FireBox:

A Hardware Building Block for the 2020 WSC

Krste Asanović & David Patterson

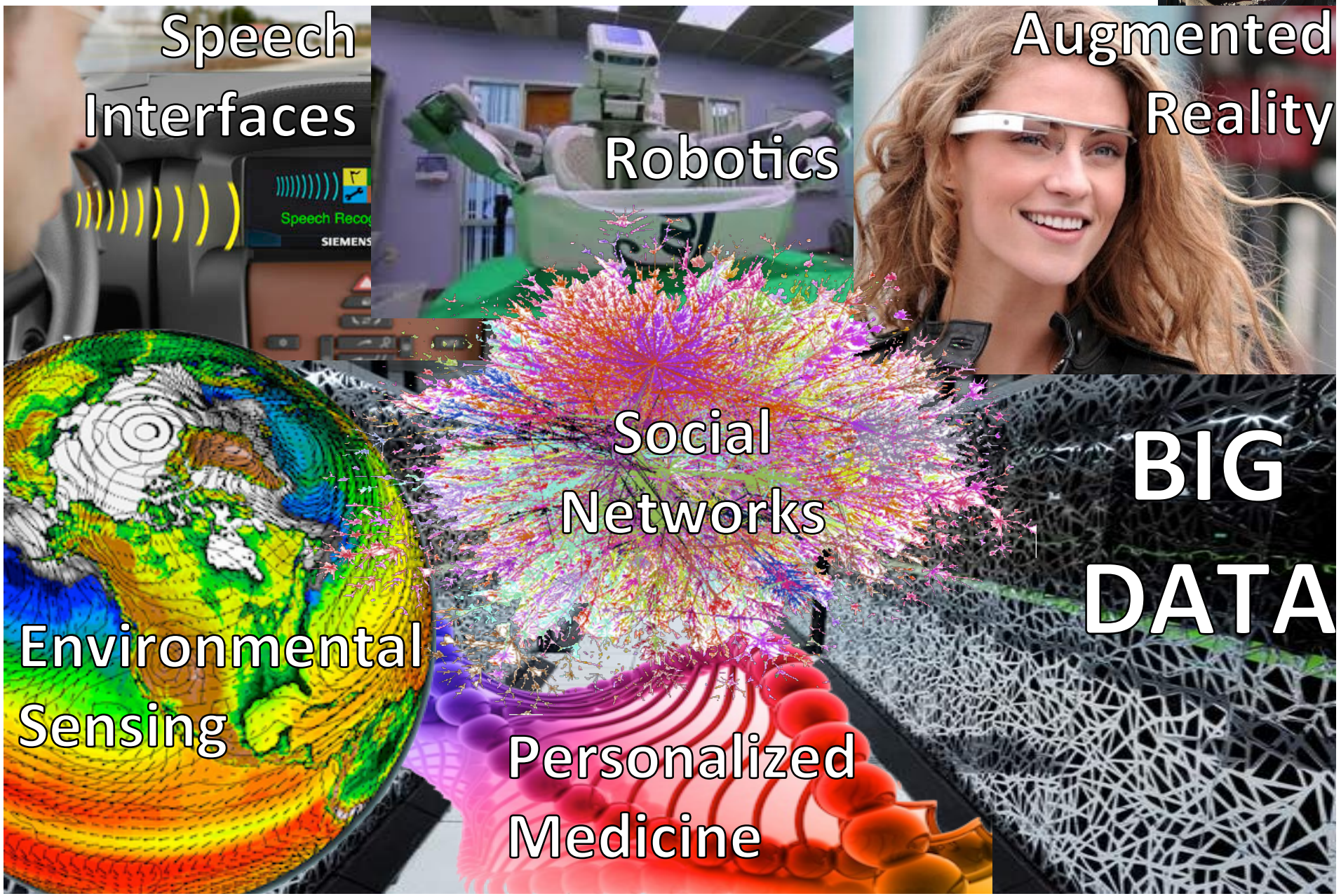
aspire.eecs.berkeley.edu

FAST, Santa Clara

February 19, 2014

ASPIRE
UC Berkeley

Upcoming Applications



Warehouse-Scale Computers (WSCs)

- Computing migrating to two extremes:
  - Mobile and the "Swarm" (Internet of Things)
  - The "Cloud"

- Most mobile/swarm apps supported by cloud compute
- All data backed up in cloud

- Ongoing demand for ever more powerful WSCs

Three WSC Generations

1. ~2000: Commercial Off-The-Shelf (COTS) computers, switches, & racks
2. ~2010: Custom computers, switches, & racks but build from COTS chips
3. ~2020: Custom computers, switches, & racks using custom chips

- Moving from horizontal Linux/x86 model to vertical integration WSC_OS/WSC_SoC (System-on-Chip) model
- Increasing impact of open-source model across generations

Talk Outline

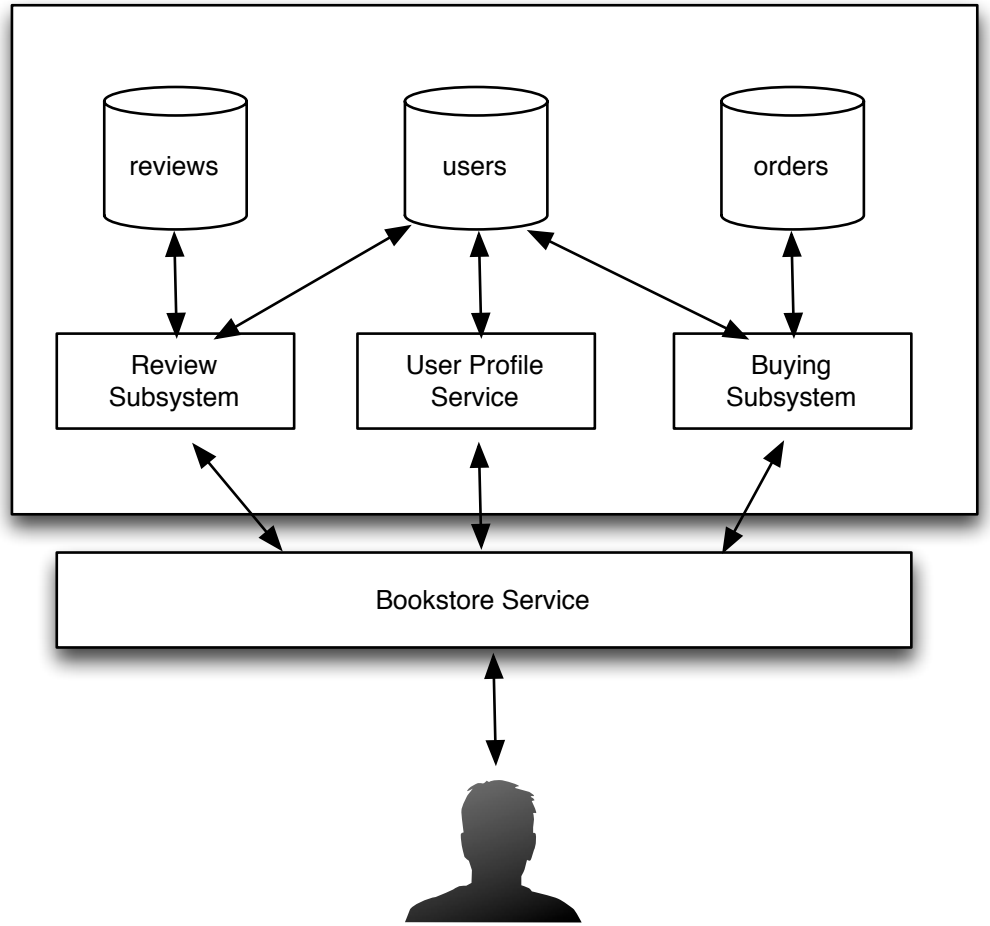
- WSC Trends and Challenges
  - Programming productivity
  - Tail latency
  - Memory hierarchy
  - Security
  - Technology scaling
  - Open-source hardware
- FireBox
  - 1,000 sockets, 100PB NVM, 4Pb/s network
- DIABLO
  - How to simulate a WSC at scale?

Service-Oriented Architecture

- SOA: Software (SW) architecture where all components are designed to be services
- ≈Abstract data types
  - access only with method, not direct address
- Communicate only via the network
- Apps composed of interoperable services
  - Easy to tailor new version for subset of users
  - Also easier to recover from mistake in design
- Contrast to "SW silo" without internal APIs

