

Peripheral Component Interconnect LOCAL BUS

Shreeya Badhe

HPC-Tech Group

C-DAC, Pune, India

Motivation

- Transfer speeds over I/O bus had become a bottleneck, especially for graphics. Moving peripheral functions with high bandwidth requirements closer to the system's processor bus can eliminate this bottleneck.

Bandwidth hungry peripherals

- Graphics and Gaming!! (1024x768, 32-bit colour@30fps)
 - Network (e.g. GbE)
 - Storage (e.g. ATA)
 - Multimedia
- The buses before PCI were mainly architecture specific. e.g. ISA/EISA(x86), SBUS (sparc). Different add-on cards were required for different systems. An open standard and processor independence were required.
- Plug and Play capabilities, bus mastering were required. Auto detection and identification, etc.

The PCI Local Bus standard

- The PCI Local Bus is a high performance, 32-bit or 64-bit bus with multiplexed address and data lines.
- The bus is intended for use as an interconnect mechanism between highly integrated peripheral controller components, peripheral add-in boards, and processor/memory systems.
- The *PCI Local Bus Specification, Rev. 2.1* includes the protocol, electrical, mechanical, and configuration specification for PCI Local Bus components and expansion boards.
 - 2.1 ➔ 2.3 : support for 3v3 cards
 - 2.3 ➔ 3.0 : Towards 3v3 support (only), + errata and ECNs
- PCI system architecture and software model are unchanged for later generations of IO buses (PCI-X and PCI-express)

PCI Bus features

High Performance

- Transparent upgrade from
 - 32-bit data path at 33 MHz (132 MB/s peak)
to 64-bit data path at 33 MHz (264 MB/s peak) &
 - 32-bit data path at 66 MHz (264 MB/s peak)
to 64-bit data path at 66 MHz (528 MB/s peak).
- Variable burst length support for both read and writes.
- Low latency random accesses
- Synchronous bus with operation up to 33 MHz or 66 MHz
- Hidden (overlapped) central arbitration.

PCI Bus features

Low Cost

Optimized for direct silicon (component) interconnection; i.e., no glue logic. Electrical/driver (i.e., total load) and frequency specifications are met with standard ASIC technologies and other typical processes.

Multiplexed architecture reduces pin count (47 signals for target; 49 for master) and package size of PCI components, or provides for additional functions to be built into a particular package size.

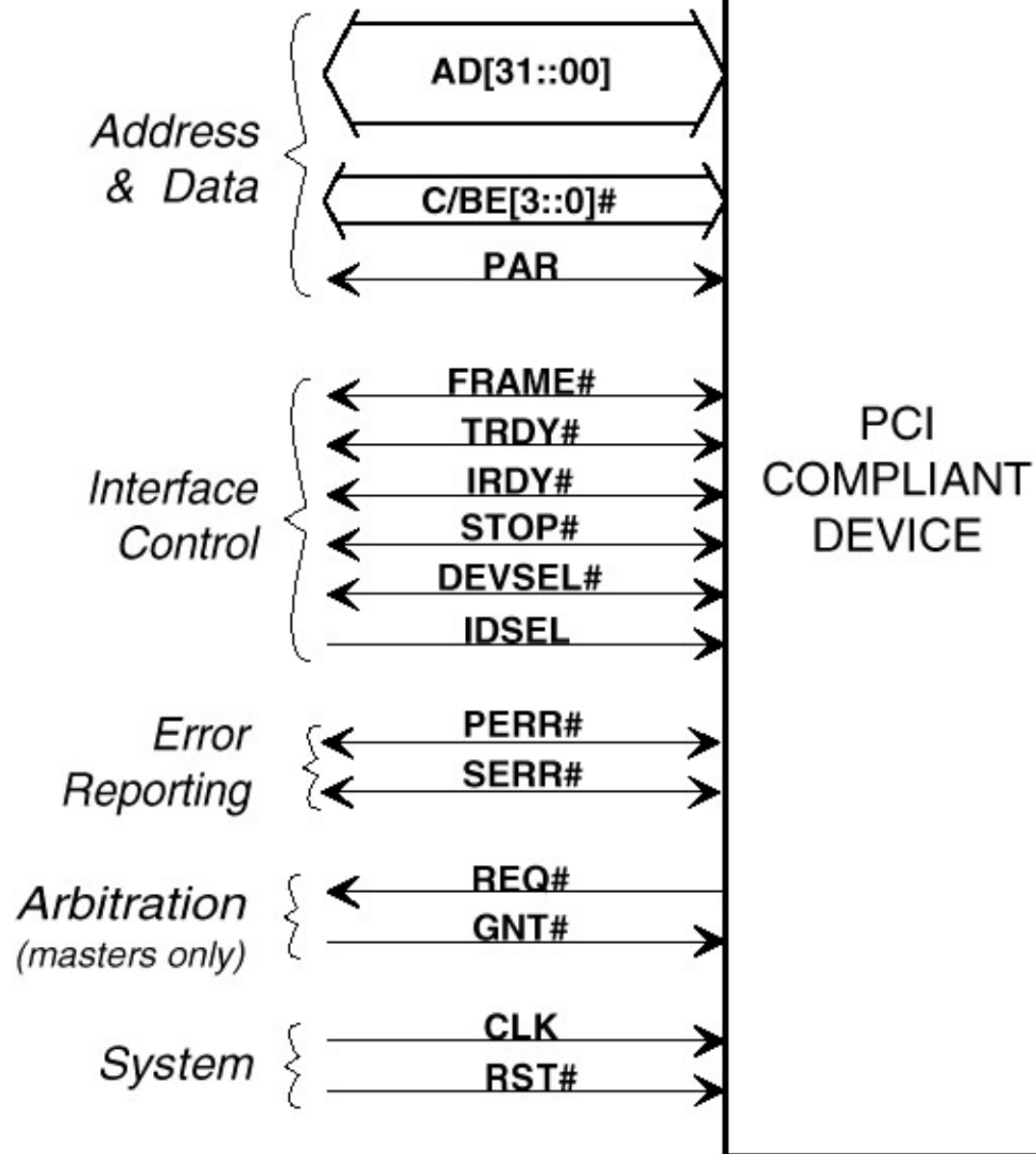
Single PCI add-in card works in different systems with minimal change to existing chassis designs reducing inventory cost and end user confusion.

PCI Bus features

Other benefits

- Ease of use - full auto configuration
- Longevity - Processor independence; 64 bit; 5V/3.3V
- Interoperability - forward and backward compatible for 64/32 bit and 33/66 MHz boards and components
- Support for multi-master and peer to peer transfers
- Data Integrity - Parity on both data and address.

Required Pins



indicates active low

Signal Type definition

- **in:** *Input* is a standard input-only signal.
- **out:** *Totem Pole Output* is a standard active driver.
- **t/s:** *Tri-State* is a bi-directional, tri-state input/output pin.
- **s/t/s:** *Sustained Tri-State* is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives an s/t/s pin low must drive it high for at least one clock before letting it float. A pull-up is required to sustain the inactive state until another agent drives it, and must be provided by the central resource.
- **o/d :** *Open Drain* allows multiple devices to share as a wire-OR.

System pins

- **CLK** (in) – Provides timing for all signals involved in PCI transactions except **RST#**, **INTA#**, **INTB#**, **INTC#**, and **INTD#**.
- **RST#** (in) – Brings all PCI devices to a consistent state. When reset is active, all output signals on PCI should be tri-stated. **AD**, **C/BE#** and **PAR** may be driven only low to avoid floating of these signals. **REQ64#** may be deasserted.

Address and data pins

- **AD[63:32]** and **AD[31:00]** (t/s) – Multiplexed address and data pins.
- **C/BE[7:4]#** and **C/BE[3:0]#** (t/s) – Command during address phase and Byte Enables during data phase.
- **PAR**(t/s) – Even Parity over AD and C/BE in both address and data phases. Parity is valid one clock after each address phase. For data, it is stable and valid one clock after either **IRDY#** is asserted on a write transaction or **TRDY#** is asserted on a read transaction.

Interface control pins

- **FRAME#** (s/t/s) - *Cycle Frame* is driven by the current master to indicate the beginning and duration of an access. When deasserted, the transaction is in final data phase or has completed.
- **IRDY#** (s/t/s) - *Initiator Ready*. During a write, **IRDY#** indicates that valid data is present on **AD[31::00]**. During a read, it indicates the master is prepared to accept data.
- **TRDY#** (s/t/s) - *Target Ready*. During a read, **TRDY#** indicates that valid data is present on **AD[31::00]**. During a write, it indicates the target is prepared to accept data.
- **STOP#** (s/t/s) - *Stop* indicates the current target is requesting the master to stop the current transaction.

Interface control pins

- **LOCK#** (s/t/s) – It indicates atomic operations to lock current target.

E.g. Reading, modifying semaphores

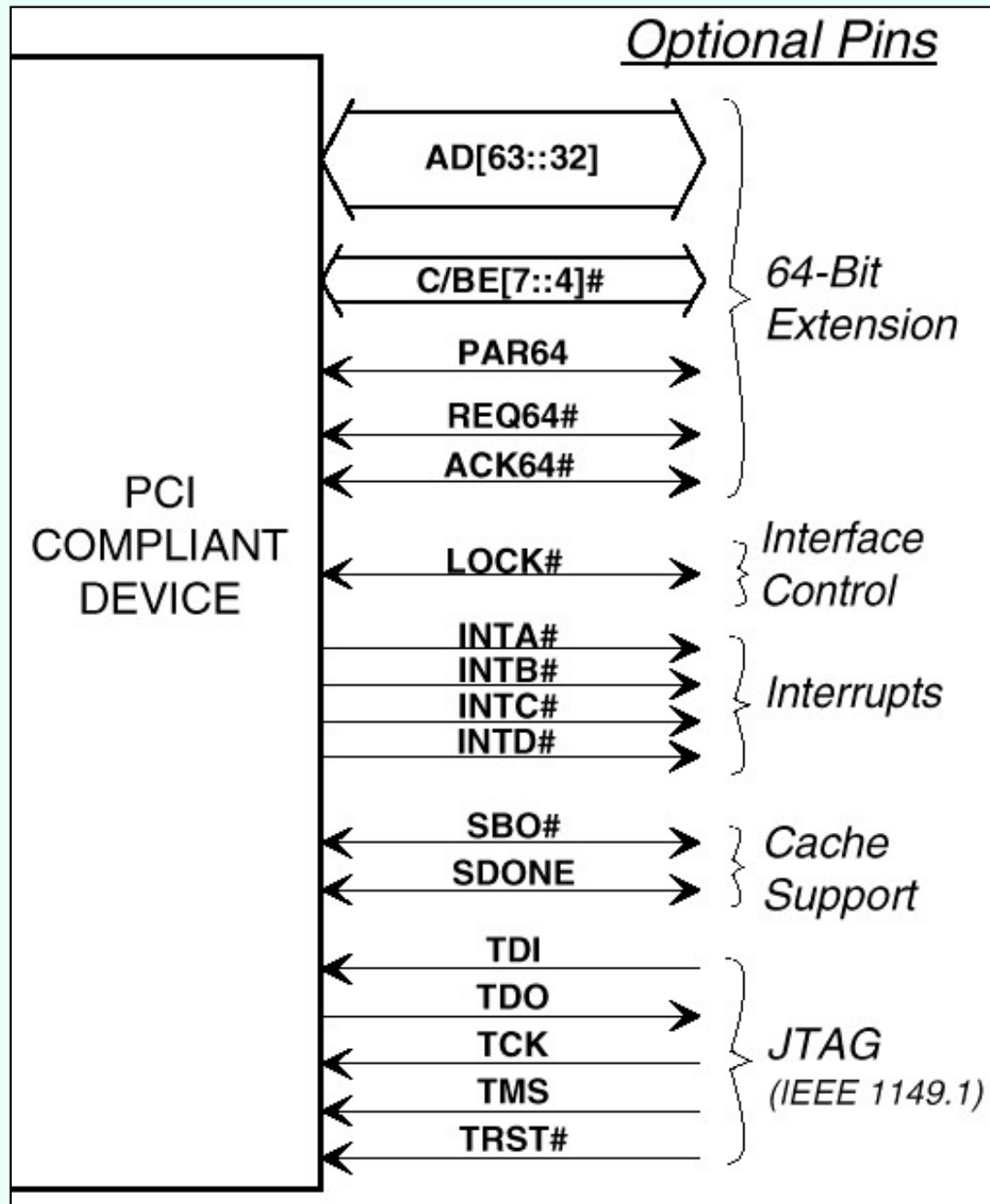
- **IDSEL** (in) - *Initialization Device Select* is used as a chip select during configuration read and write transactions.
- **DEVSEL#** (s/t/s) - *Device Select*, when actively driven, indicates the driving device has decoded its address as the target of the current access. As an input, indicates whether any device on the bus has been selected.

Arbitration pins (Bus Masters only)

- **REQ#** (t/s) – Request to the arbiter to use the bus. **REQ#** must be tri-stated while **RST#** is asserted.
- **GNT#** (t/s) – Indicates access to an agent has been granted. **GNT#** must be ignored while **RST#** is asserted.

Error reporting pins

- **PERR#** (s/t/s) – Reports data parity errors on all PCI transactions except special cycle. Reporting of this error may be implemented through settings in configuration space.
- **SERR#** (o/d) - Reports address parity errors, data parity errors on special cycle command or any other system error leading to catastrophic results.



Interrupts are level sensitive and asynchronous. Int B, C and D only for multifunction devices

64-bit support

- FRAME# \longleftrightarrow REQ64#
- AD[31:00] \longleftrightarrow AD[63:32]
- C/BE[3:0] \longleftrightarrow C/BE[7:4]
- DEVSEL \longleftrightarrow ACK64#
- PAR \longleftrightarrow PAR64

Additional pins

PRSNT[1:0]

PRSNT1#	PRSNT2#	Expansion Configuration
Open	Open	No expansion board present
Ground	Open	Expansion board present, 25W maximum
Open	Ground	Expansion board present, 15W maximum
Ground	Ground	Expansion board present, 7.5W maximum

M66EN - 66 MHz indicator

Different address spaces in PCI devices

- Memory space:
 - 2^{32} or 4GB (32-bit addressing)
 - 2^{64} (64-bit addressing).
- I/O space:
 - 2^{32} or 4GB (32-bit addressing)
 - 2^{16} or 64KB (most systems support not more than this range)
- Configuration space:
 - **256 bytes** (16 dwords + 48 dwords)
 - 64 bytes predefined header
 - 192 bytes user defined space
 - Only relevant registers are to be implemented in each part.

Basic Transfer Control

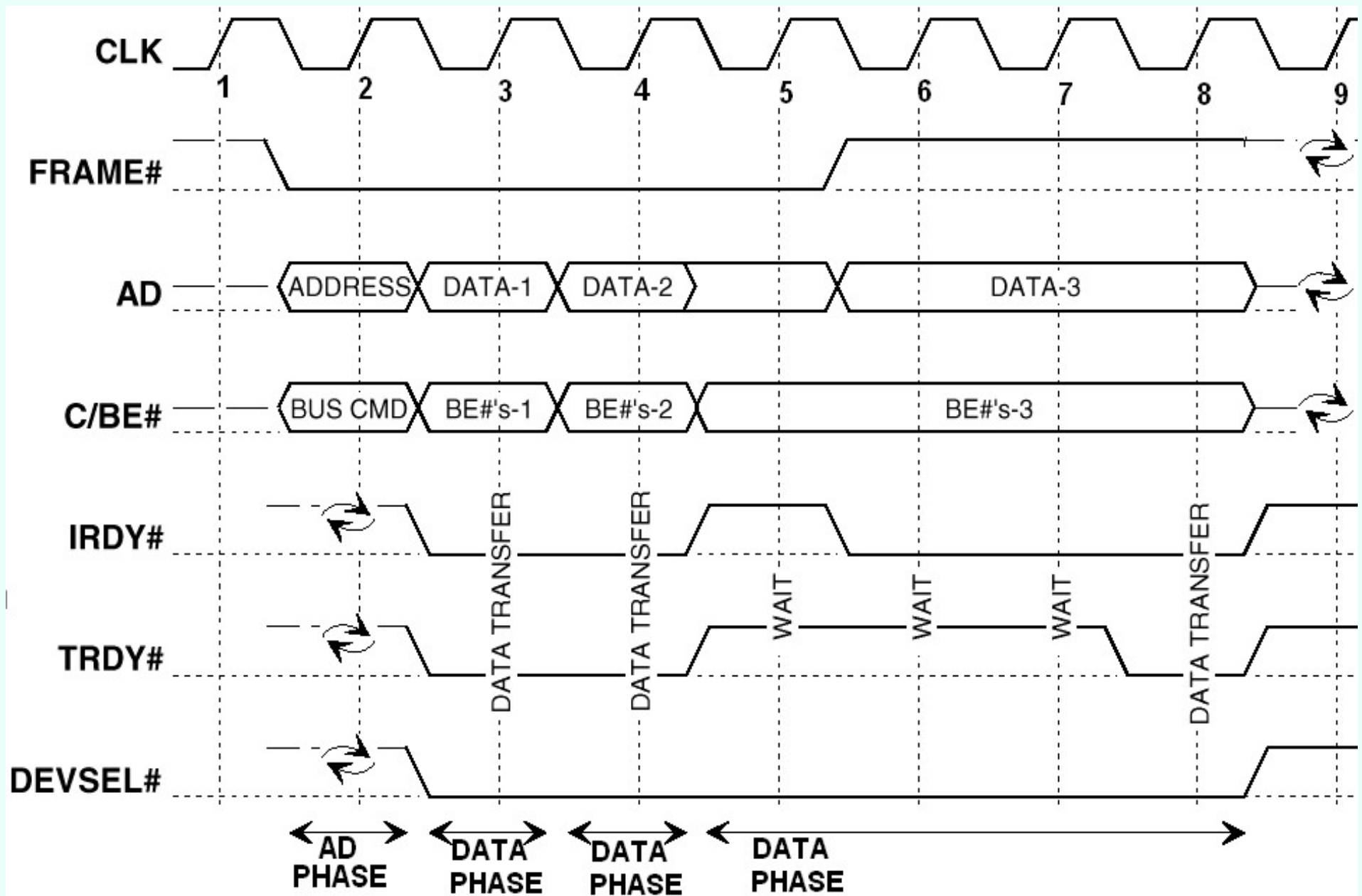
- The basic bus transfer mechanism on PCI is a burst. A burst is composed of an address phase and one or more data phases.
- PCI supports both memory and I/O address spaces.

The fundamentals of all PCI data transfers are controlled with three signals

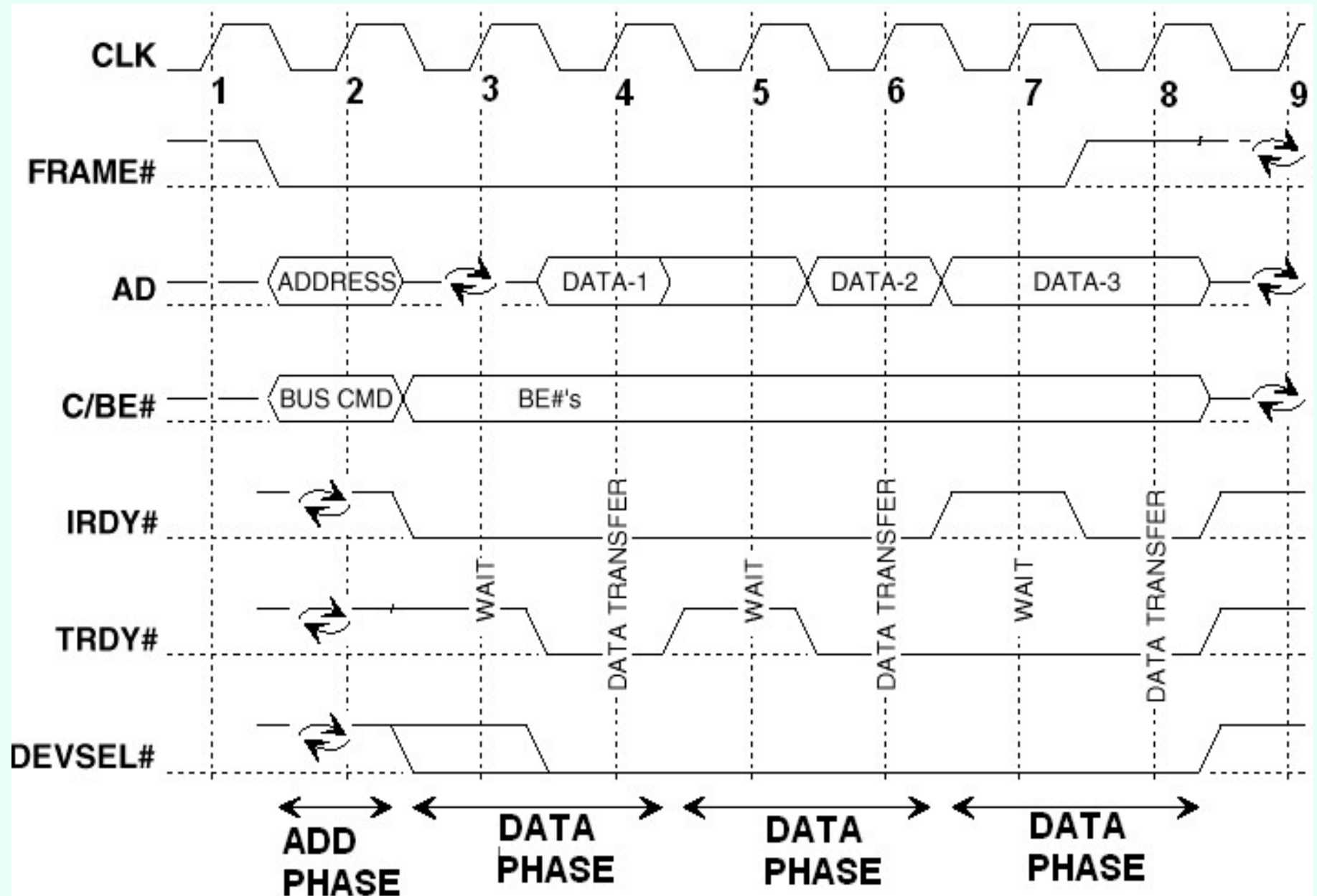
- **FRAME#** is driven by the master to indicate the beginning and end of a transaction.
- **IRDY#** is driven by the master to indicate that it is ready to transfer data.
- **TRDY#** is driven by the target to indicate that it is ready to transfer data.

Data transfer takes place whenever IRDY# and TRDY# are asserted.

Basic Write



Basic Read



Arbitration

- In order to minimize access latency, the PCI arbitration approach is access-based rather than time slot based.
- PCI uses a central arbitration scheme, where each master agent has a unique request (**REQ#**) and grant (**GNT#**) signal. A simple request-grant handshake is used to gain access to the bus.
- Arbitration is "hidden," which means it occurs during the previous access so that no PCI bus cycles are consumed due to arbitration, except when the bus is in an Idle state.

Latency

- PCI is a low latency, high throughput I/O bus. Both targets and masters are limited as to the number of wait states they can add to a transaction.
- Furthermore, each master has a programmable timer (Latency Timer specified in Configuration Header space) limiting its maximum tenure on the bus during times of heavy bus traffic.
- Given these two limits and the bus arbitration order, bus acquisition latencies can be predicted with relatively high precision for any PCI bus master.

