The CAVA Computer: Exceptional Parallelism and Energy Efficiency

Peter Hsu, Ph.D.
peter.x.hsu@oracle.com
peterhsu@cs.wisc.edu

Oracle Labs, California, USA

June 29, 2015

Presented at the 2nd UC Berkeley RISC-V Workshop

ORACLE Hardware and Software Engineered to Work Together

The following is intended to provide some insight into a line of research in Oracle Labs. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. Oracle reserves the right to alter its development plans and practices at any time, and the development, release, and timing of any features or functionality described in connection with any Oracle product or service remains at the sole discretion of Oracle. Any views expressed in this presentation are my own and do not necessarily reflect the views of Oracle.

Problem

- Designing a new computer is expensive
  - > $100 million, > 4 years
- But 80% is exactly the same every time
  - Like cars: you need electric windows, power brakes, power steering, heated seats, cup holders, navigation systems, sound system…
  - Getting it right requires skill and experience, but is taken for granted and does not command much of a premium
  - Getting it wrong, on the other hand, is a commercial disaster
- So why don’t we just standardize a design and open-source it?
  - All the patents in this 80% area have already expired anyway
  - Then everybody can concentrate on adding value in the remaining 20%

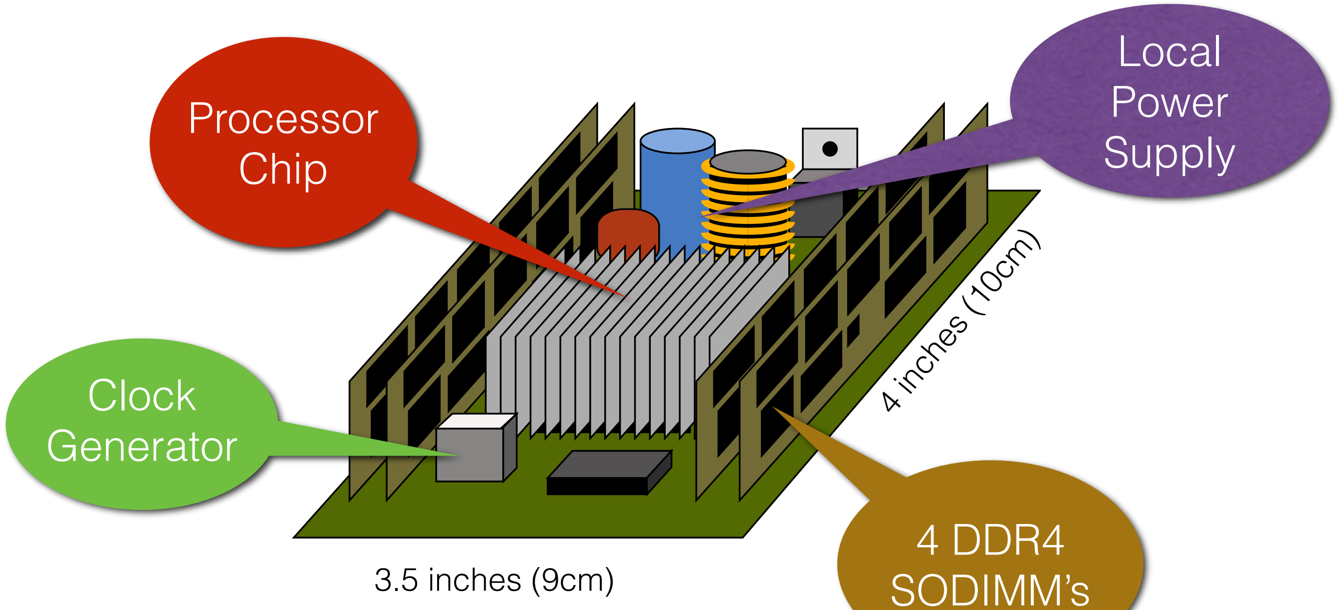
Idea

- UC Berkeley has a RISC-V Project
  - Free ISA, toolchain, processor cores, *Firebox* system definition, hardworking students, industrial participation, international cooperation, *momentum*
- My idea: define a commercially viable open-source computer that can be the “80%” generic part of what everybody wants to build
  - Helps UCB and other Universities focus their research on a credible target
  - Helps the whole industry get a leg up on a new kind of system architecture
  - Lets Oracle focus development money on the 20% that differentiates its product

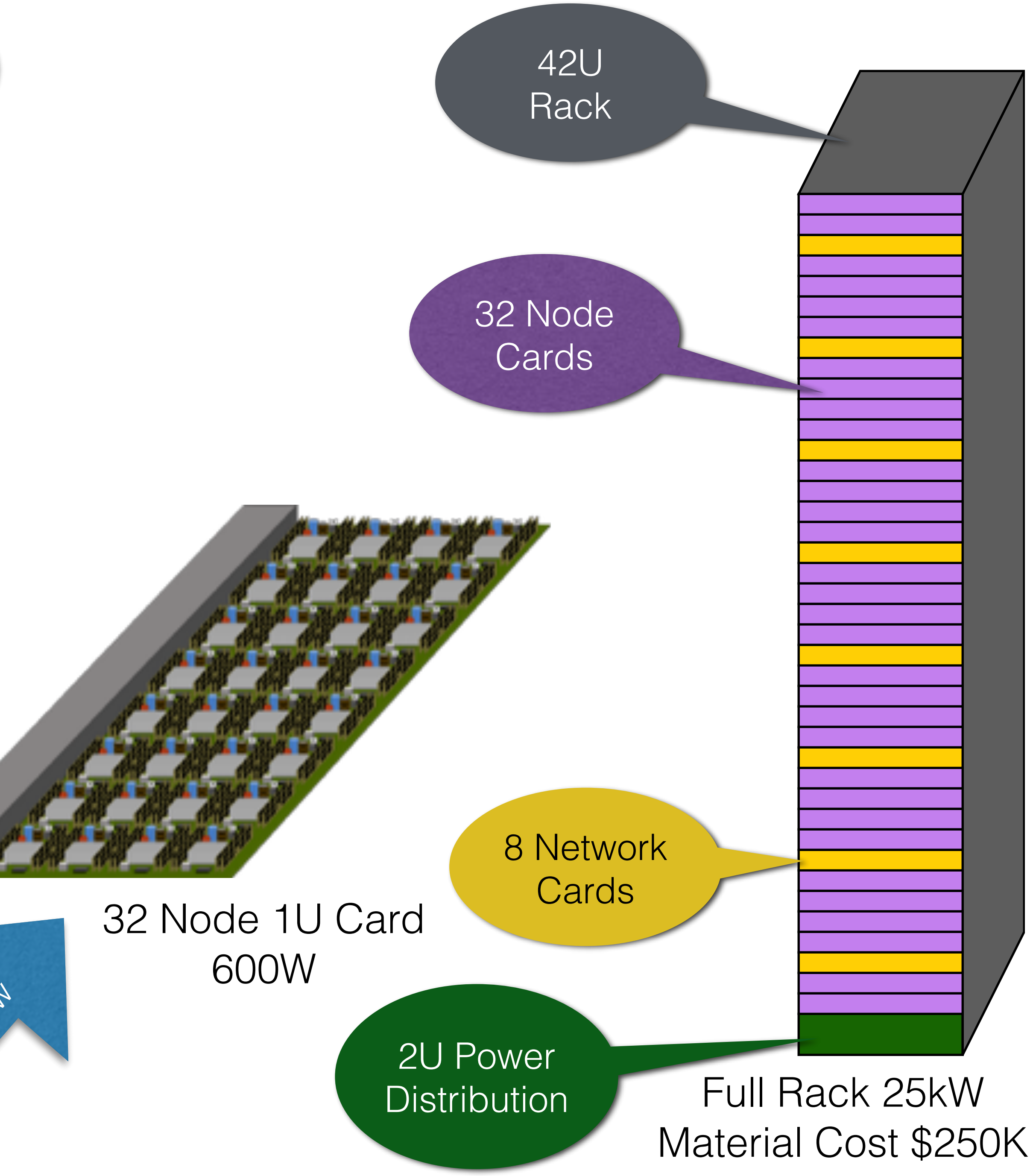
The Big Picture

- CAVA is not RAPID
  - Oracle Labs has a RAPID project, which is building an energy-efficient parallel computer based on philosophy similar to that which I will discuss in this talk
  - They have their own next-generation roadmap, which is not related to anything I will be talking about here
- CAVA is a new research initiative
  - I believe there are many areas in the energy-efficient parallel computing arena that Oracle Labs could benefit from collaborative research with Universities
  - Focusing on designing a particular system, the CAVA computer, is a way to organize the research so that we have something specific to evaluate
- The primary deliverables are a simulation environment and results (for Oracle) and published papers (for the students)
  - I would like the results to influence the direction of Oracle's RAPID project, of course
  - I would like if possible to build a prototype too (I'm very fond of hardware toys 😋)