Bookstore: Silo

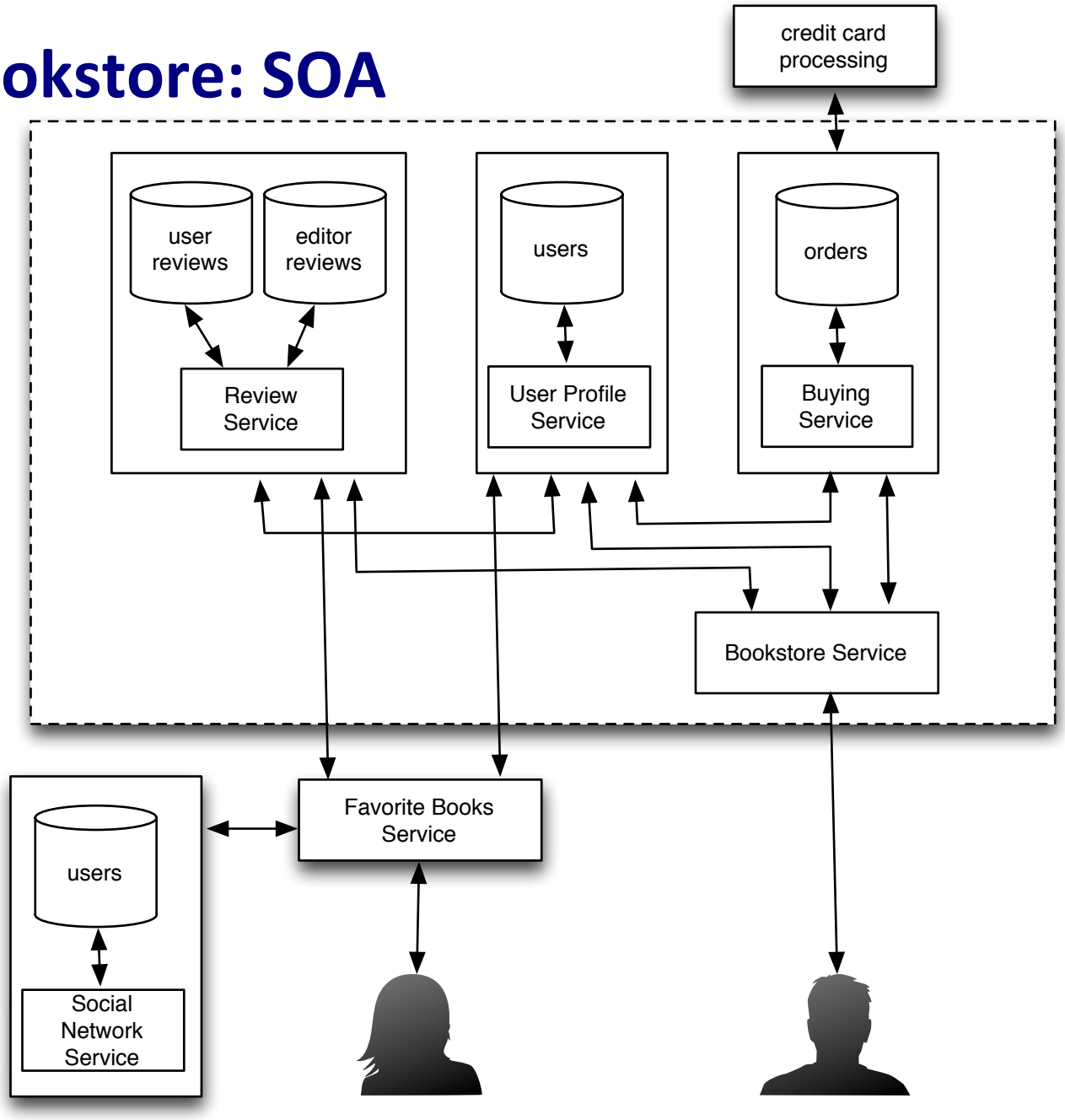
- Internal subsystems can share data directly
  - Review access user profile
- All subsystems inside single API ("Bookstore")



(Figure 1.2, *Engineering Software as a Service* by Armando Fox and David Patterson, 2nd Beta edition, 2013.)

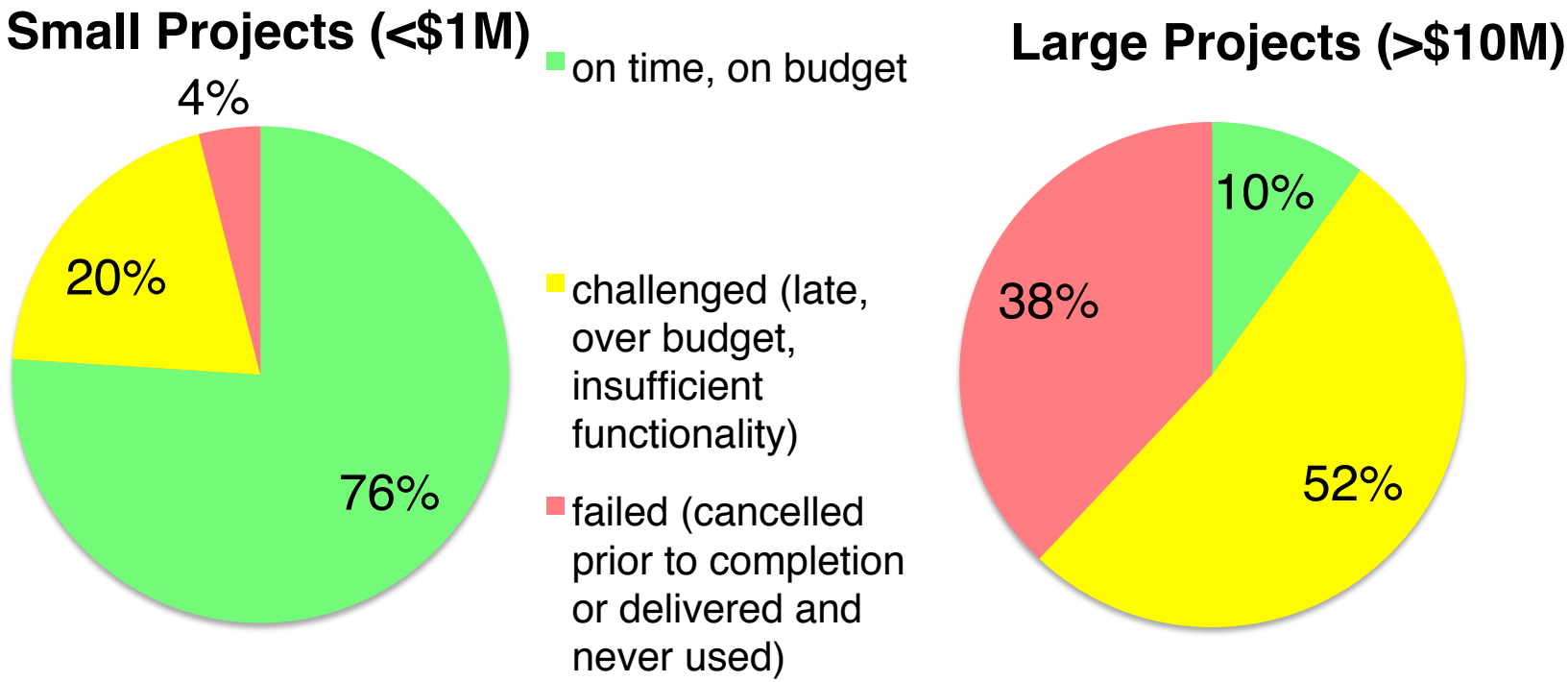
Bookstore: SOA

- Subsystems independent, as if in separate datacenters
  - Review Service access User Service API
- Can recombine to make new service ("Favorite Books")



(Figure 1.3, *Engineering Software as a Service* by Armando Fox and David Patterson, 2nd Beta edition, 2013.)

Small vs. Large Software-Development Projects



Do Big Project via Service-Oriented Architecture + Many Small Services!

CHAOS MANIFESTO 2013 Think Big, Act Small, www.standishgroup.com.
Based on the collection of project case information on 50,000 real-life IT environments and SW projects. Surveying since 1985.

CEO: Amazon shall use SOA! (2002, 7 years after started company)

1. "All teams will henceforth expose their data and functionality through service interfaces."
2. "Teams must communicate with each other through these interfaces."
3. "There will be no other form of interprocess communication allowed: no direct linking, no direct reads of another team's data store, no shared-memory model, no back-doors whatsoever."
4. "Service interfaces must be designed from the ground up to be externalizable. That is to say, the team must plan and design to be able to expose the interface to developers in the outside world. "
5. "Anyone who doesn't do this will be fired."

Old vs. New Conventional Wisdom

- Old Conventional Wisdom: Fault tolerance is critical for Warehouse-Scale Computer (WSC)
  - Build reliable whole from less reliable parts
- New Conventional Wisdom: Tail tolerance also critical for WSC, ***Slow = failure***
  - Build predictable response whole from less predictable parts

Table 1. Individual-leaf-request finishing times for a large fan-out service tree (measured from root node of the tree).

Conventional Architecture Target

Tail-Tolerant Target

	50%ile latency	95%ile latency	99%ile latency
One random leaf finishes	1ms	5ms	10ms
95% of all leaf requests finish	12ms	32ms	70ms
100% of all leaf requests finish	40ms	87ms	140ms

Dean, J., & Barroso, L. A. (2013). The tail at scale. *CACM*, 56(2), 74-80.

WSC HW Cost-Performance Target: Old vs. New Conventional Wisdom

- Old CW: Given costs to build and run a WSC, primary HW goal is best cost and best *average* energy-performance

- New CW: Given difficulty of building tail-tolerant apps, should design HW for best cost and best *tail-tolerant* energy-performance

Techniques for Tail Tolerance

Software (SW)

- Reducing Component Variation
  - Differing service classes and queues
  - Breaking up long running requests
- Living with Variability
  - *Hedged Requests* – send 2nd request after delay, 1st reply wins
  - *Tied requests* – track same requests in multiple queues

Hardware (HW)

- Higher network bisection BW, reduce queuing
- Reduce per-message overhead (helps hedged/tied req.)
- Partitionable resources (BW, cores, caches, memories)

Dean, J., & Barroso, L. A. (2013). The tail at scale. *CACM*, 56(2), 74-80.

