



# NVIDIA Field Diagnostic Software

## Software Guide

# Document History

DU-05363-001\_v42

Version	Date	Authors	Description of Change
01-27	June 19, 2010 – December 21, 2017	CC	Changes in original Fielddiag package.
28	January 19, 2018	CC	Updated the document to accompany the new universal Fielddiag package.
29	April 1, 2020	CC	Add Turing GPUs.
30	October 20, 2020	CC	Add Ampere GPUs to supported SKUs Add skip_azalia_Init argument
31	November 10, 2020	CC	Add A10, A40 to supported GPUs
32	November 16, 2020	CC	Add gen3/gen4 arguments
33	February 18, 2021	CC	Add MIG mode consideration
34	Aug 2, 2023	SN	Add H100, A800 80GB, A100 LC, A100 BF2, A30 BF2, L4, L4 CEC, L40, A2, A10M Add aarch64 H100, A2, A100 80G, A16 Add all depth speed check
35	Oct 11, 2023	SN	Add L40S, H100 NVL, H800 NVL, SXM5 94GB/64GB HBM2e. Add Section on error code display
36	Nov 9, 2023	SS	Add section on memory row remapping. Add section on running the GPU Fielddiag through baseboard diagnostics.
37	Sep 26, 2024	KL	Update the Section of Error Logs for using OneDiag as GPU FD front end.
38	Oct 8, 2024	DB	Document the use of BS_Test to quickly re-check row remapping status.
39	Oct 11, 2024	GL	Add supported SKU; Add x1, board=<sku> to argument list; Add Known Issues and Release Notes section; Reformat the line spacing and border

			margin of document.
40	Ap 4, 2025	SN	Add Support to H20 NVL16, H200 NVL and GH100 PG520 SKU 266
41	Apr 14, 2025	SN	Add Support for RTX PRO 6000
42	Jul 3, 2025	SN	Add <i>reduced_power_connectors</i> argument support to run with lower TGP for RTX PRO 6000

# The NVIDIA Field Diagnostic Software

The information in this document is confidential and is the property of NVIDIA Corporation. This document may not be distributed without prior NVIDIA authorization.

## ● Introduction

This document describes the NVIDIA Field Diagnostic software (Fielddiag). This is a powerful software program that allows users to test the following NVIDIA Tesla hardware GPU hardware. It runs on most Linux variants based on a 2.6 kernel. (now includes NVIDIA Ampere architecture GPU skus). Its supported SKUs are listed below:

### **Ampere and before:**

NVIDIA A100, A40, A10 NVIDIA Quadro RTX 8000 NVIDIA Quadro RTX 6000  
NVIDIA T10, T4

TeslaV100, V100 PCIe 12GB, V100S Tesla P100, P40, P10, P4, P6

Tesla C2075

Tesla K8, K10, K20, K20X, K40, K40d, K80 Tesla M4, M6, M40, M60

NVIDIA A800 PCIe 80GB, L40, L40G L4, L4 CEC, H100 PG133 SKU 255,

NVIDIA A100 80GB LC, A100 40GB, A10M, A2, P509 80GB OAM, A30, A10, A10G  
NVIDIA A100X, Bluefield-2 A30, A30X, Bluefield-2 A100, A16, A100 80GB PCIe

NVIDIA A100 40GB PCIe

NVIDIA A100 80GB PCIe, NVIDIA A2, H100 (ARM aarch64)

### **Hopper:**

P1010 SKU 200, SKU 205, SKU 210, SKU 215, SKU 230

P1012 SKU 267

G520 SKU 200, SKU 202, SKU 205, SKU 207, SKU 213, SKU 221, SKU 228, SKU 235, SKU 236, SKU 237, SKU 238, SKU 266, SKU 280, SKU 282, SKU 292, SKU 266

G530 SKU 200, SKU 206, SKU 215, SKU 204, SKU 203

**Blackwell:**

G525 SKU 200, SKU 220, SKU 225, SKU 230

G548 SKU 201, G548 SKU 241

G153 SKU 210

- **Known Issues**

- **GPU Auto-detection**

GPU Auto-detection is a feature in fielddiag for Blackwell products that enables fielddiag to automatically detect and select the GPU board(s) under testing. It improves fielddiag by making it more generic to boards under testing.

