Intelligent RAM (IRAM)

Richard Fromm, David Patterson,
Krste Asanovic, Aaron Brown, Jason Golbus,
Ben Gribstad, Kimberly Keeton, Christoforos
Kozyrakis, David Martin, Stylianos Perissakis,
Randi Thomas, Noah Treuhaft, Katherine
Yelick, Tom Anderson, John Wawrzynek

rfromm@cs.berkeley.edu

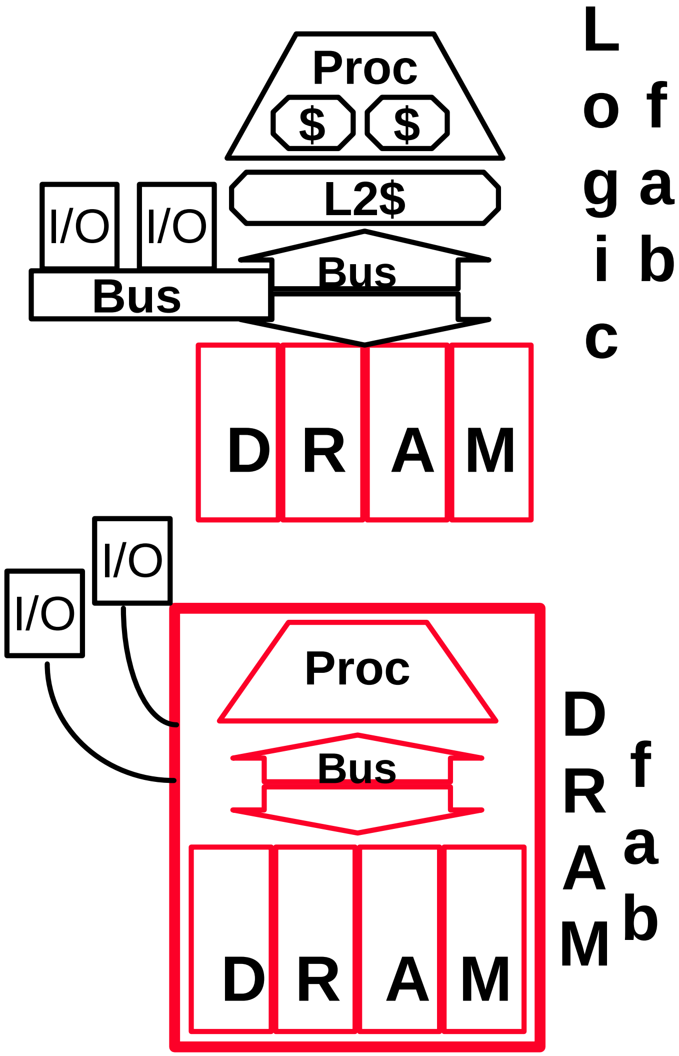
http://iram.cs.berkeley.edu/

EECS, University of California
Berkeley, CA 94720-1776 USA

IRAM Vision Statement

Microprocessor & DRAM on a single chip:

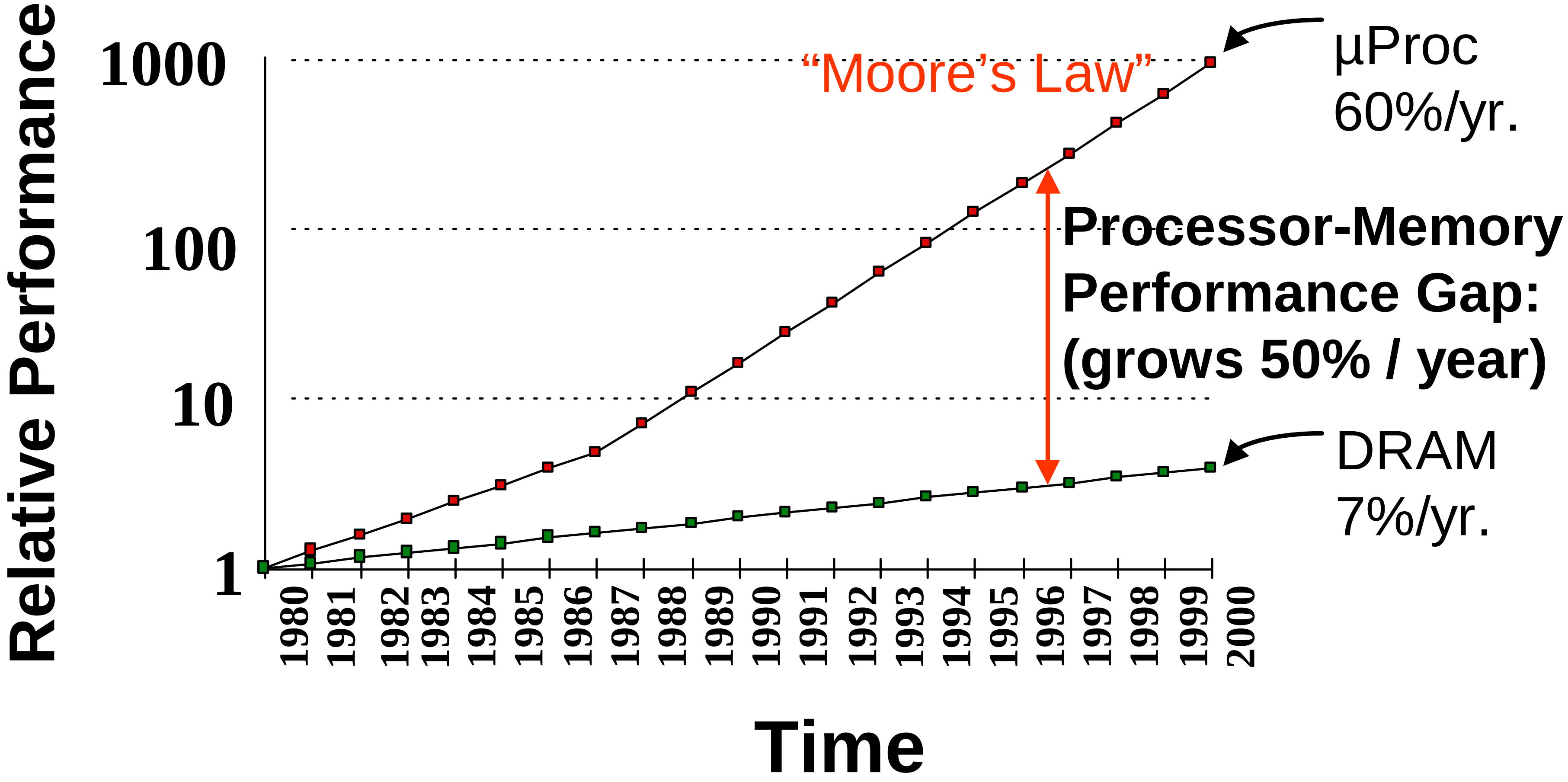
- on-chip memory latency 5-10X, bandwidth 50-100X
- improve energy efficiency 2X-4X (no off-chip bus)
- serial I/O 5-10X vs. buses
- smaller board area/volume
- adjustable memory size/width



Outline

- Today’s Situation: Microprocessor & DRAM
- IRAM Opportunities
- Initial Explorations
- Energy Efficiency
- Directions for New Architectures
- Vector Processing
- Serial I/O
- IRAM Potential, Challenges, & Industrial Impact

Processor-DRAM Gap (latency)



Processor-Memory Performance Gap “Tax”

Processor	% Area *(~cost)*	% Transistors *(~power)*
Alpha 21164	37%	77%
StrongArm SA110	61%	94%
Pentium Pro	64%	88%

