Practical introduction to PCI Express with FPGAs

Michal HUSEJKO, John EVANS
michal.husejko@cern.ch
IT-PES-ES

Agenda

- What is PCIe ?
  - System Level View
  - PCIe data transfer protocol
- PCIe system architecture
- PCIe with FPGAs
  - Hard IP with Altera/Xilinx FPGAs
  - Soft IP (PLDA)
  - External PCIe PHY (Gennum)

System Level View

- Interconnection
- Top-down tree hierarchy
- PCI/PCIe configuration space
- Protocol

Interconnection

- Serial interconnection
- Dual uni-directional
- Lane, Link, Port
- Scalable
  - Gen1 2.5/ Gen2 5.0/ Gen3 8.0 GT/s
  - Number of lanes in FPGAs: x1, x2, x4, x8
- Gen1/2 8b10b
- Gen3 128b/130b

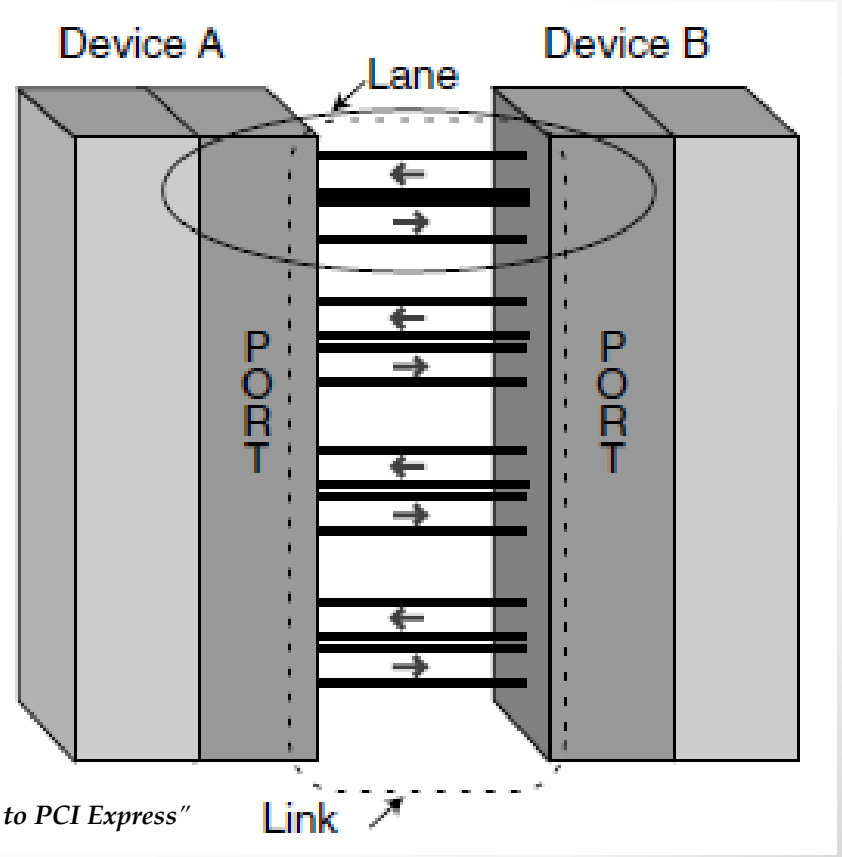


*Image taken from "**Introduction to PCI Express**"*

Tree hierarchy

- Top-down tree hierarchy with single host
- 3 types of devices: Root Complex, Endpoint, Switch
- Point-to-point connection between devices without sideband signalling
- 2 types of ports: downstream/upstream
- Configuration space

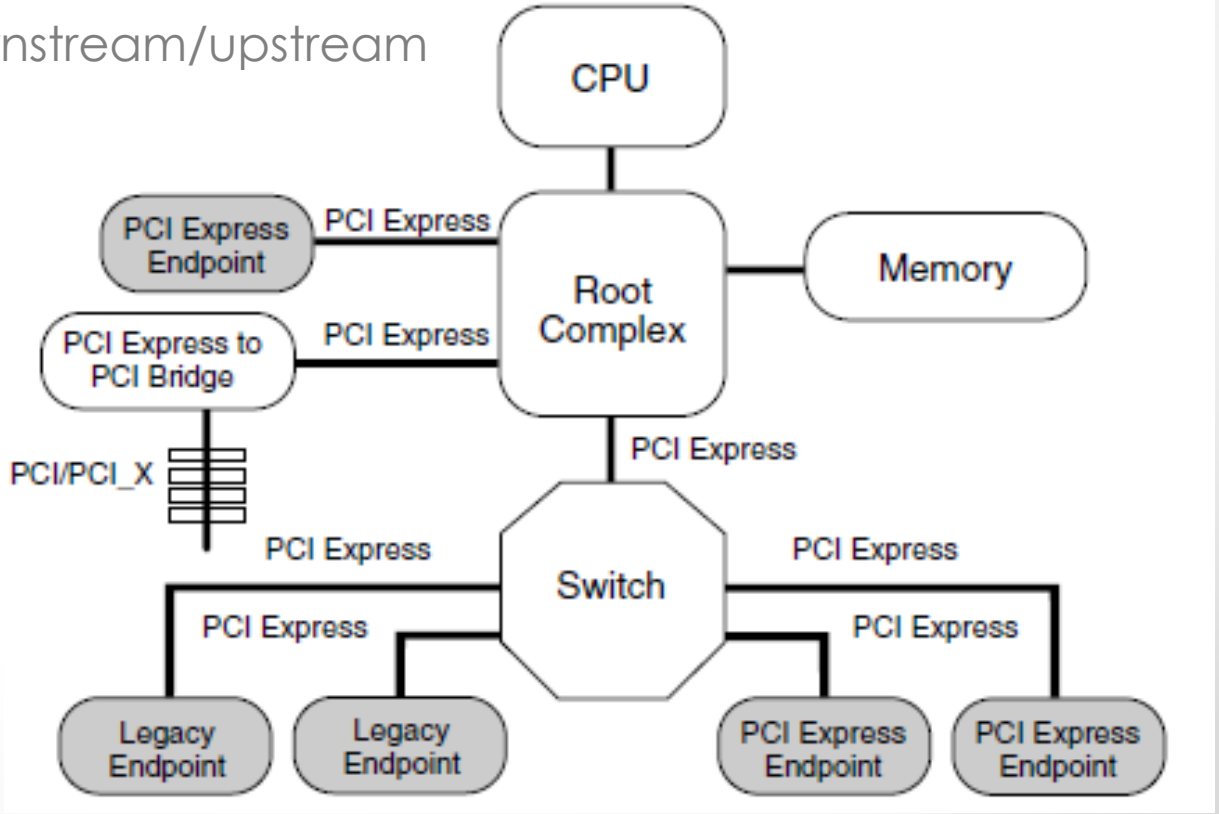


*Image taken from "**Introduction to PCI Express**"*

PCIe Configuration space

- Similar to PCI conf space – binary compatible for first 256 bytes
- Defines device(system) capabilities
- Clearly identifies device in the system
  - Device ID
  - Vendor ID
  - Function ID
  - All above
- and defines memory space allocated to device.

PCIe transfer protocol

- Transaction categories
- Protocol
- Implementation of the protocol

Transaction categories

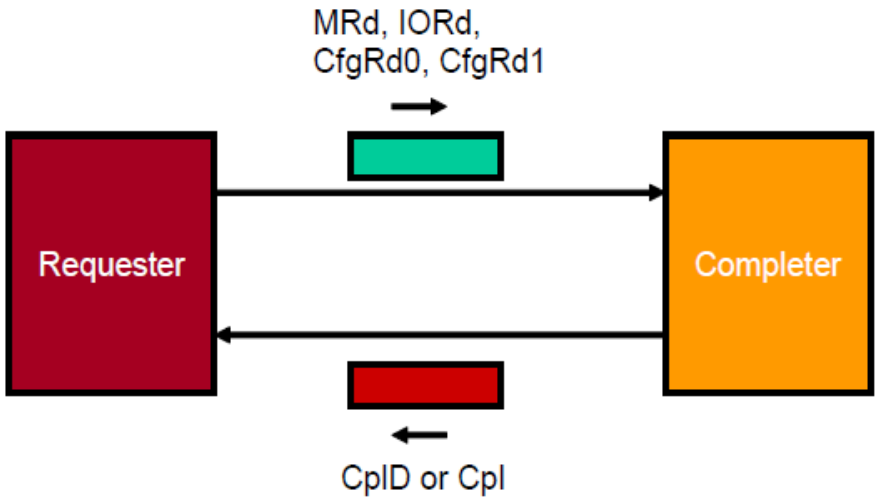
- Configuration – move downstream
- Memory – address based routing
- IO – address based routing
- Message – ID based routing

Transaction Types

Transaction Type	Non-Posted or Posted
Memory Read	Non-Posted
Memory Write	Posted
Memory Read Lock	Non-Posted
IO Read	Non-Posted
IO Write	Non-Posted
Configuration Read (Type 0 and Type 1)	Non-Posted
Configuration Write (Type 0 and Type 1)	Non-Posted
Message	Posted

*Table taken from "**PCI Express System Architecture**"*

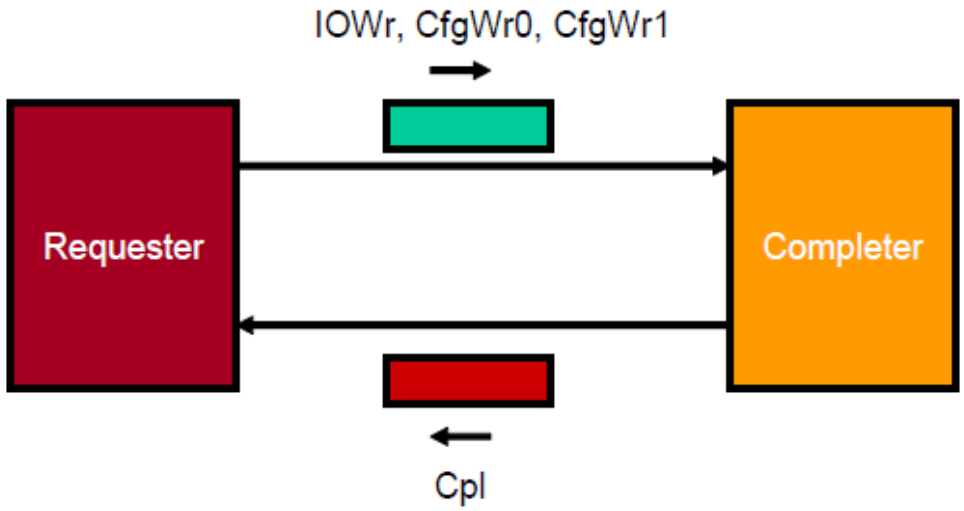
Non-posted read transactions



Legend:
MRd = Memory Read Request
IORd = IO Read Request
CfgRd0 = Type 0 Configuration Read Request
CfgRd1 = Type 1 Configuration Read Request
CplD = Completion with data for normal completion of MRd, IORd, CfgRd0, CfgRd1
Cpl = Completion without data for error completion of MRd, IORd, CfgRd0, CfgRd1

*Image taken from "**PCI Express System Architecture**"*

Non-Posted write transactions



Legend:
IOWr = IO Write Request
CfgWr0 = Type 0 Configuration Write Request
CfgWr1 = Type 1 Configuration Write Request
Cpl = Completion without data for normal or error completion of IOWr, CfgWr0, CfgWr1

*Image taken from "**PCI Express System Architecture**"*

Posted Memory Write transactions

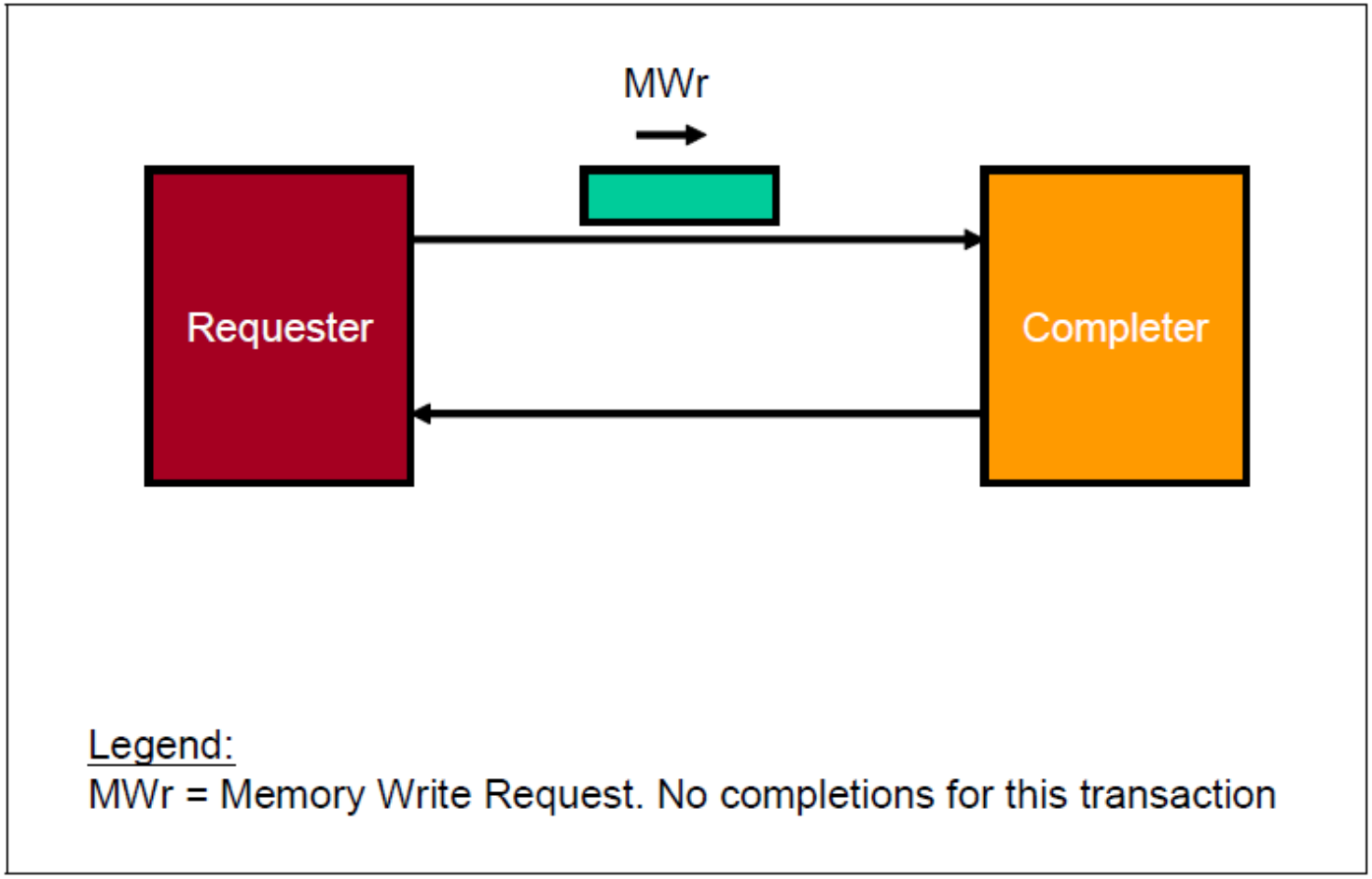


*Image taken from "**PCI Express System Architecture**"*

Posted Message transactions

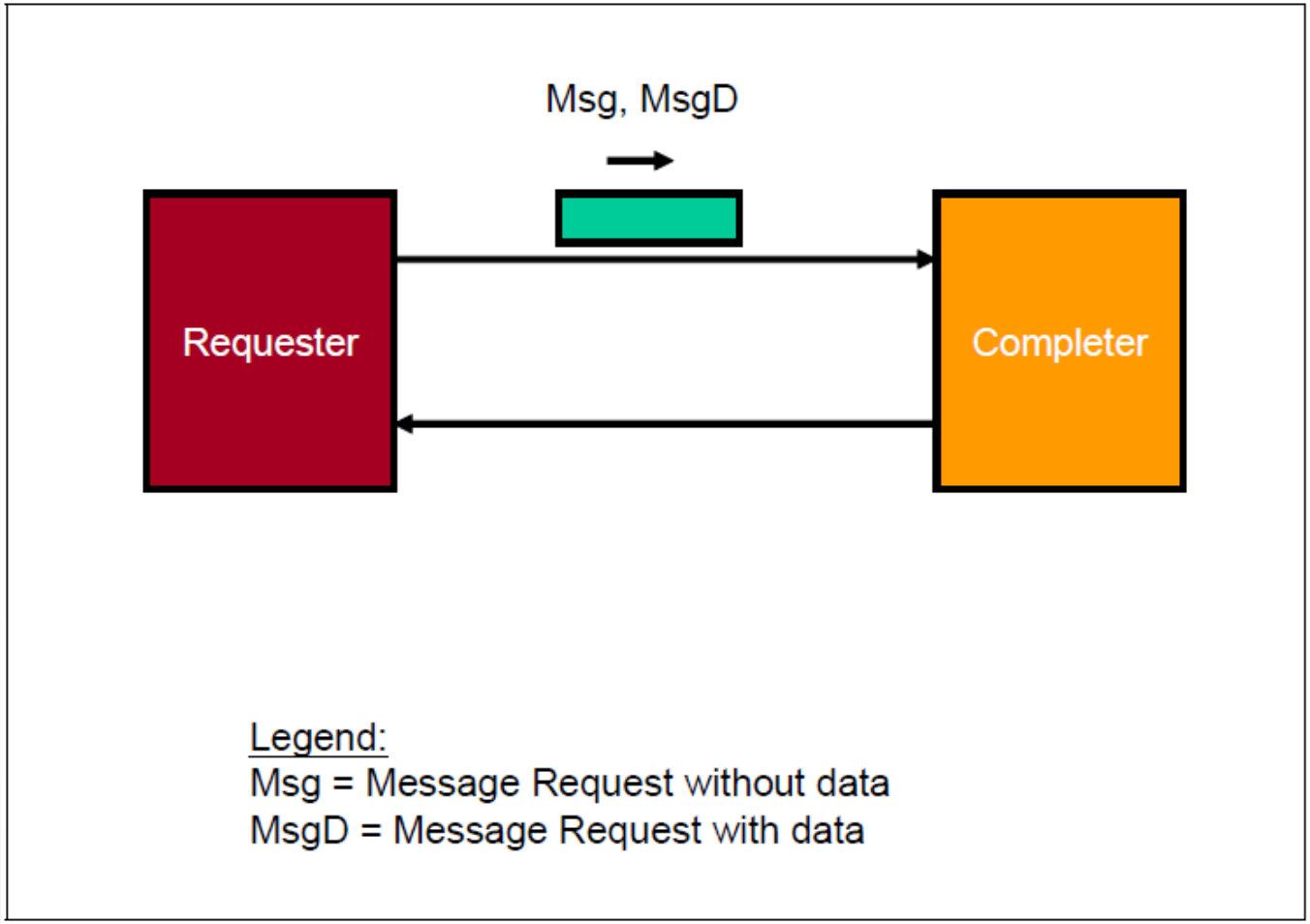


*Image taken from "**PCI Express System Architecture**"*

PCIe Device Layers

- 3 layer protocol
- Each layer split into TX and RX parts
- Ensures reliable data transmission between devices

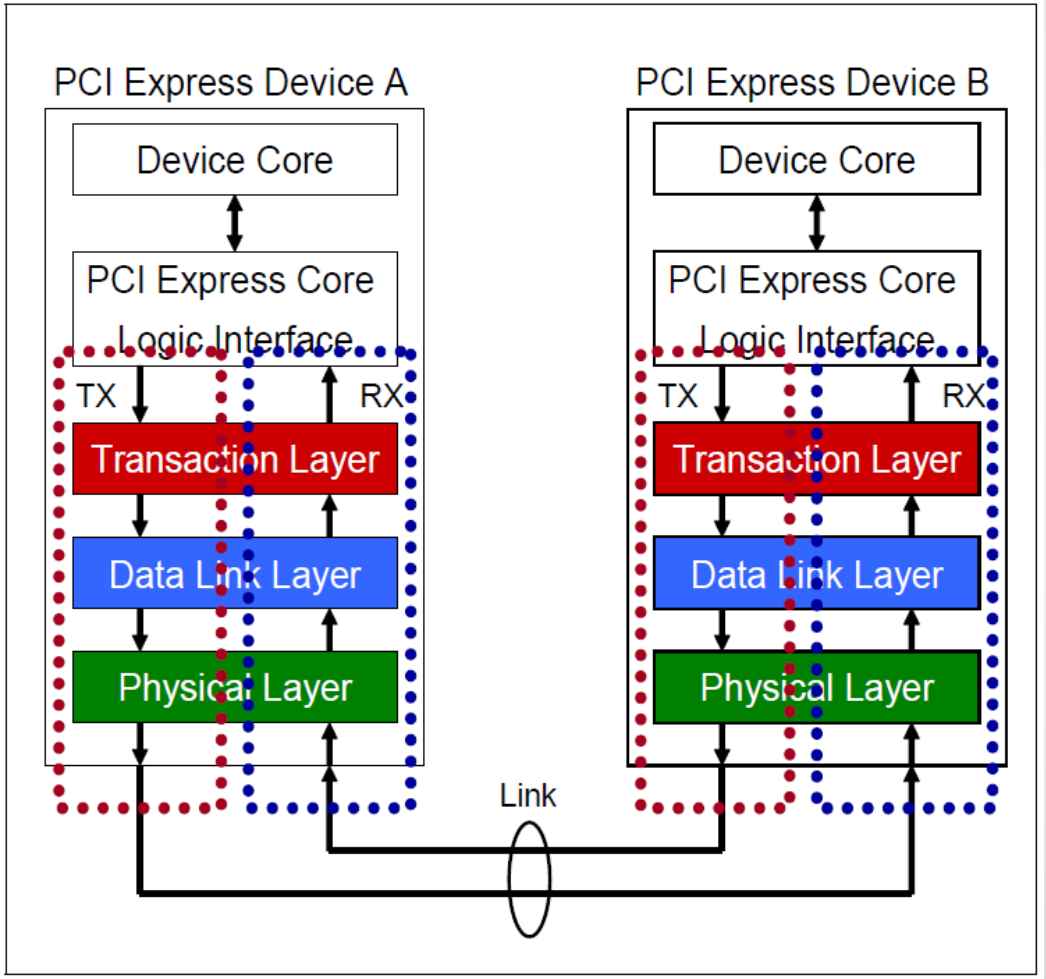


*Image taken from "**PCI Express System Architecture**"*

Physical Layer

- Contains all the necessary digital and analog circuits
- Link initialization and training
  - Link width
  - Link data rate
  - Lane reversal
  - Polarity inversion
  - Bit lock per lane
  - Symbol lock per lane
  - **Lane-to-lane deskew**

