White Paper
May 2022

Enhanced Downstream Port Containment Enablement for Hot-Plug

Deploying a successful use case using Tencent on Intel Enhanced Downstream Port Containment Enablement technology to handle PCI Express* Hot-Plug

1. Introduction

1.1. Background

In today's hyper scale Cloud Datacenters, a successful management of PCI Express* (PCIe*) devices hot-plug is a mandatory requirement to ensure for server maintenance, and debug. No server crash is allowed during the operation of Hot-Plug devices.

Customer used to take Intel® Volume Management Device (Intel® VMD) technology or Host Bus Controller (HBC) to manage PCIe device hot-plug event to ensure a smooth hot-plug transition, but both get some side effects such as no pass-through support on the granularity of disk level with Intel® VMD and extra Bills of Materials (BOM) cost adder with HBC. The other approach is to develop an OS level solution (driver, patch) to handle the demand based on a Hot-Plug Surprise function in BIOS, but it takes lots of development efforts and faces the coexistence issue with Enhanced Downstream Port Containment Enablement (EDPC). Besides, system crash and/or error ignorance are the downsides with the Hot-Plug Surprise mechanism, and Hot-Plug Surprise cannot probably handle the outstanding transitions, and thus, makes it deprecated for async removal.

Therefore, customers are seeking any other alternative solution which is convenient and easy to deploy and scale. Downstream Port Containment (DPC) is an optional normative feature of a Downstream Port. DPC halts PCIe traffic below a Downstream Port after an unmasked uncorrectable error is detected at or below the Port, avoiding the potential spread of any data corruption, and supporting Containment Error Recovery (CER) if implemented by software. EDPC is an enhancement to the DPC by adding Root Port Programmable I/O (RP PIO) errors. Customers use Hot-Plug Surprise as a traditional method for NVMe* hot-plugging method. PCI-SIG introduced DPC as the preferred mechanism for handling surprise hot removal events. The Transaction Layer Packet (TLP) stream cleanly stops upon an uncorrectable error that triggers DPC. Operating System/driver stacks that supports Containment Error Recovery (CER) can fully and transparently recover from many transient PCIe uncorrectable errors. DPC can support async removal and CER concurrently.

This document helps customer to enable the EDPC to replace the Hot-Plug Surprise for a robust error containment mechanism that can be used to manage asynchronous removal events.

1.2. Coverage of this Document

This paper is more focused on the discussion of how to leverage EDPC Reliability, Availability, Serviceability (RAS) to handle PCIe hot-plug devices. It covers several sessions:

- Basic introduction of EDPC and its workflow under different working models
- Validation Recipe for EDPC
- Client Strategic Planning Customer case study on 3rd Generation Intel® Xeon® Scalable processor servers
- Outlook for future work

1.3. Target User of this Document

The target audience for this document are users who already have familiarity with Intel® Xeon® RAS technologies:

- System engineers
- RAS/BIOS engineers
- Operative System Kernel engineers
- Platform architects/RAS architects

2. Feature Description

The target of EDPC is to improve containment within the PCIe sub-system when an unmasked uncorrectable error is detected either at the Root Port (RP) or at the Switch Downstream Port. When there is an unmasked uncorrectable error detected, DPC halts PCIe traffic below a downstream port and avoiding the potential spread of data corruption and supporting containment error recovery if implemented by software.

Description

The EDPC Extended Capability is an optional normative capability that provides a mechanism for Downstream Ports to contain uncorrectable errors and enable software to recover from them. After triggering EDPC, downstream port performs following actions:

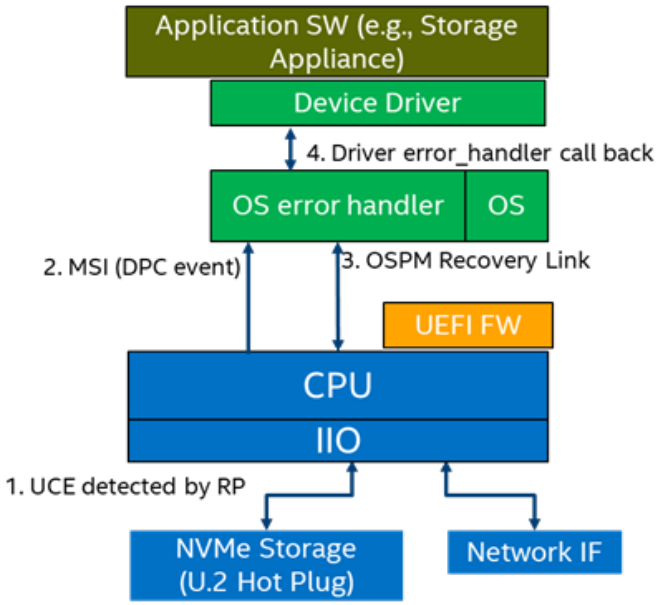
- Sets the DPC trigger status bit. Populates the DPC trigger reason field (unmasked Uncorrectable Error (UCE), ERR_FATAL, ERR_NONFATAL, Root Port Programmed I/O (RP PIO), or software (SW) triggered)
- Disables the link. Once it is in this state, it will remain so until the DPC trigger status bit is cleared
- Drops posted and returns all 1's for non-posted transactions. Gracefully handling pending transactions
- If "PCIe Hot-Plug Surprise" feature is enabled, then "surprise link down" error will not trigger DPC. It will block the reporting of surprise down error with Hot-Plug Surprise bit set. When there is a port associated with a hot-pluggable slot and Hot-Plug Surprise bit in the slot capabilities is set, then any transition from DL_Active to DL_Inactive must not be considered as an error

