violin MEMORY

Hot Chips 2013

Fast Storage for Big Data – Tutorial 2
Implications of using Flash devices in an enterprise storage server

Kevin Rowett

Biography

- Currently: Advisor at Violin Memory
- Moving into full time consulting role at Iron Heart Consulting
- SVP Engineering – Jan 2010 thru July 2013
- Other start-ups
  - Cacheflow (now Blue Coat Systems)
  - Force10 Networks
  - Gigafin
  - Seamicro
- Larger companies – Cisco, Tandem, IBM
- Many years of hardware engineering, systems design, managing

Violin Memory

- Founded 2005, first flash product in 2009
  - Well capitalized, $800+ million valuation
- 250+ top tier customers
- Toshiba, Juniper, SAS Ventures investment
- Global presence
  - Worldwide headquarters in Silicon Valley, USA; Offices in North America, EMEA and APAC
  - 500+ employees (50 in EMEA)
- Awarded Silicon Valley ‘Company of the Year’ by AlwaysOn in August 2011
  - Previous winners include Google, Salesforce.com, YouTube, Twitter, MySQL

Flash devices are…

- In a word: “tricky”
  - Read: 70us, Write: 1-2ms
  - And then there is block erase: 5-10ms
  - Block failures
  - Die failures
  - Read bit errors, program errors, read disturb
  - Parameters change on each new process update
  - Many different devices to choose from
- Summary
  - Flash **is not** “slow DRAM”
  - Lots of insight **is** needed

Flash controller design considerations

- Schedule I/O
  - Lay out blocks in devices based on expected usage and device access characteristics
  - Device blocks are 8K or 16K – usually want to spread write across more devices
  - Avoid “write cliff” effect by managing I/O types and GC across multiple blocks and devices
- Traditionally all done in the so-called “Flash translation layer” (FTL)

Flash controller design: Error handling

- Error Correction
  - Every read has many bit errors, so must use some form of forward error correction: BCH and/or LDPC
    - BCH is die area intensive, and Flash spare area consumer
    - Might not provide enough correction past 19nm
    - Lowest latency
    - LDPC in conjunction with BCH provides much better error rate
    - Requires special access to devices
    - Increases latency as much as 4 times

Flash controller design: failures

- Block failures do happen; cannot correct, or device reports program error
- Die failures: all commands get refused
- Module no longer responsive
- Some form of RAID needed

OTS (Off-the-shelf) Flash controllers

- Many very good ones exist
  - Sandforce/LSI, Marvell, PMC-Sierra, IDT
- More coming
- Much of the design issues are solved using OTS (off-the-shelf) controllers in SSDs
- But not all of them….

If your customer is Fortune 100 enterprise storage…

- These specific product requirements must influence your Flash-based design
  - No latency jitter
    - Expect 70us reads, always, every time, under all conditions and failures
  - Fault tolerant and hotswap
    - No single point of failure anywhere, and failed assemblies can be replaced while product continues to deliver
  - Cannot ever lose a byte of storage
  - **Be much faster than HDD (5x IOPs, 20-300 times for apps)**

- Looking ahead, you might also consider thin provisioning, snapshots, and backups

Performance expectations

- Ultra-low latency for reads and writes
  - 30-400 microseconds (25 times faster than disk)
- Almost unlimited IOPs
  - 500,000-1,000,000 in 3U (1000 time improvement over disk)
- High throughput
  - 3-5GB/sec in 3U (20 time improvement over disk)
- Zero contention under-load
  - No tuning, no workload separation

Application of OTS controllers

+ Solve the technical issues of dealing with Flash; in large chunks such as SSD
- Have an important lag in supporting newer devices (cost disadvantage)
- Do not solve “write cliff” effect and die failures in an enterprise product
- Major design point – making Flash usable as an HDD plug-in replacement