1024-Node Cluster in a Rack



18W Node



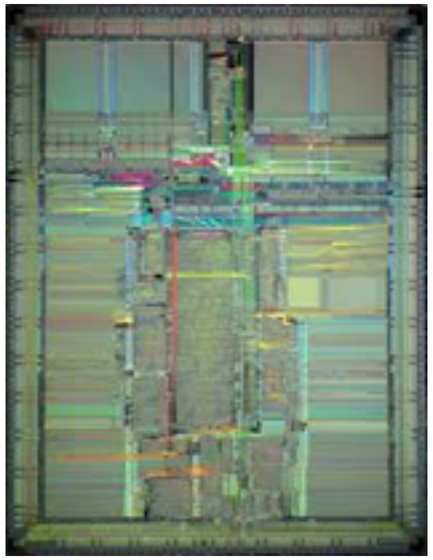
32 Node 1U Card
600W

Full Rack 25kW
Material Cost $250K

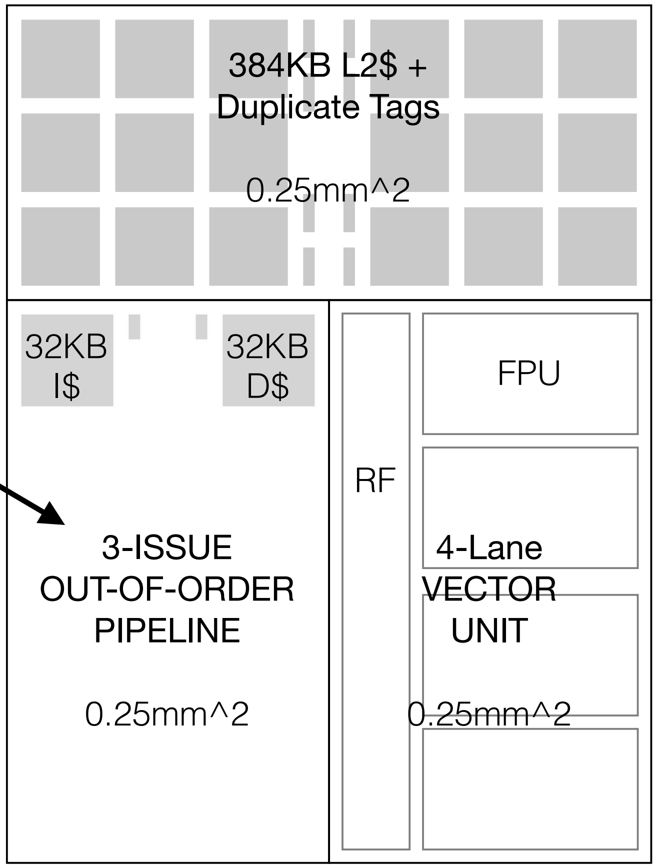
- DDR4 3200 = 25.6GB/s per 64-bit channel
- "Each dynamic instruction requires one byte of memory bandwidth†"
- 100GB/s memory BW -> 100GIPS sustained throughput

†Rule of thumb for database applications dating from the 1970's (e.g. IBM mainframes)

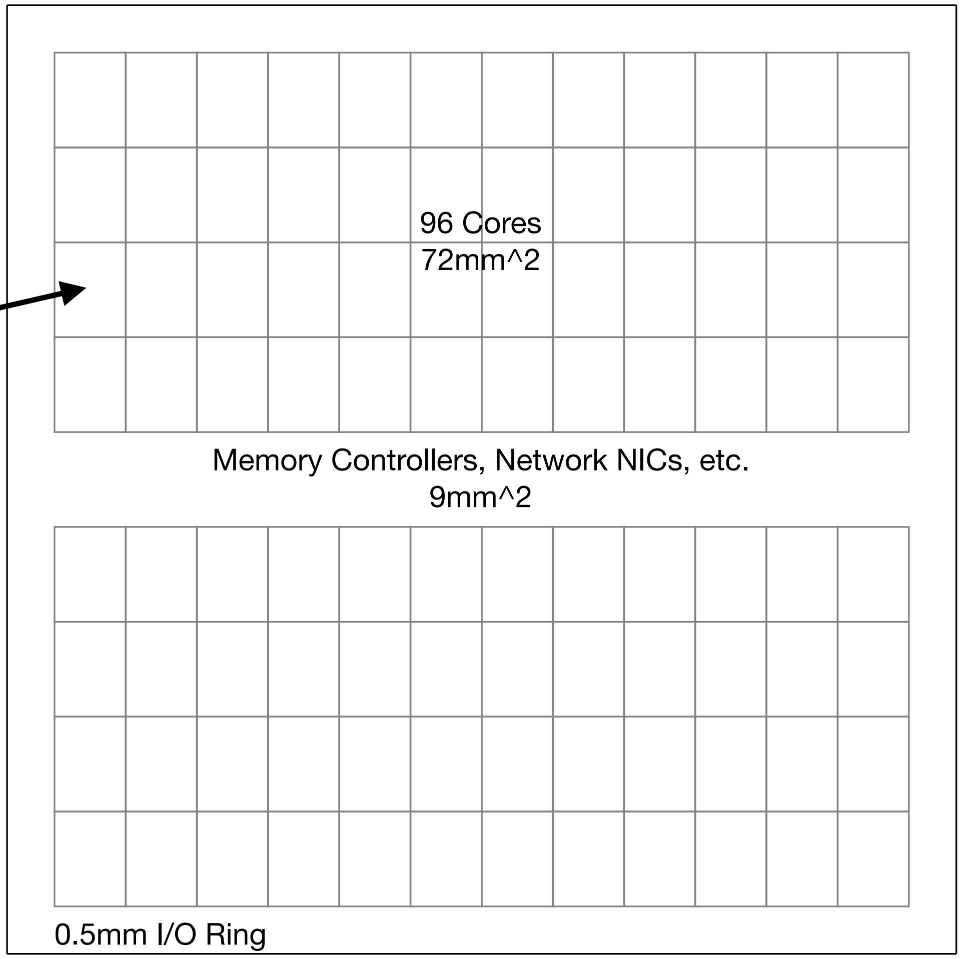
96-Core 10nm Chip



MIPS R10K 1996
0.35um 200MHz
Spec gcc 1.0 IPC



Frequency
Target
1.04GHz

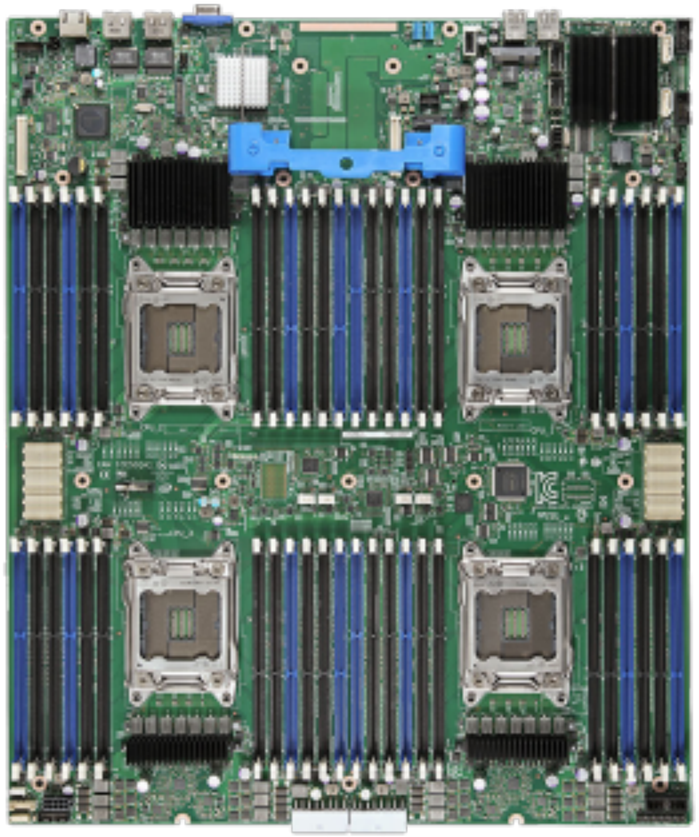


100mm^2

- 3-issue OOO 600K gates, 32KB I+D, 12-cycle L2
  - 300mm^2 in 350nm -> 0.24mm^2 in 10nm
- TSMC 10nm FinFET SRAM density 1.5MB/mm^2 (with ECC, redundancy, BIST, duplicate tags)
  - [http://forums.anandtech.com/showthread.php?p=37099401]

Sanity Check

- Four-socket 1U server with hypothetical 10nm x86 chips
  - 16 cores, 3.5GHz, 1.4 IPC = 78 GIPS
  - 92W TDP (5.75W/core), estimate $750/chip
- 1TB total, 256GB memory/socket, 9W/channel x 4 channels = 36W
  - 8 ranks (4 double-sided DIMMs) per channel, 2W active rank, 1W x 7 inactive ranks
- 1U board with 4 processors = 312GIPS vs. 3200 for CAVA
  - 1/10th throughput at same power, cost, form factor, amount of memory
  - This is a “typical 1U server,” not the most energy-efficient x86 thing imaginable!
  - Does not include network, I/O, mass storage. There things costs the same in both systems.



Modest Expectation for VU

- Vector lane = FPU + fancy integer ALU + table lookup
    - Integer ALU handles filtering, run-length decoding and other database functions
    - Table lookup functionality is programmable extension of division/SQRT ROM
    - VU directly fed from second level cache
- Estimate 4 vector lanes + generous register file = size of out-of-order scalar processor core (0.25mm^2)
- Trying to get 3X performance for 2X area
    - 4 vector lanes = 4 IPC peak (8 if single-precision)
    - But at most only utilized 50% of time
    - Scalar unit 1 IPC + vector unit 2 IPC = 3 IPC
- My philosophy: anything less is not worth doing; anything more is a bonus
- Note: chip performance primarily limited by memory bandwidth, which scalar unit can already fully consume, so vector unit is accelerator only for programs with lots of locality.