Enabling Considerations

- Unified Extensible Firmware Interface (UEFI) firmware (FW) needs to configure the EDPC error source registers during the boot time
- Run time error handler is required to manage the EDPC event. In case of OS Native mode, OS performs the error handling. System recovery post-EDPC event requires additional customer specific support solution to dump the error data

2.1. Linux* EDPC Workflow

Figure 1-1. EDPC aware Native OS (Linux*) Mode



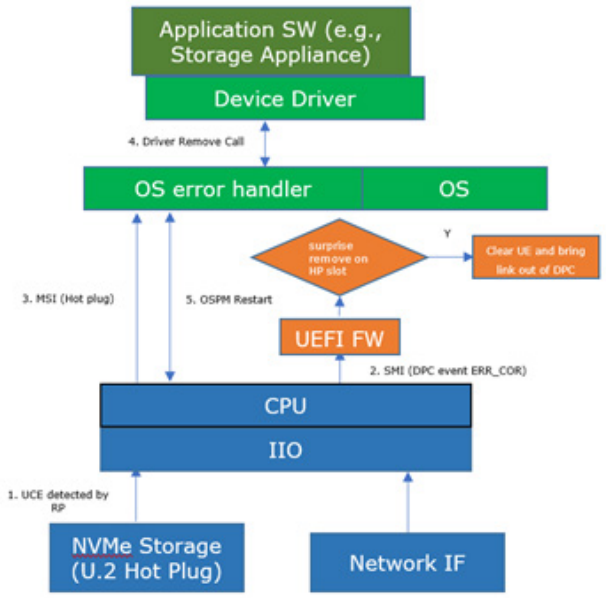
Boot time:

1. UEFI FW optionally enables EDPC prior to handing over control to OS.
2. Operating System Power Management (OSPM) requests and gets control of EDPC, Advanced Error Report (AER), and hot-plug.
3. OSPM initializes AER, DPC, and hot-plug registers. Enables EDPC interrupt.

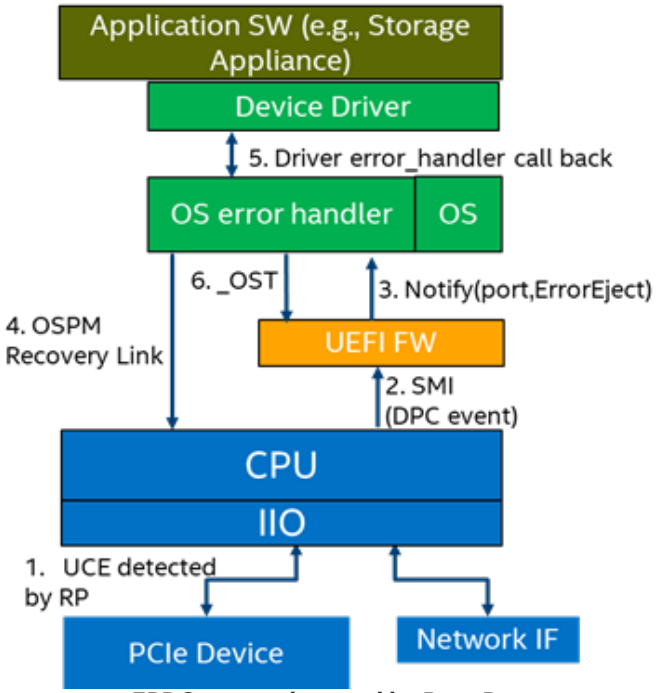
Run time:

1. Uncorrected error detected by RP (for example, Malformed TLP). SW triggered DPC could also be used for validation purpose.
2. RP sends Message Signal Interrupt (MSI) to OS Error Handler. OS handler detects DPC event. DPC Status and Error Source ID are logged; RP PIO status is logged if applicable.
3. If port implements DPC capabilities, OSPM will attempt recovery by releasing the link from DPC, link retraining and active, or restore child devices to working state.
4. OSPM notifies drivers of child devices of port, the pci_driver via pci_driver's.err_handler callback: error_detected, slot_reset, error_resume.

Figure 1-2. Firmware First Mode (FFM) with EDPC aware OS



Hot-Plug PCIe device with EDPC FFM enable



EPDC events detected by Root Port

Boot time:

1. UEFI FW may initialize EDPC registers or leave them disabled.

   Hot-Plug Surprise = Disabled if EDPC is enabled.

2. OSPM requests ownership of EDPC+ AER+ hot-plug. FW grants ownership of hot-plug and keeps ownership of AER and EDPC.
3. OPSM does not initialize AER and EDPC. It sets up hot-plug registers. Enables EDPC interrupt. FW may enable EDPC once it receives a hint from OSPM via _DSM.