Data Link layer

- Reliable transport of TLPs from one device to another across the link
- It's done by using DLL packets:
  - TLP acknowledgement
  - Flow control
  - Power Management

Transaction layer

- It turns user application data or completion data into PCIe transaction – TLP
- Header + Payload + ECRC
- used in FPGAs IPs

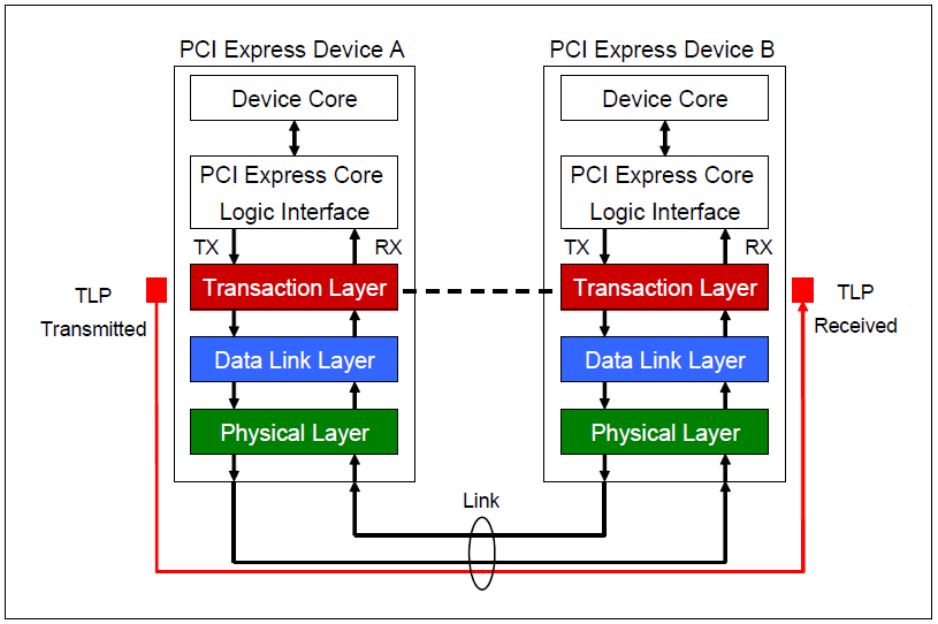


*Image taken from "**PCI Express System Architecture**"*

Flow control

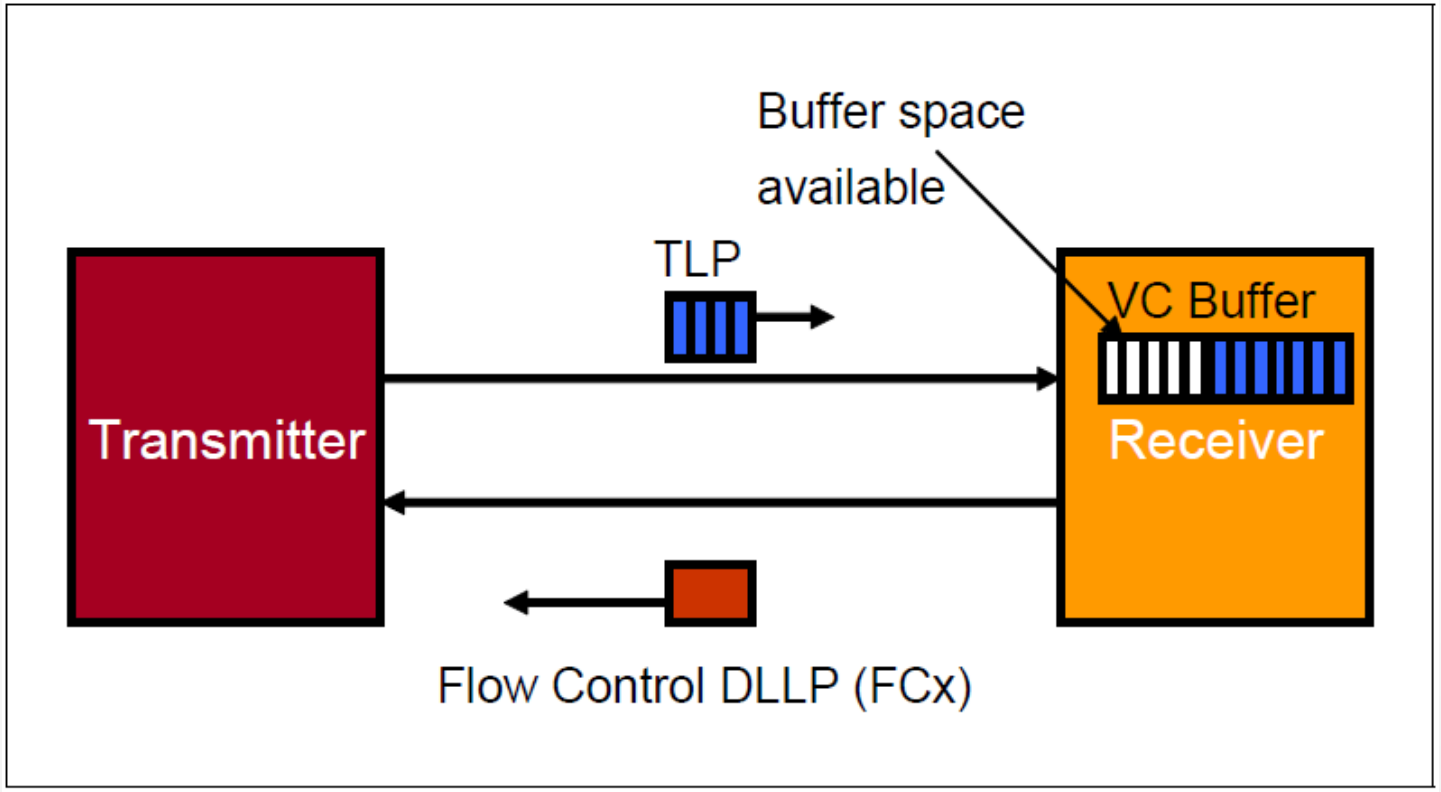


*Image taken from "**PCI Express System Architecture**"*

Flow control – posted transaction

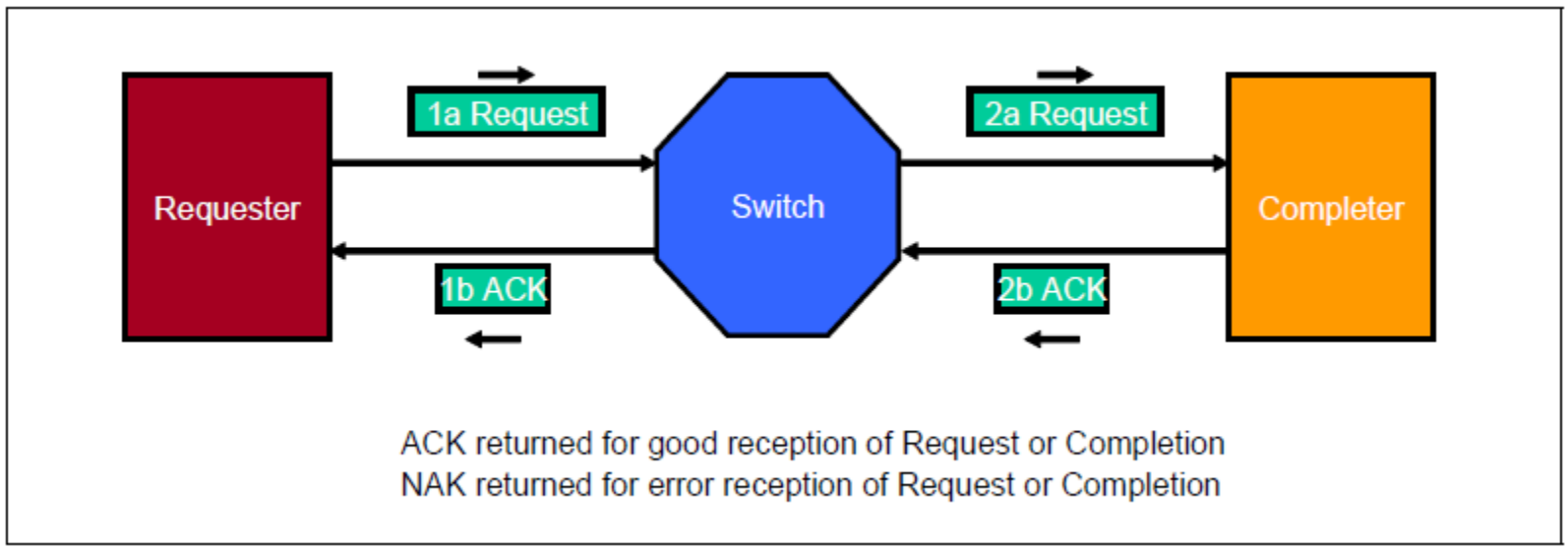


*Image taken from "**PCI Express System Architecture**"*

Flow control – non-posted transaction

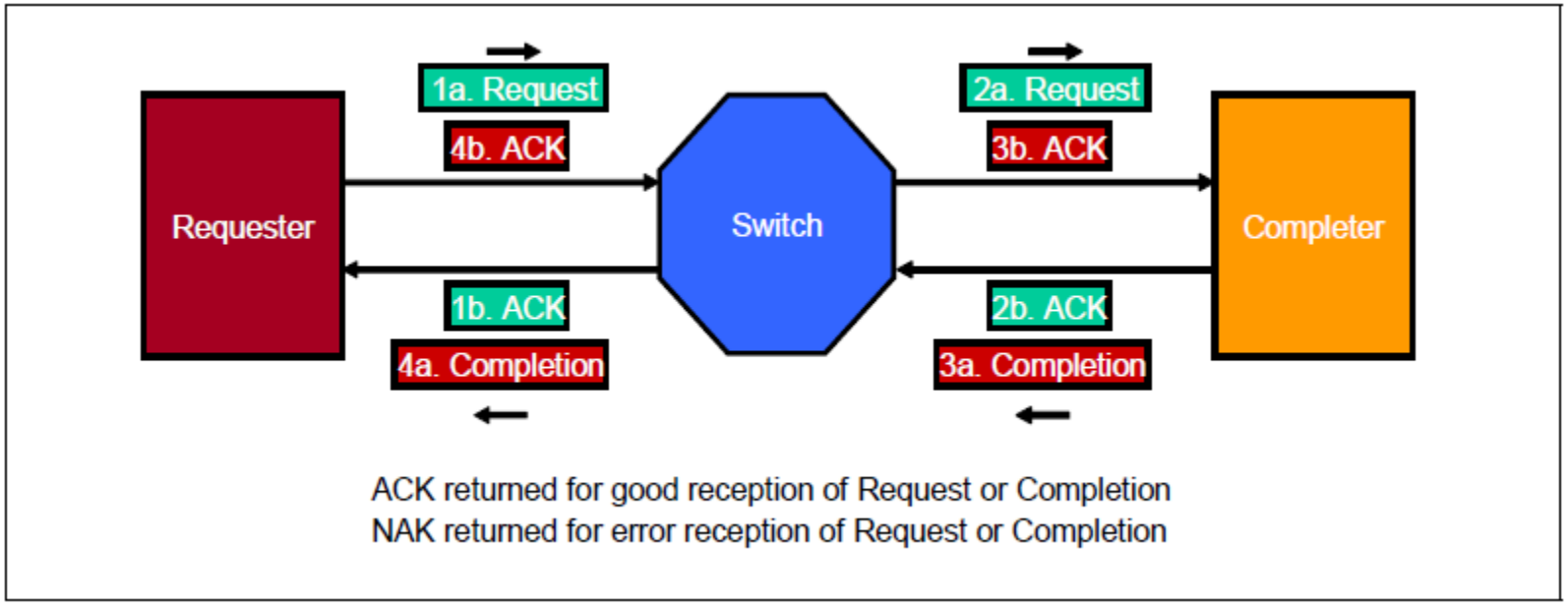


*Image taken from "**PCI Express System Architecture**"*

Building transaction

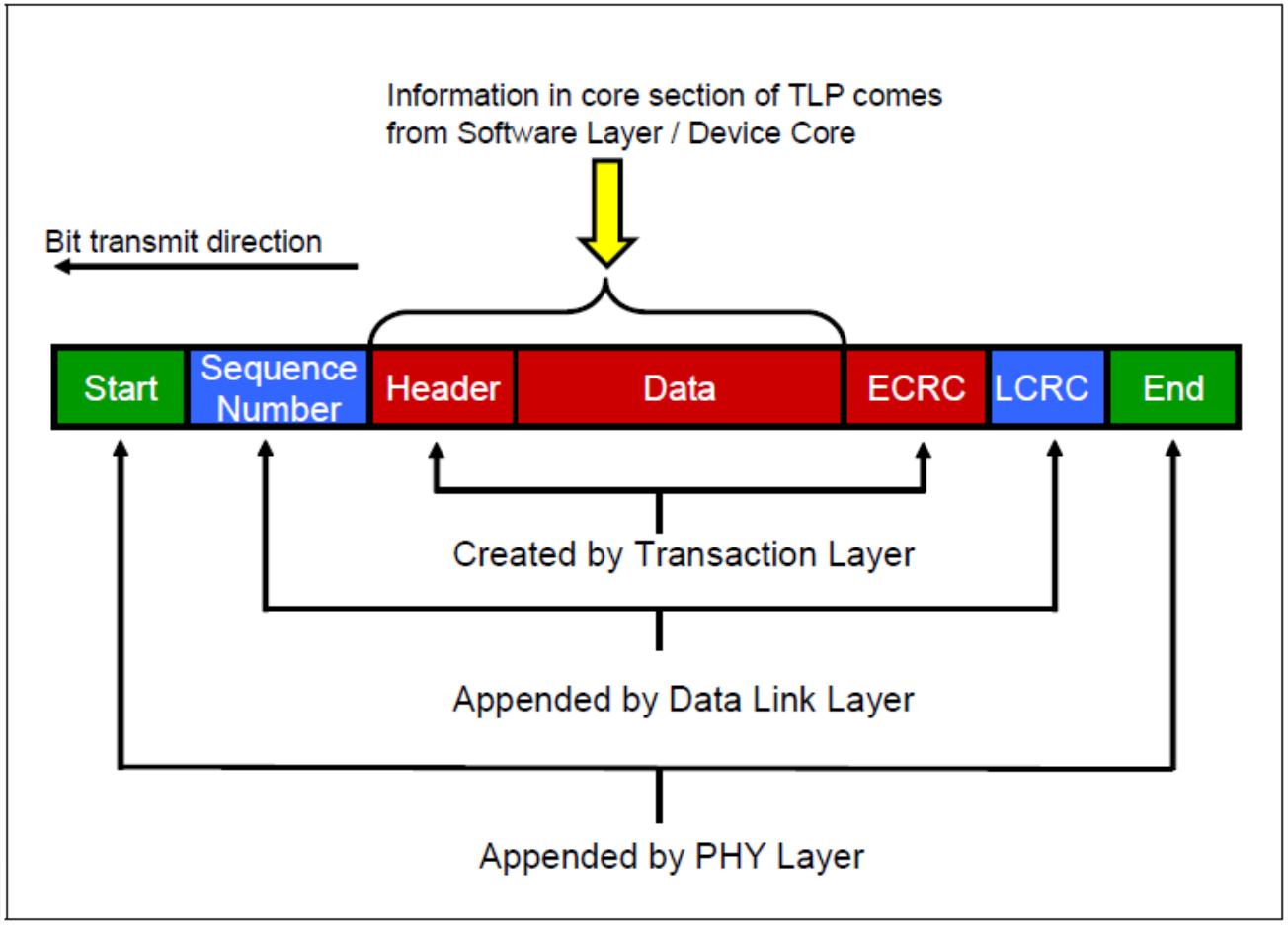


*Image taken from "**PCI Express System Architecture**"*

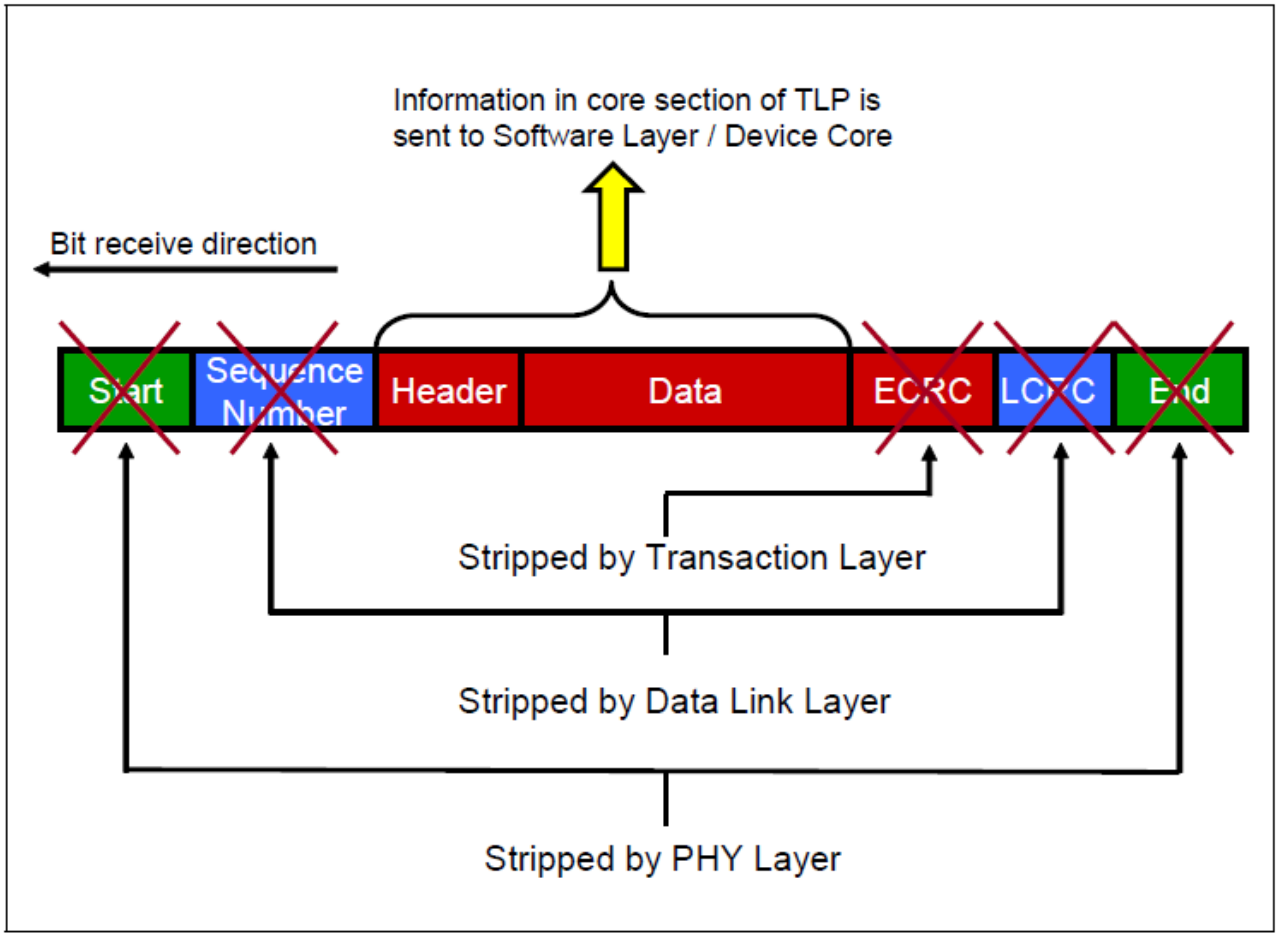


*Image taken from "**PCI Express System Architecture**"*

Example

CPU MRd targeting an Endpoint

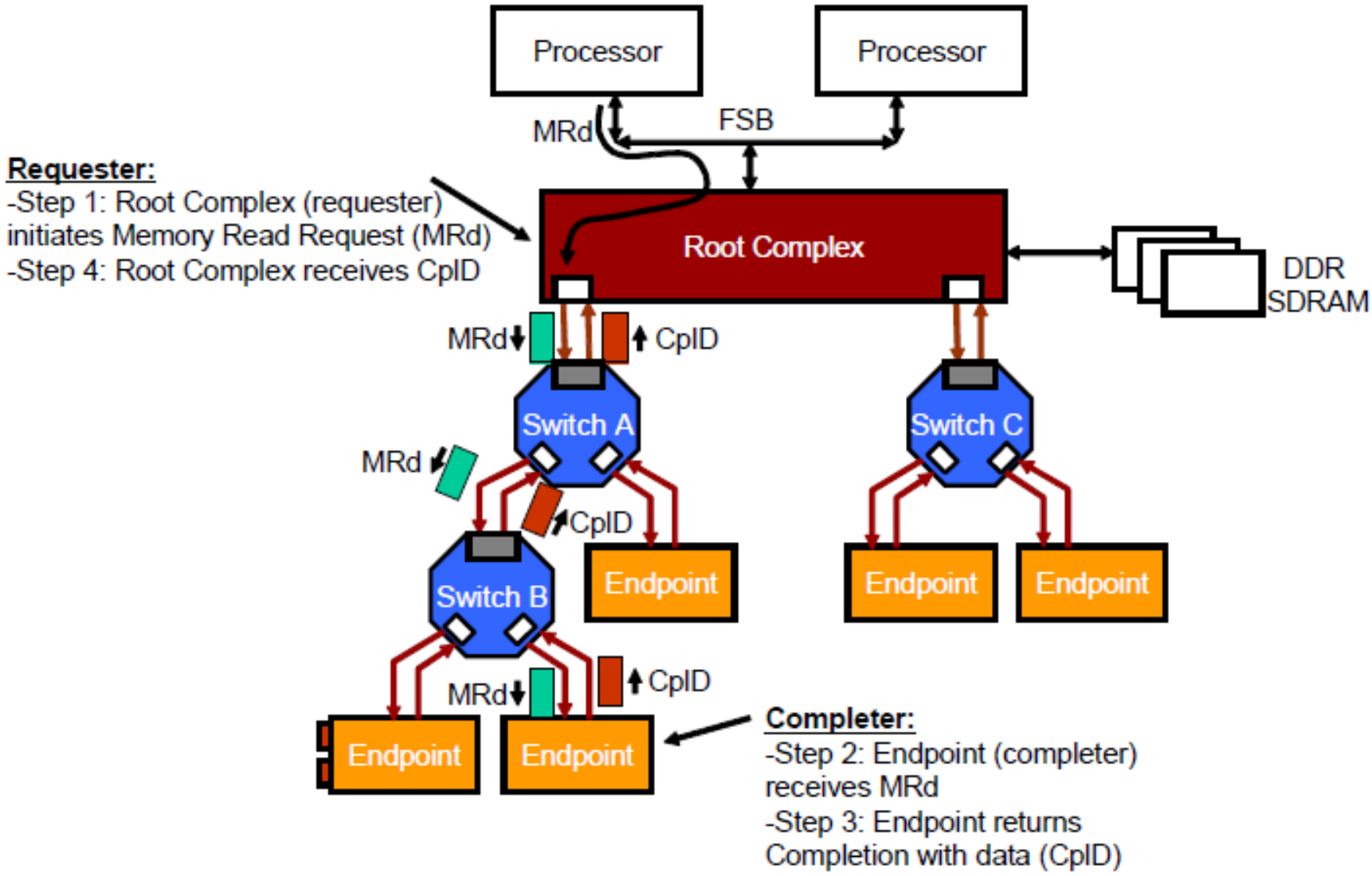


*Image taken from "**PCI Express System Architecture**"*

CPU MWr targeting Endpoint

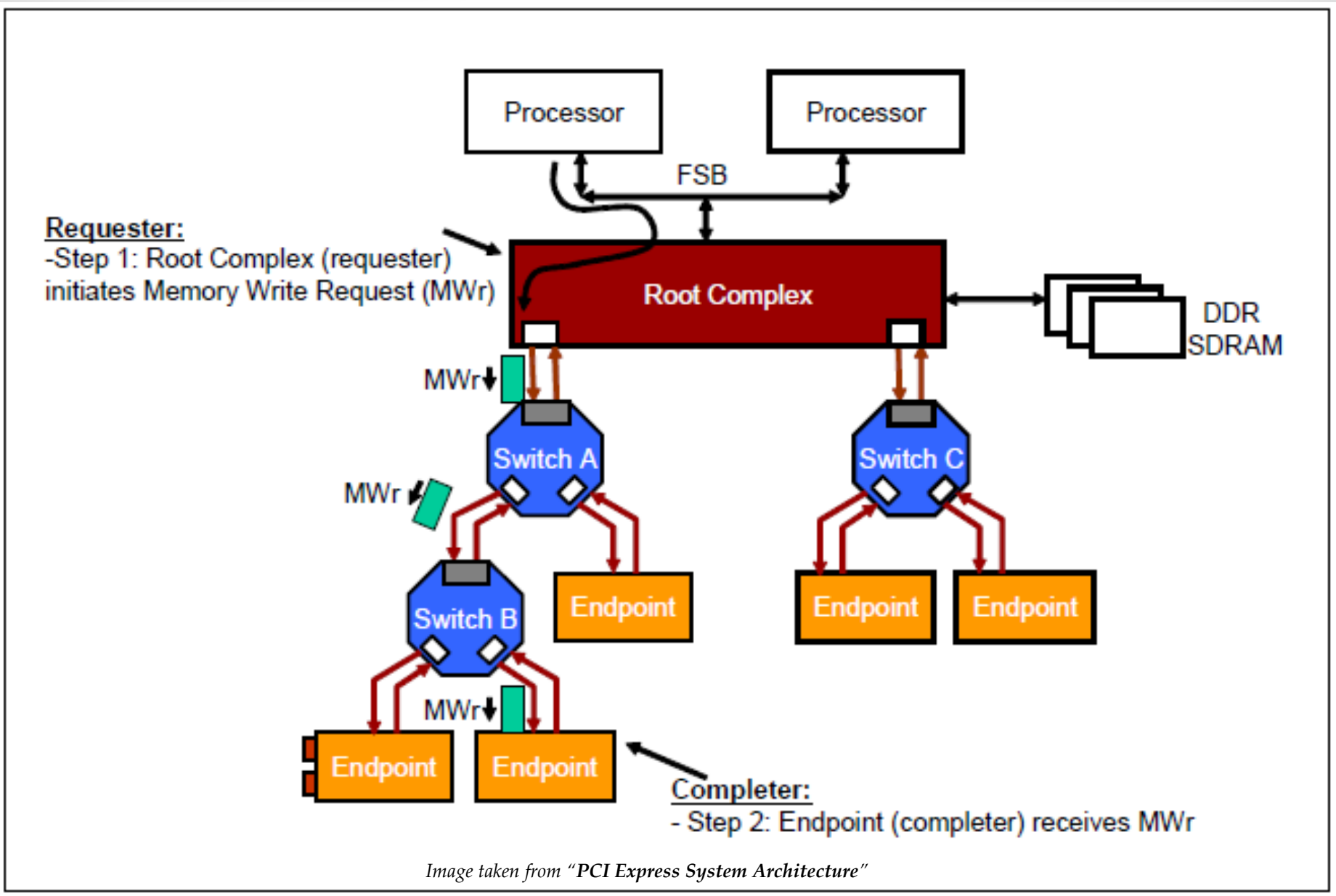


*Image taken from "**PCI Express System Architecture**"*

Endpoint MRd targeting system memory

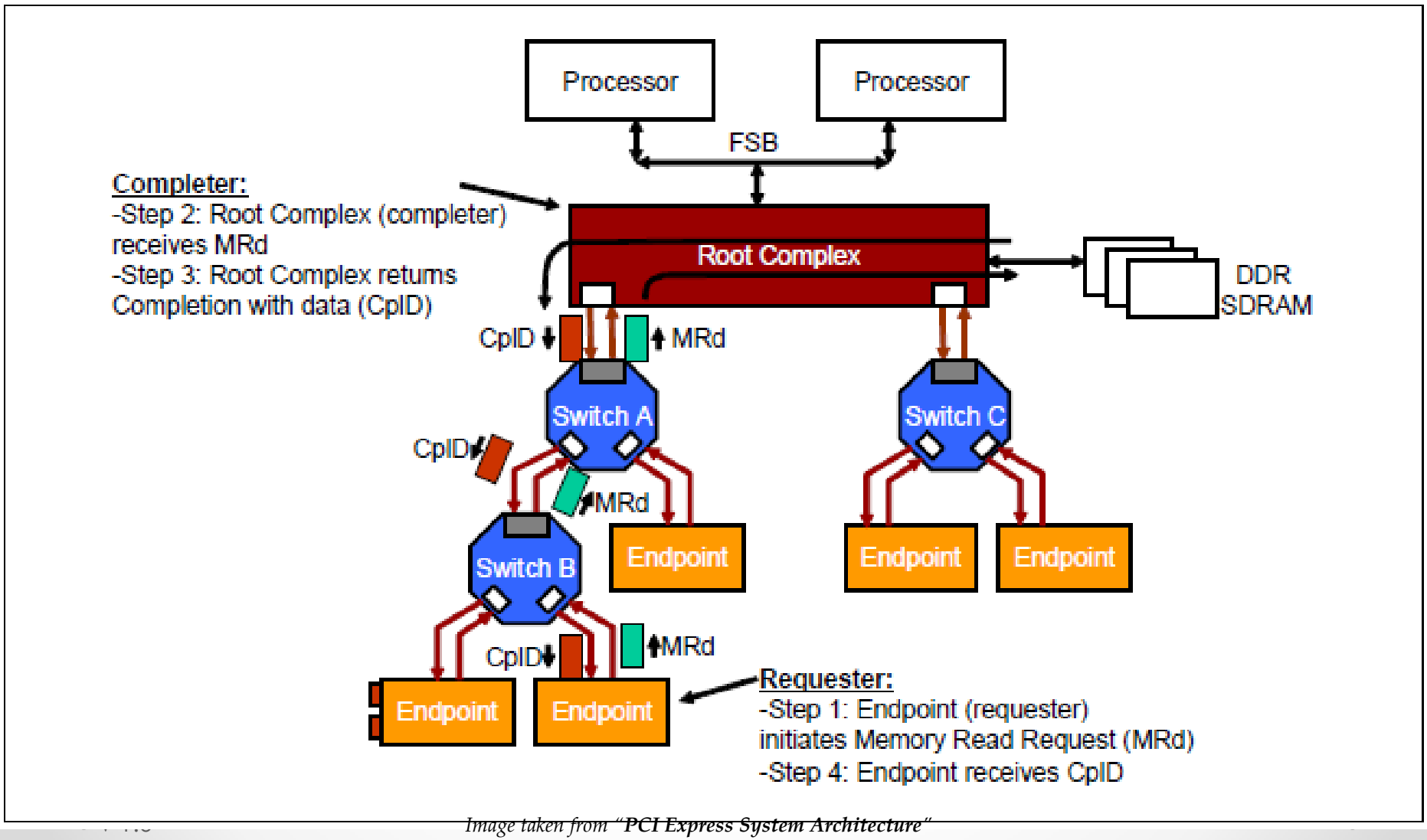


*Image taken from "**PCI Express System Architecture**"*

Packet constraints

- Maximum Payload Size (MPS)
  - default 128 Bytes
