Configuration Request Retry Status (CRS) Handling

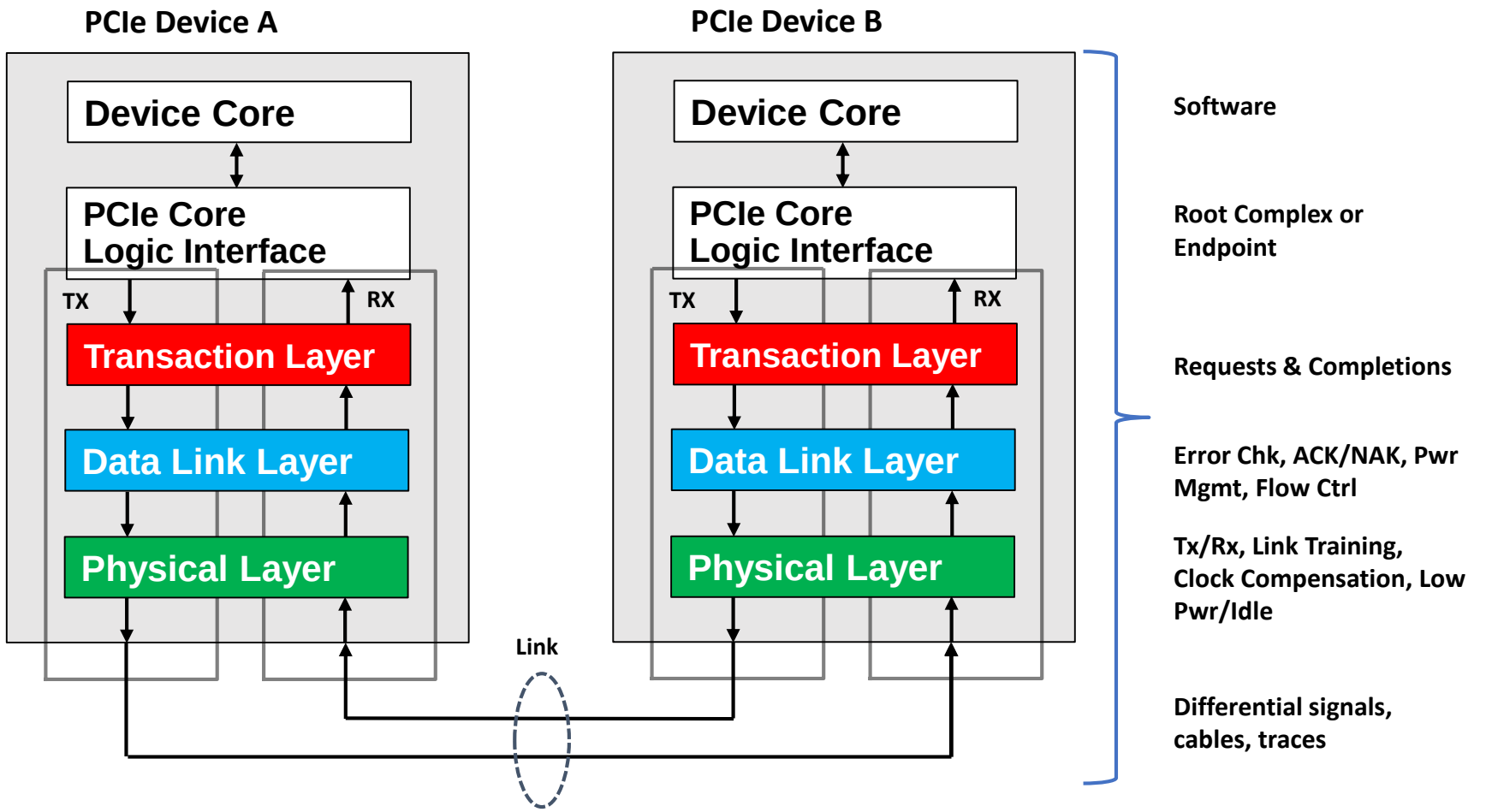
Sinan Kaya

okaya@codeaurora.org

Few words about me

- Sr. Staff Engineer @ Qualcomm Datacenter Technologies
- Focus areas
  - Arm64 Servers
  - PCI Express
  - ACPI
  - DMA Engine
  - More and more low level stuff everyday

PCIe Device Layers



https://www.mindshare.com/files/ebooks/PCI%20Express%20System%20Architecture.pdf

PCIe Transaction Layer

- PCIe defines four types of transactions:
  - Memory
    - used for data transfer
  - I/O
    - used for data transfer
  - Configuration
    - device configuration
  - Message
    - event signaling