Our Solution

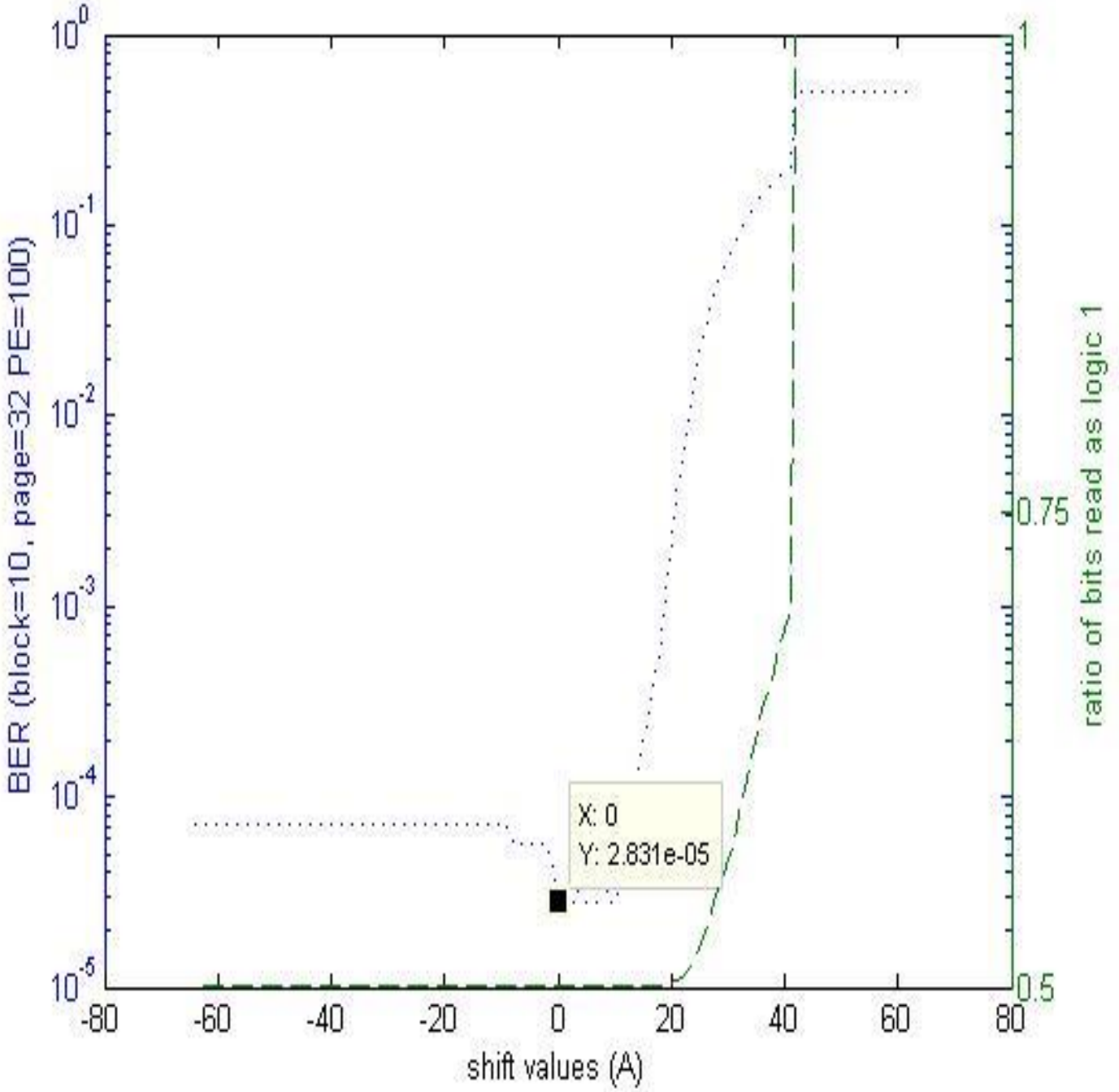
- Implement our own controller in an Altera FPGA
  - FPGAs all working together, under one design
  - Common fault handling and response
  - Did not have to correct for design mistakes of other vendors
  - Software and hardware well blended
- Design and patent (which was granted) a write hiding technique
  - Use RAID to guarantee a block can always be read
- Form good partnerships with our vendors
  - Understand Flash device issues, suggest improvements
  - Pricing, and access to newer technology
  - Steady flow of parts for a start-up

