE property set to Y.

Using the Ansys Electromagnetics job submission GUI, a user may submit jobs where the user specifies tasks automatically and specifies the **Number of Cores and (Optional) RAM**.

However, the RAM per core may only be specified if the LSF cluster is configured with parameter RESOURCE_RESERVE_PER_SLOT set.

Grid Engine (SGE, UGE, OGE)

This section covers the various versions of Grid Engine. The recent versions include Sun Grid Engine (SGE), which was renamed to Oracle Grid Engine (OGE), and later renamed to Univa Grid Engine (UGE), which includes some features not available in the earlier versions. A commercially supported open source version based on a previous (open source) version of Sun Grid Engine is known as Open Grid Scheduler/Grid Engine. The comments in this section are generally applicable to all of these versions, although there are some differences between them. This is especially true for Univa Grid Engine, which has some features not available on the other versions.

Job Submission Methods

- Manual: Number of Tasks and Cores: OK
- Manual: Individual Nodes: Not supported
- Automatic: Number of Cores and (Optional) RAM: OK
- Automatic: Number of Nodes and Cores: OK
- Automatic: Individual Nodes: Not supported

Background

For these schedulers, the main options for controlling how resources are allocated to a job are the parallel environment and the queue or queues specified for the job. The parallel environment mainly controls how resources allocated to the job (cores or slots) are distributed to the hosts allocated to the job. The queues control which hosts may be used by the job.

For a serial job (a job using a single core on a single host), no parallel environment needs to be specified by the job. For a job to use multiple cores on a single host or to use multiple hosts, a parallel environment must be specified for the job.

For a parallel environment (PE) the allocation_rule parameter indicates how the slots allocated to the job will be distributed to different hosts. The allocation rule may be an integer value, or one of the following special values:

- $fill_up: on each host, all slots will be allocated to the job, except that the “last” host may have some slots left if they are not needed to meet the number of slots required
- $round_robin: slots will be distributed across all applicable hosts in a round robin fashion
- $pe_slots: all slots will be allocated on a single host

An integer value, n, indicates that exactly n slots will be allocated to the job one each host allocated to the job. In this case, the total number of slots allocated to the job should be an integral multiple of n. It it is not, the job may remain queued without ever starting.

There is also a control_slaves parameter associated with each parallel environment. For parallel Ansys Electromagnetics jobs running on an SGE cluster, only parallel environments with the control_slaves parameter set to TRUE may be used by the job. This parameter must be TRUE so that parallel processes needed by the job may be started by the job.

There are additional restrictions on the parallel environments that may be used by a job. When the user submits a job using the Ansys Electromagnetics job submission GUI, the user may specify either a parallel environment, a queue, both, or neither. If both are specified, and the queue is valid for the parallel environment, then the job submission command will include both the specified queue and the specified parallel environment. If only a queue is specified, then the job submission command will also include the parallel environment associated with that queue that best meets the other requirements, if any. If only a parallel environment is specified, then the job submission command will include the specified parallel environment. If neither is specified, then the job submission command will include the parallel environment that best meets the other requirements.

The parallel environments which may be used by the job depend on the job submission method, and whether the user specified a parallel environment. In all cases, there must be sufficient hosts associated with the specified parallel environment (and queue, if specified) to meet the requested cores, nodes and cores, or tasks and cores. If these requirements cannot be met, then the job will not be submitted.

If the user specifies tasks manually and specifies the **Number of Tasks and Cores**, then the allowed parallel environments depend on the number of cores per task specified by the user. If the number of cores per task specified by the user is one (and it is not increased due to a limit on the number of tasks per node), then the user may specify any parallel environment, because there will be no multithreading in this case, because any distribution of cores over nodes is allowed. If the user does not specify a parallel environment, then a parallel environment with allocation rule $fill_up or $round_robin is selected, if available for the job. If no such parallel environment is available, then the job is not submitted. If the number of cores per task specified by the user is greater than one (or it is increased to a value greater than one due to a limit on the number of tasks per node), then only parallel environments with an integral allocation rule are allowed. The allocation rule must allow enough slots for at least one task to run on each host. If the allocation rule allows enough slots on each node, then more than one task may be allocated to run on each host.