- 2 dies per package: Proc/I$/D$ + L2$

- Caches have no inherent value, only try to close performance gap

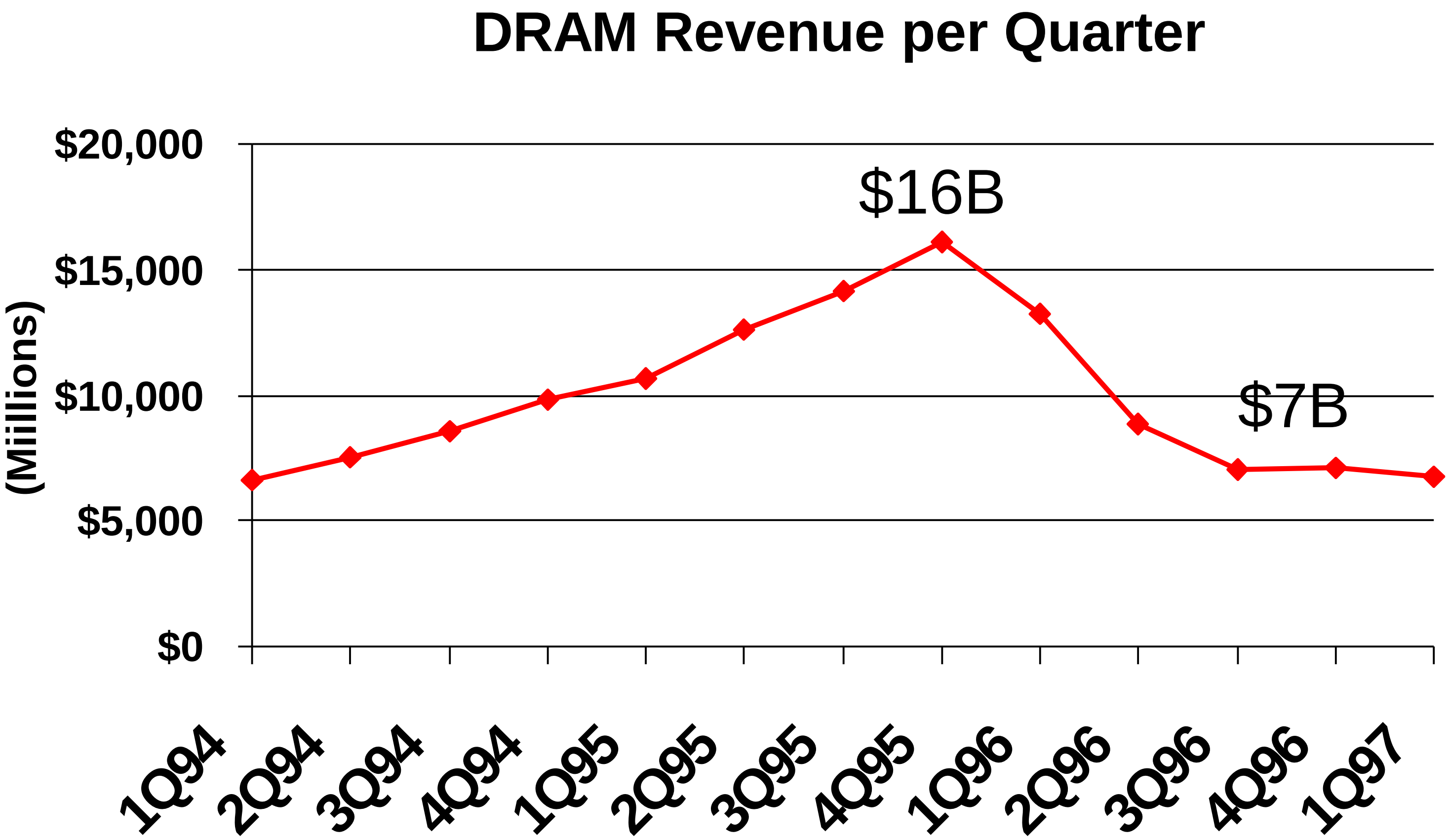
Today’s Situation: Microprocessor

- Rely on caches to bridge gap
- Microprocessor-DRAM performance gap
  - time of a full cache miss in instructions executed

1st Alpha (7000):	340 ns/5.0 ns = 68 clks x 2 or 136 ns
2nd Alpha (8400):	266 ns/3.3 ns = 80 clks x 4 or 320 ns
3rd Alpha (t.b.d.):	180 ns/1.7 ns =108 clks x 6 or 648 ns

  - ½X latency x 3X clock rate x 3X Instr/clock ⇒ - 5X
- Power limits performance (battery, cooling)
- Shrinking number of desktop ISAs?
  - No more PA-RISC; questionable future for MIPS and Alpha
  - Future dominated by IA-64?