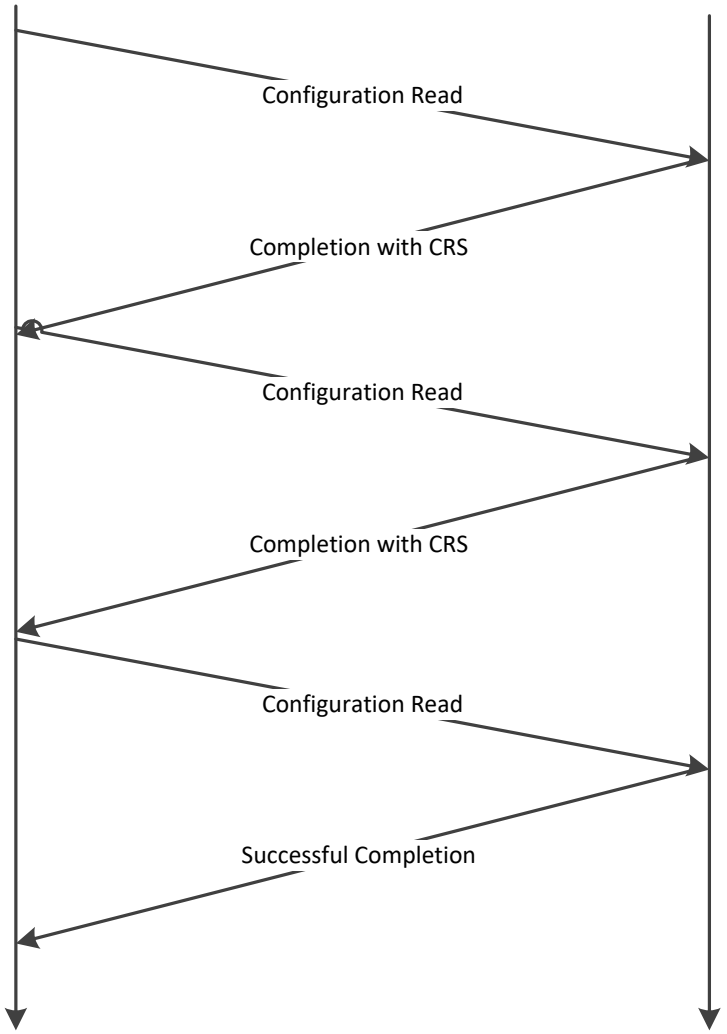
Requests and Completions

- Request Types categorized as
  - Posted
    - Memory write
    - Messages
  - Non-posted
    - Config/IO Write
    - AtomicOp Request
    - Read request

Sample Number	PCI-Express Packet	Direction	Sequence Number	Tag	Length	Register Number	Data	Payload	Completion Status
0	CfgRd_0	PCIe-101:Down	7A7	00	001	15	No data		
0	Ack	PCIe-102:Up	7A7						
1	CplD	PCIe-102:Up	809	00	001		With data	00000008	Successful Completion (SC)
1	Ack	PCIe-101:Down	809						
2	CfgRd_0	PCIe-101:Down	7A8	00	001	01	No data		
2	Ack	PCIe-102:Up	7A8						
3	CplD	PCIe-102:Up	80A	00	001		With data	00000000	Successful Completion (SC)
3	Ack	PCIe-101:Down	80A						
4	CfgWr_0	PCIe-101:Down	7A9	00	001	01	With data	79000003	
4	Ack	PCIe-102:Up	7A9						
5	Cpl	PCIe-102:Up	80B	00	000		No data		Successful Completion (SC)

CRS Definition

- Some devices take long time to initialize following a reset.
- Device responds with CRS status code during this period for any configuration request
  - Meaning please try again later



CRS Requirement

- Rev 3.1 Sec 2.3.1 Request Handling Rules
- “Valid reset conditions after which a device is permitted to return CRS are:
  - Cold, Warm, and Hot Resets
  - FLRs
  - A reset initiated in response to a D3hot to D0 uninitialized device state transition”

CRS Handling Rules

- CRS support in RC is mandatory
- PCIe spec defines CRS Software Visibility capability in Root Capabilities register.
  - If supported by HW, OS gets to know when a device is not ready by reading a value of 0x0001 for vendor id register. OS polls while configuration read is pending
  - If not supported, HW generally retries the vendor id request until CRS condition is cleared
    - Possible deadlock if HW firmware initializes during OS boot via firmware interface
    - PCI read is stuck and code never makes that far into the firmware loading phase
    - Spec says a root port can limit the number of retries.
- Linux enables CRS visibility by default in pci_scan_bridge() and relies on graceful polling.

Current Status and To-do

- pci_bus_read_dev_vendor_id() knows how to deal with CRS
- As of 4.14 kernel, Linux
  - handles CRS during
    - Probe
    - FLR (indirectly by extended polling period in pci_flr_wait())
  - does not handle
    - Warm/hot reset (secondary bus reset after pci_reset_bridge_secondary_bus())
    - D3-D0 transition

CRS following Warm Reset

```
int pci_try_reset_bus(struct pci_bus *bus)
{
        int rc;

        rc = pci_bus_reset(bus, 1);
        if (rc)
                return rc;

        pci_bus_save_and_disable(bus);

        if (pci_bus_trylock(bus)) {
                might_sleep();
                pci_reset_bridge_secondary_bus(bus->self);
                pci_bus_unlock(bus);
        } else
                rc = -EAGAIN;

        pci_bus_restore(bus);

        return rc;
}
```

Existing Proposals to Fix Warm Reset

- Facts.
  - Secondary bus reset is a concept that comes from standard PCI
  - Secondary bus reset is a broadcast message to all children under this bus
  - Hot reset messages gets forwarded to all downstream ports by switches
  - CRS is a PCIe concept
  - There can only be one device on a PCIe bus due to its serial bus structure
- Proposals
  - Initial patch posted on the maillist was too aggressive.
    - It read the vendor id of all children devices and created a function similar to walk_bus due to pci device link lists not being set up by the time it was called
  - Another patch was to get rid of the bus walk and move pci_bus_read_dev_vendor_id() calls into pci_bus_restore() function

My questions

- Where do we go from here?
- How do we fix D3->D0 case?
  - Is there a concern with extended sleep times (up to 60 seconds)
- Any other use case for CRS?