There is small chance that when the system is in a very bad state, auto-detection will fail to recognize the GPU and display a RETEST banner. User should retest as follows.

Impacted SKUs are G525 SKU 200, SKU 220, SKU 225, SKU 230, G548 SKU 201.

Procedures to retest when GPU Auto-detection cannot recognize the board:

- Explicitly pass `board=<sku>` argument to fielddiag. For example, if the board under testing is G525 SKU 200, then the commandline would be `board=g525sku0200`.
- Power cycle the board and retest.

## ● Usage

### ● System Requirements

- ▷ Intel Pentium III or later CPU, or IBM POWER9 CPU
- ▷ 2GB or more of system memory
- ▷ Linux kernel 2.6.16 or later

Kernel 2.6.29 or later is recommended for performance reasons. The tool has been tested with kernels 2.6.16 through 2.6.35.

- ▷ glibc 2.12 or later required by Python3.10
- ▷ 64-bit x86\_64 kernels are supported. 32-bit kernels are not supported.
- ▷ TinyLinux version 12.08 or later is supported.
- ▷ Configured to run one of the following supported OSs:
  - Ubuntu 22.04 w/ NVIDIA optimized kernel v6.2
  - RHEL 9.2 z-stream with kernel v5.14
  - SLES 15 SP5 Security Update (SU) with kernel v5.14

### ● Running Fielddiag

Copy the tarball containing the Fielddiag package to the system under test, then extract the package. The package contains all documentation.

- Check if MIG mode is enabled and disable it if is enabled.

Multi-instance GPU (MIG) mode is available with A100 GPUs. It should not be enabled when running Fielddiag. Refer to the following for information about nvidia-smi and MIG mode.

- To view a help page related to MIG mode, issue
- [NVIDIA System Management Interface](https://developer.nvidia.com/nvidia-system-management-interface)
- [Management Interface](https://developer.nvidia.com/nvidia-system-management-interface)
- [Enabling MIG mode](https://docs.nvidia.com/datacenter/tesla/mig-user-guide/index.html)
- Run Fielddiag using the following command.  
`fielddiag` must be run as root, otherwise you will be asked to enter the root password.  
When running Fielddiag on more than two GPUs, a separate instance of Fielddiag is run on each GPU for faster testing.  
See section 2.6 for the list of optional command arguments.

## ● Return Codes

When completed, the diagnostic will return 0 to the shell if completed normally. If an error occurs, it will return 1 to the shell, and if a retest is required it will return 2. It will also print “PASS”, “FAIL”, or “RETEST” to the screen.

**PASS** – The hardware passes the diagnostics

**FAIL** – The hardware has failed the diagnostics

**RETEST** – The hardware setup has failed the pre-check portion of the diagnostics and a warning message appears describing the problem. Correct the problem per the pre-check message and then test again.

## ● Error Logs

By default, Fielddiag produces a binary log file which is used by NVIDIA engineers to diagnose a failing card. The log file will be captured under `onediag/dgx/log-<time stamp>/gpu_fielddiag` in the working directory, and the latest log can also be accessed under `onediag/dgx/latest_log/gpu_fielddiag`.

## ● Log File Contents

The following data is captured in the log file:

- GPU configuration information

- MODS version number
- MODS kernel driver version
- Linux kernel version
- ECC errors and retired pages
- Test results

No customer data, information about the system configuration, processes running on the system, nor data from folders outside of the MODS folder is captured in the log file.

- ## Log File Name

The name of the log file incorporates basic test result information as well as the serial number of the board under test:

fielddiag\_<PASS/FAIL/CONFIG>\_<serial number>.log

- ## Console Output on MultiGPU system

While running fielddiag on multigpu system the console will display the error code on each gpu along with the GPU PCI Bus Device Function and PASS/FAIL test status. The output.log for each gpu will be captured in onediag/dgx/log-<time stamp>/gpu\_fielddiag/<GPU\_PCI\_BDF>/ that will contain the details on the test result and will direct you to either RETEST if there are pre-check failure or will display PASS/FAIL results.