Today's Situation: DRAM



- Intel: 30%/year since 1987; 1/3 income profit

Today’s Situation: DRAM

- Commodity, second source industry
  ⇒ high volume, low profit, conservative
  - Little organization innovation (vs. processors) in 20 years: page mode, EDO, Synch DRAM
- DRAM industry at a crossroads:
  - Fewer DRAMs per computer over time
    - Growth bits/chip DRAM: 50%-60%/yr
    - Nathan Myrvold (Microsoft): mature software growth (33%/yr for NT), growth MB/$ of DRAM (25%-30%/yr)
  - Starting to question buying larger DRAMs?

Fewer DRAMs/System over Time

(from Pete MacWilliams, Intel)

Minimum Memory Size	DRAM Generation					
	‘86	‘89	‘92	‘96	‘99	‘02
	1 Mb	4 Mb	16 Mb	64 Mb	256 Mb	1 Gb
4 MB	32 →	8				
8 MB		16 →	4			
16 MB			8 →	2		
32 MB				4 →	<u>1</u>	
64 MB				8 →	2	
128 MB					4 →	<u>1</u>
256 MB					8 →	2

Memory per DRAM growth @ 60% / year →

Memory per System growth @ 25%-30% / year ↓

Multiple Motivations for IRAM

- Some apps: energy, board area, memory size
- Gap means performance challenge is memory
- DRAM companies at crossroads?
  - Dramatic price drop since January 1996
  - Dwindling interest in future DRAM?
    - Too much memory per chip?
- Alternatives to IRAM: fix capacity but shrink DRAM die, packaging breakthrough, more out-of-order CPU, ...

DRAM Density

- Density of DRAM (in DRAM process) is **much** higher than SRAM (in logic process)
  - Pseudo-3-dimensional trench or stacked capacitors give very small DRAM cell sizes

	StrongARM	**64 Mbit DRAM**	**Ratio**
Process	0.35 μm logic	0.40 μm DRAM	
Transistors/cell	6	1	6:1
Cell size (μm^2)	26.41	1.62	16:1
(λ^2)	216	10.1	21:1
Density (Kbits/μm^2)	10.1	390	1:39
(Kbits/λ^2)	1.23	62.3	1:51

Potential IRAM Latency: 5 - 10X

- No parallel DRAMs, memory controller, bus to turn around, SIMM module, pins…
- New focus: Latency oriented DRAM?
  - Dominant delay = RC of the word lines
  - Keep wire length short & block sizes small?
- 10-30 ns for 64b-256b IRAM "RAS/CAS"?
- AlphaStation 600: 180 ns=128b, 270 ns=512b
  Next generation (21264): 180 ns for 512b?

Potential IRAM Bandwidth: 50-100X

- 1024 1Mbit modules(1Gb), each 256b wide
  - 20% @ 20 ns RAS/CAS = 320 GBytes/sec
- If cross bar switch delivers 1/3 to 2/3 of BW of 20% of modules
  ⇒ 100 - 200 GBytes/sec
- FYI: AlphaServer 8400 = 1.2 GBytes/sec
  - 75 MHz, 256-bit memory bus, 4 banks

Potential Energy Efficiency: 2X-4X

- Case study of StrongARM memory hierarchy vs. IRAM memory hierarchy *(more later...)*
  - cell size advantages
    ⇒ much larger cache
    ⇒ fewer off-chip references
    ⇒ up to 2X-4X energy efficiency for memory
  - less energy per bit access for DRAM