Run time:

1. Uncorrected error detected by RP (for example, Malformed TLP). EDPC Status and Error Source ID are logged; RP PIO status is logged if applicable.
2. RP sends MSI (or system control interrupt (SCI)) to UEFI FW. FW detects EDPC event and reads AER, EDPC registers. Creates system event log and updates common platform error record tables.
3. If it is Hot-Plug Surprise, remove event on slot with hot-plug capability, FW clears up UCE status and brings the link out of DPC and then Hot-Plug Surprise remove interrupt will deliver to OS. Otherwise, SCI handler issue Notify (port, ErrorEject).
4. If port implements DPC capabilities, OSPM will attempt recovery by releasing the link from DPC, link re-training and active, or restore child devices to working state.
5. OSPM notifies drivers of child devices of port, the pci_ driver via pci_driver's.err_handler callback: error_detected, slot_reset, error_resume.
6. OSPM calls _OST associated with the port to communicate status. FW reads AER status registers from the device and logs them. Other proprietary FW steps are possible.

Refer to: https://members.pcisig.com/wg/PCI-SIG/document/previewpdf/15350 PCI_Firmware_v3.3_20210120_NCB.pdf chapter 4.6.12 _DSM for Downstream Port Containment and Hot-Plug Surprise Control.

3. EDPC Validation Recipe

3.1. Hardware and Firmware Configuration

1. Wilson City CRB System with 3rd Generation Intel® Xeon® Scalable processor latest stepping.
2. Whitley Latest Best-Known Configuration.
3. System with at least one PCIe NVMe SSD plugged in.

3.2. Native OS Mode

3.2.1. UEFI-FW Setup

- EDKII Menu-> Platform Configuration -> System Event Log -> IIO Error Enabling -> OS Native AER Support = Enable
- EDKII Menu-> Platform Configuration -> System Event Log -> IIO Error Enabling -> IIO EDPC Support = On Fatal and Non-Fatal Errors
- EDKII Menu-> Platform Configuration -> System Event Log -> IIO Error Enabling -> IIO EDPC Interrupt = Enable
- EDKII Menu-> Platform Configuration -> System Event Log -> IIO Error Enabling -> IIO EDPC ERR_COR Message = Enable
- EDKII Menu-> Platform Configuration -> System Event Log -> PCIe Error Enabling -> PCIE Unsupported Request Error Enable
- EDKII Menu-> Platform Configuration -> System Event Log -> PCIe Error Enabling -> PCIE Surprise Link Down Error = Enable

- EDKII Menu-> Socket Configuration -> IIO Configuration -> PCIe Hot-Plug = Yes
- EDKII Menu-> Socket Configuration -> IIO Configuration -> PCIe ACPI Hot-Plug = No
- EDKII Menu-> Socket Configuration -> IIO Configuration ->Socket0 Configuration->Port 1A->Surprise hot-plug Capable = Disable

And, ensure Hot-Plug Surprise Capable<Disable> in all other PCIe port settings of Socket0 and Socket1.

3.2.2. Kernel Version and Configuration

OS environment: CentOS8.3

- V5.14.2 or v5.4.49 with backport patches

Kernel configuration:

- CONFIG_PCIE_DPC=y

EDPC Native OS (Linux) Mode is required to add pcie_ports = native to Kernel grub.

3.2.3. Validate with Native OS Mode

3.2.3.1. NVMe Basic Information Check

After system booted up, run following command to ensure NVMe SSD can be found in this system, that is, NVMe SSD is found at e6:00.0 and connected with Root Port e2:05.0.

[root@localhost ~]# lspci -vv | grep -i nvm

e6:00.0 Non-Volatile memory controller: Intel Corporation Device 0b60 (prog-if 02 [NVM Express])

...

[root@localhost ~]# lspci -vt

...

+-[0000:e2]-+-00.0 Intel Corporation Device 09a2

| +-00.1 Intel Corporation Device 09a4

| +-00.2 Intel Corporation Device 09a3

| +-00.3 Intel Corporation Device 09a5

| +-00.4 Intel Corporation Device 0998

| +-02.0-[e3]--

| +-03.0-[e4]--

| +-04.0-[e5]--

| \-05.0-[e6]----00.0 Intel Corporation Device 0b60

...

3.2.3.2. Check Slot Status with Hot-Plug

Unplug, then, plug several times to check if the value of slot status register is correct.

Ensure the value of bit 5 (Hot-Plug Surprise) is 0 in Slot Capabilities register

[root@localhost ~]# setpci -s e2:05.0 0x54.l

00880cd8

The value of bit 3 (Presence Detect Changed Enable) is 1 in Slot control register

[root@localhost ~]# setpci -s e2:05.0 0x58.w

11f8

Slot Status register bit 6 (Presence Detect State) is 1

[root@localhost ~]# setpci -s e2:05.0 0x5a.w

0040

Unplugged NVMe SSD, Slot Status register bit 6 (Presence Detect State) changed to 0

[root@localhost ~]# setpci -s e2:05.0 0x5a.w

0000

3.2.3.3. Check Mount Operation

Assume NVMe SSD is already formatted as ext4 file system, mount with below command:

[root@localhost ~]# mount -t ext4 /dev/nvme0n1p1 /media/nvme/

[root@localhost ~]# ls /media/nvme/

test

[root@localhost ~]# umount /media/nvme

Test NVMe SSD by plugging and unplugging it more times, the value of Slot Status register will change to 0040 when NVMe SSD is plugged in. Unplug it again, and the value will change to 0000. System will not crash, and mount operation is successful.