Latency

- Target initial latency - the number of clocks from the assertion of **FRAME#** to the assertion of **TRDY#** - 16 clks
- Target subsequent latency – the number of clocks from the assertion of **IRDY#** and **TRDY#** for one data phase to the assertion of **TRDY#** or **STOP#** for the next data phase - 8 clks
- Master data latency - the number of clocks the master takes from the assertion of **FRAME#** to the assertion of **IRDY#** - 8 clks

Transaction Termination

Master Initiated Termination

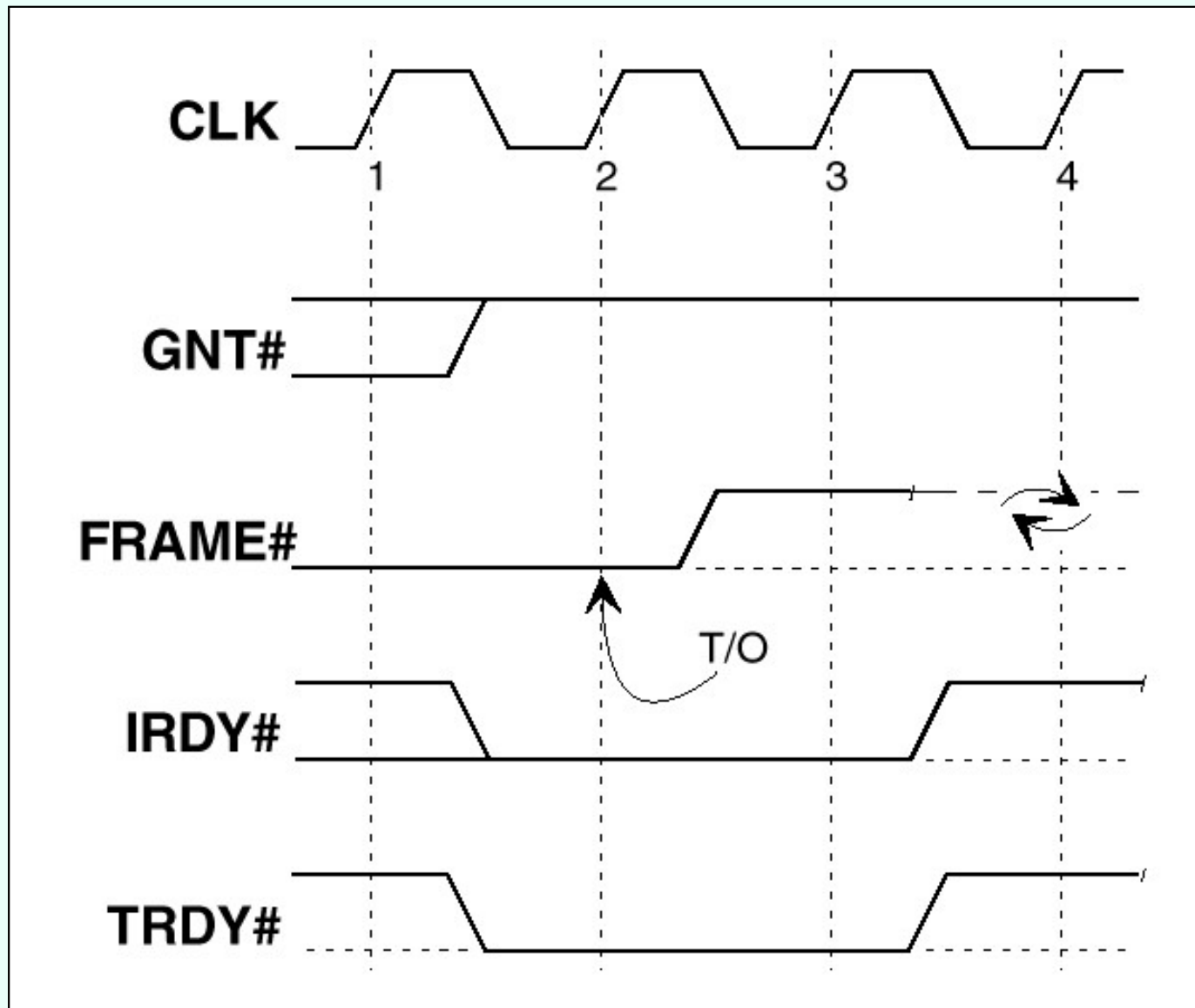
The master may terminate a transaction due to two reasons

- Completion
- Timeout

This will happen when the masters **GNT#** is de-asserted and the internal latency timer has expired.

- Master Abort - This is an abnormal case. It may indicate no target device has responded to the address.

Master Initiated Termination



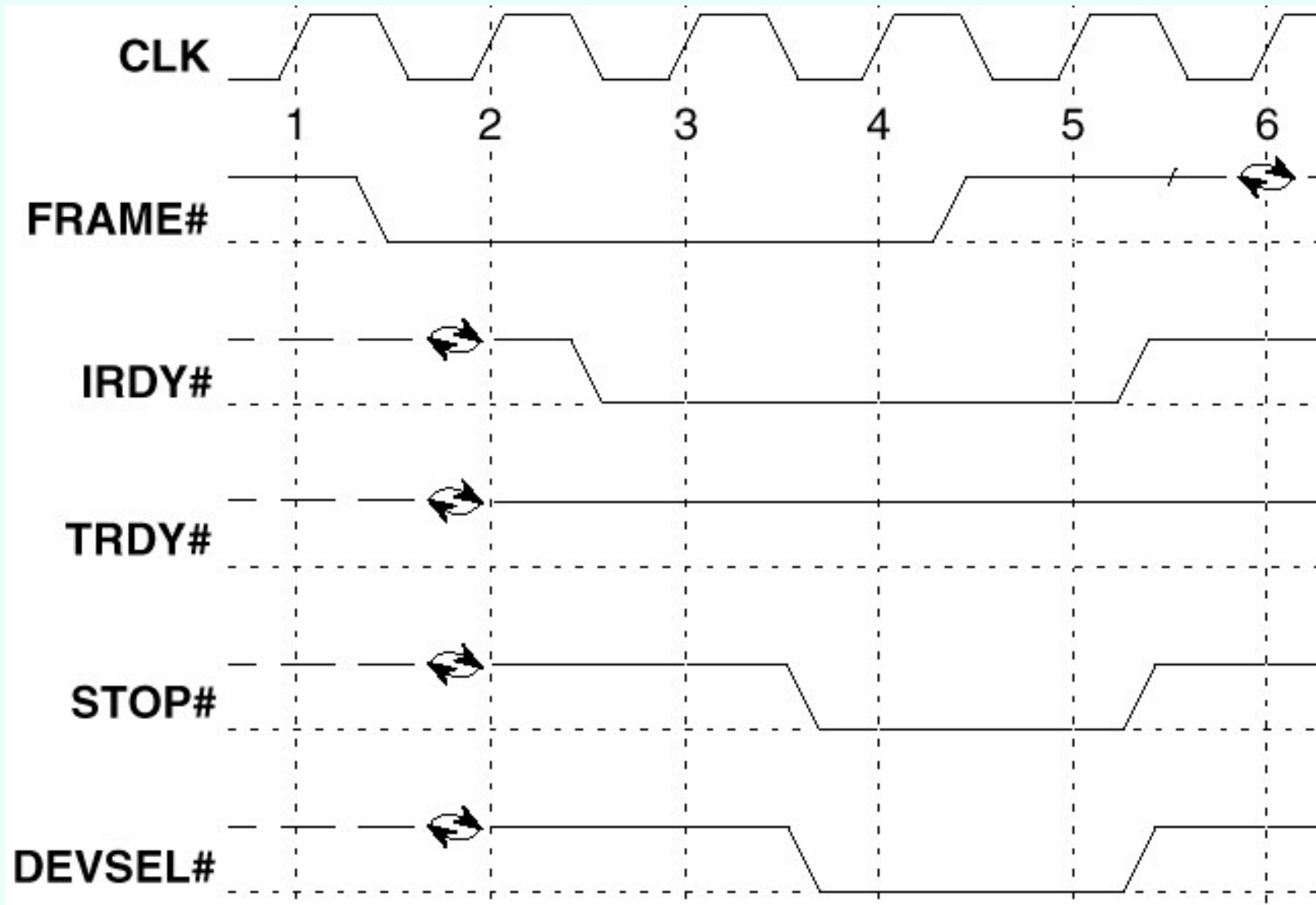
Target Initiated Termination

The target may terminate a transaction due to three reasons; Retry, Disconnect or Target abort

- **Retry** - refers to termination requested before any data is transferred because the target is busy and temporarily unable to process the transaction. This condition may occur, for example, because the device cannot meet the initial latency requirement, or there is a conflict for a internal resource or locked bridge.

The target signals Retry by asserting **STOP#** and not asserting **TRDY#** on the initial data phase of the transaction. When the target uses Retry, no data is transferred.

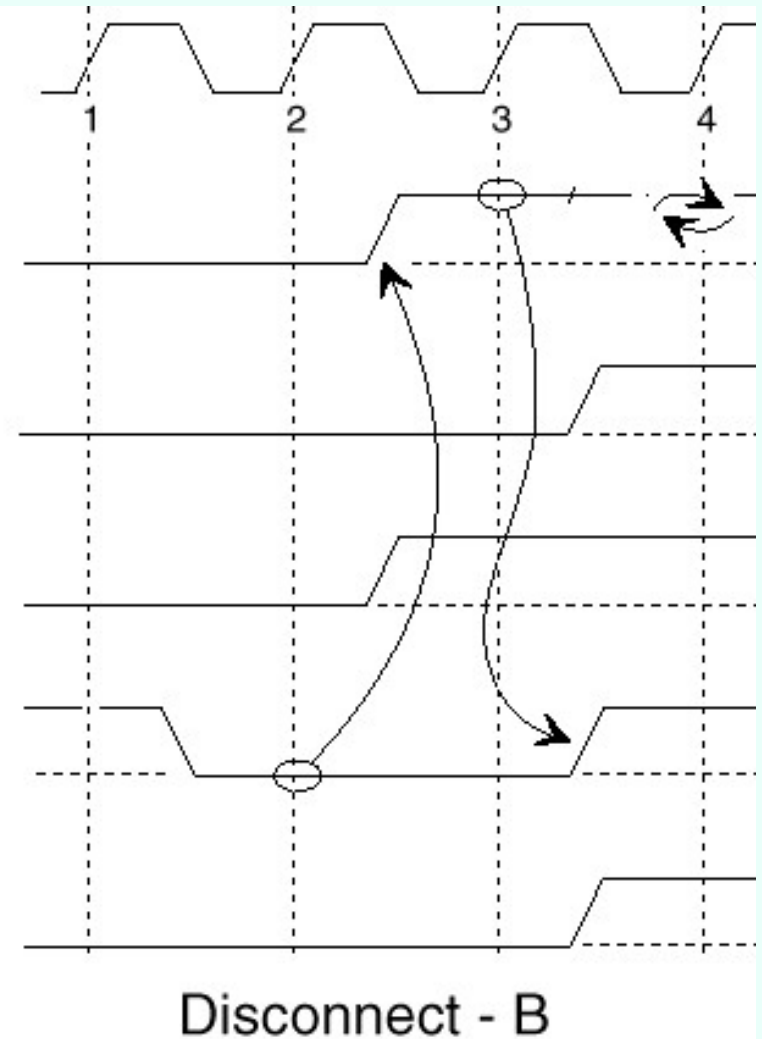
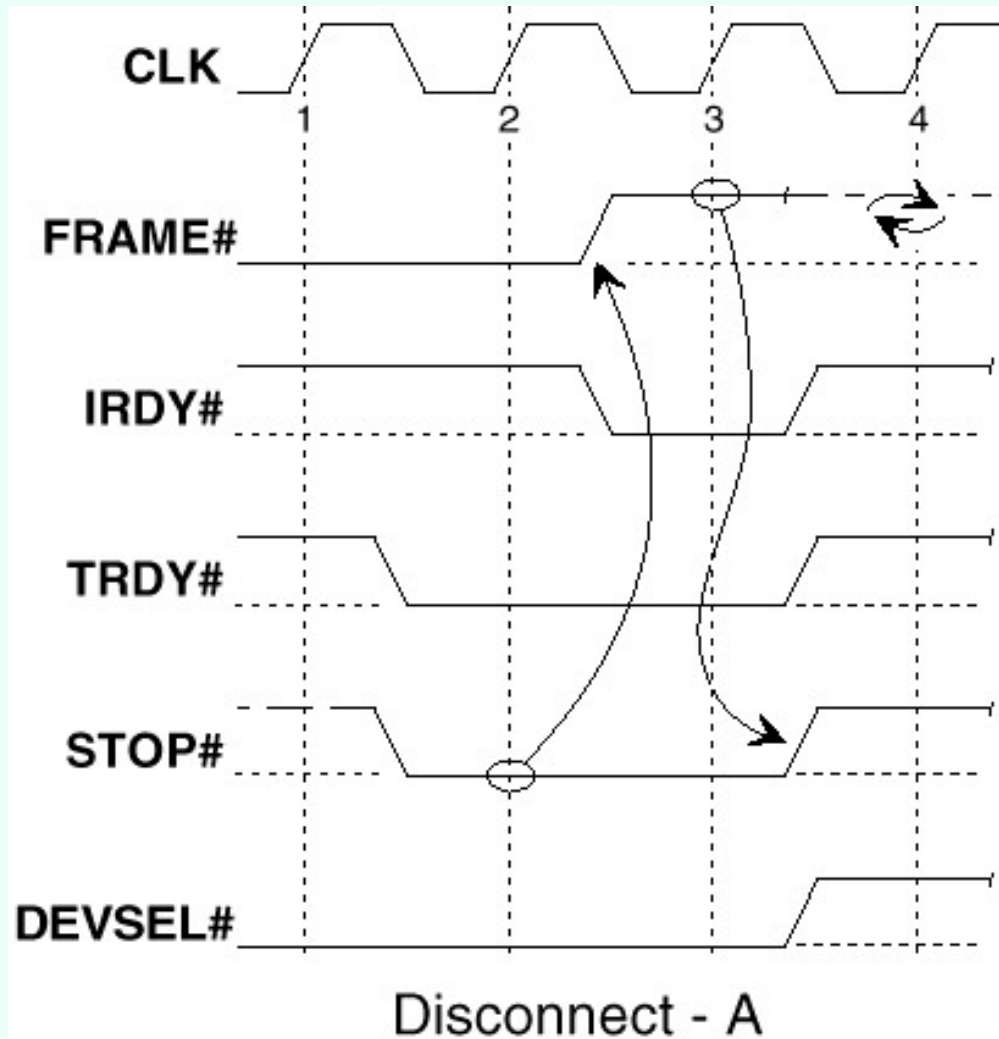
Target Retry



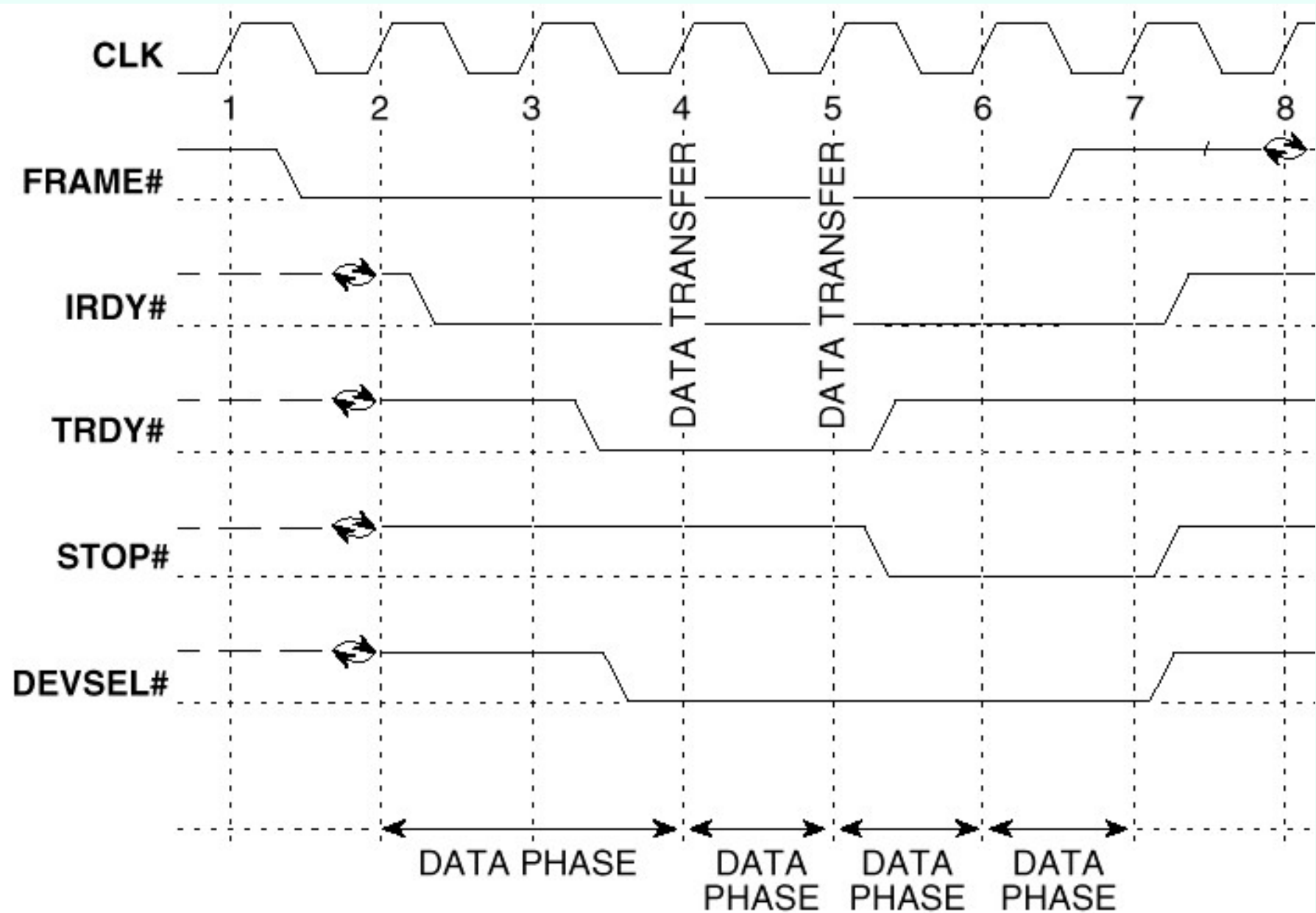
Target Disconnect

- Refers to termination requested with or after data was transferred on the initial data phase because the target is unable to respond within the target subsequent latency requirement, and, therefore, is temporarily unable to continue bursting.
- Disconnect with data may be signaled on any data phase by asserting **TRDY#** and **STOP#** together. This termination is used when the target is only willing to complete the current data phase and no more.
- Disconnect without data may be signaled on any subsequent data phase (meaning data was transferred on the previous data phase) by de-asserting **TRDY#** and asserting **STOP#**.

Disconnect with data



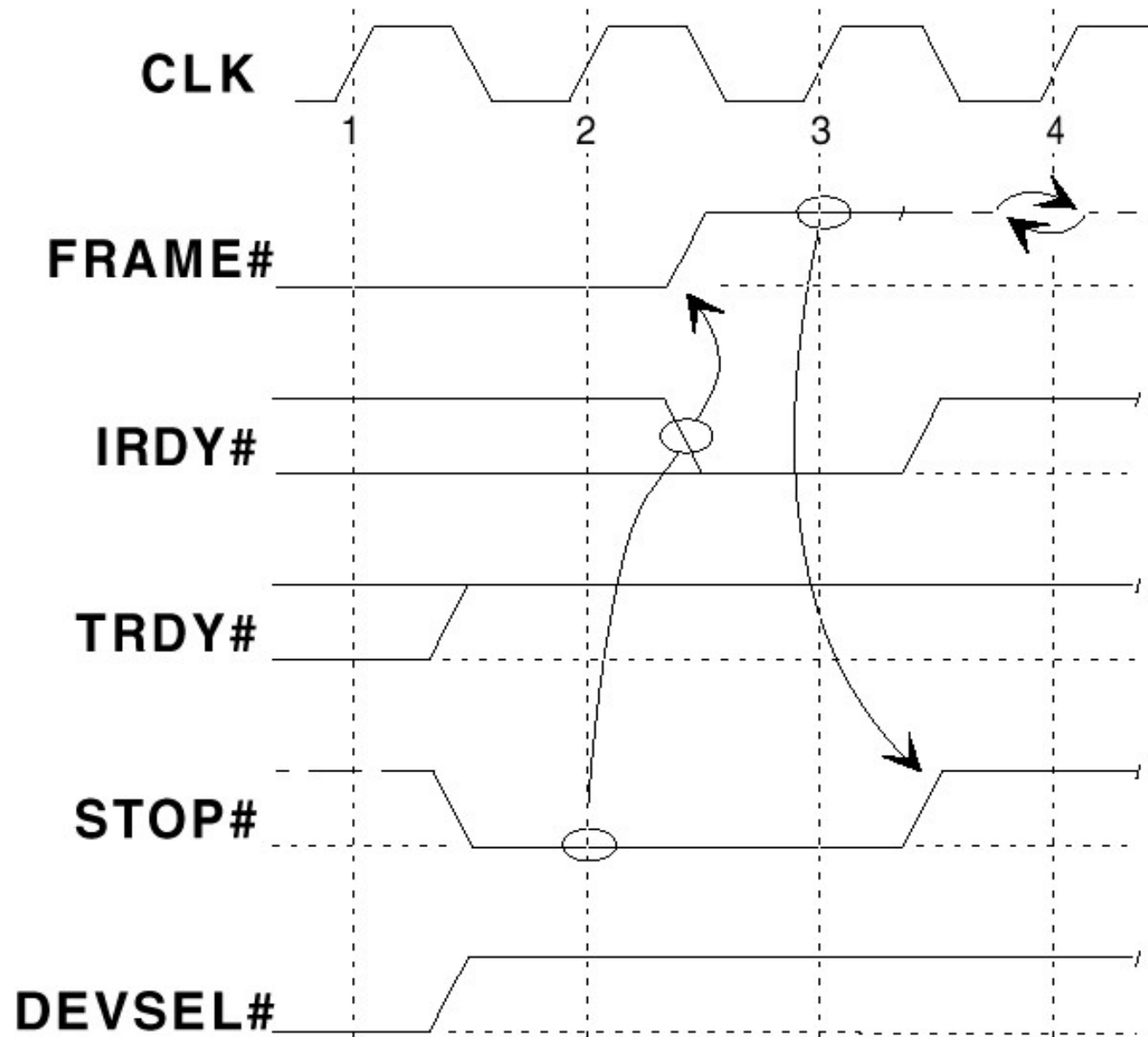
Disconnect without data



Target Abort

- Indicates the target requires the transaction to be stopped and does not want the master to repeat the request again.
- To signal Target-Abort, **TRDY#** must be deasserted when **DEVSEL#** is deasserted and **STOP#** is asserted. If any data was transferred during the previous data phases of the current transaction, it may have been corrupted.

Target Abort



Error functions

- Parity error
 - The four command lines and byte enables are included in the parity calculation.
 - The agent that is responsible for driving **AD[31::00]** on any given bus phase is also responsible for driving **even** parity on **PAR**.
- Parity error

SERR# is asserted when

 - Address parity error or data parity error on Special Cycles is detected.
 - A catastrophic error
 - e.g. DMA abort

Need for configuration

- PCI devices implement one or more PCI functions i.e. logical devices (up to 8 max)
- Each function requires some resource allocation.
e.g. IRQ, memory space
- Configuration enables PCI devices to transfer data to and from other devices efficiently using the information stored in its configuration space.
- A PnP hardware-software combination can facilitate auto configuration of PCI devices saving the user time and efforts in manual configuration of devices.

Need for configuration (contd)

The configuration space is required for

- Device Identification
- Device control/status
- Base Address registers
- Misc.

The configuration space provides ease of use and control over devices to the system software. The system software can easily identify devices, learn about address space requirements, allocate addresses, etc.

Accessing the configuration space

- Before the devices are configured, they do not have unique addresses. So a special scheme is required to access the devices.
- IDSEL line is used by the PCI controller for accessing every device on the bus. (**Note: IDSEL is not bussed**)
- The software can access the configuration space by using BIOS calls, and providing the Device ID, Vendor ID, etc as inputs.
- The software reads the configuration space and determines the addresses of devices and then these addresses are used to access the devices.