Network

- CLOS topology just one possibility
    - Target: lightweight, within-rack network
    - Explore Flattened Butterfly, Dragonfly, SlimFly, etc.
    - Leverage high-radix switches, bounded size network
- High integration necessary for low cost, energy efficiency
    - Initially electrical (25Gb/s links)
    - Migrating to integrated silicon photonics (100+Gb/s)
- Use 2 planes for higher bandwidth, redundancy/fault-tolerance
    - Bandwidth = 5GB/s per node (initially)
    - Photonics = 20GB/s per node (20% of memory BW)

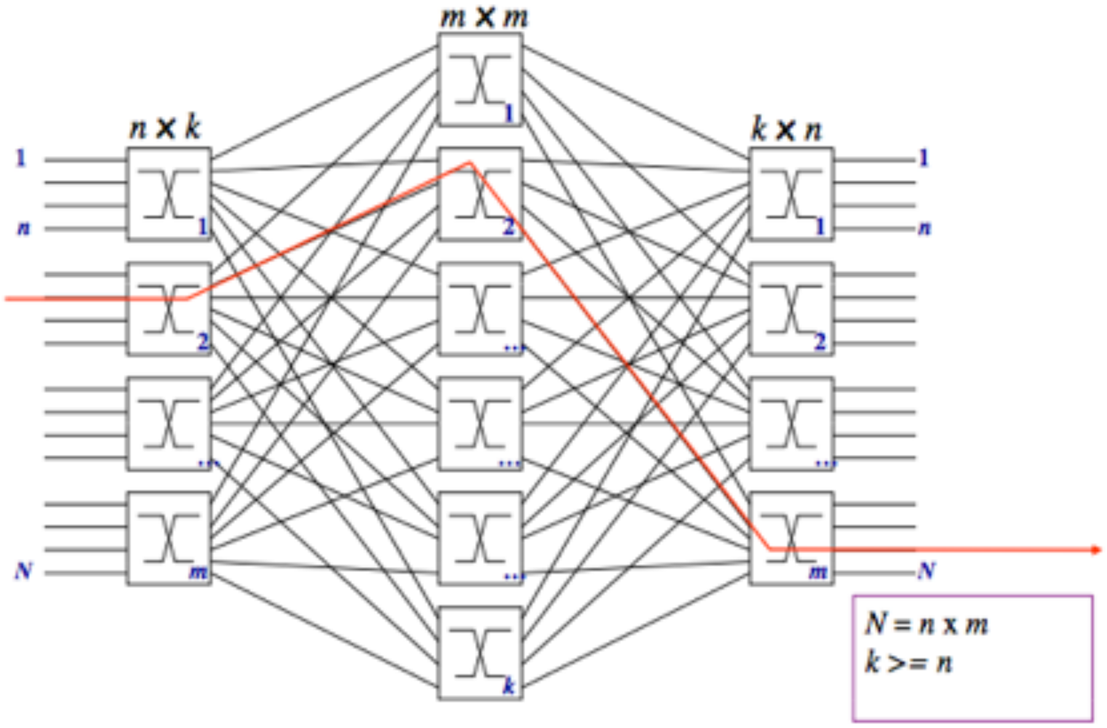
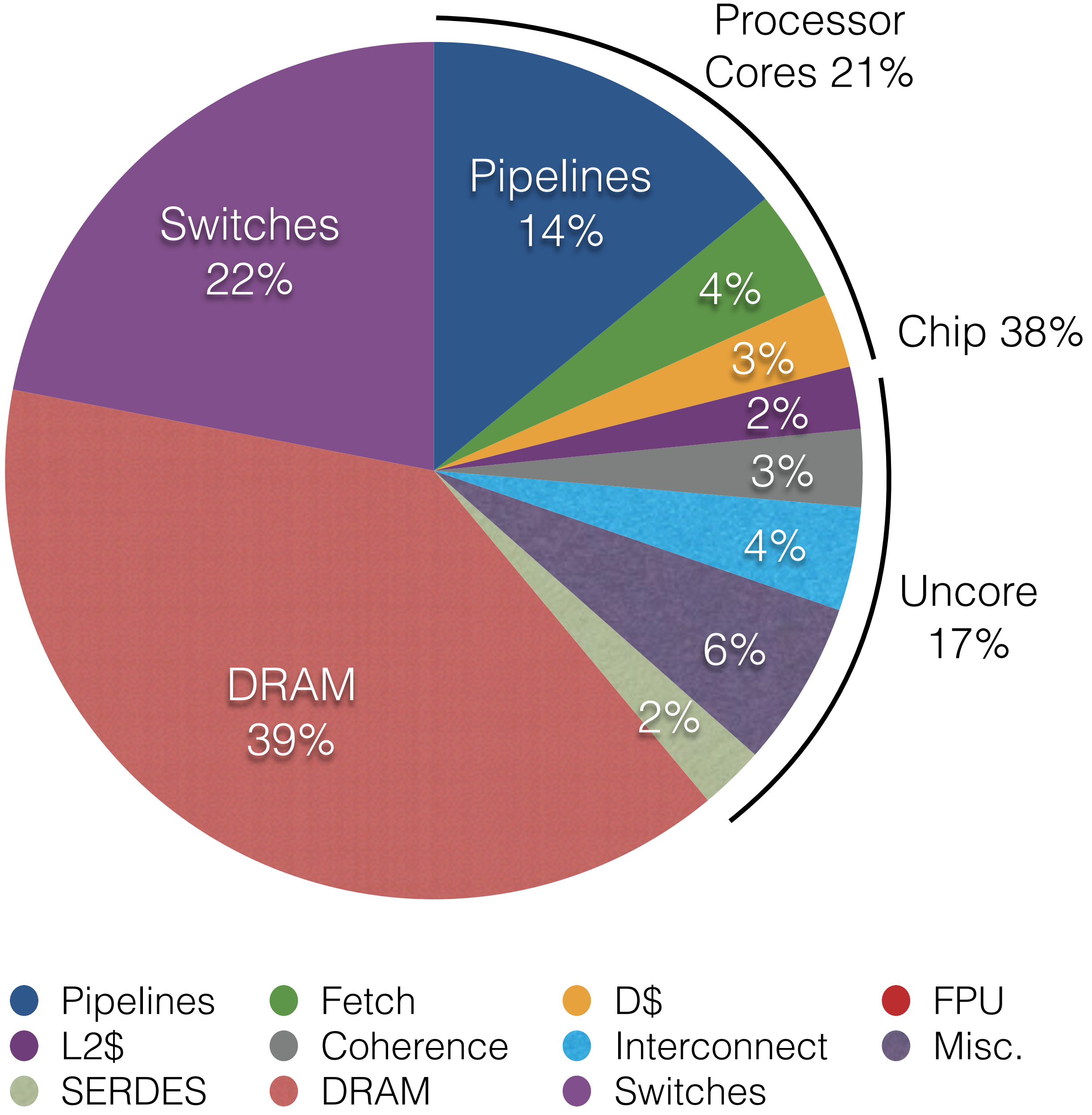


Figure 1: 3-stage clos network.

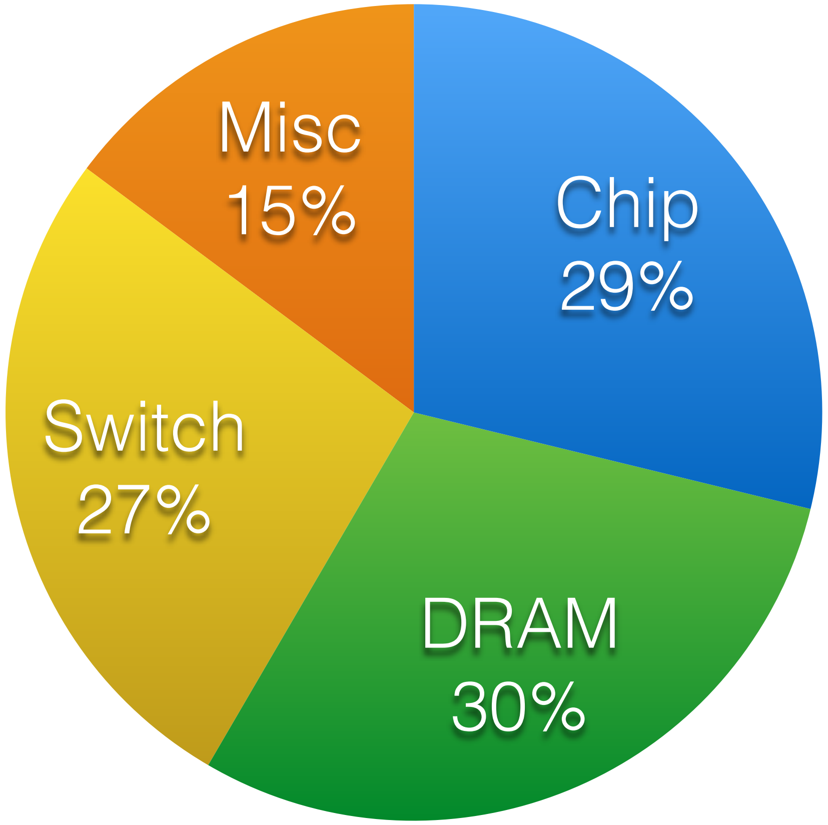
CLOS Topology
N=1024
n=m=k=32
96 chips
each 32x32

Power (Scalar Case)

- 8W processor chip (core + L1$ = 50pJ)
- 8W for 4 DDR4 channels @ 3.2Gb/s (220mW per DRAM)
- 24W per switch chip (128x128 ports, 10pJ/bit SERDES + 8W internal)
- 1024 nodes = 21kW
- Power supply efficiency 85%
- Rack = 25kW
  - High end of air-cool comfort zone
- Throughput 102 TIPS (sustained @ 1.0 IPC)
  - 245 pJ/Instruction



Costs



- Processor chips $72K ($70 x 1024)
  - A 12-inch wafer might cost $12K and makes over 600 10x10mm^2 dies. Even assuming a very conservative 50% yield, that's $40 per good die. A very expensive plastic BGA package costs $20. Testing, etc. might add $10.
- 32TB DRAM $74K (Assume $72/node @ $2/GB by 2019)
  - [$6/GB 4/29/2015 dramexchange.com]
- Switch chips $67K ($350/chip x 192)
- Misc $37K
- Total $250K material cost
  - List price < $1M