Memory Hierarchy: Old vs. New Conventional Wisdom

- Old CW: 3-Level memory hierarchy / node

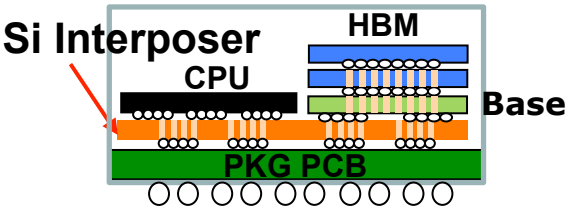
1. DRAM

2. Disk

3. (Tape)

- New CW: "Tape is Dead, Disk is Tape, Flash is Disk"*

1. Hi-BW DRAM



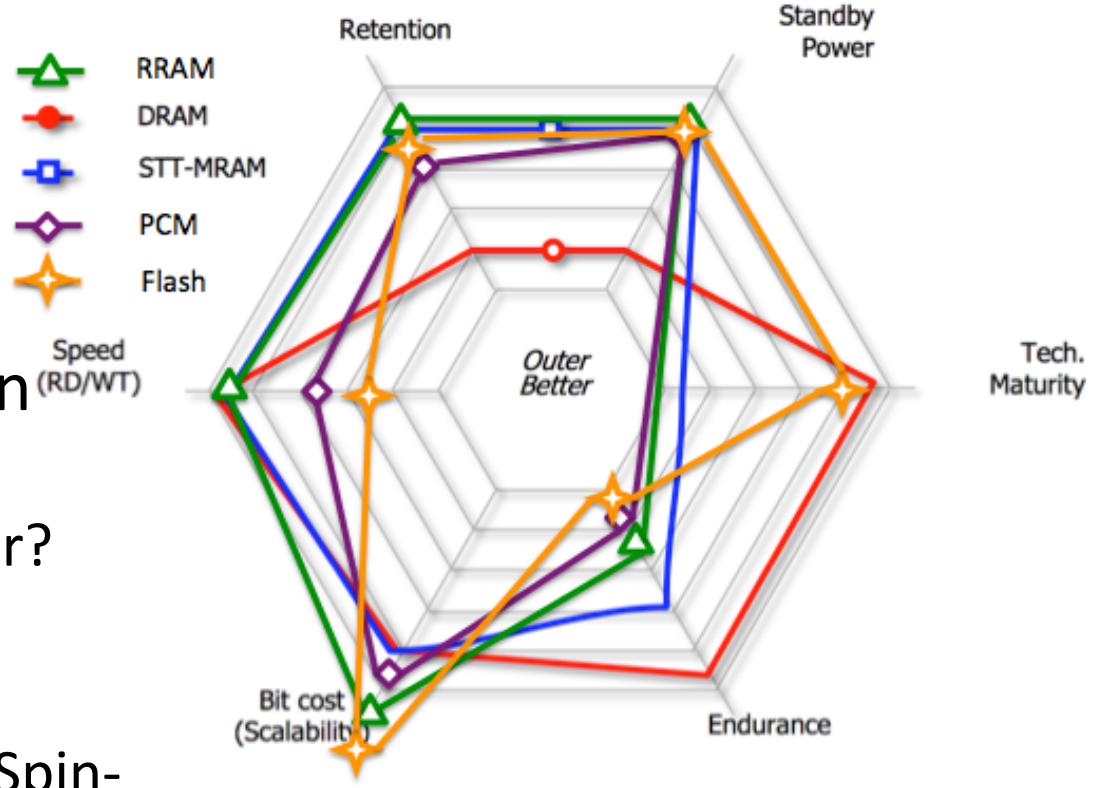
2. Bulk NVRAM

3. (Disk)

* *"Tape is Dead, Disk is Tape, Flash is Disk, RAM Locality is King" by Jim Gray, December 2006*

Non-Volatile Memory (NVM): Old vs. New Conventional Wisdom

- Old CW: 2D Flash will continue to grow at Moore's Law
- New CW: 2D ends soon
- Just 3D Flash, or new non-volatile successor?
  ≈DRAM read latency,
  + much better endurance
- Resistive RAM (RRAM) or Spin-Transfer Torque-Magneto-resistive RAM (STT-MRAM) or Phase-Change Memory (PCM)?



(Expanded from Bob Brennan, "Berkeley Next-Generation Memory Discussion," January, 2014 for DRAM & STT-MRAM; Pi-Feng Chiu added others)

Next Step: New possibilities

2010 | 2015 | 2020?

DDR DDR DDR

HDD HDD HDD

Tape

High Performance Tier — DRAM DRAM DRAM

High Capacity Tier

Intelligent Flash Tier

HDD HDD HDD HDD HDD

Higher Performance Tier — DRAM DRAM DRAM

Persistent Performance Tier

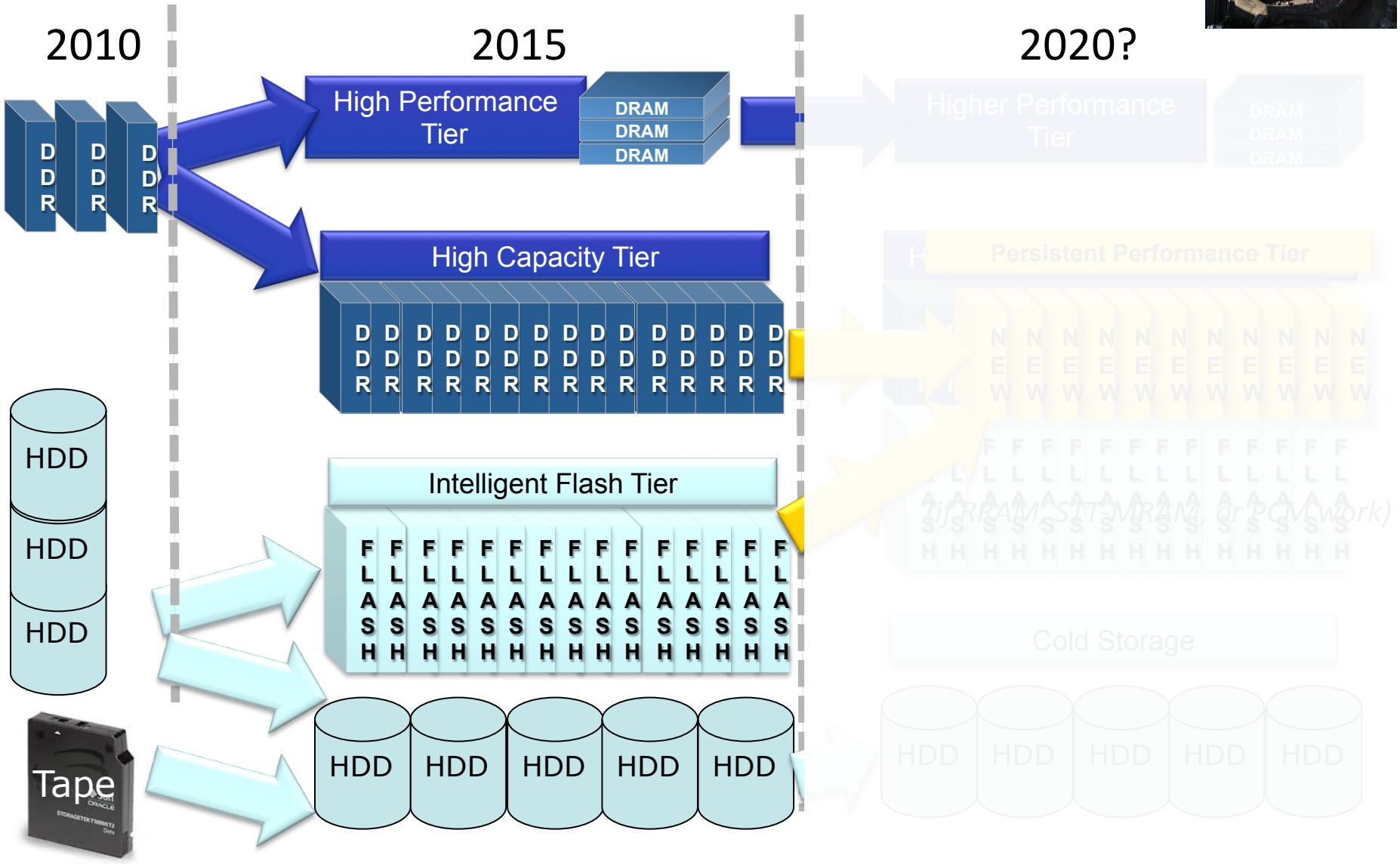
(if RRAM, STT-MRAM, or PCM work)