If the user specifies tasks automatically and specifies the **Number of Cores and (Optional) RAM**, then all parallel environments are considered if the user does not specify a parallel environment. The parallel environments are searched in the following order of allocation rule, and the first parallel environment that meets the requirements is selected: 1) $pe_slots, 2) $fill_up, 3) $round_robin, and 4) an integral allocation rule. Note that, if the number of cores is exactly

one, then no parallel environment is selected. If the user does specify a parallel environment, then the user specified parallel environment is selected. The number of slots requested is equal to the number of cores specified by the user, except in the case of a parallel environment with an integral allocation rule. In the case of an integral allocation rule, the number of slots is the smallest multiple of the allocation rule that is greater or equal to the requested number of cores.

If the user specifies tasks automatically and specifies the **Number of Nodes and Cores**, then only parallel environments with an integral allocation rule are valid for the job if the user does not specify a parallel environment. If the user does specify a parallel environment, then the results depend on the number of nodes and cores specified for the job. For a single core job, a parallel environment with any allocation rule may be specified; if a parallel environment with an integral allocation rule is specified, then the number of slots requested is increased to equal the allocation rule. For a single host job with at least two cores requested, a parallel environment with any allocation rule may be specified, except for $round_robin; if a parallel environment with an integral allocation rule is specified, then the the allocation rule must be equal to or greater than the requested number of cores. For a job with at least two nodes requested, only parallel environments with an integral allocation rule are allowed, and the allocation rule must be equal to or greater than the ratio of cores to nodes.

Configuration Guidelines

Parallel Environments

To effectively run parallel Ansys Electromagnetics jobs on an SGE cluster, appropriate parallel environments need to be available on the cluster. All parallel environments for Ansys Electromagnetics jobs must be configured with the control_slaves parameter set to TRUE.

If cluster users will submit jobs for which the tasks are specified manually, parallel environments with integral allocation rules must be configured that may be used by these jobs. At a minimum, there should be one parallel environment with an integral allocation rule equal to the maximum number of cores per node to be used for these jobs. For example, if all nodes to be used for these jobs have N cores per node, then an allocation rule of N will allow the user to specify up to N cores per task. If the user specifies less than N cores per task, then up to N/M tasks may run on each node, where M is the number of cores per task specified by the user. Configuring multiple parallel environments with an different integral allocation rules may allow a better match of resources needed by the job to the resources allocated to the job. For example, if all nodes have 8 cores, configuring parallel environments with allocation rules from 1 to 8 will allow a good resource match. For example, if the user specifies 3 tasks with 5 cores per task,and there is only a parallel environment with allocation rule of 8 available, then this parallel environment and 24 slots will be selected. If there is a parallel environment with allocation rule of 5 available, then this parallel environment and 15 slots may be selected, allowing the remaining 3 slots on each node to be allocated to other jobs.

If cluster users will submit jobs for which the tasks are specified automatically, and the user will specify the “Number of Cores and (Optional) RAM” for the job, then any available parallel

environment may be used for the job. If one prefers to isolate the nodes allocated to one job from other jobs, a parallel environment with allocation rule $fill_up may be used to accomplish this. Conversely, if one desires to distribute the load of a job over all of the cluster nodes available to the job, a parallel environment with allocation rule $round_robin may be used to accomplish this.

If cluster users will submit jobs for which the tasks are specified automatically, and the user will specify the “Number of Nodes and Cores” for the job, then parallel environments should be configured as for the case in which cluster users will submit jobs for which the tasks are specified manually, as described above.

Submitting Exclusive Jobs

In many cases, clusters are used to run "large" Ansys Electromagnetics Suite batch jobs. That is, these are jobs that may require a large quantity of resources, such as processors, memory, disk space, or run time. One way to ensure that the resources needed by the batch job are available to the job is to run the job in an "exclusive" mode. That is, any host running the job is not available for use by any other jobs. There is no SGE built in mechanism for specifying that a job is "exclusive". SGE is extensible, and it is not difficult to configure the cluster to allow exclusive jobs. The steps below show one way to do this. This example requires SGE 6.2u3 or later. Note that specifying a job as "exclusive" may delay the start of the job if there are not enough hosts available to run the job exclusively.

