Flash Memory Summit

Device Architectures to Best Enable PCIe Gen 4

Rob Sykes

Senior Distinguished Engineer

Toshiba America Electronic Components, Inc.

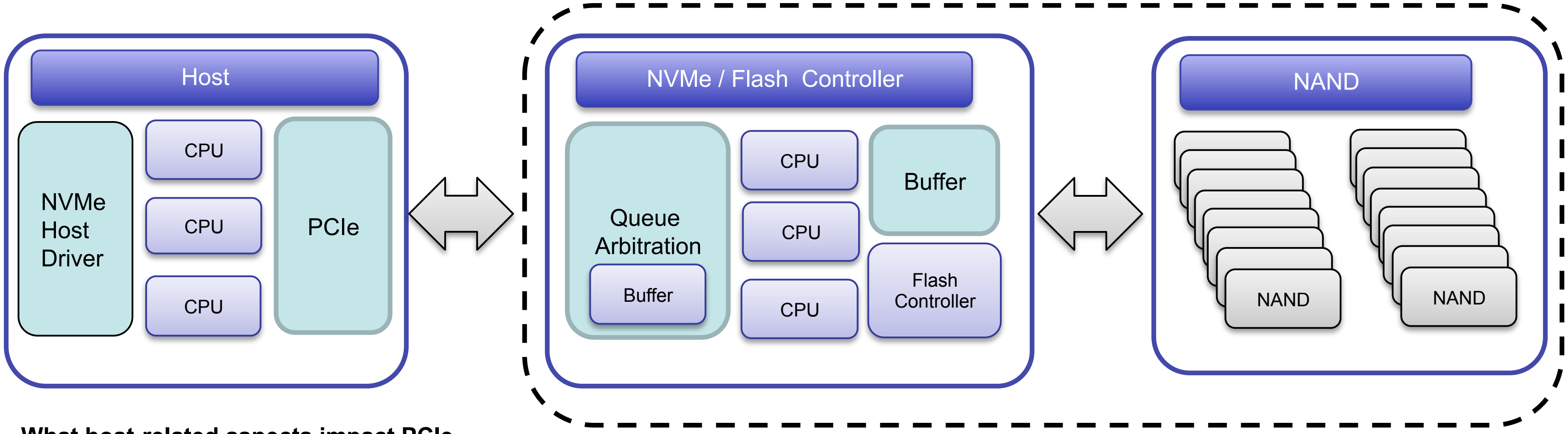
Introduction

- A review of the performance relationship between PCIe Gen 4, NAND (number of planes, channels), controller capabilities and bus architecture and how this relates to top end performance for your systems

- Specifically, the presentation will consider the bottlenecks in the system that relate to achieving the front end performance
  - Bottlenecks exist in the host, the controller architecture in terms of processing power, flash controller, DRAM etc. and the NAND capabilities.

- Power, Performance, Price!

System Level Overview



What host-related aspects impact PCIe Gen 4 performance?

- Driver impacts on performance
- Command size
- Queue depth
- CPU cores to queue

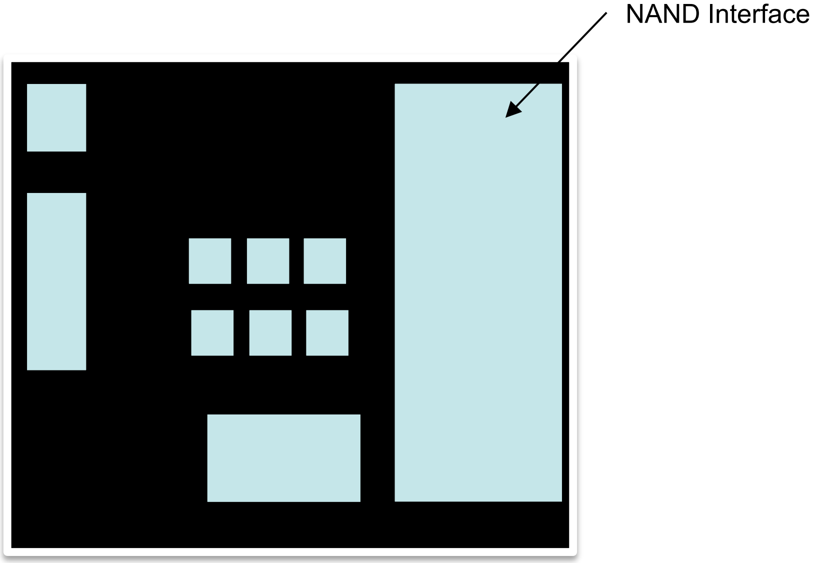
What SSD controller-related aspects impact PCIe Gen 4 performance, including flash controller-related aspects ?

