New SSD Interfaces and Their Impact on SSD Controller Architecture

Tim Canepa

Director of Architecture

LSI Flash Components Division,

an Avago Company

Legacy SSD Controller Architecture Anatomy

Traditional SOC Design

- Cached CPU architecture with TCM
- External DDR with multi-port memory controller

Interface is the main constraint

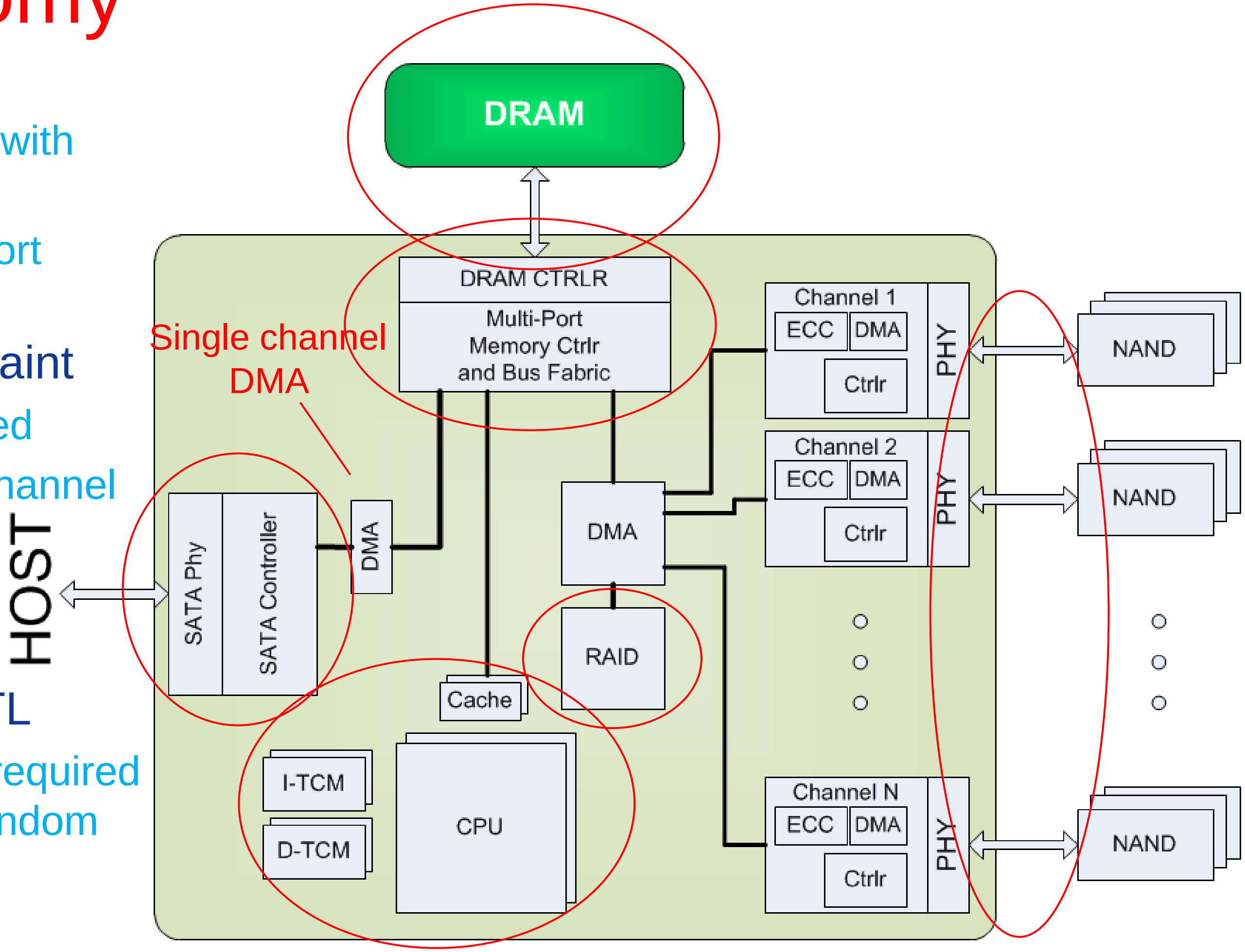
- Host limited vs. Flash limited
- 550MBps vs. 400MT per channel
- One transaction at a time on the wire, 32 max outstanding

DDR used for buffer and FTL

- Circa 2GBps of buffer BW required for 100% sequential and random corners

Not a lot of HW assists

- CPU(s) manages FTL directly in DDR via write-thru cache.