1. Use the command qconf -mc to add a new complex to the table of complexes. Recommended attributes are:
   - name : exclusive
   - shortcut : excl
   - type : BOOL
   - relop : EXCL
   - requestable : YES
   - consumable : YES
   - default : 0
   - urgency : 0
2. Set the value of "exclusive" to TRUE for each execution host using the command qconf -me hostname, where hostname is the name of the host. The values of all host configuration parameters may be displayed using the command qconf -se hostname. The complex_values line should look similar to:

   ```
   complex_values exclusive=TRUE
   ```

   but other values may also be included.
3. When submitting a job, the job will be "exclusive" if the value "excl" is included in the resource list specified by the qsub -l option. If the resource list does not include "excl,"

then the job will not be exclusive, and other jobs may run on the same host or hosts as this job.

4. Example qsub command line for exclusive serial job:

   ```
   qsub -b y -l excl /opt/ansys_inc/v252/AnsysEM/ansysedt -ng
   -BatchSolve -machinelist num=1 ~/projects/OptimTee.aedt
   ```

   Although serial jobs use only one slot, no other jobs will run on the host where this job is running, even if additional slots are present.

5. Example qsub command line for exclusive parallel job using eight engines, each using a single thread of execution:

   ```
   qsub -b y -l excl -pe pe1 8 /opt/ansys_inc/v252/AnsysEM/ansysedt
   -ng -BatchSolve -Distributed -machinelist num=8
   ~/projects/OptimTee.aedt
   ```

   None of the hosts used for this job will be allowed to run other jobs while this job is running.

Consumable Memory Limits

SGE contains several built-in complexes related to memory, including mem_total, for example, but none of these are "consumable". If a job is submitted with resource list including one of these non-consumable memory complexes (such as mem_total), then the job will run on a host or hosts only if sufficient memory is available. If a second job is submitted, the memory request for the second job is compared to the original total when determining if the job may run on a host. This may result in both jobs running out of memory. For example, if host A has mem_total=16G of memory, and two jobs are submitting with option "-l mt=16G", then both jobs could run on host A, if sufficient slots are available on host A.

SGE allows complexes to be "consumable" to avoid this type of problem. If a complex is consumable and a job requests x amount of the complex in the -l resource list, then the available amount of the resource is decreased by x for subsequent jobs. For the same example as above, if the mem_total complex was consumable, then the first job would run on host A. This would decrease the available mem_total from 16G to 16G-16G = 0. The second job could not run on host A because there is no memory available for this job.

We do not recommend changing the behavior of the built-in complexes (such as mem_total) because other scripts may expect normal behavior of the built-in complexes.

Note	Recent versions of UGE (Univa Grid Engine) come with "m_mem_free" and "mem_ free" complexes already configured, and if so then there is no more configuration required. You can just use mem_free when per-host memory request is desired, and m_mem_free when per-core (per-slot really) memory request is desired. SGE may already have the "mem_free" complex configured, which can be used for per-host memory request..

The following shows how to configure the mem_free consumable resource.

Recommended attributes are:

- name : mem_free
- shortcut : mf
- type : MEMORY
- relop : <=
- requestable : YES
- consumable : YES
- default : 0
- urgency : 0

Note	R16 has the capability in auto cores and RAM to automatically select the memory complex (if one is available) and create this command line option for the user, if you check the "Use RAM constraint" and enter a non-zero amount of GB to use per core.

qrsh command

Ansys EM parallel batch jobs use the SGE qrsh command to launch engine processes on remote hosts. If the qrsh command is not working correctly, then the parallel job is unable to launch engine processes on remote hosts. If this problem occurs, the batch log for the job typically includes one or more error messages indicating that a COM engine was unable to be started on a remote host. If this occurs, the user or cluster administrator should verify that the SGE qrsh command is working correctly, and correct the problem if the SGE qrsh command is not working correctly.