  - least denominator of all devices in the tree
- Maximum Read Request Size (MRRS)
  - Defined by RC
- Maximum Payload/ Read req. size 4 kB
  - defined by spec
  - No 4kB boundary crossing allowed
- Example: Intel x58 : MPS=256B, MRRS=512B

HEADER description

- Little endian
- 3DW or 4DW (Double Word – 4 bytes)

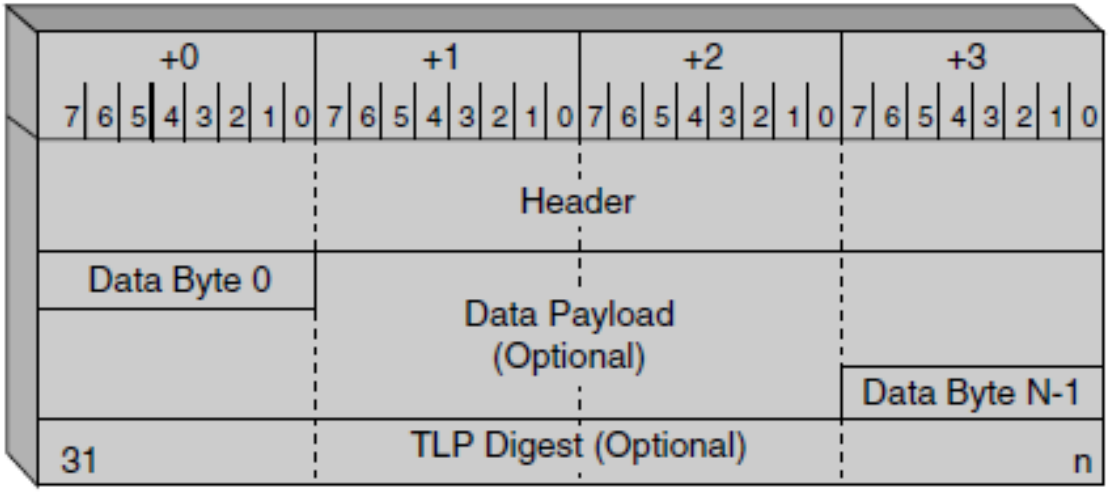


*Image taken from "**Introduction to PCI Express**"*

HEADER – base part

- Fmt – size of the header, is there payload ?
- Length – in DW
- EP – Poisoned
- TC – Traffic class
- TD – TLP digest – ECRC field
- Attr – status (success, aborted)

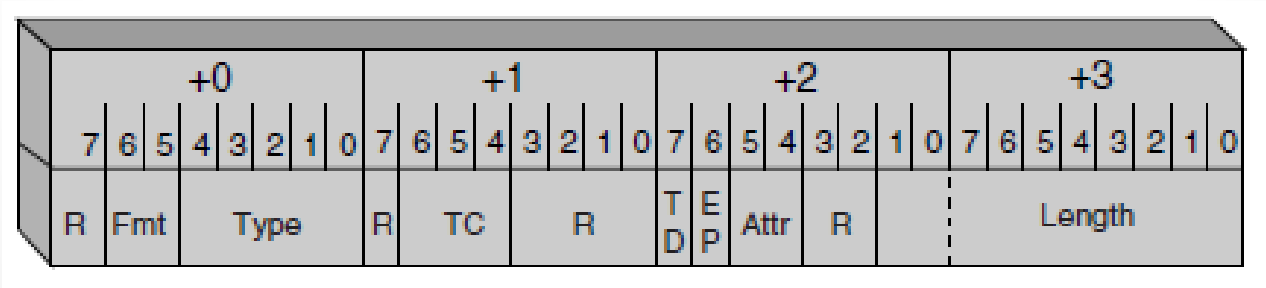
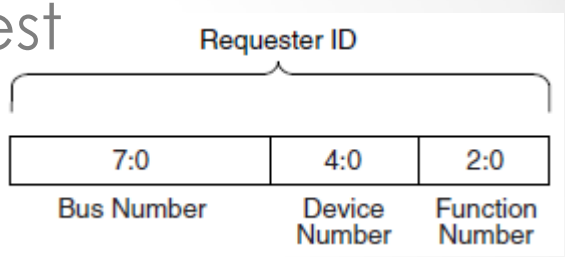


*Image taken from "**Introduction to PCI Express**"*

HEADER Memory Request

- TAG - Number of outstanding request
- Requester ID



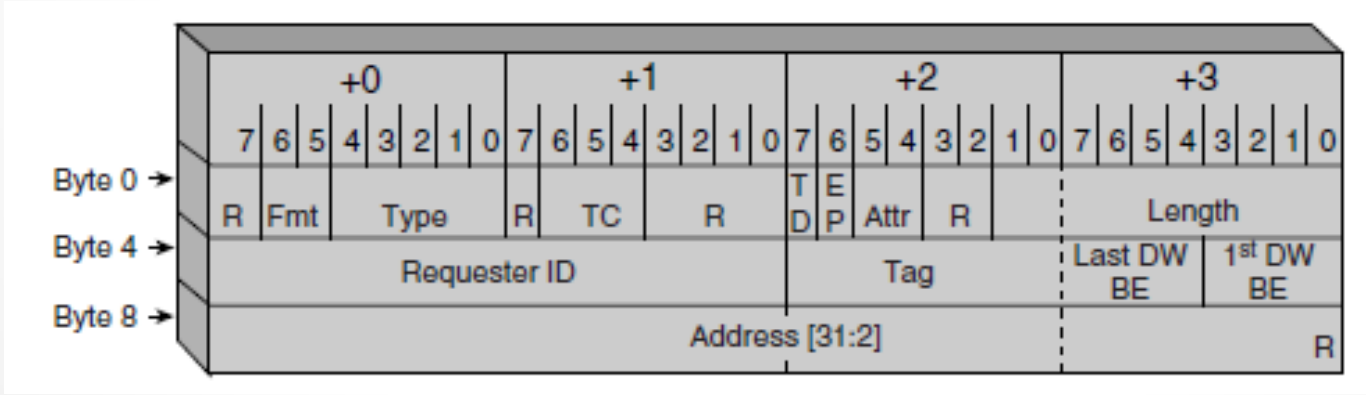
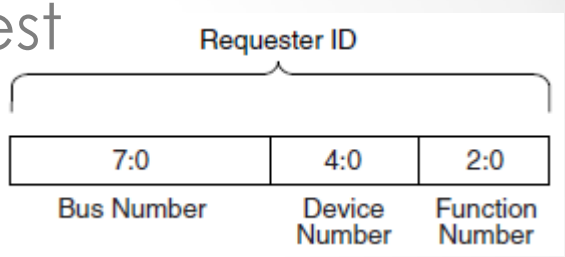


*Image taken from "**Introduction to PCI Express**"*

HEADER Completion

- TAG - Number of outstanding request
- Requester ID



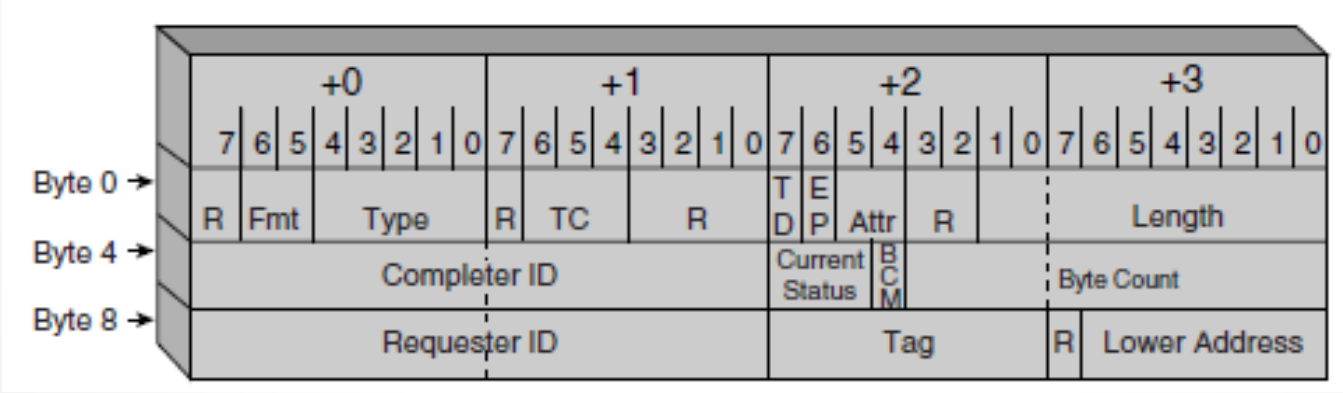


*Image taken from "**Introduction to PCI Express**"*

PCIe System Architecture

- Switches
  - Extend interconnection possibilities
  - DMA
  - Performance improvement functions
  - Non Transparent Bridging
- Extending distance
  - Bus re-drivers
  - Copper and optical cables

PCIe switches

- Non Transparent Bridging (NTB)
- Virtual Partitioning
- Multicasting
- DMA
- Failover

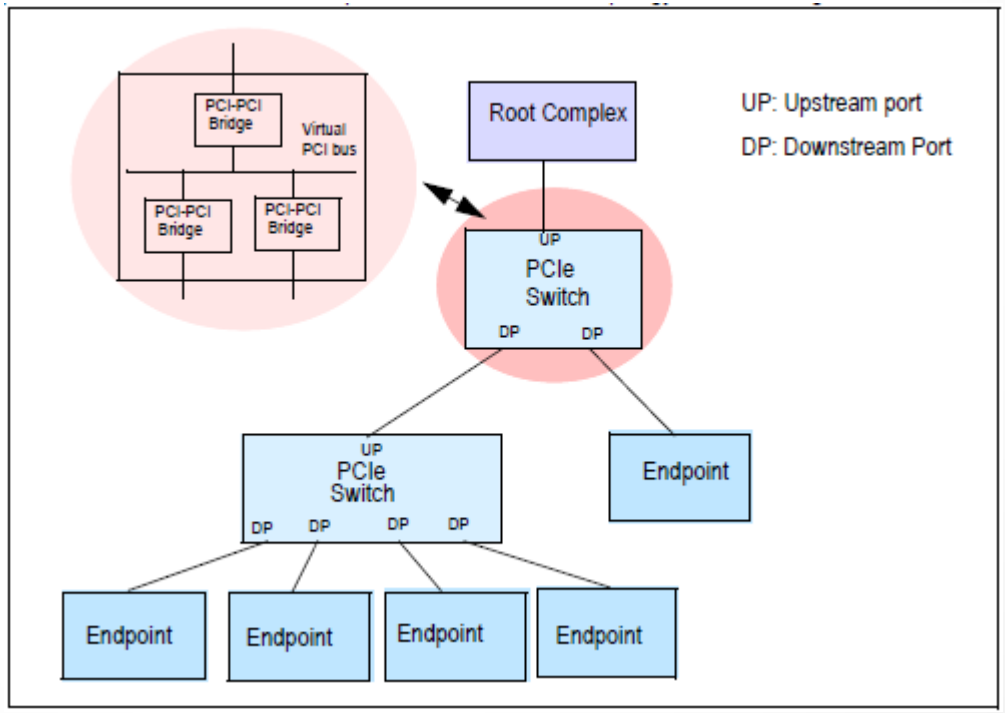
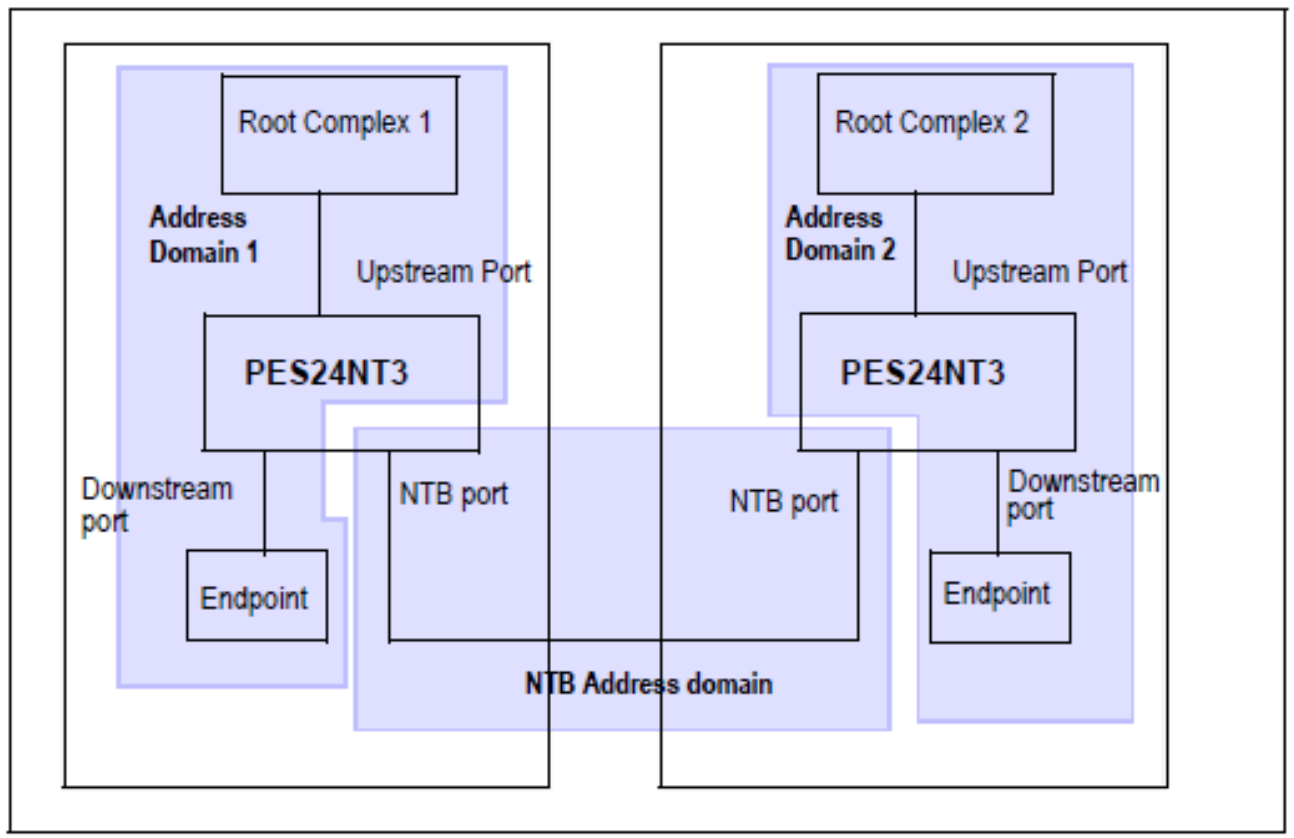