Along Comes PCIe Direct Attach Storage

- Scalable Interface
  - 500MBs x1 Gen2 – 16GBs x16 Gen3
- Full Duplex
- New, lightweight queuing interfaces (NVMe)
- Transaction interleaving
  - Enables ability to access all the flash bandwidth
- New command semantics
  - Fused commands
  - Hinting
  - Name spaces

Side-by-Side Comparison

SATA

- 550MB/sec
- ½ Duplex
- 32 command slots
- Serialized Data Transfers

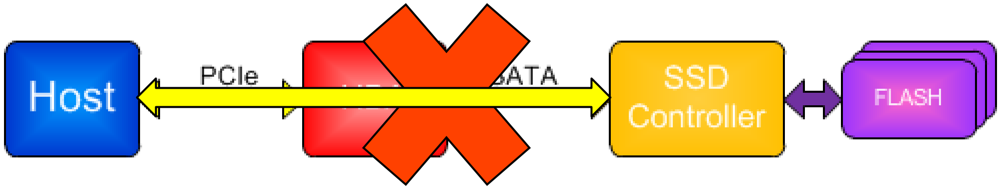
PCIe/NVMe

- Up to 16GB/sec
- Full Duplex
- Multiple deep command queues
- Interleaved data transfers

No more host interface BW limits

No more host interface Command submission limits

No more host interface concurrent transfer limits



Cut out the middle man and get to all the flash bandwidth!

Legacy SSD Controller Architectures (bottle necks)