- Other interesting configurations:
  - 2U 64-node system: $12K, 1200W, household 110V plug, street price $25K?
  - 4U 128-node system: $25K, 2400W, standard 220V plug, street price $50K?
  - Quarter rack (10U) 320-node system: $50K, 6kW
  - Half-rack (20U) 640-node system: $110K, 12kW

Costs

- Processor chips $72K ($70 x 1024)
  - A 12-inch wafer might cost $12K and makes over 600 10x10mm^2 dies. Even assuming a very conservative 50% yield, that's $40 per good die. A very expensive plastic BGA package costs $20. Testing, etc. might add $10.
- 32TB DRAM $74K (Assume $72/node @ $2/GB by 2019)
  - [$6/GB 4/29/2015 dramexchange.com]
- Switch chips $67K ($350/chip x 192)
- Misc $37K
- Total $250K material cost
  - List price < $1M

New standard of cost effectiveness:
$5/thread equipment purchase
17¢/thread/year electricity cost

Average US price 12¢ per kilowatt-hour

- Other interesting configurations:
  - 2U 64-node system: $12K, 1200W, household 110V plug, street price $30K?
  - 4U 128-node system: $25K, 2400W, standard 220V plug, street price $60K?
  - Quarter rack (10U) 320-node system: $50K, 6kW
  - Half-rack (20U) 640-node system: $110K, 12kW

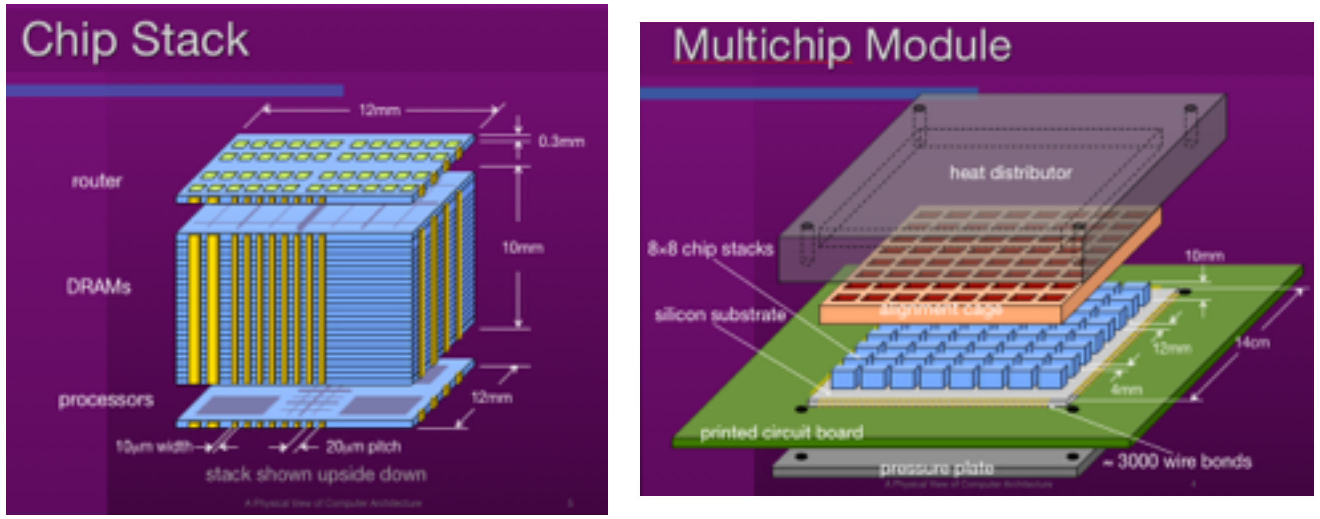
Long History of Parallel Computers

- 1964 ILLIAC-IV 64 cpus
- 1982 Denelcore HEP 16 cpus, 800 threads
- 1983 Goodyear Aerospace MPP 16K cpus
- 1985 nCUBE 1024 32-bit cpus
- 1986 Thinking Machine CM-1 64K cpus
- 1992 Stanford DASH 64 cpus
- 1992 MasPar MP-2 16K 32-bit cpus
- 1992 Kendal Square KSR-1 128 cpus
- 1993 MIT J-Machine 512 cpus
- 2011 SeaMicro SM10000 2K Atom cores
- 2012 Calxeda 480 ARM cores

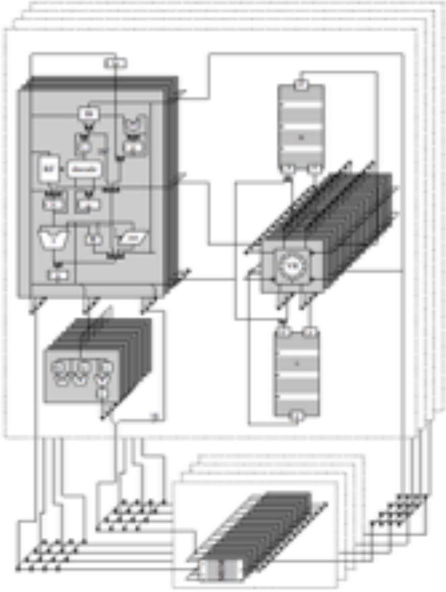
- Making one commercially successful is a lot harder than it looks! Need to be:
  - Cost effective
  - Energy efficient
  - Physically compact
  - *Not too hard to program!*
- Must achieve <u>all</u> of these things simultaneously to have a chance
  - Competition is a cluster of off-the-shelf energy-efficient x86 servers
- We realized early on (2010) that neither using Intel Atom chips nor existing ARM cores would suffice

Thinking About This a While

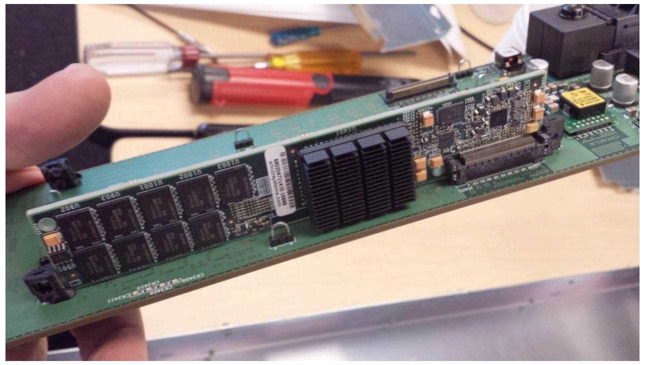
Brainstorming with
Toshiba DRAM Engineers
1995



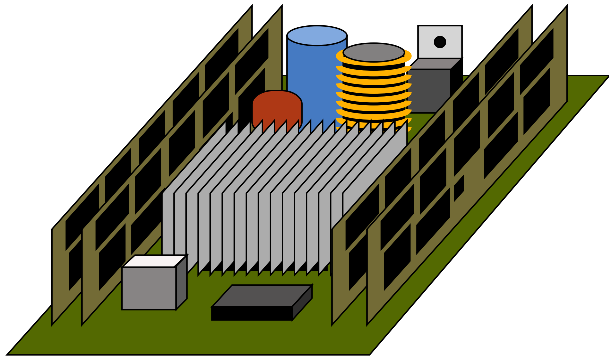
“Computer Architecture From
Many Perspectives,” UPC,
Barcelona, Spain, 2001



“DRAM+CPU: Build It, They Will
Come,” keynote, IEEE Computer
Element Vail Workshop, Vail,
Colorado, 26 June 2005