Image taken from IDT documentation

NTB + Virtual Partitioning

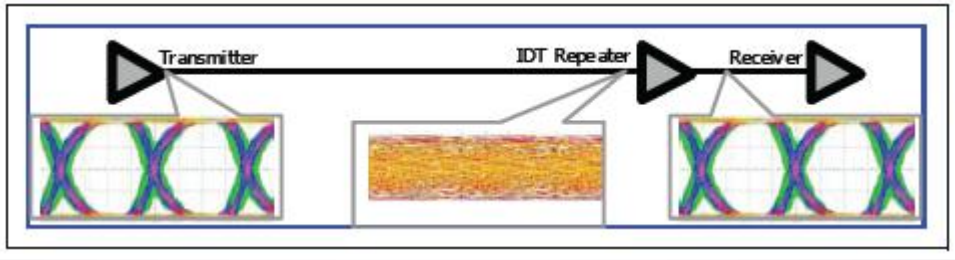


Cabling

- Copper cables
- Optical cables
- Cable re-drivers(repeaters)

Image taken from www.ioxos.ch

http://www.alpenio.com/products/pciex4.html



www.idt.com

PCIe with FPGAs

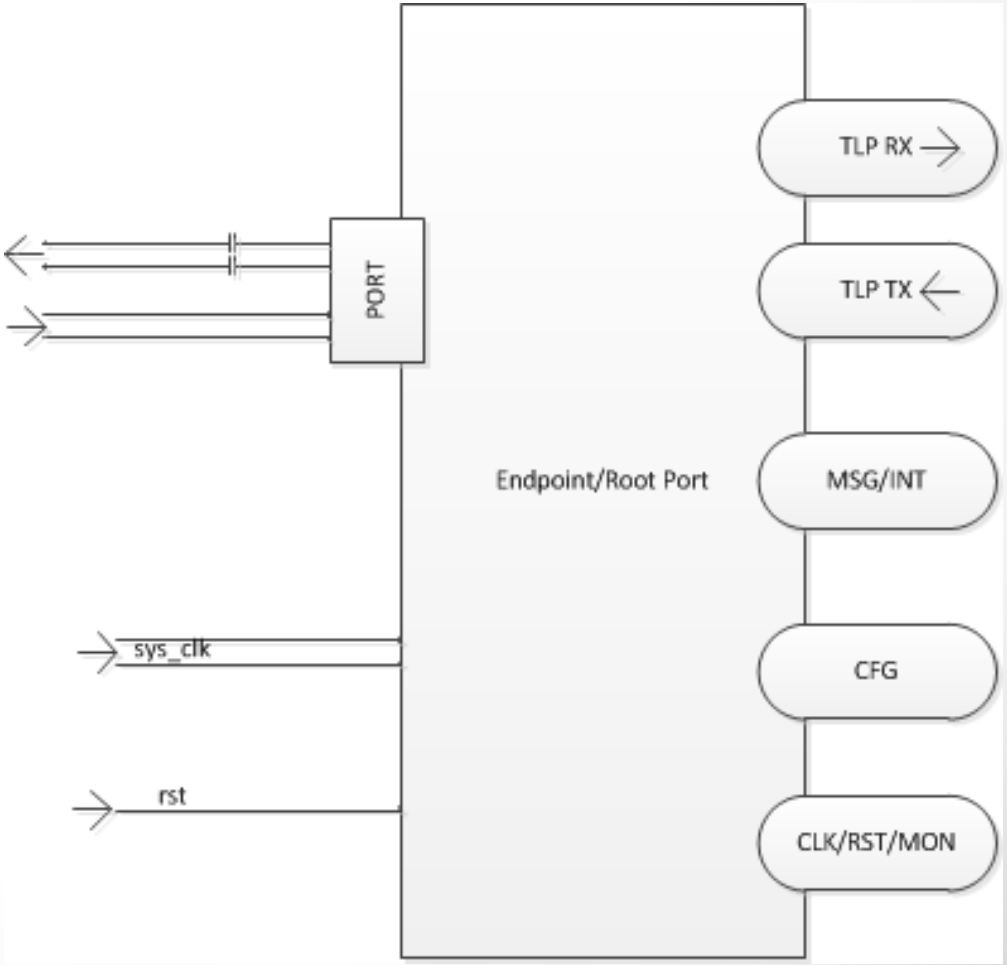
- Technology overview:
  - Hard IP – Altera and Xilinx
  - Soft IP – PLDA
  - External PHY – Gennum PCIe to local bus bridge
- Vendor documents – app notes, ref designs, Linux/Win device drivers
- Simulation – Endpoint/Root port

Xilinx Hard IP solution

- User backend protocol same for all devices
  - Spartan – 6
  - Virtex – 5
  - Virtex – 6
  - Virtex – 7
- Xilinx Local Link (LL) Protocol and ARM AXI
- For new designs: use AXI
- Most of the Xilinx PCIe app notes uses LL

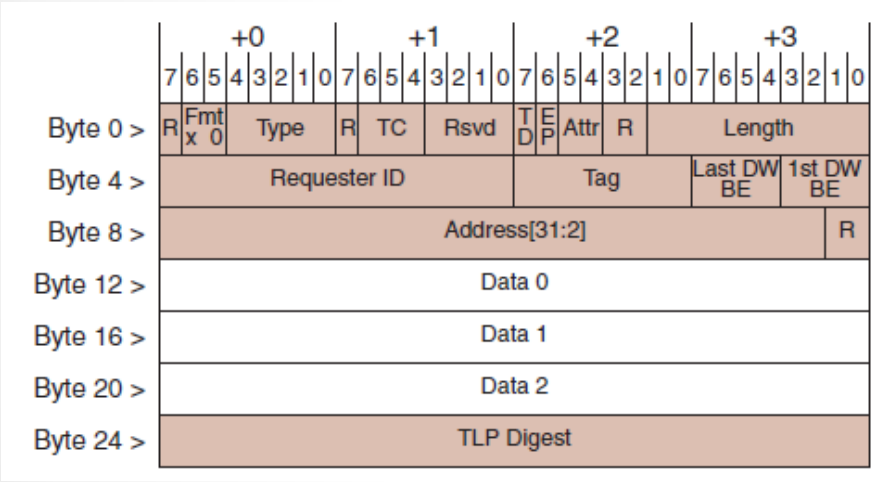
Xilinx Hard IP interface

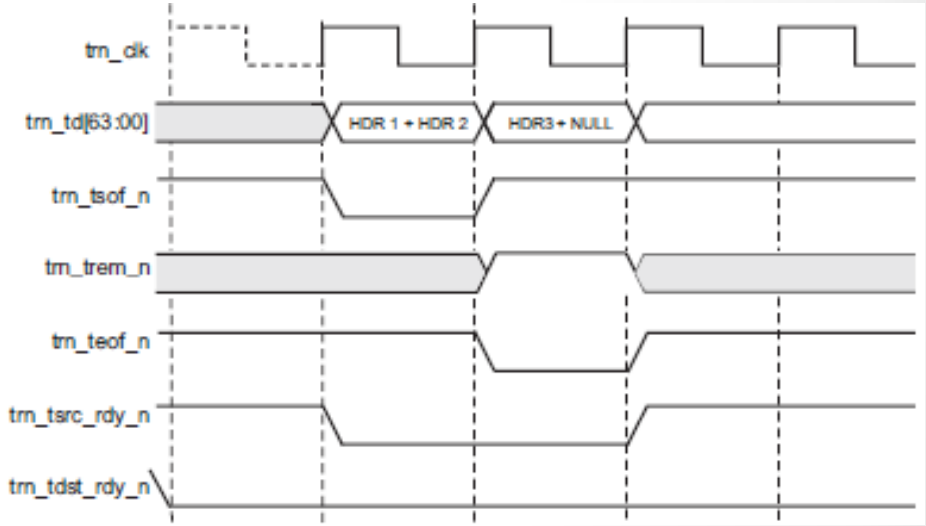
- External world: gt, clk, rst – (example x1 needs 7 wires)
- CLK/RST/Monitoring
- TLP TX if
- TLP RX if
- CFG if
- MSG/INT if

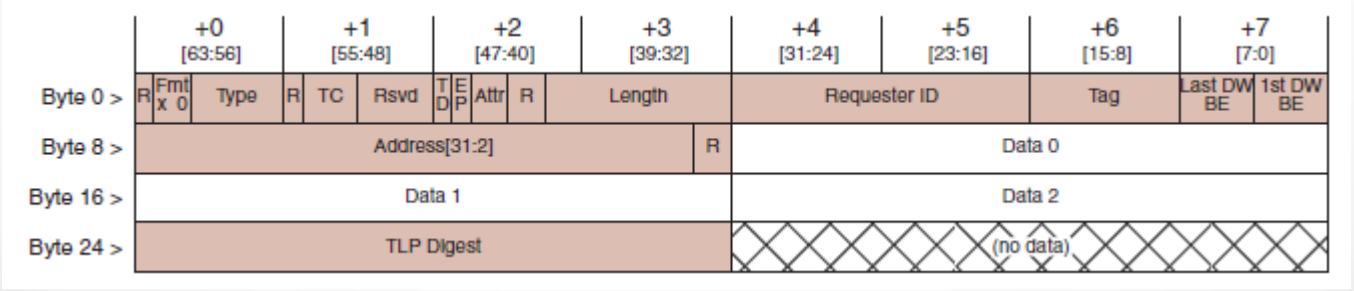


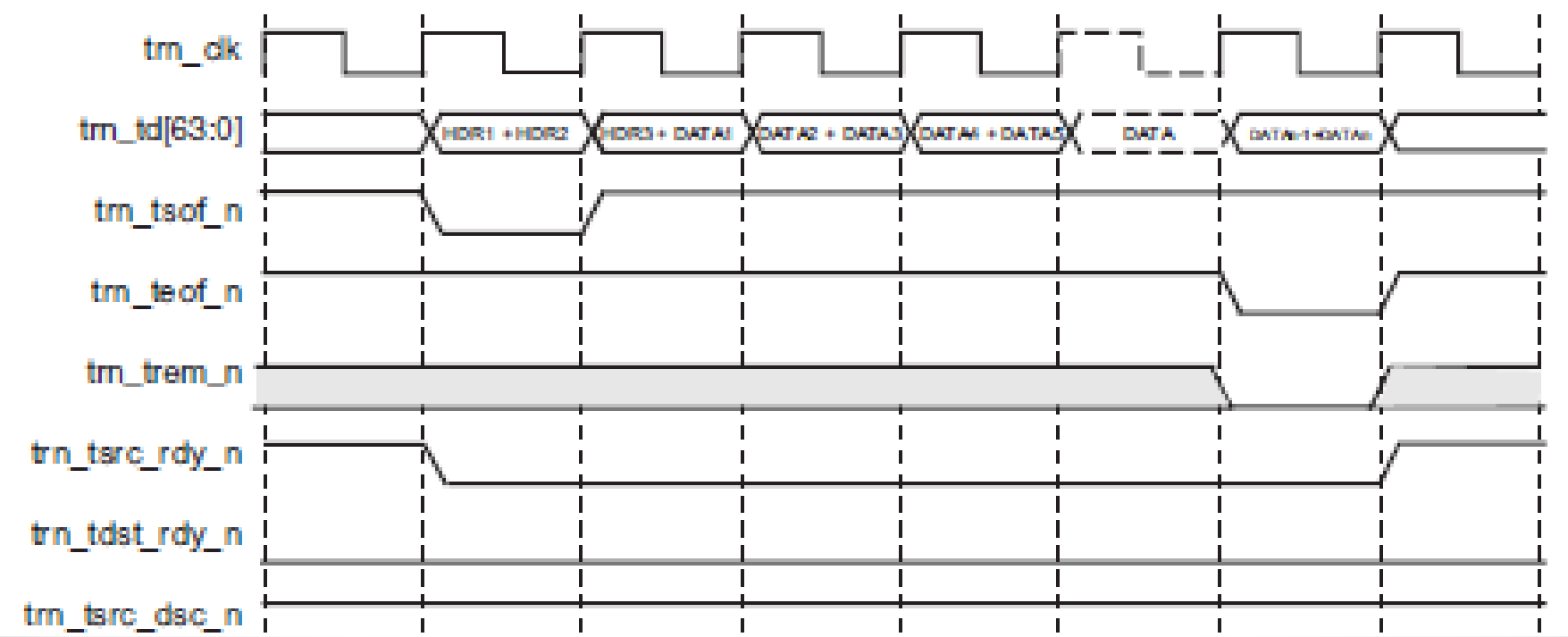
PCIe LL protocol

- TLP packets are mapped on 32/64/128 bit TRN buses



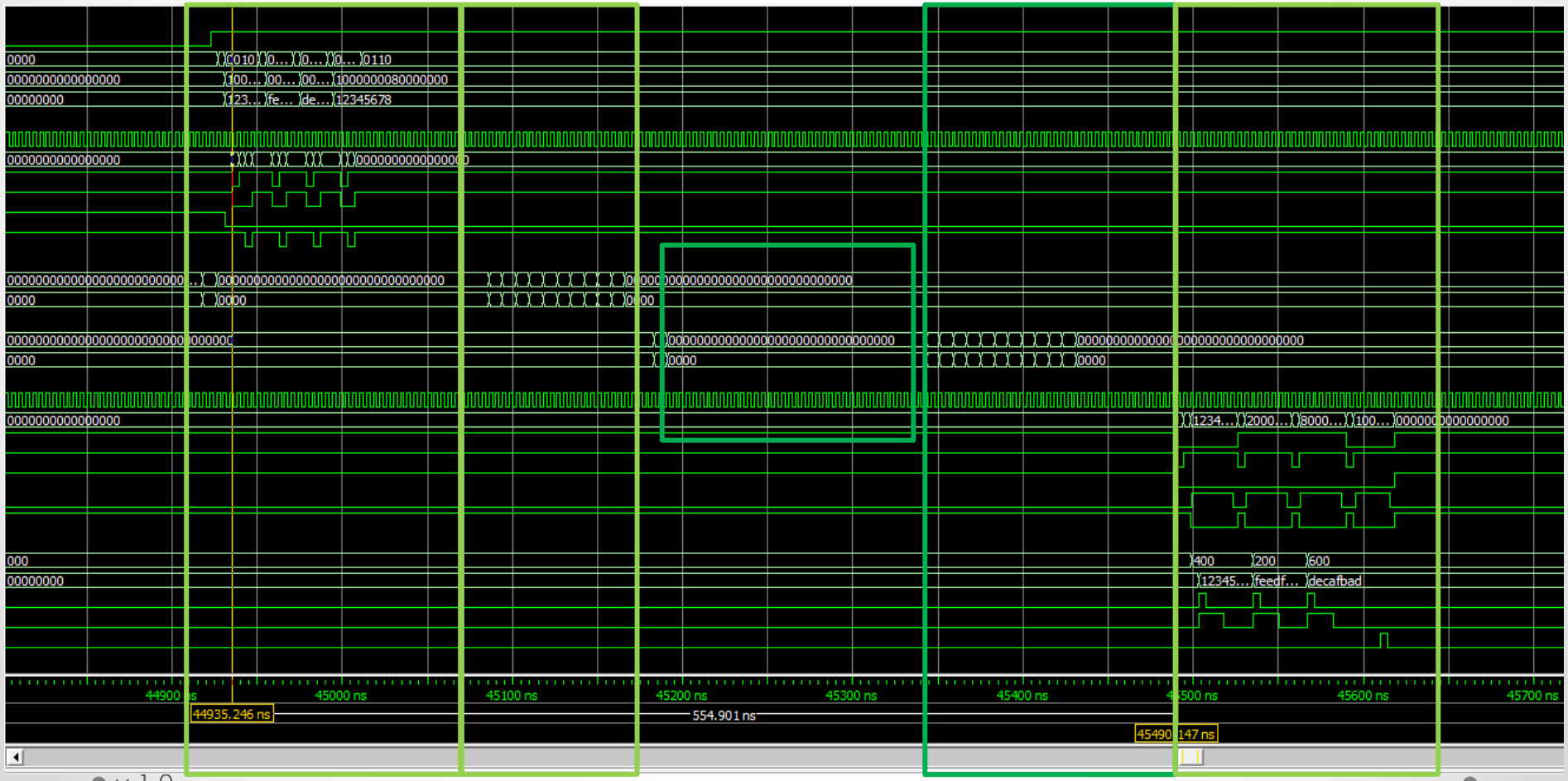





trn_clk
trn_td[63:0]
HDR1 + HDR2
HDR3 + DATA1
DATA2 + DATA3
DATA4 + DATA5
DATA
trn_tsof_n
trn_teof_n
trn_trem_n
trn_tsrc_rdy_n
trn_tdst_rdy_n
trn_tsrc_dsc_n

Xilinx simulation RP <-> EP

- Gen1, x8, Scrambling disabled in CORE Gen

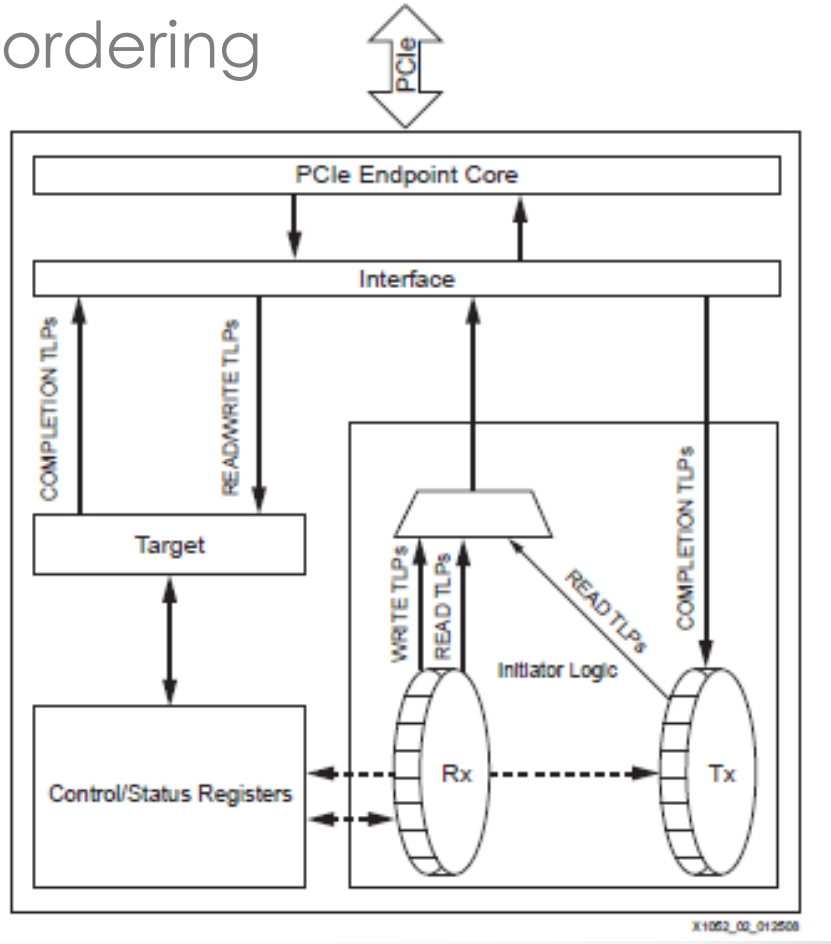


How to design with Xilinx PCIe Hard IP

- Application notes
- Reference designs
- CORE Gen Programmable IO (PIO) hardware/simulation examples