3		2		1		0		
Device ID				Vendor ID				00h
Status				Command				04h
Class Code						Revision ID		08h
BIST	Header Type		Latency Timer		Cache Line Size			0Ch
Base Address Registers (0 to 5)								10h
Cardbus CIS Pointer								28h
Subsystem ID				Subsystem Vendor ID				2Ch ←
Expansion ROM Base Address								30h
Reserved								34h
Reserved						Capabilities Pointer		38h ←
Max_Lat	Max_Gnt		Interrupt Pin		Interrupt Line			3Ch

← 2.2

← 2.2

Configuration Space

- Device Identification
 - ✓ VendorID: PCI-SIG assigned
 - ✓ DeviceID: Vendor self-assigned
 - ✓ Subsystem VendorID: PCI-SIG
 - ✓ Subsystem DeviceID: Vendor
- Address Decode controls
 - ✓ Software reads/writes BARs to determine required size and maps appropriately
 - ✓ Memory, I/O, and bus-master enables
- Other bus-oriented controls

Byte				Doubleword Number (in decimal)
3	2	1	0	
Device ID		Vendor ID		00
Status Register		Command Register		01
Class Code			Revision ID	02
BIST	Header Type	Latency Timer	Cache Line Size	03
Base Address 0				04
Base Address 1				05
Base Address 2				06
Base Address 3				07
Base Address 4				08
Base Address 5				09
CardBus CIS Pointer				10
Subsystem ID		Subsystem Vendor ID		11
Expansion ROM Base Address				12
Reserved			Capabilities Pointer	13
Reserved				14
Max_Lat	Min_Gnt	Interrupt Pin	Interrupt Line	15

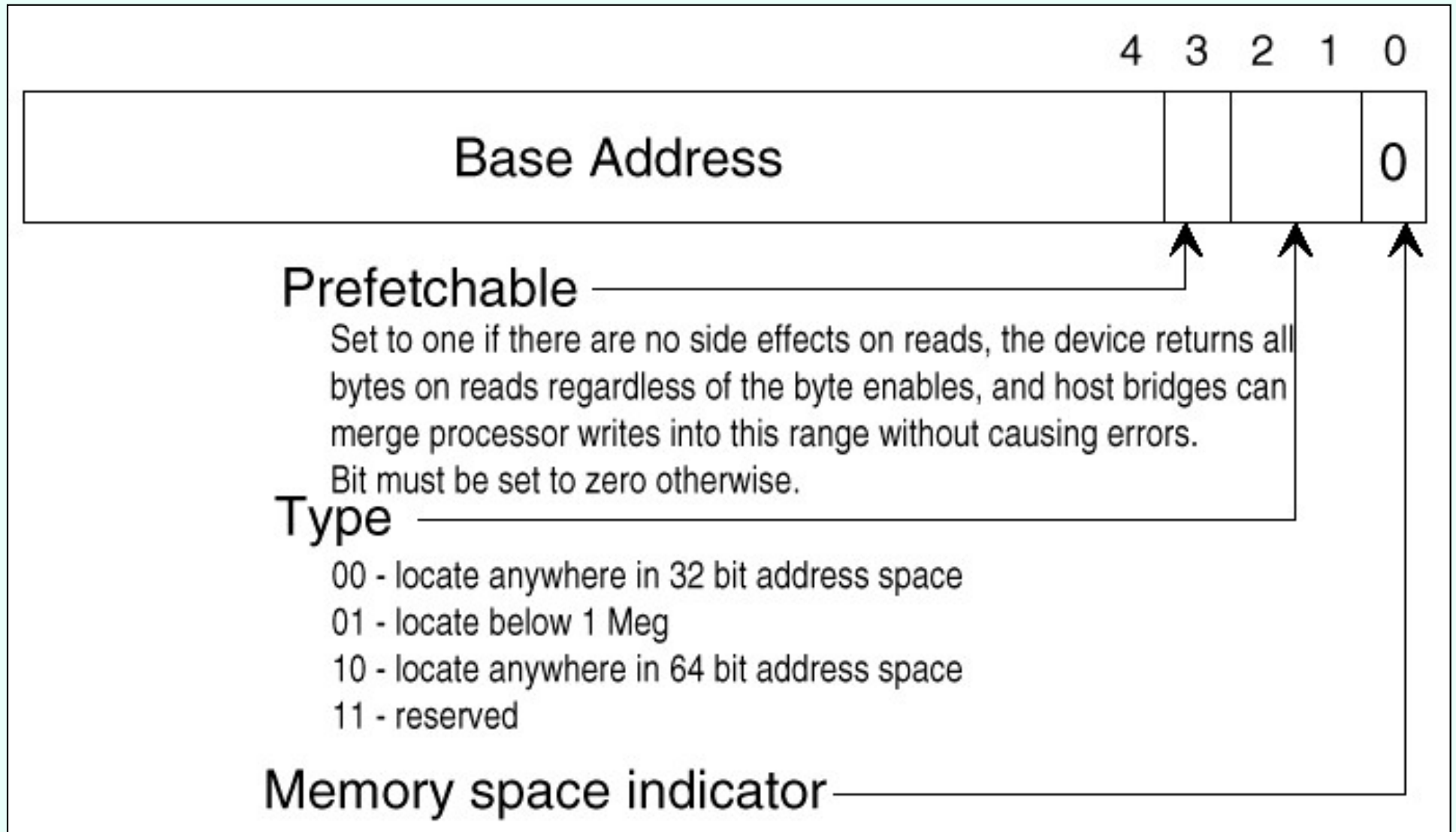
Base Address Registers (24-bytes)

- Required for devices that implement memory and/or IO decoders.
- A device may be located in memory and/or IO space.
- It is recommended to use memory mapping since IO space is much crowded and some processors do not support IO transactions.
- 16 bytes is the smallest memory block a PCI memory decoder can be designed for.
- 4 bytes is the smallest IO block a PCI IO decoder can be designed for.

Address allocation

- The devices will write their address space requirements into base registers.
- The BIOS will read the registers in the configuration space during initial configuration.
- It will then reserve the space for the device, in the physical address map, and write back the addresses to the base address registers. These addresses can then be read by the device itself and also by system software for accessing the devices.

Base Address



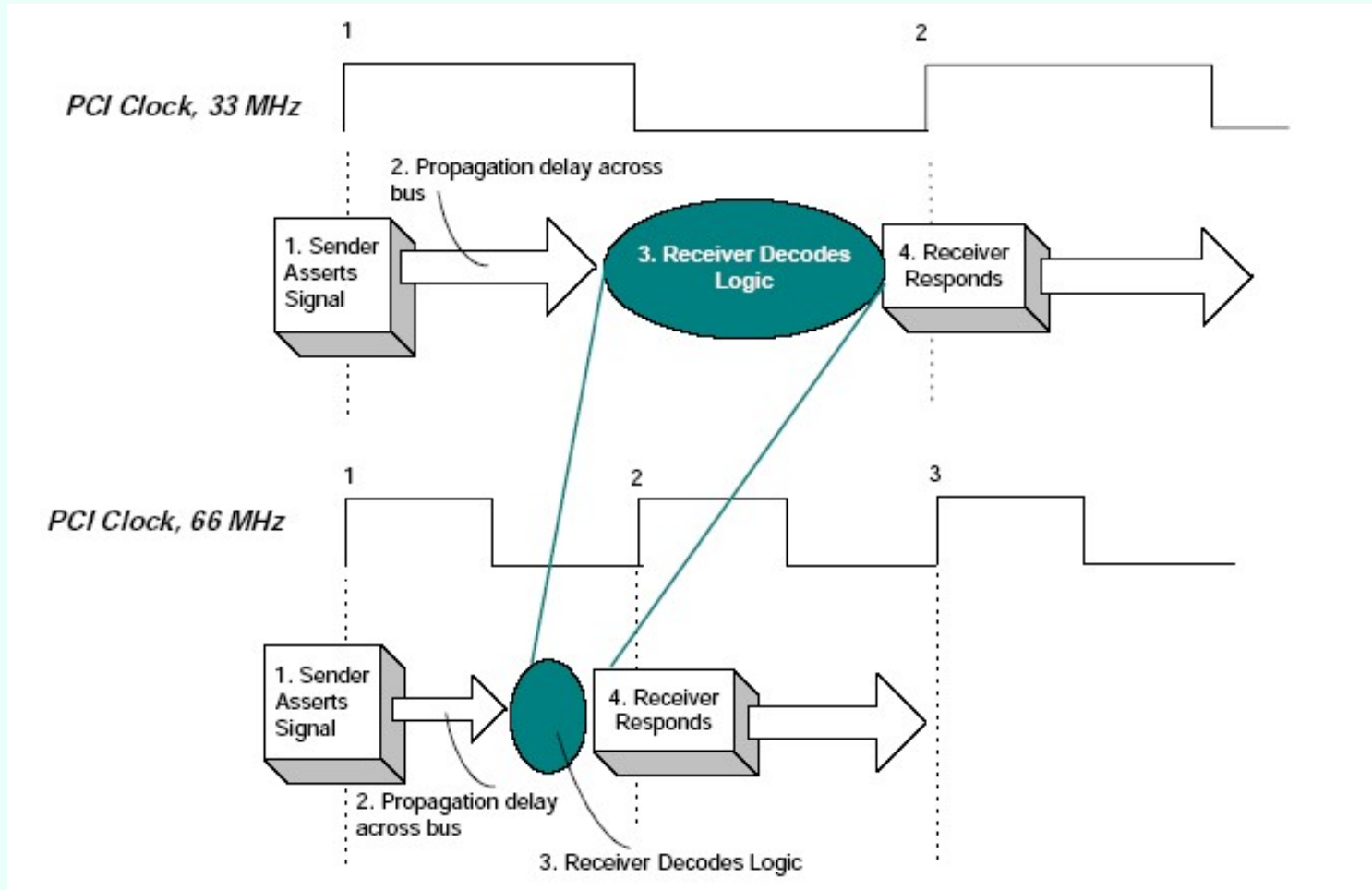
The BIOS writes all 1's to this location and reads back. The result indicates the memory requirements.

PCI - X Overview

PCI-X 1.0 Overview

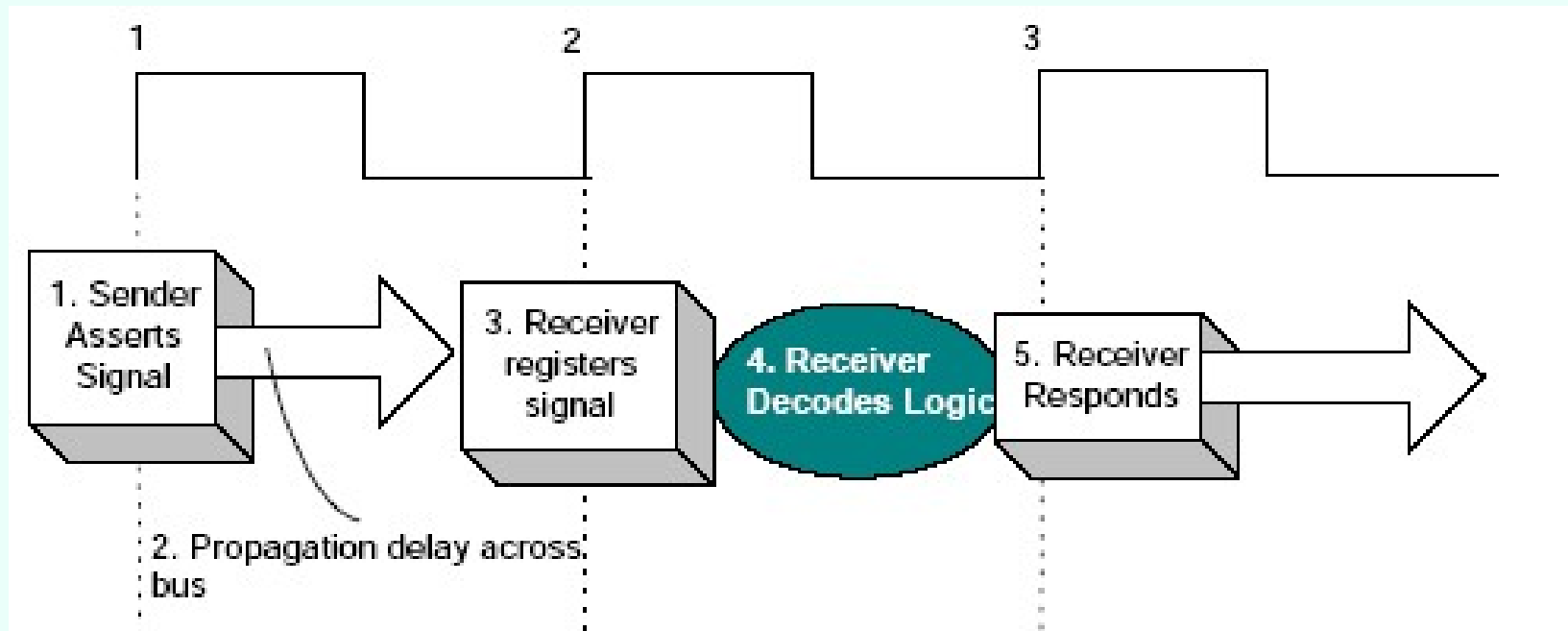
- Evolutionary I/O upgrade to conventional PCI.
- Bus capacity increased to more than 8 times from 133 MB/s with 32 bit 33 MHz PCI to 1066MB/s (~1GB/s)with 64 bit 133MHz PCI-X
- Bus is faster but timing easier because of registered protocol
- ECC Introduced in this version
- Attributed transactions
- 100 % Backward compatible with PCI.

Limitations of PCI



With conventional PCI, the time available to decode a transaction decreases as the bus frequency increases from 33 MHz to 66 MHz

Registered protocol



- The PCI-X protocol allows an entire clock cycle for the decode logic to occur.
- The net difference is that PCI-X transactions generally require one clock cycle more than conventional PCI transactions.
- A write transaction that completes in nine clock cycles for conventional PCI will complete in ten clock cycles for PCI-X(6 data phases and no wait states)
- System designers benefit from eased timing constraints.

PCI-X 1.0 Overview

- Enhancements for bus efficiency
 - Attribute phase
 - Delayed transactions replaced by split transactions
 - Optimized wait states
 - Standard block size movements

Attribute phase

- The PCI-X protocol includes a new transaction phase called the attribute phase.
- The attribute phase uses a 36-bit attribute field that describes bus transactions in more detail than the conventional PCI specification allows.
- It follows immediately after the address phase and contains several bit assignments that include information about the size of the transaction, ordering of transactions, cache snooping requirements, and the identity of the transaction initiator.
- Enhancements included in Attribute phase are
 - Relaxed ordering
 - Non-cache-coherent transactions
 - Transaction byte count
 - Sequence number

Split Transaction support

Conventional PCI protocol supports delayed transactions. With a delayed transaction, the device requesting data must poll the target to determine when the request has been completed and its data is available.

With a split transaction as supported in PCI-X, the device requesting the data sends a signal to the target. The target device informs the requester that it has accepted the request.

The requester is free to process other information until the target device initiates a new transaction and sends the data to the requester. Thus, split transactions enable more efficient use of the bus.

Bus efficiency of Read almost as good as Write

Optimized wait states

Conventional PCI devices often add extra clock cycles, or wait states, into their transactions. The wait states are added to “stall” the bus if the PCI device is not ready to proceed with the transaction.

This can slow bus throughput dramatically.

PCI-X eliminates the use of wait states, except for initial target latency. When a PCI-X device does not have data to transfer, it will remove itself from the bus so that another device can use the bus bandwidth.

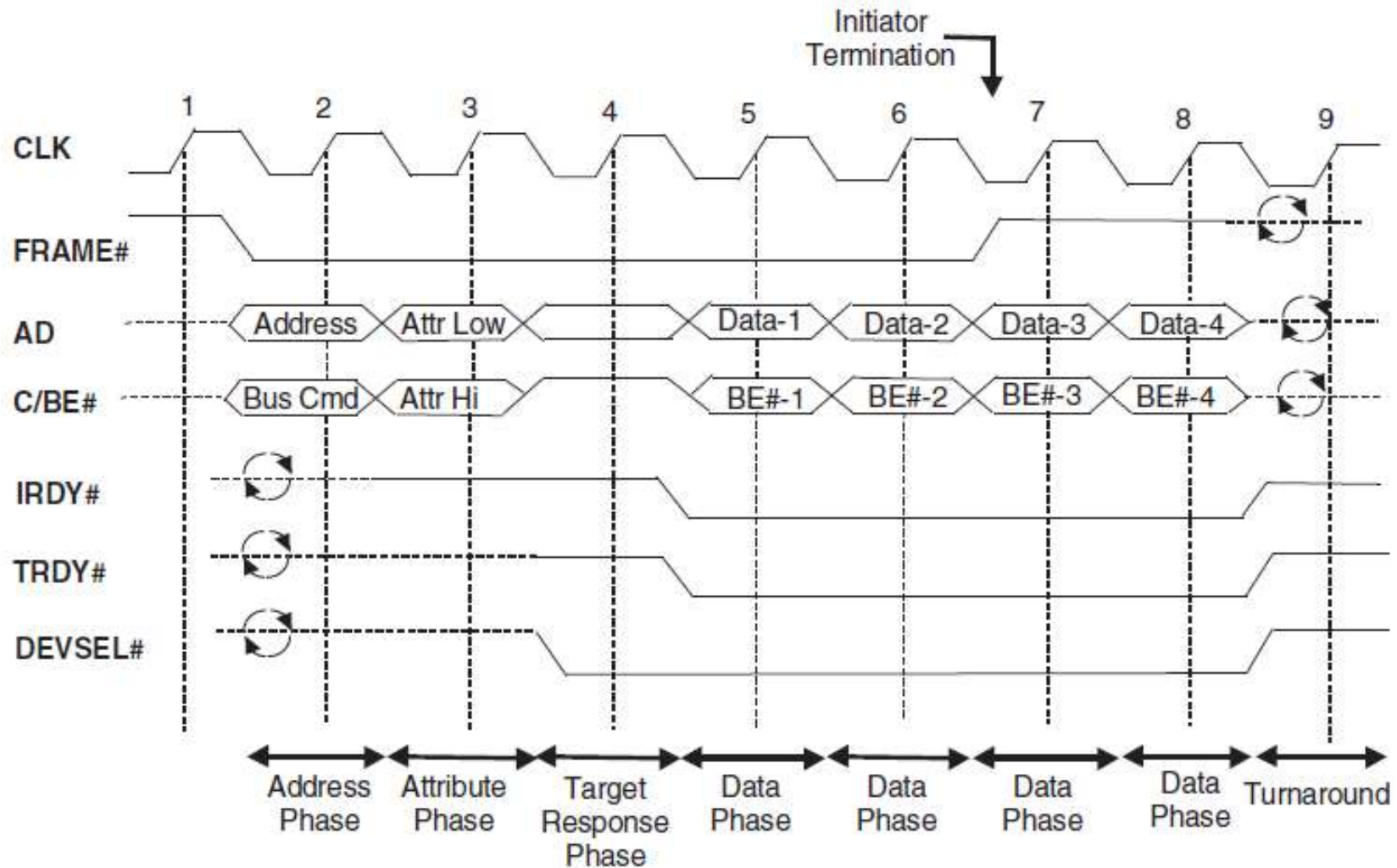
This provides more efficient use of bus and memory resources.

Standard Block size

With PCI-X, adapters and bridges (host-to-PCI-X and PCI-X to PCI-X) are permitted to disconnect transactions only on naturally aligned 128-byte boundaries. This encourages longer bursts and enables more efficient use of cache-line-based resources such as the processor bus and main memory.

It also facilitates a more pipelined architecture within PCI-X devices.

Basic Write



Attribute phase. It is always the clock immediately following Address Phase.

PCI and PCI-X 2.0 performance

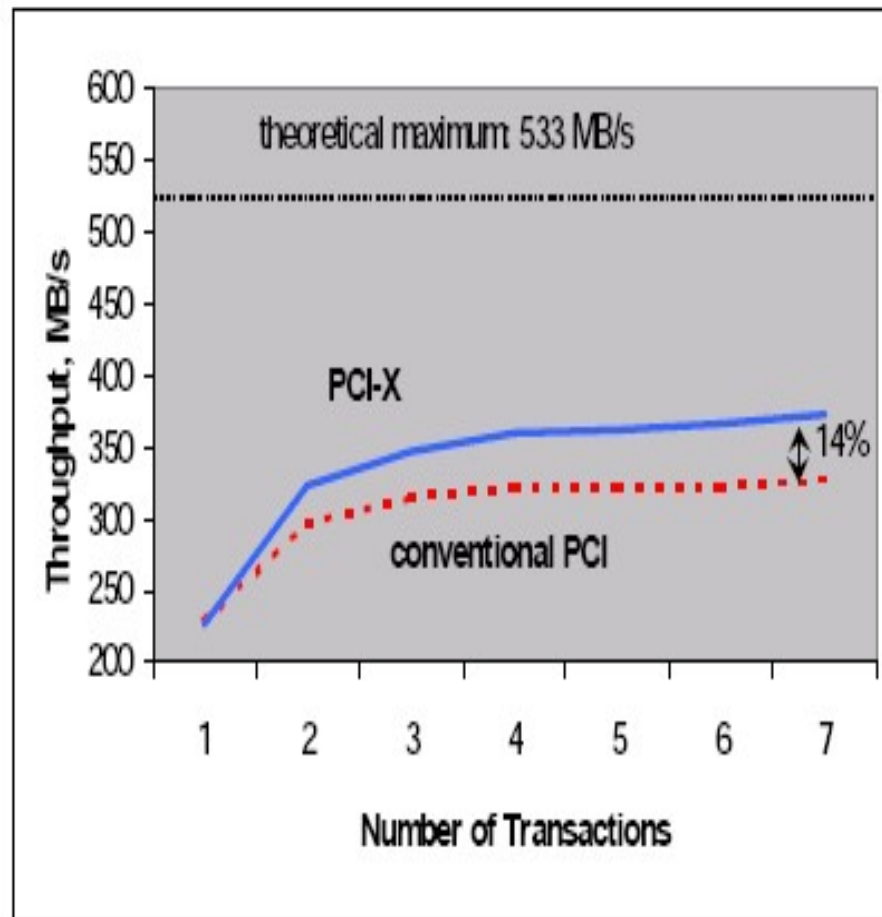


Figure 6: System-level comparison of PCI and PCI-X buses, both using 512-byte reads on a 64-bit, 66-MHz bus.

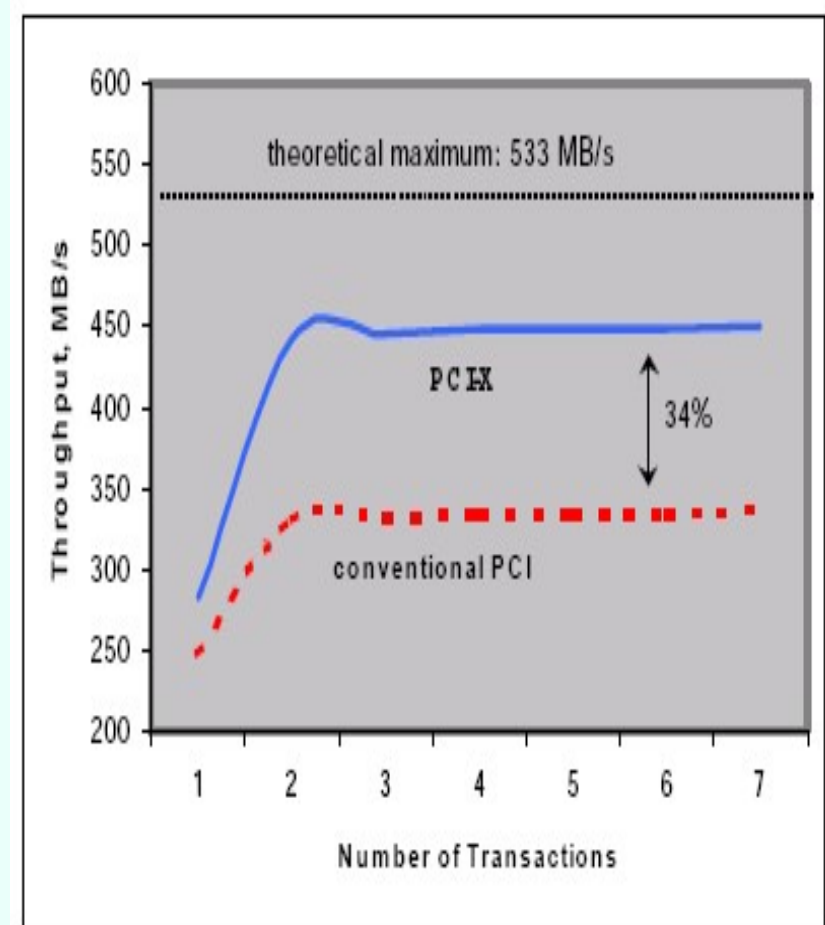
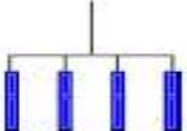
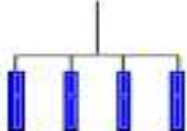
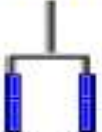

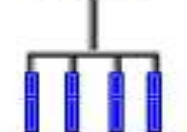
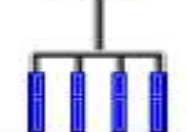
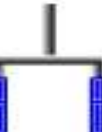
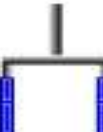








Figure 7: System-level comparison of PCI and PCI-X buses, both using 4-KB reads on a 64-bit, 66-MHz bus.

