NVIDIA Field Diagnostic Software

Software Guide

Document History

DU-05363-001_v43

Version	Date	Authors	Description of Change
01-27	June 19, 2010 – December 21, 2017	CC	ChangesinoriginalFieldiagpackage.
28	January 19, 2018	CC	Updated the document to accompany the new universal Fieldiag package.
29	April 1, 2020	CC	Add Turing GPUs.
30	October 20, 2020	CC	AddAmpereGPUsto supportedSKUs 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 MIGmode 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 Fieldiag 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 *reduced_power_connectors* argument support to run with lower TGP for RTX PRO 6000
43	Sep 3, 2025	SN	Add Support for Quadro SKU's

The NVIDIA Field Diagnostic Software

• Introduction

This document describes the NVIDIA Field Diagnostic software (Fieldiag). 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

ADA:

HahnZ RTX 2000, BetheZ2, CosterZ2, HahnZ RTX 4000, BetheZ, Oppenheimer

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 fieldiag for Blackwell products that enables fieldiag to automatically detect and select the GPU board(s) under testing. It improves fieldiag 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 fieldiag. 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 Fieldiag

Copy the tarball containing the Fieldiag 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 Fieldiag. Refer to the following for information about nvidia-smi and MIG mode.

  - To view a help page related to MIG mode, issue
  - NVIDIA System HYPERLINK "https://developer.nvidia.com/nvidia-system-management-interface" HYPERLINK "https://developer.nvidia.com/nvidia-system-management-interface"Management HYPERLINK "https://developer.nvidia.com/nvidia-system-management-interface" HYPERLINK "https://developer.nvidia.com/nvidia-system-management-interface"Interface
  - Enabling HYPERLINK "https://docs.nvidia.com/datacenter/tesla/mig-user-guide/index.html" HYPERLINK "https://docs.nvidia.com/datacenter/tesla/mig-user-guide/index.html"MIG HYPERLINK "https://docs.nvidia.com/datacenter/tesla/mig-user-guide/index.html" HYPERLINK "https://docs.nvidia.com/datacenter/tesla/mig-user-guide/index.html"mode.

- Run Fieldiag using the following command.

  `fieldiag` must be run as root, otherwise you will be asked to enter the root password.

  When running Fieldiag on more than two GPUs, a separate instance of Fieldiag 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, Fieldiag 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_fieldiag in the working directory, and the latest log can also be accessed under onediag/dgx/latest_log/gpu_fieldiag.

• 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:

fieldiag_<PASS/FAIL/CONFIG>_<serial number>.log

• Console Output on MultiGPU system

While running fieldiag 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_fieldiag/<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_fieldiag OK [ 1:51s ]