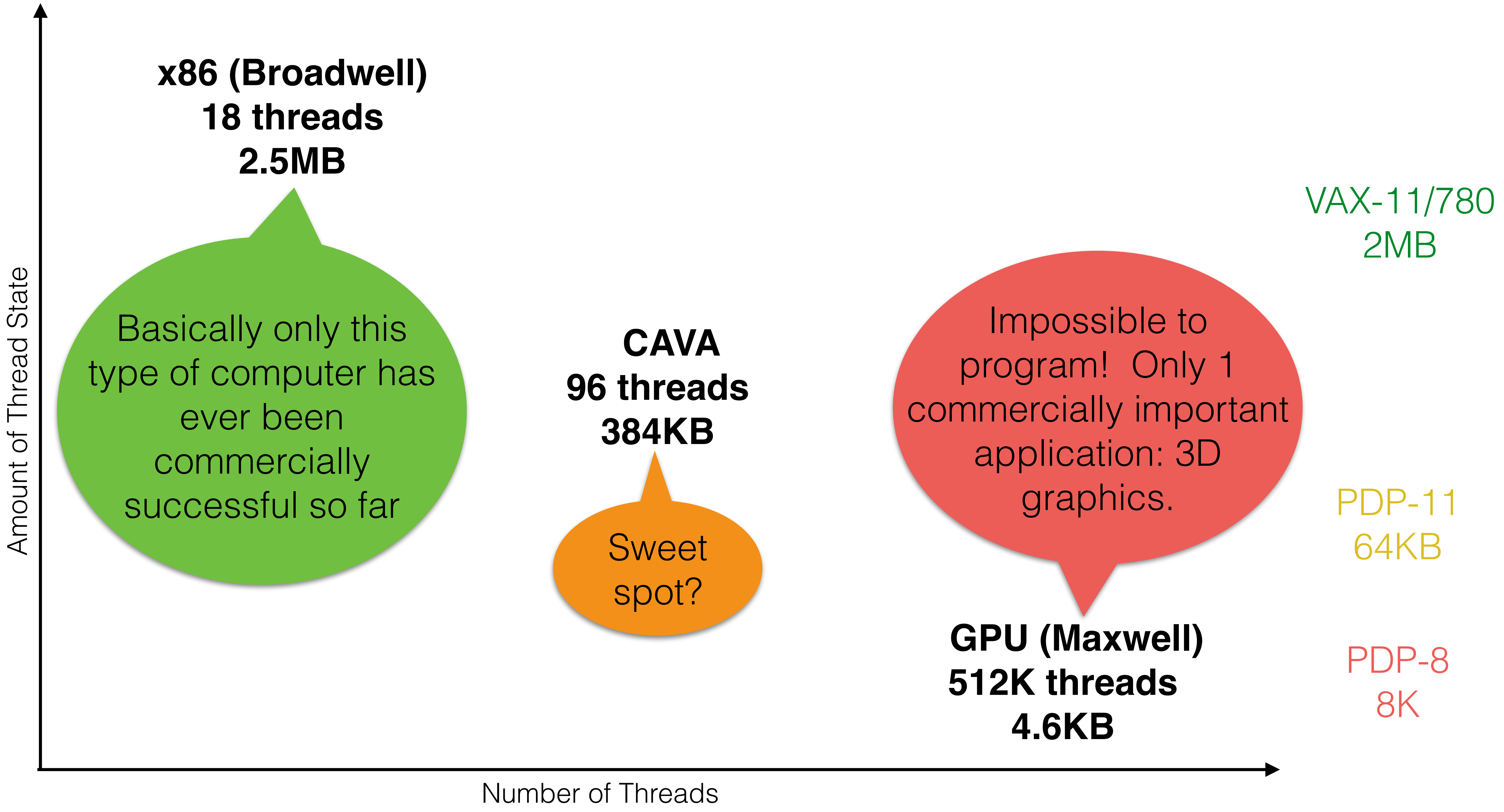


Oracle Labs RAPID
First discussion 2010



CAVA 2015

How Much Thread State?



x86 (Broadwell)
18 threads
2.5MB

Basically only this type of computer has ever been commercially successful so far

CAVA
96 threads
384KB

Sweet spot?

Impossible to program! Only 1 commercially important application: 3D graphics.

GPU (Maxwell)
512K threads
4.6KB

VAX-11/780
2MB

PDP-11
64KB

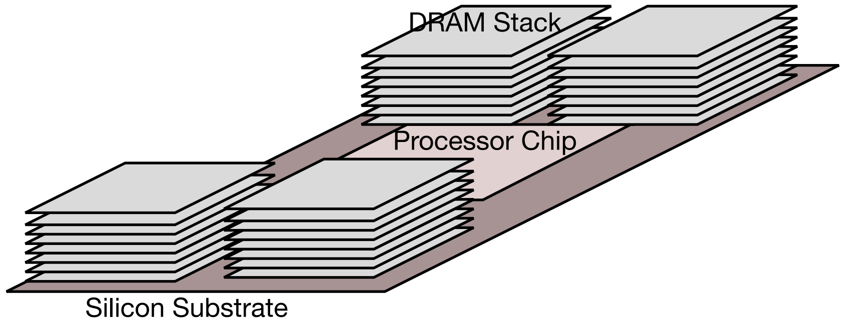
PDP-8
8K

Amount of Thread State

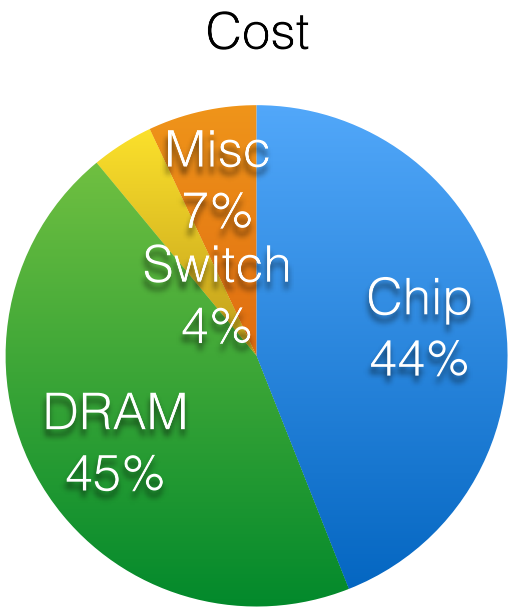
Number of Threads

Exa-Scale HPC Story

- Ideal device for “intelligent memory” paradigm
  - JEDEC Wide I/O 2 Standard using Through Silicon Vias
  - 512 nodes per RU, 16K nodes per rack
  - Plenty of room left over for switches, water cooling pumps



- Challenge will be heat dissipation
  - Keeping DRAM’s below 85C when the entire unit dissipates 22W but is <1 inch^3
- ExaFLOP system in 76 racks
  - 35MW in 2019[†] (but material cost approximately $200M 😱)
- Moore’s Law: 2X cheaper every 2 years
  - 2019 -> 2021 -> 2023 -> 2025: $200M -> $100M -> $50M -> $25M?
  - Well maybe not quite, but it’ll get a lot cheaper

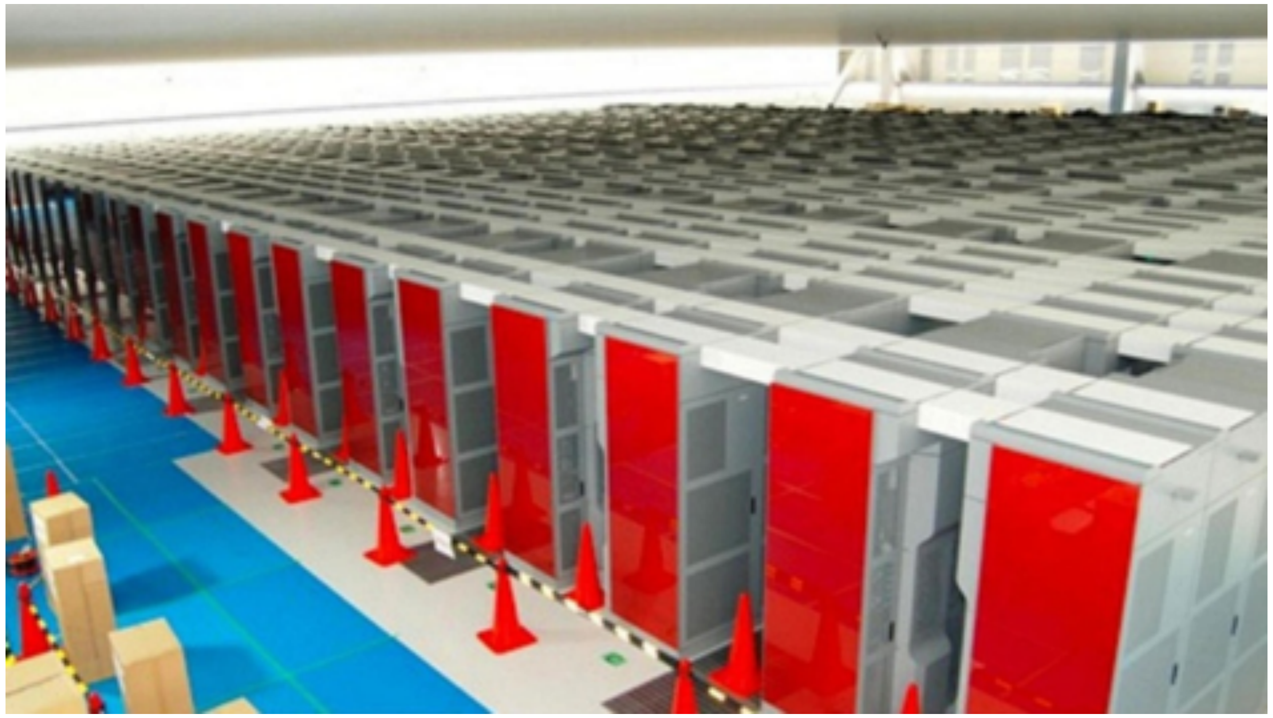


[†]USDOE power target is 20-40MW in 2020 [http://science.energy.gov/ascr/research/scidac/exascale-challenges/]

“Trickle Down” Product

- 16K-node one-rack mini-supercomputer
  - 32 512-node boards
  - 13 PetaFLOPS
  - 450kW
  - Cost $2.5M, price < $10M
- 32K-node twin-rack version
  - 26 PetaFLOPS
  - 900kW
  - Cost $5M, price < $20M
- Need 209V three-phase power
- Use building chilled water with heat exchanger

- Fujitsu K Computer, 2011
  - 11 PetaFLOPS
  - “Fastest computer in the world,” [http://www.top500.org/lists/2011/11/]
  - 12.7MW
  - $$?



Research Plan

- How to maximize impact of this research on industry?
  - Unified simulation environment
    - Runs on generic clusters of x86 using open-source software, replicable everywhere
    - Everyone “plugs into” same simulator: “apples to apples” comparison
  - Industry people can subsequently replicate benchmark runs—this is almost never possible with published papers—and even make their own variations
- What does Oracle Labs bring to the table?
  - Benchmarks—we know what people pay money to compute 😊