Cold Storage

HDD HDD HDD HDD HDD

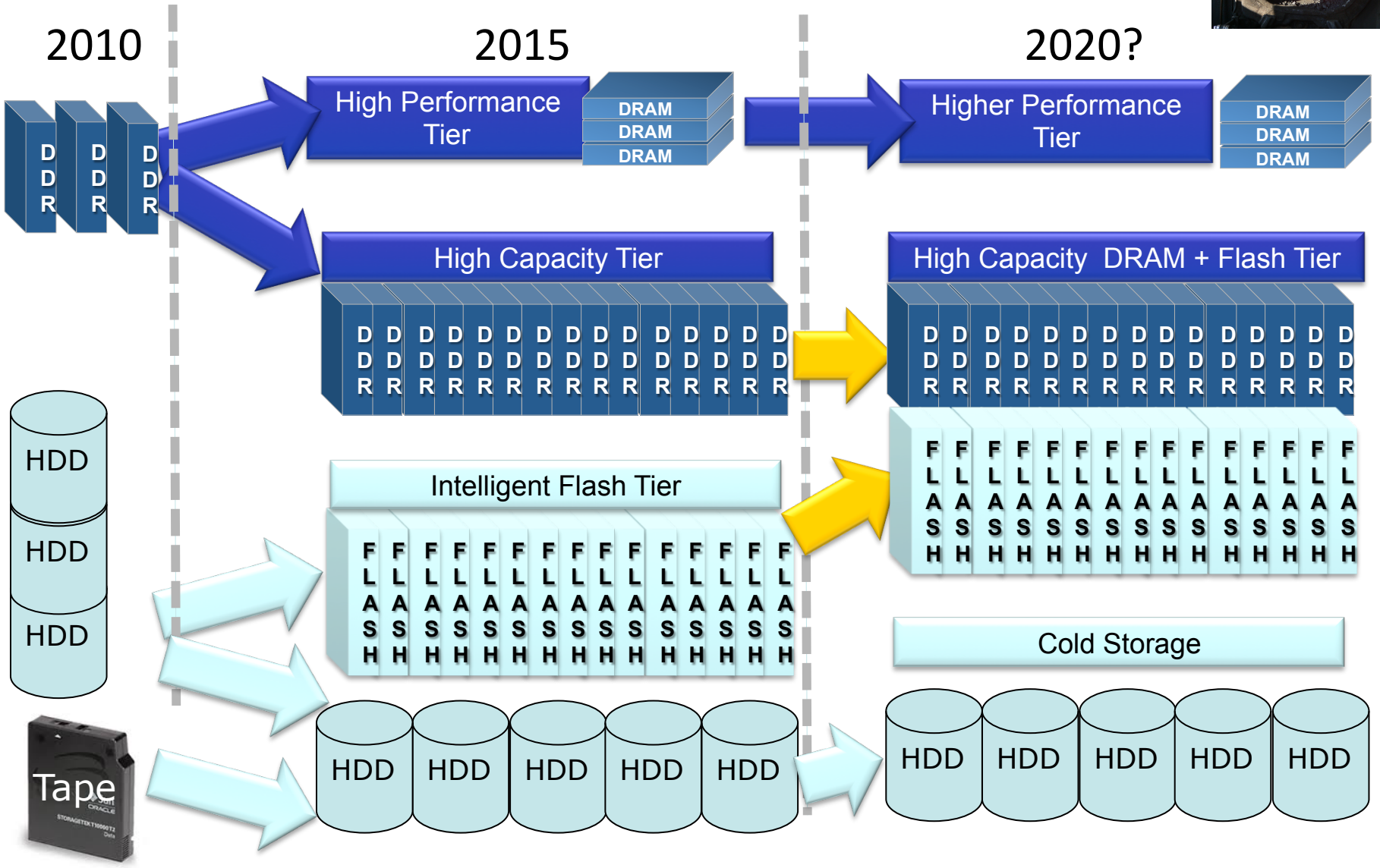
[Revised, based on slide from Bob Brennan, "Berkeley Next-Generation Memory Discussion," Jan. 2014
© Samsung]

Next Step: New possibilities



[Revised, based on slide from Bob Brennan, "Berkeley Next-Generation Memory Discussion," Jan. 2014
© Samsung]

Next Step: New possibilities



[Revised, based on slide from Bob Brennan, "Berkeley Next-Generation Memory Discussion," Jan. 2014
© Samsung]

Next Step: New possibilities

2010 2015 2020?

DDR DDR DDR

High Performance Tier — DRAM DRAM DRAM

Higher Performance Tier — DRAM DRAM DRAM

High Capacity Tier — DDR DDR DDR DDR DDR DDR DDR DDR DDR DDR DDR DDR DDR DDR DDR DDR

Persistent Performance Tier — NEW NEW NEW NEW NEW NEW NEW NEW NEW NEW NEW NEW

(if RRAM, STT-MRAM, or PCM work)

Intelligent Flash Tier — FLASH FLASH FLASH FLASH FLASH FLASH FLASH FLASH FLASH FLASH FLASH FLASH FLASH FLASH FLASH

HDD HDD HDD

Tape

HDD HDD HDD HDD HDD

Cold Storage

HDD HDD HDD HDD HDD

[Revised, based on slide from Bob Brennan, "Berkeley Next-Generation Memory Discussion," Jan. 2014
© Samsung]

Security: Old vs. New Conventional Wisdom

- Old CW: Given cyber and physical security at borders of WSC, don't normally need encryption inside WSC
- New CW: Given attacks on WSCs by disgruntled employees, industrial spies, foreign and even *domestic* government agencies, data must be encrypted whenever transmitted or stored inside WSCs

Moore's Law: Old vs. New Conventional Wisdom

- Old CW: Moore's Law, each 18-month technology generation, transistor performance/energy improves, cost/transistor decreases
- New CW: generations slowing to 3 year -> 5+ year, transistor performance/energy slight improvement, cost/transistor increases!

2020: Moore's Law has ended for logic, SRAM, & DRAM (Maybe 3D Flash & new NVM continues?)

WSC Hardware Design: Old vs. New Conventional Wisdom

- Old CW: Build WSC from cheap Commercial Off-The-Shelf (COTS) Components, which run LAMP stack
  - Microprocessors, racks, NICs, rack switches, array switches, …

- New CS: Build WSC from custom components, which support SOA, tail tolerance, fault tolerance/ detection/recovery/prediction, …
  - Custom high radix switches, custom racks and cooling, System on a Chip (SoC) integrating processors & NIC

Why Custom Chips in 2020?

- Without transistor scaling, improvements in system capability have to come above transistor-level
  - More specialized hardware
- WSCs proliferate @ $100M/WSC
  - Economically sound to divert some $ if yield more cost-performance-energy effective chips
- Good news: when scaling stops, custom chip costs drop
  - Amortize investments in capital equipment, CAD tools, libraries, training, … over decades vs. 18 months
- New HW description languages supporting parameterized generators improve productivity and reduce design cost
  - E.g., Stanford Genesis2; Berkeley's Chisel, based on Scala

ARM versus x86?

- Much debate about relative merits of these two ISAs in 2nd/3rd generation WSC design
- Real important differences are in:
  - Quality of COTS implementations
  - Can build custom chips around ARM IP, not Intel

- Or maybe people just propose to use ARM in servers to make Intel drop their x86 server chip prices.

What is an ISA?

- The most important interface in a computing system
- Third-party software/tools developers need to build to an ISA standard
- In an open computing future, why have a proprietary ISA standard?
- Are x86 and ARM v8 innovative interface designs?

- ISA should be an open industry standard!

Berkeley RISC-V ISA

www.riscv.org

- A new completely open ISA
  - Already runs GCC, Linux, glibc, LLVM, …
  - RV32, RV64, and RV128 variants for 32b, 64b, and 128b address spaces defined
- Base ISA only 40 integer instructions, but supports compiler, linker, OS, etc.
- Extensions provide full general-purpose ISA, including IEEE-754/2008 floating-point
- Comparable ISA-level metrics to other RISCs
- Designed for extension, customization,
- Eight 64-bit silicon prototype implementations completed at Berkeley so far (45nm, 28nm)

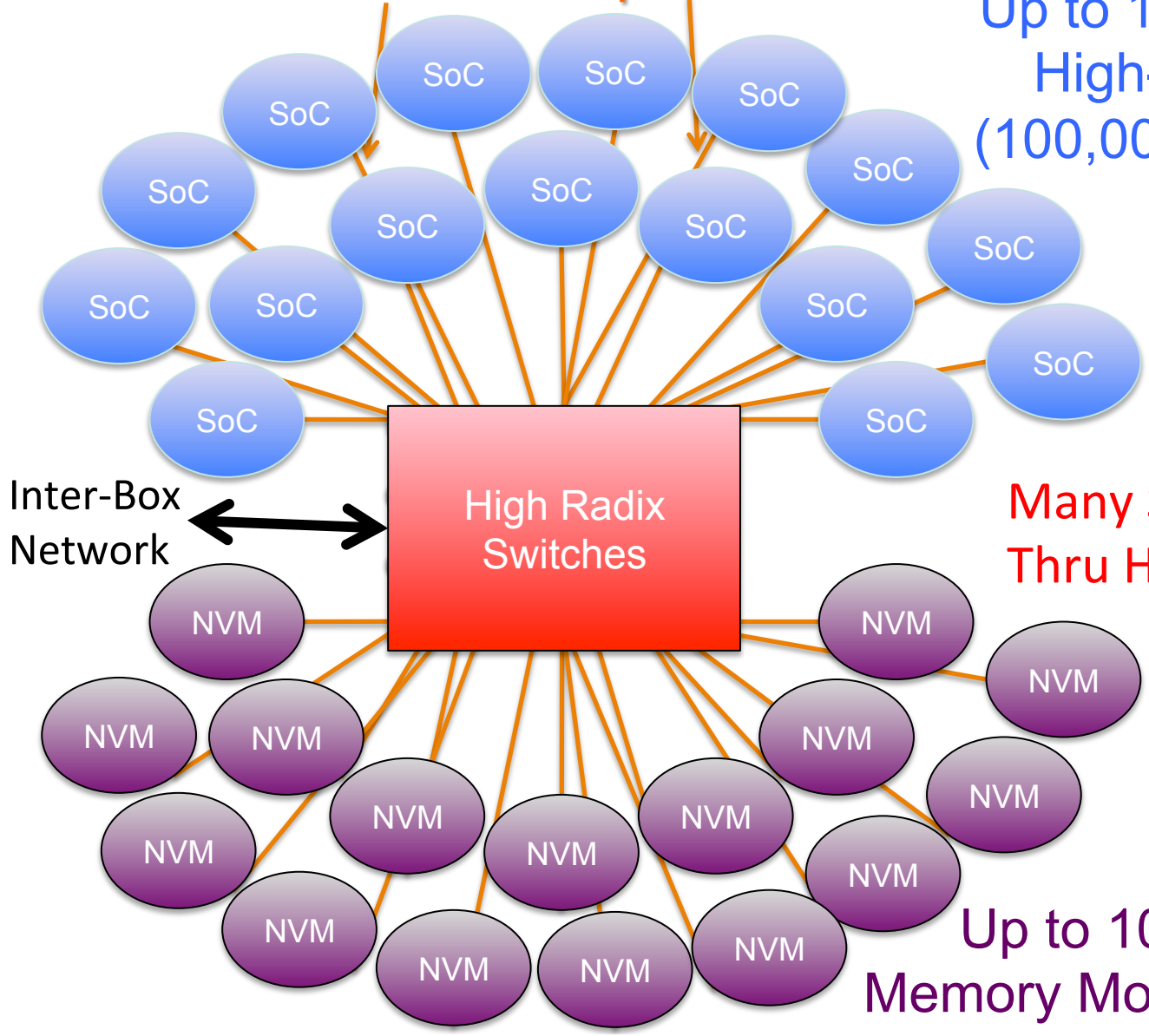
Open-Source & WSCs

- 1st generation WSC leveraged open-source software
- 2nd generation WSC also pushing open-source board designs (OpenCompute), OpenFlow API for networking
- 3rd generation – open-source chip designs?
  - We plan to release FireBox WSC chip generator