3.2.3.4. Hot-Plug Test during Read Operation

Unplug SSD during data read procedure for several times to check if the data transaction still works normally.

That is,

[root@localhost ~]# dd if=/dev/nvme0n1 of=/home/rd_4g-1 bs=1M count=4M

<- in the middle of reading, unplugged the SSD disk, and then hot-plug it back.

1416+1 records in

1416+1 records out

1485045760 bytes (1.5 GB, 1.4 GiB) copied, 3.49759 s, 425 MB/s

[root@localhost ~]# dd if=/dev/nvme0n1 of=/home/rd_4g-1 bs=1M count=4M

<- in the middle of reading, unplugged the SSD disk

3019+1 records in

3019+1 records out

3166437376 bytes (3.2 GB, 2.9 GiB) copied, 7.0216 s, 451 MB/s

#dmesg

[962.032560] pcieport 0000:e2:05.0: AER: Corrected error received: 0000:e2:05.0

[962.039826] pcieport 0000:e2:05.0: AER: PCIe Bus Error: severity=Corrected, type=Physical Layer, (Receiver ID)

[962.049834] pcieport 0000:e2:05.0: AER: device [8086:347d] error status/mask=00000001/00000000

[962.058632] pcieport 0000:e2:05.0: AER: [0] RxErr

[962.106900] pcieport 0000:e2:05.0: pciehp: pending interrupts 0x0108 from Slot Status

[962.114767] pcieport 0000:e2:05.0: pciehp: Slot(17): Link Down

[962.120614] pcieport 0000:e2:05.0: pciehp: Slot(17): Card not present

[962.127059] pcieport 0000:e2:05.0: pciehp: pciehp_unconfigure_device: domain:bus:dev = 0000:e6:00

[962.135935] nvme 0000:e6:00.0: PME# disabled

[962.140248] pcieport 0000:e2:05.0: DPC: containment event, status:0x1f01 source:0x0000

[962.148173] pcieport 0000:e2:05.0: DPC: unmasked uncorrectable error detected

[962.155320] pcieport 0000:e2:05.0: AER: PCIe Bus Error: severity=Uncorrected (Fatal), type=Transaction Layer, (Receiver ID)

[962.166443] pcieport 0000:e2:05.0: AER: device [8086:347d] error status/mask=00000020/00000000

[962.175231] pcieport 0000:e2:05.0: AER: [5] SDES (First)

[962.254376] pcieport 0000:e2:05.0: pciehp: pciehp_set_indicators: SLOTCTRL 58 write cmd 300

[962.262737] pcieport 0000:e2:05.0: pciehp: pending interrupts 0x0010 from Slot Status

[962.271743] pcieport 0000:e2:05.0: pciehp: pciehp_check_link_active: lnk_status = d041

[963.478920] pcieport 0000:e2:05.0: Data Link Layer Link Active not set in 1000 msec

[963.486590] pcieport 0000:e2:05.0: link reset at upstream device 0000:e2:05.0 failed

[963.494385] pcieport 0000:e2:05.0: AER: Device recovery failed

[1002.999272] pcieport 0000:e2:05.0: pciehp: pending interrupts 0x0008 from Slot Status

[1003.007223] pcieport 0000:e2:05.0: pciehp: pciehp_check_link_active: lnk_status = d041

[1003.015165] pcieport 0000:e2:05.0: pciehp: Slot(17): Card present

[1003.021271] pcieport 0000:e2:05.0: pciehp: pciehp_set_indicators: SLOTCTRL 58 write cmd 200

[1003.029630] pcieport 0000:e2:05.0: pciehp: pending interrupts 0x0010 from Slot Status

[1003.512257] pcieport 0000:e2:05.0: pciehp: pending interrupts 0x0100 from Slot Status

[1003.622906] pcieport 0000:e2:05.0: pciehp: pciehp_check_link_status: lnk_status = f044

3.3. Firmware First Mode

3.3.1. UEFI-FW Setup

Setup the environment and configure the settings

- EDKII -> Platform Configuration -> System Event Log -> IIO Error Enabling -> **OS Native AER Support = Disable**
- EDKII -> Platform Configuration -> System Event Log -> IIO Error Enabling -> IIO EDPC Support = On Fatal and Non-Fatal Errors
- EDKII -> Platform Configuration -> System Event Log -> IIO Error Enabling -> IIO EDPC Support = IIO EDPC ERR_COR Message = Enable

3.3.2. Kernel Version and Configuration

OS environment: CentOS8.3

- V5.14.2 or v5.4.49 with backport patches

Kernel configuration:

- CONFIG_PCIE_DPC=y
- CONFIG_PCIE_EDR=y

3.3.3. Validate with Firmware First Mode

3.3.3.1. NVMe Basic Information Check

After system booted up, run following command to ensure NVMe SSD can be found in this system, that is, NVMe SSD is found at 65:00.0 and connected with Root Port 64:02.0.

