# Setting up the PCIe hotplug in Kernel for flexible setups

Sergei Miroshnichenko

September 27, 2019

# Why do we need PCIe hotplug: replace

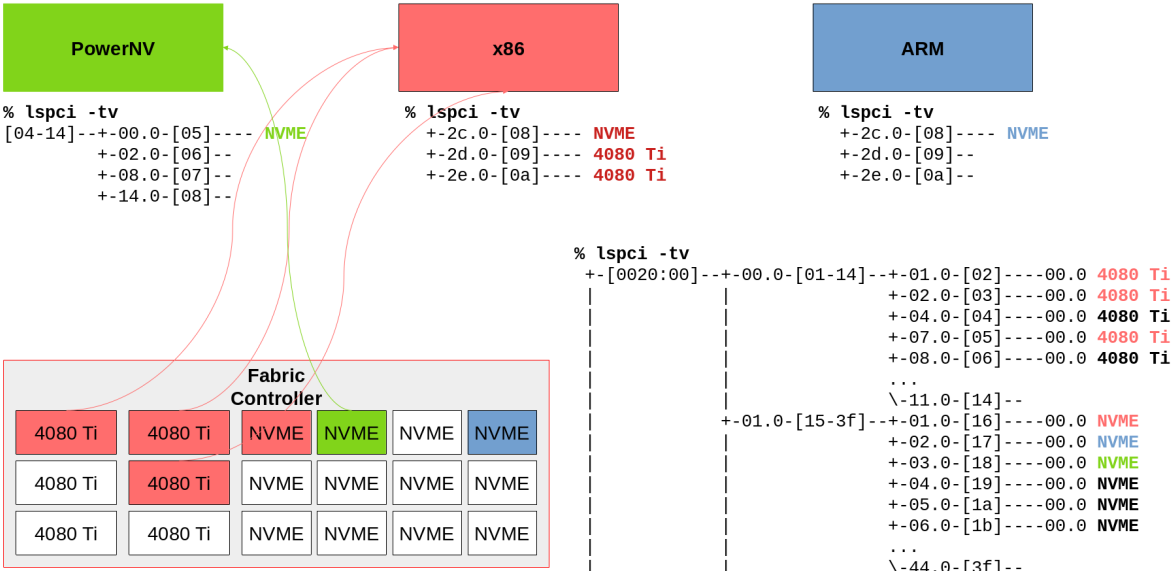
Replace/add/remove NVME drives in front panel of the server chassis

# Why do we need PCIe hotplug: PCIe-JBOD

Connect/detach a PCIe-JBOD to a server

# Why do we need PCIe hotplug: Fabric Mode

Handle rerouting of TLP packets for virtualized topologies in Fabric Mode



# Why do we need PCIe hotplug: misc

But there's more:

- Connect/detach a enclosure full of NVME drives inside a chassis
- Reconnect to a switch after changing its settings
- Testing procedures during manufacturing
- Some x86_64 machines (even servers) can't boot with NVME enclosures attached

# Hotplug today

Check if the Linux kernel supports PCIe hotplug:

```
% grep HOTPLUG_PCI /boot/config-5.2.0-pciehp+
CONFIG_HOTPLUG_PCI_PCIE=y
CONFIG_HOTPLUG_PCI=y
```

Check if a switch supports PCIe hotplug:

```
% sudo lspci -vvv -s 0000:02:00.0 | grep --color HotPlug
                SltCap: AttnBtn- PwrCtrl+ MRL- AttnInd- PwrInd- HotPlug+ Surprise-
```

Check if a free slot available:

```
% lspci -tv

 +-[0021:00]---00.0-[01-11]--+-00.0-[02-11]--+-00.0-[03-07]--
 |                           |                +-01.0-[08-0c]--
 |                           |                \-02.0-[0d-11]--
```

Check if a free space for new BARs available:

```
% sudo cat /proc/iomem

    3fe880000000-3fe8ffeffffff : PCI Bus 0021:02
      3fe880000000-3fe8807fffff : PCI Bus 0021:03
      3fe880800000-3fe880ffffff : PCI Bus 0021:08
      3fe881000000-3fe8817fffff : PCI Bus 0021:0d
```

# What can still go wrong

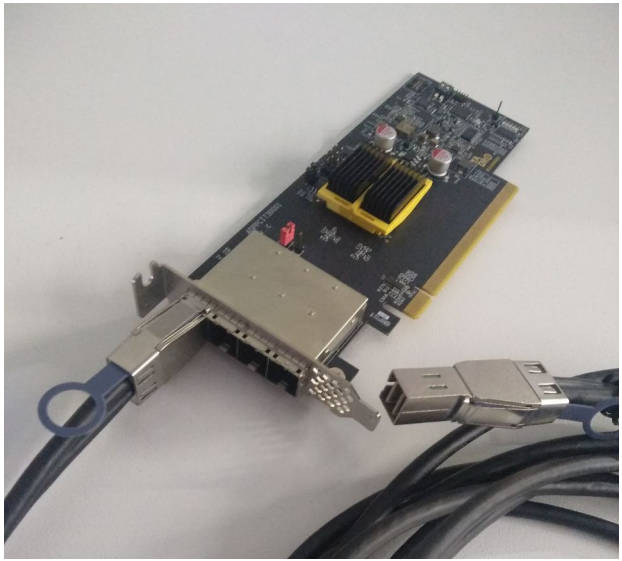
- BIOS/bootloader/firmware can reserve gaps in address space not big enough to fit new devices
- hotplugged switch may have more ports (we have 16 and 32 mostly) than reserved bus numbers (typically around 5)
- powerpc/PowerNV needs the special driver pnv_php which is incompatible with DPC, AER and the standard hotplug driver pciehp
- pnv_php also doesn't support manually initiated PCI rescan via sysfs
- surprise unplug may lead to unpredictable consequences: broken output of the lspci, hang, kernel oops