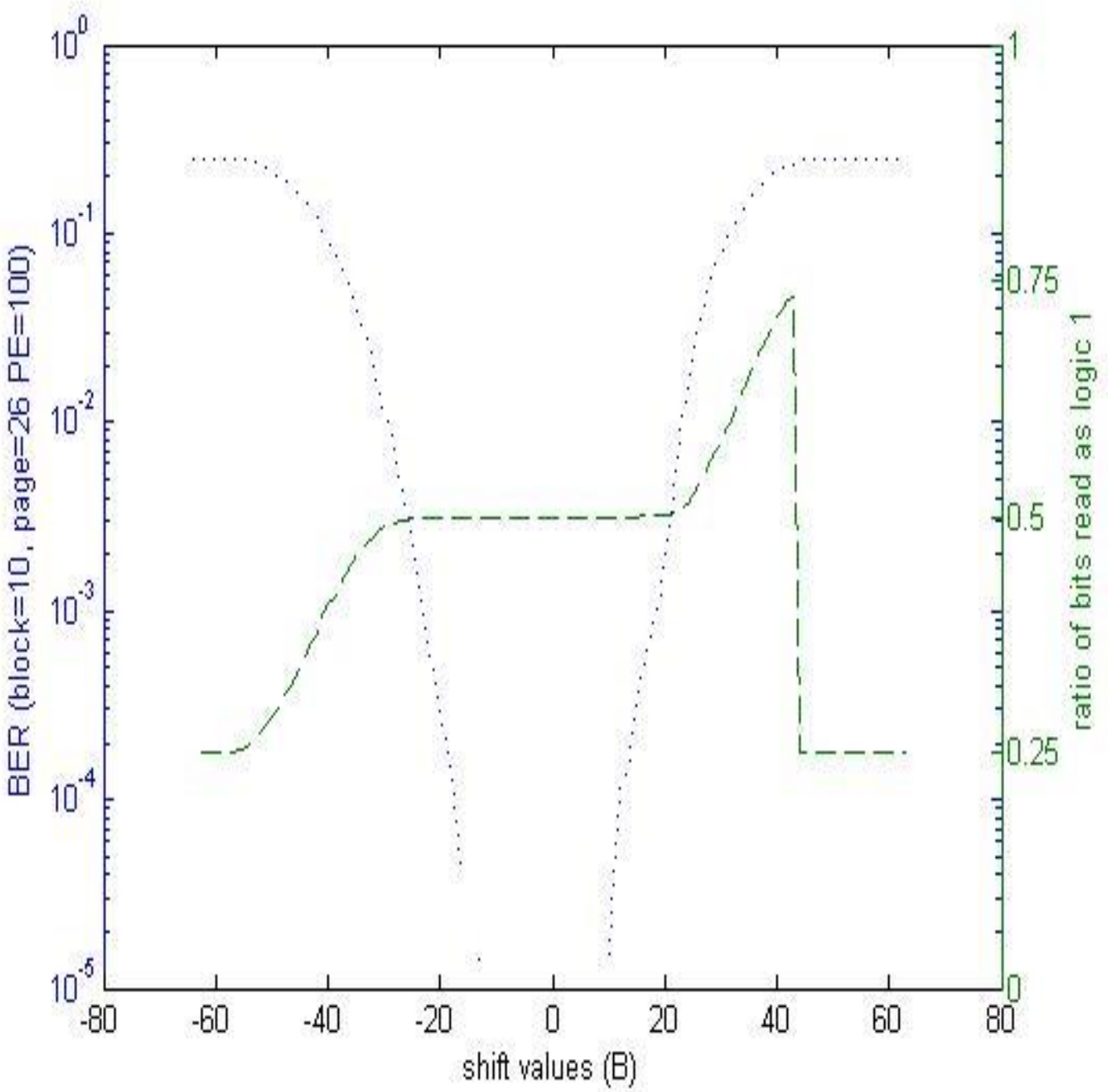
ALTERA
MEASURABLE ADVANTAGE™

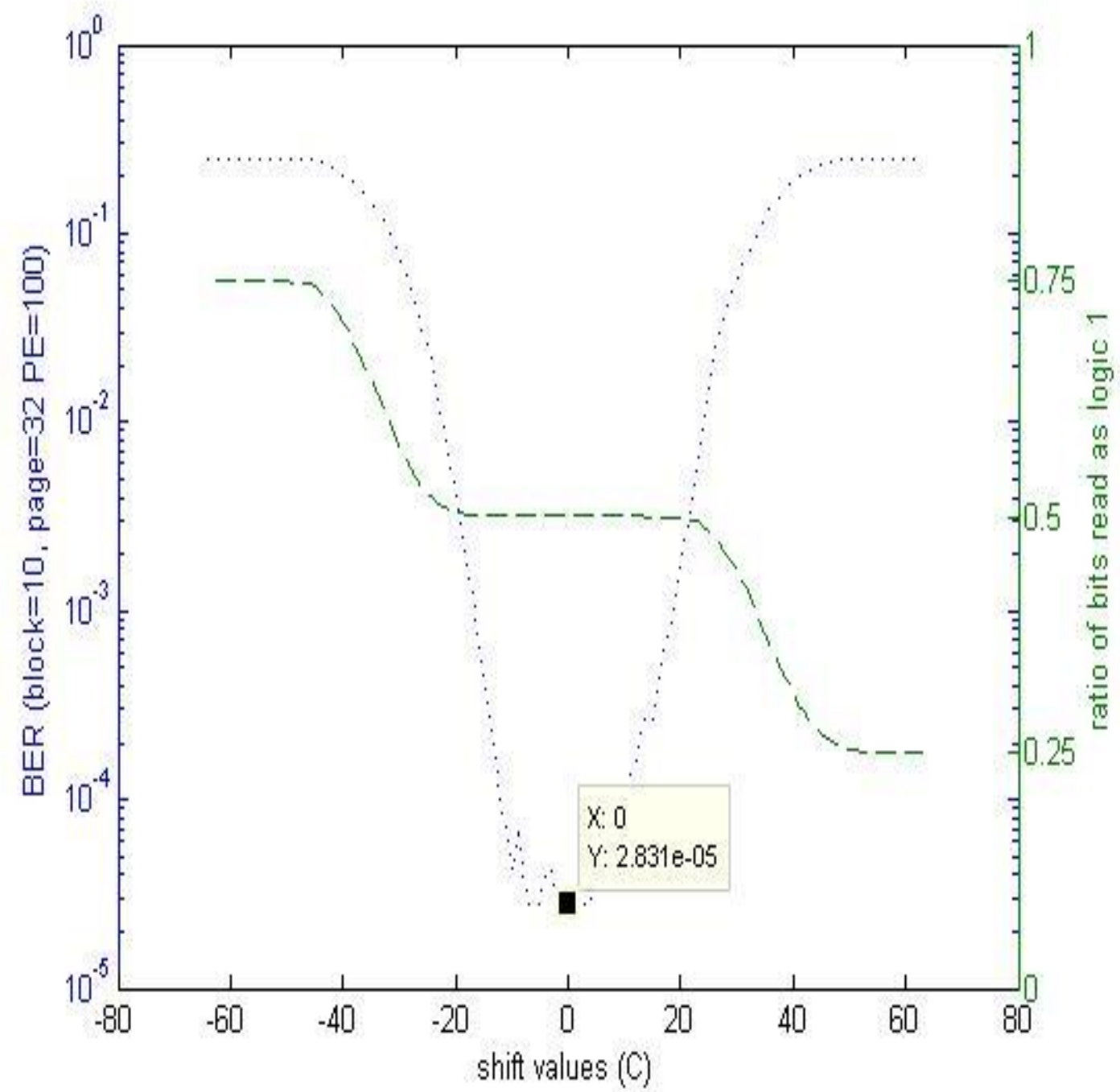
TOSHIBA

Needed device data for our own controller

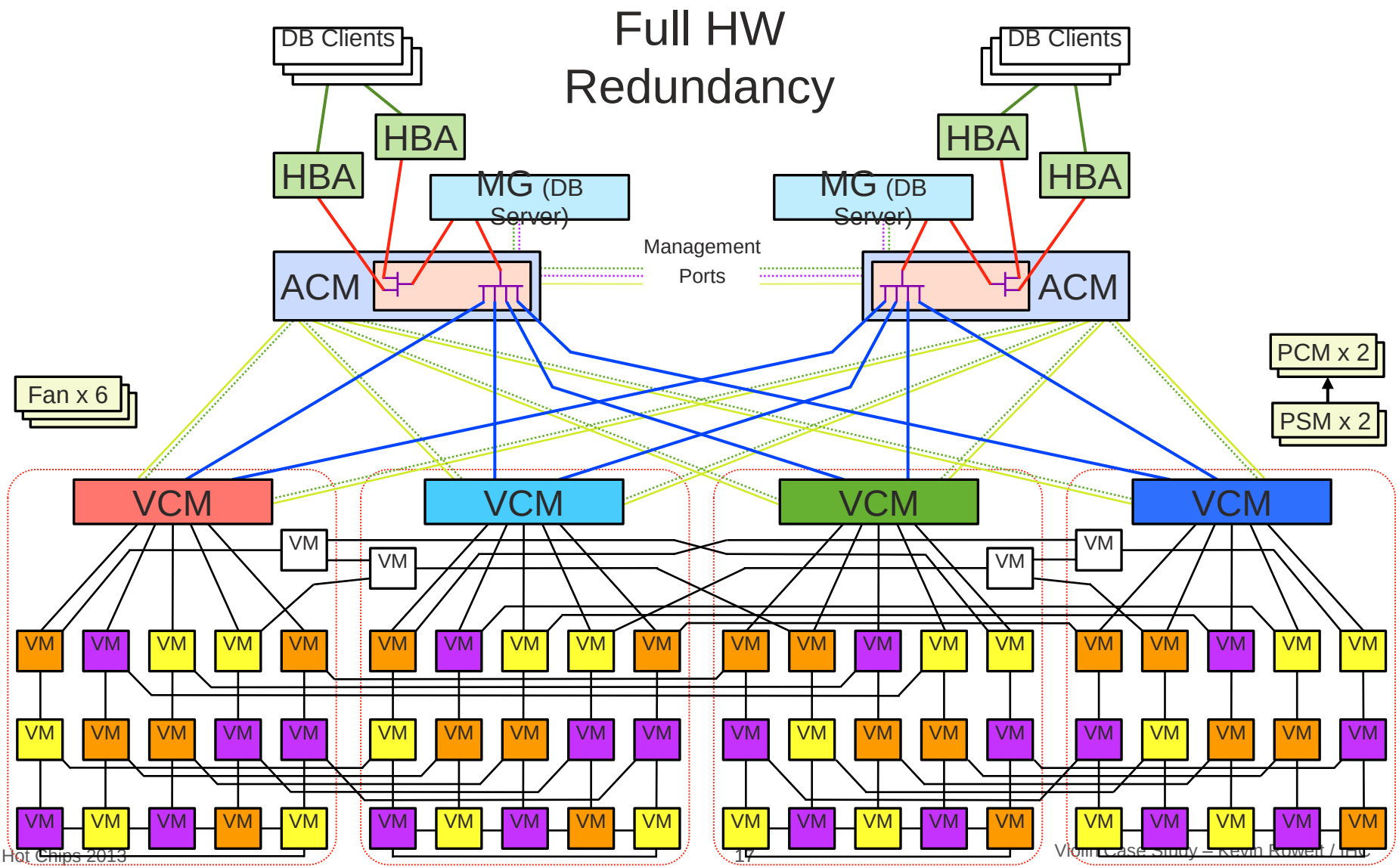
[This page intentionally left blank]


BER (block=10, page=32 PE=100)
ratio of bits read as logic 1
shift values (A)
X: 0
Y: 2.831e-05
10^0
10^-1
10^-2
10^-3
10^-4
10^-5
1
0.75
0.5
-80
-60
-40
-20
0
20
40
60
80


BER (block=10, page=26 PE=100)
ratio of bits read as logic 1
shift values (B)
10^0
10^-1
10^-2
10^-3
10^-4
10^-5
1
0.75
0.5
0.25
0
-80
-60
-40
-20
0
20
40
60
80


BER (block=10, page=32 PE=100)
ratio of bits read as logic 1
shift values (C)
X: 0
Y: 2.831e-05
10^0
10^-1
10^-2
10^-3
10^-4
10^-5
1
0.75
0.5
0.25
0
-80
-60
-40
-20