PCI-X Roadmap

Mode	V _{I/O}	64-Bit		32-Bit		16-Bit
		Slots	MB	Slots	MB	
PCI 33	5V/3.3V		266		133	N/A
PCI 66*	3.3V		533		266	N/A
PCI-X 66	3.3V		533		266	N/A
PCI-X 133 (operating at 100 MHz)	3.3V		800		400	N/A
PCI-X 133	3.3V		1066		533	N/A
PCI-X 266	1.5V		2133		1066	533
PCI-X 533	1.5V		4266		2133	1066

PCI Express

Brief

- ❖ Introduction
- ❖ Basic PCIe Topology
- ❖ PCIe Architecture
- ❖ Transaction Layer
- ❖ Data Link Layer
- ❖ Physical Layer
- ❖ Software Layer
- ❖ QoS Features
- ❖ Case Study

Introduction

- High performance, general purpose I/O interconnect defined for wide variety of future computing/communication platforms
- Evolved from PCI and PCI-X architectures

Yet PCI Express architecture is significantly different from its predecessors PCI and PCI-X
- Serial point-to-point interconnect
- Implements packet based protocol for information transfer

PCI Evolution

- **1993 Conventional PCI**
- **1998 PCI-X**
- **2003 PCI Express Gen1**
- **2007 PCI Express Gen2**
- **2010 PCI Express Gen3**
- **2017 PCI Express Gen4**

PCI-SIG formed in 1992

Difference PCI/PCI-X and PCI-XP

PCI/PCI-X

Multidrop, Parallel interconnect

Many devices share one bus

Communication Protocol
Packet based

BW

32 bit / 33MHz - 133MBps

64 bit / 66MHz - 533MBps

Layer Implementation
Physical layer only

PCI-XP

Point to point, Serial interconnect

Devices communicate via switches

Switch based

x1 - 500MBps

x32 - 16GBps

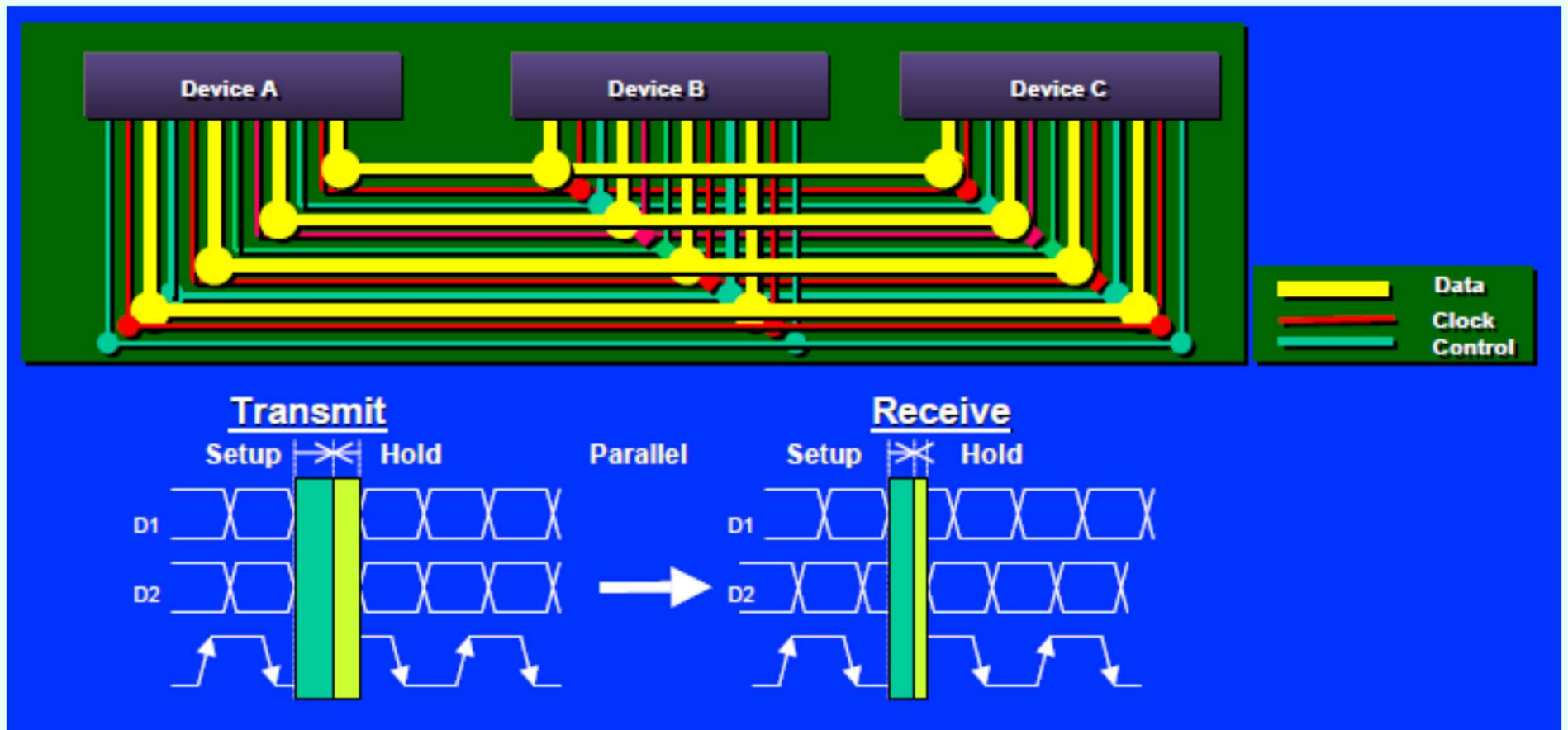
Upto Transaction layer

Limitations of PCI Architecture

PCI Bus has reduced efficiency because of following reasons

- Master and target devices introduce wait states
- PCI bus cycles do not indicate transfer size.
 - This makes buffer management inefficient
- Delayed transactions introduce latency.
- Shared interrupt signal, so s/w latency for deciding source
- Applications such as video on demand, Audio streaming demand very high bandwidth

Limitations of PCI Architecture



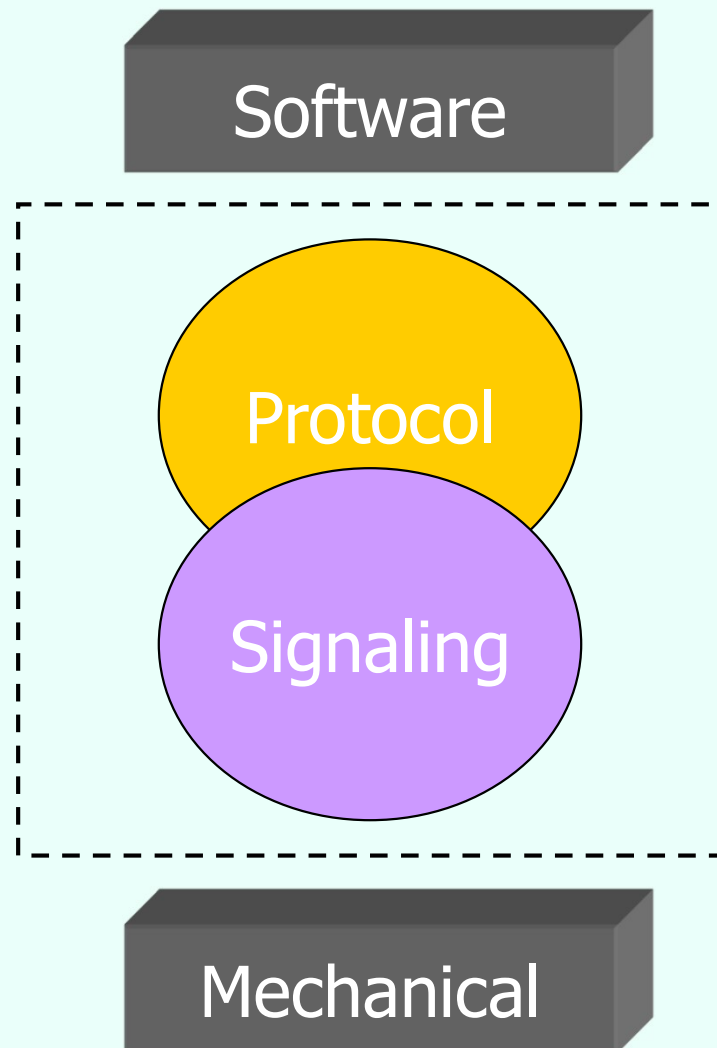
Limitations of PCI Architecture

- Multidrop parallel interconnect bus
Many devices share single bus
- Increase in power dissipation
- Stringent routing rules
- Limited scaling with frequency and voltage
- Difficult to clock-to-data skew management
- Lack of advanced features

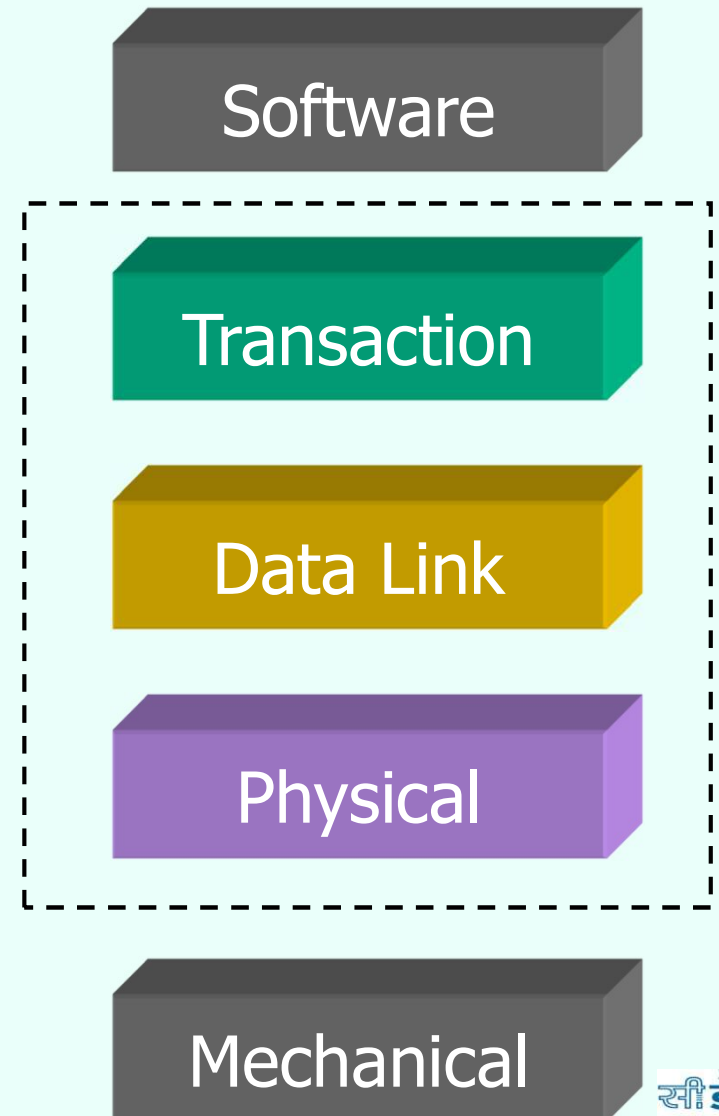
Parallel I/O have reached practical limits

Layered Architecture

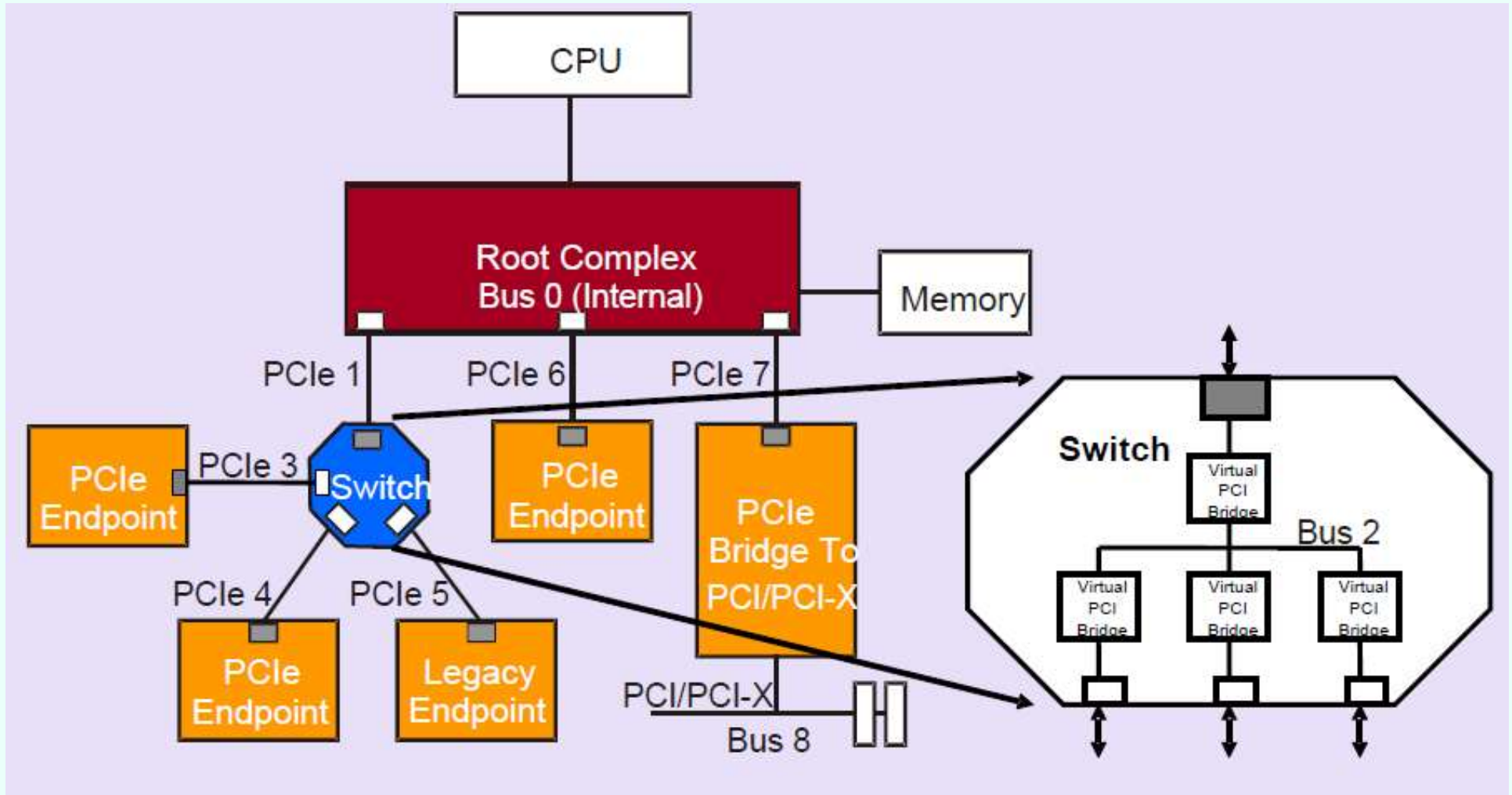
PCI



PCI Express



PCI Express Topology



Root Complex, Switch and Endpoint

Root Complex

A Root Complex (RC) denotes the root of an I/O hierarchy that connects the CPU/memory subsystem to the I/O

Each interface defines a separate hierarchy domain.

Each hierarchy domain composed of

- A single Endpoint

OR

- A sub-hierarchy containing one or more Switch components and Endpoints.

RC Supports

- One or more PCI Express Ports.
- Generation of configuration requests as a Requester.
- The generation of I/O requests as a Requester.
- Root complex is permitted to split packet

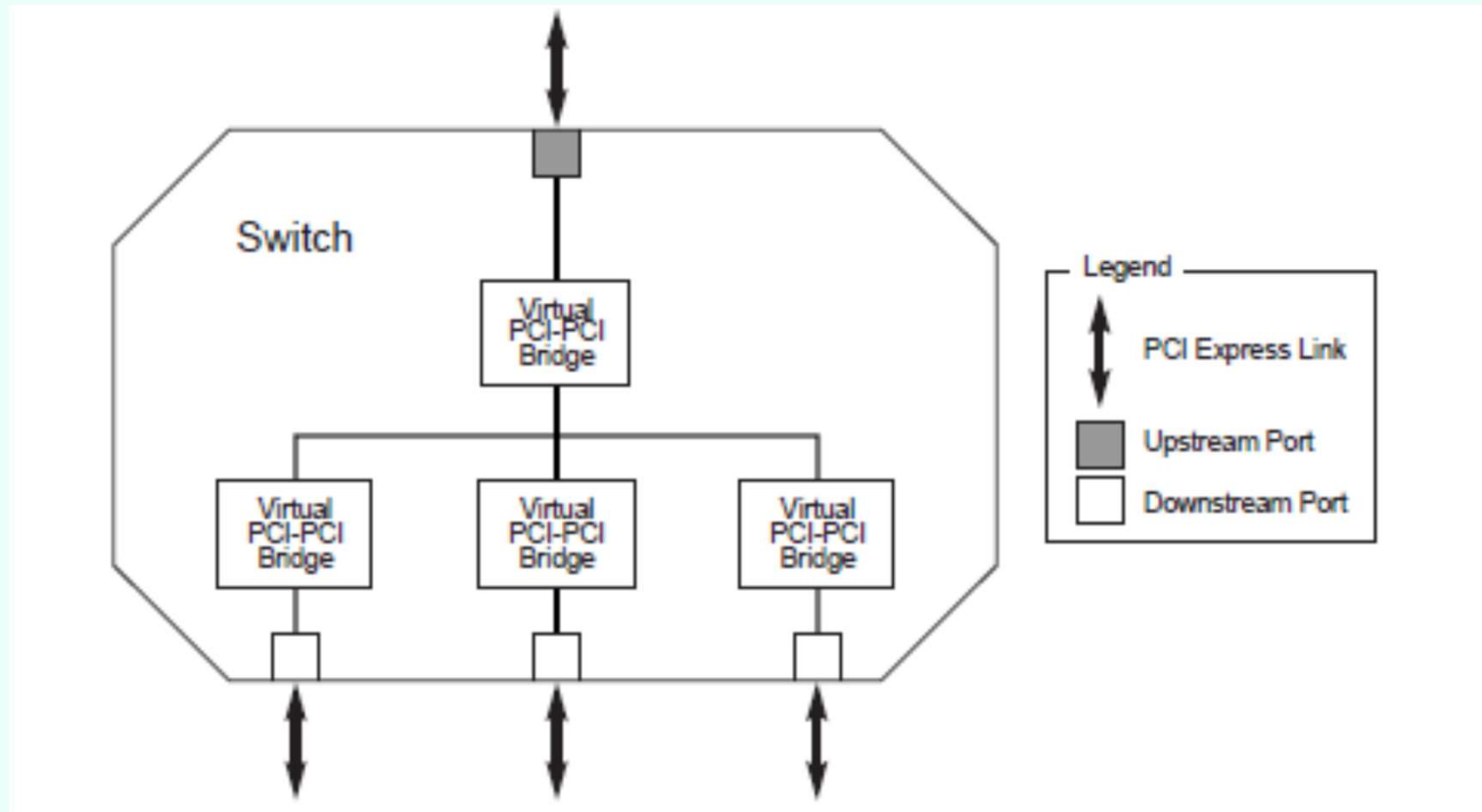
Endpoint

Endpoint refers to a type of Function that can be the Requester or Completer of a PCI Express transaction

e.g., a PCI Express attached graphics controller or a PCI Express-USB host controller.

Endpoints are classified as either legacy, PCI Express, or Root Complex Integrated Endpoints.

Switch



A Switch is defined as a logical assembly of multiple virtual PCI-to-PCI Bridge devices connects downstream ports to upstream port

Some more Terminology

Port: Interface between PCI express component and link

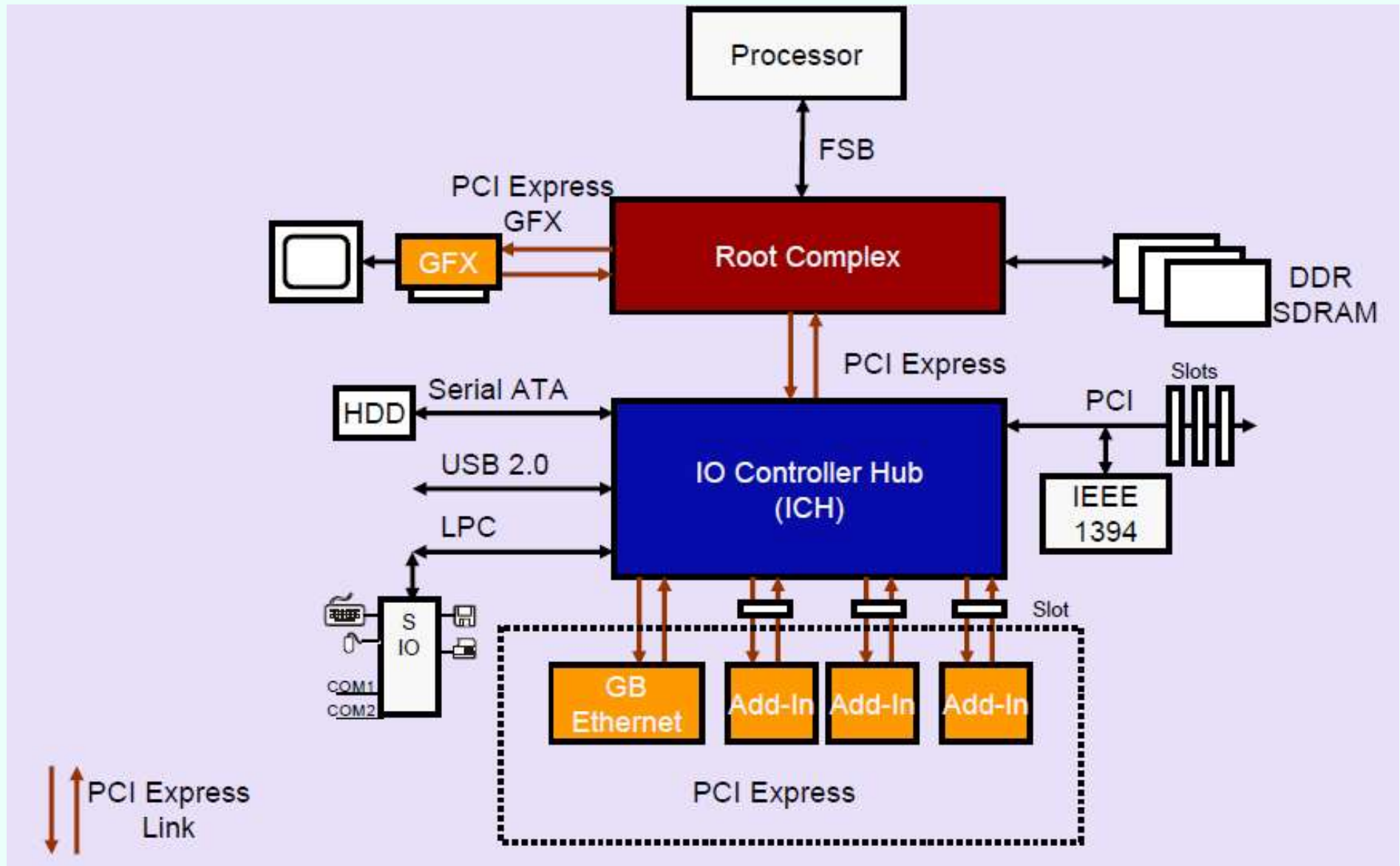
Upstream Port: points in the direction of the root complex.

Downstream Port: points away from the root complex

Ingress Port: that receives a packet.