Potential Innovation in Standard DRAM Interfaces

- Optimizations when chip is a system vs. chip is a memory component
  - Lower power via on-demand memory module activation?
  - Improve yield with variable refresh rate?
  - “Map out” bad memory modules to improve yield?
  - Reduce test cases/testing time during manufacturing?
- IRAM advantages even greater if innovate inside DRAM memory interface? *(ongoing work...)*

“Vanilla” Approach to IRAM

- Estimate performance of IRAM implementations of conventional architectures
- Multiple studies:
  - “Intelligent RAM (IRAM): Chips that remember and compute”, *1997 Int’l. Solid-State Circuits Conf.*, Feb. 1997.
  - “Evaluation of Existing Architectures in IRAM Systems”, *Workshop on Mixing Logic and DRAM, 24th Int’l. Symp. on Computer Architecture*, June 1997.
  - “The Energy Efficiency of IRAM Architectures”, *24th Int’l. Symp. on Computer Architecture*, June 1997.

"Vanilla" IRAM - Performance Conclusions

- IRAM systems with existing architectures provide only moderate performance benefits
- High bandwidth / low latency used to speed up memory accesses but not computation
- Reason: existing architectures developed under the assumption of a low bandwidth memory system
  - Need something better than "build a bigger cache"
  - Important to investigate alternative architectures that better utilize high bandwidth and low latency of IRAM

IRAM Energy Advantages

- IRAM reduces the frequency of accesses to lower levels of the memory hierarchy, which require more energy
- IRAM reduces energy to access various levels of the memory hierarchy
- Consequently, IRAM reduces the average energy per instruction:

Energy per memory access =

$AE_{L1} + (MR_{L1}$ ´ $AE_{L2} + (MR_{L2}$ ´ $AE_{\text{off-chip}}))$

where AE = access energy
and MR = miss rate

Energy to Access Memory by Level of Memory Hierarchy

- For 1 access, measured in nJoules:

	Conventional	IRAM
on-chip L1$(SRAM)	0.5	0.5
on-chip L2$(SRAM vs. DRAM)	2.4	1.6
L1 to Memory (off- vs. on-chip)	98.5	4.6
L2 to Memory (off-chip)	316.0	*(n.a.)*

  - Based on Digital StrongARM, 0.35 µm technology
  - Calculated energy efficiency (nanoJoules per instruction)
  - See “The Energy Efficiency of IRAM Architectures,” *24th Int’l. Symp. on Computer Architecture*, June 1997

IRAM Energy Efficiency Conclusions

- IRAM memory hierarchy consumes as little as 29% (Small) or 22% (Large) of corresponding conventional models
- In worst case, IRAM energy consumption is comparable to conventional: 116% (Small), 76% (Large)
- Total energy of IRAM CPU and memory as little as 40% of conventional, assuming StrongARM as CPU core
- Benefits depend on how memory-intensive the application is

A More Revolutionary Approach

- "...wires are not keeping pace with scaling of other features. … In fact, for CMOS processes below 0.25 micron ... *an unacceptably small percentage of the die will be reachable during a single clock cycle*."
- "Architectures that require long-distance, rapid interaction will not scale well ..."
  - "Will Physical Scalability Sabotage Performance Gains?" Matzke, *IEEE Computer* (9/97)

New Architecture Directions

- "…media processing will become the dominant force in computer arch. & microprocessor design."
- "... new media-rich applications... involve significant real-time processing of continuous media streams, and make heavy use of vectors of packed 8-, 16-, and 32-bit integer and floating pt."
- Needs include high memory BW, high network BW, continuous media data types, real-time response, fine grain parallelism
  - "How Multimedia Workloads Will Change Processor Design", Diefendorff & Dubey, *IEEE Computer* (9/97)

PDA of 2002?