Following will be the format:

Testing gpu\_fielddiag OK [ 1:51s ]

Exit Code	Virtual Id	Test	Subtest	Component	Component Id	Notes
MODS-000000000000	gpu_fielddiag	gpu_fielddiag		GPU	05:00.0	OK
MODS-000000000000	gpu_fielddiag	gpu_fielddiag		GPU	06:00.0	OK
MODS-000000000000	gpu_fielddiag	gpu_fielddiag		GPU	45:00.0	OK
MODS-000000000000	gpu_fielddiag	gpu_fielddiag		GPU	46:00.0	OK
MODS-000000000000	gpu_fielddiag	gpu_fielddiag		GPU	85:00.0	OK
MODS-000000000000	gpu_fielddiag	gpu_fielddiag		GPU	86:00.0	OK
MODS-000000000000	gpu_fielddiag	gpu_fielddiag		GPU	b5:00.0	OK
MODS-000000000000	gpu_fielddiag	gpu_fielddiag		GPU	b6:00.0	OK
#####	####	#####	#####			
#####	####	#####	#####			
## ##	## ##	## ##	## ##			
## ##	## ##	## ##	## ##			
#####	#####	## ##	## ##			
#####	#####	## ##	## ##			
##	##	##	##			
##	##	##	##			
##	##	##	##			

- ## Fielddiag Command-Line Arguments
- ## Argument List

Table 1 lists the optional command-line arguments and provides a brief description of each. Further explanation is given in the next section.

Table 1. Optional Command Line Arguments

Option	Description	Application
device=<n>	Test the <i>n</i> th device.	All
gen1	Run fielddiag compliant with Gen1 PCIe.	All
gen2	Run fielddiag compliant with Gen2PCIe	All
gen3	Run fielddiag compliant with Gen3PCIe	NVIDIA A100, A40, A10
gen4	Run fielddiag compliant with Gen4PCIe	NVIDIA A100, A40, A10
gen5	Run fielddiag compliant with Gen5PCIe	Hopper, Blackwell
x1	Run fielddiag compliant with x1PCIe	All
x8	Run fielddiag compliant with x8PCIe.	All
gpu_temp=[ext,ADT7473,ADT7461, disabled,ipmi,int]	<p>Use the specified sensor for acquiring the GPU temperature:</p> <p>gpu_temp=ext: Read the GPU temperature using an external temperature chip that communicates back to the GPU.</p> <p>gpu_temp=ADT7473: Read the GPU temperature from the ADT7473 over a supported motherboard SMBus.</p> <p>gpu_temp=disabled: Disable temperature logging if motherboard SMBus support is not available and results in fielddiag test failures.</p> <p>gpu_temp=int: Read the temperature using the internal GPU sensor circuit for boards that support internal temperature reading.</p> <p>gpu_temp=ipmi:&lt;id0&gt;:...:&lt;idn&gt;</p> <p>Read the GPU temperature over the IPMI bus for each GPU in the system, where &lt;idx&gt; refers to each GPU's sensor ID (colon separated). Ex: &lt;id0&gt; is the sensor ID for GPU0, etc.</p> <p>To specify one GPU in a multi-GPU system when testing with device=&lt;n&gt;, populate non-tested ids with "0". Ex: gpu_temp=ipmi:0:0:0&lt;id3&gt; for GPU3.</p> <p>When not testing with device=&lt;n&gt;, a valid</p>	<p>Tesla M2075 Tesla M2090</p> <p>All</p> <p>All</p> <p>Tesla M2075 Tesla M2090 Linux</p>