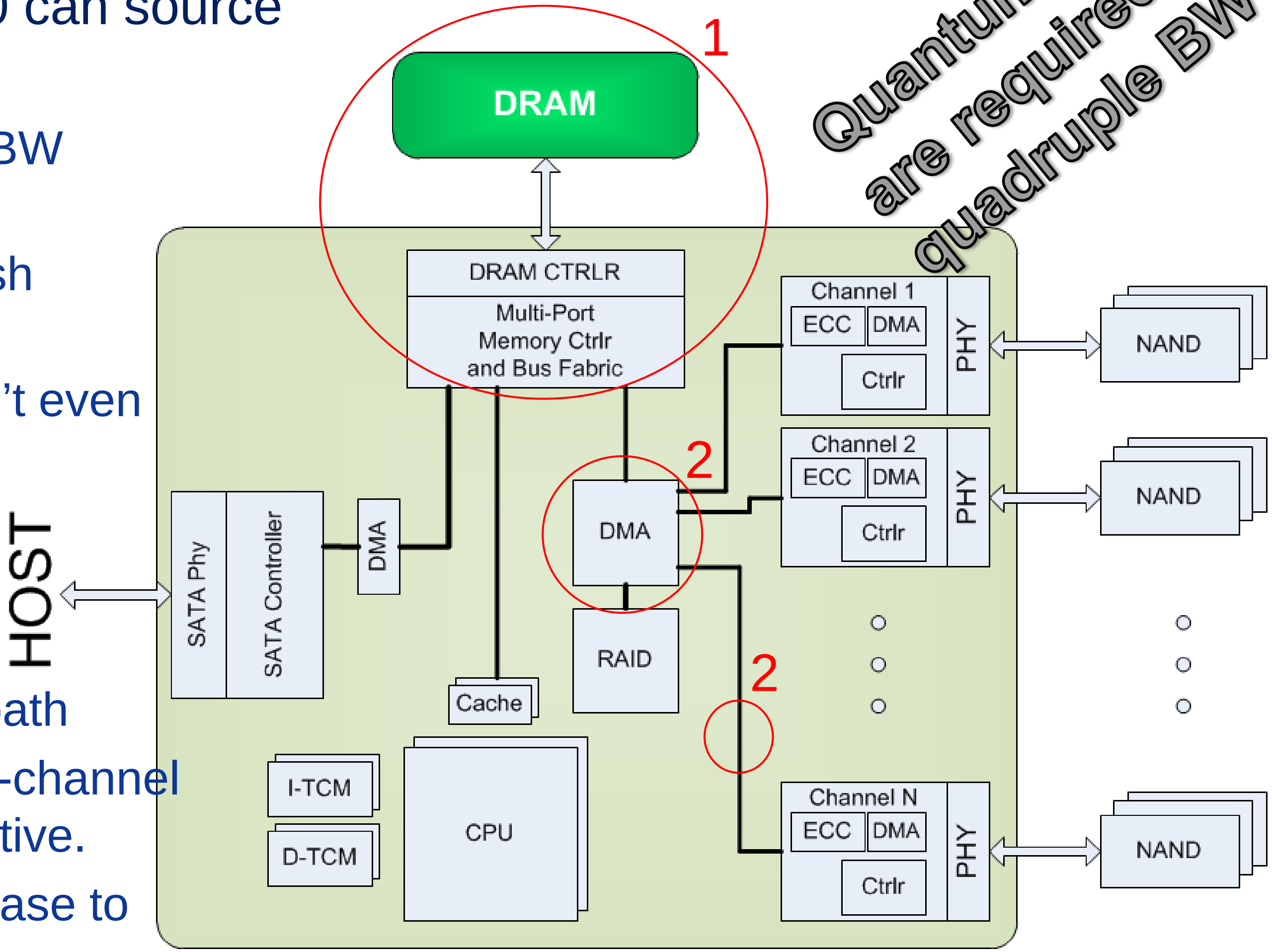
8 channels of ONFI 3.0 can source 3.2GBps

Bottleneck 1: DDR buffer BW

- Need well in excess of 6.4GBps to service flash alone.
- 32Bit DDR3 1600 won't even cut it
- Wider or faster DDR I/F is not attractive (power & cost)

Bottleneck 2: Flash Data path

- DMA needs to be multi-channel to keep all channels active.
- Bus BW needs to increase to >3.2GBps