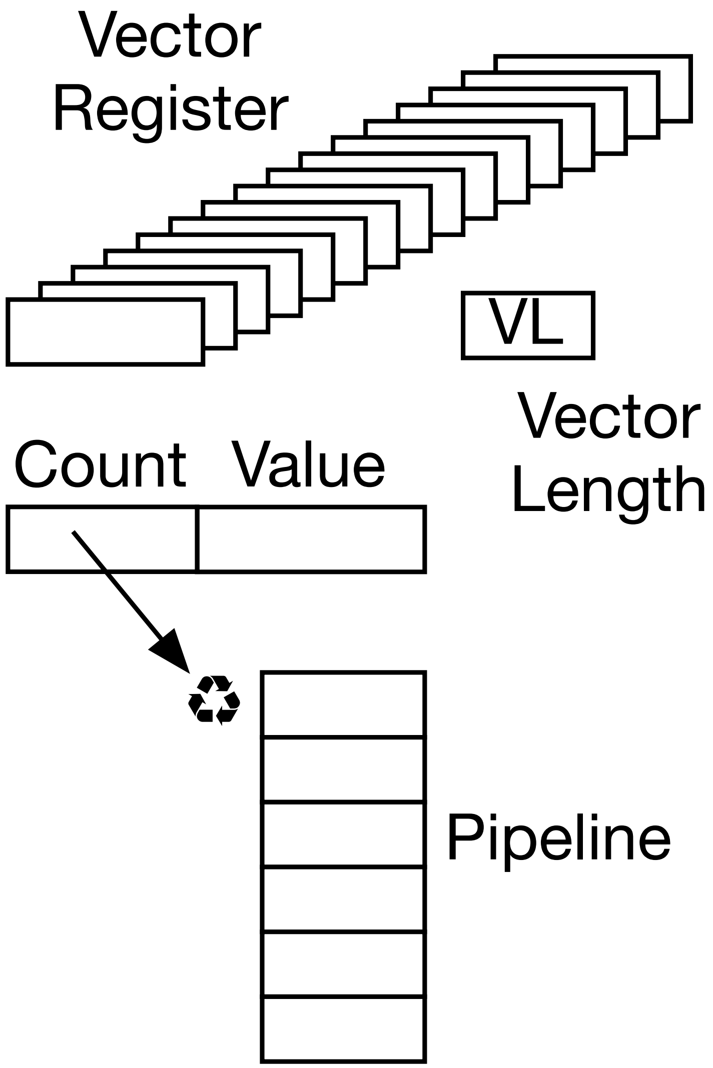
Simulation Infrastructure

1. Pipeline simulator
    - Cycle-accurate, like SimpleScalar (some versions already exist for RISC-V). Scale up to several cores to test coherency, cache-to-cache transfers. Plugs into next level for verification.
2. SoC simulator
    - Entire SoC including all UnCore logic (i.e. NIC, memory controller) cycle-accurate. Cores emulated (QEMU?). Scale up to several SoC's with network switch to calibrate and verify next level simulator.
3. Network simulator
    - Switch chip and NIC logic cycle-accurate, rest of SoC's functionally emulated. Should scale to full network size.

- Should be programmed in open-source software portable to any Linux cluster
    - Must be replicable "for free" so low entry barrier
    - Do not want dependency on expensive FPGA hardware (could be optional)

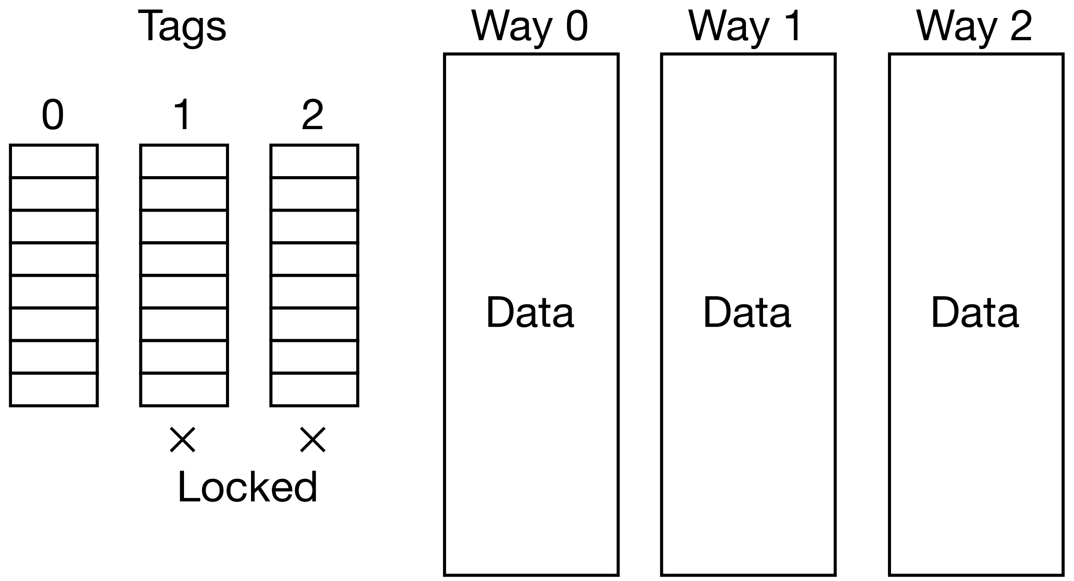
Cray-Style Vectors (1)

- Vector instructions are good match for database analytics
  - Accommodate variable-length compression schemes like run-length encoding
  - Efficiently scatter data into cache because data stored one at a time
- Narrow datapath utilized continuously more energy efficient than SIMD's wide datapath utilized in bursts
  - Rigorous energy comparison using realistic workload would be valuable



Transporter (2)

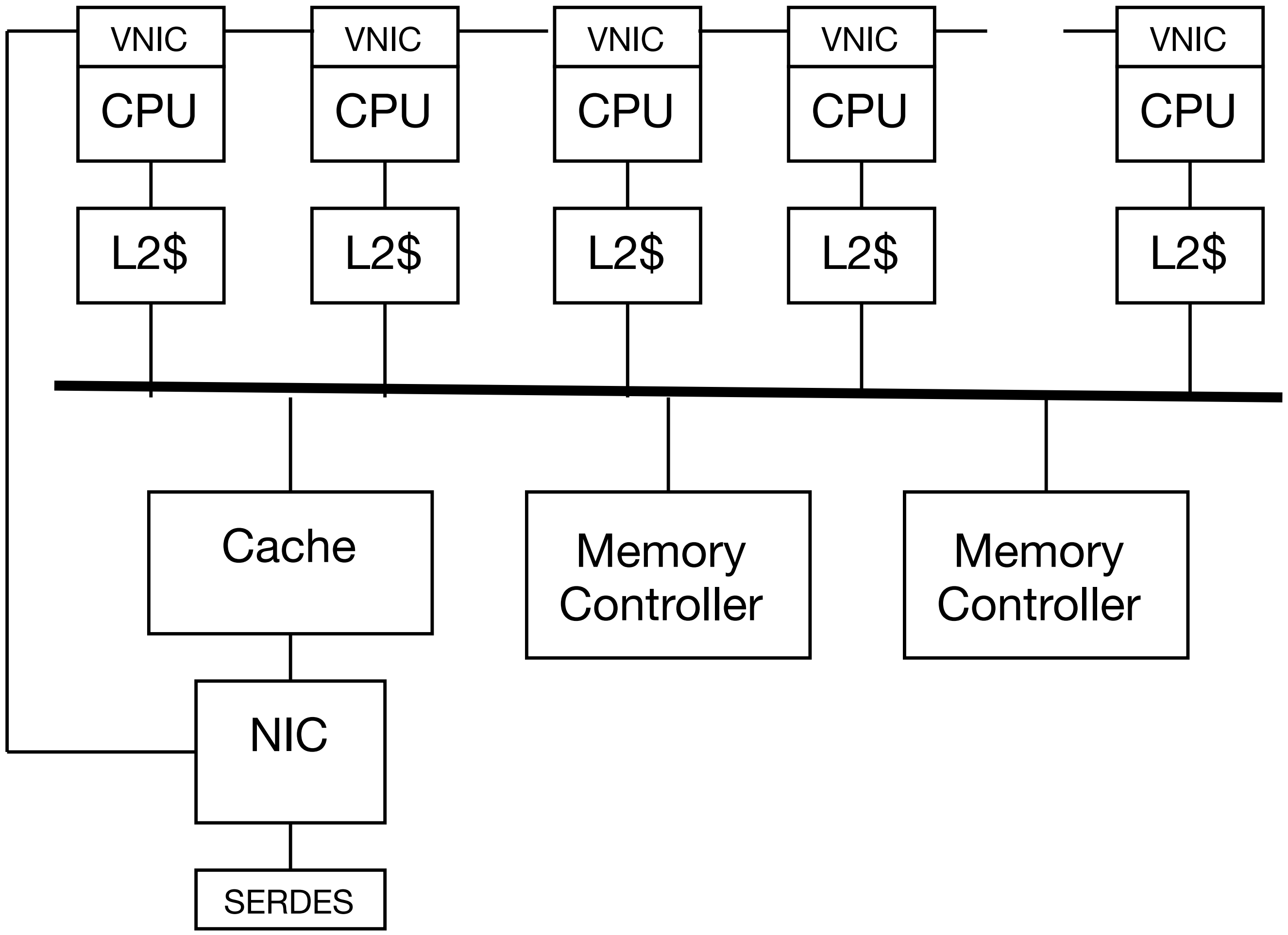
- Make associativity ways and possibly individual cache lines lockable
- Cache loading instructions that inspect some field to decide where to put data
  - Compiler-friendly BCOPY semantics
  - Cache coherent
  - Synchronized with other vector instructions



- Out-of-order core makes L1 invisible
  - Programmer need only explicitly manage much larger L2 cache
  - Vector instructions and BCOPY semantics gives me hope that we can write in high-level language and compile programs involving database acceleration

Lightweight NIC (3)

- Network interface should be fully cache coherent
  - Long history of people thinking it can be done efficiently in software
- “On-loading” instead of off-loading
  - Make use of many nimble cores close by
- Optimize latency for small (64 byte) messages



- Investigate Active Messages
- Investigate small FPGA directing disposition of message
  - e.g. Which core to send interrupt to, what priority

Network Topology (4)

- Single high radix switch chip (128x128?) to accommodate
  - 1K air cooled nodes in a rack using 2-3 hops
  - 16K water cooled nodes in a rack using 3-4 hops
  - 1.5M nodes in 100 racks using 5-7 hops with only 1 long hop (e.g. 2 short hops in source rack, one long haul to destination rack, 2 short hops in that rack)
  - Flexibility for other configurations involving a quarter to half a rack, a couple of racks, up to perhaps a dozen racks
- Integrated photonics implications
  - Bulk DRAM, flash memory, other non-volatile storage on network
  - What happens to algorithms when network approaches memory bandwidth?