The qrsh command may be tested by running a simple command on a specified host, such as qrsh -l hostname=host1 hostname or qrsh -l hostname=host1 ls /tmp, where host1 is the remote host name. The first test should simply echo back the hostname of the remote machine. The second test should list the contents of the /tmp directory on the remote machine.

The failures of the SGE qrsh command are associated with the following global sge configuration parameters, listed below with values that may cause the failures:

- qrsh_command /usr/bin/ssh -t
- rsh_command /usr/bin/ssh -t
- rlogin_command /usr/bin/ssh -t

If these parameter settings are removed or changed to builtin, then the SGE built-in mechanisms are used for qrsh, rsh, and rlogin. No problems with the built-in versions have been reported. The SGE qconf -sconf global command may be used to view these parameter settings. The SGE qconf -mconf global command may be used to modify these parameter settings.

PBSPro

Job Submission Methods

- Manual: Number of Tasks and Cores: OK
- Manual: Individual Nodes: OK
- Automatic: Number of Cores and (Optional) RAM: OK
- Automatic: Number of Nodes and Cores: OK
- Automatic: Individual Nodes: OK

Background

When submitting a job using the Ansys Electromagnetics job submission GUI, the user may select a queue from the list of queues at the default server, or may type in a queue@server specification, or specify neither, but both may not be specified together. If neither is specified, then the job will be submitted to the default queue at the default server. If the queue is selected, then the job will be submitted to the selected queue at the default server. If the @server is omitted from the user specified queue@server string, then the job will be submitted to the specified queue at the default server. If the queue is omitted from the user specified queue@server string, then the job will be submitted to the default queue at the specified server. If both the queue and @server are included in the user specified queue@server string, then the job will be submitted to the specified queue at the specified server. If the job is submitted to any queue at the default server, then the job resources are checked against the available resources and the job is only submitted if there are sufficient resources to eventually run the job. If the job is submitted to a server which is not the default server, then this resource check is omitted.

Any node attributes specified by the user will be included for each chunk or node specified by a select resource specification.

For jobs submitted to a PBSPro cluster using the Ansys Electromagnetics job submission GUI, the user specified exclusive option affects the place resource specification of the qsub submission command. For non-exclusive jobs, the submission command place resource option is “-l place=free”. For exclusive jobs, the submission command place resource option is “-l place=excl”.

Distribution of Nodes, Tasks and Cores

- If the user specifies tasks manually and specifies the **Number of Tasks and Cores**, then the job submission parameters depend on the number of cores per task specified by the user. If the number of cores per task specified by the user is one (and it is not increased due to a limit on the number of tasks per node), then this is the no multiprocessing case, so the single cores tasks may be distributed to nodes with no additional restrictions. The job is submitted using the “-l select=x:ncpus=1:mpiprocs=1” qsub option. The value of x is

the number of tasks specified by the user. If the number of cores per task specified by the user is greater than one (or it is increased to a value greater than one due to a limit on the number of tasks per node), then this is the multiprocessing case. For the multiprocessing case, the available nodes are considered in groups with the same number of cores per node, in order from greatest number of cores per node to least number of cores per node. One or more tasks may be allocated to each node, depending on the number of cores per task and the number of cores per node. If enough nodes are available with enough cores for the job, then the job is submitted. The job is submitted using the "-l select=x1:ncpus=y1:mpiprocs=1+x2:ncpus=y2:mpiprocs=1+..." qsub option. The value of x1 is the number of nodes requested for the job with y1 cores, the value of x2 is the number of nodes requested for the job with y2 cores, etc. For the multiprocessing case, if the job is not submitted to the default server (and in certain other cases), then the job submission command requests exactly one task on each node. The job is submitted using the "-l select=x:ppn=y:mpiprocs=1" qsub option. The value of x is the number of tasks specified by the user and the value of y is the number of cores per task specified by the user.