What NAND-related aspects impact PCIe Gen 4 performance?

NAND

NAND impact on SoC:

- In SoC real-estate terms, the NAND controller is one of the most costly items
- SSD companies may limit NAND channels for cost-sensitive markets but will they hit the performance
- Many channels requires many pads.



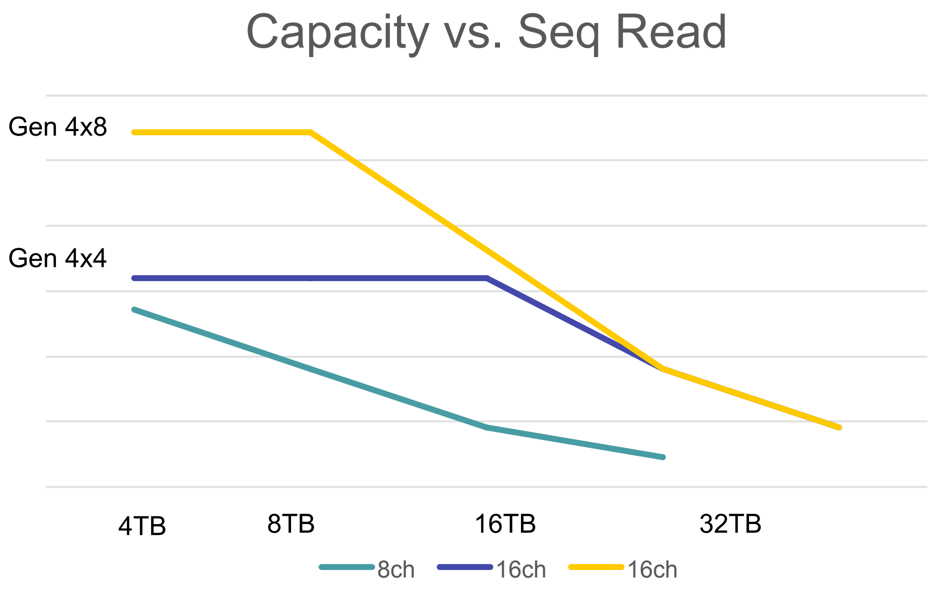
8ch Controller

NAND Limitations

- Performance depends on channel count, NAND frequency, flash die/channel count
- The bus speed and number of channels of the controller can cause bottlenecks
- Higher capacity drives may have a lower performance threshold due to bus loading
- Flash connectivity is more likely to be a bottleneck in high capacity drives due to bus loading

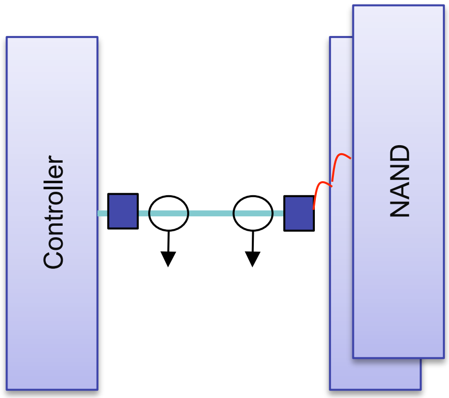
NAND Capacity vs Read speed

- As more die are added, channel loading increases and effectively reduces the performance
- Consideration has to be given to the max intended capacity required
- NAND speed (MT/s) impacts performance
- More channels can be added to match font end bandwidth but costs more.