	GPU sensor ID must be provided for all connected GPUs.	
p0only	Run the field diagnostic only in PState 0.	All
BS_Test	Run a basic system test that completes in a few minutes. Can be useful for quick reporting of certain kinds of GPU status.	All
filename=<filename-path>[_%r_%s]	<p>Specify a unique log file name other than the default.</p> <p>Ex. filename=/var/log/mylogfile.log            You can also append the unique filename with the same PASS/FAIL and SERIAL NUMBER information that is generated for the default logfile name:            Ex.            filename=mylogfile_%r_%s.log</p>	All
enable_graphics	Enable graphics capability (useful for additional test coverage). See requirements for using this option in section <a href="#">2.6.2 Additional Instructions</a> .	Tesla K20(X)
poll_interrupts	Use CPU polling for handling Tesla card interrupts.	All
power_cap_limit=<X>	Specify the limit, in watts, at which the board power consumption will be capped. Ex. power_cap_limit=200 (Limit power to 200 watts)	Tesla K8 Tesla K10 Tesla K20(X) Tesla K40 Tesla K40d Tesla K80 Linux
disable_progress_bar	Prevents the test progress from being displayed on the screen (see <a href="#">2.3 Test Progress</a> ).	All
board=<sku>	Explicitly specify the board SKU under test. If this argument is not set, the GPU Auto-detection will automatically detect the board to use. When GPU Auto-detection encounters issues, it will require retest with this argument being set. See section 2.1 for more details.	Blackwell
pciid=<w:x:y.z>	<p>Specify the board to be tested, where w, x, y, z are hexadecimal numbers:</p> <ul style="list-style-type: none"> <li>• is the PCI domain (required)</li> <li>• is the PCI bus (basically, which PCIe slot the board is installed in)</li> <li>• is the device</li> <li>• is the function</li> </ul> <p>Ex. pciid=0:2:0.0            See <a href="#">2.6.2 Additional Instructions</a> for more information.</p>	All
pci_devices=<w:x:y.z>, ...	Specify a subset of GPUs to be tested, where w, x, y, z are hexadecimal numbers. Each GPU address is comma-	All

	<p>separated:</p> <ul style="list-style-type: none"> <li>• is the PCI domain (required)</li> <li>• is the PCI bus (basically, which PCIe slot the board is installed in)</li> <li>• is the device</li> <li>• is the function</li> </ul> <p>Ex. pci_devices=0002:03:00.0,0003:04:00.0,0 004:05:00.0 Ex. pci_devices=0002:01:00.0,0003:01:00.0</p>	
max_file_size=<nnn>	<p>By default, the maximum log file size is limited to 50 MB, (500MB in Blackwell+) but you can specify a smaller or larger limit using this argument.</p> <p>Specify the maximum log file size, where <i>nnn</i> is the file size in MB.</p> <p>Ex. max_file_size=100 (Set maximum log file size to 100 MB) Ex. max_file_size=25 (Set maximum log file size to 25 MB)</p>	All
only_nvlink	<p>Run only the CheckConfig test and NvLink-related tests.</p> <p>Note: If the SKU does not support NVLink, the diagnostic returns the RETEST banner.</p> <p>Do not use this argument in conjunction with skip_nvlink in the same command line; doing so will result in a Fielddiag failure.</p>	NVIDIA T10 Tesla P100 Tesla V100 NVIDIA A100
skip_nvlink	<p>Skip (do not run) NvLink-related tests.</p> <p>Do not use this argument in conjunction with only_nvlink in the same command line; doing so will result in a Fielddiag failure. Run skip_nvlink while running fielddiag along with DGX/HGX systems</p>	Quadro RTX 8000 Quadro RTX 6000
nvlink_mask=<nnn>	<p>Specify which NvLink links to test with the CheckConfig test, where <i>nnn</i> is a binary mask that specifies the links. <i>nnn</i> can be decimal or hexadecimal (using 0x prefix). Ex. nvlink_mask=14 Ex. nvlink_mask=0xE</p>	
pex_lanes=<n>	<p>Run fielddiag compliant with PCIe, where <i>n</i> specifies the number of PCIe lanes.</p> <p>Ex. pex_lanes=2 (Run fielddiag compliant with x2 PCIe.)</p>	Tesla V100

single_instance	Applies only when there are more than two GPUs installed. Runs a single instance of fielddiag across all GPUs in the system. To run a single instance across a subset of GPUs, specify the GPUs using the <code>pci_devices</code> argument.	All
skip_azalia_init	Skip (do not run) initialization for Azalia devices.	Quadro RTX 8000 Quadro RTX 6000 NVIDIA A40
gen,<speed>,<depth_no>	Run fielddiag with gen speed and depth other than depth 0. Ex. gen3,1	All boards running other than depth 0
reduced_power_connectors	Run fielddiag with lower TGP or board with different boot modes via sideband strapping	RTX PRO 6000

Note: The GPU Field Diag for Blackwell SKU's will not run nvlink tests by default. To run nvlink tests, we need to specify `only_nvlink` option in the command line.