[root@localhost ~]# lspci -vv | grep -i nvm

65:00.0 Non-Volatile memory controller: Intel Corporation NVMe Datacenter SSD [3DNAND, Beta Rock Controller] (prog-if 02 [NVM Express])

Subsystem: Intel Corporation NVMe Datacenter SSD [3DNAND] SE 2.5" U.2 (P4510)

Kernel driver in use: NVMe

Kernel modules: NVMe

+-[0000:64]-+-00.0 Intel Corporation Device 09a2

| +-00.1 Intel Corporation Device 09a4

| +-00.2 Intel Corporation Device 09a3

| +-00.3 Intel Corporation Device 09a5

| +-00.4 Intel Corporation Device 0998

| +-02.0-[65]----00.0 Intel Corporation NVMe Datacenter SSD [3DNAND, Beta Rock Controller]

| +-03.0-[66]--

| +-04.0-[67]--

| \-05.0-[68]—

3.3.3.2. Hot-Plug Test

1. Boot to the OS and check the NVMe device is recognized successfully.
2. Unplug the NVMe device, and hot-plug it back.
3. Capture the MSI handler serial log, there should be a surprise link down error and EDPC triggered.

.....

CreatePcieErrorRecord Entry,Bus: 100,Device: 2,Function: 0,ErrorType: 1

PcieAerGetUncErrSts UncErrSts = 0x20

PcieAerGetCorrErrSts CorrErrSts = 0x1

GetPcieRootErrSts RpErrSts = 0x3

PcieAerGetUncErrMsk UncErrMsk = 0x100020

PcieAerGetCorrErrMsk CorrErrMsk = 0x0

CreatePcieErrorRecord, ErrType = 2

.....

PcieClearDeviceStatus Socket 0, Bus 64, Dev 2, Func0

Clearing Aer correct status on Bus:0x64, Dev:0x2 func:0x0 CorErrSts:0x1

After Clearing Aer correct status on Bus:0x64, Dev:0x2 func:0x0 CorErrSts:0x0

[Helper] PcieClearDeviceStatus RegTmp16 = 0x9

PcieIsEdpcTiggered PCIE_DPC_OFFSET_REG_STS = 0x1F00

PcieIsEdpcTiggered PCIE_DPC_OFFSET_REG_CTL = 0x8

PcieIsEdpcTiggered PCIE_DEVICE_CONTROL = 0x5147

[AER] Input UncErrMsk 0x100020

[AER] Input CorrErrMsk 0x2000

[AER] Input UncErrMsk 0x100020

[AER] Input CorrErrMsk 0x0

.....

Plug the NVMe back to the slot, the NVMe can be recognized successfully.

Check the dmesg after that:

[1210.643340] {1}[Hardware Error]: Hardware error from APEI Generic Hardware Error Source: 0

[1210.643343] {1}[Hardware Error]: It has been corrected by h/w and requires no further action

[1210.643344] {1}[Hardware Error]: event severity: corrected

[1210.643345] {1}[Hardware Error]: Error 0, type: corrected

[1210.643347] {1}[Hardware Error]: section_type: PCIe error

[1210.643347] {1}[Hardware Error]: port_type: 4, root port

[1210.643348] {1}[Hardware Error]: version: 3.0

[1210.643348] {1}[Hardware Error]: command: 0x0547, status: 0x0010

[1210.643349] {1}[Hardware Error]: device_id: 0000:64:02.0

[1210.643350] {1}[Hardware Error]: slot: 10

[1210.643351] {1}[Hardware Error]: secondary_bus: 0x65

[1210.643351] {1}[Hardware Error]: vendor_id: 0x8086, device_id: 0x347a

[1210.643352] {1}[Hardware Error]: class_code: 060400

[1210.643353] {1}[Hardware Error]: bridge: secondary_ status: 0x2000, control: 0x0003

[1210.643385] pcieport 0000:64:02.0: AER: aer_status: 0x00000001, aer_mask: 0x00000000

[1210.643443] pcieport 0000:64:02.0: pciehp: Slot(10): Link Down

[1210.643906] pcieport 0000:64:02.0: pciehp: Slot(10): Card not present

[1210.643906] pcieport 0000:64:02.0: [0] RxErr (First)

[1210.643909] pcieport 0000:64:02.0: AER: aer_ layer=Physical Layer, aer_agent=Receiver ID

[1210.670304] nvme0n1: detected capacity change from 1953525168 to 0

-> Hot-plug back the NVMe device:

[1216.897623] pcieport 0000:64:02.0: pciehp: Slot(10): Card present

[1217.681027] pci 0000:65:00.0: [8086:0a54] type 00 class 0x010802

[1217.681056] pci 0000:65:00.0: reg 0x10: [mem 0x00000000-0x00003fff 64bit]

[1217.681092] pci 0000:65:00.0: reg 0x30: [mem 0x00000000-0x0000ffff pref]

[1217.681103] pci 0000:65:00.0: Max Payload Size set to 512 (was 128, max 512)

[1217.681111] pci 0000:65:00.0: enabling Extended Tags

[1217.681478] pci 0000:65:00.0: BAR 6: assigned [mem 0xbbc00000-0xbbc0ffff pref]

[1217.681486] pci 0000:65:00.0: BAR 0: assigned [mem 0xbbc10000-0xbbc13fff 64bit]

[1217.681500] pcieport 0000:64:02.0: PCI bridge to [bus 65]

[1217.681506] pcieport 0000:64:02.0: bridge window [io 0x6000-0x6fff]

[1217.681513] pcieport 0000:64:02.0: bridge window [mem 0xbbc00000-0xbbcfffff]

[1217.681519] pcieport 0000:64:02.0: bridge window [mem 0x204000000000-0x2040001fffff 64bit pref]

[1217.681902] nvme nvme0: pci function 0000:65:00.0

[1217.681918] nvme 0000:65:00.0: enabling device (0140 -> 0142)

[1219.016664] nvme nvme0: 128/0/0 default/read/poll queues

3.3.3.3. Hot-Plug Test during Read Operation

1. Boot to the OS and check the NVMe device is recognized successfully.
2. Try to do data read from the NVMe disk and then unplug the NVMe device during data reading, plug it back to check if the data reading is continued.