FireBox Overview

1 Terabit/sec optical fibers

Up to 1000 SoCs +
High-BW Mem
(100,000 core total)



Many Short Paths
Thru High-Radix Switches

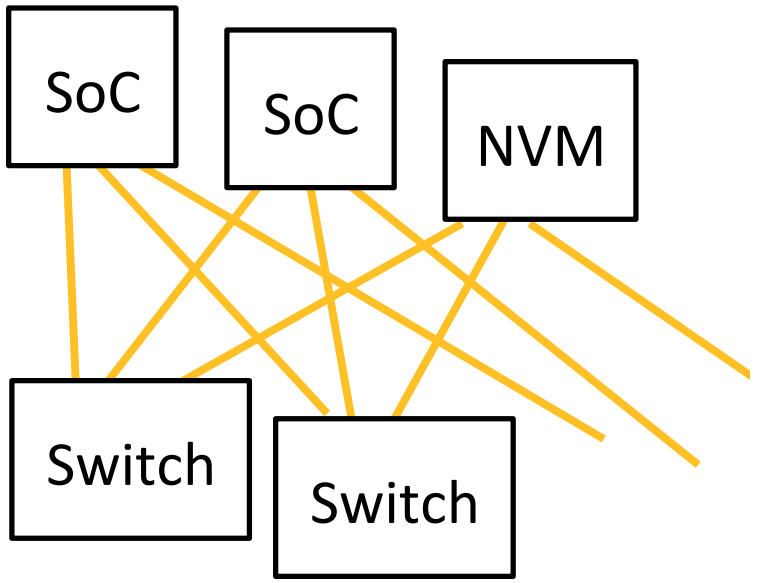
Up to 1000 NonVolatile
Memory Modules (100PB total)

FireBox Big Bets

- Reduce OpEx, manage units of 1,000+ sockets
- Support huge in-memory (NVM) databases directly
- Massive network bandwidth to simplify software
- Re-engineered software/processor/NIC/network for low-overhead messaging between cores, low-latency high-bandwidth bulk memory access
- Data always encrypted on fiber and in bulk storage
- Custom SoC with hardware to support above features
- Open-source hardware generator to allow customization within WSC SoC template

Photonic Switches

- Monolithically integrated silicon photonics with Wave-Division Multiplexing (WDM)
  - A fiber carries 32 wavelengths, each 32Gb/s, in each direction
  - Off-chip laser optical supply, on-chip modulators and detectors
- Multiple radix-1000 photonic switch chips arranged as middle stage of Clos network (first and last Clos stage inside sockets)
- 2K endpoints can be configured as either SoC or NVM modules
- In Box, all paths are two fiber hops:
  - Electrical-photonic at socket
  - One fiber hop socket-to-switch
  - Photonic-electrical at switch
  - Electrical packet routing in switch
  - Electrical-photonic at socket
  - One fiber hop switch-to-socket
  - Photonic-electrical at socket



Inter-Socket Communication: Old vs. New Conventional Wisdom

- Old CW: Shared memory simplifies programming of large server
  - Don't need to figure out what to send
- New CW: Message passing simplifies programming of large server
  - Message passing more predictable than cache-coherent shared memory
  - Failure conditions also easier to reason about
  - Single consistent programming abstraction:
    - Message passing WSC to WSC
    - Message passing cluster-to-cluster within WSC
    - Message passing node-to-node within cluster
  - Better match to Service-Oriented Architecture (SOA)

- Message passing won the war: MPI, RPC, RDMA, TCP/IP

FireBox SoC Highlights

- ~100 (homogenous) cores per SoC
  - Simplify resource management, software model
- Each core has vector processor++ (>> SIMD)
  - "General-purpose specialization"
- Uses RISC-V instruction set
  - Open source, virtualizable, modern 64-bit RISC ISA
  - GCC/LLVM compilers, runs Linux
- Cache coherent on-chip so only need one OS per SoC
  - Core/outer caches can be split into local/global scratchpad/cache to improve tail tolerance
- Compress/Encrypt engine so reduce size for storage and transmission yet always encrypted outside node
- Implemented as parameterized Chisel chip generator
  - Easy to add custom application accelerators, tune architectural parameters

FireBox NVM Module

- Each module is 100TB of future NVM
- E.g., 1K chips * 1Tb/chip
- Possible packaging, 32 stacks * 32 chips/stack
- Parallel photonic I/O to base chips in each stack supports multiple 1Tb/s fiber links
- Redundancy across cells, banks, chips, stacks, modules
- Large number of parallel chips, with multiple banks/chip and replicas reduce NVM queuing delays
- Highly parallel with huge network bandwidth to support background parallel rebuild if failure detected

FireBox Hardware Highlights

- 8-32 DRAM chips on interposer for high BW
  - 32Gb chips give 32-128GB DRAM capacity/node
  - 500GB/s DRAM bandwidth
- Message Passing is RPC: can return/throw exceptions
  - ≈20 ns overhead for send or receive, including SW
  - ≈100ns latency to access Bulk Memory: ≈2X DRAM latency
- Error Detection/Correction on Bulk Memory
- No Disks in Standard Box; special Disk Boxes instead
  - Disk Boxes for Cold Storage
- ≈50 KW/box
- ≈35KW for 1000 sockets
  - 20W for socket cores, 10W for socket I/O, 5W for local DRAM
- ≈15KW for Bulk NVRAM + Crossbar switch
  - 10^{-12} joule/bit transfer => Terabit/sec/Watt

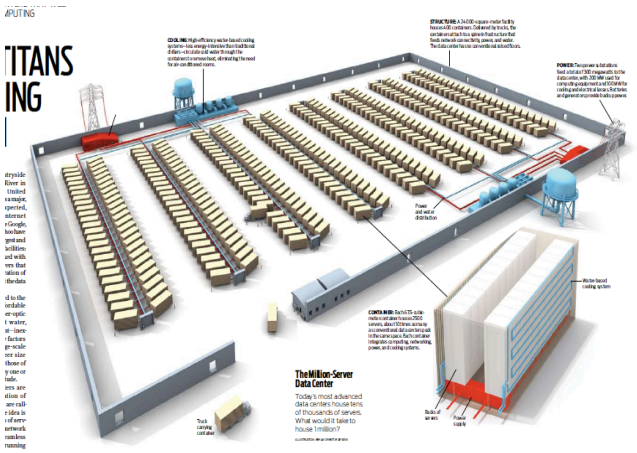
10 Open Questions for FireBox

- Techniques to improve tail latency at high utilization?
- What is In-Box network protocol? Bridge to TCP/IP?
- What is the API/ABI for RPC? Instructions vs. Library?
- Role of virtualization? Virtual Box vs. Virtual Service?
- OK that encryption => processing inside SoCs vs. processing near Bulk Memory?
- How much failure detection/recovery/prediction/… in cores, caches, network? HW versus SW?
- How to securely manage 100PB of NVM shared across organizations?
- Do other organizations want to collaborate in design and construction of FireBox prototypes?