## Additional Instructions

This section provides addition instructions for specific command-line arguments.

- **enable\_graphics**

To run Fielddiag with graphics mode enabled, the diagnostic must be run directly after a reboot cycle without the NVIDIA driver loaded. Additionally, after Fielddiag is completed, a reboot is required to restore the card to normal operation.

- **pciid**

- This command-line argument can appear only once on the command line.
  - The `device=` command-line argument cannot appear in the same command line. Use `nvidia-smi` to obtain the values to enter. For example:

In this example, obtain the values from either “Bus Id” or “GPU” => 0000:01:00.0. The command line argument would then be `pciid=0:1:0.0`.

- **Run GPU Field Diagnostics from Baseboard Field Diagnostics**

Starting with HGX Fielddiag version 22.06-13, the GPU Fielddiag is packaged as part of the HGX Fielddiag and can be launched through the

HGX Fielddiag CLI. For more information, see the Baseboard Field Diagnostics user guide.

## ● **Running Under Linux**

This section applies to users who wish to run Fielddiag on a Linux distribution other than the one provided by NVIDIA. If you are using the NVIDIA-supplied distribution, you can skip this section.

- **Prerequisites for Running Under Linux**
- **Kernel Version**

Linux manufacturing Fielddiag requires a minimum kernel version of 2.6.16. Version 2.6.29 or later is recommended for performance reasons. Older versions have not been tested and may not work. Kernel 2.4 is not supported. The version of the running kernel can be established by running:

```
$ uname -r
```

- **Kernel Architecture**

Linux manufacturing Fielddiag is a 64-bit application and requires kernel compiled for x86\_64 architecture. To determine kernel architecture, type:

```
$ uname -m
```

- **glibc Version**

The system on which Fielddiag is run must be built on glibc-2.12 or newer. To determine glibc version, type:

```
$ /lib/libc.so.6
```

- **Do Not Run the NVIDIA Kernel Module**

For successful Fielddiag runs, the NVIDIA GPUs in the system must be in their original, unaltered state, as initialized by the VBIOS. This means X must not have been run on the NVIDIA GPUs prior to running Fielddiag. Make absolutely sure that the NVIDIA kernel module is not loaded, otherwise the system may become unstable. In order to unload the NVIDIA kernel module it is necessary to first kill X. Killing X is also recommended even if it is using Vesa or fb driver.

To disable X in SUSE, disable the xdm service in YaST.

On Debian-based systems (including Ubuntu), type:

```
$ sudo update-rc.d -f gdm remove
```

If not using gnome, type kde or xdm instead of gdm (as applicable).

- ## Unload the Nouveau Driver

Some newer Linux distributions include the nouveau driver in the kernel. This driver performs a kernel mode set and it also supports a framebuffer console. For Fielddiag to function correctly, this driver has to be unloaded (preferably blacklisted) so that it is not automatically loaded at boot.

- ## Disable any Framebuffer Consoles

Framebuffer consoles are also not recommended, because they may modify memory of the tested device during the tests. To disable the framebuffer console, edit /boot/grub/menu.lst and make sure the kernel arguments contain vga=normal instead of any other value. Make sure they do not contain anything like video=.

## Installing the Kernel Module

Linux manufacturing Fielddiag includes a kernel module. The purpose of this module is to expose certain kernel-mode APIs to Fielddiag, which runs as a user-mode application. In order to be able to install the kernel module, the system must contain configured kernel sources and development tools, including make and gcc. Without them it is not possible to compile the kernel module. Use package manager provided by your distribution to install kernel sources. Typically, the package name is kernel-sources or linux-sources. For example, on Debian type:

```
$ sudo apt-get install linux-source-`uname -r`
```

If you run Fielddiag as root, it will automatically run the included install\_module.sh script to compile and insert the Fielddiag kernel module. However if Fielddiag is not run by the root user, it is necessary to install the kernel module, which is recommended.

