



# NIC Subsystem Management

# Table of contents

## Redfish NIC Subsystem Management

---

Privilege Modes (Presets)

---

Configuring BlueField Mode of Operation

---

Host Privileges Configuration

---

Getting Host RShim

---

Disabling Host RShim

---

Enabling Host RShim

---

Getting Strap Options

---

Example Usage

---

Configuration Breakdown by Mode

---

Privilege Settings Definitions

---

## Setting Host Privilege Configuration

---

Set Specific Properties

---

View Pending Settings

---

Transitioning to Restricted Mode

---

Logic and Constraints

---

Parameter Precedence

---

Configuration Examples

---

Set Privilege Mode

---

## IPMItool NIC Subsystem Management

---

Enabling/Disabling RShim from Host

---

Setting Operation Mode

---

## **Note**

This content is relevant for NVIDIA® BlueField®-3 devices only.

## **Redfish NIC Subsystem Management**

### **Configuring BlueField Mode of Operation**

Refer to "[BlueField Modes of Operation Configuration](#)" for information.

### **Getting Host RShim**

```
curl -k -u root:'<password>' -X GET  
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Oem/Nvidia
```

### **Enabling Host RShim**

```
curl -k -u root:'<password>' -H "Content-Type: application/json"  
-X POST -d '{"HostRshim": "Enabled"}'  
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Oem/Nvidia/Actions/Hc
```

### **Disabling Host RShim**

```
curl -k -u root:'<password>' -H "Content-Type: application/json"
-X POST -d '{"HostRshim":"Disabled"}'
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Oem/Nvidia/Actions/Hc
```

## Getting Strap Options

```
curl -k -u root:'<password>' -X GET
https://<bmc_ip>/redfish/v1/Systems/Bluefield/Oem/Nvidia/Connectx/S
```

## Host Privileges Configuration

This resource manages the security privileges assigned to the host interface. It allows administrators to restrict the host's ability to modify device configurations or access sensitive parameters.

```
https://<bmc_ip>/redfish/v1/Chassis/Card1/NetworkAdapters/NvidiaNe
```

### Privilege Modes (Presets)

The `PrivilegeMode` attribute acts as a master switch, applying a predefined set of permissions.

Mode	Description
Privileged (default)	Grants full access. The host can modify firmware, flash, and global parameters.
Restricted	Locks down the host. Prevents modification of firmware, flash, and global parameters. RSHIM and Tracer access are disabled.

## Configuration Breakdown by Mode

The following table shows exactly which permissions are enabled or disabled for each mode:

Setting	Privileged Mode	Restricted Mode
HostPrivilegeLevel	Privileged	Restricted
FirmwareUpdate	Enabled	Disabled
FlashAccess	Enabled	Disabled
GlobalParametersAccess	Enabled	Disabled
HostParametersAccess	Enabled	Disabled
InternalCPUAccess	Enabled	Disabled
NicReset	Enabled	Disabled
PccUpdate	Enabled	Disabled
PortAccess	Enabled	Disabled
ManagementInterfaceEnabled	true	false
PortOwnerEnabled	true	false
ReadCountersEnabled	true	false
TracerEnabled	true	false

## Privilege Settings Definitions

This table defines the specific behavior controlled by each permission setting.

Setting	Description	Options	Default
FlashAccess	Permission to perform any device flash access.	Default, Enable, Disable	Default

Setting	Description	Options	Default
PccUpdate	Permission to update the Programmable Congestion Control (PCC) algorithm.	Default, Enable, Disable	Default
FirmwareUpdate	Permission to perform firmware updates.	Default, Enable, Disable	Default
NicReset	Permission to perform a NIC Reset.	Default, Enable, Disable	Default
GlobalParametersAccess	Permission to access global non-volatile (NV) parameters.	Default, Enable, Disable	Default
HostParametersAccess	Permission to access host NV parameters.	Default, Enable, Disable	Default
PortAccess	Permission to access port NV parameters.	Default, Enable, Disable	Default
InternalCPUAccess	Permission to access Internal CPU NV parameters.	Default, Enable, Disable	Default
ManagementInterfaceEnabled	Controls RSHIM function. If <code>false</code> , the host cannot access embedded CPU registers.	<code>true</code> , <code>false</code>	—
PortOwnerEnabled	Controls Port Ownership. If <code>false</code> , the host cannot become the Port Owner.	<code>true</code> , <code>false</code>	—