0
20
40
60
80

VXM Architecture



Conclusion

End-to-end cohesive design approach worked

Thank You!

Backup slides

[This page intentionally left blank]

Violin Memory Flash Memory Arrays

INSANELY POWERFUL	AMAZINGLY ECONOMICAL	ALWAYS AVAILABLE
Eliminate IO bottlenecks Drastically reduce latency - 1m IOPS/3U appliance. - Sub-millisecond latency	Cost approaching T1 disk. Cost/IOP class leading. Minimized TCO for analytics.	Full enterprise class availability features: - 100% serviceable. - No downtime.

Engineered for Flash
Tier 1 storage

Design approach of Violin Flash Memory Arrays

- End to end design (IOP request to response)
- Educate staff about Fault Tolerant design
- Understand Flash devices
- Re use of existing design elements
- Implications of SCSI protocol
- Full error detection, fault prediction, fault testing, fault injection for error recovery testing

Major design issues

- Logic gate count for BCH in 19nm devices
- PCIe switches
- Idiosyncrasies of various server operating systems
  - VDI, MS Clustering, AIX, and even OpenVMS!
- Tuning foreground and background GC bandwidth
- VIMM hotswap support
- Backward compatibility

If we had the chance for a “do over” we would…

- Re-write the control plane software, rather than reuse
- Go to a 4U form factor due to power density
- Not reuse the DIMM connector
- Do much more depth mechanical and thermal modelling than was actually done
- Change the use of PCIe bus inside
- Split out fabric from flash controller