# X86_64 PCI topology example

Red - root bus / host bridge, Black - endpoint, Gray - virtual function, Orange - upstream port of a switch, Blue - downstream port of a switch

```
% lspci -tv
-[0000:00]-+-00.0  Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0  Intel Corporation I350 Gigabit Network Connection
           |            +-00.1  Intel Corporation I350 Gigabit Network Connection
           |            +-00.2  Intel Corporation I350 Gigabit Network Connection
           |            \-00.3  Intel Corporation I350 Gigabit Network Connection
           +-02.0  Intel Corporation Device 3e92
           +-14.0  Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-16.0  Intel Corporation 200 Series PCH CSME HECI #1
           +-17.0  Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[03]--
           +-1b.4-[04-07]----00.0-[05-07]--+-04.0-[06]--
           |                               \-05.0-[07]--
           +-1c.0-[08]--
           +-1c.4-[09]----00.0  ASMedia Technology Inc. Device 2142
           +-1d.0-[0a]----00.0  Lite-On Technology Corporation Device 23f1
           +-1f.0  Intel Corporation Z370 Chipset LPC/eSPI Controller
           +-1f.2  Intel Corporation 200 Series/Z370 Chipset Family Power Management
           +-1f.3  Intel Corporation 200 Series PCH HD Audio
           +-1f.4  Intel Corporation 200 Series/Z370 Chipset Family SMBus Controller
           \-1f.6  Intel Corporation Ethernet Connection (2) I219-V
```

# Switches: PCIe over Mini-SAS cables Adapter



x16 → x4x4x4x4

```
-00.0-[xx-xx]--+-00.0-[xx]--
               +-01.0-[xx]--
               +-02.0-[xx]--
               +-03.0-[xx]--
```

x16 → x8x4x4

```
-00.0-[xx-xx]--+-00.0-[xx]--
               +-01.0-[xx]--
               +-02.0-[xx]--
```

x16 → x8x8

```
-00.0-[xx-xx]--+-00.0-[xx]--
               +-01.0-[xx]--
```

x16 → x16

```
-00.0-[xx-xx]---00.0-[xx]--
```

# PCI Express capability registers: Device/Port Type

linux/include/uapi/linux/pci_regs.h:

```
#define   PCI_EXP_TYPE_ENDPOINT    0x0 /* Express Endpoint */
#define   PCI_EXP_TYPE_LEG_END     0x1 /* Legacy Endpoint */
#define   PCI_EXP_TYPE_ROOT_PORT   0x4 /* Root Port */
#define   PCI_EXP_TYPE_UPSTREAM    0x5 /* Upstream Port */
#define   PCI_EXP_TYPE_DOWNSTREAM  0x6 /* Downstream Port */
#define   PCI_EXP_TYPE_PCI_BRIDGE  0x7 /* PCIe to PCI/PCI-X Bridge */
#define   PCI_EXP_TYPE_PCIE_BRIDGE 0x8 /* PCI/PCI-X to PCIe Bridge */
#define   PCI_EXP_TYPE_RC_END      0x9 /* Root Complex Integrated Endpoint */
#define   PCI_EXP_TYPE_RC_EC       0xa /* Root Complex Event Collector */
```

# ID routing

ID = BDF address - domain:bus:device.function
Samsung NVME device has address 0000:8a:00.0

```
+-[0000:80]-+-02.0-[81]--
|           +-02.1-[82]--
|           +-03.0-[83-8e]----00.0-[84-8e]--+-04.0-[85]--
|           |                               +-05.0-[86-8c]----00.0-[87-8c]--+-00.0-[88]--
|           |                               |                               +-01.0-[89]--
|           |                               |                               +-02.0-[8a]----00.0  Samsung a804
|           |                               |                               +-03.0-[8b]--
|           |                               |                               \-04.0-[8c]--
|           |                               +-06.0-[8d]--
|           |                               \-07.0-[8e]--
|           ...
```

```
...--[primary_bus-xx]----xx.x-[secondary_bus-subordinate_bus]--...
```

Secondary bus - number of the port, (Secondary bus + 1) - first bus below the port, subordinate bus - last bus below the port

Route: 0000:80 [80-fe] → 80:03.0 [83-8e] → 83:00.0 [84-8e] → 84:05.0 [86-8c] →
→ 86:00.0 [87-8c] → 87:02.0 [8a] → 8a:00.0

# Powerpc/PowerNV PCI topology example

4 CPUs, 3 domain per CPU, 1 root per domain

```
-+-[0062:00]---00.0-[01]--
 +-[0061:00]---00.0-[01]--
 +-[0060:00]---00.0-[01]--
 +-[0042:00]---00.0-[01]--
 +-[0041:00]---00.0-[01]----00.0  SAS3216 PCI-Express Fusion-MPT SAS-3
 +-[0040:00]---00.0-[01]--
 +-[0022:00]---00.0-[01]--
 +-[0021:00]---00.0-[01]--
 +-[0020:00]---00.0-[01-0e]----00.0-[02-0e]----04.0-[03-0e]---00.0-[04-0e]--+-00.0-[05]--
 |                                                                           +-04.0-[06]--
 |                                                                           ...
 |                                                                           \-15.0-[0e]--
 +-[0002:00]---00.0-[01]--
 +-[0001:00]---00.0-[01-07]----00.0-[02-07]--+-01.0-[03-04]----00.0-[04]----00.0  ASPEED Graphics Family
 |                                           +-02.0-[05]----00.0  Texas Instruments USB 3.0 xHCI
 |                                           +-03.0-[06]--+-00.0  BCM5720 Gigabit Ethernet PCIe
 |                                           |            \-00.1  BCM5720 Gigabit Ethernet PCIe
 |                                           \-04.0-[07]--
 \-[0000:00]---00.0-[01-08]--+-00.0-[02-08]--+-00.0-[03]--
                             |               +-01.0-[04]--
                             |               +-02.0-[05]--
                             |               \-05.0-[08]--
                             \-00.1  PMC-Sierra Inc. Device 8535
```