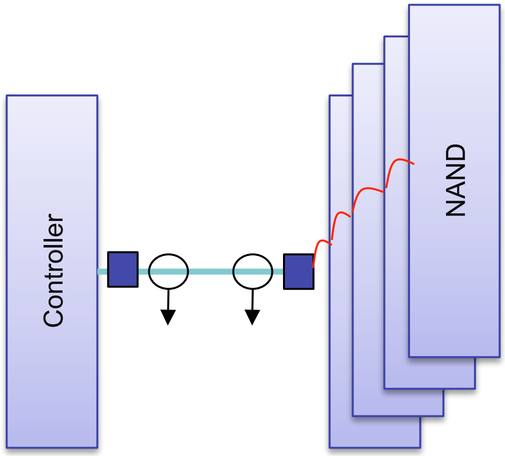
Based on 512Gb NAND with increasing die stacks to reach capacity points

NAND 2 Die per Channel



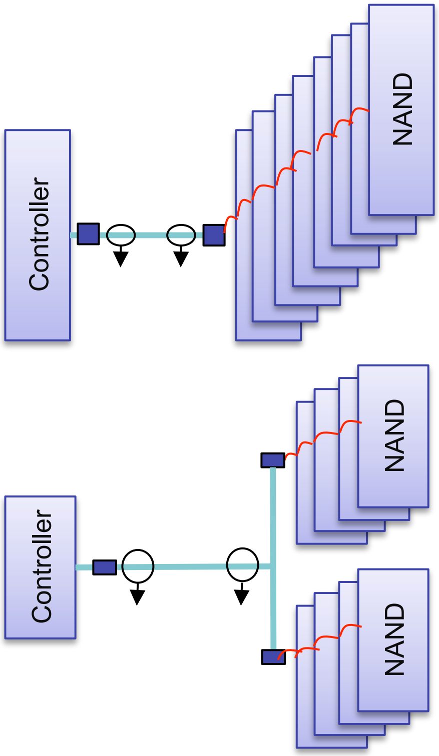
- Each NAND channel has a single package load with two die per bus inside the NAND package
- Assumes PCB with only one package per channel so no stubs
- Performance can be achieved at fast NAND speeds but capacity is limited

NAND 4 Die per Channel



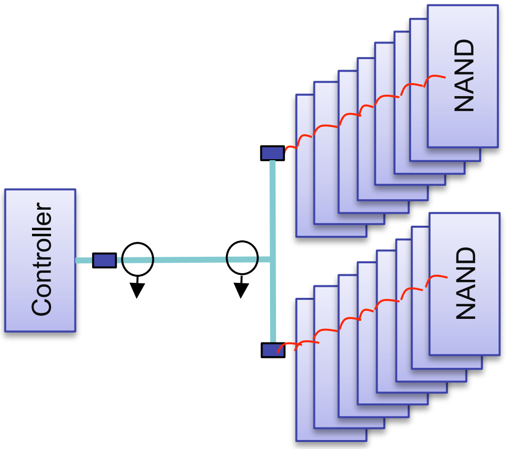
- Each NAND Channel has a single package load with four die per bus inside the NAND package
- Assumes PCB with only one package per channel so no stubs
- Performance can be achieved at fast NAND speeds with some risks, but top performance maybe achieved at lower NAND speeds
- Capacities are increased

NAND 8 Die per Channel



- Each NAND channel has a single package load with eight die per bus inside the NAND package
- Assumes PCB with only one package per channel so no stubs
- Multi-package solution is also possible but results in stubs and may require ODT solution resulting in increased PCB costs
- Fastest NAND may not work in this scenario; highest PCIe performance may not be achieved
- Capacities are increased