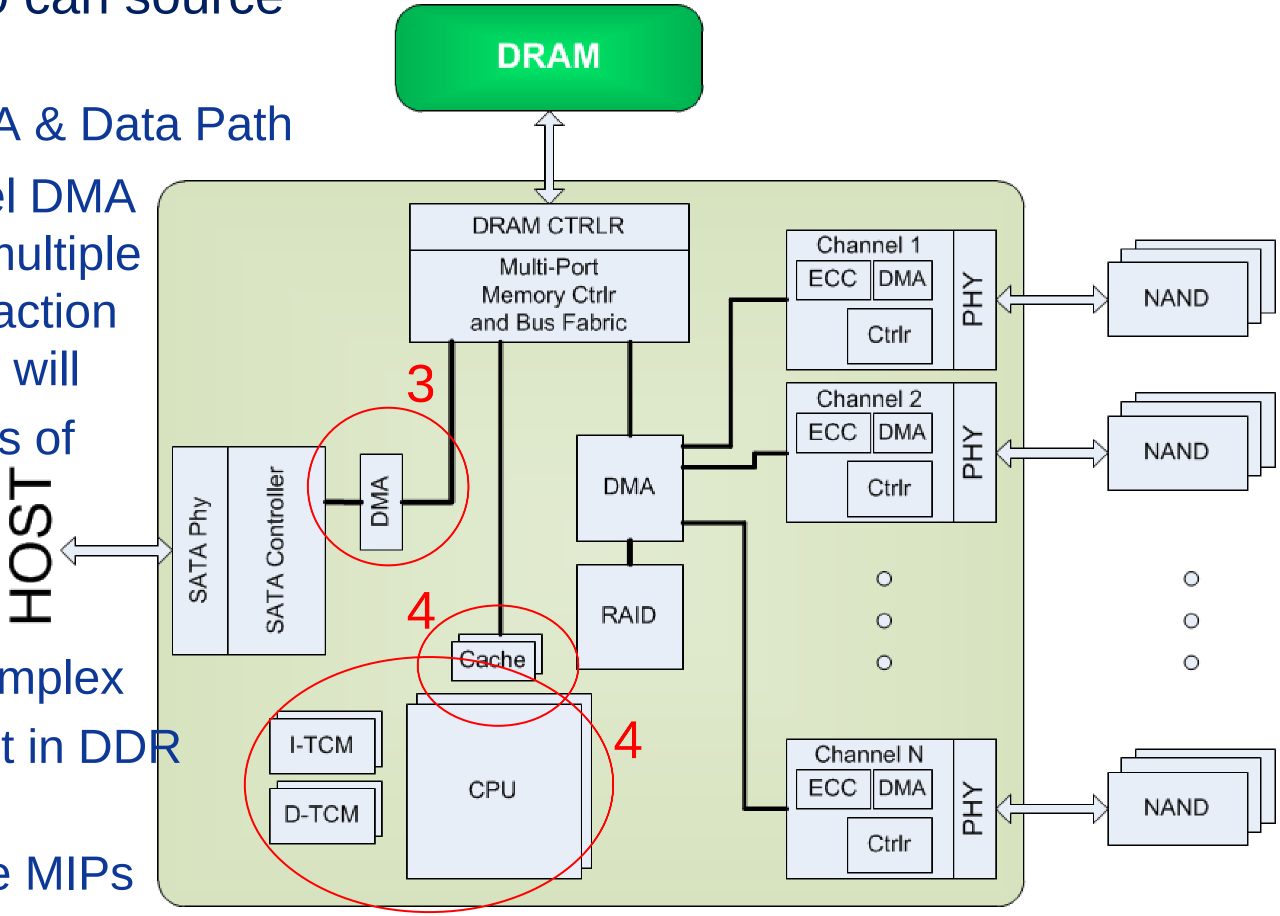
Quantum leaps are required to quadruple BW

Legacy SSD Controller Architectures (bottle necks - continued)

8 channels of ONFI 3.0 can source 3.2GBps

Bottleneck 3: Host I/F DMA & Data Path

- Requires multi-channel DMA to interleave data for multiple requests. If not, transaction tenure will increase as will
- requires well in excess of 3.2GBps of BW



Bottleneck 4: Processor complex

- Direct FTL management in DDR becomes prohibitive.
- More IOPs require more MIPs and more context tracking