Setting	Description	Options	Default
ReadCountersEnabled	Controls physical counter access. If <code>false</code> , the host cannot read physical port counters.	<code>true</code> , <code>false</code>	—
TracerEnabled	Controls Tracer ownership. If <code>false</code> , the host cannot own the Tracer.	<code>true</code> , <code>false</code>	—

## Example Usage

The following example demonstrates a `GET` request to retrieve the current privilege settings.

- Request:

```
curl -u 'root': '<password>' -X GET
https://<bmc_ip>/redfish/v1/Chassis/Card1/NetworkAdapters/NvidiaNetworkAdapter/Oem/Nvidia/
```

- Response:

```

{
  "@Redfish.Settings": {
    "@odata.type": "#Settings.v1_3_5.Settings",
    "SettingsObject": {
      "@odata.id":
"/redfish/v1/Chassis/Card1/NetworkAdapters/NvidiaNetworkAdapter/Oem/Nvidia/HostPrivilegeConfig/S
    },
    "@odata.id":
"/redfish/v1/Chassis/Card1/NetworkAdapters/NvidiaNetworkAdapter/Oem/Nvidia/HostPrivilegeConfig",
    "@odata.type": "#NvidiaHostPrivilegeConfig.v1_0_0.NvidiaHostPrivilegeConfig",
    "Id": "HostPrivilegeConfig",
    "Name": "Host Privilege Configuration",
    "PrivilegeMode": "Privileged",
    "PrivilegeSettings": {
      "FirmwareUpdate": "Enabled",
      "FlashAccess": "Enabled",
      "GlobalParametersAccess": "Enabled",
      "HostParametersAccess": "Enabled",
      "HostPrivilegeLevel": "Privileged",
      "InternalCPUAccess": "Enabled",
      "ManagementInterfaceEnabled": true,
      "NicReset": "Enabled",
      "PccUpdate": "Enabled",
      "PortAccess": "Enabled",
      "PortOwnerEnabled": true,
      "ReadCountersEnabled": true,
      "TracerEnabled": true
    }
  }
}

```

# Setting Host Privilege Configuration

To modify host privileges, send a `PATCH` request to the `Settings` URI.

```
PATCH
```

```
https://<bmc_ip>/redfish/v1/Chassis/Card1/NetworkAdapters/NvidiaNe
```

## Note

The `/Settings` path displays pending values. These changes do not take effect immediately.

## Configuration Examples

### View Pending Settings

Before making changes, you can verify the current pending configuration.

- Request

```
curl -u 'root': '<password>' -X GET  
https://<bmc_ip>/redfish/v1/Chassis/Card1/NetworkAdapters/Nvid
```

- Response

```
{
  "@odata.id":
"/redfish/v1/Chassis/Card1/NetworkAdapters/NvidiaNetworkAdapte
  "@odata.type":
"#NvidiaHostPrivilegeConfig.v1_0_0.NvidiaHostPrivilegeConfig",
  "Id": "Settings",
  "Name": "Host Privilege Configuration Settings",
  "PrivilegeMode": "Privileged",
  "PrivilegeSettings": {
    "FirmwareUpdate": "Enabled",
    "FlashAccess": "Enabled",
    "GlobalParametersAccess": "Enabled",
    "HostParametersAccess": "Enabled",
    "HostPrivilegeLevel": "Privileged",
    "InternalCPUAccess": "Enabled",
    "ManagementInterfaceEnabled": true,
    "NicReset": "Enabled",
    "PccUpdate": "Enabled",
    "PortAccess": "Enabled",
    "PortOwnerEnabled": true,
    "ReadCountersEnabled": true,
    "TracerEnabled": true
  }
}
```