ASPIRE
UC Berkeley

DIABLO – Datacenter-In-A-Box at LOw-cost 1

- Build a "wind tunnel" for datacenter designs using FPGAs
  - Simulate O(10,000) nodes: each is capable of running real software
  - Simulate O(1,000) datacenter switches (all levels) in detail & accurate timing
  - Simulate O(100) seconds in target
  - Runtime configurable architectural parameters (link speed/latency, host speed)
- 1000X hardware accelerator for software simulator
  - Run in parallel at 1/1000th real time vs. 1/1,000,000th real time

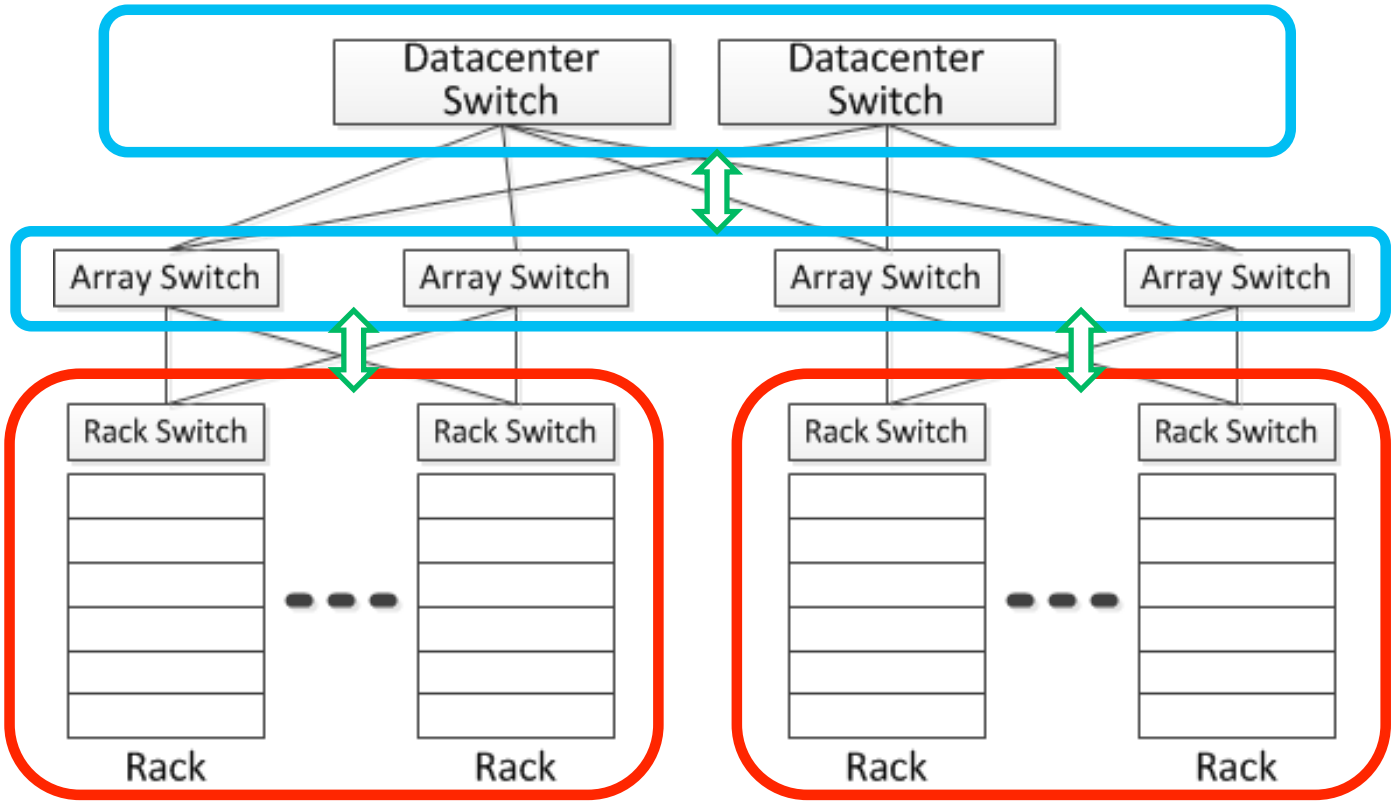


Executing real instructions, moving real bytes in the network!

DIABLO 1 Cluster Prototype

- 6 BEE3 boards total 24 Xilinx Virtex5 FPGAs
  - Physical characteristics:
    - Full-custom FPGA implementation with many reliability features @ 90/180 MHz
    - Memory: 384 GB (128 MB/node), peak bandwidth 180 GB/s
    - Connected with SERDES @ 2.5 Gbps
    - Host control bandwidth: 24 x 1 Gbps control bandwidth to the switch
    - Active power: ~1.2 kWatt
  - Simulation capacity
    - 3,072 simulated servers in 96 simulated racks, 96 simulated switches
    - 8.4 B instructions / second

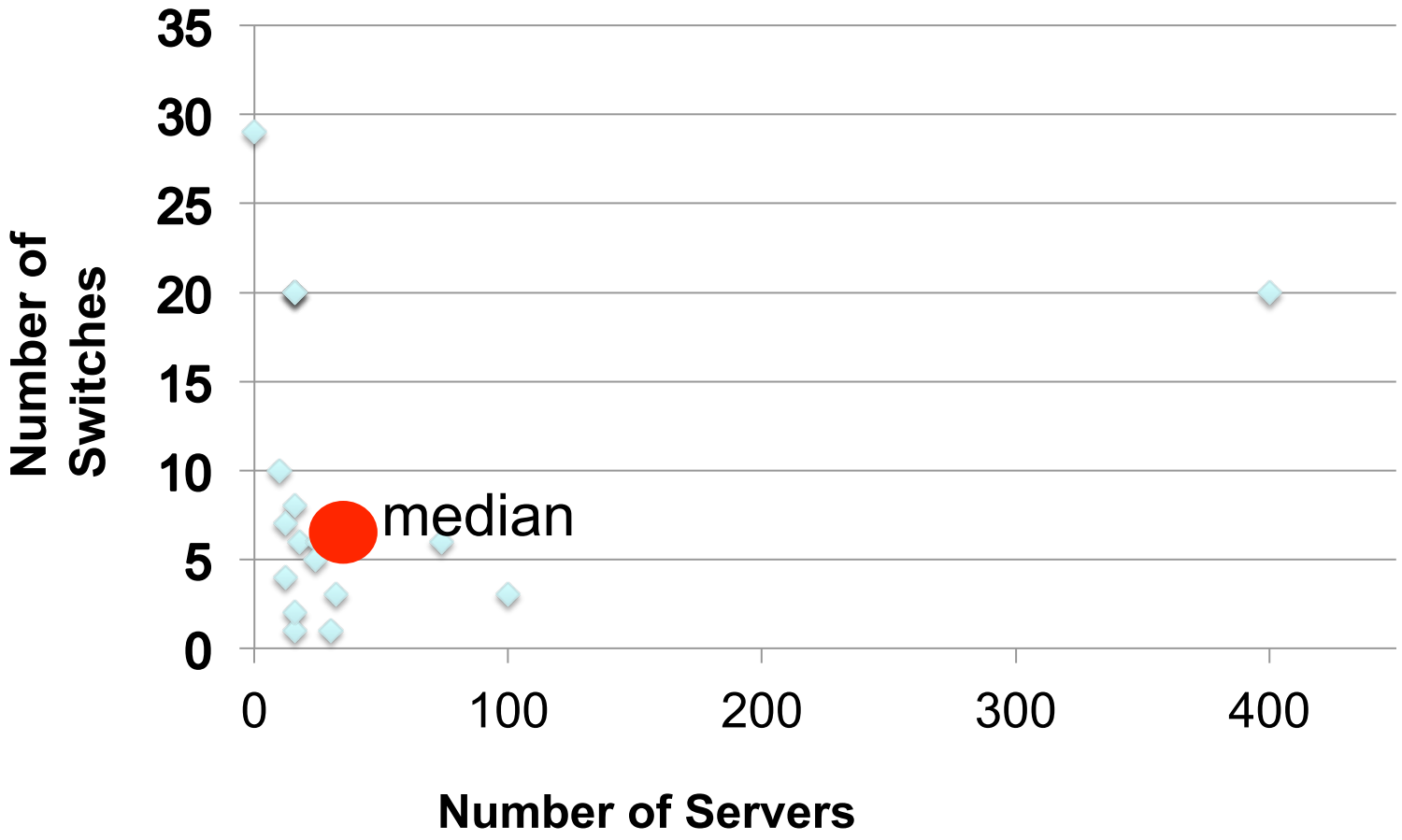
Mapping a Datacenter to DIABLO 1



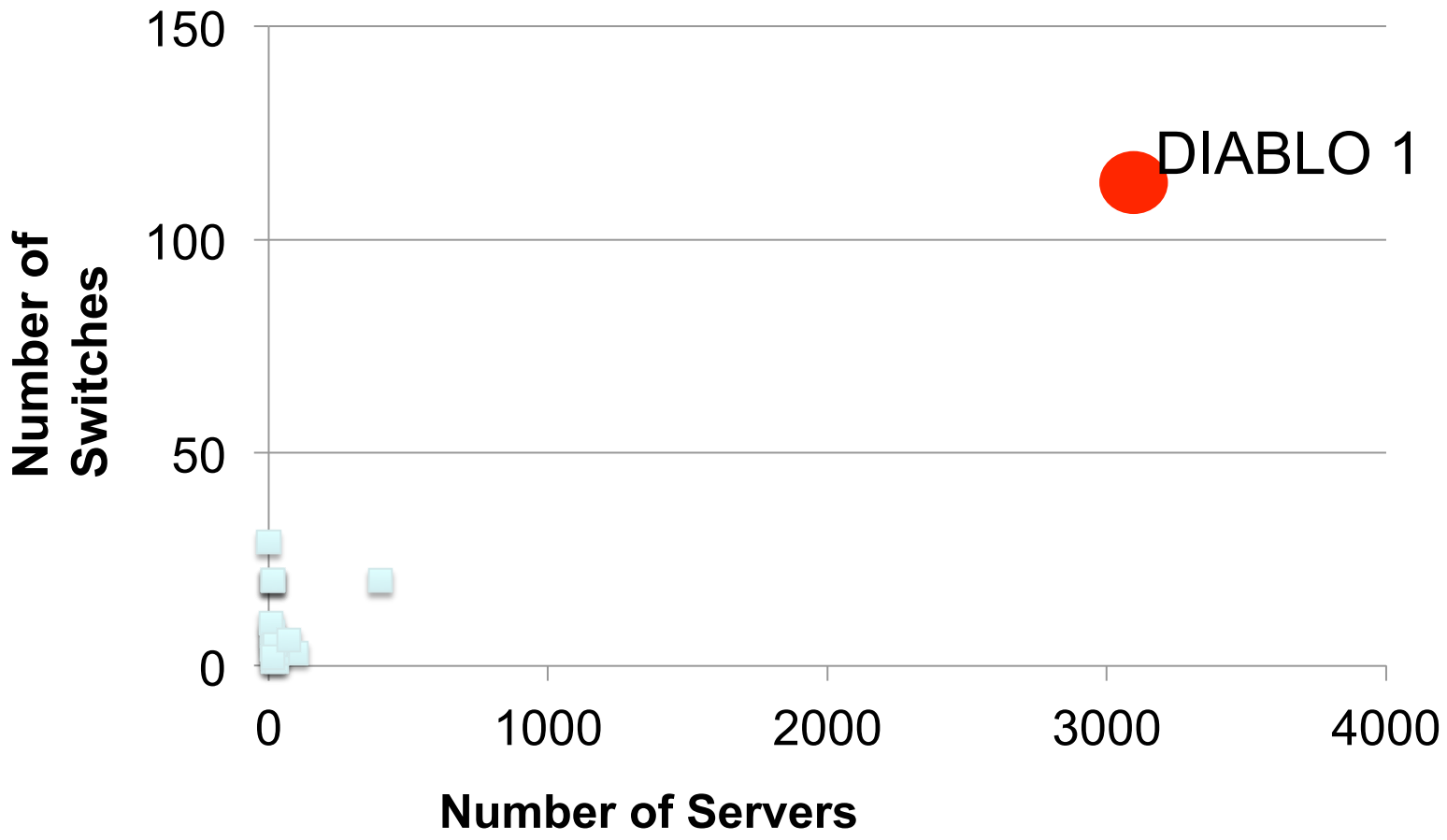
- NOT an FPGA computer, but instead an FPGA-accelerated simulator
- FAME-7 multithreaded functional plus timing models for all components
- Two types of FPGA: 1) Server+Rack Switch, 2) Array+Datacenter Switch
- Connecting multiple FPGAs using multi-gigabit transceivers according to desired target topology