NAND 16 Die per Channel



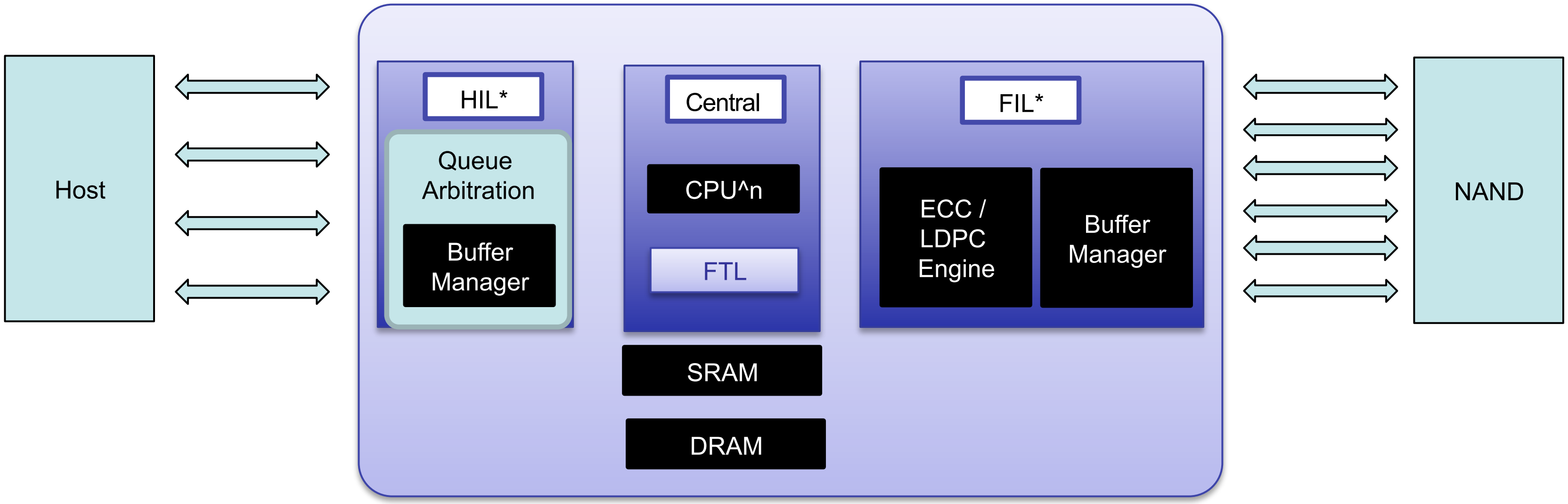
- The NAND channel has two package loads with eight die per bus inside each side of the NAND package
- Assumes PCB with only one package per channel so limited stubs
- Self-terminating ODT scheme may work if the stubs are short, but a matrix ODT may be advised
- Bus speed would be low, PCIe Gen 4x4 speeds may not be met
- Capacities are increased

Controller

Controller & firmware architecture implications on performance:

- CPU architecture
- Buffering techniques
- FTL size (4K, 8k, 16K FTL - Capacity)
- DRAM
- Controller latency
- Power limits

Controller



Generic controller architecture

*Host / Flash Interface Layer

Controllers for PCIe Gen 4

What needs to be considered for a controller to achieve PCIe Gen 4 line rates?

- It's a big step up from SATA
  - 600MB/s(*) for SATA compared to 8000MB/s(*) PCIe Gen4x4
  - More than 13X performance and that just for a 4 lane system
- Where do you put, and how does one handle all that data through the controller to the NAND?
- SATA controllers may have had a straightforward buffering mechanism and a single CPU or a few CPUs (depending on the application)
- PCIe Gen 4 will need either speciality front end hardware assistance or/and multiple CPUs to handle that data rate whilst keeping the latency low
  - The aim is to process as much data as possible within a single clock cycle; we may need to consider achieving a reasonable level of 9s latency!