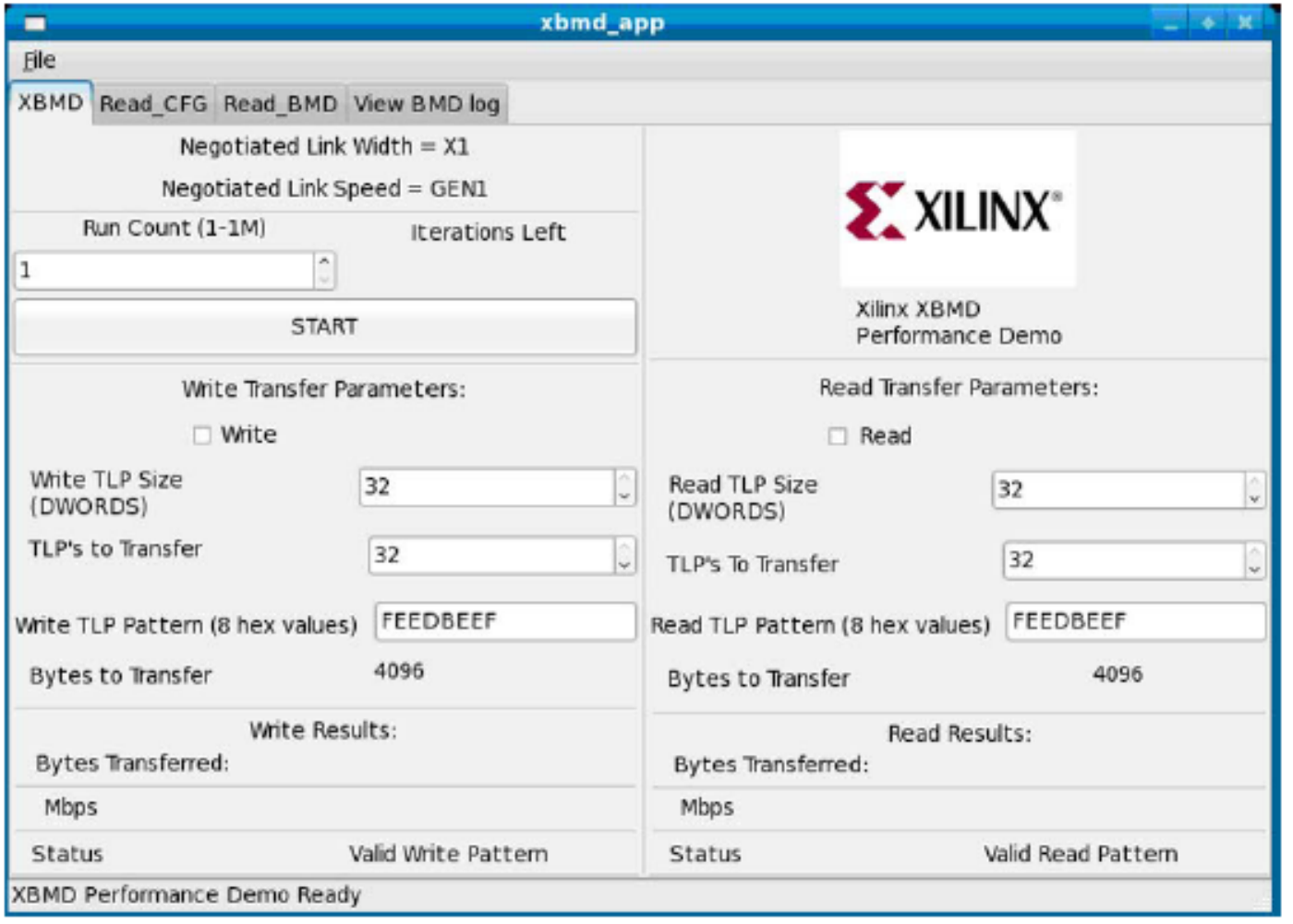
XAPP 1052

- Block DMA in Streaming mode
- No CplD transaction re-ordering

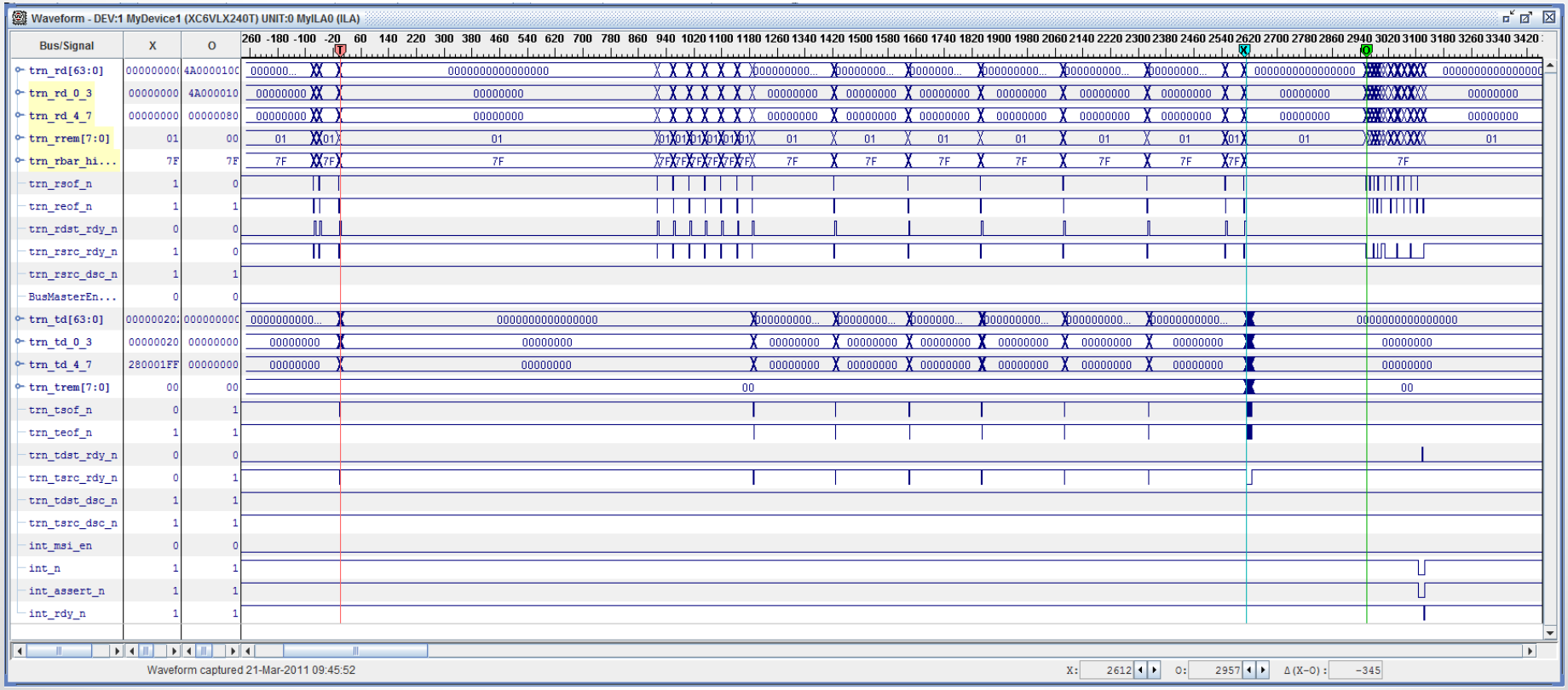


XAPP 1052

- GUI for Win(VisualBasic)
- GUI for Linux (Glade)
- Driver for Win/Linux


xbmd_app
File
XBMD
Read_CFG
Read_BMD
View BMD log
Negotiated Link Width = X1
Negotiated Link Speed = GEN1
Run Count (1-1M)
Iterations Left
1
START
XILINX
Xilinx XBMD
Performance Demo
Write Transfer Parameters:
Write
Write TLP Size (DWORDS)
32
TLP's to Transfer
32
Write TLP Pattern (8 hex values)
FEEDBEEF
Bytes to Transfer
4096
Write Results:
Bytes Transferred:
Mbps
Status
Valid Write Pattern
Read Transfer Parameters:
Read
Read TLP Size (DWORDS)
32
TLP's To Transfer
32
Read TLP Pattern (8 hex values)
FEEDBEEF
Bytes to Transfer
4096
Read Results:
Bytes Transferred:
Mbps
Status
Valid Read Pattern
XBMD Performance Demo Ready

XAPP1052

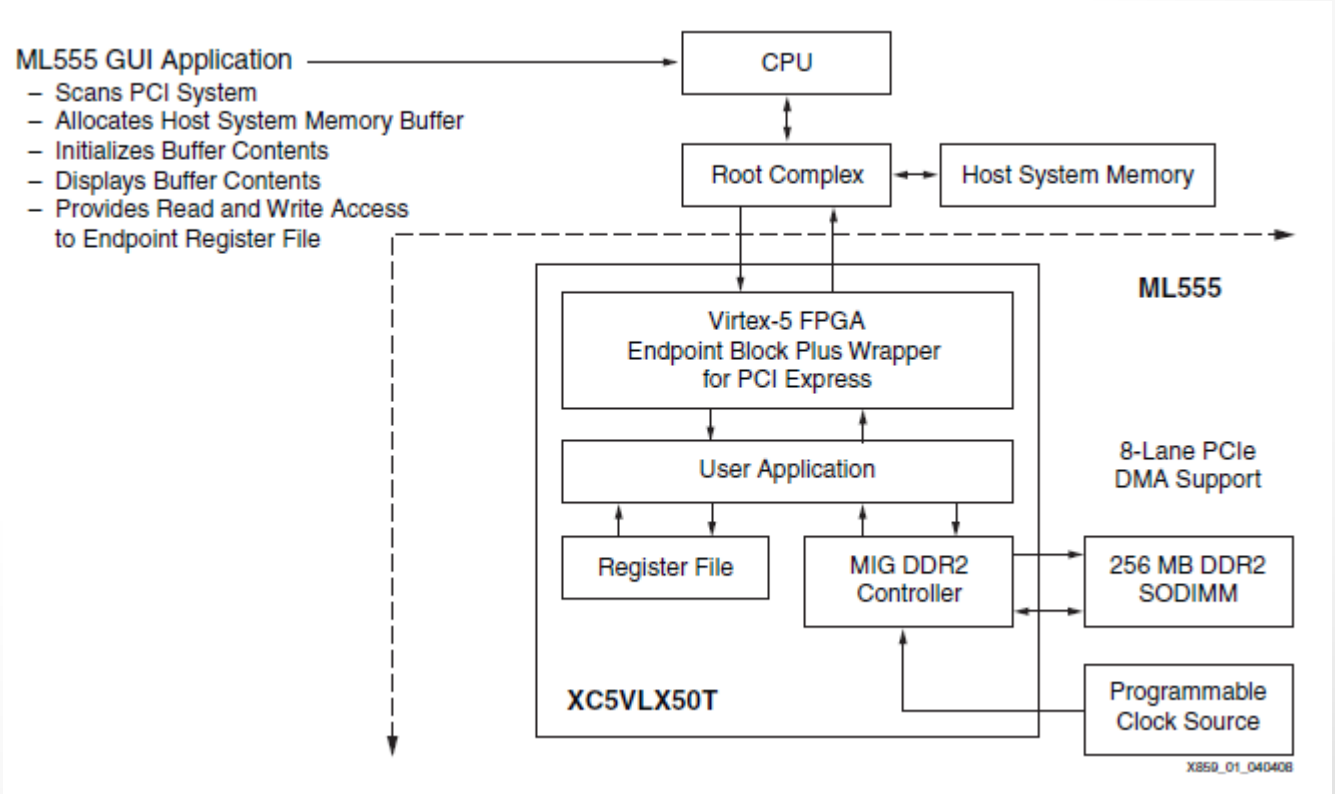

Waveform - DEV:1 MyDevice1 (XC6VLX240T) UNIT:0 MyILA0 (ILA)
Bus/Signal
X
O
trn_rd[63:0]
trn_rd_0_3
trn_rd_4_7
trn_rrem[7:0]
trn_rbar_hi...
trn_rsof_n
trn_reof_n
trn_rdst_rdy_n
trn_rsrc_rdy_n
trn_rsrc_dsc_n
BusMasterEn...
trn_td[63:0]
trn_td_0_3
trn_td_4_7
trn_trem[7:0]
trn_tsof_n
trn_teof_n
trn_tdst_rdy_n
trn_tsrc_rdy_n
trn_tdst_dsc_n
trn_tsrc_dsc_n
int_msi_en
int_n
int_assert_n
int_rdy_n
Waveform captured 21-Mar-2011 09:45:52
X: 2612
O: 2957
Δ(X-O): -345

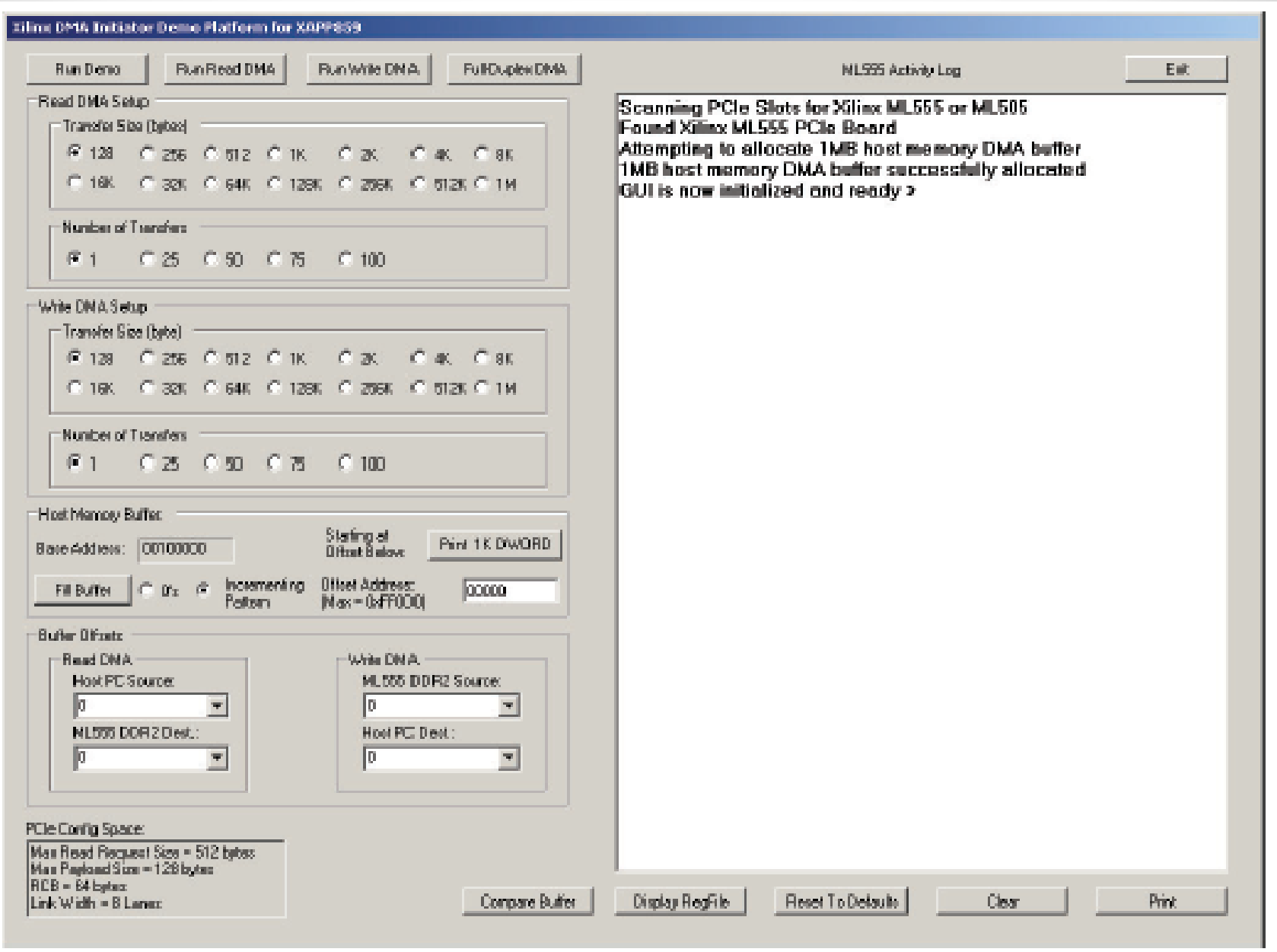
XAPP1052 – performance

- Intel Nehalem 5540 platform
- Fedora 14, 2.35. PAE kernel
- Gen1, x4, PCIe LeCroy analyser
- DMA config
  - Host configures (MWr) DMA engine – around 370 ns between 1DW writes
  - Host checks DMA status: MRd (1DW) to CplD (1DW) response time – around 40 ns
- DMA operation:
  - DMA MRd(1st) -> CplD response time around 2.76 µs
  - DMA MRd(8th) -> CplD response time around 3.82 µs
  - DMA MWr -> around 750-800 MB/s (Gen1,

XAPP 859

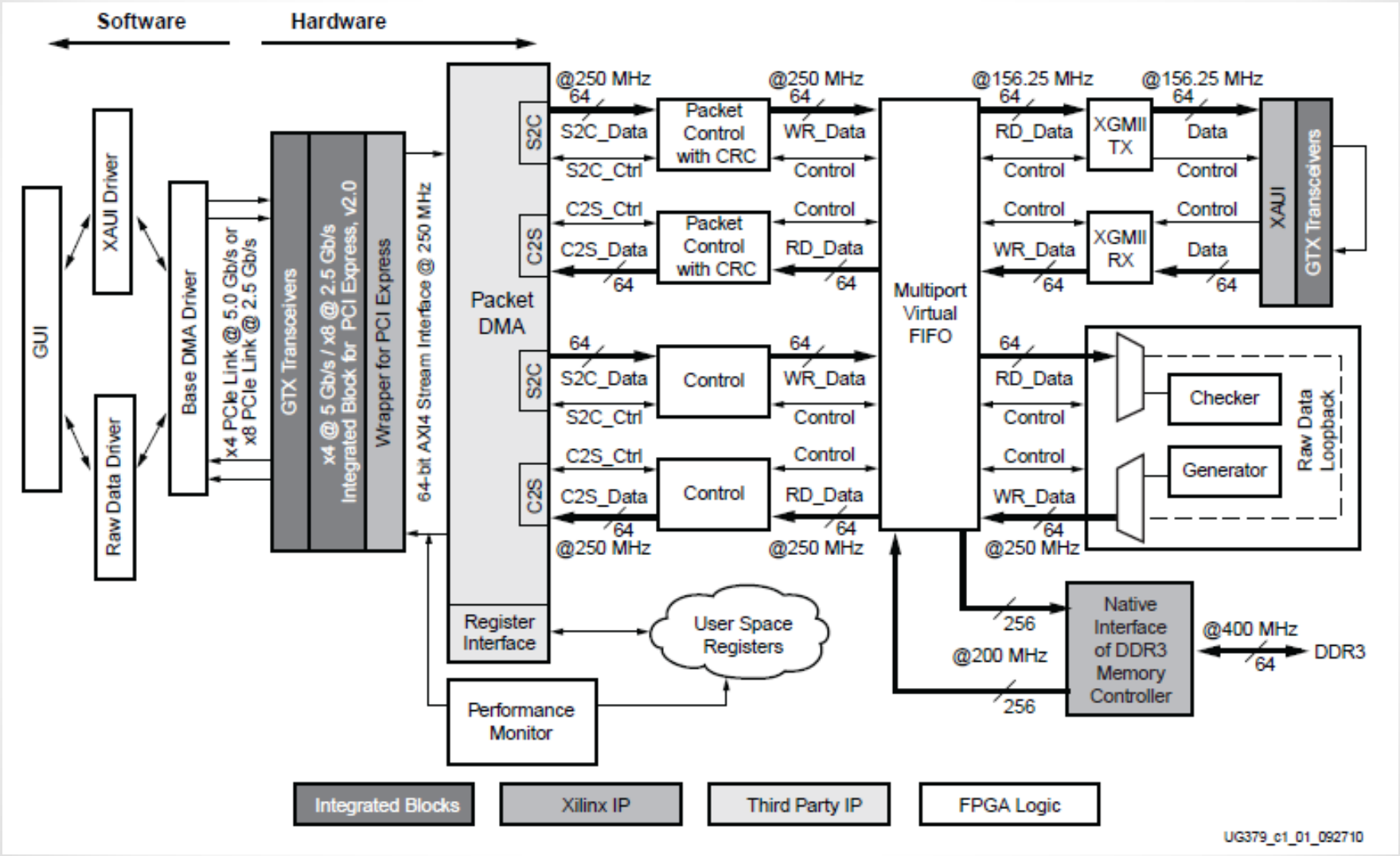
- Block DMA: Host <-> DDR2
- Jungo Win device driver
- C# GUI




Xilinx DMA Initiator Demo Platform for XAPP859
Run Demo
Run Read DMA
Run Write DMA
Full Duplex DMA
ML555 Activity Log
Exit
Read DMA Setup
Transfer Size (bytes)
128 256 512 1K 2K 4K 8K
16K 32K 64K 128K 256K 512K 1M
Number of Transfers
1 25 50 75 100
Write DMA Setup
Transfer Size (bytes)
128 256 512 1K 2K 4K 8K
16K 32K 64K 128K 256K 512K 1M
Number of Transfers
1 25 50 75 100
Host Memory Buffer
Base Address: 00100000
Starting at Offset Below
Print 1K DWORD
Fill Buffer
0's
Incrementing Pattern
Offset Address: (Max = 0xFF000)
00000
Buffer Offsets
Read DMA
Host PC Source:
0
ML555 DDR2 Dest.:
0
Write DMA
ML555 DDR2 Source:
0
Host PC Dest.:
0
PCIe Config Space:
Max Read Request Size = 512 bytes
Max Payload Size = 128 bytes
RCB = 64 bytes
Link Width = 8 Lanes
Scanning PCIe Slots for Xilinx ML555 or ML505
Found Xilinx ML555 PCIe Board
Attempting to allocate 1MB host memory DMA buffer
1MB host memory DMA buffer successfully allocated
GUI is now initialized and ready >
Compare Buffer
Display RegFile
Reset To Defaults
Clear
Print
X859_31_062608

