



**Reset Flow**

# Table of contents

Kernel ULPs

---

User Space Applications (IB/RoCE)

---

SR-IOV

---

Forcing the VF to Reset

---

Extended Error Handling (EEH)

---

CRDUMP

---

Firmware Tracer

---

Reset Flow is activated by default. Once a "fatal device" error is recognized, both the HCA and the software are reset, the ULPs and user application are notified about it, and a recovery process is performed once the event is raised.

Currently, a reset flow can be triggered by a firmware assert with Recover Flow Request (RFR) only. Firmware RFR support should be enabled explicitly using mlxconfig commands.

➤ **To query the current value, run:**

```
mlxconfig -d /dev/mst/mt4115_pciconf0 query | grep  
SW_RECOVERY_ON_ERRORS
```

➤ **To enable RFR bit support, run:**

```
mlxconfig -d /dev/mst/mt4115_pciconf0 set  
SW_RECOVERY_ON_ERRORS=true
```

## Kernel ULPs

Once a "fatal device" error is recognized, an IB\_EVENT\_DEVICE\_FATAL event is created, ULPs are notified about the incident, and outstanding WQEs are simulated to be returned with "flush in error" message to enable each ULP to close its resources and not get stuck via calling its "remove\_one" callback as part of "Reset Flow".

Once the unload part is terminated, each ULP is called with its "add\_one" callback, its resources are re-initialized and it is re-activated.

## User Space Applications (IB/RoCE)

Once a "fatal device" error is recognized an IB\_EVENT\_DEVICE\_FATAL event is created, applications are notified about the incident and relevant recovery actions are taken.

Applications that ignore this event enter a zombie state, where each command sent to the kernel is returned with an error, and no completion on outstanding WQEs is expected.

The expected behavior from the applications is to register to receive such events and recover once the above event is raised. Same behavior is expected in case the NIC is unbounded from the PCI and later is rebounded. Applications running over RDMA CM should behave in the same manner once the RDMA\_CM\_EVENT\_DEVICE\_REMOVAL event is raised.

The below is an example of using the unbind/bind for NIC defined by "0000:04:00.0"

```
echo 0000:04:00.0 > /sys/bus/pci/drivers/mlx5_core/unbind  
echo 0000:04:00.0 > /sys/bus/pci/drivers/mlx5_core/bind
```

## SR-IOV

If the Physical Function recognizes the error, it notifies all the VFs about it by marking their communication channel with that information, consequently, all the VFs and the PF are reset.

If the VF encounters an error, only that VF is reset, whereas the PF and other VFs continue to work unaffected.

## Forcing the VF to Reset

If an outside "reset" is forced by using the PCI sysfs entry for a VF, a reset is executed on that VF once it runs any command over its communication channel.

For example, the below command can be used on a hypervisor to reset a VF defined by 0000:04:00.1:

```
echo 1 > /sys/bus/pci/devices/0000:04:00.1/reset
```

## Extended Error Handling (EEH)

Extended Error Handling (EEH) is a PowerPC mechanism that encapsulates AER, thus exposing AER events to the operating system as EEH events.

The behavior of ULPs and user space applications is identical to the behavior of AER.

# CRDUMP

CRDUMP feature allows for taking an automatic snapshot of the device CR-Space in case the device's FW/HW fails to function properly.

## Snapshots Triggers:

The snapshot is triggered after firmware detects a critical issue, requiring a recovery flow.

This snapshot can later be investigated and analyzed to track the root cause of the failure.

Currently, only the first snapshot is stored, and is exposed using a temporary virtual file. The virtual file is cleared upon driver reset.

When a critical event is detected, a message indicating CRDUMP collection will be printed to the Linux log. User should then back up the file pointed to in the printed message. The file location format is: `/proc/driver/mlx5_core/crdump/<pci address>`

Snapshot should be copied by Linux standard tool for future investigation.

## Firmware Tracer

This mechanism allows for the device's FW/HW to log important events into the event tracing system (`/sys/kernel/debug/tracing`) without requiring any NVIDIA tool.

### Note

To be able to use this feature, trace points must be enabled in the kernel.

This feature is enabled by default, and can be controlled using sysfs commands.

### **To disable the feature:**

```
echo 0 > /sys/kernel/debug/tracing/events/mlx5/fw_tracer/enable
```

### **To enable the feature:**

```
echo 1 > /sys/kernel/debug/tracing/events/mlx5/fw_tracer/enable
```

### **To view FW traces using vim text editor:**

```
vim /sys/kernel/debug/tracing/trace
```

#### **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 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 2025, NVIDIA. PDF Generated on 06/04/2025