(*) Theoretical max performance

Flash Translation Layer (FTL) Review

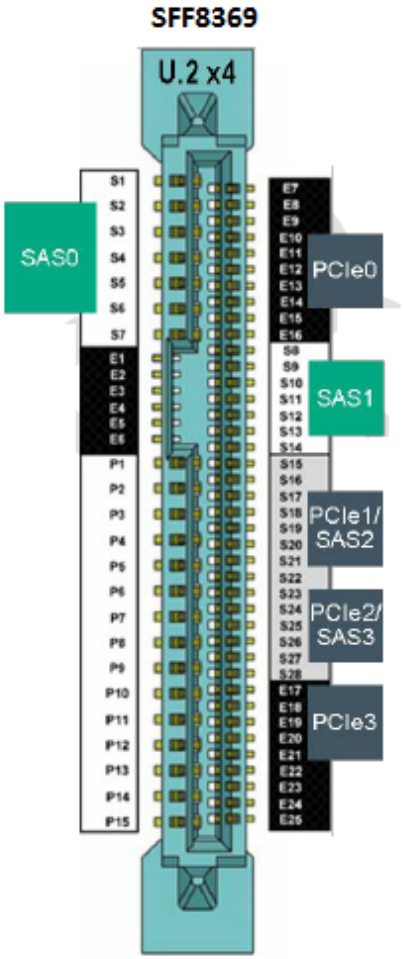
Purpose / Function

- The FTL is responsible for logical to physical mapping of data!
- The FTL is not a FIFO
  - Ensure that the SSD is evenly worn to prolong the life of the NAND (Wear-levelling) or background refresh.
  - Reclaim blocks previously deleted / unmapped by the OS so that a new write will not have to do a read / modify / write
    - Ensures peak performance is maintained (Garbage Collection)
  - Bad Block Management to handle invalid blocks
- FTL Size
  - Align to the OS low level drivers (512Bytes or 4KBytes)
  - 4K FTL is the most common, it aligns to the driver stack and meets the current DRAM vs SSD capacity requirements

Controller FTL

How far does a 4K FTL get us in terms of capacity?

- DDR size may become the limiting factor as NAND die size continuously increase
- What is achievable / affordable considering real-estate limitations? 16GBytes, maybe 32GBytes in the Future. With a 4K FTL (assuming a map entry of 4Bytes) that gives us a possible 32TB drive
- NAND will be capable of significantly larger capacities than this, even on a single 2.5” SFF 8639 drive



Controller Latency

- Hardware is playing an ever more important role in managing data in the controller, especially for high performance systems
- With so much data coming into the front end it cannot be expected that a simple buffer manager be singularly capable of processing multiple queues of data
  - *Hardware has to play a part to keep the latency down*
- It's likely that specialized front end hardware will be required with dedicated CPU capabilities

Power

Implications of Power

- We want the largest SSD that is capable of the fastest speeds for the smallest price…Right!
- Our new multicore controller are capable of achieving those blistering fast PCIe Gen4 speeds within acceptable power limits for the intended market segment
- Through-Silicon Via (TSV) NAND technology will have a significant positive impact on power, performance and real estate.

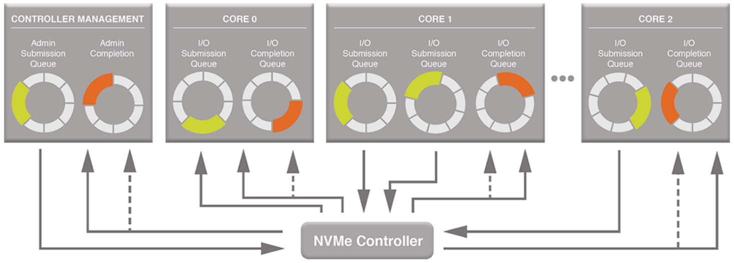
System Level Power

- Multiple die stack NAND also is (of course) a contributor to power, especially when running flat out
- This will highly depend on the NAND capabilities / performance / die stack etc.
- DDR will also contribute
- All the PCB discretes, voltage regulation, etc. cost power
- If we want to max out performance can we achieve that within the customers power budgets

NVMe Host

NVMe Queues

- Specification is very well future-proofed with support for up to 64K queues with 64K commands per queue
- Potential of 4096K commands