- If the user specifies tasks manually and specifies the specific nodes to allocate to the job, including the total number of tasks and total number of cores for each node, then the job submission command will include a chunk in the select resource specification for each of the specified nodes. Each node chunk will include a host resource set to value hostname, where hostname is the name of the specified node. The ncpus resource for the chunk will be set to the total number of cores available for that node. That is, all cores on each specified node will be requested. The mpiprocs resource will be set to 1.
- If the user specifies tasks automatically and specifies the **Number of Cores and (Optional) RAM**, then if there are sufficient resources available, the job is submitted. The job is submitted using the "-l select=x:ncpus=1:mpiprocs=1:mem=y" qsub option. The value of x is the number of core specified by the user, and the value of y is equal to the RAM per core in MB specified by the user. If no RAM requirement is specified by the user, then the job is submitted using the "-l select=x:ncpus=1:mpiprocs=1" qsub option, where the value of x is the number of nodes specified by the user.
- If the user specifies tasks automatically and specifies the **Number of Nodes and Cores**, then the available nodes are separated into groups with the same number of cores per node. The groups are considered in order from largest number of cores per node to the smallest number of cores per node. If the required number of cores may be obtained by selecting the specified number of nodes from a single group of nodes with the same number of cores per node, then the job is submitted using the -l select=x:ncpus=y qsub option. The value of x is the same as the number of nodes specified by the user, and the value of y is equal to the number of cores per node for the group. Note that the total number of cores for the job (the product of x and y) will be equal to or greater than the user specified number of cores for the job., and it may be much greater in some cases.
- If the user specifies tasks automatically and specifies the specific nodes to allocate to the job, including the total number of cores for each node, then the job submission command will include a chunk in the select resource specification for each of the specified nodes.

Each node chunk will include a host resource set to value hostname, where hostname is the name of the specified node. The ncpus resource for the chunk will be set to the total number of cores available for that node. That is, all cores on each specified node will be requested. The mpiprocs resource will be set to 1. This is the same submission command as for the case for which the user specifies tasks manually and specifies the specific nodes to allocate to the job, including the total number of tasks and total number of cores for each node.

Configuration Guidelines

If the PBS_CONF environment variable is set, then the value of this environment variable is the pathname of the PBSPro configuration file. If the PBS_CONF environment variable is not set, then the PBSPro configuration file pathname is /etc/pbs.conf, the default.

If the PBS_DEFAULT environment variable is set, then the default server name is the value of this environment variable. If the PBS_DEFAULT environment variable is not set, then the default server name is the value of the PBS_SERVER parameter in the PBSPro configuration file.

If some users plan to specify tasks automatically and specify the **Number of Nodes and Cores**, then the user should also specify a queue, and the queue should be configured to efficiently supply the number of nodes and cores requested. For example, consider a case in which the user specifies tasks automatically, and specifies the **Number of Nodes and Cores** to be 4 and 16, so which may be satisfied by 4 nodes with 4 cores each. Because nodes are considered in order from larger numbers of cores per node to smaller numbers of cores per node, if the nodes with the most cores for the specified queue have 32 cores each, then the submission command will include the -l select=4:ncpus=32 qsub option, if there are at least 4 such nodes.

PBS/Torque

Job Submission Methods

- Manual: Number of Tasks and Cores: OK
- Manual: Individual Nodes: OK
- Automatic: Number of Cores and (Optional) RAM: OK
- Automatic: Number of Nodes and Cores: OK
- Automatic: Individual Nodes: OK

Background

For jobs submitted to a Torque cluster using the Ansys Electromagnetics job submission GUI, the user specified exclusive option is ignored.

The procs resource is not handled as expected for some versions of Torque. For these versions of Torque, only one core is allocated to a job even if the procs parameter is set to a value greater

than one. As a result, the procs resource should not be used for these versions of Torque. By default, the procs resource is only used for versions of Torque with a major version of 2. The ANSOFT_TORQUE_USE_PROCS environment variable may be used to override the default behavior. If this environment variable is set to value “1” (one), then the procs resource is used for all Torque versions. If this environment variable is set to value “0” (one), then the procs resource is not used for all Torque versions.