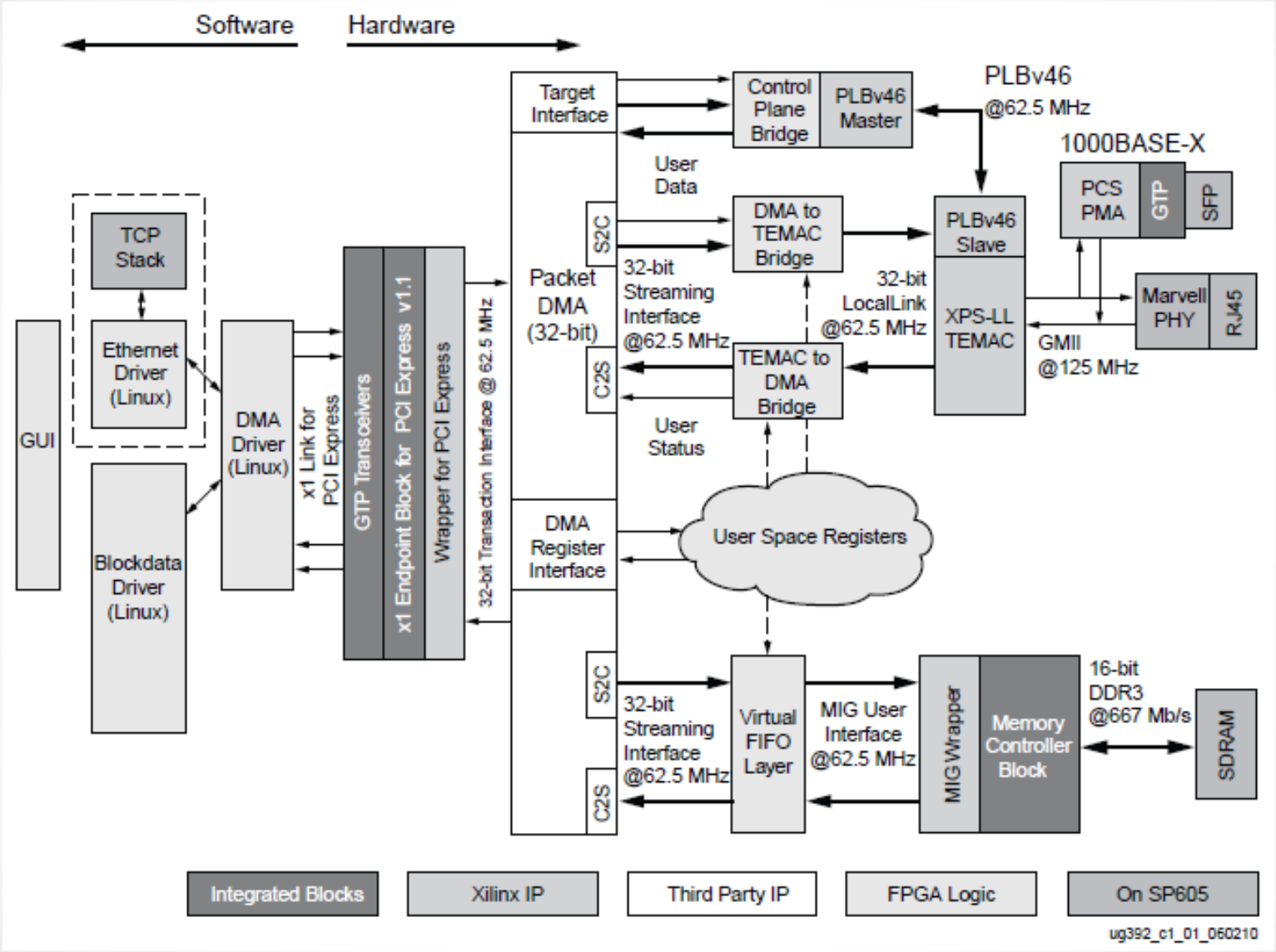
Xilinx V6 Connectivity Kit

- PCIe to XAUI
- PCIe to parallel loopback
- VirtualFIFO based on DDR3 (MIG, SODIMM)
- Northwest Logic User Backend IP – Packet (SG) DMA


Software
Hardware
GUI
XAUI Driver
Raw Data Driver
Base DMA Driver
x4 PCIe Link @ 5.0 Gb/s or x8 PCIe Link @ 2.5 Gb/s
GTX Transceivers
x4 @ 5 Gb/s / x8 @ 2.5 Gb/s Integrated Block for PCI Express, v2.0
Wrapper for PCI Express
64-bit AXI4 Stream Interface @ 250 MHz
Packet DMA
S2C
C2S
S2C
C2S
@250 MHz
64
S2C_Data
S2C_Ctrl
C2S_Ctrl
C2S_Data
Packet Control with CRC
Packet Control with CRC
Control
Control
WR_Data
RD_Data
Control
Multiport Virtual FIFO
@156.25 MHz
RD_Data
Control
WR_Data
XGMII TX
XGMII RX
Data
XAUI
GTX Transceivers
Checker
Generator
Raw Data Loopback
Register Interface
User Space Registers
Performance Monitor
256
@200 MHz
Native Interface of DDR3 Memory Controller
@400 MHz
DDR3
Integrated Blocks
Xilinx IP
Third Party IP
FPGA Logic
UG379_c1_01_092710

Xilinx S6 Connectivity Kit

- PCIe to 1 Gb Eth
- PCIe to parallel loopback
- VirtualFIFO based on DDR3 (MIG, Component)
- Northwest Logic User Backend – Packet (SG) DMA


Software
Hardware
Target Interface
Control Plane Bridge
PLBv46 Master
PLBv46
@62.5 MHz
1000BASE-X
PCS PMA
GTP
SFP
User Data
S2C
DMA to TEMAC Bridge
PLBv46 Slave
32-bit Streaming Interface @62.5 MHz
Packet DMA (32-bit)
32-bit LocalLink @62.5 MHz
XPS-LL TEMAC
Marvell PHY
RJ45
GMII @125 MHz
C2S
TEMAC to DMA Bridge
User Status
TCP Stack
Ethernet Driver (Linux)
GUI
Blockdata Driver (Linux)
DMA Driver (Linux)
x1 Link for PCI Express
GTP Transceivers
x1 Endpoint Block for PCI Express v1.1
Wrapper for PCI Express
32-bit Transaction Interface @ 62.5 MHz
DMA Register Interface
User Space Registers
S2C
32-bit Streaming Interface @62.5 MHz
Virtual FIFO Layer
MIG User Interface @62.5 MHz
MIG Wrapper
Memory Controller Block
16-bit DDR3 @667 Mb/s
SDRAM
C2S
Integrated Blocks
Xilinx IP
Third Party IP
FPGA Logic
On SP605
ug392_c1_01_060210

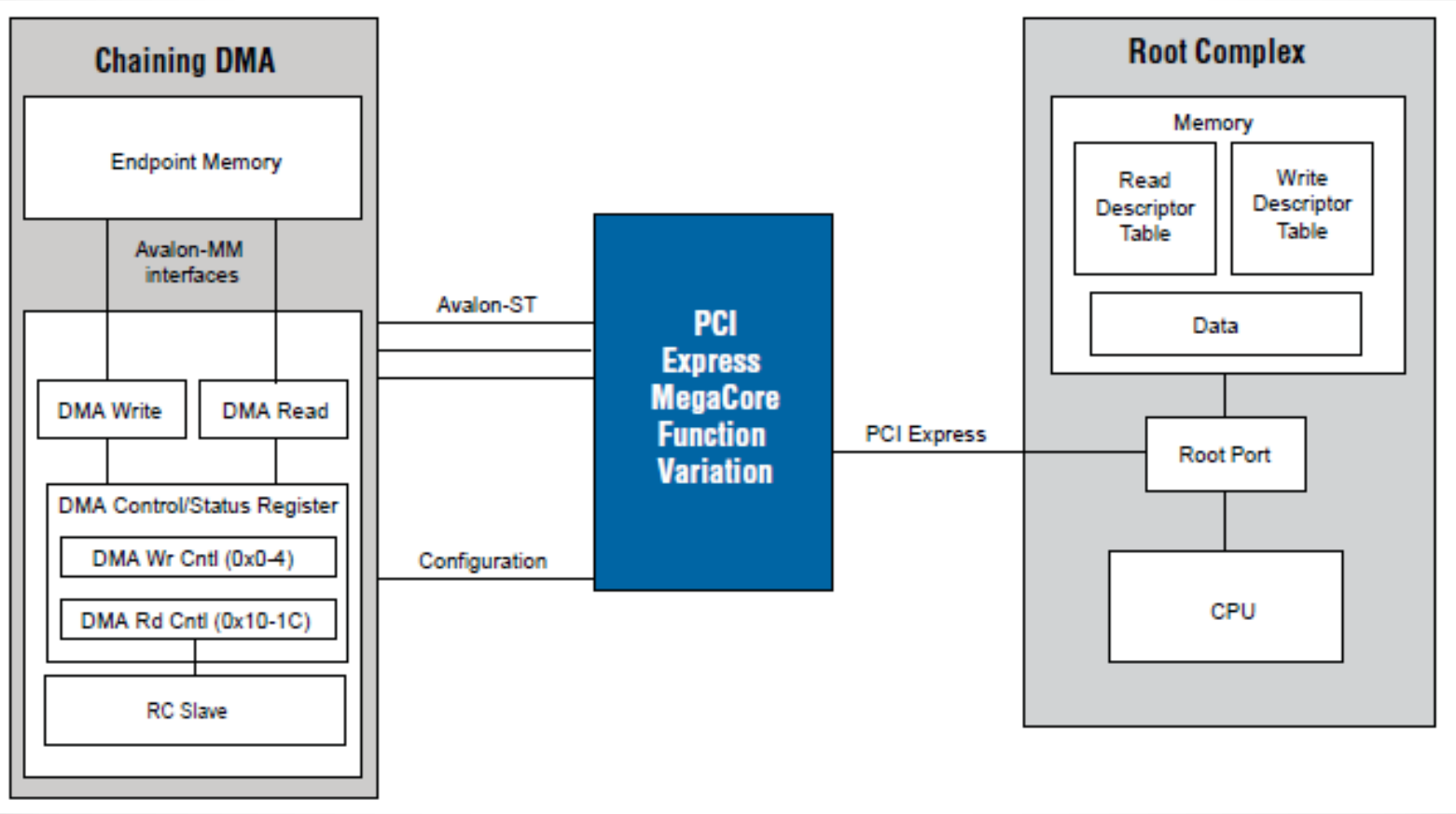
Altera Hard IP solution

- Target devices:
  - Cyclone IV GX
  - Arria I/II GX
  - Stratix II/IV GX
- Similar to Xilinx in terms of user interface – TLP over Avalon ST or User application with Avalon MM
  - ST – streaming mode, for high performance designs
  - MM – memory mapped, for SOPC builder, lower performance
- CvPCIe – FPGA reconfiguration over PCIe
  - I/O and PCIe programmed faster than the rest of the core

Altera Megacore Reference Designs

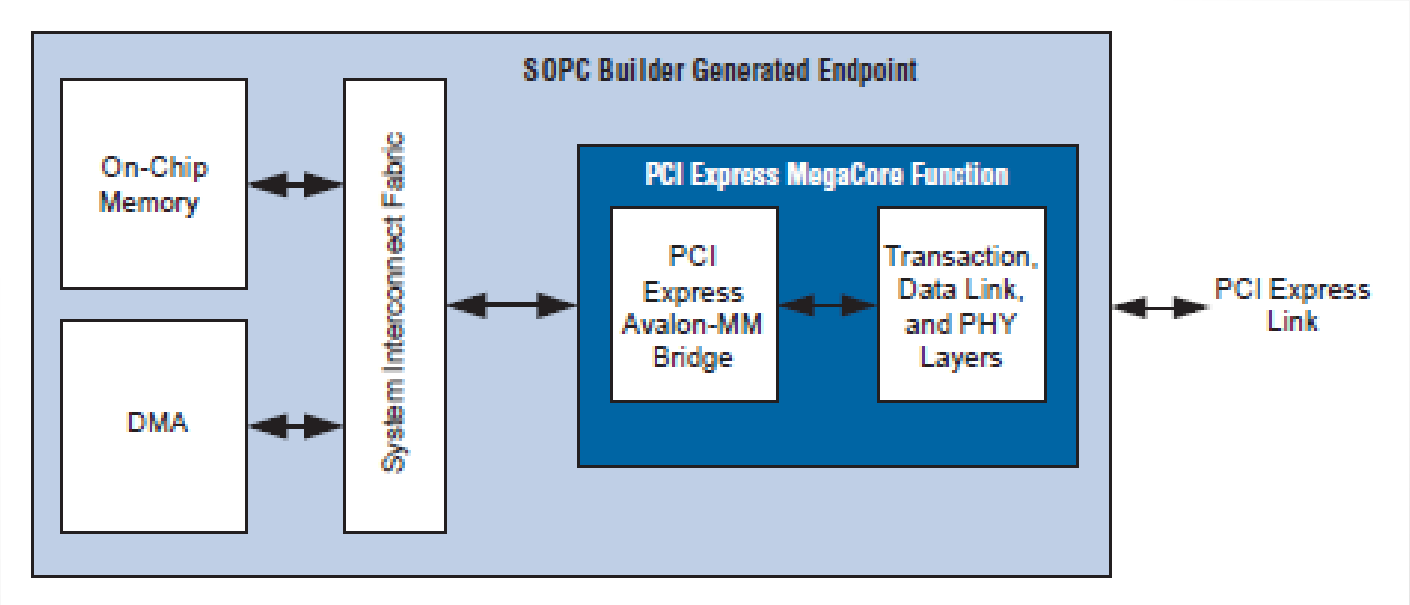
- Endpoint Reference Design
  - PCIe High Performance Reference Design (AN456) – Chained DMA, uses internal RAM, binary win driver
  - PCIe to External Memory Reference Design (AN431) – Chained DMA, uses DDR2/DDR3, binary win driver
- Root Port Reference Design
- SOPC PIO

- Chained DMA documentation
  - also Linux device driver available
- BFM documentation
  - Extensive simulation with Bus Functional Models

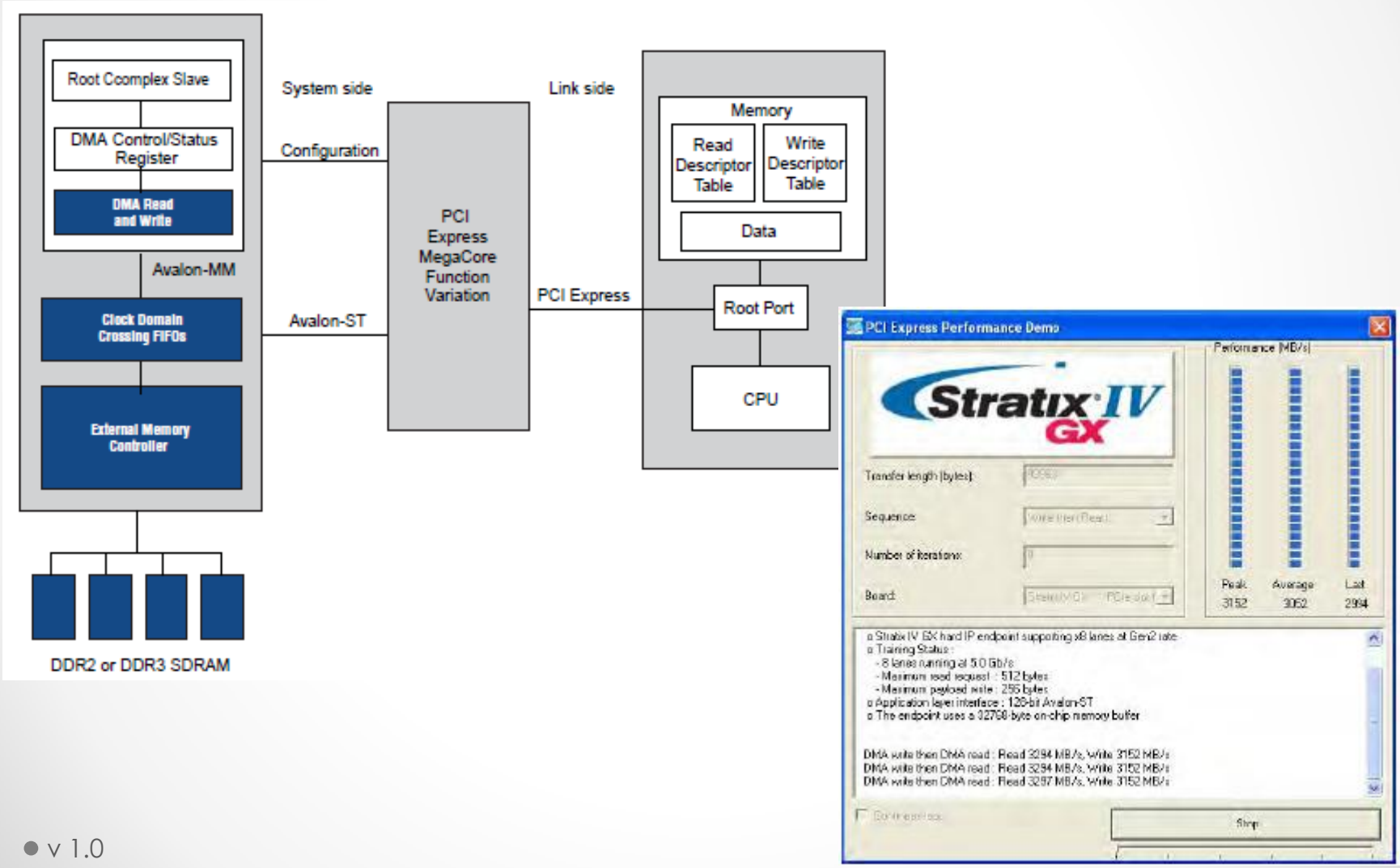

Chaining DMA
Endpoint Memory
Avalon-MM interfaces
DMA Write
DMA Read
DMA Control/Status Register
DMA Wr Cntl (0x0-4)
DMA Rd Cntl (0x10-1C)
RC Slave
Avalon-ST
Configuration
PCI Express MegaCore Function Variation
PCI Express
Root Complex
Memory
Read Descriptor Table
Write Descriptor Table
Data
Root Port
CPU

SOPC Based Design

- SOPC Builder Based
- Gen 1, x4
- DMA
- Sim and HW



AN431 – PCIe to DDR3



PLDA PCIe IPs

- XpressLite
  - currently available at CERN
  - Soft IP, Gen1 Endpoint only, x1/x2/x4
  - Stratix GX, Stratix II GX, and Arria GX support
  - No S4GX, C4GX and A2GX Hard IP support
- EZDMA2 Altera/Xilinx
  - Support Hard IP inside Altera: Cyclone IV GX, Arria II GX, and Stratix IV GX
  - Hard IP inside Xilinx: Virtex-5/6, Spartan-6
  - Same user/DMA interface as XpressLite
- XpressRich – rich version
  - Are you rich ?
- Northwest Logic ?

PLDA XpressLite

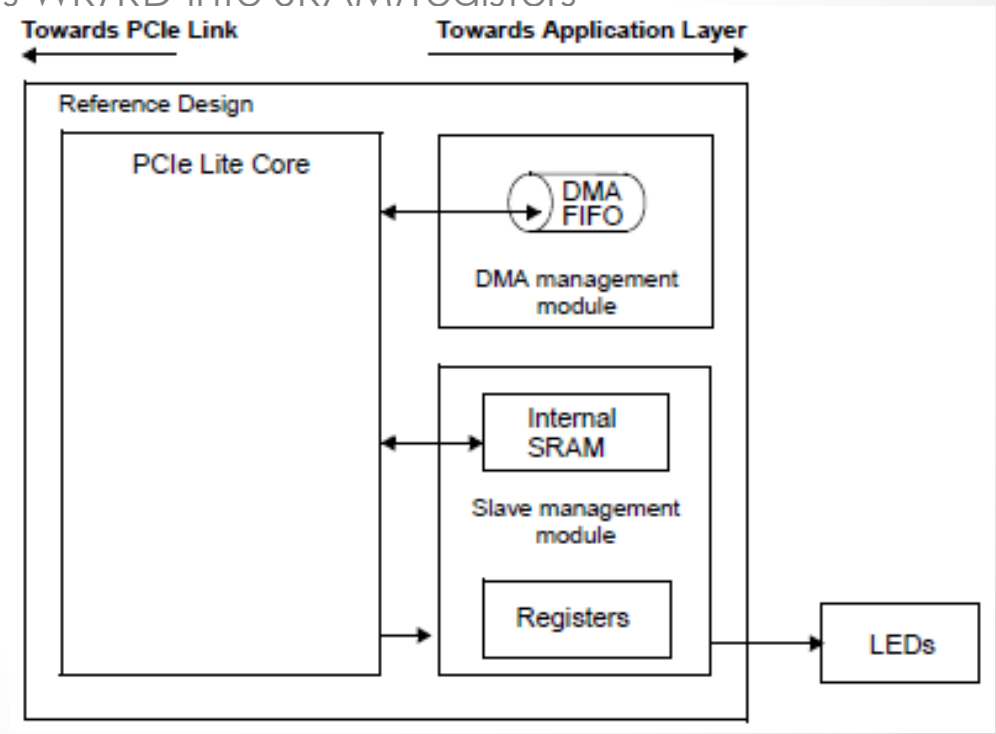
- Stratix GX, Stratix II GX, and Arria GX support only
  - No S4GX, C4GX and A2GX Hard IP support

- Generated with JAVA GUI: Windows/Linux
- Synthesis: single VHDL/Verilog encrypted file
- ModelSim: pre-compiled lib (Win/Linux)
- Ncsim: protected lib (Linux)
- Testbench: RP emulation

- Device drivers, API, tools (C++ source available)

PLDA XpressLite

- Maximum 8 DMA channels with Scatter Gather
- Reference design:
  - PCIe Lite – Endpoint only
  - Single DMA engine – C2S(WR) + S2C(RD)
  - Single target module – accepts WR/RD into SRAM/registers