Legacy Architecture Dealing with Full Duplex

80% Reads / 20% Writes Mix

Design Max: ~1,440MB/s

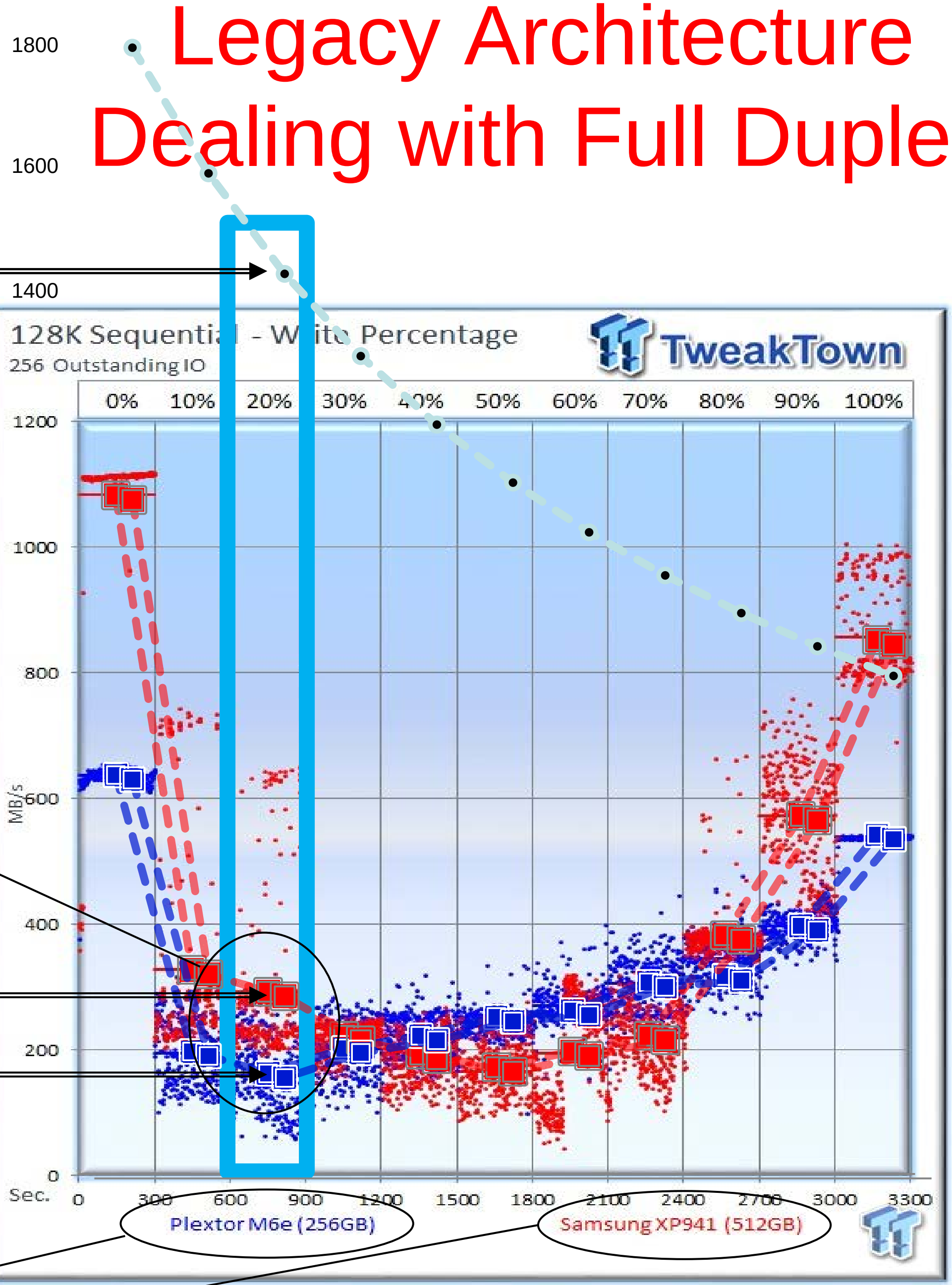
SF3700 architected for bi-directional PCIe traffic

Legacy Architecture bottlenecks capping performance with mixed workload.

Samsung XP941: ~290MB/s

Plextor M6e: ~160MB/s

Bathtub curve – not optimized for bi-directional traffic



PCIe Gen 2 x2

PCIe Gen 2 x4

Source: Plextor and Samsung from TweakTown
SF3700 from LSI @ 50% entropy, 480GB, 7% OP

How is Throughput Determined?

- Throughput is influenced by many factors
  - Flash die limits – a 1TB drive can produce16GBps of read BW
  - Flash channel limits
  - Internal B/W limits (e.g., buffers, compression engines, etc)
  - Tenures – how long are resources held before they can be reused
  - Host I/F limits
  - CPU limits
- What's key is understand what the limiting factor(s) is(are)
  - And at what range of parameters, as the limiting factor(s) may vary
  - And how to strike a balance to get the best performance in all corners