- Pilot PDA (calendar, to do list, notes, address book, calculator, memo, ...)
- + Cell phone
- + Nikon Coolpix (camera, tape recorder, paint ...)
- + Gameboy
- + speech, vision recognition
- + wireless data connectivity (web browser, e-mail)

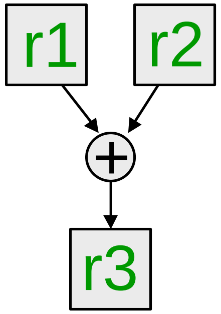
- Vision to see surroundings, scan documents
- Voice input/output for conversations

Potential IRAM Architecture (“*New*”?)

- Compact: Describe N operations with 1 short instruction
- Predictable (real-time) performance vs. statistical performance (cache)
- Multimedia ready: choose N * 64b, 2N * 32b, 4N * 16b
- Easy to get high performance; N operations:
  - are independent (Þ short signal distance)
  - use same functional unit
  - access disjoint registers
  - access registers in same order as previous instructions
  - access contiguous memory words or known pattern
  - can exploit large memory bandwidth
  - hide memory latency (and any other latency)
- Scalable (get higher performance as more HW resources available)
- Energy-efficient
- Mature, developed compiler technology

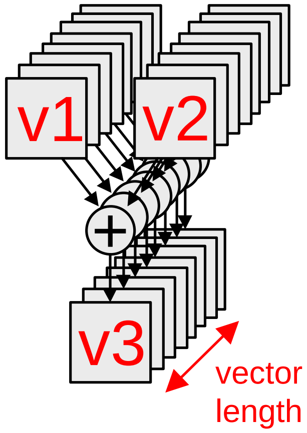
Vector Processing

SCALAR
(1 operation)



```
add r3, r1, r2
```

VECTOR
(N operations)