# PCI topology example for Xeon with 2 CPU modules

Single domain with 4 roots, 2 per CPU: 0xff and 0x7f for "system peripherals", and 0x80-0xfe with 0x00-0x7e for external devices

```
-+-[0000:ff]-+-08.0  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D QPI Link 0
 |           +-08.2  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D QPI Link 0
 |           ...
 |           \-1f.2  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D Power Control Unit
 +-[0000:80]-+-02.0-[81]--
 |           +-02.1-[82]--
 |           +-03.0-[83-8e]----00.0-[84-8e]--+-04.0-[85-8b]----00.0-[86-8b]--+-00.0-[87]----00.0  Samsung a804
 |           |                               |                                ...
 |           |                               |                                \-04.0-[8b]--
 |           ...
 |           +-05.2  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D IIO RAS/Control Status
 |           \-05.4  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D I/O APIC
 +-[0000:7f]-+-08.0  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D QPI Link 0
 |           ...
 |           \-1f.2  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D Power Control Unit
 \-[0000:00]-+-00.0  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D DMI2
             +-01.0-[01]--
             ...
             \-1f.6  Intel Corporation C610/X99 series chipset Thermal Subsystem
```

# What is needed for topology enumeration

The only known about PCI topology before it is enumerated - are properties of root buses. We need to know:

- How to scan a bus
- How to validate if a device is present on an ID
- How to identify a type of a found device
- How to set up Primary/Secondary/Subordinate Bus Numbers for a switch

# Full search within a domain

- Bus Number is 8 bits: 0 to ff
- Device Number is 5 bits: 0 to 1f
- Function Number is 3 bits: 0 to 7

So the full search within a domain is:

- Functions of device 00 on bus 00: 00:00.0-00:00.7
- Functions of device 01 on bus 00: 00:01.0-00:01.7
- Functions of device 02 on bus 00: 00:02.0-7
- ...
- Functions of device 1f on bus 00: 00:1f.0-7
- Functions of device 00 on bus 01: 01:00.0-7
- Functions of device 01 on bus 01: 01:01.0-7
- ...
- Functions of device 1f on bus ff: ff:1f.0-7

But it can be optimized:

- If no function 0 - no need to look for other functions
- There is only one device on a downstream port, and it can only be xx:00.0-7

# Forbidden numbers

Reading 32-bit all ones (0xffffffff) is used as indicator of errors.
Reading 16-bit all ones (0xffff) from a special register is used as a criterion of an absent device.

# Configuration space registers of an endpoint (Type 0 Header)

| 31..24 | 23..16 | 15..8 | 7..0 | Byte Offset |
|---|---|---|---|---|
| Device ID | | Vendor ID | | 0x00 |
| Status | | Command | | 0x04 |
| Class Code | | | Revision ID | 0x08 |
| BIST | Header Type | Primary Latency Timer | Cache Line Size | 0x0c |
| Base Address Register 0 | | | | 0x10 |
| Base Address Register 1 | | | | 0x14 |
| Base Address Register 3 | | | | 0x18 |
| Base Address Register 4 | | | | 0x1c |
| Base Address Register 5 | | | | 0x20 |
| Base Address Register 6 | | | | 0x24 |

# Configuration space registers of a bridge (Type 1 Header)

| 31..24 | 23..16 | 15..8 | 7..0 | Byte Offset |
|---|---|---|---|---|
| Device ID | | Vendor ID | | 0x00 |
| Status | | Command | | 0x04 |
| Class Code | | | Revision ID | 0x08 |
| BIST | Header Type | Primary Latency Timer | Cache Line Size | 0x0c |
| Base Address Register 0 | | | | 0x10 |
| Base Address Register 1 | | | | 0x14 |
| Secondary Latency Timer | Subordinate Bus Number | Secondary Bus Number | Primary Bus Number | 0x18 |
| Secondary Status | | I/O Limit | I/O Base | 0x1c |
| Memory Limit | | Memory Base | | 0x20 |
| Prefetchable Memory Limit | | Prefetchable Memory Base | | 0x24 |
| Prefetchable Base Upper 32 Bits | | | | 0x28 |
| Prefetchable Limit Upper 32 Bits | | | | 0x2c |
| I/O Limit Upper 16 Bits | | I/O Base Upper 16 Bits | | 0x30 |

# The PCI ID Repository

linux/include/linux/pci_ids.h:

```
#define PCI_VENDOR_ID_IBM               0x1014
#define PCI_VENDOR_ID_DELL              0x1028
#define PCI_VENDOR_ID_HP                0x103c

/* Not yet there */
#define PCI_VENDOR_ID_YADRO             0x1d93
```

# PCI Class Code

linux/include/linux/pci_ids.h:

```
#define PCI_BASE_CLASS_STORAGE          0x01
#define PCI_CLASS_STORAGE_SCSI          0x0100
#define PCI_CLASS_STORAGE_IDE           0x0101
#define PCI_CLASS_STORAGE_FLOPPY        0x0102
...
#define PCI_CLASS_STORAGE_EXPRESS       0x010802
#define PCI_CLASS_STORAGE_OTHER         0x0180

#define PCI_BASE_CLASS_NETWORK          0x02
#define PCI_CLASS_NETWORK_ETHERNET      0x0200
...

#define PCI_BASE_CLASS_DISPLAY          0x03
#define PCI_CLASS_DISPLAY_VGA           0x0300
...
```

