Codensity T408™
Crash Auto Recovery Application Note

Reference Number: 20TD003-02-01
09/02/2020

Table of Contents

1 Table of Abbreviations

Abbreviation:	Full Form:
FLR	function level reset
DUT	Device Under Test
NVMe	Non-volatile Memory express (Transport protocol)

2 Background

If the T408 Transcoder were ever to enter a crashed state (FW assertions, CPU exceptions, dead-loops, etc.) there is no non-intrusive method to recover the device. Typical solutions would be to power-cycle the host machine, or hot-swap the device. This step would either negatively affect overall host functions or require manual interference to recover the transcoder which is unsuitable for large-scale operations.

In order to recover the crashed device, a top-priority CPU interrupt was set in the FW such that if the host software issues a PCIe function level reset (FLR) to the PCIe address pertaining to the crashed T408 , the interrupt received by the FW will trigger internal hardware registers to issue a DUT reset. In the event of a DUT reset, the device will behave as if it were hot-swapped and the net effect is a recovered device. The remaining step the host needs to execute after an FLR is to simply rescan the PCIe bus.

2.1 Intended Audience

This document is intended to help engineers/technicians/developers operating NETINT transcoders troubleshoot common problems.

2.2 Compatibility

Software Compatibility

This guide is intended to be used with NETINT Codensity T408 Video Transcoder software Release after V1.6.o.

Hardware Compatibility

NETINT Codensity T408 Video Transcoder hardware.

Operating system

Any linux operating system.

3 Theory

In this example scenario, our T408 card [/dev/nvme0n1] has unfortunately hit an error condition such as division by zero and is stuck in a CPU exception. There are no responses returning from the NVMe drivers so any attempt to ping the T408 is ineffective. Host software has attempted an NVMe shutdown and restart but even the shutdown command hangs, and the restart brings nothing up.

The host software at this point may attempt to use the FLR to reboot the device.

Find the PCIe bus, slot, and function number of the offending card:

lspci -n | grep 1d82

1. Output: 86:00.0 0108: 1d82:0202
2. 86:00.0 corresponds to <bus>:<slot>.<func>
3. 1d82 is an identifier for T408 devices
4. '–n' parameter is to list such identifiers

If there are numerous T408 cards in the same host, several different addresses may appear, in that case since we know our device ID is **/dev/nvme0n1**, we can cross reference with our previous lspci output like so:

1. ls -l /sys/class/block | grep nvme
2. output:

```
[nvme@CLI216 ~]$ ls -l /sys/class/block | grep nvme
lrwxrwxrwx 1 root root 0 Feb  4 11:37 nvme0n1 -> ../../devices/pci0000:85/0000:85:00.0/0000:86:00.0/nvme/nvme0/nvme0n1
lrwxrwxrwx 1 root root 0 Feb  4 11:37 nvme1n1 -> ../../devices/pci0000:85/0000:85:01.0/0000:87:00.0/nvme/nvme1/nvme1n1
lrwxrwxrwx 1 root root 0 Feb  4 11:37 nvme2n1 -> ../../devices/pci0000:85/0000:85:02.0/0000:88:00.0/nvme/nvme2/nvme2n1
lrwxrwxrwx 1 root root 0 Feb  4 11:37 nvme3n1 -> ../../devices/pci0000:85/0000:85:03.0/0000:89:00.0/nvme/nvme3/nvme3n1
lrwxrwxrwx 1 root root 0 Feb  4 11:37 nvme4n1 -> ../../devices/pci0000:ae/0000:ae:00.0/0000:af:00.0/nvme/nvme4/nvme4n1
lrwxrwxrwx 1 root root 0 Feb  4 11:37 nvme5n1 -> ../../devices/pci0000:ae/0000:ae:01.0/0000:b0:00.0/nvme/nvme5/nvme5n1
```

Figure 1 multiple T408 cards

3. The PCIe bus, slot, and function number can once again be confirmed.

With the PCIe address known, FLR can now be issued in a read-modify-write:

sudo setpci -s 86:00.0 CAP_EXP+8.l

1. The '-s' parameter is to specify the PCIe bus, slot, and function number.
2. CAP_EXP+8.l is to specify the register to read
3. Output: 0009283e

Only bit[15] matters and we want to set it to '1', 0x9283e AND 0x8000=0x9a83e

Now send the FLR by writing to bit[15]:

sudo setpci -s 86:00.0 CAP_EXP+8.l=9a83e

At this point T408 FW will have received the interrupt and reset itself, host need only rescan for PCIe devices:

sudo sh -c "echo 1 > /sys/bus/pci/rescan"

Device is now ready for use.

User can use : “sudo nvme list” which should list the recovered card.

NOTE: When using the PCIe reset/rescan function, please ensure there is no background process (e.g., ffmpeg) sending any traffic to the T408 card.

NOTE: After the PCIe reset / rescan process, the NVME device ID for T408 could be changed from the original value before reset.

4 Testing

Using a custom version of firmware designed to hang on a given command, the T408 Transcoder can be put into a hanging state by issuing a nvme format command to the card:

```
sudo nvme format /dev/nvme0 -n 1 -l 0
```

After the transcoder is in a hanging state the device can be recovered by triggering a DUT reset through issuing a PCIe reset to the transcoder’s PCIe address likewise in Example Usage section:

1. lspci -n | grep 1d82

Output: 02:00.0 0108: 1d82:0202

2. sudo setpci -v -s 02:00.0 CAP_EXP+8.l

Output: 0000:02:00.0 (cap 10 @c0) @c8 = 0009283f

3. sudo setpci -v -s 02:00.0 CAP_EXP+8.l=9a83f

Output: 0000:02:00.0 (cap 10 @c0) @c8 0009283f

After issuing the PCIe reset the transcoder disappears from the list of PCIe devices. The device can then be re-linked by issuing a PCIe rescan:

```
sudo sh -c "echo 1 > /sys/bus/pci/rescan"
```

After the card has been recovered, it is able to perform transcoding of 2x4K, or 6x1080p, or 32x360p H264-to-H265 and H265-to-H265 instances without any issues.

5 Possible Side-effects or Shortcomings

- FLR to the T408 transcoder will not gracefully exit the transcoding operations.
- Not all hosts support PCIe rescan and will still require power-cycle to reset device.

6 Revision History

Version	Change	Create Date	Author
1.0	Document Created	Feb 05,2020	Tom Han

7 Legal Notice