Reset Flow

Table of contents

List of Figures

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.

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
```