The easiest way to install the Fielddiag kernel module is to use the provided installation script, which you will find in the Fielddiag runspace:

This script also creates a udev configuration file in /etc/udev/rules.d/99-mods.rules. This file specifies the group which will be able to access the kernel module. By default this is the video group, such as for the NVIDIA driver. Make sure your user is in this group or change the group in the 99-mods.rules file to match one of yours. On some systems, the install script created file /etc/udev/permissions.d/99-mods.permissions instead, which simply lists user, group, and mode for the driver file.

- D To find out which group the driver has been assigned to, type:
- D To find out which groups you are in, type:

- D If you decide to modify the group in 99-mods.rules, you have to reload the kernel module:

To make sure the kernel module is always loaded when the system starts up, follow your distribution-specific guidelines:

### Debian-based distros (such as Ubuntu)

Add the mods module name to /etc/modules if the installation script didn't add it.

### SUSE-based distros

Add the mods module name to /etc/sysconfig/kernel file in the MODULES\_LOADED\_ON\_BOOT variable.

### RedHat-based distros (such as CentOS),

Add line modprobe mods to /etc/rc.d/rc.local.

## • Running an Upgraded Fielddiag Version

If you have run an older version of Fielddiag on your Linux distribution, the older MODS kernel module will conflict with the newer Fielddiag and cause failures.

Before running a newer Fielddiag, remove the older MODS kernel module as follows:

- Navigate to the Fielddiag subdirectory.
- From the command line, run

```
./install module.sh -u
```

When you run the later Fielddiag, the updated MODS kernel module will be installed automatically.

## • GPU Tests

The various tests in the package provide confirmation of the numerical processing engines in the GPU, integrity of data transfers to and from the GPU, and test coverage of the full onboard memory address space that is available to CUDA programs.

A typical GPU test performs the following operations:

- Disable the windowing system to take over the entire screen.
- Set the display mode and refresh rate.
- Loop N times:

- Exercise some aspect of the graphics hardware.
- Read back the resulting image.
- Calculate a 32-bit CRC, or possibly a checksum, to compare against the known correct value (golden value) for this GPU version and platform. For video and cursor tests use the hardware DAC CRC.
- If the golden values do not match, report an error and abort the loop.
- Restore previous display mode and refresh rate.
- Release screen to the operating system.
- Report test status.

Each test carefully chooses the random test parameters, i.e. invalid values are avoided, edge cases are properly covered, and proper weighting is given to more common cases.



## Row Remapping

The GPU Fielddiag triggers memory row remapping during execution, when available, after encountering ECC errors. Once a row remap occurs, it is necessary to retest the GPU.

Row-remapping, available starting in Ampere architecture, allows spare rows to replace defective row. The process of row-remapping requires a GPU reset to take effect and it will remain persistent throughout the life of the GPU.

Prior to version 22 of the GPU Fielddiag a retest needed to be triggered manually by re-running the diagnostic.

Version 22 introduces automatic retesting after row remapping has occurred. It will automatically re-run the fielddiag for up to 8 times. The retest limit is used to bound the amount of time the fielddiag will run in a single invocation. If during a retest more ECC errors are encountered, it will continue to row remap and retest until either:

- The diagnostic passes.
- Another type of failure occurs which would result in a FAIL.
- The retest limit is reached.

In the event the retest limit is reached, a RETEST banner will be printed, and the diagnostic will have to be retriggered manually.

A quick basic system test called BS\_Test can be run to check if a previously recorded number of row remaps has exceeded a critical threshold. This test completes within a few minutes. If a critical threshold has been exceeded, a return code of 363 “Row remapping failed” will be displayed.

### Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation (“NVIDIA”) makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice. Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer (“Terms of Sale”). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer’s sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer’s product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a

third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

#### Trademarks

NVIDIA and the NVIDIA logo are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

#### Copyright

© 2010, 2015-2021 NVIDIA Corporation. All rights reserved.



**NVIDIA** NVIDIA Corporation | 2788 San Tomas

Expressway, Santa Clara, CA 95051 <http://www.nvidia.com>