## Set Privilege Mode

To apply a high-level preset (Privileged or Restricted):

```
curl -u 'root': '<password>' -X PATCH -H "Content-Type:
application/json" \
https://<bmc_ip>/redfish/v1/Chassis/Card1/NetworkAdapters/NvidiaNe-
\
-d '{"PrivilegeMode": "Privileged"}'
```

## Set Specific Properties

To apply granular permissions. Note that these are nested within the `PrivilegeSettings` object:

```
curl -u 'root': '<password>' -X PATCH -H "Content-Type:
application/json" \
https://<bmc_ip>/redfish/v1/Chassis/Card1/NetworkAdapters/NvidiaNe-
\
-d '{
  "PrivilegeSettings": {
    "NicReset": "Default",
    "PccUpdate": "Default",
    "PortAccess": "Default",
    "FirmwareUpdate": "Default",
    "FlashAccess": "Disabled",
    "GlobalParametersAccess": "Disabled",
    "HostParametersAccess": "Disabled",
    "InternalCPUAccess": "Disabled",
    "HostPrivilegeLevel": "Privileged"
  }
}'
```

## Logic and Constraints

## **i** Note

Every change requires a System Power Cycle to take effect.

## Parameter Precedence

If a conflict arises between granular access controls and global/functional flags, the granular access controls take precedence.

Category	Parameters	Precedence
Granular Access Controls	HostParametersAccess, PortAccess, GlobalParametersAccess, InternalCPUAccess, PccUpdate, FirmwareUpdate, FlashAccess, NicReset	High (wins conflicts)
Global and Functional Flags	HostPrivilegeLevel, ManagementInterfaceEnabled, PortOwnerEnabled, ReadCountersEnabled, TracerEnabled	Low

## Transitioning to Restricted Mode

If any of the following properties are currently set to `Enabled`, you cannot change `HostPrivilegeLevel` to `Restricted` immediately:

- `GlobalParametersAccess`
- `PortAccess`
- `InternalCPUAccess`
- `NicReset`

- FirmwareUpdate
- FlashAccess

You must first (or simultaneously) set the conflicting property to `Default` before the system accepts the `Restricted` level.

## IPMItool NIC Subsystem Management

Since the standard IPMItool commands do not cover all functionality, a set of custom NVIDIA IPMItool `raw` commands is available to enable configuring the NIC subsystem on the BlueField directly.

IPMItool raw commands follow the following format:

```
ipmitool -C 17 -I lanplus -H <bmc_ip_addr> -U <username> -P  
<password> raw <netfunc> <cmd> <data>
```

Where:

- `netfunc` – network function which identifies the functional message class, and clusters IPMI commands into sets
- `cmd` – one byte command within a network function
- `data` – optional element which provides additional parameters for a request or response message

The following table lists the supported IPMItool raw commands:

netfunc	cmd	data	Description																		
0x32	0x9A	N/A	<p>Get external host privileges. Prints current state for all fields:</p> <table border="1"> <thead> <tr> <th>Byte</th> <th>Field</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>HOST_PRIV_FLASH_ACCESS</td> </tr> <tr> <td>1</td> <td>HOST_PRIV_FW_UPDATE</td> </tr> <tr> <td>2</td> <td>HOST_PRIV_NIC_RESET</td> </tr> <tr> <td>3</td> <td>HOST_PRIV_NV_GLOBAL</td> </tr> <tr> <td>4</td> <td>HOST_PRIV_NV_HOST</td> </tr> <tr> <td>5</td> <td>HOST_PRIV_NV_INTERNAL_CPU</td> </tr> <tr> <td>6</td> <td>HOST_PRIV_NV_PORT</td> </tr> <tr> <td>7</td> <td>HOST_PRIV_PCC_UPDATE</td> </tr> </tbody> </table> <p>Each state is represented by binary byte in order.</p> <ul style="list-style-type: none"> <li>• 00 — Default</li> <li>• 01 — Enabled</li> <li>• 02 — Disabled</li> </ul>	Byte	Field	0	HOST_PRIV_FLASH_ACCESS	1	HOST_PRIV_FW_UPDATE	2	HOST_PRIV_NIC_RESET	3	HOST_PRIV_NV_GLOBAL	4	HOST_PRIV_NV_HOST	5	HOST_PRIV_NV_INTERNAL_CPU	6	HOST_PRIV_NV_PORT	7	HOST_PRIV_PCC_UPDATE
Byte	Field																				
0	HOST_PRIV_FLASH_ACCESS																				
1	HOST_PRIV_FW_UPDATE																				
2	HOST_PRIV_NIC_RESET																				
3	HOST_PRIV_NV_GLOBAL																				
4	HOST_PRIV_NV_HOST																				
5	HOST_PRIV_NV_INTERNAL_CPU																				
6	HOST_PRIV_NV_PORT																				
7	HOST_PRIV_PCC_UPDATE																				

netfunc	cmd	data	Description																		
0x32	0x9B	Byte0 Byte1	<p>Set external host privilege:  Byte0 selects privilege according to the following table:</p> <table border="1"> <thead> <tr> <th>Byte</th> <th>Field</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>HOST_PRIV_FLASH_ACCESS</td> </tr> <tr> <td>1</td> <td>HOST_PRIV_FW_UPDATE</td> </tr> <tr> <td>2</td> <td>HOST_PRIV_NIC_RESET</td> </tr> <tr> <td>3</td> <td>HOST_PRIV_NV_GLOBAL</td> </tr> <tr> <td>4</td> <td>HOST_PRIV_NV_HOST</td> </tr> <tr> <td>5</td> <td>HOST_PRIV_NV_INTERNAL_CPU</td> </tr> <tr> <td>6</td> <td>HOST_PRIV_NV_PORT</td> </tr> <tr> <td>7</td> <td>HOST_PRIV_PCC_UPDATE</td> </tr> </tbody> </table> <p>Byte1 is the value being set.  Supported values:</p> <ul style="list-style-type: none"> <li>00 — Default</li> <li>01 — Enabled</li> <li>02 — Disabled</li> </ul>	Byte	Field	0	HOST_PRIV_FLASH_ACCESS	1	HOST_PRIV_FW_UPDATE	2	HOST_PRIV_NIC_RESET	3	HOST_PRIV_NV_GLOBAL	4	HOST_PRIV_NV_HOST	5	HOST_PRIV_NV_INTERNAL_CPU	6	HOST_PRIV_NV_PORT	7	HOST_PRIV_PCC_UPDATE
Byte	Field																				
0	HOST_PRIV_FLASH_ACCESS																				
1	HOST_PRIV_FW_UPDATE																				
2	HOST_PRIV_NIC_RESET																				
3	HOST_PRIV_NV_GLOBAL																				
4	HOST_PRIV_NV_HOST																				
5	HOST_PRIV_NV_INTERNAL_CPU																				
6	HOST_PRIV_NV_PORT																				
7	HOST_PRIV_PCC_UPDATE																				

netfunc	cmd	data	Description
			<p><b>(i) Note</b></p> <ul style="list-style-type: none"> <li>The parameter <code>HOST_PRIV_NV_INTERNAL_CPU</code> should either equal the parameter <code>HOST_PRIV_NV_GLOBAL</code> or one of them should be set to <code>DEVICE_DEFAULT</code>;</li> <li>If the parameter <code>HOST_PRIV_FLASH_ACCESS</code> is not set to <code>DEVICE_DEFAULT</code> then the following parameters should all be set to <code>DEVICE_DEFAULT</code> or be equal to the value of <code>HOST_PRIV_FLASH_ACCESS</code>:  <code>HOST_PRIV_NV_HOST</code> ;  <code>HOST_PRIV_NV_PORT</code> ;  <code>HOST_PRIV_NV_GLOBAL</code> ;  <code>HOST_PRIV_NV_INTERNAL_CPU</code> ;  <code>HOST_PRIV_PCC_UPDATE</code> ;  <code>HOST_PRIV_FW_UPDATE</code> ;</li> </ul>
<code>0x32</code>	<code>0x9C</code>	N/A	<p>Get SmartNIC mode. Prints current configuration:  <code>INTERNAL_CPU_OFFLOAD_ENGINE</code> .</p> <ul style="list-style-type: none"> <li><code>00</code> – Disabled</li> <li><code>01</code> – Enabled</li> </ul>