# PCI Device ID

linux/drivers/nvme/host/pci.c:

```
static const struct pci_device_id nvme_id_table[] = {
        { PCI_VDEVICE(INTEL, 0x0953),
                .driver_data = NVME_QUIRK_STRIPE_SIZE |
                               NVME_QUIRK_DEALLOCATE_ZEROES, },
        { PCI_VDEVICE(INTEL, 0x0a53),
                .driver_data = NVME_QUIRK_STRIPE_SIZE |
                               NVME_QUIRK_DEALLOCATE_ZEROES, },
        ...
        { PCI_DEVICE_CLASS(PCI_CLASS_STORAGE_EXPRESS, 0xffffff) },
        { PCI_DEVICE(PCI_VENDOR_ID_APPLE, 0x2001) },
        { PCI_DEVICE(PCI_VENDOR_ID_APPLE, 0x2003) },
        { 0, }
};

static struct pci_driver nvme_driver = {
        .name           = "nvme",
        .id_table       = nvme_id_table,
        ...
};

static int _init nvme_init(void)
{
        ...
        return pci_register_driver(&nvme_driver);
}
```

# Accesssing configuration space in Linux

linux/include/linux/pci.h:

```
int pci_read_config_byte(const struct pci_dev *dev, int where, u8 *val);
int pci_read_config_word(const struct pci_dev *dev, int where, u16 *val);
int pci_read_config_dword(const struct pci_dev *dev, int where, u32 *val);
int pci_write_config_byte(const struct pci_dev *dev, int where, u8 val);
int pci_write_config_word(const struct pci_dev *dev, int where, u16 val);
int pci_write_config_dword(const struct pci_dev *dev, int where, u32 val);
```

These are reduced to bus→ops→read() and bus→ops→write(), the struct pci_ops is defined for the plaform in linux/arch/*:

- x86_64: linux/arch/x86/pci/common.c: pci_read() and pci_write()
- PowerNV: linux/arch/powerpc/platforms/powernv/pci.c: pnv_pci_read_config() and pnv_pci_write_config()

# Enumeration example. Initial state: root bus only

```
-[0000:00]--
```

# First PCI_EXP_TYPE_ENDPOINT

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
```

# First PCI_EXP_TYPE_DOWNSTREAM

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0 PCI_EXP_TYPE_DOWNSTREAM
```

# Allocate and assign a new bus number to the port

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
```

# First port was empty, scan further

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1 PCI_EXP_TYPE_DOWNSTREAM
```

# Allocate and assign a new bus number to the port

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--
```

# Network card with virtual functions

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
```

# No more devices on bus 02, scan further

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
           +-14.0 Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-17.0 Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[03]--
           +-1b.4-[04]--
```

# PCI_EXP_TYPE_UPSTREAM - second level switch

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
           +-14.0 Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-17.0 Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[03]--
           +-1b.4-[04]--00.0 PCI_EXP_TYPE_UPSTREAM
```

# Set temp subordinate buses

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
           +-14.0 Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-17.0 Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[03]--
           +-1b.4-[04-ff]----00.0-[05-ff]--+-04.0-[06]--
```

# Found a third level switch

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
           +-14.0 Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-17.0 Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[03]--
           +-1b.4-[04-ff]----00.0-[05-ff]--+-04.0-[06-ff]----00.0-[07-ff]--+-00.0-[08]--
           |                               |                               +-04.0-[09]--
           |                               |                               +-08.0-[0a]--
           |                               |                               +-09.0-[0b]--
           |                               |                               +-0c.0-[0c]--
           |                               |                               +-10.0-[0d]--
           |                               |                               +-11.0-[0e]--
           |                               |                               +-14.0-[0f]--
           |                               |                               \-15.0-[10]--
```

# The switch is fully scanned, update subordinate buses

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
           +-14.0 Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-17.0 Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[03]--
           +-1b.4-[04-ff]----00.0-[05-ff]--+-04.0-[06-10]----00.0-[07-10]--+-00.0-[08]--
           |                               |                               +-04.0-[09]--
           |                               |                               +-08.0-[0a]--
           |                               |                               +-09.0-[0b]--
           |                               |                               +-0c.0-[0c]--
           |                               |                               +-10.0-[0d]--
           |                               |                               +-11.0-[0e]--
           |                               |                               +-14.0-[0f]--
           |                               |                               \-15.0-[10]--
```

# Found one more third level switch

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
           +-14.0 Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-17.0 Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[03]--
           +-1b.4-[04-ff]----00.0-[05-ff]--+-04.0-[06-10]----00.0-[07-10]--+-00.0-[08]--
           |                               |                               +-04.0-[09]--
           |                               |                               ...
           |                               |                               \-15.0-[10]--
           |                               \-05.0-[11-1b]----00.0-[12-1b]--+-00.0-[13]--
           |                                                               +-04.0-[14]--
           |                                                               +-08.0-[15]--
           |                                                               +-09.0-[16]--
           |                                                               +-0c.0-[17]--
           |                                                               +-10.0-[18]--
           |                                                               +-11.0-[19]--
           |                                                               +-14.0-[1a]--
           |                                                               \-15.0-[1b]--
```

# Nothing more on the bus 04, update subordinate buses

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
           +-14.0 Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-17.0 Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[03]--
           +-1b.4-[04-1b]----00.0-[05-1b]--+-04.0-[06-10]----00.0-[07-10]--+-00.0-[08]--
           |                               |                               +-04.0-[09]--
           |                               |                               ...
           |                               |                               \-15.0-[10]--
           |                               \-05.0-[11-1b]----00.0-[12-1b]--+-00.0-[13]--
           |                                                               +-04.0-[14]--
           |                                                               +-08.0-[15]--
           |                                                               +-09.0-[16]--
           |                                                               +-0c.0-[17]--
           |                                                               +-10.0-[18]--
           |                                                               +-11.0-[19]--
           |                                                               +-14.0-[1a]--
           |                                                               \-15.0-[1b]--
```

# Scan the rest

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01]--
           +-01.1-[02]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
           +-14.0 Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-17.0 Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[03]--
           +-1b.4-[04-1b]----00.0-[05-1b]--+-04.0-[06-10]----00.0-[07-10]--+-00.0-[08]--
           |                               |                               +-04.0-[09]--
           |                               |                               ...
           |                               |                               \-15.0-[10]--
           |                               \-05.0-[11-1b]----00.0-[12-1b]--+-00.0-[13]--
           |                                                               +-04.0-[14]--
           |                                                               ...
           |                                                               \-15.0-[1b]--
           +-1c.4-[09]----00.0  ASMedia Technology Inc. Device 2142
```