ASPIRE
UC Berkeley

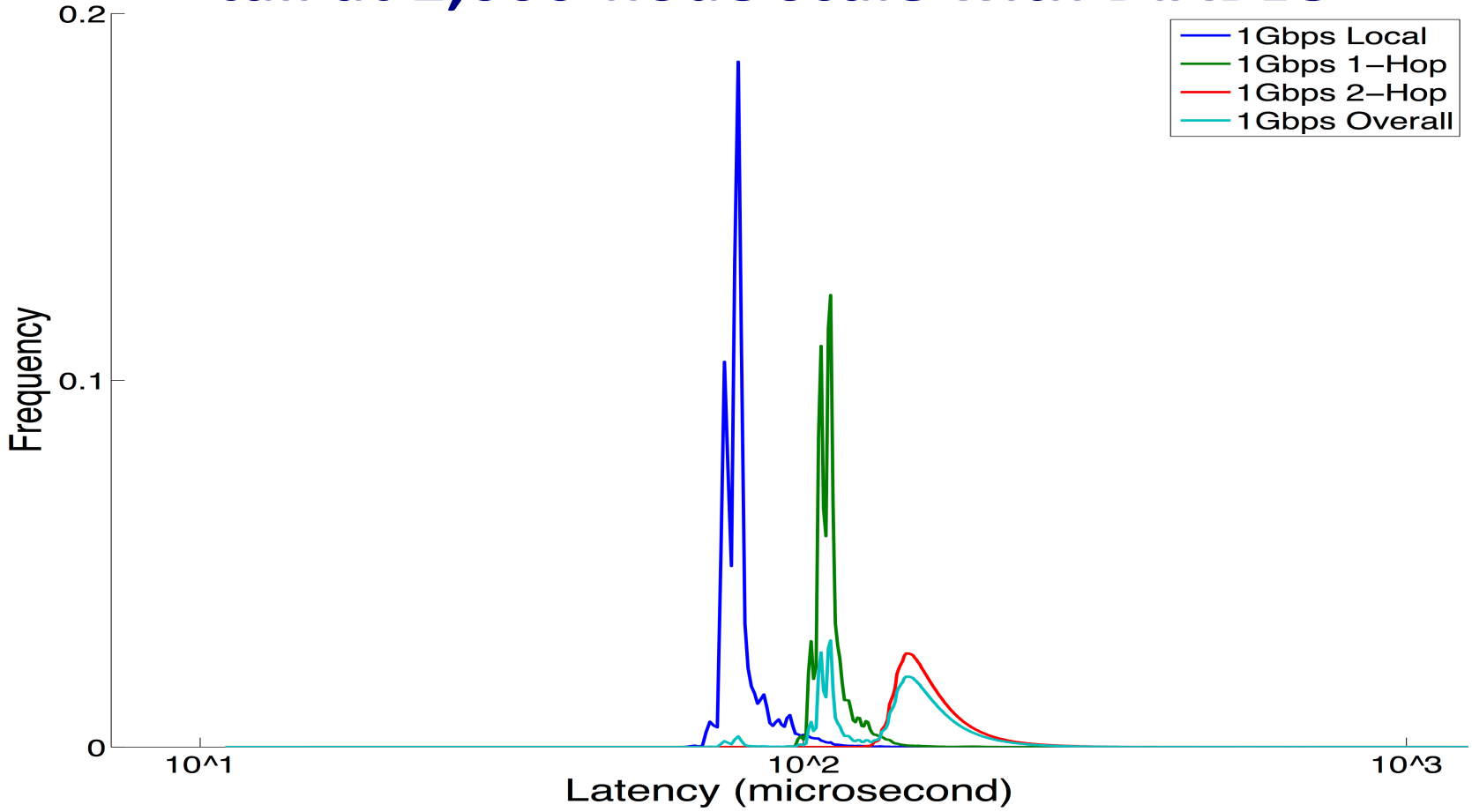
Testbed Scale in Recent SIGCOMM Papers



DIABLO 1 vs. Others



Reproducing memcached latency long tail at 2,000-node scale with DIABLO

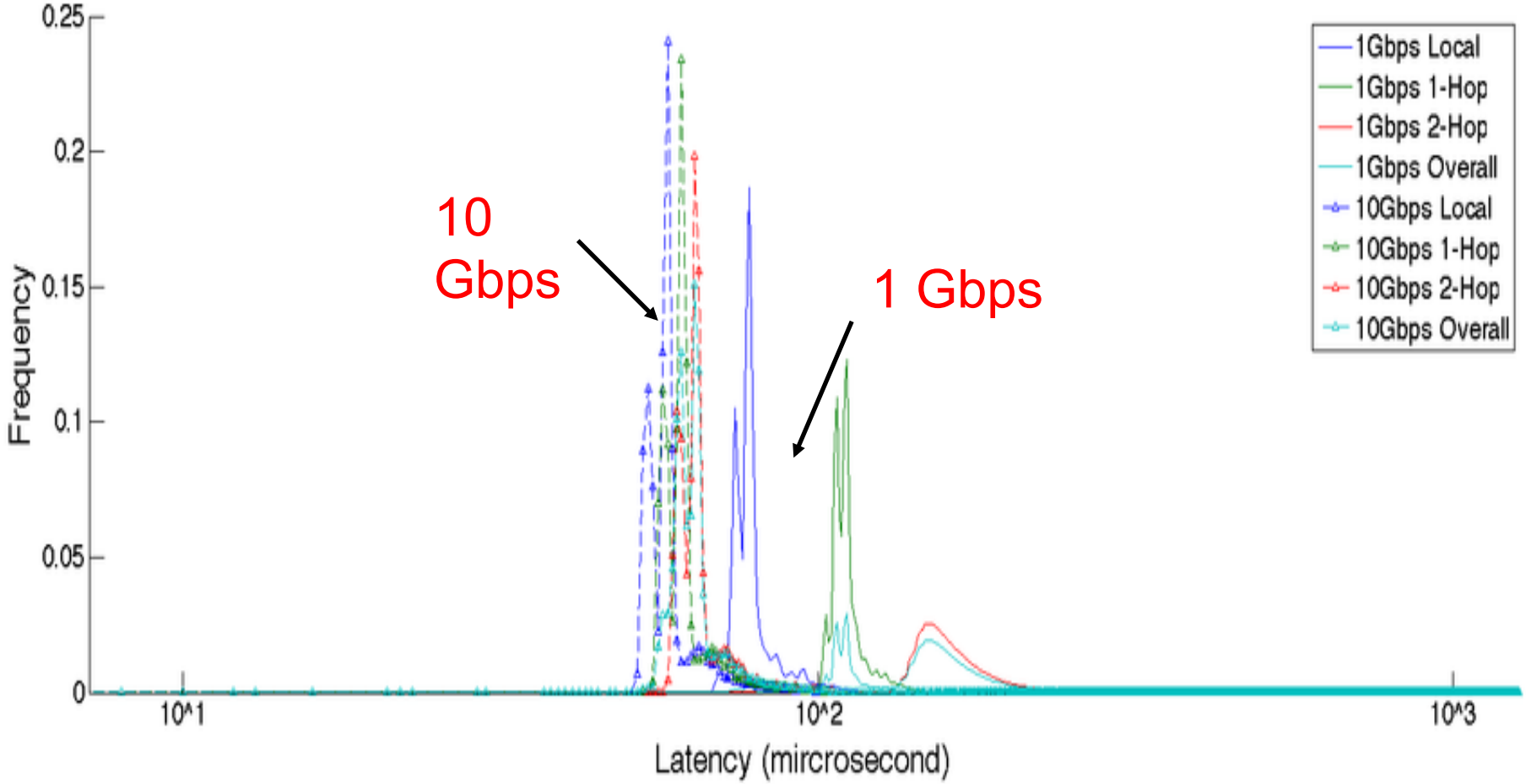


- Most requests complete ~100µs, but some 100x slower
- More switches -> greater latency variations