[root@localhost dev]# dd if=/dev/nvme0n1 of=/home/test/test bs=1MB count=4MB

<- in the middle of reading, unplugged the SSD disk, and then hot-plug it back.

43842+0 records in

43842+0 records out

43842000000 bytes (44 GB, 41 GiB) copied, 421.119 s, 104 MB/s

3. Capture the MSI handler serial log, there should be a surprise link down error and EDPC triggered.

.....

CreatePcieErrorRecord Entry,Bus: 100,Device: 2,Function: 0,ErrorType: 1

PcieAerGetUncErrSts UncErrSts = 0x20

PcieAerGetCorrErrSts CorrErrSts = 0x1

GetPcieRootErrSts RpErrSts = 0x3

PcieAerGetUncErrMsk UncErrMsk = 0x100020

PcieAerGetCorrErrMsk CorrErrMsk = 0x0

CreatePcieErrorRecord, ErrType = 2

.....

PcieClearDeviceStatus Socket 0, Bus 64, Dev 2, Func0

Clearing Aer correct status on Bus:0x64, Dev:0x2 func:0x0 CorErrSts:0x1

After Clearing Aer correct status on Bus:0x64, Dev:0x2 func:0x0 CorErrSts:0x0

[Helper] PcieClearDeviceStatus RegTmp16 = 0x9

PcieIsEdpcTiggered PCIE_DPC_OFFSET_REG_STS = 0x1F00

PcieIsEdpcTiggered PCIE_DPC_OFFSET_REG_CTL = 0x8

PcieIsEdpcTiggered PCIE_DEVICE_CONTROL = 0x5147

[AER] Input UncErrMsk 0x100020

[AER] Input CorrErrMsk 0x2000

[AER] Input UncErrMsk 0x100020

[AER] Input CorrErrMsk 0x0

.....

4. Plug the NVMe back to the slot, the NVMe can be recognized successfully.

Check the dmesg after that:

[11398.373917] {2}[Hardware Error]: Hardware error from APEI Generic Hardware Error Source: 0

[11398.373920] {2}[Hardware Error]: It has been corrected by h/w and requires no further action

[11398.373921] {2}[Hardware Error]: event severity: corrected

[11398.373922] {2}[Hardware Error]: Error 0, type: corrected

[11398.373924] {2}[Hardware Error]: section_type: PCIe error

[11398.373924] {2}[Hardware Error]: port_type: 4, root port

[11398.373925] {2}[Hardware Error]: version: 3.0

[11398.373926] {2}[Hardware Error]: command: 0x0547, status: 0x0010

[11398.373927] {2}[Hardware Error]: device_id: 0000:64:02.0

[11398.373928] {2}[Hardware Error]: slot: 10

[11398.373929] {2}[Hardware Error]: secondary_bus: 0x65

[11398.373929] {2}[Hardware Error]: vendor_id: 0x8086, device_id: 0x347a

[11398.373930] {2}[Hardware Error]: class_code: 060400

[11398.373930] {2}[Hardware Error]: bridge: secondary_ status: 0x2000, control: 0x0003

[11398.373963] pcieport 0000:64:02.0: AER: aer_status: 0x00000001, aer_mask: 0x00000000

[11398.374008] pcieport 0000:64:02.0: pciehp: Slot(10): Link Down

[11398.374473] pcieport 0000:64:02.0: [0] RxErr (First)

[11398.374474] pcieport 0000:64:02.0: pciehp: Slot(10): Card not present

[11398.374476] pcieport 0000:64:02.0: AER: aer_ layer=Physical Layer, aer_agent=Receiver ID

[11398.400209] nvme0n1: detected capacity change from 1953525168 to 0

[11399.358776] sched: RT throttling activated

[11400.662119] pcieport 0000:64:02.0: pciehp: Slot(10): Card present

[11401.501846] pci 0000:65:00.0: [8086:0a54] type 00 class 0x010802

[11401.501875] pci 0000:65:00.0: reg 0x10: [mem 0x00000000-0x00003fff 64bit]

[11401.501911] pci 0000:65:00.0: reg 0x30: [mem 0x00000000-0x0000ffff pref]

[11401.501922] pci 0000:65:00.0: Max Payload Size set to 512 (was 128, max 512)

[11401.501929] pci 0000:65:00.0: enabling Extended Tags

[11401.502302] pci 0000:65:00.0: BAR 6: assigned [mem 0xbbc00000-0xbbc0ffff pref]

[11401.502311] pci 0000:65:00.0: BAR 0: assigned [mem 0xbbc10000-0xbbc13fff 64bit]

[11401.502324] pcieport 0000:64:02.0: PCI bridge to [bus 65]

[11401.502329] pcieport 0000:64:02.0:   bridge window [io 0x6000-0x6fff]

[11401.502337] pcieport 0000:64:02.0:   bridge window [mem 0xbbc00000-0xbbcfffff]

[11401.502342] pcieport 0000:64:02.0:   bridge window [mem 0x204000000000-0x2040001fffff 64bit pref]

[11401.502676] nvme nvme1: pci function 0000:65:00.0

[11401.502751] nvme 0000:65:00.0: enabling device (0140 -> 0142)

[11402.807580] nvme nvme1: 128/0/0 default/read/poll queues

3.3.3.4. Error Injection on NVMe Device

1. Set the UEFI and Kernel as previously mentioned.

2. Check the DPC capabilities.

[root@localhost ~]# lspci -vvv -s 64:02.0

.....