Switch Chip (5)

- Mostly focused on the “user interface”
  - Statistics on queuing depth, how long messages are blocked, etc.
  - Requires many strategically placed counters (similar to CPU performance counters) and elaborate software to interpret them
  - May require additional high-priority virtual channels to support timely reporting of statistics
- Higher-level inter-node communication management software using these statistics
  - Intuitively: many opportunities for adaptive routing [not my area]
  - Collaboration research with RAPID software team?

Other Research (6)

- Managed runtime parallel languages (talk to Mario Wolczko)
    - Parallel Java, R, etc.
- Platform for database architectural support research
    - Simulation environment valuable for all kinds of other architecture research
    - e.g. “Selective Virtual Memory:” bit-vector representing pages—cleared=page is mapped by base+bound; set=page is virtual and uses page table
    - Some databases use virtual memory to implement multi-version concurrency control
        - In a large (petabytes) memory-resident database only a tiny fraction of pages would be undergoing changes at any given time
        - Only those pages need multiple versions, thus only those pages need to be virtual and use TLB entries

Synthetic Data

- Oracle has valuable customer datasets
  - We saw many un-obvious program behavior during RAPID development
  - But we cannot, of course, divulge data to academia
- Idea: generate random data with statistical characteristics that are the same as the real data
  - Students run experiments using synthetic data
  - When finished, give program to us to run in-house on real data
  - We report percentage difference between real and synthetic data, which student can publish
  - Student never sees or touches customer dataset
- Could be a “carrot” to attract academia collaboration

Conclusion

- Proposed a new research initiative
  - Energy-efficient parallel computing arena where Oracle Labs could benefit from collaborative research with Universities
  - Focused on designing a commercially viable system, the CAVA computer, as a way to organize research so we have something specific to evaluate
- Primary deliverables
  - Simulation environment and results for Oracle
  - Published papers for students
- Opportunistic results
  - Influence the future direction of Oracle Lab RAPID project
  - Build a prototype (Oracle is a very big company: 2014 revenue $38B…)
  - Rally the computer industry around a new commodity architecture

What is CAVA?

- CAVA is a research initiative
  - Partnership between Oracle Labs and Universities to study energy-efficient parallel computers
- CAVA is a simulation infrastructure
  - Common simulation environment to be used by Oracle Labs and Universities
- CAVA is a computer
  - Open-source design for a new genre of parallel computers that captures the cost effectiveness of GPGPU's while preserving the programmability of generic x86 clusters
- CAVA is a style of architecture
  - I use it to mean clusters that are cache-coherent within a chip, but message-passing and not cache-coherent between chips, among other things
- CAVA™ is just a name I use for all my projects
  - I had created a toolchain based on GCC whereby you specify the ISA and it would automatically generate the compiler, assembler, linker, and an interpreter
- I originally chose the name when I was at a Cava winery in Barcelona during an excursion at a conference

Thank You

Title: The CAVA Computer: Exceptional Parallelism and Energy Efficiency

Abstract: Designing a new computer system is a very expensive proposition. But 80% of it is exactly the same as every other computer—you need caches, multiprocessing, coherency protocols, memory systems, etc. Getting all of that right requires skill and experience, but is taken for granted and does not command much of a premium. Getting it wrong, on the other hand, is a commercial disaster. In this talk I propose a research initiative to standardize and open-source a design for the 80% that is the same in every design, so that everyone can concentrate on adding value to their own remaining 20%. The CAVA computer is a “cluster in a rack” architecture targeting 10nm CMOS technology. The first part of the talk describes a 1024-node system where each node consists of 96-core, 3-issue out-of-order processor chips running at 1GHz with four DDR4 memory channels. Power estimates of different components are discussed, as well as cost projections. The second part of the talk discusses architectural tradeoffs that were made, how this architecture might play in the HPC exa-scale arena, and broader market implications. The talk concludes with how I envision the simulation infrastructure is organized, what Oracle Labs brings to the table, and a list of research topics that I and others at Oracle Labs are actively researching and would be interested in working with students at Universities.

Bio: Peter Hsu was born in Hong Kong and came to the United States at age 15. He received a B.S. degree from the University of Minnesota at Minneapolis in 1979, and the M.S. and Ph.D. degrees from the University of Illinois at Urbana-Champaign in 1983 and 1985, respectively, all in Computer Science. His first job was at IBM Research in Yorktown Heights from 1985-1987, working on superscalar code generation with the 801 compiler team. He then joined his ex-professor at Cydrome, which developed an innovative VLIW computer. In 1988 he moved to Sun Microsystems and tried to build a water-cooled gallium arsenide SPARC processor, but the technology was not sufficiently mature and the effort failed. He joined Silicon Graphics in 1990 and designed the MIPS R8000 TFP microprocessor, which shipped in the SGI Power Challenge systems in 1995. He became a Director of Engineering until 1997, then left to co-found his own startup, ArtX, best known for designing the Nintendo GameCube. ArtX was acquired by ATI Technologies in 2000, which has since been acquired by AMD. Peter left ArtX in 1999 and worked briefly at Toshiba America, then became a visiting Industrial Researcher at the University of Wisconsin at Madison in 2001. He then consulted part time at various startups, and attended the Art Academy University and the California College of the Arts in San Francisco where he learned to paint oil portraits, and a Paul Mitchell school where he learned to cut and color hair. In the late 2000's he consulted for Sun Labs, which lead to discussions about the RAPID project post acquisition. Peter joined Oracle Labs as an Architect in 2011.

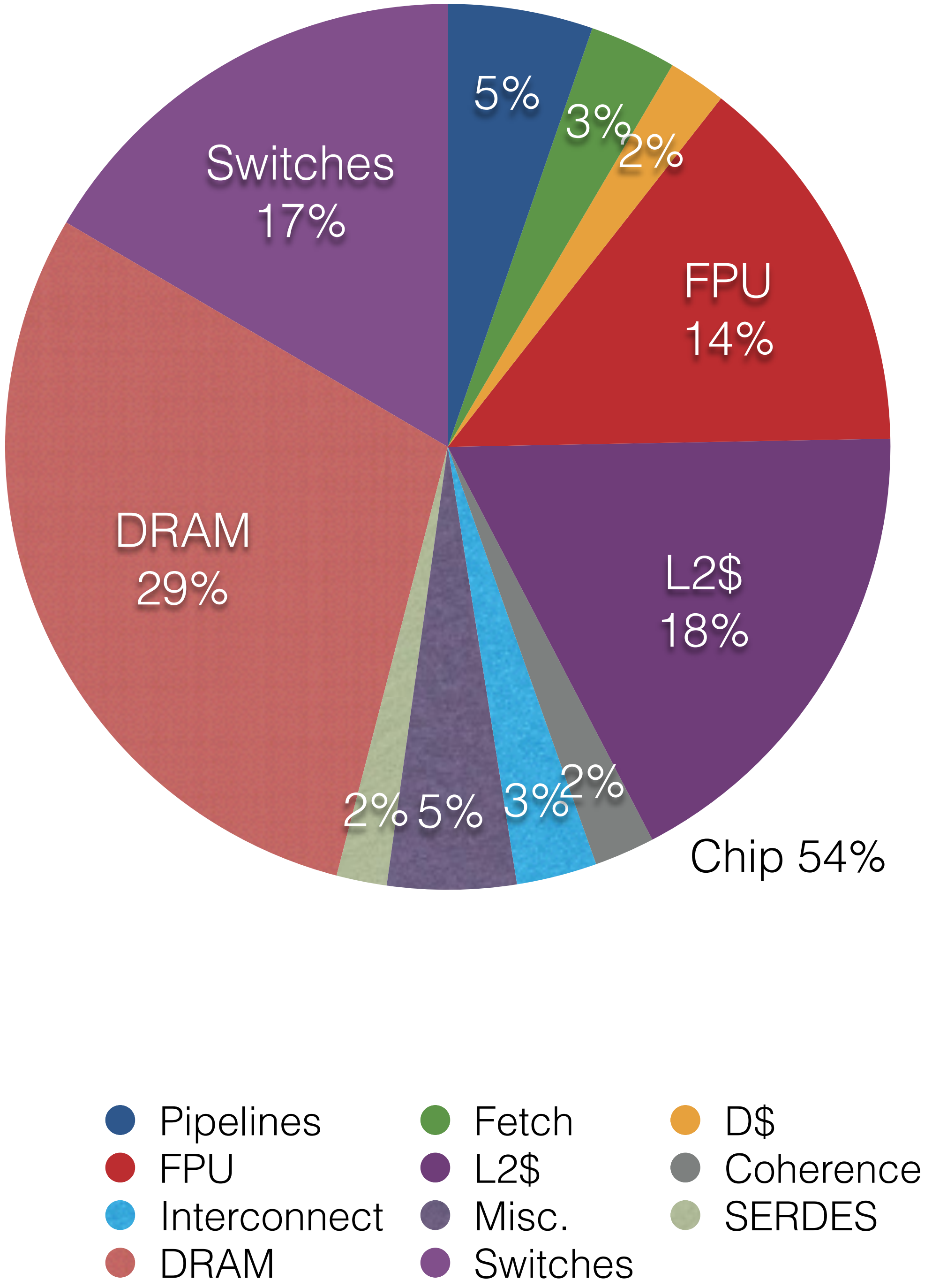
Backup Slides

Outline of Talk

- Description of the CAVA system
    - Size, power, cost
        - Determines how many units one can reasonably expect to sell, which affects how interested people will be in porting software to it
    - Packaging sophistication, process technology
        - Big effect on development costs
- Discussion of tradeoffs
    - Parallelism and amount of thread state
    - Fewer, beefier cores vs. more skinny cores
- Areas that Oracle Labs would be interested in supporting University research