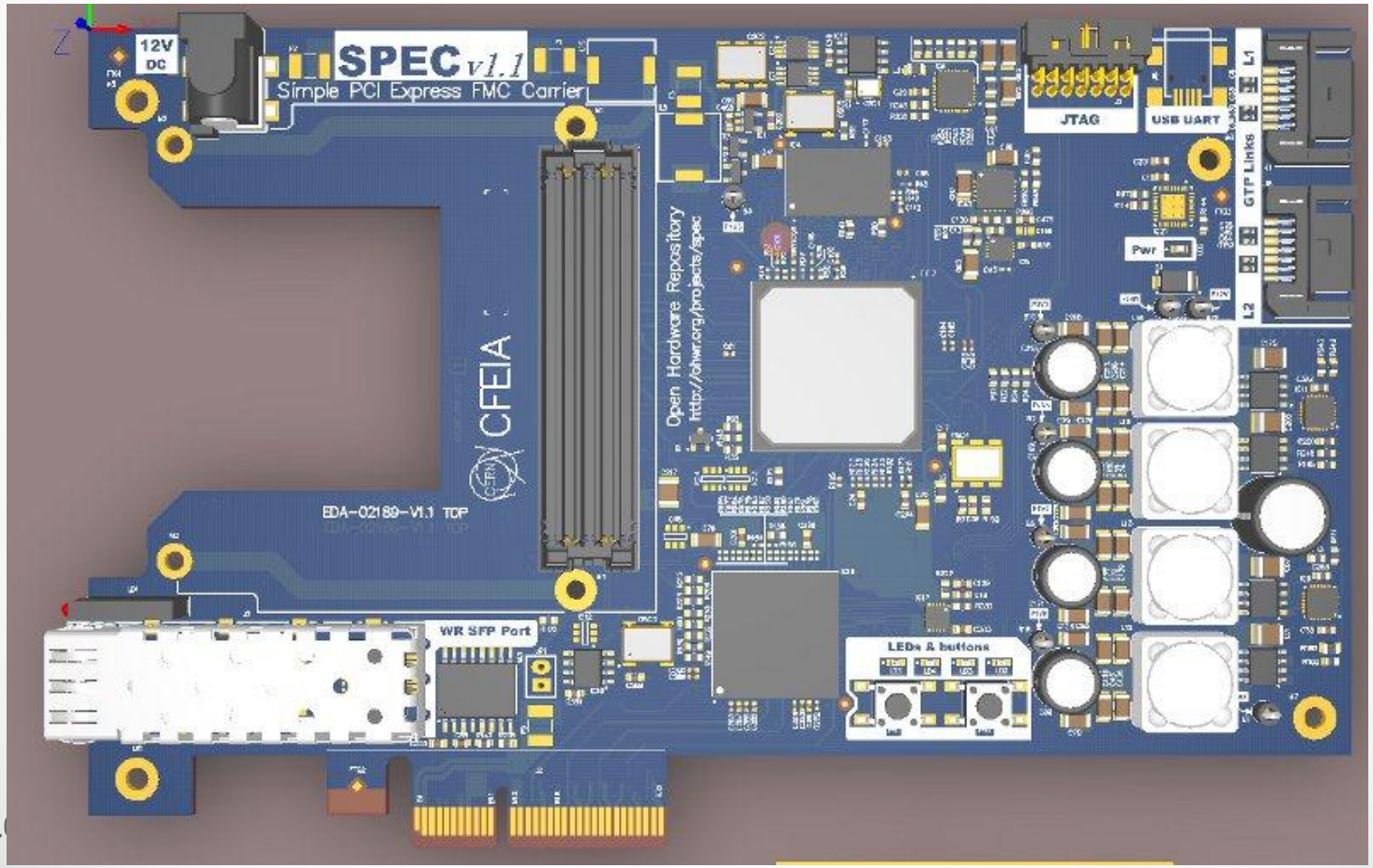


External PCIe chips - Gennum

- TLP interface with simple framing signalling
- FPGA serial programming
  - FPGA can be reprogramed without affecting PCIe link
- GPIO interface/Interrupts
- IP (with DMA) provided for Altera and Xilinx
- Device drivers and Software DK provided
- Already used at CERN:
  - Open source IP for Xilinx device developed by CERN group
  - Wishbone
  - SG DMA
  - device driver
  - More info www.ohwr.org

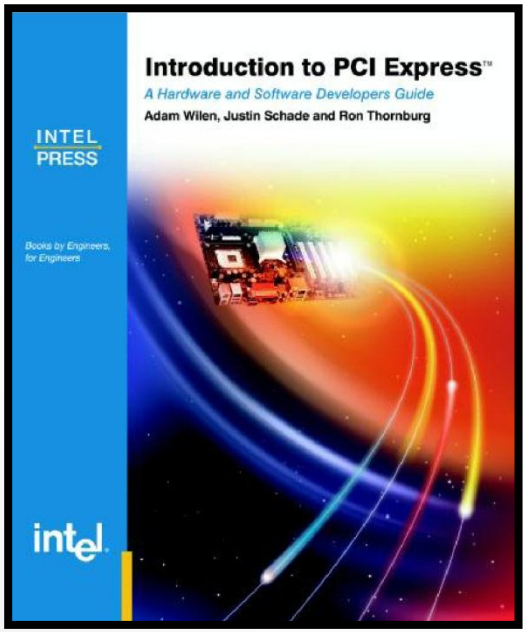
Gennum PHY + Spartan6

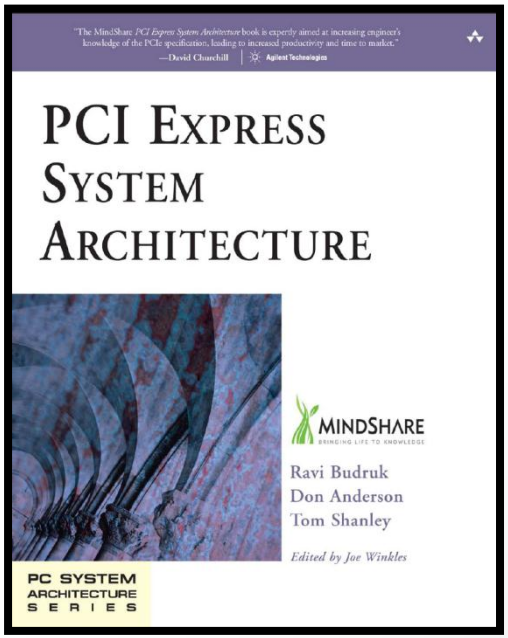
- http://www.ohwr.org/projects/spec/wiki
- Open source IP, SG DMA, device driver



More information

- Books:
  - Introduction to PCI Express – CERN Library (hardcopy)
  - PCI Express standards – CERN Library – CDS.CERN.CH
  - PCI Express System Architecture – mindshare.com (ebook+ hardcopy)





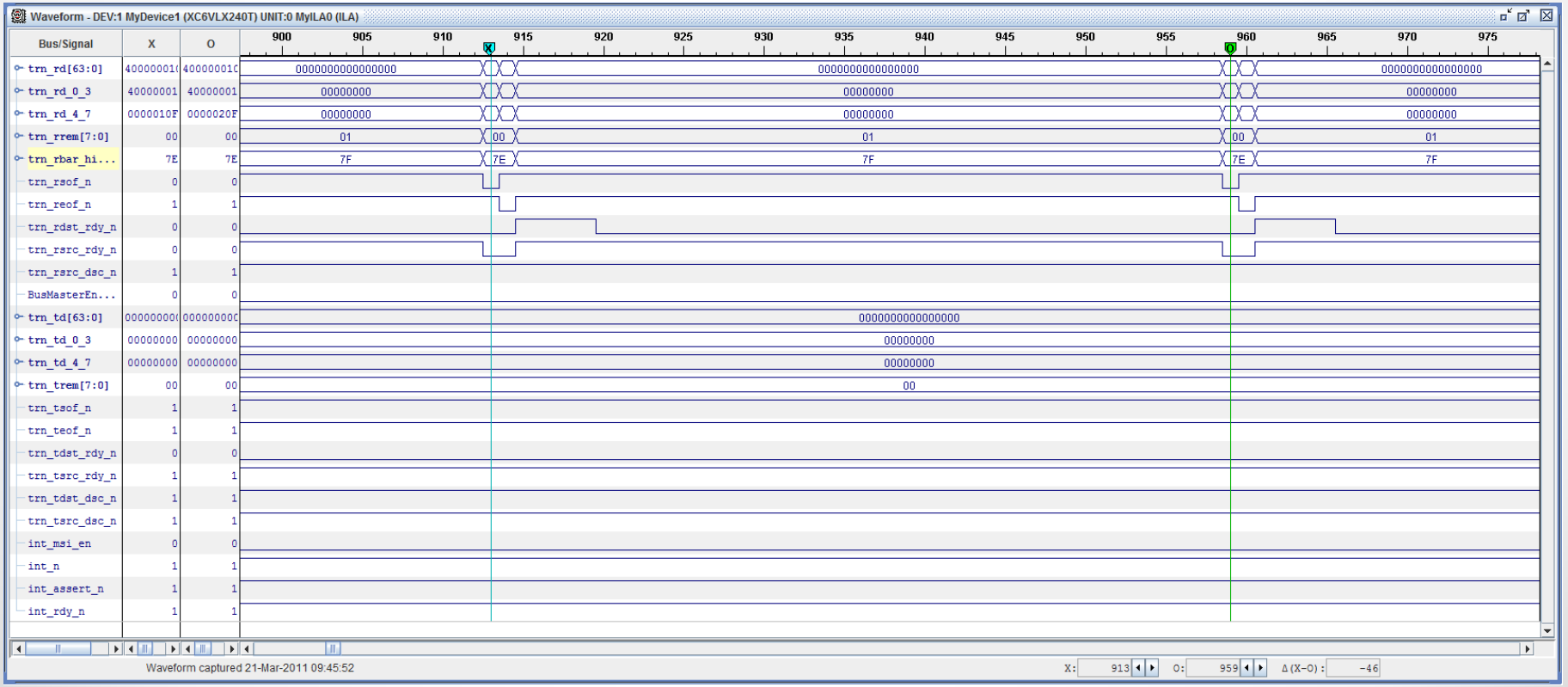
eda.support@cern.ch

- PCIe demos available on request
- IDT PCIe Switch dev. kit. coming soon
- Evaluating EZDMA2 for Xilinx.

Extras

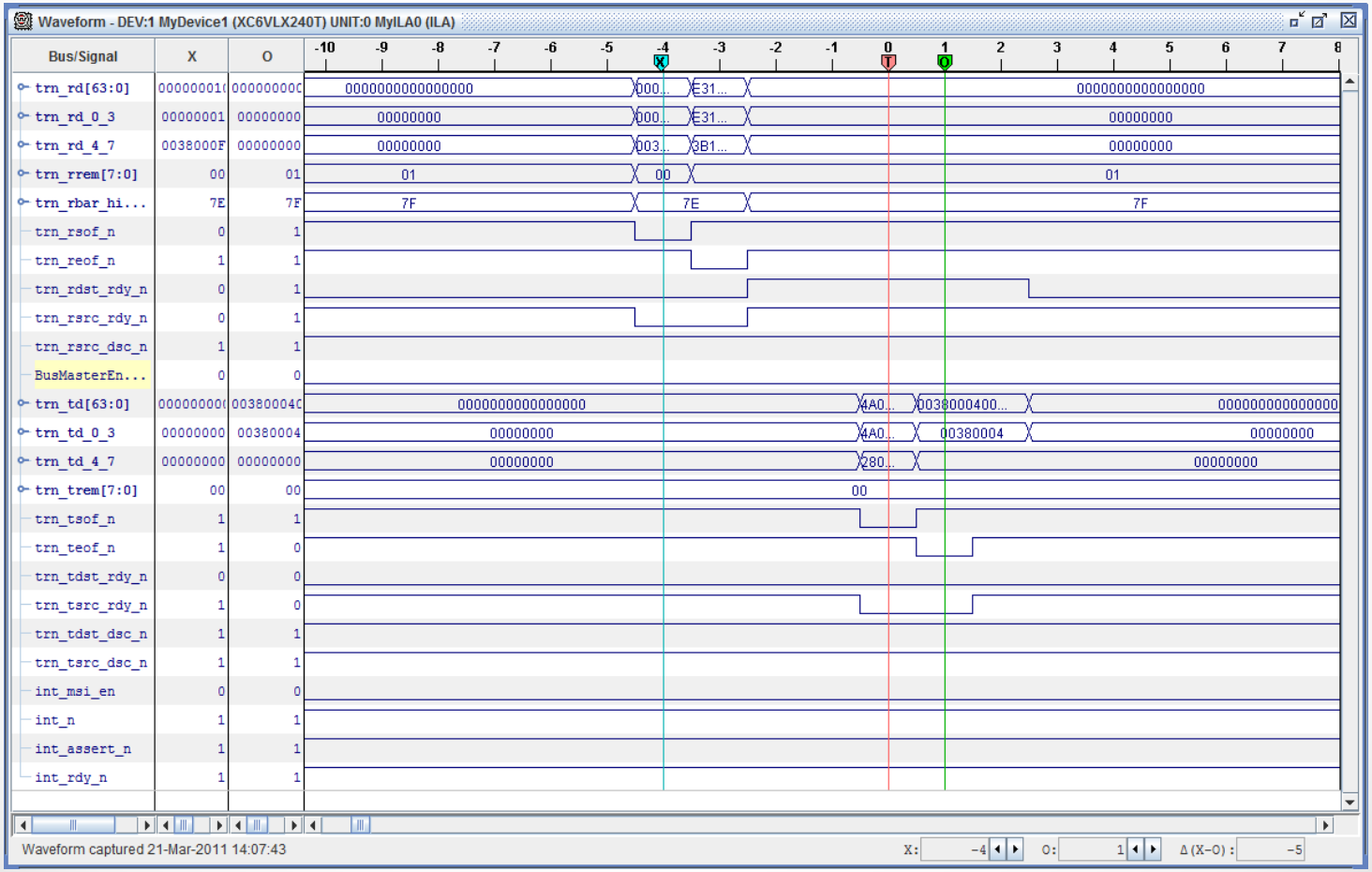
XAPP1052 DMA Config WR

- Host configures (MWr) DMA engine – around 370 ns between 1DW writes



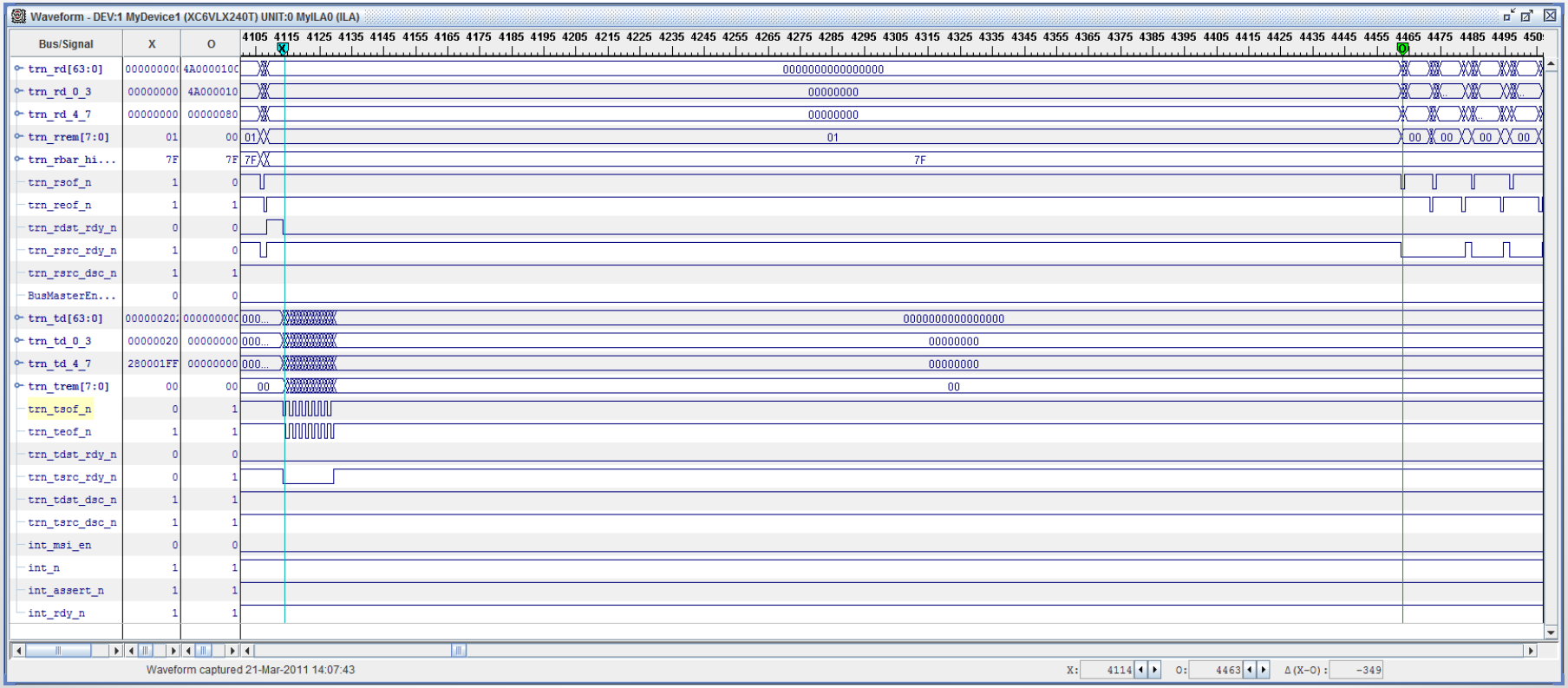
XAPP 1052 DMA Config RD

- MRd (1DW) to CplD (1DW) – around 40 ns



MRd to System Memory

- Intel Nehalem 5540 platform
- MRd(1st) -> CplD response time around 2.76 µs
- MRd(8th) -> CplD response time around 3.82 µs



Link Tra			TLP		MWr(32)	Length	RequesterID	Tag	Address	1st BE	Last BE	Data	VC ID	Explicit ACK		# Packets	Time Delta	Time Stamp
0	R→	2.5 x4	2083	Mem	10:00000	1	000:00:0	3	E3100000	1111	0000	1 dword	0	Packet #2639	Metrics	2	104.000 ns	- 0000 . 000 000 104 s
1	R→	2.5 x4	2084	Mem	10:00000	1	000:00:0	0	E3100000	1111	0000	1 dword	0	Packet #2641	Metrics	2	448.000 ns	0000 . 000 000 000 s

Split Tra				MRd(32)	RequesterID	CompleterID	Tag	TC	VC ID	Address	Status	Data		# LinkTras	Time Delta	Time Stamp
0	R→	2.5 x4	Mem	00:00000	000:07:0	040:00:0	0	0	0	E3100004	SC	1 dword	Metrics	2	7.336 µs	0000 . 000 000 448 s

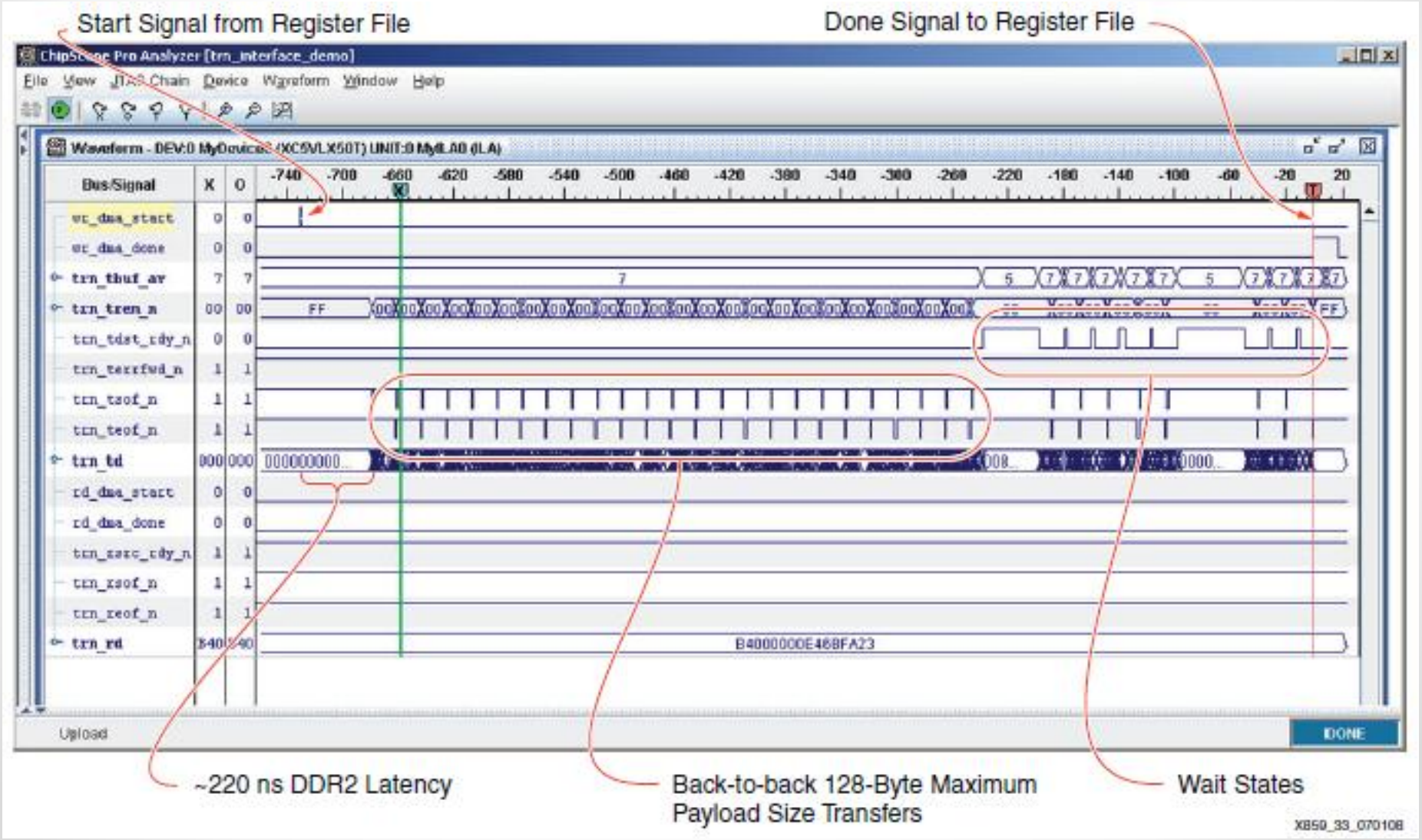
Link Tra			TLP		MWr(32)	Length	RequesterID	Tag	Address	1st BE	Last BE	Data	VC ID	Explicit ACK		# Packets	Time Delta	Time Stamp
4	R→	2.5 x4	2086	Mem	10:00000	1	000:00:0	1	E3100010	1111	0000	1 dword	0	Packet #2655	Metrics	2	368.000 ns	0000 . 000 007 784 s
5	R→	2.5 x4	2087	Mem	10:00000	1	000:00:0	2	E310000C	1111	0000	1 dword	0	Packet #2658	Metrics	2	400.000 ns	0000 . 000 008 152 s
6	R→	2.5 x4	2088	Mem	10:00000	1	000:00:0	3	E3100024	1111	0000	1 dword	0	Packet #2661	Metrics	2	368.000 ns	0000 . 000 008 552 s
7	R→	2.5 x4	2089	Mem	10:00000	1	000:00:0	0	E3100020	1111	0000	1 dword	0	Packet #2665	Metrics	2	336.000 ns	0000 . 000 008 920 s
8	R→	2.5 x4	2090	Mem	10:00000	1	000:00:0	1	E3100014	1111	0000	1 dword	0	Packet #2668	Metrics	2	368.000 ns	0000 . 000 009 256 s
9	R→	2.5 x4	2091	Mem	10:00000	1	000:00:0	2	E3100018	1111	0000	1 dword	0	Packet #2671	Metrics	2	368.000 ns	0000 . 000 009 624 s