# Oops, forgot to reserve bus numbers

```
-[0000:00]-+-00.0 Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers
           +-01.0-[01-04]--
           +-01.1-[05]--+-00.0 Intel Corporation I350 Gigabit Network Connection
           |            +-00.1 Intel Corporation I350 Gigabit Network Connection
           |            +-00.2 Intel Corporation I350 Gigabit Network Connection
           |            \-00.3 Intel Corporation I350 Gigabit Network Connection
           +-14.0 Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller
           +-17.0 Intel Corporation 200 Series PCH SATA controller [AHCI mode]
           +-1b.0-[06-09]--
           +-1b.4-[0a-57]----00.0-[0b-57]--+-04.0-[0c-31]----00.0-[0d-31]--+-00.0-[0e-11]--
           |                               |                               +-04.0-[12-15]--
           |                               |                               ...
           |                               |                               \-15.0-[2e-31]--
           |                               \-05.0-[32-57]----00.0-[33-37]--+-00.0-[34-37]--
           |                                                               +-04.0-[38-3b]--
           |                                                               ...
           |                                                               \-15.0-[54-57]--
           +-1c.4-[09]----00.0  ASMedia Technology Inc. Device 2142
```

# Base Address Registers - BARs

Bridges can have up to 2, endpoints - up to 6 Base Address Registers, used for memory mapping

```
 +-[0001:00]---00.0-[01-07]----00.0-[02-07]--+-01.0-[03-04]--...
 |                                           +-02.0-[05]----00.0  TUSB73x0 SuperSpeed USB 3.0 xHCI
 |                                           +-03.0-[06]--+-00.0  Broadcom BCM5720 Gigabit Ethernet PCIe
 |                                           |            \-00.1  Broadcom BCM5720 Gigabit Ethernet PCIe
```

```
% sudo cat /proc/iomem
210000000000-21fdfffffff : /pciex@3fffe40100000
  210000000000-21fdfff0fff : PCI Bus 0001:01
    210000000000-21fdfff0fff : PCI Bus 0001:02
      210000000000-2100fffffff : PCI Bus 0001:06
        210000000000-21000000fff : 0001:06:00.0
          210000000000-21000000fff : tg3
        210000010000-21000001fff : 0001:06:00.0
          210000010000-21000001fff : tg3
        210000030000-21000003fff : 0001:06:00.1
          210000030000-21000003fff : tg3
        210000040000-21000004fff : 0001:06:00.1
          210000040000-21000004fff : tg3
3fe080000000-3fe0fffefff : /pciex@3fffe40100000
  3fe080000000-3fe0ffefffff : PCI Bus 0001:01
    3fe080000000-3fe0ffefffff : PCI Bus 0001:02
      3fe080000000-3fe0817fffff : PCI Bus 0001:03
      3fe081800000-3fe081ffffff : PCI Bus 0001:05
        3fe081800000-3fe08180ffff : 0001:05:00.0
          3fe081800000-3fe08180ffff : xhci-hcd
        3fe081810000-3fe081811fff : 0001:05:00.0
```

# BARs: Address Routing

USB controller has ID 0001:05:00.0, but it is not used anymore when I/O via Memory Requests. Instead, the address is used for routing.

Each switch has bridge windows, which contain the range of all BARs “below”.

```
+-[0001:00]---00.0-[01-07]----00.0-[02-07]--+-01.0-[03-04]----00.0-[04]----00.0
|                                           +-02.0-[05]----00.0  TUSB73x0 SuperSpeed USB 3.0 xHCI

3fe080000000-3fe0fffeffff : /pciex@3fffe40100000
  3fe080000000-3fe0ffefffff : PCI Bus 0001:01
    3fe080000000-3fe0ffefffff : PCI Bus 0001:02
      3fe080000000-3fe0817fffff : PCI Bus 0001:03
        3fe080000000-3fe0817fffff : PCI Bus 0001:04
          3fe080000000-3fe080ffffff : 0001:04:00.0
          3fe081000000-3fe08101ffff : 0001:04:00.0
      3fe081800000-3fe081ffffff : PCI Bus 0001:05
        3fe081800000-3fe08180ffff : 0001:05:00.0
          3fe081800000-3fe08180ffff : xhci-hcd
        3fe081810000-3fe081811fff : 0001:05:00.0
    3fe082000000-3fe08203ffff : 0001:01:00.0
```

# BARs: Bridge Windows

Bridge window set the range [Base - Limit] and is similar to [secondary - subordinate] for buses.

| 31..24 \| 23..16 | 15..8 | 7..0 | Byte Offset |
|---|---|---|---|
| Secondary Status | I/O Limit | I/O Base | 0x1c |
| Memory Limit | Memory Base | | 0x20 |
| Prefetchable Memory Limit | Prefetchable Memory Base | | 0x24 |
| Prefetchable Base Upper 32 Bits | | | 0x28 |
| Prefetchable Limit Upper 32 Bits | | | 0x2c |
| I/O Limit Upper 16 Bits | I/O Base Upper 16 Bits | | 0x30 |

Three types of memory requests, each served by separate bridge window

- IO space (IO) - legacy, PowerNV doesn't support it
- Non-Prefetchable space (MEM) - may have read side-effects (status registers self-clearing after being read)
- Prefetchable space (MEM, MEM64) - recommended by the PCI Express Base Specification