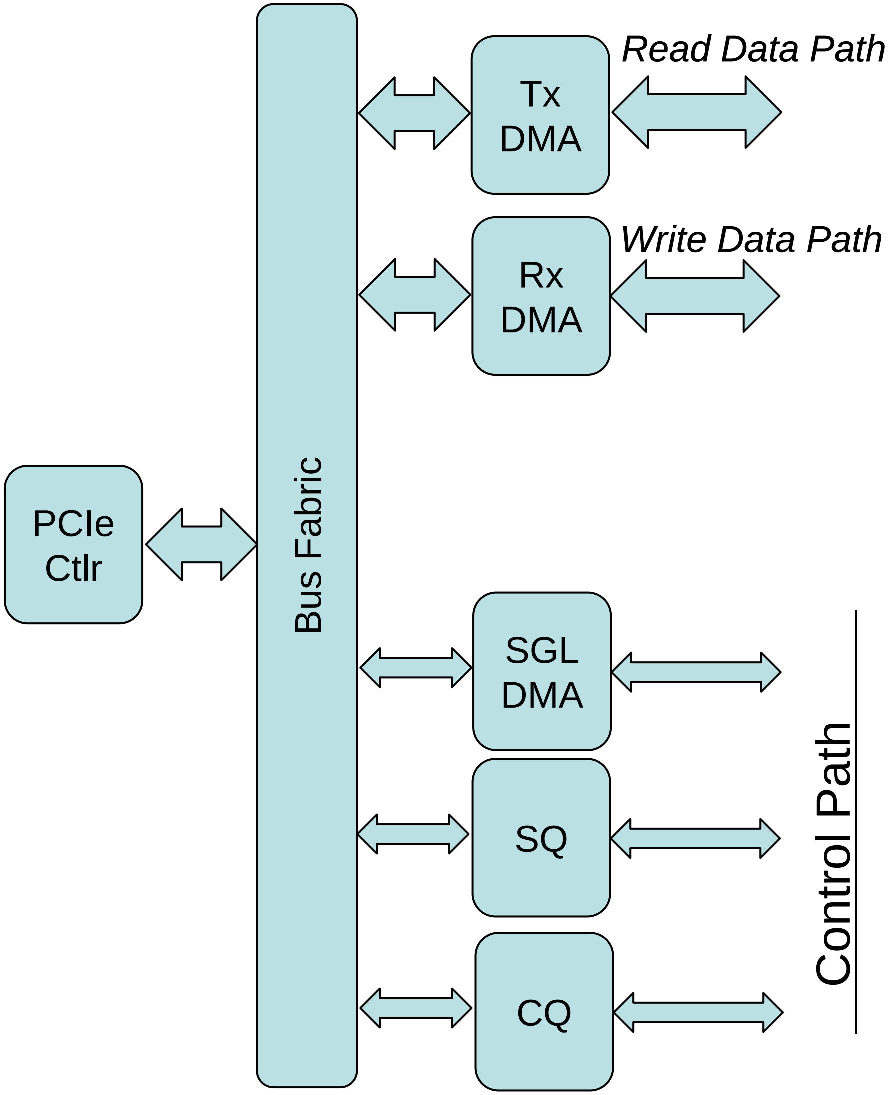
Scaling up and Striking a balance

- Host Interface
- Data Paths
- Flash bandwidth
- Processing Power

What does it take to build a scalable architecture?