Capabilities: [190 v1] Downstream Port Containment

DpcCap: INT Msg #0, RPExt+ PoisonedTLP+ SwTrigger+ RP PIO Log 4, DL_ActiveErr+

DpcCtl: Trigger:2 Cmpl- INT+ ErrCor+ PoisonedTLP- SwTrigger- DL_ActiveErr-

DpcSta: Trigger- Reason:00 INT+ RPBusy- TriggerExt:00 RP PIO ErrPtr:1f

Source: 0000

3. Trigger DPC flow by CScripts with completion timeout error injection.

>>>ei.injectPcieError(0,port='pxp3.port0',errType='uce')

4. Check the serial log after the error injection.

.....

[Device Error] error on skt:0x0 Bus:0x64 Device:0x2 func:0x0

PcieRootPortErrorHandler MailBox->PcieInitPar. SerrEmuTestEn = 0x0

GetPcieRootErrSts RpErrSts = 0x1

PcieAerGetErrSid ERRSID = 0x6410

PcieDeviceErrorHandler Socket = 0x0,Bus = 0x64,Device = 0x2,Function = 0x0, SevPerStack = 0x1

PcieIsCorrectableDeviceError Socket = 0x0,Bus = 0x64,Device = 0x2,Function = 0x0

PcieIsCorrectableDeviceError PCIE_DEVICE_STATUS = 0x2

PcieIsCorrectableDeviceError PCIE_DEVICE_CONTROL = 0x5147

PcieIsEdpcTiggered PCIE_DPC_OFFSET_REG_STS = 0x1F09

PcieIsEdpcTiggered PCIE_DPC_OFFSET_REG_CTL = 0x1A

PcieIsEdpcTiggered PCIE_DEVICE_CONTROL = 0x5147

PcieHandleEdpc Socket = 0x0,Bus = 0x64,Device = 0x2,Function = 0x0

PcieAerGetUncErrSts UncErrSts = 0x4000

PcieHandleEdpc Init ACPI Parameters...

EDPC Root Port - Socket:0 Bus:100 Dev:2 Fun:0

PcieHandleEdpc Triggering SW SCI...

IioGetXpErrorStatus Socket = 0x0,Bus = 64, Dev = 2, Func = 0,

PcieClearDeviceStatus Socket 0, Bus 64, Dev 2, Func0

Clear IEH error status S:0x0 B:0x64 D:0x0 F:0x4 Sev : IEH CORRECT ERROR

Enabling GlobalSysCtl S:0x0 B:0x64 D:0x0 F:0x4

after Enabling GlobalSysCtl S:0x0 B:0x64 D:0x0 F:0x4

Clear IEH error status S:0x0 B:0x7E D:0x0 F:0x3 Sev : IEH CORRECT ERROR

5. Check the dmesg in OS.

[1447.068712] pcieport 0000:64:02.0: EDR: EDR event received

[1447.068712] pcieport 0000:64:02.0: DPC: containment event, status:0x1f09 source:0x0000

[1447.068712] pcieport 0000:64:02.0: DPC: unmasked uncorrectable error detected

[1447.068712] pcieport 0000:64:02.0: PCIe Bus Error: severity=Uncorrected (Non-Fatal), type=Transaction Layer, (Requester ID)

[1447.068712] pcieport 0000:64:02.0:   device [8086:347a] error status/mask=00004000/00100020

[1447.068712] pcieport 0000:64:02.0:   [14] CmpltTO (First)

[1447.068712] nvme nvme0: frozen state error detected, reset controller

[1447.213041] nvme nvme0: restart after slot reset

[1447.213129] pcieport 0000:64:02.0: pciehp: Slot(10): Link Down/Up ignored (recovered by DPC)

[1447.235411] nvme nvme0: 128/0/0 default/read/poll queues

[1447.245785] pcieport 0000:64:02.0: AER: device recovery successful

6. Also try to do this error injection while reading data from the NVMe.

[root@localhost dev]# dd if=/dev/nvme0n1 of=/home/test/test bs=1MB count=4MB

<- in the middle of reading, do this completion timeout error injection with ITP.

kernel:watchdog: BUG: soft lockup - CPU#117 stuck for 48s! [dd:7435]

44010+0 records in

44009+0 records out

44009062400 bytes (44 GB, 41 GiB) copied, 441.387 s, 99.7 MB/s

7. Check the serial log again.

[Device Error] error on skt:0x0 Bus:0x64 Device:0x2 func:0x0

PcieRootPortErrorHandler MailBox->PcieInitPar.SerrEmuTestEn = 0x0

GetPcieRootErrSts RpErrSts = 0x1

PcieAerGetErrSid ERRSID = 0x6410

PcieDeviceErrorHandler Socket = 0x0,Bus = 0x64,Device = 0x2,Function = 0x0, SevPerStack = 0x1