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

#######     ####     ######    ######
########   ######   ########  ########
##    ##  ##    ##  ##     #  ##     #
##     ## ##    ##   ###       ###
########  ########    ####      ####
#######   ########      ###       ###
##        ##    ##  #     ##  #     ##
##        ##    ##  ########  ########
##        ##    ##   ######    ######
```

• Fieldiag 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 *n*th device.	All
gen1	Run fieldiag compliant with Gen1 PCIe.	All
gen2	Run fieldiag compliant with Gen2 PCIe	All
gen3	Run fieldiag compliant with Gen3 PCIe	NVIDIA A100, A40, A10
gen4	Run fieldiag compliant with Gen4 PCIe	NVIDIA A100, A40, A10
gen5	Run fieldiag compliant with Gen5 PCIe	Hopper, Blackwell
x1	Run fieldiag compliant with x1 PCIe	All
x8	Run fieldiag compliant with x8 PCIe.	All
gpu_temp=[ext,ADT7473,ADT7461, disabled,ipmi,int]	Use the specified sensor for acquiring the GPU temperature:	
	gpu_temp=ext: Read the GPU temperature using an external temperature chip that communicates back to the GPU.	All
	gpu_temp=ADT7473: Read the GPU	Tesla M2075

	temperaturefrom theADT7473overa supportedmotherboardSMBus.	Tesla M2090
	gpu_temp=disabled: Disable temperature logging if motherboard SMBus support is not available and resultsinfieldiagtestfailures.	All
	gpu_temp=int: Read the temperature using the internal GPU sensor circuit for boards that support internal temperature reading.	All
	gpu_temp=ipmi:<id0>:…:<id*n*> ReadtheGPUtemperatureovertheIPMI bus for each GPU in the system, where <idx>refers to each GPU's sensorID (colon separated). Ex: <id0> is the sensorID for GPU0, etc. To specify one GPU in a multi-GPU system when testing with device=<n>, populate non-tested ids with "0". Ex: gpu_temp=ipmi:0:0:0<id3> for GPU3. When not testing with device=<n>,a valid GPU sensor ID must be provided for all connected GPUs.	Tesla M2075 Tesla M2090 Linux
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
logfilename=<*filename-path*>[_%r_%s]	Specify a unique log file name other than the default. Ex. logfilename=/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. logfilename=mylogfile_%r_%s.log	All
enable_graphics	Enable graphics capability (useful for additional test coverage). See requirements for using this option in section 2.6.2 Additional Instructions.	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 2.3 Test Progress).	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*>	Specify the board to be tested, where w, x, y, z are hexadecimal numbers: • is the PCI domain (required) • is the PCI bus (basically, which PCIe slot the board is installed in) • is the device • is the function Ex. pciid=0:2:0.0 See 2.6.2 Additional Instructions for more information.	All
pci_devices=<*w*:*x*:*y*.*z*>, …	Specify a subset of GPUs to be tested, where w, x, y, zare hexadecimal numbers. Each GPU address is comma-separated: • is the PCI domain (required) • is the PCI bus (basically, which PCIe slot the board is installed in) • is the device • is the function Ex. pci_devices=0002:03:00.0,0003:04:00.0,0004:05:00.0 Ex. pci_devices=0002:01:00.0,0003:01:00.0	All
max_file_size=<*nnn*>	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. Specify the maximumlog file size, where *nnn* is the file size in MB. 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)	All

only_nvlink	Run only the CheckConfig test and NvLink-related tests. Note: If the SKU does not support NVLink, the diagnostic returns the RETEST banner. Do not use this argument in conjunction with skip_nvlink in the same command line; doing so will result in a Fieldiag failure.	NVIDIA T10 Tesla P100 Tesla V100 NVIDIA A100
skip_nvlink	Skip (do not run) NvLink-related tests. Do not use this argument in conjunction with only_nvlink in the same command line; doing so will result in a Fieldiag failure. Run skip_nvlink while running fieldiag along with DGX/HGX systems	Quadro RTX 8000 Quadro RTX 6000
nvlink_mask=<*nnn*>	Specify which NvLink links to test with the CheckConfig test, where *nnn* is a binary mask that specifies the links. *nnn* can be decimal or hexadecimal (using 0x prefix). Ex. nvlink_mask=14 Ex. nvlink_mask=0xE	
pex_lanes=<*n*>	Run fieldiag compliant with PCIe, where *n* specifies the number of PCIe lanes. Ex. pex_lanes=2 (Run fieldiag compliantwithx2 PCIe.)	Tesla V100
single_instance	Applies only when there are more than two GPUs installed. Runs a single instance of fieldiag across all GPUs in the system. To run a single instance across a subset of GPUs, specify the GPUs using the pci_devices 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 fieldiag with gen speed and depth other than depth 0. Ex. gen3,1	All boards running other than depth 0
reduced_power_connectors	Run fieldiag 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 Fieldiag with graphics mode enabled, the diagnostic must be run directly after a reboot cycle without the NVIDIA driver loaded. Additionally, after Fieldiag 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 Fieldiag version 22.06-13, the GPU Fieldiag is packaged as part of the HGX Fieldiag and can be launched through the HGX Fieldiag CLI. For more information, see the Baseboard Field Diagnostics user guide.

Running Under Linux

This section applies to users who wish to run Fieldiag 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 Fieldiag 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 Fieldiag 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 Fieldiag 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 Fieldiag 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 Fieldiag. 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 gdb (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 alsosupports a framebuffer console. For Fieldiag to function correctly, this driver has to be unloaded (preferably blacklisted) so that it is not automatically loadedatboot.

- ## Disable any Framebuffer Consoles

Framebuffer consoles are alsonot recommended, becausethey may modify memory ofthe tested device during the tests. To disablethe framebuffer console, edit /boot/grub/menu.lst andmakesure thekernelarguments contain vga=normal instead ofany other value. Make suretheydonotcontain anything likevideo=.

sInstalling the Kernel Module

Linux manufacturing Fieldiag includes a kernel module. The purpose of this module is to expose certain kernel-mode APIs to Fieldiag, 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. With out 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 Fieldiag as root, it will automatically run the included install_module.sh script to compile and insert the Fieldiag kernel module. However if Fieldiag is not run by the root user, it is necessary to install the kernel module, which is recommended.

The easiest way to install the Fieldiag kernel module is to use the provided installation script, which you will find in the Fieldiag 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.

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

- 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 Fieldiag Version

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

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

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

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

When you run the later Fieldiag, 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 memoryaddressspacethatisavailableto CUDAprograms.

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 Fieldiag 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 Fieldiag 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 fieldiag for up to 8 times. The retest limit is used to bound the amount of time the fieldiag 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.