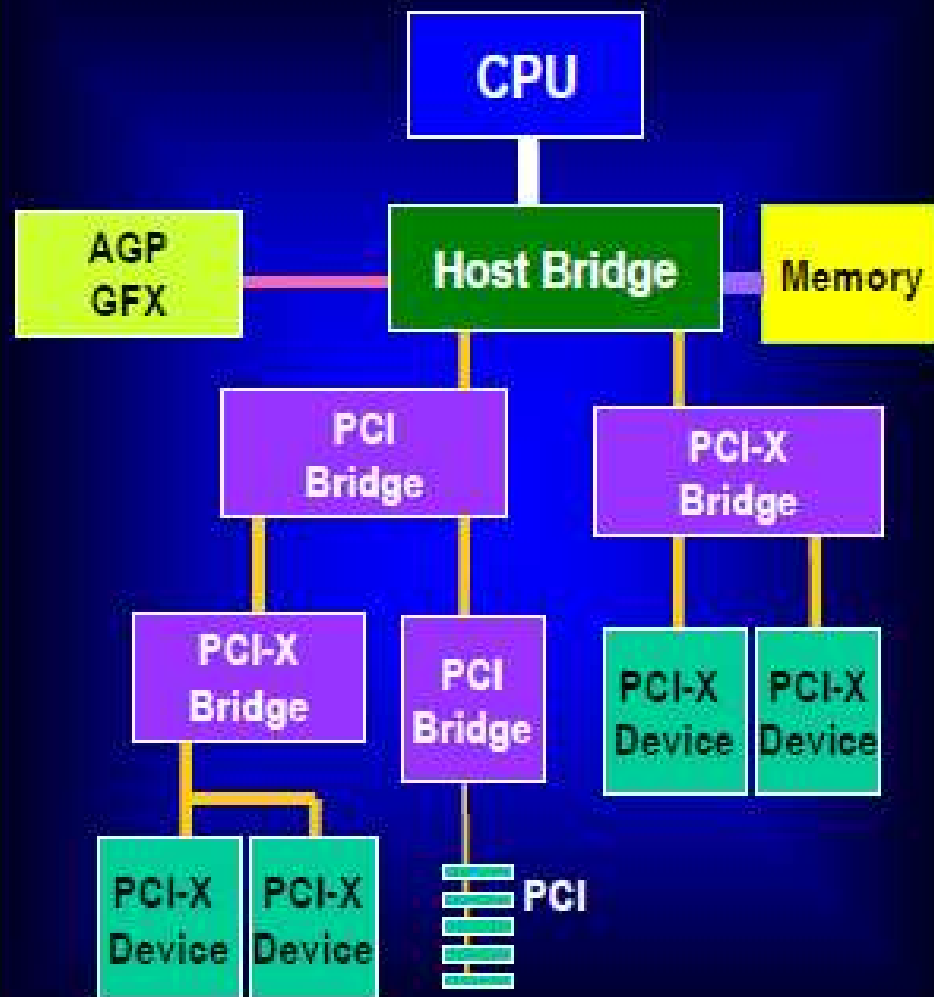
Egress Port: that transmits a packet

PCI Express System

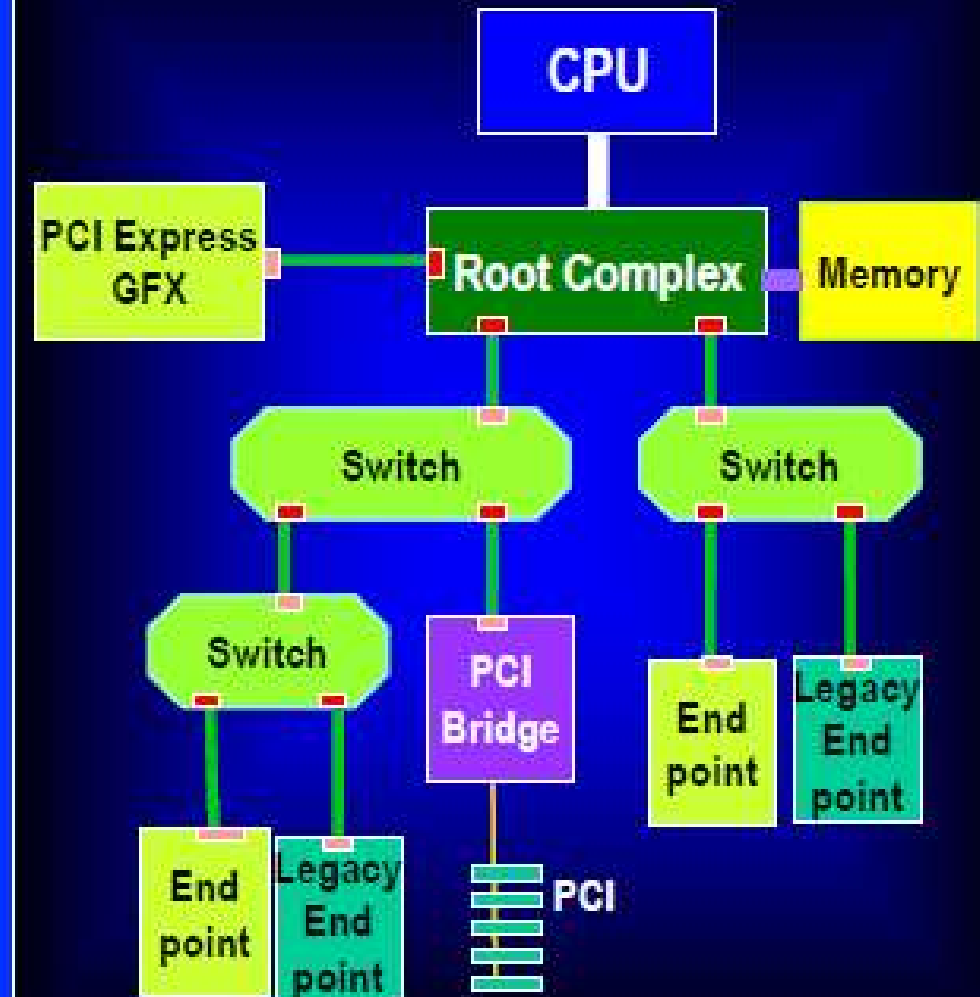


Difference in PCI and PCI Express systems

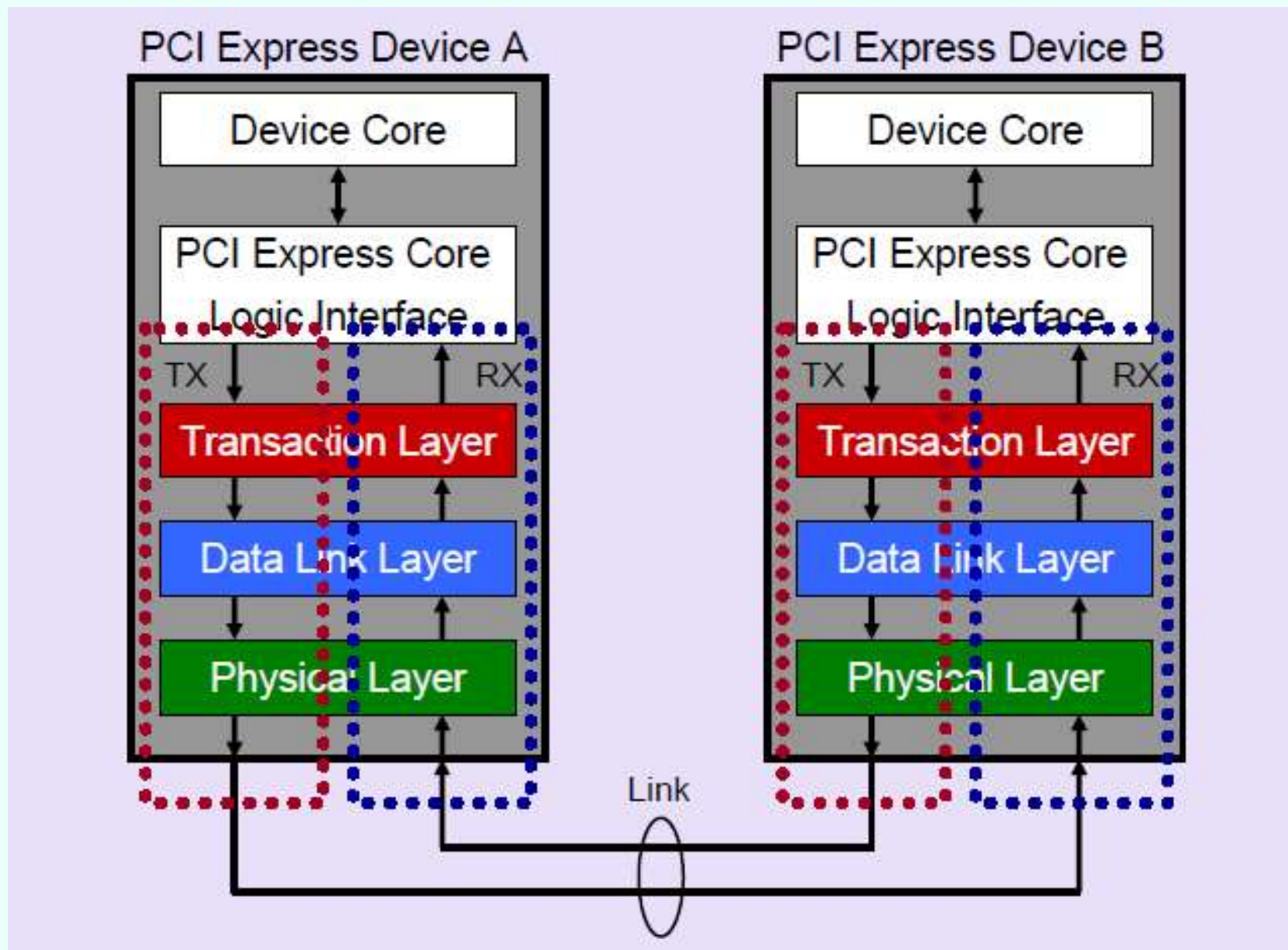
PCI System



PCI Express System



Layered Architecture



Layered Architecture

Transaction Layer functionality:

- The assembly and disassembly of Transaction Layer Packets (TLP) which are used to communicate transactions, such as read and write
- Managing credit-based flow control for TLPs

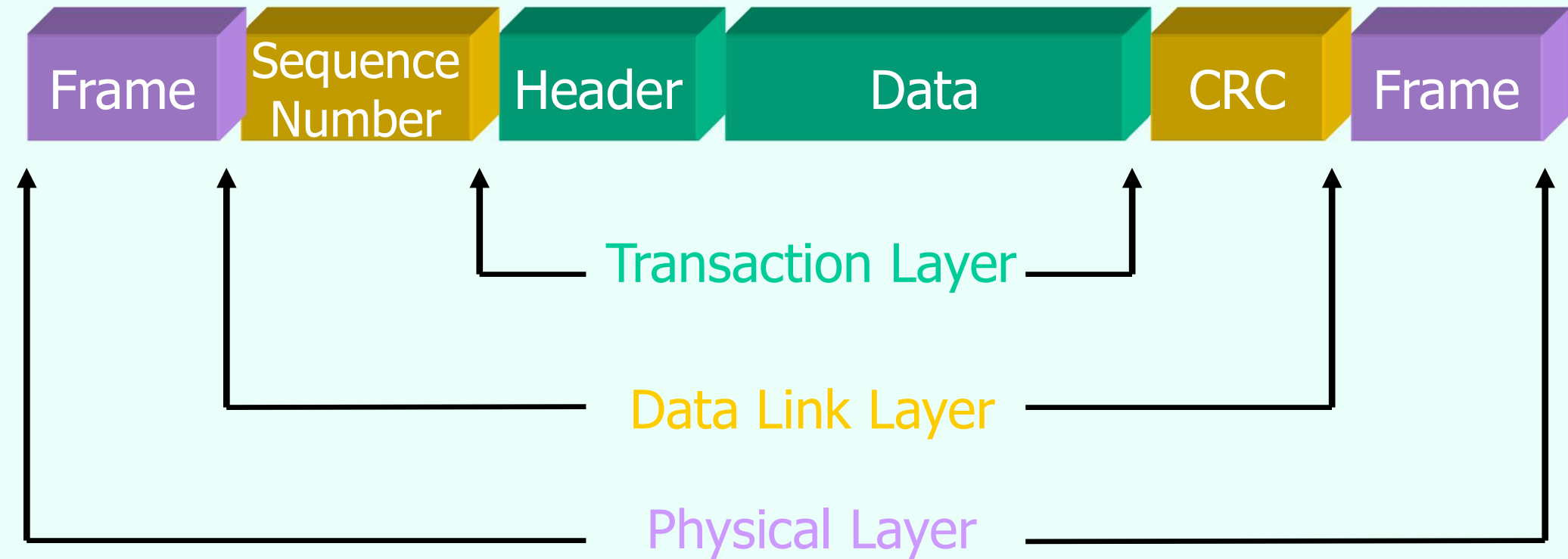
Data link Layer functionality :

- Link management
- data integrity, including error detection and error correction
- Packets that are generated and consumed at the Data Link Layer are (DLLPs)

Physical Layer functionality :

- Converting information received from the Data Link Layer into an appropriate serialized format
- Transmitting it across the PCI Express Link at a frequency and width compatible with the device connected to the other side of the Link
- Packets that are generated and consumed at the physical Layer are (PLPs)

Packet Formation



Transaction Layer

- The upper Layer of the architecture
- Primary responsibility : The assembly and disassembly of Transaction Layer Packets (TLPs).

TLPs are used to communicate transactions, such as read and write, as well as certain types of events.

- Responsible for managing credit-based flow control for TLPs.
- Every request packet requiring a response packet is implemented as a split transaction.

Each packet has a unique identifier that enables response packets to be directed to the correct originator.

Transaction Layer services

Packet generation and processing services

- Generate TLPs from device core Requests
- Convert received Request TLPs into Requests for the device core
- Convert received Completion Packets into a payload, or status information, deliverable to the Core
- Detect unsupported TLPs and invoke appropriate mechanisms for handling them
- If end-to-end data integrity is supported, generate the end-to-end data integrity CRC and update the TLP header accordingly.

Transaction Layer services

Flow control services

The Transaction Layer tracks flow control credits for TLPs across the Link.

Transaction credit status is periodically transmitted to the remote Transaction Layer using transport services of the Data Link Layer.

Remote Flow Control information is used to throttle TLP transmission.

Transaction Layer services

Power management services

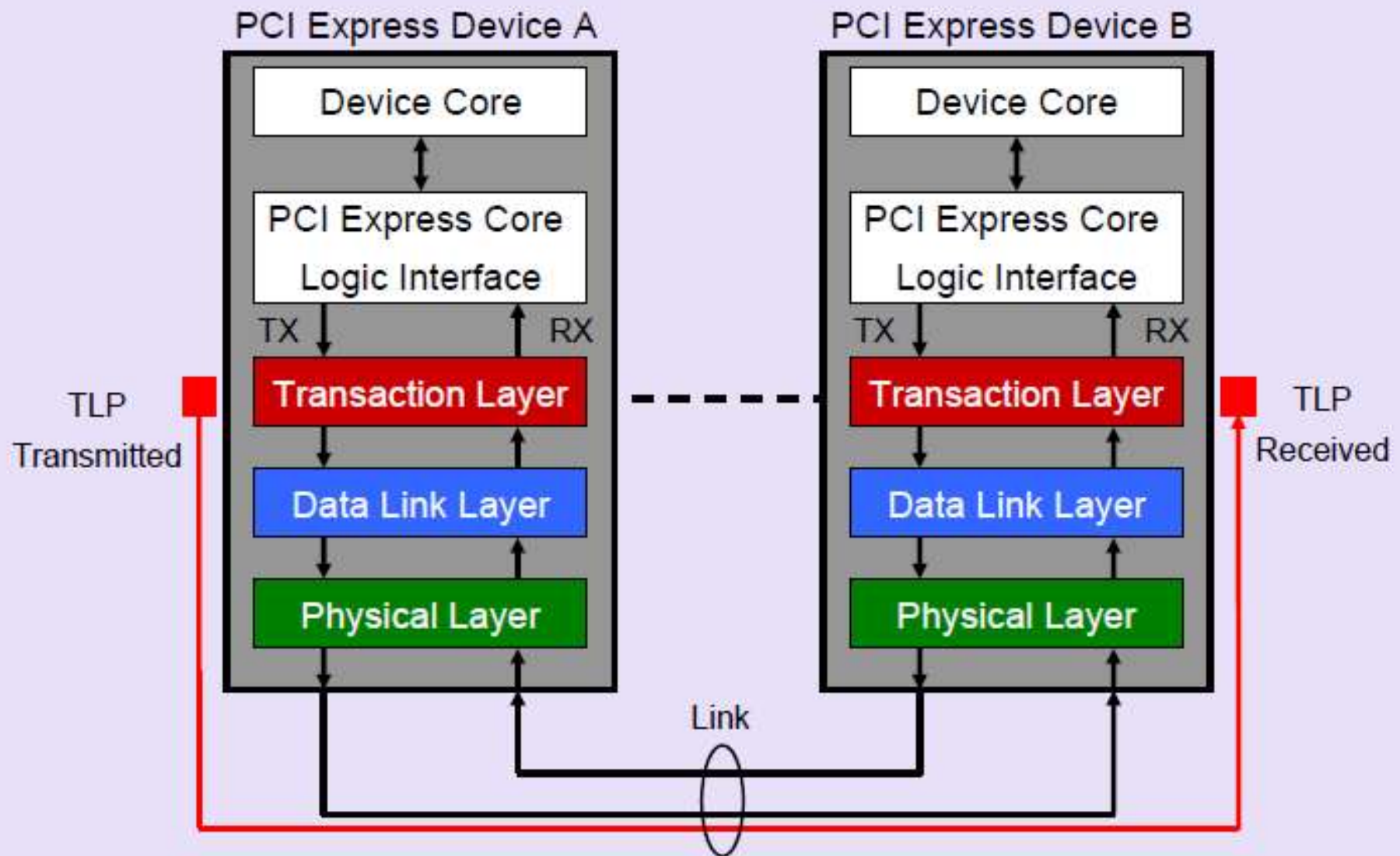
- PCI power management, as dictated by system software.
- Hardware-controlled autonomous power management minimizes power during full-on power states.

Also takes care of ordering rules

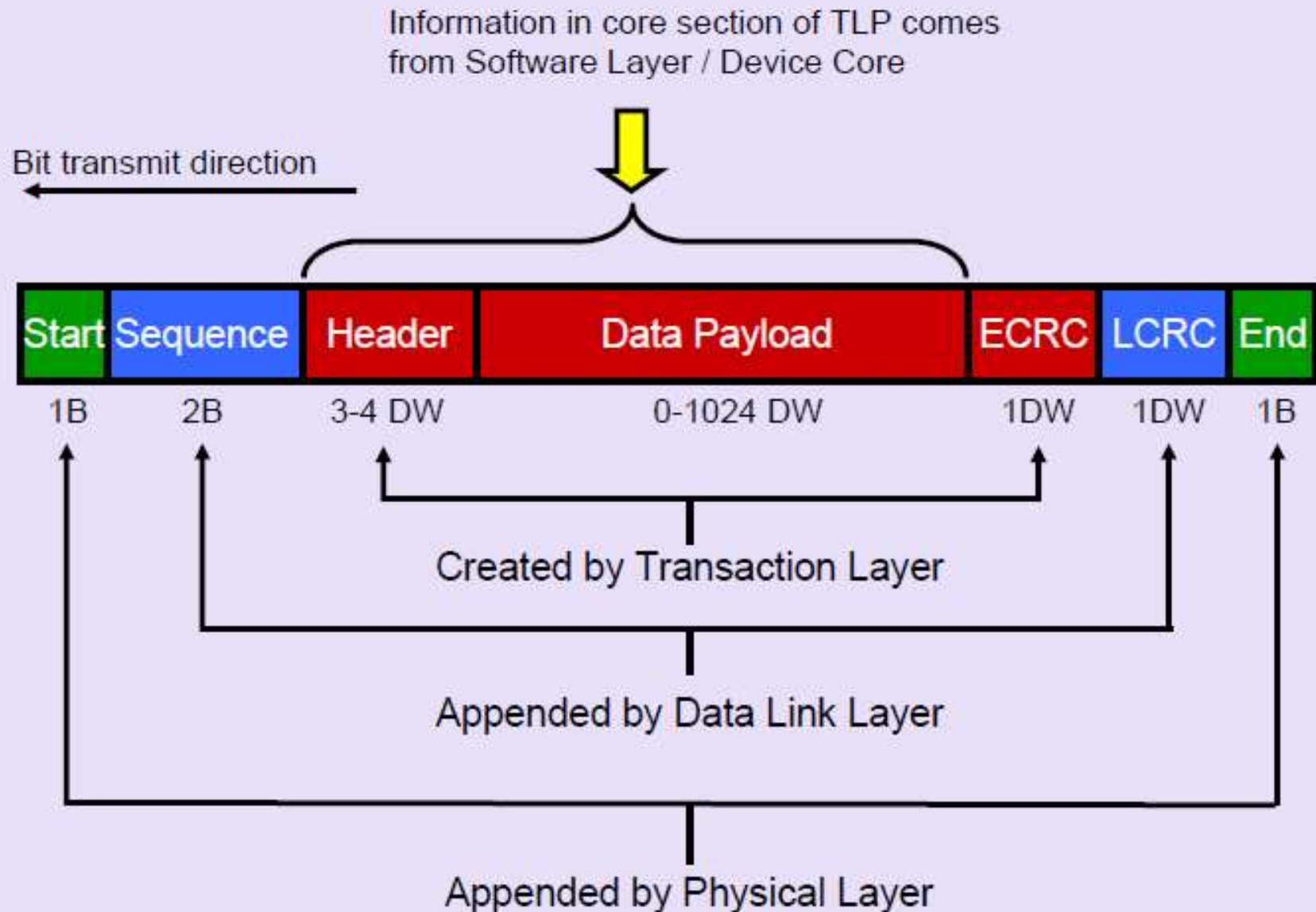
- PCI/PCI-X compliant producer consumer ordering model
- Extensions to support Relaxed Ordering
- Extensions to support ID-Based Ordering

Supports QOS using Virtual Channels and Traffic Class

TLP origin and destination



TLP Structure



PCI Express transactions

- Memory, IO and configuration address space same as PCI
Similar transaction types as PCI
Message transaction is additional
- PCI Express Transactions include:
memory read/write, memory read lock, IO read/write, configuration read/write, message requests
- Split transaction model for non-posted
- Message requests - Eliminates the wide array of sideband signals currently used in a platform implementation
- Locked transaction sequences are generated by the Host CPU(s) as one or more reads followed by a number of writes to the same location(s).

When a lock is established, all other traffic is blocked from using the path between the Root Complex and the locked Legacy Endpoint or Bridge.

Transaction Model

Transactions are divided in two categories

- **Posted** (*Memory Writes, message, message with data*)

Memory write packets are transmitted unidirectional from requester to completer.

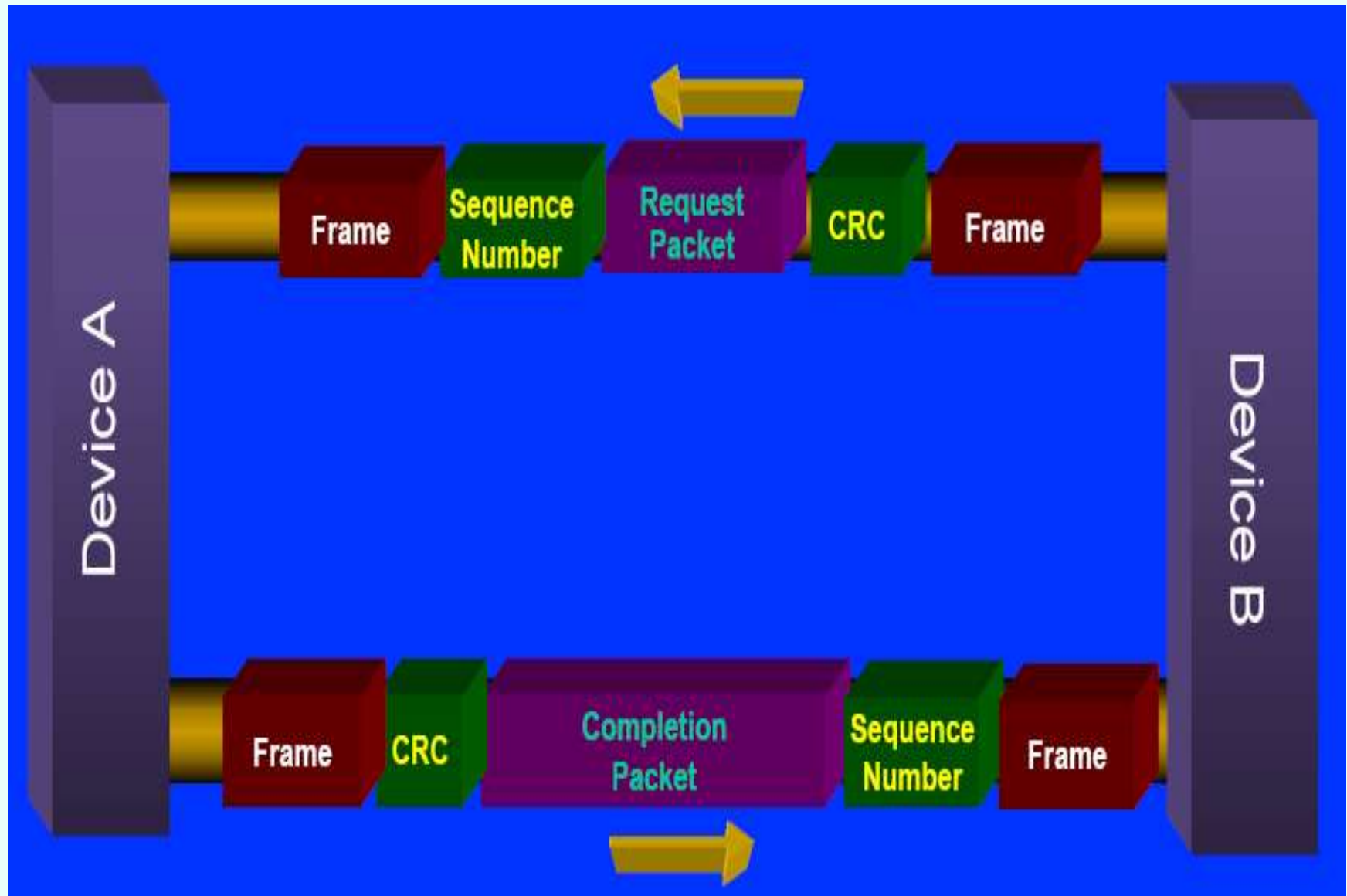
No completion packet from completer to receiver

- **Non posted** (*Memory Reads, Memory Read lock , IO write , IO read , configuration write , configuration read*)

Memory write packets are transmitted unidirectional from requester to completer.

Completion packets with data are transmitted from completer to receiver (cpl, cpld or cpldlk)

PCI Express transactions



Transaction types

Requests are translated to one of four transaction types by the Transaction Layer:

1. **Memory Read or Memory Write:** Used to transfer data from or to a memory mapped location
The protocol also supports a *locked memory read* transaction variant.
2. **I/O Read or I/O Write:** Used to transfer data from or to an I/O location
These transactions are restricted to supporting legacy endpoint devices.
3. **Configuration Read or Configuration Write:** Used to discover device capabilities, program features, and check status in the 4KB PCI Express configuration space.
4. **Messages:** Handled like posted writes. Used for event signaling and general purpose messaging.