# BAR usage

linux/drivers/nvme/host/pci.c:

```
static int nvme_remap_bar(struct nvme_dev *dev, unsigned long size)
{
        ...
        dev->bar = ioremap(pci_resource_start(pdev, 0), size);
        ...
}

static int nvme_pci_configure_admin_queue(struct nvme_dev *dev)
{
        ...
        dev->subsystem = readl(dev->bar + NVME_REG_VS) >= NVME_VS(1, 1, 0) ?
                                NVME_CAP_NSSRC(dev->ctrl.cap) : 0;
        ...
}

static int nvme_pci_reg_read32(struct nvme_ctrl *ctrl, u32 off, u32 *val)
{
        *val = readl(to_nvme_dev(ctrl)->bar + off);
        return 0;
}

static int nvme_pci_reg_write32(struct nvme_ctrl *ctrl, u32 off, u32 val)
{
        writel(val, to_nvme_dev(ctrl)->bar + off);
        return 0;
}
```

# Problems with hotplug

Typical situation:

```
% lspci -tv

 +-[0021:00]---00.0-[01-11]--+-00.0-[02-11]--+-00.0-[03-07]--
 |                           |                +-01.0-[08-0c]--
 |                           |                \-02.0-[0d-11]--

% sudo cat /proc/iomem

    3fe880000000-3fe8ffefffff : PCI Bus 0021:02
      3fe880000000-3fe8807fffff : PCI Bus 0021:03
      3fe880800000-3fe880ffffff : PCI Bus 0021:08
      3fe881000000-3fe8817fffff : PCI Bus 0021:0d
```

- Typically 5 bus numers reserved per slot, not enough to fit a switch with 16 or 32 ports
- Too small reserved space for new BARs (just 8MiB here)
- Bridge window may have enough free space, but too fragmented
- Some bridges report about hotplug events with MSI interrupts, but some are not, so a manual rescan triggering required: echo 1 > /sys/bus/pci/rescan

# Movable BARs: Concept

Before the rescan:

Bridge 0b has range [0c-1c] and an nvme on bus 18

```
% sudo cat /proc/iomem
3fe800000000-3fe8007fffff : PCI Bus 0020:0b      ←
  3fe800000000-3fe8007fffff : PCI Bus 0020:18
    3fe800000000-3fe8000fffff : 0020:18:00.0
      3fe800000000-3fe8000fffff : nvme
    3fe800100000-3fe80017ffff : 0020:18:00.0
```

New NVME with two BARs is going to be added to bus 10

```
04000 : 0020:10:00.0
10000 : 0020:10:00.0
```

After the rescan:

```
% sudo cat /proc/iomem

          3fe800000000-3fe800ffffff : PCI Bus 0020:0b
            3fe800000000-3fe8007fffff : PCI Bus 0020:10
              3fe800000000-3fe800003fff : 0020:10:00.0
                3fe800000000-3fe800003fff : nvme
              3fe800010000-3fe80001ffff : 0020:10:00.0
            3fe800800000-3fe800ffffff : PCI Bus 0020:18
              3fe800800000-3fe8008fffff : 0020:18:00.0
                3fe800800000-3fe8008fffff : nvme
              3fe800900000-3fe80097ffff : 0020:18:00.0
```

# Movable BARs

Currently, hotplug event on a slot is local to its direct switch only, just utilizing the reserved gaps between the used BARs, without altering the switch's registers.

But when BARs are movable, hotplug event in the middle of the PCIe tree affects almost every bridge window of almost every switch in the whole topology.

An interrupt from pciehp (standard hotplug driver) will lead to full domain rescan.

# Movable BARs: Pausing the drivers

Prior to moving BARs of a device which is bound to a driver, this driver must be informed about the coming rescan, so it will:

- Pause its threads and block the API to prevent generating memory requests (if any)
- Unmap the used BARs (if any)
- Invalidate pointers to previously used BARs

When rescan is done, it is safe to resume the drivers:

- Take new addresses and map them
- Unpause the threads, unblock the API

If a device is not used by any driver, its BARs can be moved freely.

# Movable BARs: Pausing the drivers - code

linux/include/linux/pci.h:

```
struct pci_driver {
        int  (*sriov_configure)(struct pci_dev *dev, int num_vfs); /* On PF */
+       void (*rescan_prepare)(struct pci_dev *dev);
+       void (*rescan_done)(struct pci_dev *dev);
        const struct pci_error_handlers *err_handler;
};
```

linux/drivers/nvme/host/pci.c:

```
+static void nvme_rescan_prepare(struct pci_dev *pdev)
+{
+       nvme_dev_disable(dev, false);
+       nvme_dev_unmap(dev);
+       dev->bar = NULL;
+}
+static void nvme_rescan_done(struct pci_dev *pdev)
+{
+       nvme_dev_map(dev);
+       nvme_reset_ctrl_sync(&dev->ctrl);
+}

static struct pci_driver nvme_driver = {
        .err_handler    = &nvme_err_handler,
+       .rescan_prepare = nvme_rescan_prepare,
+       .rescan_done    = nvme_rescan_done,
};
```

# Movable BARs: Immovable BARs

BARs are not always movable:

- A driver is not yet patched to support pausing and BAR remapping
- BAR movement is not feasible at all for some devices
- A device is not bound to a driver, but is still used - like VGA

Devices with immovable BARs must be guaranteed to have them remain on the same address

drivers/pci/probe.c:

```
+bool pci_dev_movable_bars_supported(struct pci_dev *dev)
+{
+       if (!dev)
+               return false;
+
+       if (dev->driver && dev->driver->rescan_prepare)
+               return true;
+
+       if ((dev->class >> 8) == PCI_CLASS_DISPLAY_VGA)
+               return false;
+
+       return !dev->driver;
+}
```