[Luiz Barroso "Entering the teenage decade in warehouse-scale computing" FCRC'11]

ASPIRE
UC Berkeley

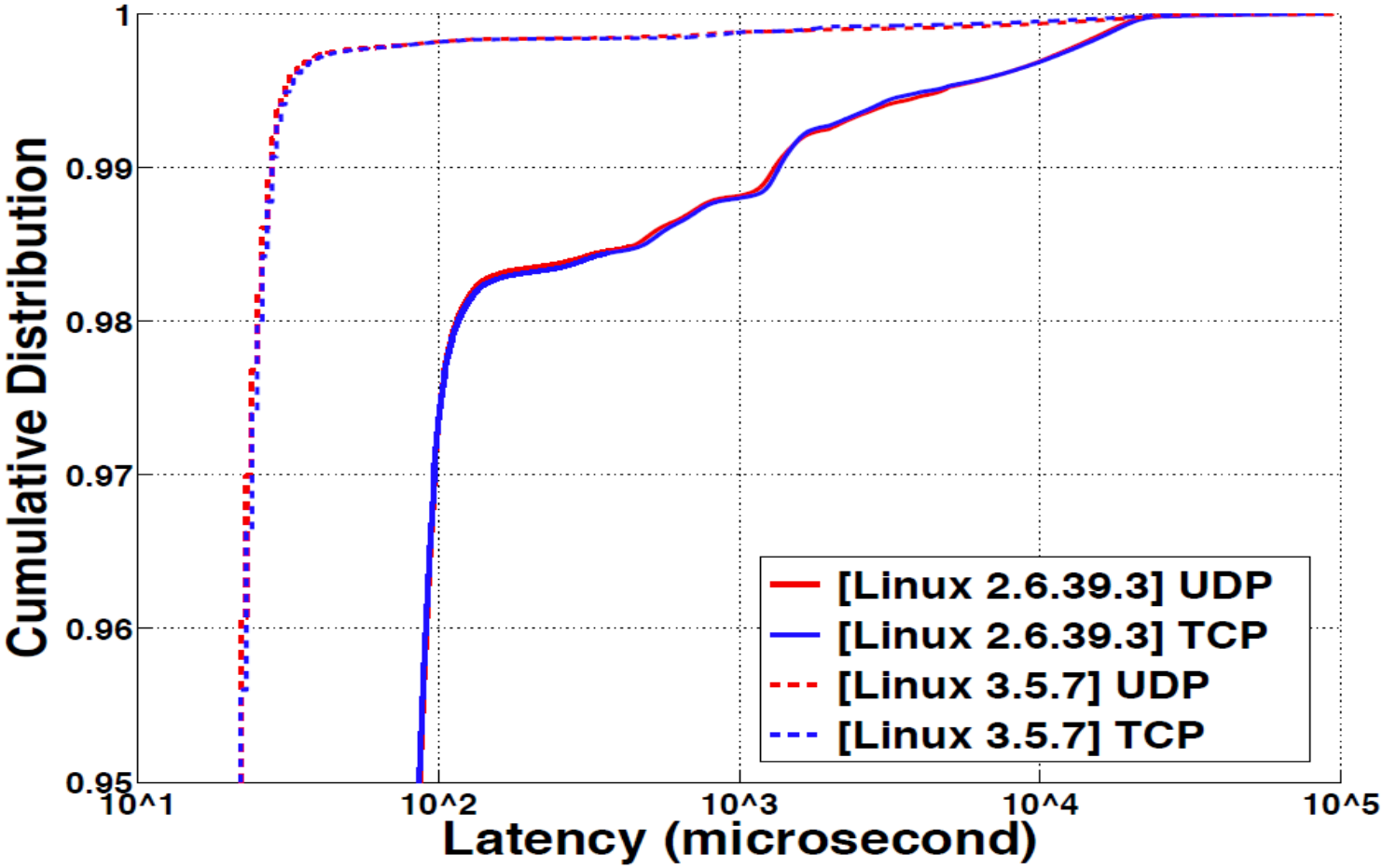
Adding 10x Better Interconnect



- Low-latency 10Gbps switches improve access latency but only <2x
- The software stack dominates!

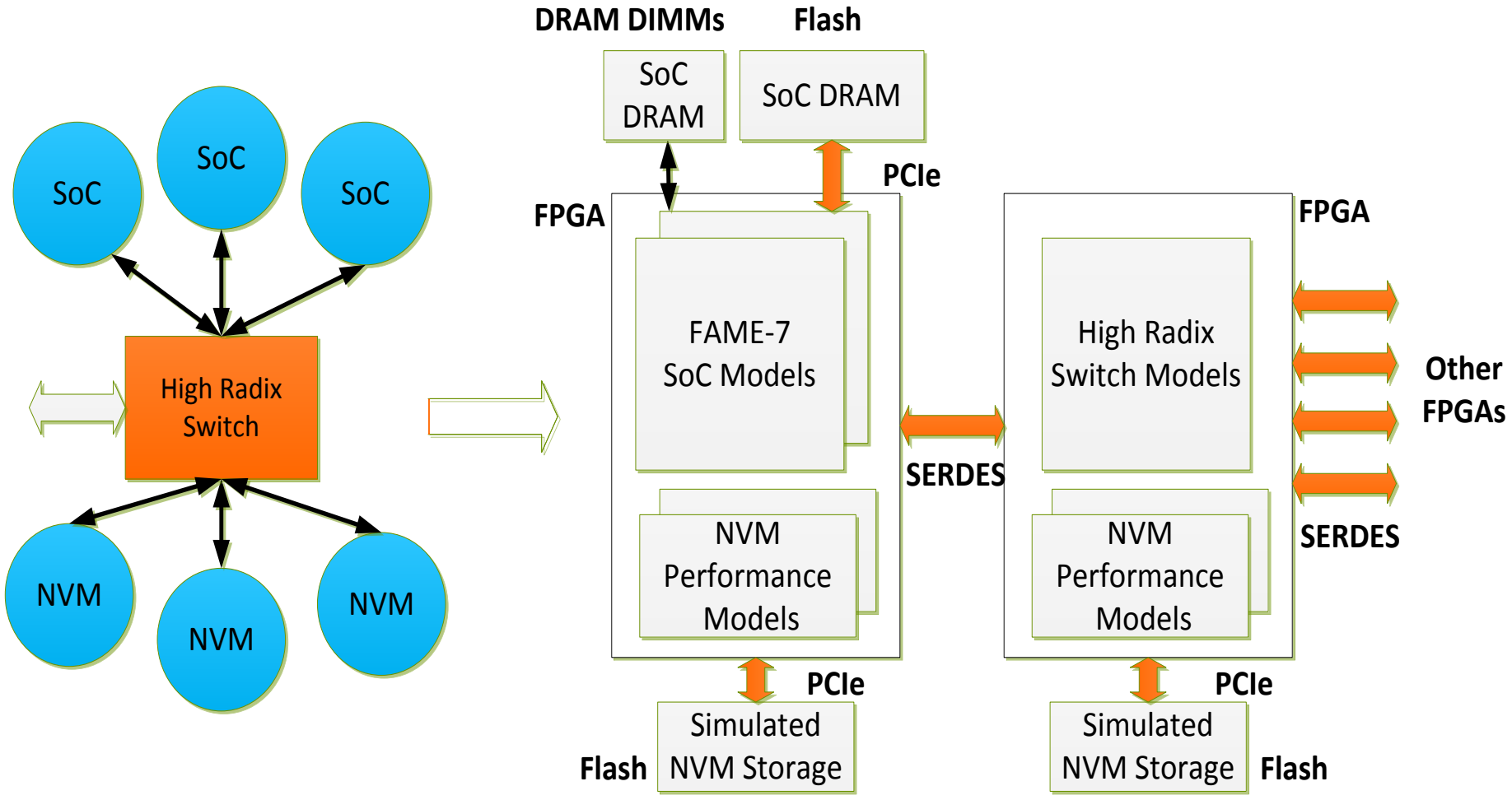
ASPIRE
UC Berkeley

Impact of kernel versions on 2,000-node *memcached* latency long tail



- Better implementations in newer kernel helps the latency long tail

Mapping FireBox to DIABLO 2



FireBox Target Design

DIABLO 2

FireBox Conclusion

- FireBox is a 2020 WSC Server
  - Raise abstraction for programming, control, management, …
  - Custom chips, optical fiber, solid-state store
  - Supercomputer

- FireBox is ambitious for academia; others collaborate?

Acknowledgements

Help from ASPIRE outsiders:

- Bob Brennan, Samsung
- Eric Brewer, UCB/Google
- Zhangxi Tan, UCB/startup
- Amin Vahdat, Google

ASPIRE Sponsors:

- Founding: DARPA, SRC, Intel
- Affiliates: Google, Nokia, NVIDIA, Oracle, Samsung

Help from ASPIRE insiders:

- Elad Alon
- Scott Beamer
- Pi-Feng Chiu
- Adam Izraelevitz
- Kurt Keutzer
- Forrest Iandola
- Eric Love
- Martin Mass
- Bora Nikolic
- Frank Nothaft
- Colin Schmidt
- Vladimir Stojanovic
- Andrew Waterman