netfunc	cmd	data	Description
0x32	0x9D	Byte0	Set SmartNIC mode ( <code>INTERNAL_CPU_OFFLOAD_ENGINE</code> ) to <code>Byte0</code> . Supported values: <ul style="list-style-type: none"> <li>• <code>00</code> – Disabled</li> <li>• <code>01</code> – Enabled</li> </ul>
0x32	0x9E	N/A	Get host access. Prints current <code>HOST_PRIV_RSHIM</code> . <ul style="list-style-type: none"> <li>• <code>00</code> – Disabled</li> <li>• <code>01</code> – Enabled</li> </ul>
0x32	0x9F	Byte0	Set host access. Sets <code>HOST_PRIV_RSHIM</code> to <code>Byte0</code> . Supported values: <ul style="list-style-type: none"> <li>• <code>00</code> – Disabled</li> <li>• <code>01</code> – Enabled</li> </ul>

netfunc	cmd	data	Description																																														
0x32	0xA2	N/A	<p>Query strap options. Prints current state for all fields:</p> <table border="1"> <thead> <tr> <th>Byte</th> <th>Field</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>VERSION</td> </tr> <tr> <td>1</td> <td>DISABLE_INBAND_RECOVER_VALUE</td> </tr> <tr> <td>2</td> <td>PRIMARY_IS_PCORE_1_VALUE</td> </tr> <tr> <td>3</td> <td>2PCORE_ACTIVE_VALUE</td> </tr> <tr> <td>4</td> <td>SOCKET_DIRECT_VALUE</td> </tr> <tr> <td>5</td> <td>PCI_REVERSAL_VALUE</td> </tr> <tr> <td>6</td> <td>PCI_PARTITION_1_VALUE</td> </tr> <tr> <td>7</td> <td>PCI_PARTITION_0_VALUE</td> </tr> <tr> <td>8</td> <td>OSC_FREQ_1_VALUE</td> </tr> <tr> <td>9</td> <td>OSC_FREQ_0_VALUE</td> </tr> <tr> <td>10</td> <td>CORE_BYPASS_N_VALUE</td> </tr> <tr> <td>11</td> <td>FNP_VALUE</td> </tr> <tr> <td>12</td> <td>DISABLE_INBAND_RECOVER_VALUE</td> </tr> <tr> <td>13</td> <td>PRIMARY_IS_PCORE_1_MASK</td> </tr> <tr> <td>14</td> <td>2PCORE_ACTIVE_MASK</td> </tr> <tr> <td>15</td> <td>SOCKET_DIRECT_MASK</td> </tr> <tr> <td>16</td> <td>PCI_REVERSAL_MASK</td> </tr> <tr> <td>17</td> <td>PCI_PARTITION_1_MASK</td> </tr> <tr> <td>18</td> <td>PCI_PARTITION_0_MASK</td> </tr> <tr> <td>19</td> <td>OSC_FREQ_1_MASK</td> </tr> <tr> <td>20</td> <td>OSC_FREQ_0_MASK</td> </tr> <tr> <td>21</td> <td>CORE_BYPASS_N_MASK</td> </tr> </tbody> </table>	Byte	Field	0	VERSION	1	DISABLE_INBAND_RECOVER_VALUE	2	PRIMARY_IS_PCORE_1_VALUE	3	2PCORE_ACTIVE_VALUE	4	SOCKET_DIRECT_VALUE	5	PCI_REVERSAL_VALUE	6	PCI_PARTITION_1_VALUE	7	PCI_PARTITION_0_VALUE	8	OSC_FREQ_1_VALUE	9	OSC_FREQ_0_VALUE	10	CORE_BYPASS_N_VALUE	11	FNP_VALUE	12	DISABLE_INBAND_RECOVER_VALUE	13	PRIMARY_IS_PCORE_1_MASK	14	2PCORE_ACTIVE_MASK	15	SOCKET_DIRECT_MASK	16	PCI_REVERSAL_MASK	17	PCI_PARTITION_1_MASK	18	PCI_PARTITION_0_MASK	19	OSC_FREQ_1_MASK	20	OSC_FREQ_0_MASK	21	CORE_BYPASS_N_MASK
Byte	Field																																																
0	VERSION																																																
1	DISABLE_INBAND_RECOVER_VALUE																																																
2	PRIMARY_IS_PCORE_1_VALUE																																																
3	2PCORE_ACTIVE_VALUE																																																
4	SOCKET_DIRECT_VALUE																																																
5	PCI_REVERSAL_VALUE																																																
6	PCI_PARTITION_1_VALUE																																																
7	PCI_PARTITION_0_VALUE																																																
8	OSC_FREQ_1_VALUE																																																
9	OSC_FREQ_0_VALUE																																																
10	CORE_BYPASS_N_VALUE																																																
11	FNP_VALUE																																																
12	DISABLE_INBAND_RECOVER_VALUE																																																
13	PRIMARY_IS_PCORE_1_MASK																																																
14	2PCORE_ACTIVE_MASK																																																
15	SOCKET_DIRECT_MASK																																																
16	PCI_REVERSAL_MASK																																																
17	PCI_PARTITION_1_MASK																																																
18	PCI_PARTITION_0_MASK																																																
19	OSC_FREQ_1_MASK																																																
20	OSC_FREQ_0_MASK																																																
21	CORE_BYPASS_N_MASK																																																