# Movable bus numbers: reserving is not a good option

Any bus number assignment policy can be broken by hotplugging nested bridges:

```
+-[0020:00]---00.0-[01-20]--+-00.0-[02-20]--+-00.0-[03-07]--    <--  *BRIDGE*
|                           |               +-...
|                           |               \-05.0-[1c-20]--

04.0-[xx-xx]--+-00.0-[xx-xx]--
              +-04.0-[xx-xx]--
              +-09.0-[xx-xx]--
              +-0c.0-[xx-xx]----00.0-[xx-x]--+-01.0-[xx-xx]--
              |                              +-02.0-[xx-xx]--
              |                              +-04.0-[xx-xx]--
              |                              +-05.0-[xx-xx]--
              |                              +-06.0-[xx-xx]--
              |                              +-07.0-[xx-xx]--
              |                              +-08.0-[xx-xx]--
              |                              +-09.0-[xx-xx]--
              |                              +-0a.0-[xx-xx]--
              |                              +-0b.0-[xx-xx]--
              |                              +-0c.0-[xx-xx]--
              |                              +-0d.0-[xx-xx]--
              |                              +-0e.0-[xx]----00.0 Toshiba America Info Systems Device 010e
              |                              +-0f.0-[xx-xx]--
              |                              +-10.0-[xx-xx]--
              |                              \-11.0-[xx-xx]--
              +-10.0-[xx-xx]--
              +-11.0-[xx-xx]--
              ...
```

# Movable bus numbers: Concept

Movable bus numbers imply that reservation is not needed anymore

```
+-[0020:00]---00.0-[01-20]--+-00.0-[02-08]--+-00.0-[03]--    <--  *BRIDGE*
|                           |               +-01.0-[04]--
|                           |               +-02.0-[05]--
|                           |               +-03.0-[06]--
|                           |               +-04.0-[07]--
|                           |               \-05.0-[08]--

+-[0020:00]---00.0-[01-22]--+-00.0-[02-22]--+-00.0-[03-1d]----04.0-[04-1d]--+-00.0-[05]--
|                           |               |                               +-04.0-[06]--
|                           |               |                               +-09.0-[07]--
|                           |               |                               +-0c.0-[08-19]----00.0-[09-19]--+-01.0-[0a]--
|                           |               |                               |                               +-02.0-[0b]--
|                           |               |                               |                               +-04.0-[0c]--
|                           |               |                               |                               +-05.0-[0d]--
|                           |               |                               |                               +-06.0-[0e]--
|                           |               |                               |                               +-07.0-[0f]--
|                           |               |                               |                               +-08.0-[10]--
|                           |               |                               |                               +-09.0-[11]--
|                           |               |                               |                               +-0a.0-[12]--
|                           |               |                               |                               +-0b.0-[13]--
|                           |               |                               |                               +-0c.0-[14]--
|                           |               |                               |                               +-0d.0-[15]--
|                           |               |                               |                               +-0e.0-[16]----00.0 Toshiba
|                           |               |                               |                               +-0f.0-[17]--
|                           |               |                               |                               +-10.0-[18]--
|                           |               |                               |                               \-11.0-[19]--
|                           |               |                               ...
|                           |               |                               \-15.0-[1d]--
|                           |               +-01.0-[1e]--
|                           |               +-02.0-[1f]--
|                           |               +-03.0-[20]--
|                           |               +-04.0-[21]--
|                           |               \-05.0-[22]--
```

# Movable bus numbers

linux/include/linux/pci.h: BDFs are not used explicitly:

```
int pci_read_config_byte(const struct pci_dev *dev, int where, u8 *val);
int pci_read_config_word(const struct pci_dev *dev, int where, u16 *val);
int pci_read_config_dword(const struct pci_dev *dev, int where, u32 *val);
int pci_write_config_byte(const struct pci_dev *dev, int where, u8 val);
int pci_write_config_word(const struct pci_dev *dev, int where, u16 val);
int pci_write_config_dword(const struct pci_dev *dev, int where, u32 val);
```

# Movable bus numbers: sysfs, procfs

sysfs and procfs entries and symlinks of devices are all based on their BDFs

```
% ls -la /sys/bus/pci/devices
0000:00:00.0 -> ../../../devices/pci0000:00/0000:00:00.0
0000:00:02.0 -> ../../../devices/pci0000:00/0000:00:02.0
...
0000:04:00.0 -> ../../../devices/pci0000:00/0000:00:1c.6/0000:04:00.0
0000:40:00.0 -> ../../../devices/pci0000:00/0000:00:1d.2/0000:40:00.0
```

```
% ls -la /proc/bus/pci/*
/proc/bus/pci/00:
00.0
02.0
...
1f.4
1f.6

/proc/bus/pci/04:
00.0

/proc/bus/pci/40:
00.0
```

```
% ls -la /sys/devices/pci0000:00/
0000:00:00.0
0000:00:02.0
...
0000:00:1f.3
0000:00:1f.4
0000:00:1f.6

% ls -la /sys/devices/pci0000:00/0000:00:1c.6/0000:04:00.0/driver
driver -> ../../../../bus/pci/drivers/iwlwifi
```

# Movable bus numbers: bus renaming

The only function of the kernel API serving the purpose of renaming devices is used currently only to rename network interfaces - device_rename(&dev→dev, new_name) - and it doesn't handle sysfs nor procfs.

There is no event in the kernel/udev for seamless changing of topology (except of the deprecated device_move(struct device *dev, struct device *new_parent)) - only for adding and removing.

Sysfs entries are created by the kernel on bus_add_device(dev) and pci_create_sysfs_dev_files(dev), removed on bus_remove_device(dev) and pci_remove_sysfs_dev_files(dev).
Sysfs symlinks - device_add_class_symlinks(dev) and device_remove_class_symlinks(dev).
Procfs entries - pci_proc_attach_device(dev) and pci_proc_detach_device(dev)

# Movable bus numbers: bus renaming

So all sysfs and procfs entries and symlinks are to be destroyed before renaming the device, and then recreated again, based on new BDFs.