TLP Types

Description	Abbreviated Name
Memory Read Request	MRd
Memory Read Request – Locked Access	MRdLk
Memory Write Request	MWr
IO Read Request	IORd
IO Write Request	IOWr
Configuration Read Request Type 0 and Type 1	CfgRd0, CfgRd1
Configuration Write Request Type 0 and Type 1	CfgWr0, CfgWr1
Message Request without Data Payload	Msg
Message Request with Data Payload	MsgD
Completion without Data (used for IO, configuration write completions and read completion with error completion status)	Cpl
Completion with Data (used for memory, IO and configuration read completions)	CplD
Completion for Locked Memory Read without Data (used for error status)	CplLk
Completion for Locked Memory Read with Data	CplDLk

Memory Transactions

Memory Transactions include the following types:

- Read Request/Completion MRd /CplD/Cpl
- Locked access Read Request
- Read Request/Completion MRdLk /CplDLk
- Write Request MWr/Cpl

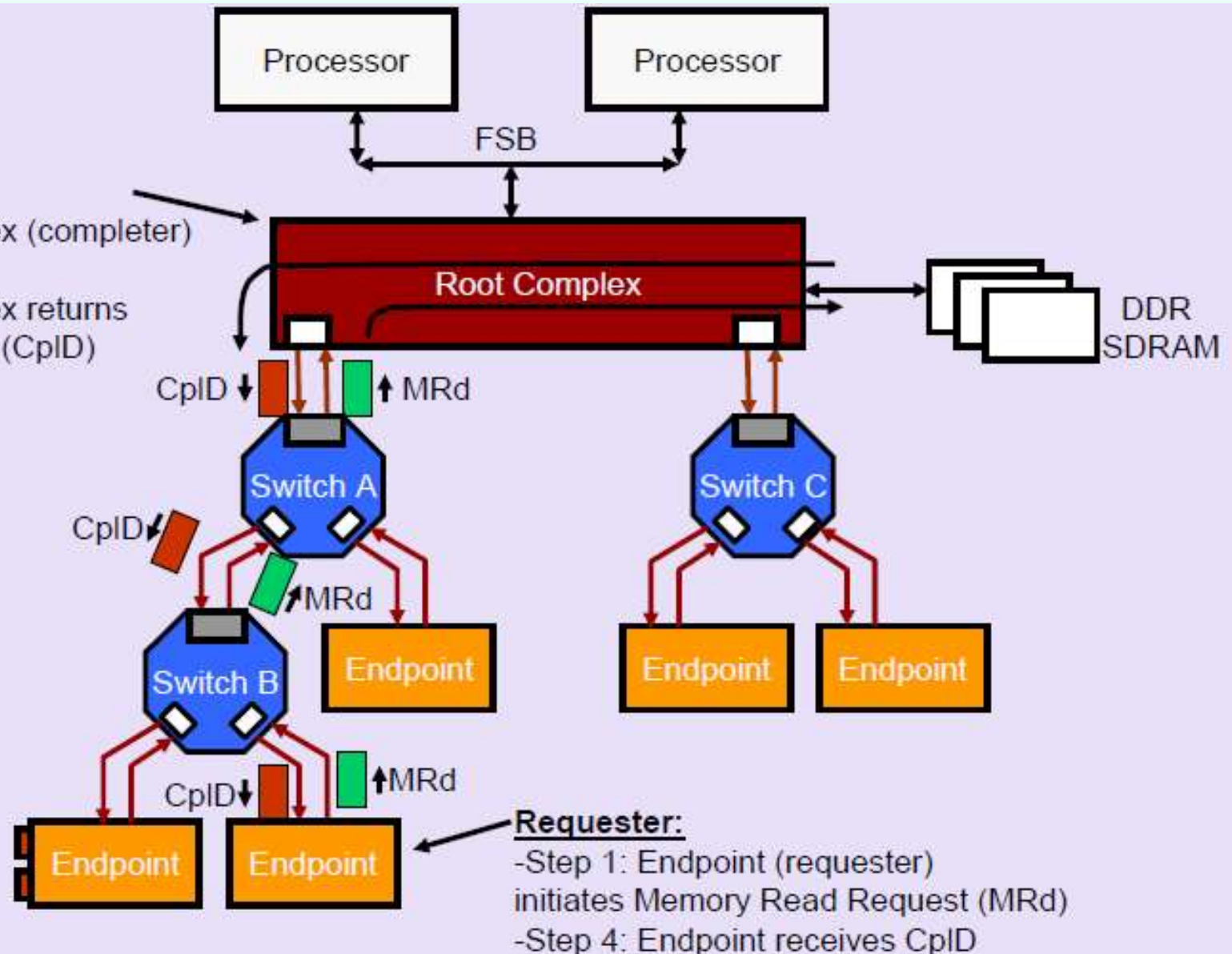
Memory Transactions use two different address formats:

- Short Address Format: 32-bit address
- Long Address Format: 64-bit address

Memory Transaction

Completer:

- Step 2: Root Complex (completer) receives MRd
- Step 3: Root Complex returns Completion with data (CpID)



Configuration Transactions

Configuration Transactions are used to access configuration registers of Functions within devices.

Configuration Transactions include the following types:

Read Request/Completion

CfgRd0 CfgRd1/Cpld

Write Request/Completion

CfgWr0 CfgWr1/Cpl

Message Transactions

- The Message Transactions, or simply Messages, are used to support in-band communication of events between devices
- All prior sideband signals, such as interrupts, power-management requests are sent as in-band Message transactions. Message transactions as “virtual wires”
- In addition to the specified Messages, PCI Express provides support for vendor-defined Messages using specified Message codes.
- This specification establishes a standard framework within which vendors can specify their own vendor-defined Messages tailored to fit the specific requirements of their platforms
- These vendor-defined Messages are not guaranteed to be interoperable with components from different vendors.

TLP Header Format

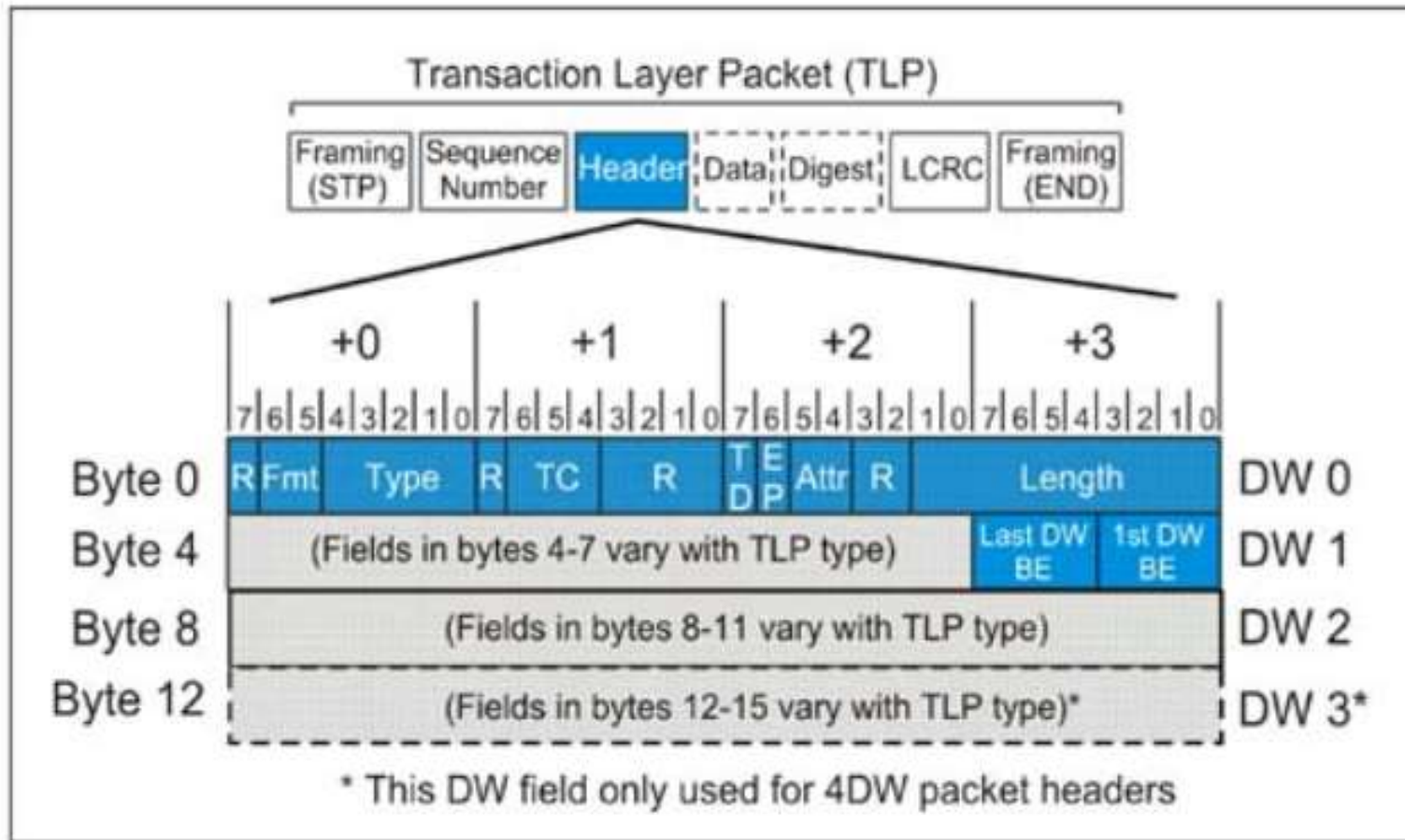
Header fields define Packet format

Depending on the type of a packet, routing the header for that packet will include some of the following types of fields:

- Format of the packet
- Type of the packet
- Length for any associated data
- Transaction Descriptor, including:
 - i. Transaction ID
 - ii. Attributes
 - iii. Traffic Class
- Address/routing information
- Byte Enables
- Message encoding
- Completion status

TLP Header Format

Generic TLP Header fields



TLP Header Format

Header Field	Header Location	Field Use
Length [9:0]	Byte 3 Bit 7:0 Byte 2 Bit 1:0	TLP data payload transfer size, in DW. Maximum transfer size is 10 bits, $2^{10} = 1024$ DW (4KB). Encoding: 00 0000 0001b = 1DW 00 0000 0010b = 2DW . . 11 1111 1111b = 1023 DW 00 0000 0000b = 1024 DW
Attr (Attributes)	Byte 2 Bit 5:4	Bit 5 = Relaxed ordering. When set = 1, PCI-X relaxed ordering is enabled for this TLP. If set = 0, then strict PCI ordering is used. Bit 4 = No Snoop. When set = 1, requester is indicating that no host cache coherency issues exist with respect to this TLP. System hardware is not required to cause processor cache snoop for coherency. When set = 0, PCI -type cache snoop protection is required.

TLP Header Format

Header Field	Header Location	Field Use
EP (Poisoned Data)	Byte 2 Bit 6	If set = 1, the data accompanying this data should be considered invalid although the transaction is being allowed to complete normally.
TD (TLP Digest Field Present)	Byte 2 Bit 7	<p>If set = 1, the optional 1 DW TLP Digest field is included with this TLP that contains an ECRC value. Some rules: Presence of the Digest field must be checked by all receivers (using this bit).</p> <ul style="list-style-type: none">•A TLP with TD = 1, but no Digest field is handled as a Malformed TLP.•If a device supports checking ECRC and TD=1, it must perform the ECRC check.•If a device does not support checking ECRC (optional) at the ultimate destination, the device must ignore the digest.

TLP Header Format

Header Field	Header Location	Field Use
TC (Traffic Class)	Byte 1 Bit 6:4	<p>These three bits are used to encode the traffic class to be applied to this TLP and to the completion associated with it (if any).</p> <p>000b = Traffic Class 0 (Default)</p> <p>·</p> <p>·</p> <p>111b = Traffic Class 7</p> <p>TC 0 is the default class, and TC 1-7 are used in providing differentiated services.</p>
Type[4:0]	Byte 0 Bit 4:0	<p>These 5 bits encode the transaction variant used with this TLP. The Type field is used with Fmt [1:0] field to specify transaction type, header size, and whether data payload is present. See below for additional information of Type/Fmt encoding for each transaction type.</p>

TLP Header Format

Header Field	Header Location	Field Use
Fmt[1:0] Format	Byte 0 Bit 6:5	<p>These two bits encode information about header size and whether a data payload will be part of the TLP:</p> <p>00b 3DW header, no data 01b 4DW header, no data 10b 3DW header, with data 11b 4DW header, with data</p> <p>See below for additional information of Type/Fmt encoding for each transaction type.</p>
First DW Byte Enables	Byte 7 Bit 3:0	<p>These four high-true bits map one-to-one to the bytes within the first double word of payload.</p> <p>Bit 3 = 1: Byte 3 in first DW is valid; otherwise not Bit 2 = 1: Byte 2 in first DW is valid; otherwise not Bit 1 = 1: Byte 1 in first DW is valid; otherwise not Bit 0 = 1: Byte 0 in first DW is valid; otherwise not</p> <p>See below for details on Byte Enable use.</p>

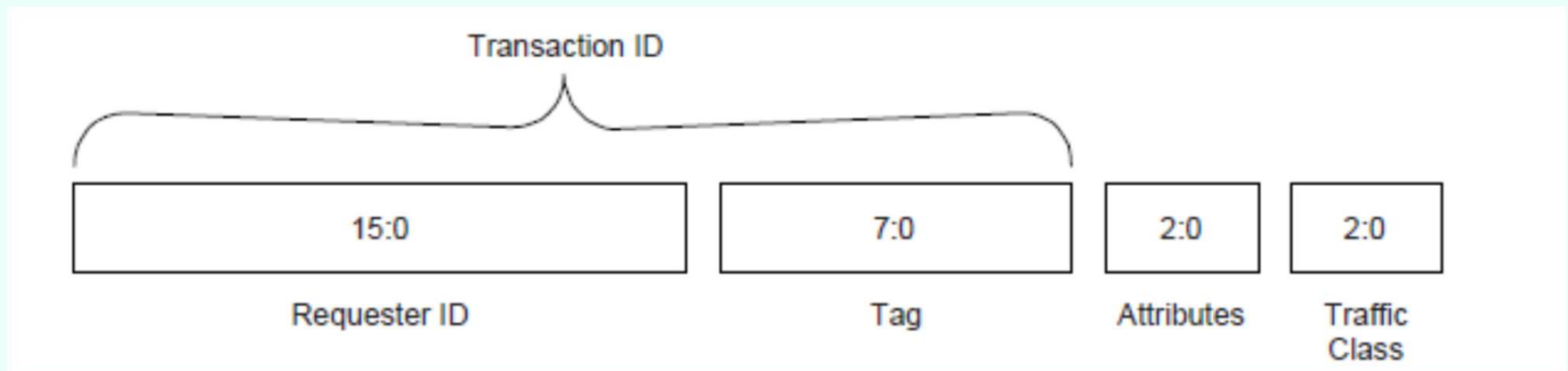
TLP Header Format

Header Field	Header Location	Field Use
Last DW Byte Enables	Byte 7 Bit 7:4	<p>These four high-true bits map one-to-one to the bytes within the first double word of payload.</p> <p>Bit 3 = 1: Byte 3 in last DW is valid; otherwise not Bit 2 = 1: Byte 2 in last DW is valid; otherwise not Bit 1 = 1: Byte 1 in last DW is valid; otherwise not Bit 0 = 1: Byte 0 in last DW is valid; otherwise not</p> <p>See below for details on Byte Enable use.</p>

TLP Descriptor fields

As transactions move between requester and completer, it is important to uniquely identify a transaction, since many split transactions may be pending at any instant.

While the Transaction Descriptor fields are not in adjacent header locations, collectively they describe key transaction attributes

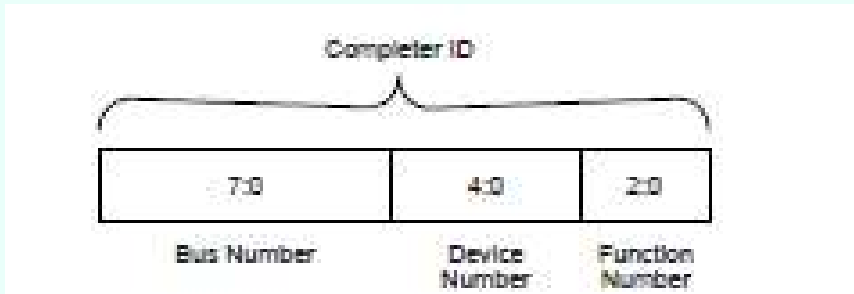


TLP Descriptor fields

Transaction Descriptor fields comprises of

- **Transaction ID**
This is comprised of the Bus, Device, and Function Number of the TLP requester (Requester ID) AND the Tag field of the TLP
- **Attributes**
These consist of the Relaxed Ordering and No Snoop bits. These are also set by the requester and travel with the packet to the completer.
- **Traffic Class**
Traffic Class (TC 0 -7) is inserted in the TLP by the requester, and travels unmodified through the topology to the completer. At every link, Traffic Class is mapped to one of the available virtual channels.

TLP Header Fields



Byte 8, 7:0 = Completer Bus Number

Byte 9, 7:3 = Completer Device Number

Byte 9, 2:0 = Completer Function Number

Completion headers must supply the same values for the Requester ID, Tag, and Traffic Class as were supplied in the header of the corresponding Request.

Completion headers must supply the same values for the Attribute as were supplied in the header of the corresponding Request.

A Completion including data must specify the actual amount of data returned in that Completion, and must include the amount of data specified.

Three Methods of TLP Routing

TLPs are routed using one of the **three schemes**:

➤ Address Routing

Address routing is used with Memory and I/O Requests.

Two address formats are specified, a 64-bit format used with a 4 DW header and a 32-bit format used with a 3 DW header

Memory and IO requests use **address routing**.

➤ ID Routing

ID routing uses the Bus, Device, and Function Numbers (as applicable) to specify the destination for the TLP

Two ID routing formats are specified, one used with a 4 DW header and one used with a 3 DW header

Completions and Configuration cycles use **ID routing**.

➤ Implicit Routing

Message requests have **selectable routing** based on a 3-bit code in the message routing sub-field of the header type field.

Transaction Layer services

QOS support using Virtual Channels and Traffic Class:

Quality of service means a capability of routing packets from different applications through the fabric with differentiated priorities, deterministic latencies and bandwidth

These are routed through virtual channel (VCs) buffers implemented in switches, endpoints and root complex devices.

8 Virtual channels (VCs) and 8 traffic classes (TCs) are available to support Quality of Service (QoS)

Quality of service

The combination of Virtual Channel mechanism and Traffic Class identification is provided to support differentiated services and QoS support for certain classes of applications.

Video data packets move through fabric with higher priority than some other control packets.

- **Virtual Channels:** Virtual Channels provide a means to support multiple independent logical data flows over given common physical resources of the Link. Conceptually this involves multiplexing different data flows onto a single physical Link.
- **Traffic Class:** The Traffic Class is a Transaction Layer Packet label that is transmitted unmodified end-to-end through the fabric, Traffic Class labels are used to apply appropriate servicing policies. Each Traffic Class label defines a unique ordering domain - no ordering guarantees are provided for packets that contain different Traffic Class labels.

Packets with different traffic classes (TCs) move with different priority

Data Link Layer

Responsible for reliably conveying Transaction Layer Packets (TLPs) supplied by the Transaction Layer across a PCI Express Link to the other component's Transaction Layer.

Services provided by the Data Link Layer include:

Reliable data transport services :

Accept TLPs for transmission from the Transmit Transaction Layer and convey them to the Transmit Physical Layer

Accept TLPs received over the Link from the Physical Layer and convey them to the Receive Transaction Layer

Data Link Layer

Error detection and retry:

TLP Sequence Number and LCRC generation

Transmitted TLP storage for Data Link Layer Retry Data integrity checking for TLPs and Data Link Layer Packets (DLLPs)

Positive and negative acknowledgement DLLPs

Error indications for error reporting and logging mechanisms

Link Acknowledgement Timeout replay mechanism

• Initialization and Power management

Track Link state and convey active/reset/disconnected state to Transaction Layer and Physical Layers

Performs Link management through the Physical Layer

The Data Link Layer contains the Data Link Control and Management State Machine (DLCMSM) to perform these tasks.

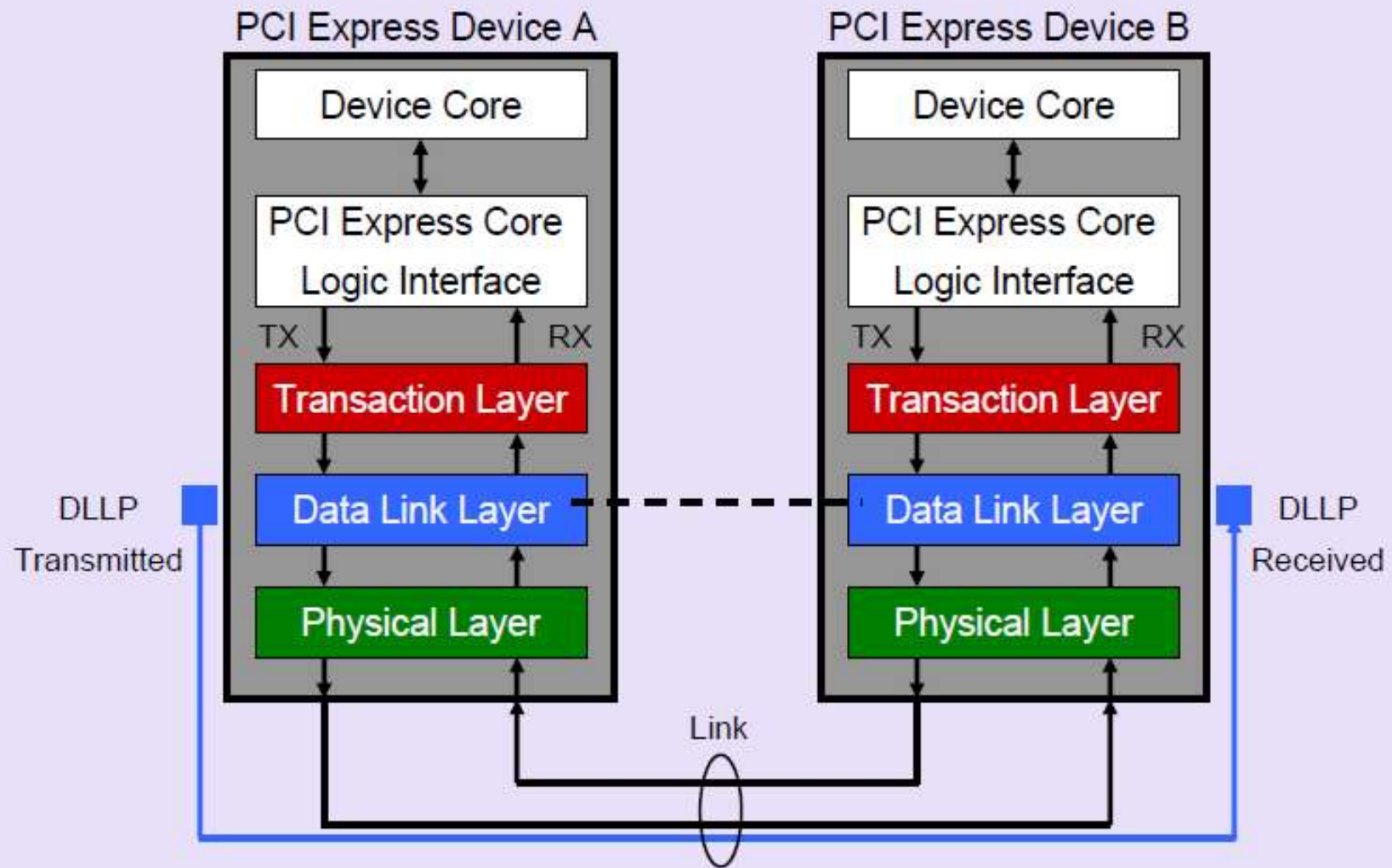
Data Link Layer Packets

Data Link Layer Packets (DLLPs) are:

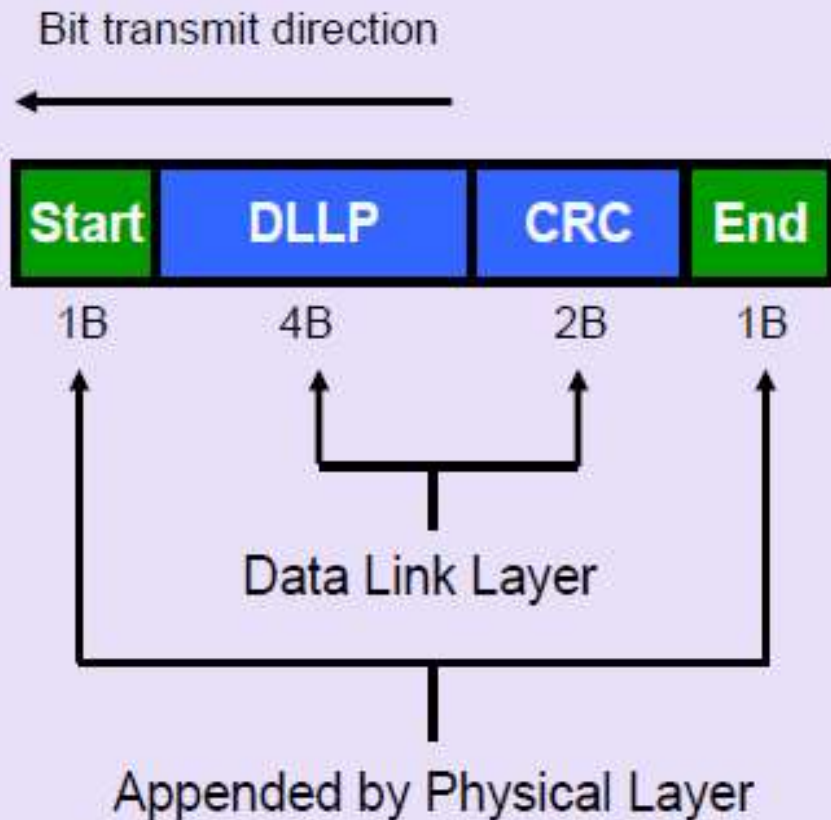
- Used for Link Management functions including TLP acknowledgement, power management, and exchange of Flow Control information.
- Transferred between Data Link Layers of the two directly connected components on a Link

DLLPs are sent point-to-point, between the two components on one Link. TLPs are routed from one component to another, potentially through one or more intermediate components

DLLP Origin and Destination



DLLP Structure



- ACK / NAK Packets
- Flow Control Packets
- Power Management Packets
- Vendor Defined Packets

DLLP Types

DLLPs used to support Link operations:

Ack DLLP: TLP Sequence number acknowledgement; used to indicate successful receipt of some number of TLPs

Nak DLLP: TLP Sequence number negative acknowledgement; used to initiate a Data Link Layer Retry

DLLPs used to support Flow Control operations:

InitFC1, InitFC2, and UpdateFC

For posted, non posted and completion packet types

DLLPs used for Power Management:

PM_Enter_L1, PM_Enter_L23, PM_Active_State_Request_L1, PM_Request_Ack

Data Integrity

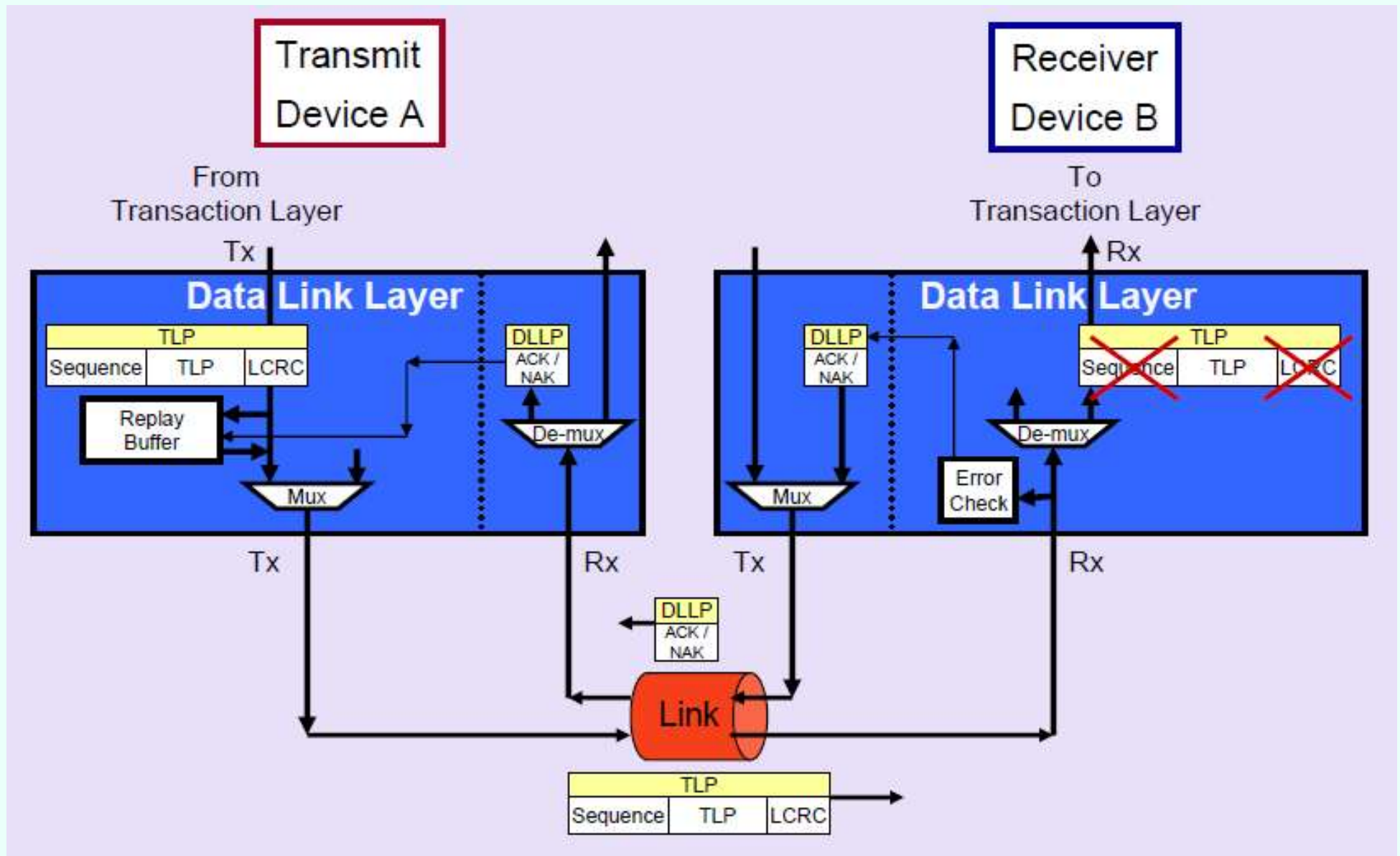
The Transaction Layer provides TLP boundary information to Data Link Layer.

The Data Link Layer apply a Sequence Number and Link CRC (LCRC) error detection to the TLP

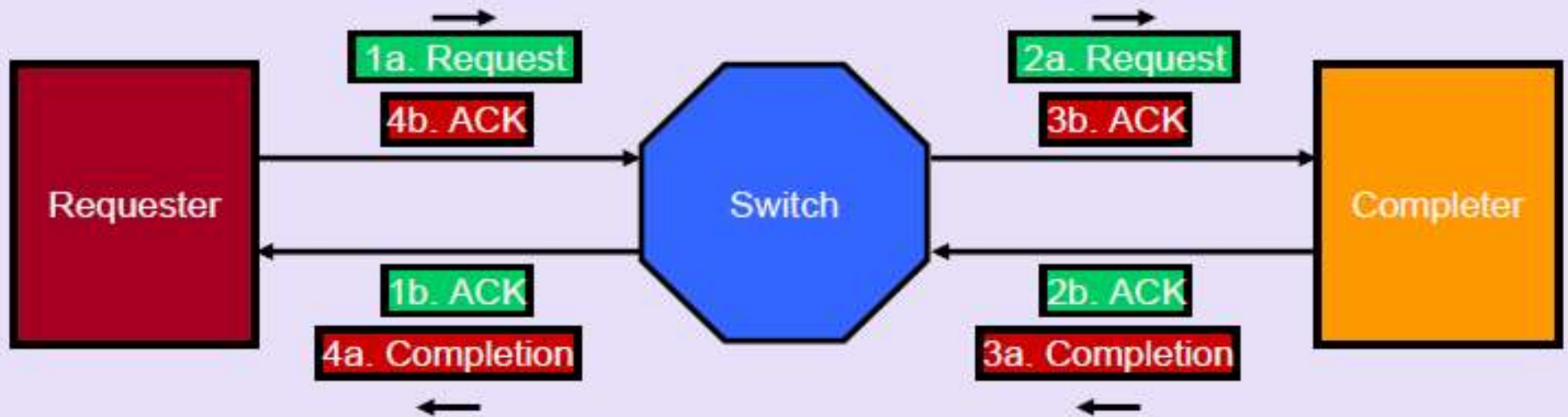
The Receive Data Link Layer validates received TLPs by checking the Sequence Number, LCRC code and any error indications from the Receive Physical Layer.

In case of error in a TLP, Data Link Layer Retry is used for recovery using ACK/NACK protocol

ACK/NAK Protocol



ACK/NAK Protocol : Point to Point



ACK returned for good reception of Request or Completion
NAK returned for error reception of Request or Completion

Error Protection

Data Integrity checking using LCRC:

Data Integrity checking for DLLPs and TLPs is done using a CRC included with each packet sent across the Link.

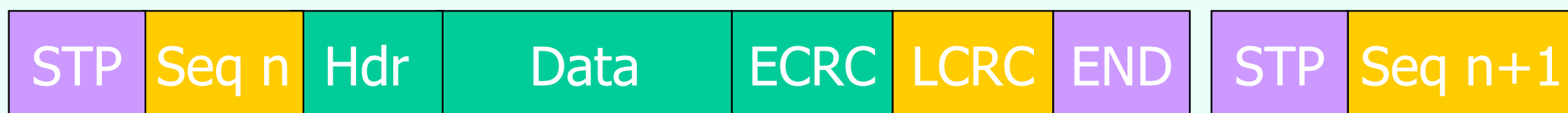
- DLLPs use a 16-bit CRC
- TLPs (which can be much longer than DLLPs) use a 32-bit LCRC.
- TLPs additionally include a sequence number, which is used to detect cases where one or more entire TLPs have been lost.

Error Protection

Unambiguous framing with 8B10B

Explicit error forwarding mechanism

End to end 32 bit CRC



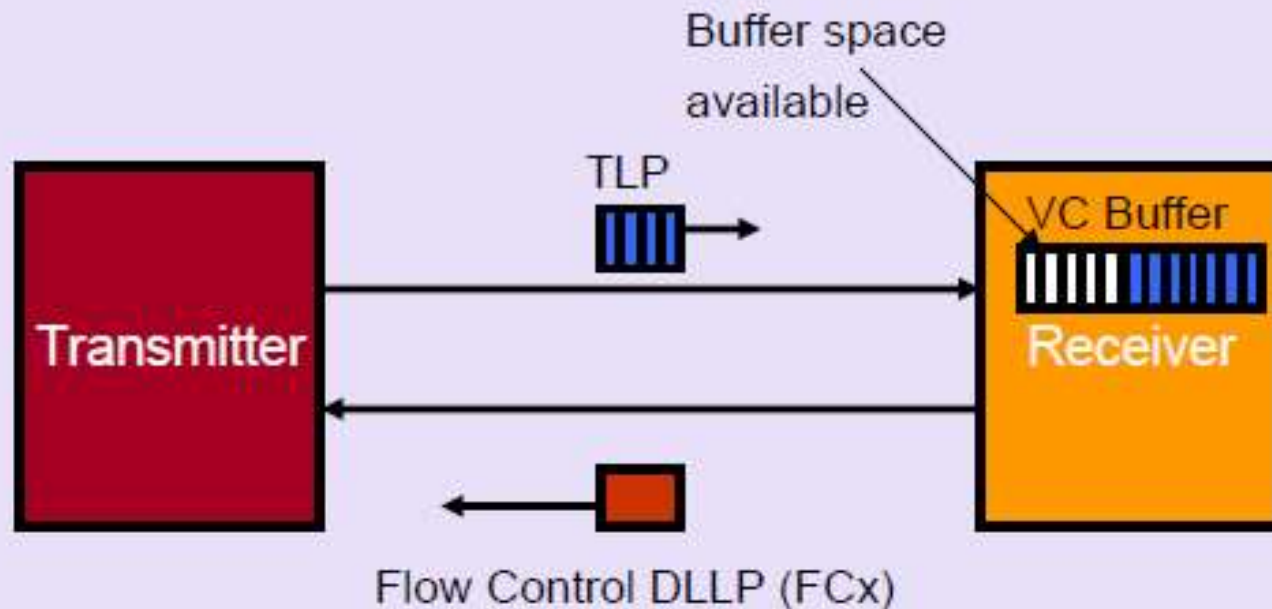
TLP protected with 32 bit CRC

False /missed start

False /missed termination

PCI Express Flow Control

Credit-based *flow control* is point-to-point based, not end-to-end



Receiver sends Flow Control Packets (FCP) which are a type of DLLP (Data Link Layer Packet) to provide the transmitter with credits so that it can transmit packets to the receiver

Physical Layer

- Isolates the Transaction and Data Link Layers from the signalling technology used for Link data interchange.
- Divided into the logical and electrical Sub-blocks.

Services provided by the Physical Layer include:

- Symbol encoding/decoding using 8b/10b transmission code
- Serialization and de-serialization
- Uses special symbols for framing and management (k symbols)
- Link Management and Training

Physical Layer

Link Management and Training

Configures and initializes each device's Physical Layer, port, and associated link for normal operation

Initiated automatically after reset without any software involvement

A Link re-training, is initiated automatically as a result of a wakeup event from a low power mode, or due to an error condition that renders the Link inoperable

The following things are discovered, negotiated and determined during the training process:

- Link width
- Link data rate
- Lane reversal
- Polarity inversion
- Lane-to-Lane de-skew within a multi-Lane Link.

Physical Layer

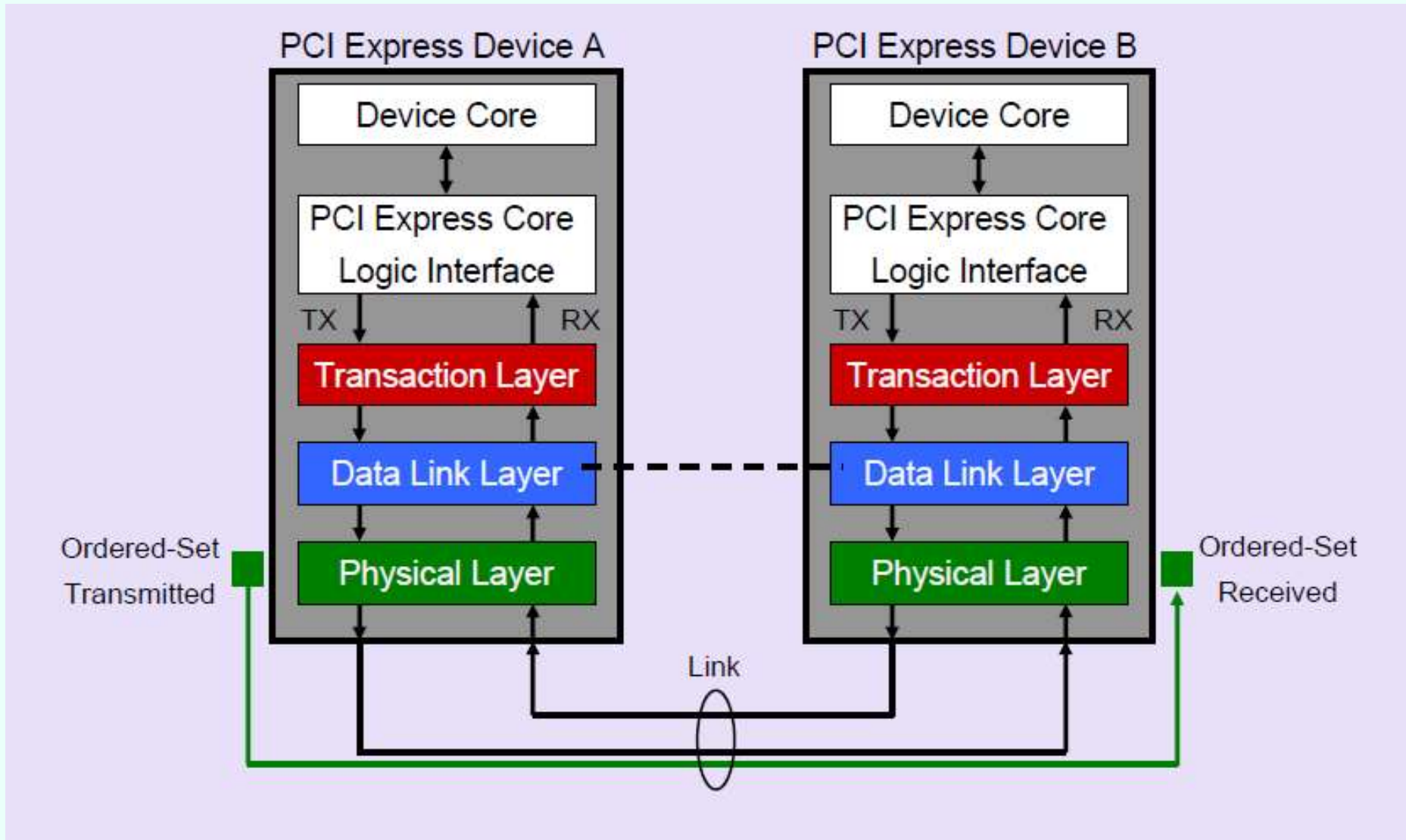
- Provides isolation from higher layers
- Logical functions
 - Encoding/decoding, Scrambling
 - Reset, initialization, de-skew
 - Configuration speed, link width, lane mapping
- Electrical functions
 - Transmit/ receive
 - Packet exchange
 - Link power management

Physical Layer

Training Sequences

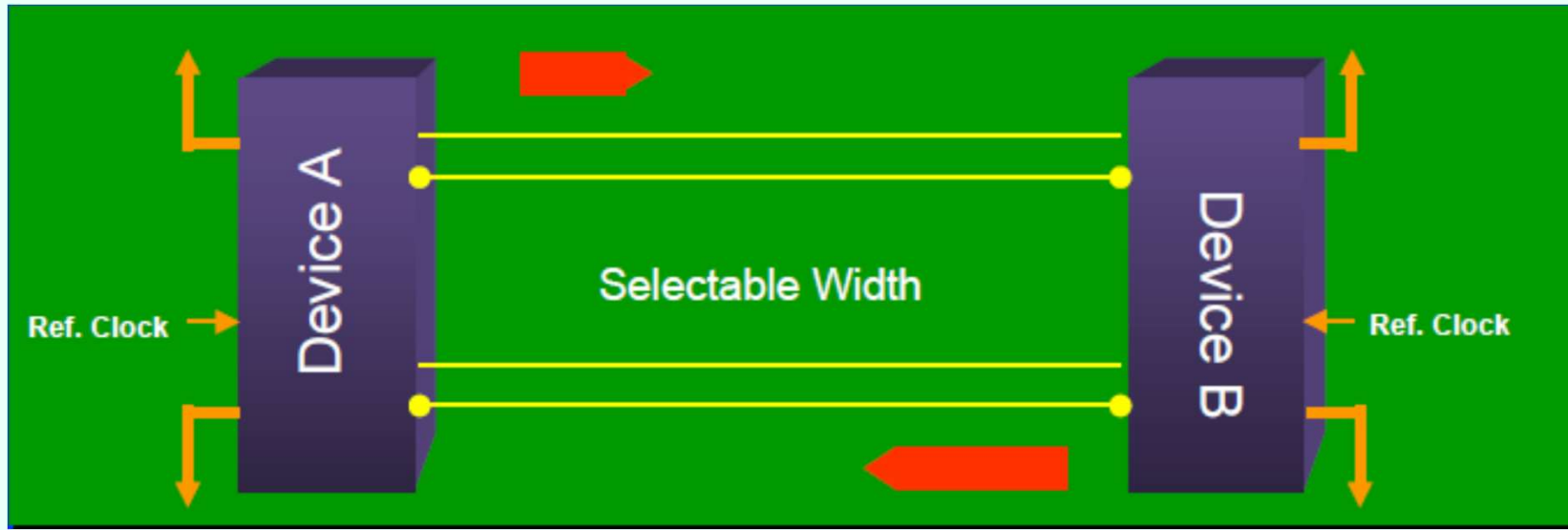
Training sequences are composed of Ordered Sets used for initializing bit alignment, Symbol alignment and to exchange Physical Layer parameters

Ordered Set Origin And Destination



Isolates the Transaction and Data Link Layers from the signalling technology used for Link data interchange.

PCI Express Link



Lane : A set of differential signal pairs, one pair for transmission and one pair for reception. A by-N Link is composed of N *Lanes*.

Link : The collection of two Ports and their interconnecting Lanes.

Link is a dual simplex communications path between two components.

PCI Express Link

- Low voltage differential Links LVDS
- Data clock embedded using 8b/10b encoding
- Aggregate multiple lanes to scale bandwidth
- Selectable lane width :x1, x2, x4, x8, x12, x16, x32

❖ Gen 1 Bit rate: 2.5Gb/sec/lane/direction

❖ Gen 2 Bit rate: 5Gb/sec/lane/direction

❖ Gen 3 Bit rate: 8Gb/sec/lane/direction

❖ Gen 4 Bit rate: 16Gb/sec/lane/direction

Link must support at least one lane

PCI Express Throughput

	Link Width						
	x1	x2	x4	x8	x12	x16	x32
PCIe 1.xBW(GB/s)	0.5	1	2	4	6	8	16
PCIe 2.xBW(GB/s)	1	2	4	8	12	16	32

Derivation of these numbers

2.5GT/s(PCIe 1.x) or 5.0GT/s(PCIe 2.0) signaling in each direction

20% overhead due to 8b/10b encoding

Bandwidth described as “aggregate”, implying simultaneous traffic in both directions

$(\text{Link speed} \times 2(\text{both directions}) \times 8/10(8\text{b}/10\text{b overhead})) / 8 (\text{byte})$

$(2.5 \times 2 \times 8/10) / 8 = 0.5 \text{ GB/s}$

Power Budget

	x1		x4/x8	x16	
Standard Height	10W	25W	25W	25W	40W
Low Profile	10W		10W	25W	

Allowed Maximum Power Consumption

Slots



PCI-X

PCI Express x8

PCI (32 bit, 5V)

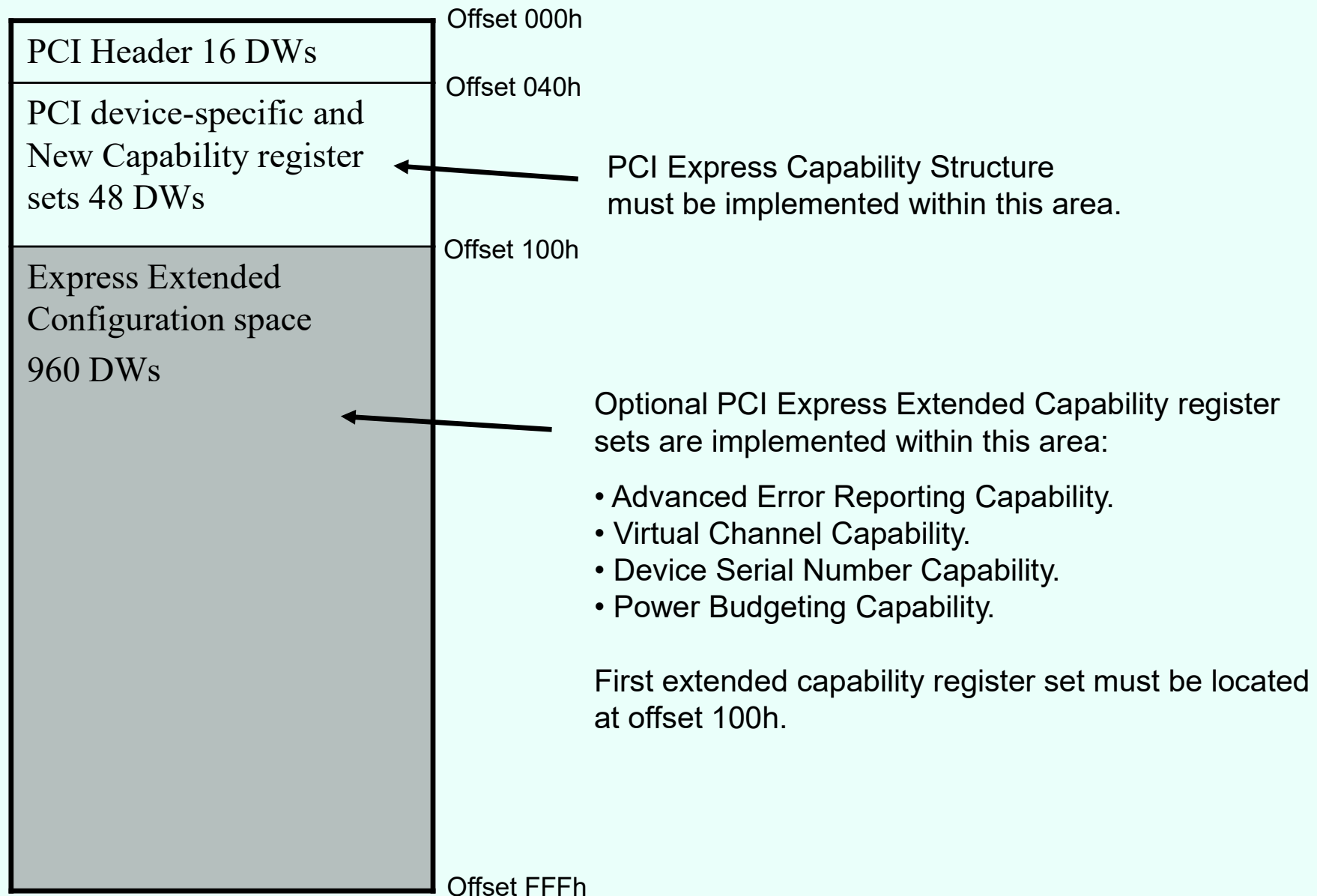
PCI Express x16

Software layers

PCI compatible software model

- PCI configuration and enumeration software can be used to enumerate PCI Express hardware
- PCI Express system will boot with existing OS
- PCI Express supports existing device drivers
- New additional configuration address space requires OS and driver update

Express Specific Configuration Registers



Express Specific Configuration Registers

Device capability Register

It identifies PCI Express device specific capabilities like MPS, MRR etc.