netfunc	cmd	data	Description				
			<table border="1"> <thead> <tr> <th>Byte</th> <th>Field</th> </tr> </thead> <tbody> <tr> <td>22</td> <td>FNP_MASK</td> </tr> </tbody> </table> <p>Each state is represented by binary byte in order. Supported values:</p> <ul style="list-style-type: none"> <li>• 00 – Disabled</li> <li>• 01 – Enabled</li> </ul>	Byte	Field	22	FNP_MASK
Byte	Field						
22	FNP_MASK						
0x32	0xA3	N/A	<p>Get SmartNIC OS State.</p> <ul style="list-style-type: none"> <li>• 00 – BootRom</li> <li>• 01 – BL2</li> <li>• 02 – BL31</li> <li>• 03 – UEFI</li> <li>• 04 – OsStarting</li> <li>• 05 – OsIsRunning</li> <li>• 06 – LowPowerStandby</li> <li>• 07 – FirmwareUpdateInProgress</li> <li>• 08 – OsCrashDumpInProgress</li> <li>• 09 – OsCrashDumpsComplete</li> <li>• 0A – FWFaultCrashDumpInProgress</li> <li>• 0B – FWFaultCrashDumpsComplete</li> <li>• 0C – Invalid</li> </ul>				

## Setting Operation Mode

netfunc	cmd	data	Description
0x32	0x9D	0x1	Set DPU mode
0x32	0x9D	0x0	Set NIC mode

## Enabling/Disabling RShim from Host

<b>netfunc</b>	<b>cmd</b>	<b>data</b>	<b>Description</b>
0x32	0x9F	0x1	Enable RShim from host
0x32	0x9F	0x0	Disable RShim from host

## 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 products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

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 NON-INFRINGEMENT, 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 2026, NVIDIA. PDF Generated on 02/28/2026