The bus_remove_device(dev) is too aggressive - it also detaches the device from its driver, and it completely breaks our concept, so we added a bus_disconnect_device(dev) to the Base API, to keep an affected device bound to its driver.

That wasn't the only required change in the Base kernel API: the existing bus_add_device(dev), device_add_class_symlinks(dev) and device_remove_class_symlinks(dev) are moved from linux/drivers/base/base.h to linux/include/linux/device.h

# Movable bus numbers: renaming buses and devices

```
+static void pci_buses_remove_sysfs(int domain, int busnr, int max_bus_number)
+{
+       list_for_each_entry(dev, &bus->devices, bus_list) {
+               device_remove_class_symlinks(&dev->dev);
+               pci_remove_sysfs_dev_files(dev);
+               pci_proc_detach_device(dev);
+               bus_disconnect_device(&dev->dev);
+       }
+
+       device_remove_class_symlinks(&bus->dev);
+       pci_proc_detach_bus(bus);
+}
+static void pci_rename_bus_devices(struct pci_bus *bus, const int domain,
+                                  const int new_busnr)
+{
+       list_for_each_entry(dev, &bus->devices, bus_list) {
+               char new_name[64];
+               sprintf(new_name, "%04x:%02x:%02x.%d", domain, new_busnr,
+                       PCI_SLOT(dev->devfn), PCI_FUNC(dev->devfn));
+               class = dev->dev.class;
+               dev->dev.class = NULL;
+               device_rename(&dev->dev, new_name);
+               dev->dev.class = class;
+       }
+}
```

# Movable BARs and buses: activating

Linux kernel will only reassign bus numbers if the “pci=realloc” command line argument is passed.

On x86_64 it’s a little longer: “pci=realloc,assign-busses,use_crs” - these flags allow to ignore settings proposed by BIOS

# Movable BARs and buses: OpenPower

Currently, arch code for OpenPower only refers to Device Tree for device info. It was required to let it ignore this blob and probe the PCIe fabric directly.

PowerNV has a heavy mechanism of detecting PCIe errors and isolating failing fragments of the topology - EEH. EEH prevents the topology from changes. If don't prepare the EEH code to changes in topology and BARs, it will consider them as errors.

Linux kernel is not alone on CPUs, there is another (BIOS-alike) entity running in parallel - OPAL, it contains a cache of PCIe topology (hierarchy of IDs - BDFs), that must be updated when moving buses.

New pcibios_* hooks are introduced to warn the platfrom about ongoing changes:
linux/arch/powerpc/kernel/pci-hotplug.c

```
+void pcibios_rescan_prepare(struct pci_dev *pdev)
+{
+       eeh_addr_cache_rmv_dev(pdev);
+}
+
+void pcibios_rescan_done(struct pci_dev *pdev)
+{
+       eeh_addr_cache_insert_dev(pdev);
+}
```

# Suprise Unplug: DPC

Without a special feature called DPC, if a device has been unexpectedly detached, the phantom remains in system, causing unmasked, uncorrectable errors on the bus when accessed.

A switch with DPC masks these events when finds out that the device is not there anymore. Instead, it sends an interrupt to the host and emulates the answers on read/write requests with 0xff.

The DPC driver in the kernel catches that interrupt and informs the affected driver to detach, and removes the device phantom from the system. In that way the affected drivers will much less probably crash the kernel.

# Suprise Unplug: DPC: show me the code

linux/drivers/pci/pcie/dpc.c:

```
static irqreturn_t dpc_handler(int irq, void *context)
{
    pcie_do_recovery(pdev, pci_channel_io_frozen, PCIE_PORT_SERVICE_DPC);
}
```

linux/drivers/pci/pcie/err.c:

```
static int report_error_detected(struct pci_dev *dev, enum pci_channel_state state, ...
{
    dev->driver->err_handler->error_detected(dev, state);
}
```

linux/drivers/nvme/host/pci.c:

```
static pci_ers_result_t nvme_error_detected(struct pci_dev *pdev, pci_channel_state_t state)
{
    switch (state) {
        case pci_channel_io_frozen:
            nvme_dev_disable(dev, false);
}

static const struct pci_error_handlers nvme_err_handler = {
    .error_detected    = nvme_error_detected,
};
```

# Unexpected incidents

Runtime PM is a power saving feature that tends to shutdown devices it considers unused - unexpectedly. To wake it up before using:

```
static void pci_bus_rescan_prepare(struct pci_bus *bus)
{
+       if (bus->self)
+               pci_config_pm_runtime_get(bus->self);
...
}

static void pci_bus_rescan_done(struct pci_bus *bus)
{
...
+       if (bus->self)
+               pci_config_pm_runtime_put(bus->self);
+}
```

When rescanning the domain via /sys/bus/pci/rescan , the PCIe setting wasn't applied:

```
unsigned int pci_rescan_bus(struct pci_bus *bus)
{
...
+       list_for_each_entry(child, &root->children, node)
+               pcie_bus_configure_settings(child);
...
}
```

# Bugs to fix

Not only features was introduces, but also bugs fixed to make hotplug working:

Fix an old concurrency issue, when:

- two or more devices are being hot-added into a bridge which was initially empty;
- a bridge with two or more devices is being hot-added;
- during boot, if BIOS/bootloader/firmware doesn't pre-enable bridges.

Fixed a regression for pciehp introduced in v4.19, merged in v5.1: commit 3943af9d01e94330d0cfac6fccdbc829aad50c92 PCI: pciehp: Ignore Link State Changes after powering off a slot

Added yet another quirk in pciehp, now for PLX switches: when an interrupt for an event is disable, it's flag in the status register will not raise as well.

Fin