PcieIsCorrectableDeviceError Socket = 0x0,Bus = 0x64,Device = 0x2,Function = 0x0

PcieIsCorrectableDeviceError PCIE_DEVICE_STATUS = 0x2

PcieIsCorrectableDeviceError PCIE_DEVICE_CONTROL = 0x5147

PcieIsEdpcTiggered PCIE_DPC_OFFSET_REG_STS = 0x1F09

PcieIsEdpcTiggered PCIE_DPC_OFFSET_REG_CTL = 0x1A

PcieIsEdpcTiggered PCIE_DEVICE_CONTROL = 0x5147

PcieHandleEdpc Socket = 0x0,Bus = 0x64,Device = 0x2,Function = 0x0

PcieAerGetUncErrSts UncErrSts = 0x4000

PcieHandleEdpc Init ACPI Parameters...

EDPC Root Port - Socket:0 Bus:100 Dev:2 Fun:0

PcieHandleEdpc Triggering SW SCI...

IioGetXpErrorStatus Socket = 0x0,Bus = 64, Dev = 2, Func = 0,

PcieClearDeviceStatus Socket 0, Bus 64, Dev 2, Func0

8. Clear IEH error status S:0x0 B:0x64 D:0x0 F:0x4 Sev: IEH CORRECT ERROR

Enabling GlobalSysCtl S:0x0 B:0x64 D:0x0 F:0x4

After Enabling GlobalSysCtl S:0x0 B:0x64 D:0x0 F:0x4

9. Clear IEH error status S:0x0 B:0x7E D:0x0 F:0x3 Sev: IEH CORRECT ERROR

10. Check the dmesg again.

[718.829838] pcieport 0000:64:02.0: EDR: EDR event received

[718.829943] pcieport 0000:64:02.0: DPC: containment event, status:0x1f09 source:0x0000

[718.829945] pcieport 0000:64:02.0: DPC: unmasked uncorrectable error detected

[718.829948] pcieport 0000:64:02.0: PCIe Bus Error: severity=Uncorrected (Non-Fatal), type=Transaction Layer, (Requester ID)

[718.829980] pcieport 0000:64:02.0: device [8086:347a] error status/mask=00004000/00100020

[718.830001] pcieport 0000:64:02.0: [14] CmpltTO (First)

[718.830021] nvme nvme0: frozen state error detected, reset controller

[718.982130] nvme nvme0: restart after slot reset

[718.982217] pcieport 0000:64:02.0: pciehp: Slot(10): Link Down/Up ignored (recovered by DPC)

[719.004387] nvme nvme0: 128/0/0 default/read/poll queues

[719.014695] pcieport 0000:64:02.0: AER: device recovery successful

4. Tencent Case Study on 3rd Generation Intel® Xeon® Scalable processor Servers

4.1. Problem

Tencent cloud has ever deployed Intel® VMD technology for some types of servers since Purley platform. In Q3 of 2021, it popped up a new demand that some of their 3rd Generation Intel® Xeon® Scalable processor servers need pass-through visit to each NVMe disk, however, enabling Intel® VMD will be against that usage. Tencent must disable Intel® VMD but still needs a solution to manage PCIe hot-plug devices. After comparison, EDPC was proposed mainly due to its less developed efforts and easy implementation.

4.2. Test Results and Deployment

Tencent system configuration: Following system config details described in 3.1 and 3.2 sessions to verify OS native mode.

Using the error injection card to inject hardware errors on the Root Port, Tencent validated all the use cases mentioned in [7] . No system hang, or reboot was observed.

Tencent did the physical hot-plug in/out of NVMe devices many times on each NVMe disk under extreme condition that there is heavy data traffic between host server and NVMe disks, no system hang or reboot.

Tencent has deployed the solution on 3rd Generation Intel® Xeon® Scalable processor servers and no system hang, or reboot was observed yet from hot-plug related events. It proves this EDPC is a reliable solution.

5. Summary

This technical paper introduced the working flow, validation recipe on how to apply EDPC RAS technology to the specific usage of PCIe hot-plug handling. Tencent real deployment case on 3rd Generation Intel® Xeon® Scalable processor on this technology is also demonstrated successfully.

6. References

6.1. References

1. 3rd Generation Intel® Xeon® Scalable Processors, Codename Ice Lake-SP External Design Specification (EDS), Volume One: Architecture.
2. Intel® Xeon® Processor Scalable Memory Family External Design Specification (EDS), Volume Two: Registers, Part A.
3. 602005-whitley-cooper-lake-ice-lake-ras-deepdive-rev0p76.
4. 614168_RAS_Tech_Integr_Val_Guide_Rev1_0.
5. PCI Express* Base Specification Revision 4.0 Version.
6. PCI Firmware Specification Revision 3.3.
7. PEI 4.0 User Guide, Intel, Oct. 2020
8. EDPC Configuration and Test Guide for PICE Hot-Plug Surprised Support, Intel, Sept. 2021

Reference Number: 729412, Revision: 0.7