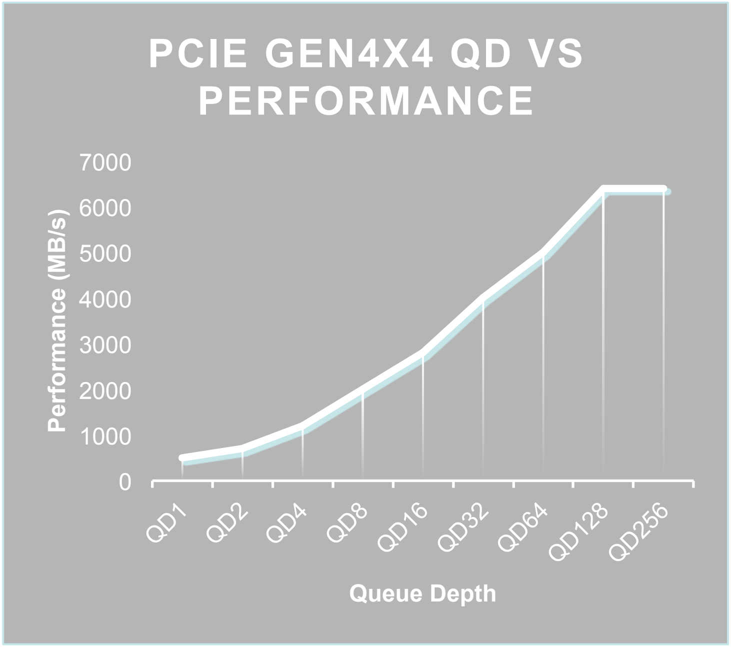


http://www.nvmexpress.org/nvm-express-overview/

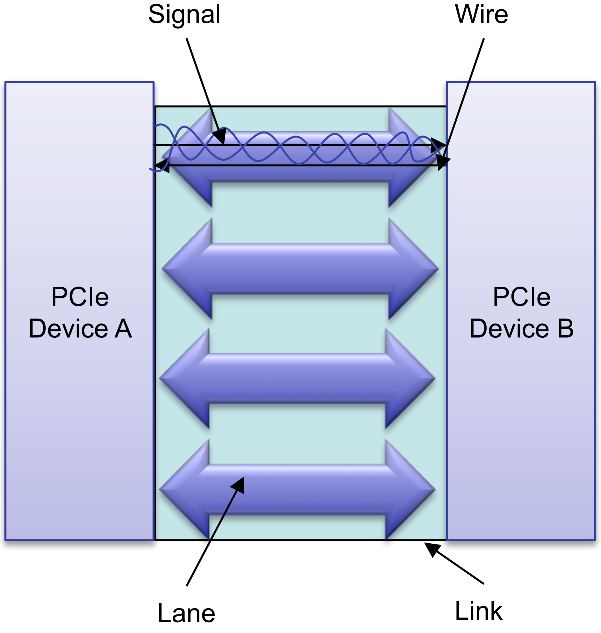
NVMe Host

In Practice:

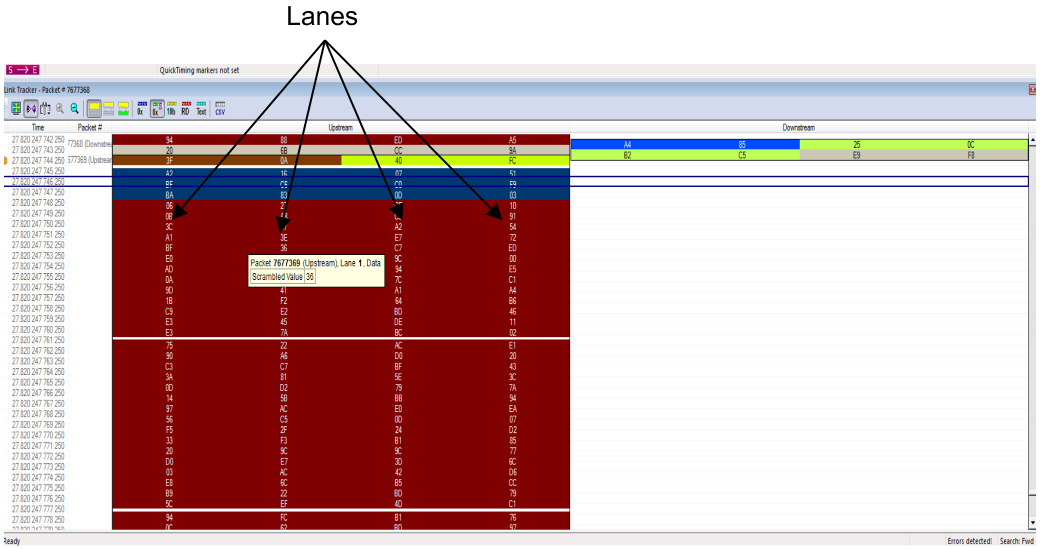
- Queues are mapped to CPU cores (this is set by the NVMe Driver)
- A server may have 128 CPU cores (including hyper threaded systems)
  - The number of NVMe queues will likely match the system capabilities
  - Queue depth is set by the controller (MQES), for example it could be set to a size of 1K commands per queue which would equate to a possible 128K commands
  - Command size can also be set by the controller (MDTS)
    - If set to zero the host driver determines the size
      - Note: Microsoft driver defaults to 1MB if MDTS is zero



PCIe Host



PCIe Gen 4x4



QuickTiming markers not set

Link Tracker - Packet # 7677368

Time	Packet #	Upstream				Downstream			
27.820 247 742 250	77368 (Downstrea	94	88	ED	A5	A4	85	25	0C
27.820 247 743 250		20	6B	CC	9A	B2	C5	E9	F8
27.820 247 744 250	577369 (Upstrean	3F	0A	40	FC				
27.820 247 745 250		A2	16	07	51				
27.820 247 746 250		BF	C6	C0	F9				
27.820 247 747 250		BA	83	0D	03				
27.820 247 748 250		06	27	[illegible]	10				
27.820 247 749 250		0B	[illegible]	[illegible]	91				
27.820 247 750 250		3C	[illegible]	A2	54				
27.820 247 751 250		A1	3E	E7	72				
27.820 247 752 250		BF	36	C7	ED				
27.820 247 753 250		E0	[illegible]	9C	00				
27.820 247 754 250		AD	[illegible]	94	E5				
27.820 247 755 250		0A	[illegible]	7C	C1				
27.820 247 756 250		9D	41	A1	A4				
27.820 247 757 250		1B	F2	64	B6				
27.820 247 758 250		C9	E2	BD	46				
27.820 247 759 250		E3	45	DE	11				
27.820 247 760 250		E3	7A	BC	02				
27.820 247 761 250		75	22	AC	E1				
27.820 247 762 250		90	A6	D0	20				
27.820 247 763 250		C3	C7	BF	43				
27.820 247 764 250		3A	81	5E	3C				
27.820 247 765 250		0D	D2	79	7A				
27.820 247 766 250		14	5B	BB	94				
27.820 247 767 250		97	AC	E0	EA				
27.820 247 768 250		56	C5	0D	07				
27.820 247 769 250		F5	2F	24	D2				
27.820 247 770 250		33	F3	B1	85				
27.820 247 771 250		20	9C	9C	77				
27.820 247 772 250		D0	E7	3D	6C				
27.820 247 773 250		03	AC	42	D6				
27.820 247 774 250		E8	6C	B5	CC				
27.820 247 775 250		B9	22	BD	79				
27.820 247 776 250		5C	EF	4D	C1				
27.820 247 777 250		94	FC	B1	76				
27.820 247 778 250		0C	62	BD	97				

Packet **7677369** (Upstream), Lane **1**, Data

Scrambled Value	36

Ready

Errors detected!