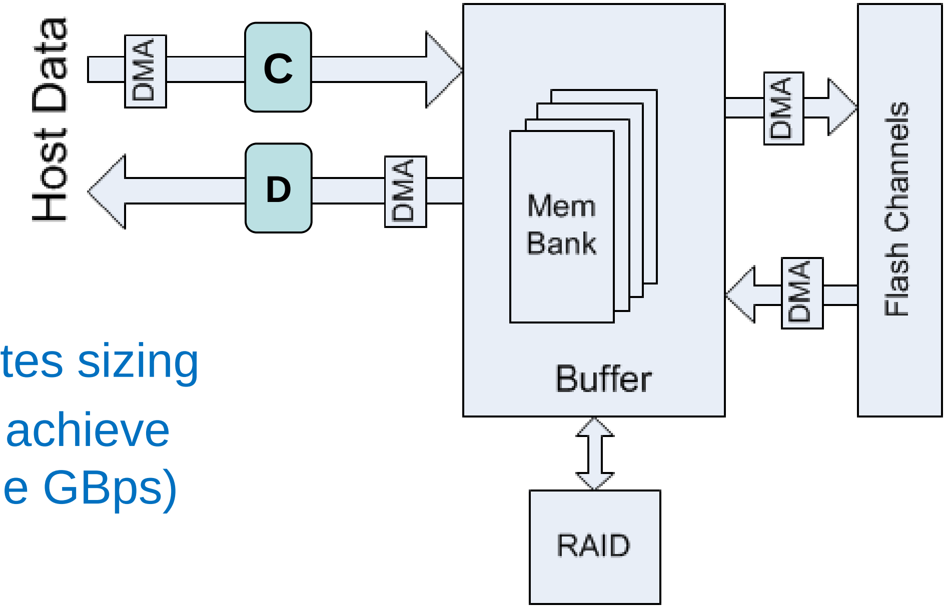
Host Interface

- Scalability and Interleaving are critical
- Start with Configurable PCIe Controller
  - multi-ported & cut-thru for CplD
- Add separate, multi-layer bus fabric with arbitration at TLP frame level.
- Individual DMAs and Control path logic connected to separate ports
  - Must have local buffering
- Clock domain crossings can convolute the design