Split Tra				MRd(32)	RequesterID	CompleterID	Tag	TC	VC ID	Address	Status	Data		# LinkTras	Time Delta	Time Stamp
1	R→	2.5 x4	Mem	00:00000	000:07:0	040:00:0	0	0	0	E3100010	SC	1 dword	Metrics	2	1.912 µs	0000 . 000 009 992 s
2	R→	2.5 x4	Mem	00:00000	000:07:0	040:00:0	0	0	0	E310000C	SC	1 dword	Metrics	2	1.696 µs	0000 . 000 011 904 s
3	R→	2.5 x4	Mem	00:00000	000:07:0	040:00:0	0	0	0	E3100024	SC	1 dword	Metrics	2	1.704 µs	0000 . 000 013 600 s
4	R→	2.5 x4	Mem	00:00000	000:07:0	040:00:0	0	0	0	E3100020	SC	1 dword	Metrics	2	1.904 µs	0000 . 000 015 304 s
5	R→	2.5 x4	Mem	00:00000	000:07:0	040:00:0	0	0	0	E3100014	SC	1 dword	Metrics	2	1.936 µs	0000 . 000 017 208 s
6	R→	2.5 x4	Mem	00:00000	000:07:0	040:00:0	0	0	0	E3100018	SC	1 dword	Metrics	2	1.800 µs	0000 . 000 019 144 s

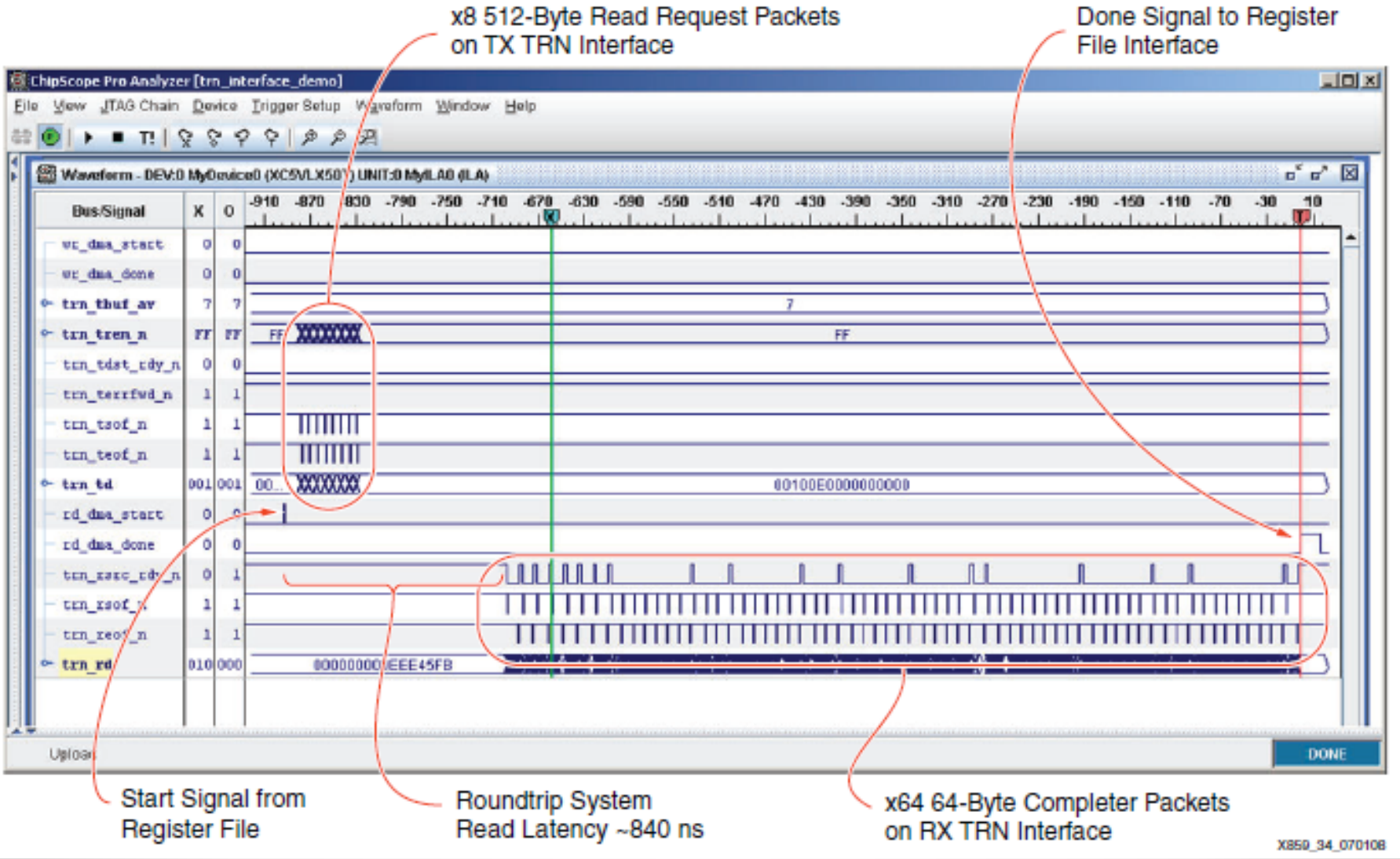
Link Tra			TLP		MWr(32)	Length	RequesterID	Tag	Address	1st BE	Last BE	Data	VC ID	Explicit ACK		# Packets	Time Delta	Time Stamp
22	R→	2.5 x4	2098	Mem	10:00000	1	000:00:0	3	E3100048	1111	0000	1 dword	0	Packet #2712	Metrics	2	400.000 ns	0000 . 000 020 944 s
23	R→	2.5 x4	2099	Mem	10:00000	1	000:00:0	0	E3100004	1111	0000	1 dword	0	Packet #2714	Metrics	2	868.000 ns	0000 . 000 021 344 s

Split Tra				MRd(32)	RequesterID	CompleterID	Tag	TC	VC ID	Address	Status	Data		# LinkTras	Time Delta	Time Stamp
7	R←	2.5 x4	Mem	00:00000	040:00:0	000:00:0	1	0	0	FFC00000	SC	32 dwords	Metrics	3	36.000 ns	0000 . 000 022 212 s
8	R←	2.5 x4	Mem	00:00000	040:00:0	000:00:0	2	0	0	FFC00080	SC	32 dwords	Metrics	3	44.000 ns	0000 . 000 022 248 s
9	R←	2.5 x4	Mem	00:00000	040:00:0	000:00:0	3	0	0	FFC00100	SC	32 dwords	Metrics	2	36.000 ns	0000 . 000 022 292 s
10	R←	2.5 x4	Mem	00:00000	040:00:0	000:00:0	4	0	0	FFC00180	SC	32 dwords	Metrics	2	44.000 ns	0000 . 000 022 328 s
11	R←	2.5 x4	Mem	00:00000	040:00:0	000:00:0	5	0	0	FFC00200	SC	32 dwords	Metrics	2	36.000 ns	0000 . 000 022 372 s
12	R←	2.5 x4	Mem	00:00000	040:00:0	000:00:0	6	0	0	FFC00280	SC	32 dwords	Metrics	2	44.000 ns	0000 . 000 022 408 s
13	R←	2.5 x4	Mem	00:00000	040:00:0	000:00:0	7	0	0	FFC00300	SC	32 dwords	Metrics	2	36.000 ns	0000 . 000 022 452 s

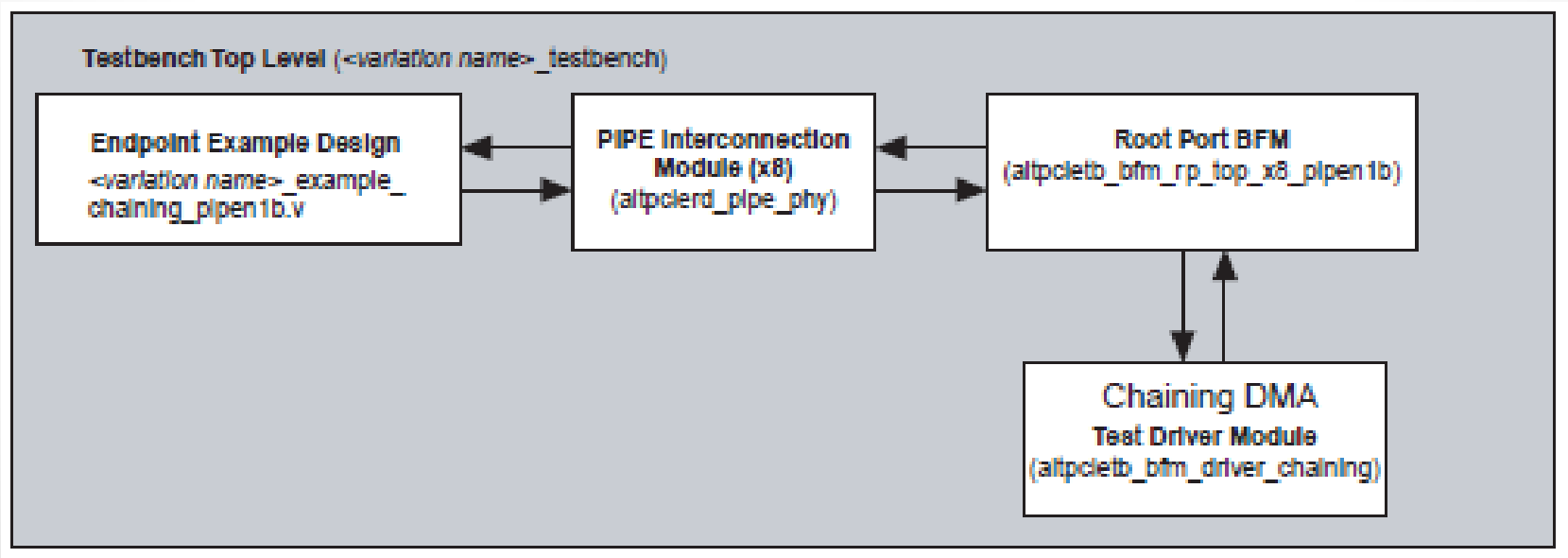
XAPP 859 – Write



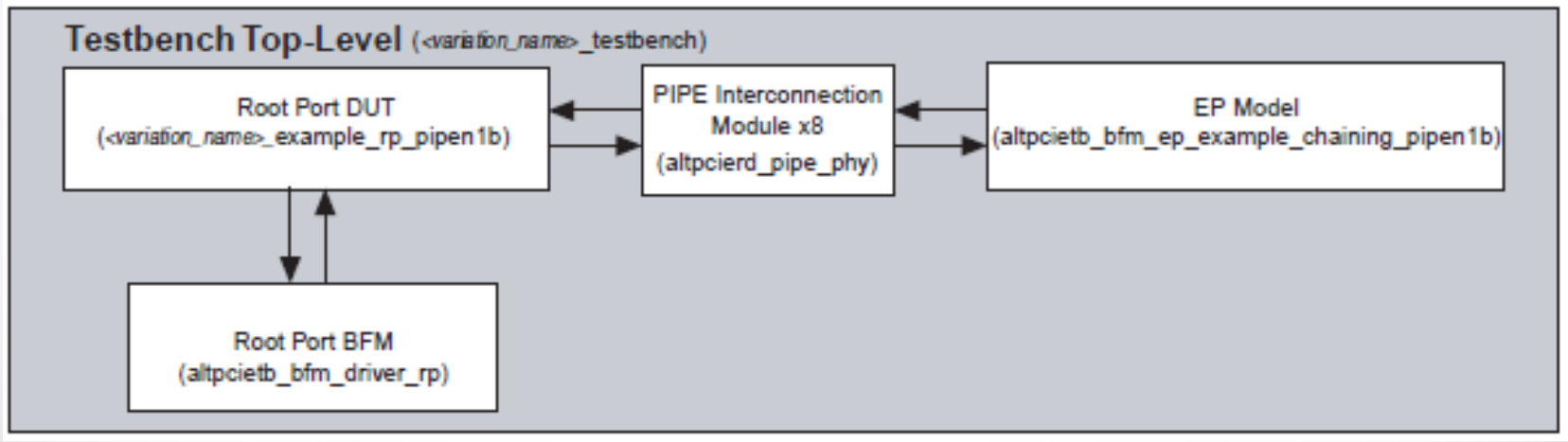
XAPP 859 – Read



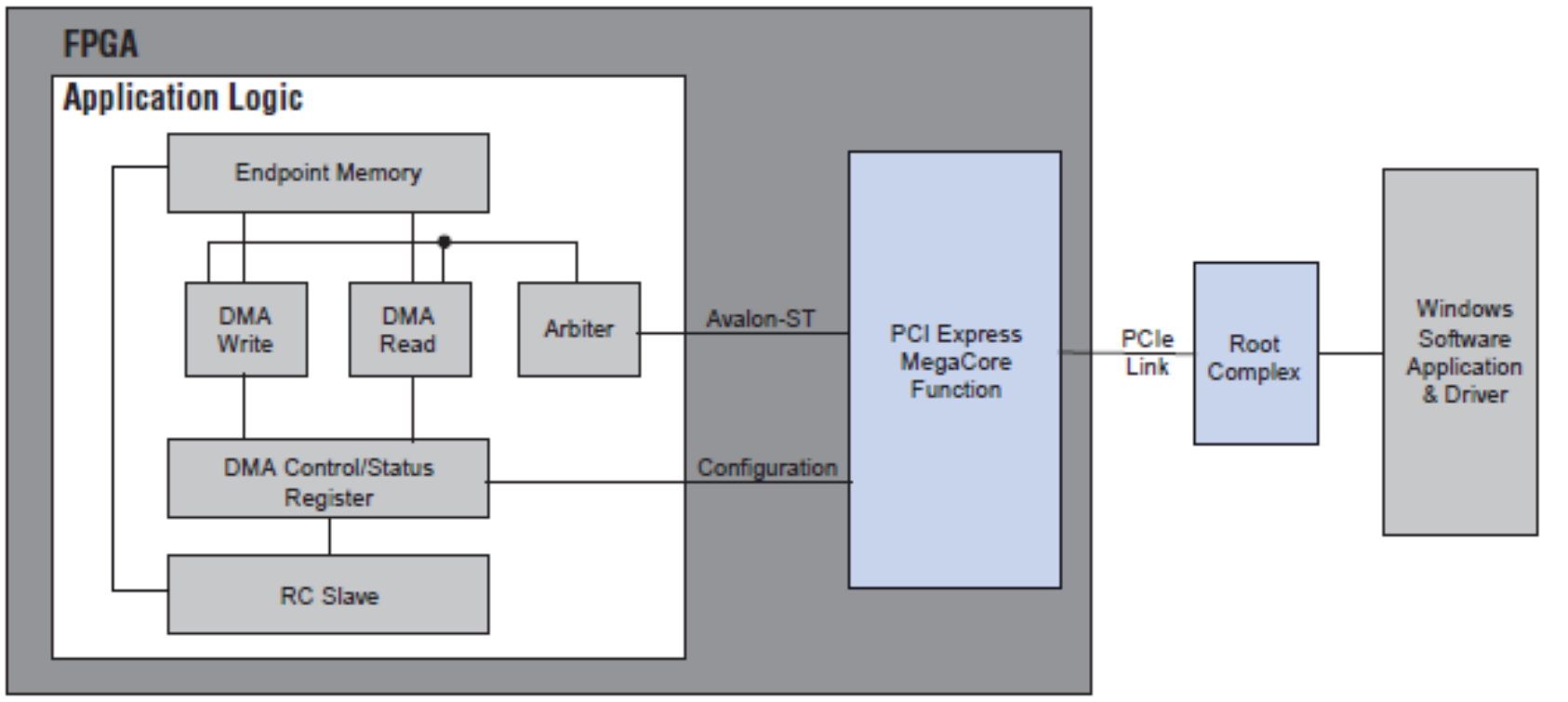
Endpoint TB

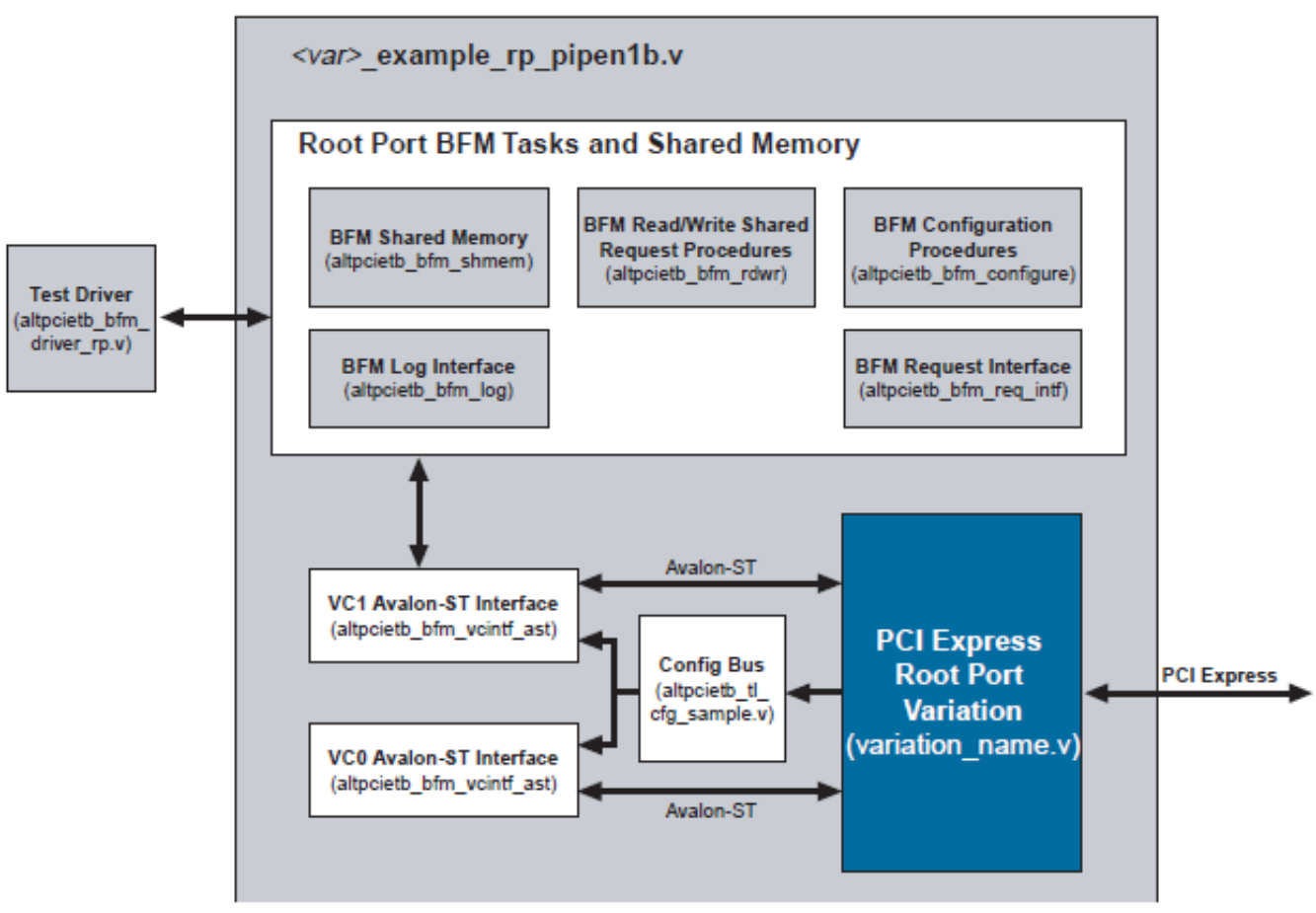

Testbench Top Level (<variation name>_testbench)
Endpoint Example Design
<variation name>_example_
chaining_pipen1b.v
PIPE Interconnection
Module (x8)
(altpcierd_pipe_phy)
Root Port BFM
(altpcietb_bfm_rp_top_x8_pipen1b)
Chaining DMA
Test Driver Module
(altpcietb_bfm_driver_chaining)

Root Port TB


Testbench Top-Level (<variation_name>_testbench)
Root Port DUT
(<variation_name>_example_rp_pipen1b)
PIPE Interconnection
Module x8
(altpcierd_pipe_phy)
EP Model
(altpcietb_bfm_ep_example_chaining_pipen1b)
Root Port BFM
(altpcietb_bfm_driver_rp)

AN456 – Chained DMA




<var>_example_rp_pipen1b.v
Root Port BFM Tasks and Shared Memory
BFM Shared Memory (altpcietb_bfm_shmem)
BFM Read/Write Shared Request Procedures (altpcietb_bfm_rdwr)
BFM Configuration Procedures (altpcietb_bfm_configure)
BFM Log Interface (altpcietb_bfm_log)
BFM Request Interface (altpcietb_bfm_req_intf)
Test Driver (altpcietb_bfm_driver_rp.v)
VC1 Avalon-ST Interface (altpcietb_bfm_vcintf_ast)
VC0 Avalon-ST Interface (altpcietb_bfm_vcintf_ast)
Avalon-ST
Avalon-ST
Config Bus (altpcietb_tl_cfg_sample.v)
PCI Express Root Port Variation (variation_name.v)
PCI Express

Endianness

- 0x12345678
- Big-Endian stores the MSB at the lowest memory address. Little-Endian stores the LSB at the lowest memory address. The lowest memory address of multi-byte data is considered the starting address of the data. In Figure 1, the 32-bit hex value 0x12345678 is stored in memory as follows for each Endian-architecture. The lowest memory address is represented in the leftmost position, Byte 00.
- http://en.wikipedia.org/wiki/Endianness

Endian Order	Byte 00	Byte 01	Byte 02	Byte 03
Big Endian	12	34	56	78 (LSB)
Little Endian	78 (LSB)	56	34	12