Distribution of Nodes, Tasks and Cores

- If the user specifies tasks manually and specifies the **Number of Tasks and Cores**, then the job submission parameters depend on the number of cores per task specified by the user. If the number of cores per task specified by the user is one (and it is not increased due to a limit on the number of tasks per node), then this is the no multiprocessing case, so the single cores tasks may be distributed to nodes with no additional restrictions. If the procs resource should not be used, as described above, then this case is treated the same as the multiprocessing case. For the no multiprocessing case, the job is submitted using the “-l procs=x” qsub option. The value of x is the number of tasks specified by the user. If the number of cores per task specified by the user is greater than one (or it is increased to a value greater than one due to a limit on the number of tasks per node), then this is the multiprocessing case. For the multiprocessing case, the available nodes are considered in groups with the same number of cores per node, in order from greatest number of cores per node to least number of cores per node. One or more tasks may be allocated to each node, depending on the number of cores per ask and the number of cores per node. If enough nodes are available with enough cores for the job, then the job is submitted. The job is submitted using the “-l nodes=x1:ppn=y1:+x2:ppn=y2+...” qsub option. The value of x1 is the number of nodes requested for the job with y1 cores, the value of x2 is the number of nodes requested for the job with y2 cores, etc. For the multiprocessing case, if the job is not submitted to the default server (and in certain other cases), then the job is always submitted to run one task on each node. The job is submitted using the “-l nodes=x:ppn=y” qsub option. The value of x is the number of tasks specified by the user and the value of y is the number of cores per task specified by the user.
- If the user specifies tasks manually and specifies the specific nodes to allocate to the job, including the total number of tasks and total number of cores for each node, then the job submission command will include a nodes option that includes each of the specified nodes. For each node, the ppn parameter will be equal to the total number of cores available for that node. That is, all cores on each specified node will be requested.
- If the user specifies tasks automatically and specifies the **Number of Cores and (Optional) RAM**, then if there are sufficient resources available, the job is submitted. The job is submitted using the “-l nodes=x1:ppn=y1,pmem=z+x2:ppn=y2,pmem=z+...” qsub option. The value of z is equal to the RAM per core in MB specified by the user. The value of x1 is the number of nodes having y1 cores per node, the value of x2 is the number of nodes have y2 cores per node, etc. If no RAM requirement is specified by the user, then

the “,pmem=z” specifications are omitted from the qsub option. The nodes are selected in order from having the largest number of cores per node to having the smallest number of core per node. If the procs resource may be used and if the user has specified no node attributes, then the job is submitted using the “-l procs=x,pmem=y” qsub option. The value of x is the total number of cores specified by the user and the value of y is equal to the RAM per core in MB specified by the user. If no RAM requirement is specified by the user, then the “,pmem=y” specifications are omitted from the qsub option.

- If the user specifies tasks automatically and specifies the **Number of Nodes and Cores**, then the available nodes are separated into groups with the same number of cores per node. The groups are considered in order from largest number of cores per node to the smallest number of cores per node. If the required number of cores may be obtained by selecting the specified number of nodes from a single group of nodes with the same number of cores per node, then the job is submitted using the -l nodes=x:ppn=y qsub option. The value of x is the same as the number of nodes specified by the user, and the value of y is equal to the number of cores per node for the group. Note that the total number of cores for the job (the product of x and y) will be equal to or greater than the user specified number of cores for the job., and it may be much greater in some cases.
- If the user specifies tasks automatically and specifies the specific nodes to allocate to the job, including the total number of cores for each node, then the job submission command will include a nodes option that includes each of the specified nodes. For each node, the ppn parameter will be equal to the total number of cores available for that node. That is, all cores on each specified node will be requested. This is the same as for the case for which the user specifies tasks manually and specifies the specific nodes to allocate to the job, including the total number of tasks and total number of cores for each node.

Configuration Guidelines

If the Torque home directory, TORQUEHOME, is different from the default directory, /var/spool/torque, then the environment variable ANSOFT_TORQUEHOME should be set to the pathname of the Torque home directory. The environment of each Ansys Electromagnetics job should include this environment variable setting.