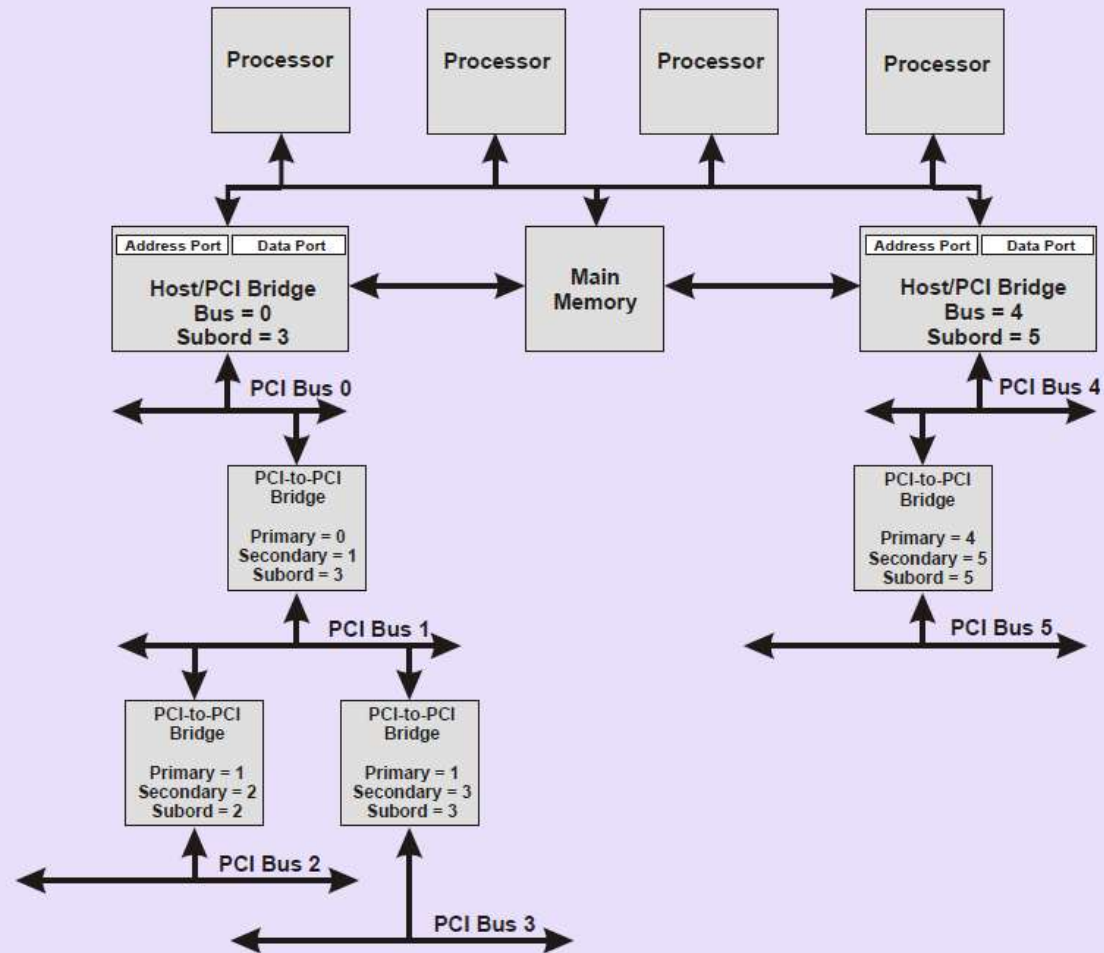
PCI compatible space occupies the first 64 DWs of the 4 KB space.

The remaining 960 DWs is referred to as the PCI Express Extended Configuration Space.

It is utilized to implement the **optional** Extended capabilities

- Advanced Error Reporting Capability
- Virtual Channel Capability
- Power Budgeting
- Device Serial Number Capability

Configuration Space (cont'd)



Case Study

Case Study

PCI Express x8 Network Interface Card for PARAMNet-3 System Area Network:

Specifications:

- Programmable NIC based on “**Gemini**” Co-processor
- PCI e based 2.5Gx8 (20G) Host interface
- CX4 based 3.125x4 (10G) network Link interface with Ethernet MAC framing
- 16 Mbytes On Board SRAM

Uses Xilinx PCI Express Endpoint Core

Xilinx PCIe Endpoint Core

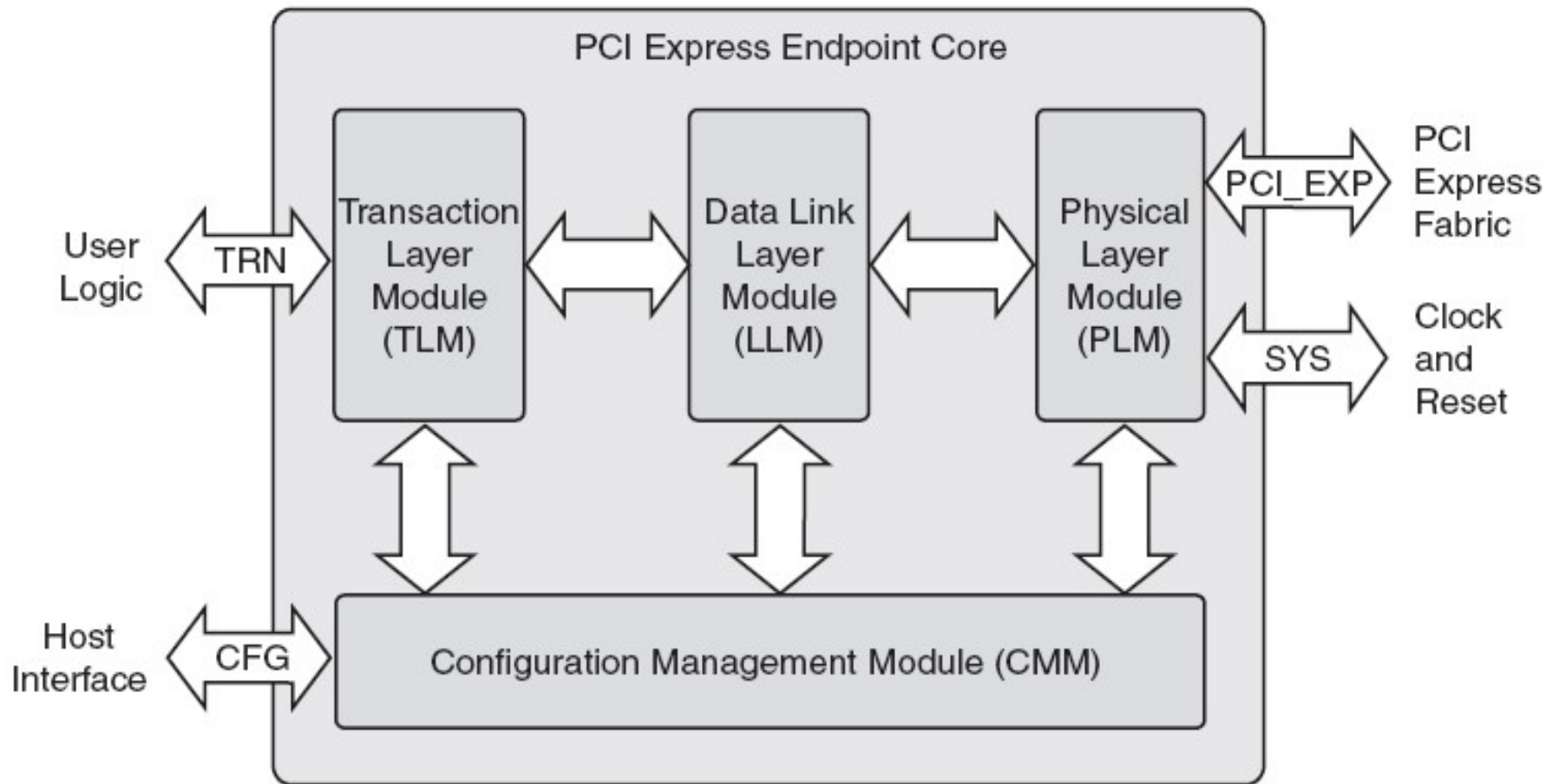


Figure 1: PCIe Endpoint Top-level Functional Blocks and Interfaces

PCI Express EndPoint Core

Functional Description

The Endpoint cores for PCI Express are organized into four main modules based on the three discrete logical layers defined by the *PCI Express Base Specification*.

The four logic modules, which manage all the system-level functions, include the following:

- Physical Layer Module (PLM)
- Data Link Layer Module (LLM)
- Transaction Layer Module (TLM)
- Configuration Management Module (CMM)

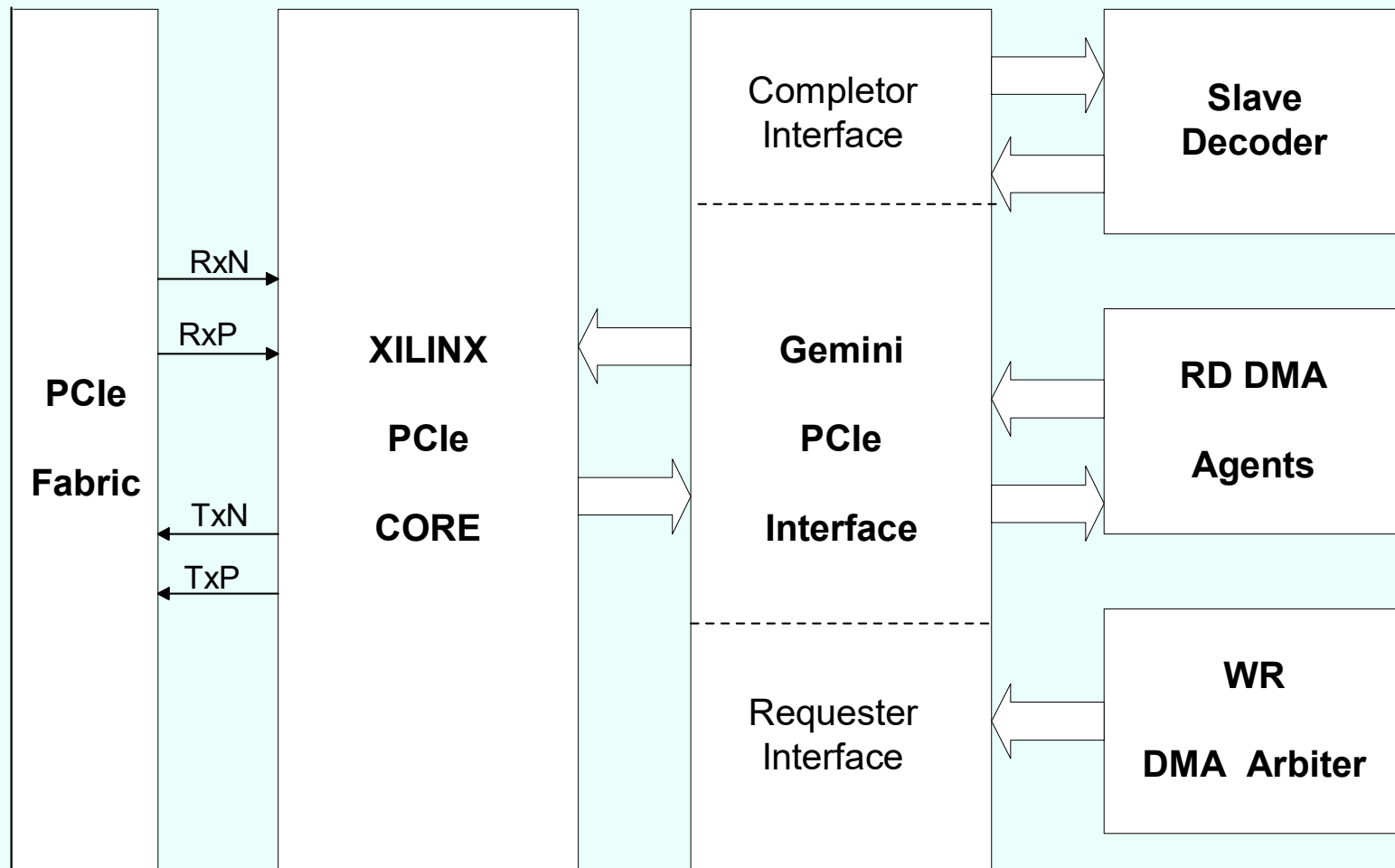
PCI Express End Point Core

Each module is further partitioned into the Receive and the Transmit sections. The Receive section processes the inbound information, and the Transmit section processes the outbound information.

The main modules interfaces with one another and the user application using the following set of four interfaces:

- System interface (SYS)
- PCI Express interface (PCI EXP)
- Configuration interface (CFG)
- Transaction interface (TRN)

Gemini PCIe Data Flow



Summary

PCI Express Protocol

- Packetized protocol: Time multiplexing versus circuit switching.

This allows more than two-way communication at one time

unlike circuit switching where only a two-way communications can occur.

With packet based protocol there is no wasted bandwidth.

- More bandwidth per pin
- Lane and Polarity reversal simplifies board routing

Bandwidth Comparison

Bus Type	Frequency (in MHz)	Peak Bandwidth (in Mbytes/sec)		Slots per Bus
		32 bit	64 bit	
PCI	33	133	266	4-5
PCI	66	266	533	1-2
PCI-X	66	266	533	4
PCI-X	133	533	1066	1-2
PCI-X	266(effective)	1066	2131	1
PCI-X	533(effective)	2131	4262	1

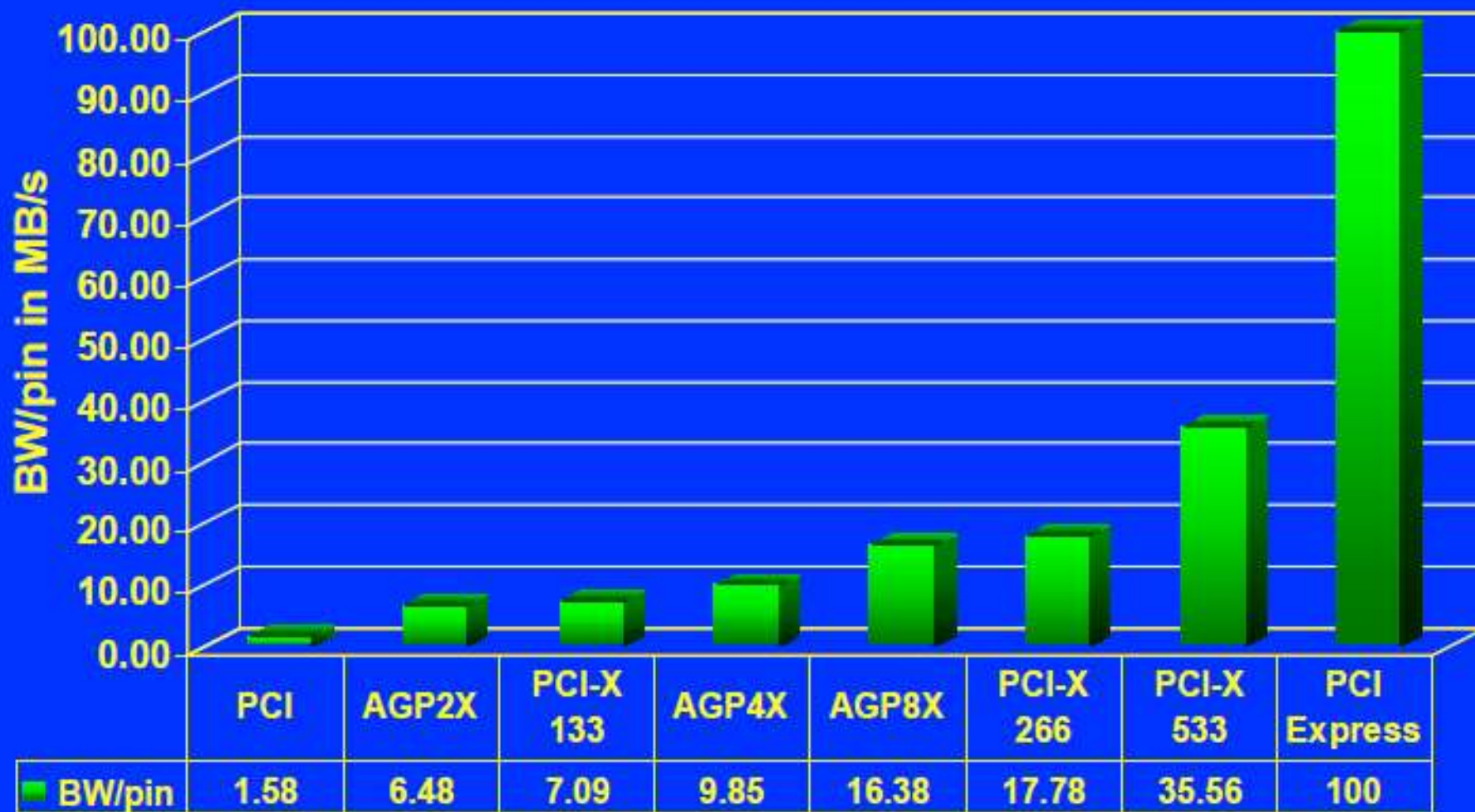
PCI Express Link Width	x1	x2	x4	x8	x12	x16	x32
Aggregate Bandwidth (Gbytes/sec)	0.5	1	2	4	6	8	16

PCI Express Link Performance

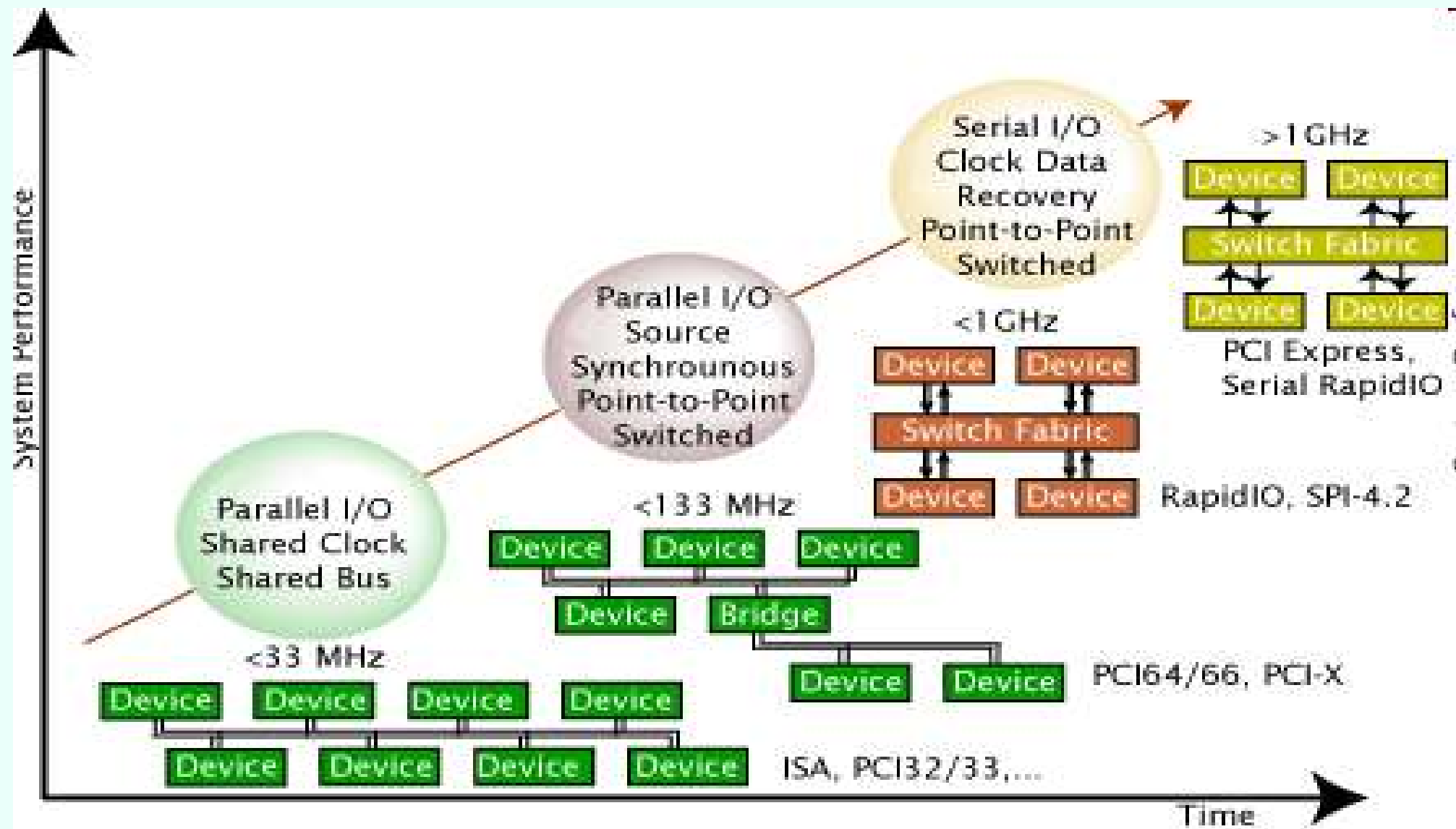
PCI Express version	Line code	Transfer rate	Bandwidth	
			Per lane	In a $\times 16$ (16-lane) slot
1.0	8b/10b	2.5 GT/s	2 Gbits/s (250 MB/s)	32 Gbit/s (4 GB/s)
2.0	8b/10b	5 GT/s	4 Gbit/s (500 MB/s)	64 Gbit/s (8 GB/s)
3.0	128b/130b	8 GT/s	7.877 Gbit/s (984.6 MB/s)	126.032 Gbit/s (15.754 GB/s)
4.0	128b/130b	16 GT/s	15.754 Gbit/s (1969.2 MB/s)	252.064 Gbit/s (31.508 GB/s)

- 20% overhead due to 8B/10B in Gen1 and Gen2
- ~1.5% overhead due to 128B/130B in Gen3 and above

PCI Express Performance



System Performance



PCI Express Features

Performance

- Scalable frequency (2.5Gbits/s per lane per direction)
- Scalable width (1,2,4,8,12,16,32)
- Low latency and high utilization (Requester/Completer model)

Protocol

- Packet based split transaction protocol
- Credit based Flow control
- Traffic Class Virtual channel mechanism for arbitration and priority of data transfer

PCI Express Features

Physical interface

- Point to point full duplex interconnect
- Differential low voltage signaling
- Embedded clocking

Advanced features

- Enhanced Configuration and Power management
- RAS – CRC based data integrity, Hot plug, Advanced Error logging and reporting
- QoS, Isochrony

Conclusion

Features

- PCI transparency
- TC/VC mechanism
- High bandwidth
- Flow control
- Reliable link layer

Benefits

- Smooth migration, s/w reuse, simple validation
- QoS & Isochrony
- Peak traffic loads, support high throughput applications
- Buffer size flexibility, Cost flexibility
- No dropped packets, simplified SW, high availability

Conclusion

Features

- Error reporting, Fault isolation
- ECRC
- Hot plug
- Power management
- High speed serial

Benefits

- System management, serviceability, availability
- End to end data Integrity
- Optimize density
- Reduced power consumption and emissions
- Reduced cost, pin count, PCB layers and area

References

- **PCI Express Base Specification Revision 1.1
(PCI-SIG)**
- **PCI Express Electrical Interconnect Design
Intel Press**
- **PCI Express System Architecture
MindShare, Inc**

Thank You

shilpav@cdac.in
HPC-Tech, C-DAC Pune