Search: Fwd

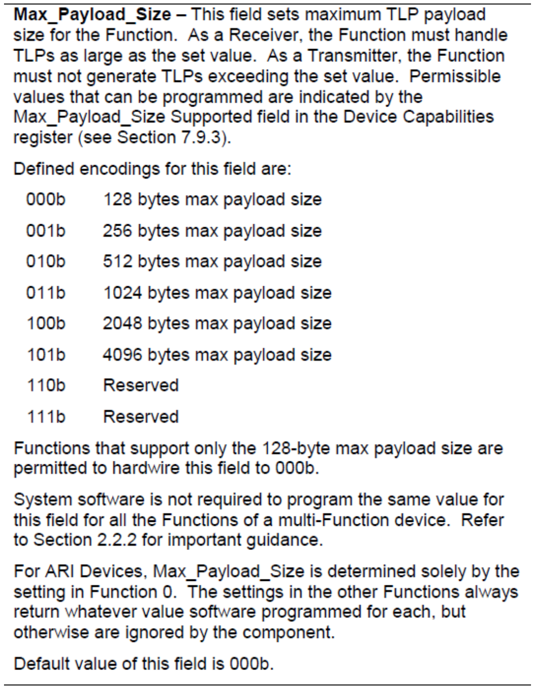
Upstream data to the host (read)

Downstream data from the host (ACK)

PCIe Gen 3x4 on the analyser

PCIe Host

- As can be seen from the previous slide, data on the PCIe bus is laid out like RAID0 data; small parts of data split across the 4 lanes
- Max payload across the PCIe interface is 4KBytes, however most systems are set to 128Bytes or 256Bytes for backwards compatibility
- Knowing your environment may improve your efficiency



Max_Payload_Size – This field sets maximum TLP payload size for the Function. As a Receiver, the Function must handle TLPs as large as the set value. As a Transmitter, the Function must not generate TLPs exceeding the set value. Permissible values that can be programmed are indicated by the Max_Payload_Size Supported field in the Device Capabilities register (see Section 7.9.3).

Defined encodings for this field are:

000b	128 bytes max payload size
001b	256 bytes max payload size
010b	512 bytes max payload size
011b	1024 bytes max payload size
100b	2048 bytes max payload size
101b	4096 bytes max payload size
110b	Reserved
111b	Reserved

Functions that support only the 128-byte max payload size are permitted to hardwire this field to 000b.

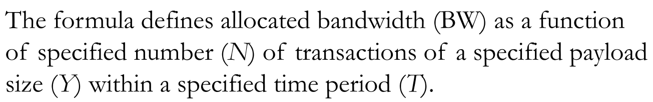
System software is not required to program the same value for this field for all the Functions of a multi-Function device. Refer to Section 2.2.2 for important guidance.

For ARI Devices, Max_Payload_Size is determined solely by the setting in Function 0. The settings in the other Functions always return whatever value software programmed for each, but otherwise are ignored by the component.

Default value of this field is 000b.

PCIe Host

$$BW = \frac{N.Y}{T}$$



The formula defines allocated bandwidth (BW) as a function of specified number (*N*) of transactions of a specified payload size (*Y*) within a specified time period (*T*).

Impact on Payload

- Using the previous example, everything staying the same except the payload, the following can be calculated
- Increasing the payload does have a positive impact but does not increase linearly
- The ratio between data transferred and TLP overhead decreases as payload increases

Payload	Throughput	% Increase
128 bytes	5 GB/s	
256 bytes	5.6 GB/s	12%
512 bytes	6 GB/s	7%
1024 bytes	6.16 GB/s	2.7%
2048 bytes	6.25 GB/s	1.5%
4096 bytes	6.3 GB/s	0.8%

Call to action!

- Host Drivers to align to system dynamics to increase efficiency and reduce latencies
- Observing the best PCIe Gen 4 performance will require large queue depths
  - Testing at QD1 doesn't provide the system with enough data to reach top performance
- Controller Capabilities
  - Power of controller, CPU capabilities, flash channel bandwidth
- NAND Capabilities
  - Reduce power with TSV NAND
- Align application level software could be better aligned to PCIe Gen 4
  - Example – If IOMeter had its own PCIe driver we could manipulate payloads to test efficiency