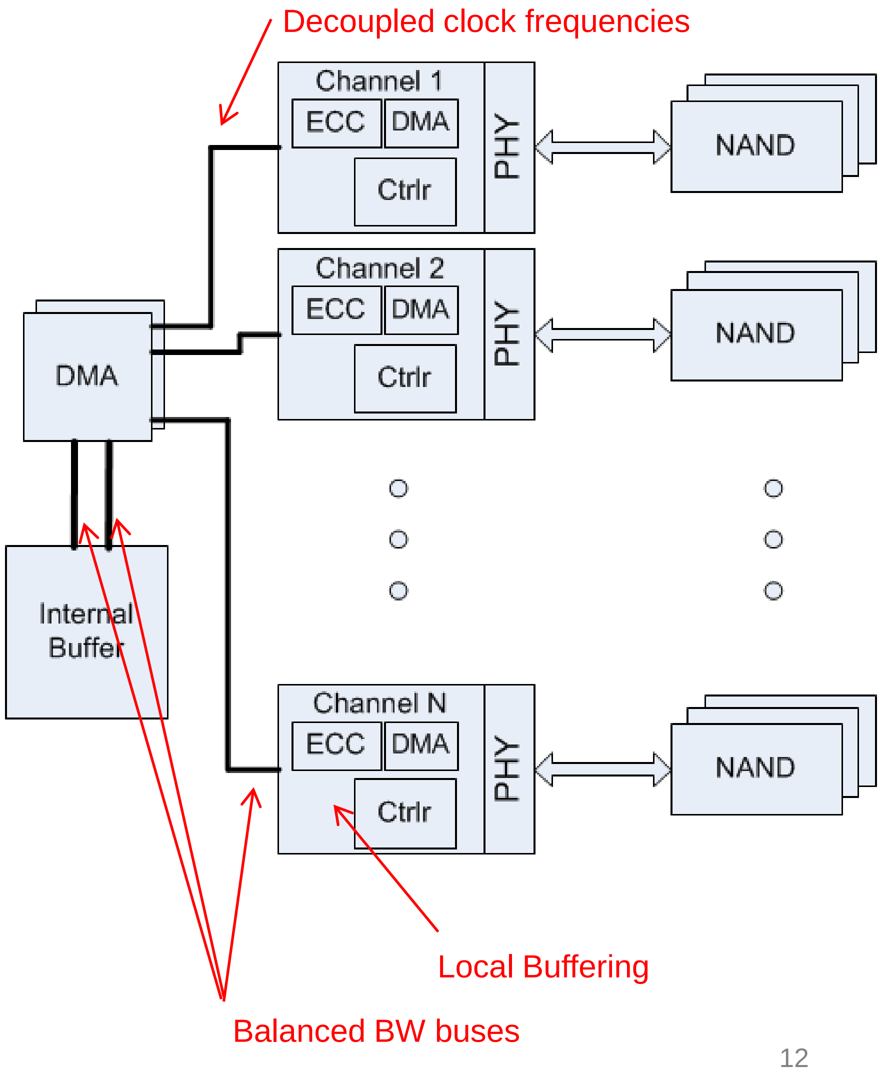
Data Path

- Bus Bandwidth
  - Generally good to have 1.5x or more internal BW than you want to deliver
- Internal Buffers
  - Write staging buffers
  - Flash transfer buffers
  - Size is based on tenure
  - Mixed workload complicates sizing
  - Multi-banking required to achieve BW requirements (multiple GBps)
- DMAs
  - Multiple Queued Requests
- Inline modules complicate things
  - Pull through compression engines requires pipelining



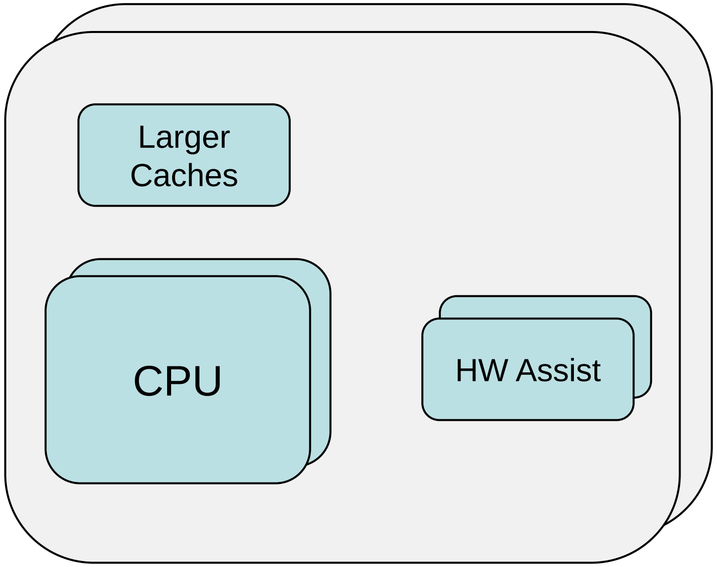
Flash Bandwidth

- Multi-Channel DMA
  - One read, one write is sufficient with balanced BW on buses
- Local Buffering in each channel
- Data bus frequency decoupled from flash clock



Processing Power

- Hardware assists needed to relieve processors from expensive tasks
  - Map Lookup assists
  - Recycling Assists
  - Buffer allocation assists
  - Even add assists for background operations
- Resize cache to hold working set
  - Elevated miss rates will crater MIPs
- Scale with asymmetric MP
  - Group activities into multiple asymmetric processor groups
  - Easiest way to scale “run to completion” architectures



Asymmetric processor Groups

Adding up all the Architectural Changes

- When you sum up the required architectural changes…
- You realize that you practically have to throw your old SATA architecture out and start from scratch

FTL Impacts

- Bounding recovery Time requires new FTL techniques
  - Journaling FTL is essential to manage flush rates and to bound the recovery interval.
- High IOPs need FTL HW assists
  - Map lookups
  - Recycling
  - Extensible structures for Storing and Managing Hints

Firmware Impacts

- More commands in flight
  - More resources required to track/manage commands
  - Algorithmic changes – sub-scalar algorithms don’t show up at low IOPs (data coherency, searches, etc)
  - Scheduling becomes more challenging to keep the pipe full
  - Minimizing Tenure is critical
- Flash scheduling considerations
  - Same amount of flash as SATA SSD, but substantially more IOPs
  - On chip buffering requires more robust resource management
  - Requires entire scheduling layer redesign
- Background task management
  - Shorten background task segments. On “run to completion” architectures

Questions?