Power (HPC Case)

- FPU 6pJ FMAD + 4pJ RF = 10pJ/lane x 4 lanes
  - [Marc Snir, Argonne National Lab, http://press3.mcs.anl.gov/computingschool/files/2014/01/dinner-talk-8-13.pdf]
- Out-of-order pipelines not fully busy when vector unit busy (50%?), but L2$ fully busy
- Core + VU + L2$ = 25pJ + 40pJ + 50pJ = 115pJ
- 13.7W chip (+70%), 26W node
- Rack = 32kW (28% higher than scalar use case)



Physical Considerations

- Four memory channels = 500 signals = economical limit
  - Printed-circuit boards (PCB) with 4-layers are very much cheaper than boards with more layers
  - Four DRAM channels can be arranged top and bottom on each side of the chip, with power and ground planes on the inner layers
  - Eight channels would require using extra Buffer-on-Board (BoB) chips. In my opinion that leads to a suboptimal design—each chip takes roughly the same amount of board space, so why not just have more nodes instead of BoB chips
- A heat sink with 1m/s (200 ft/min) airflow dissipates 5W/inch^3
  - A 15W chip (worse-case, HPC usage) needs 3inch^3 (49cm^3) heatsink
    - e.g. 4x4x3cm high (rack unit is 4.4cm high)
- Standard rack depths are 600, 800, and 1010 mm. The 32-node 1U card requires 800mm just for the nodes, so the 1010mm version is necessary for fans in the front and room for network cables in the back.

100 GIPS Design Points

Similar Chip Sizes	High Frequency	Medium Frequency	Low Frequency
1.0 IPC 3-issue out-of-order	64 cores 1.6 GHz 512KB	96 cores 1.0 GHz 386KB	128 cores 800 MHz 256KB
0.75 IPC 2-issue in-order	96 cores 1.4 GHz 384KB	128 cores 1.0 GHz 256KB	192 cores 700 MHz 192KB
0.5 IPC 1-issue in-order	128 cores 1.6 GHz 256KB	192 cores 1.0 GHz 192KB	256 cores 800 MHz 128KB

MIPS R10000

- 2 ALU, 1 Load/Store, 1 FP pipelines
- 32 architecture + 32 rename integer registers
- 32KB I$ + 32KB D$, both 2-way set-associative
- External 2-way set associative L2$, configurable size
- 6.8M transistors, 4.4M in SRAM, 2.4 in logic (600K gates)
- 298mm^2 in 0.35um, introduced January 1996
- Achieves 1.0 IPC on SpecInt gcc, >1 IPC on "easy" codes
- Makes L1$ invisible—programmer need only allocate in L2
- I believe this type of microarchitecture has good potential for low power, given the very low gate count
- My personal experience is that out-of-order pipelines are much easier to scale to higher frequencies (or equivalently lower voltage) than in-order pipelines

Chip Power

	Unit pJ	Number	Total pJ
I-Cache SRAM (32-Bits)	3	3	9
Instruction Execution Pipeline	10	3	30
Load-Store SRAM (64-Bits)	6	1	6
Misc.	5	1	5
Core Total			**50**
32KB SRAM Block (64-Bits)	6	4	24
Transmit 100 bits (address+data) 1mm	5	4	20
Tag Lookup 4Kx32b	2	3	6
L2 Total			50
L2 Miss rate	10%		
Effective L2 Contribution			**5**
Number of Cores	96		
Cores + L2's			**5280**
Duplicate Tags for Coherence	**6**	**96**	**576**
72B/cycle Read Traffic (1% L2 Miss Rate)	0.05/bit/mm	20mm	576
18B/cycle Write Traffic (25% Write-Back)			144
Control, Misc.			180
Global Interconnect			**900**
Network Link (single)	10	25	250
Network SERDES	**250**	**2**	**500**
Miscelaneous Logic			**689**
Chip Power			**8000**

Node Power

- DRAM = 8W
  - 220mW per chip -> 180mA @ 1.2V
  - 1% miss rate = 80GB/s = 80% utilization
  - 12.5pJ/bit effective, 8.9pJ/bit raw
- Switch chip = 24W
  - 10pJ/bit SERDES @ 25Gb/s x 64 ports = 16W
  - Internal router logic = 8W
- 192 switches / 1024 nodes = 4.5W / node
- Total node power = 8W chip + 8W DRAM + 4.5W switches = 20.5W

Total Power (HPC Use Case)

- Vector unit = 40pJ
  - FPU 6pJ FMAD + 4pJ RF = 10pJ/lane x 4 lanes
    - [Marc Snir, Argonne National Lab, http://press3.mcs.anl.gov/computingschool/files/2014/01/dinner-talk-8-13.pdf]
- Out-of-order pipelines not fully busy when vector unit busy (50%?)
- Core + VU + L2$ = 25pJ + 40pJ + 50pJ = 115pJ
- 96 cores = 11W (vs. 5.3W)
- 0.6W coherency + 0.8W wires + 0.5W network + 0.8W misc
- 13.7W chip (+70%), 26.2W node
- Rack = 32kW (28% higher than scalar use case)

Knee of the Curve

- 96KB working memory
  - 2 Input streams (double buffered): 4 x 2B = 8KB
  - Dictionary: 2K entries x 4B key, 4B data = 16KB
  - Partition table: 2K x 32B = 64KB
  - End pointers: 2K x 1B = 2KB
  - Stack frames: 1K x 6 frames = 6KB
- Nothing left over

- 384KB working memory
  - 2 Input streams (double buffered): 4 x 8KB = 32KB
  - Dictionary: 8K entries x 4B key, 4B data = 64KB
  - Partition table: 8K x 32B = 256KB
  - End pointers: 8K x 1B = 8KB
  - Stack frames: 1K x 6 frames = 6KB
- 18KB left over

Knee of the Curve

- 96KB working memory
  - 2 Input streams (double buffered): 4 x 2B = 8KB
  - Dictionar ntries x 4B key, 4B B
  - Partiti 32B = 64KB
  - End pointers: 2K x 1B = 2KB
  - Stack frames: 1K x 6 frames = 6KB
- Nothing left over

- 384KB working memory
  - 2 Input streams (double buffered): 4 x 8KB = 32KB
  - Dictionary: ntries x 4B key, 4B data
  - Partitio 32B = 256KB
  - End pointers: 8K x 1B = 8KB
  - Stack frames: 1K x 6 frames = 6KB
- 18KB left over

Development Time

- Much forgotten experience from days when computers had 64KB memories
  - Programmers spent much time battling algorithms to minimize footprint
  - Subtle bugs due to reusing storage location (temporal overlay)
  - Everything a bit too small to be really efficient
- Although nothing was impossible to do, everything took much longer to do
  - One reason caches and virtual memory were so wonderful when they were introduced—they gave the illusion of unbounded memory, so you didn't have to worry about correctness when you develop your algorithm, only later when you tune it for performance
- Breaking the 64KB barrier on 16-bit minicomputers such as PDP-11's was commercially very significant—many new applications became possible (Oracle Version 1 ran on PDP-11 in 128KB in 1978)

Architecture (1)

- CAVA is a very general-purpose architecture
  - A short, out-of-order pipeline like the MIPS R10K is very robust and achieves excellent IPC on diverse workloads.
  - Private L2 caches provide high bandwidth even under high load, e.g. vector operations.
  - CLOS (or Dragonfly, Flattened Butterfly) have low diameter and save power by using high-radix switches. They also reduce latency, especially for short messages.
- We should be thinking about new markets
  - Once we migrate to integrated silicon photonics,100+ Gb/s optical links at under 2pJ/bit are possible, and network bandwidth approaches memory bandwidth.
  - Convergence of database analytics, graph, web, HPC, big data. CAVA feels like the right kind of architecture—energy efficiency and parallelism are the right metrics.

Architecture (2)

- The new era of “Internet of Things (IoT)” will have an enormous amount of sensors generating data continuously. This data cannot all be shipped to the cloud to be analyzed—transmitting data long distance will always cost energy—so it will need to be crunched locally and summarized. CAVA is the right kind of computer for this—a small, on-location cloud (fog?)

- “Internet responsible for 2 per cent of global energy usage[†].” Continuing to simply aggregate existing PCs into large data centers without improving their energy efficiency does not seem good for the earth. CAVA leverages low-power cellphone technology in the server space.

- Possibility of a new standard commodity architecture. There are no existing installed base of parallel programs, so backwards compatibility is not an issue. Many developing countries are creating semiconductor industries and need to rally around an open architecture. The RISC-V ISA and the CAVA system architecture could be a good fit.

[†] http://www.newscientist.com/blogs/onepercent/2011/10/307-gw-the-maximum-energy-the.html

Microarchitecture

- Out-of-order multi-issue pipeline
  - Condenses finite on-chip memory into fewer cores for larger thread state
  - Makes L1 cache "invisible"—programmer allocate in larger L2 and hardware manages L1 automatically
  - Perform better on wider class of applications whose performance are not dominated by tight inner loops
  - Considerably more costly to develop, several times larger in chip area, dissipates more power (but still only 20% of system)
- Set-associative cache with lockable ways
  - Programmer need not manage fix-sized partition of cache and scratchpad
  - Plays well with "transporter" (discussed later)

Note: Microarchitecture and architecture are separate issues—you can disagree with me on this without necessarily disagreeing with me on bigger system architecture issues

System Components

- Storage nodes
  - Flash or other NV-RAM technologies
    - May have DRAM cache
  - Accessed through network
- Switch chips
  - 128x128 ports? (high radix desirable)
    - 256 SERDES (512 signals)
  - 1 DDR4 channel (125 signals)
    - 16 processor cores
  - Ethernet ports
  - Gen3 x8 PCIe bus