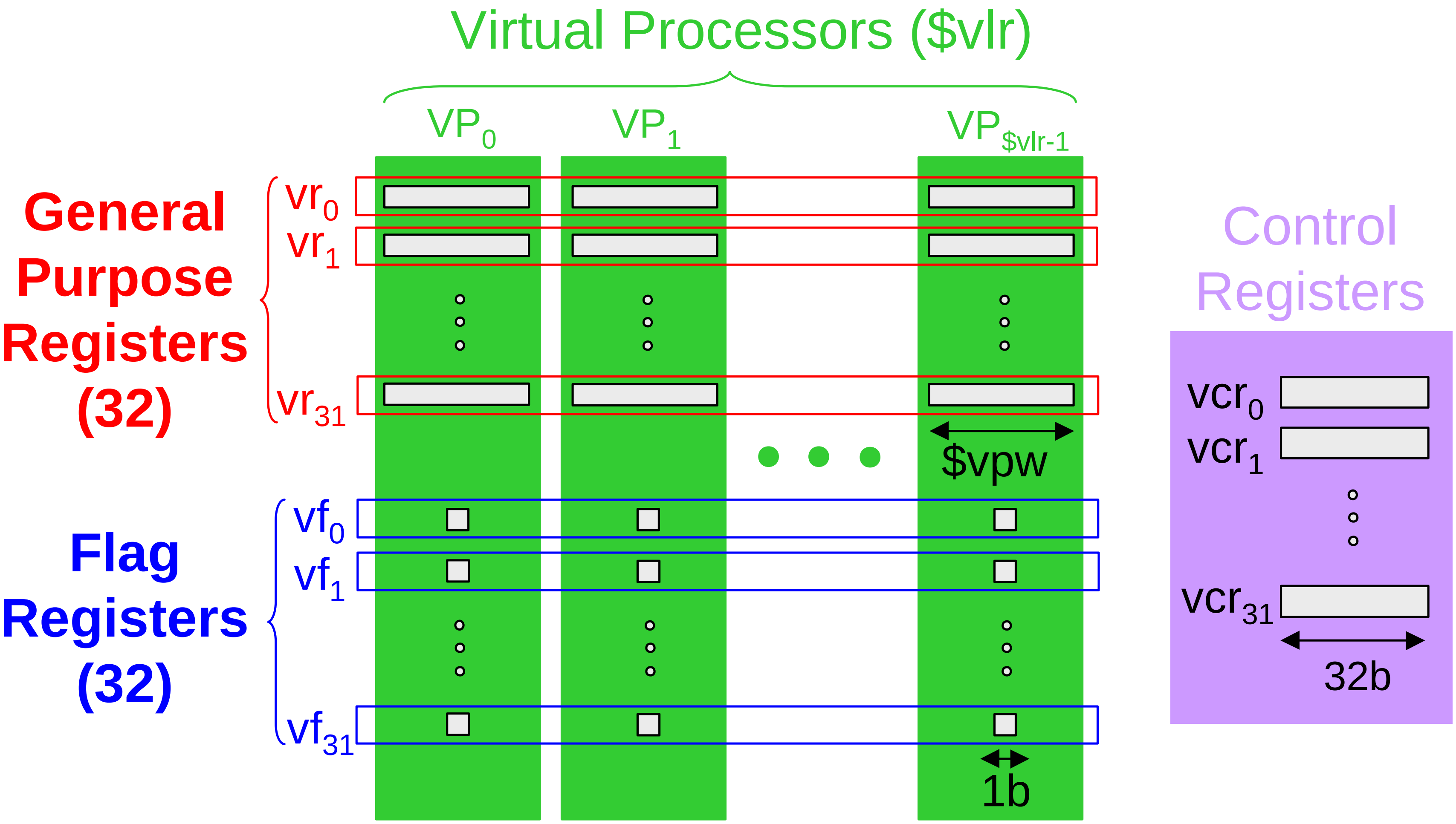
```
add.vv v3, v1, v2
```

Vector Model

- Vector operations are SIMD operations on an array of virtual processors (VP)

- Number of VPs given by the vector length register `vlr`
- Width of each VP given by the virtual processor width register `vpw`

Vector Architectural State



Variable Virtual Processor Width

- Programmer thinks in terms of:
  - Virtual processors of width 16b / 32b / 64b (or vectors of data of width 16b / 32b / 64b)
- Good model for multimedia
  - Multimedia is highly vectorizable with long vectors
  - More elegant than MMX-style model
    - Many fewer instructions (SIMD)
    - Vector length explicitly controlled
    - Memory alignment / packing issues solved in vector memory pipeline
  - Vectorization understood and compilers exist

Virtual Processor Abstraction

- Use vectors for inner loop parallelism (no surprise)
  - One dimension of array: **A[0, 0]**, **A[0, 1]**, **A[0, 2]**, ...
  - Think of machine as 32 vector regs each with 64 elements
  - 1 instruction updates 64 elements of 1 vector register
- and for outer loop parallelism!
  - 1 element from each column: **A[0,0]**, **A[1,0]**, **A[2,0]**, ...
  - Think of machine as 64 “virtual processors” (VPs)
    each with 32 scalar registers! (~ multithreaded processor)
  - 1 instruction updates 1 scalar register in 64 VPs
- Hardware identical, just 2 compiler perspectives

Flag Registers

- Conditional Execution
  - Most operations can be masked
  - No need for conditional move instructions
  - Flag processor allows chaining of flag operations
- Exception Processing
  - Integer: overflow, saturation
  - IEEE Floating point: Inexact, Underflow, Overflow, divide by Zero, inValid operation
  - Memory: speculative loads

Overview of V-IRAM ISA

Scalar Standard scalar instruction set (e.g. ARM, MIPS)

Vector ALU

- alu op
- s.int / u.int / s.fp / d.fp
- 16 / 32 / 64
- .v / .vv / .vs / .sv

All ALU / memory operations under mask

Vector Memory

- load / store
- s.int / u.int
- 16 / 32 / 64
- 8 / 16 / 32 / 64
- unit stride / constant stride / indexed

Vector Registers

32 x 32 x 64b data
32 x 64 x 32b data
32 x 128 x 16b data

+

32 x 32 x 1b flag
32 x 64 x 1b flag
32 x 128 x 1b flag

Plus: **flag**, **convert**, **fixed-point**, and **transfer** operations

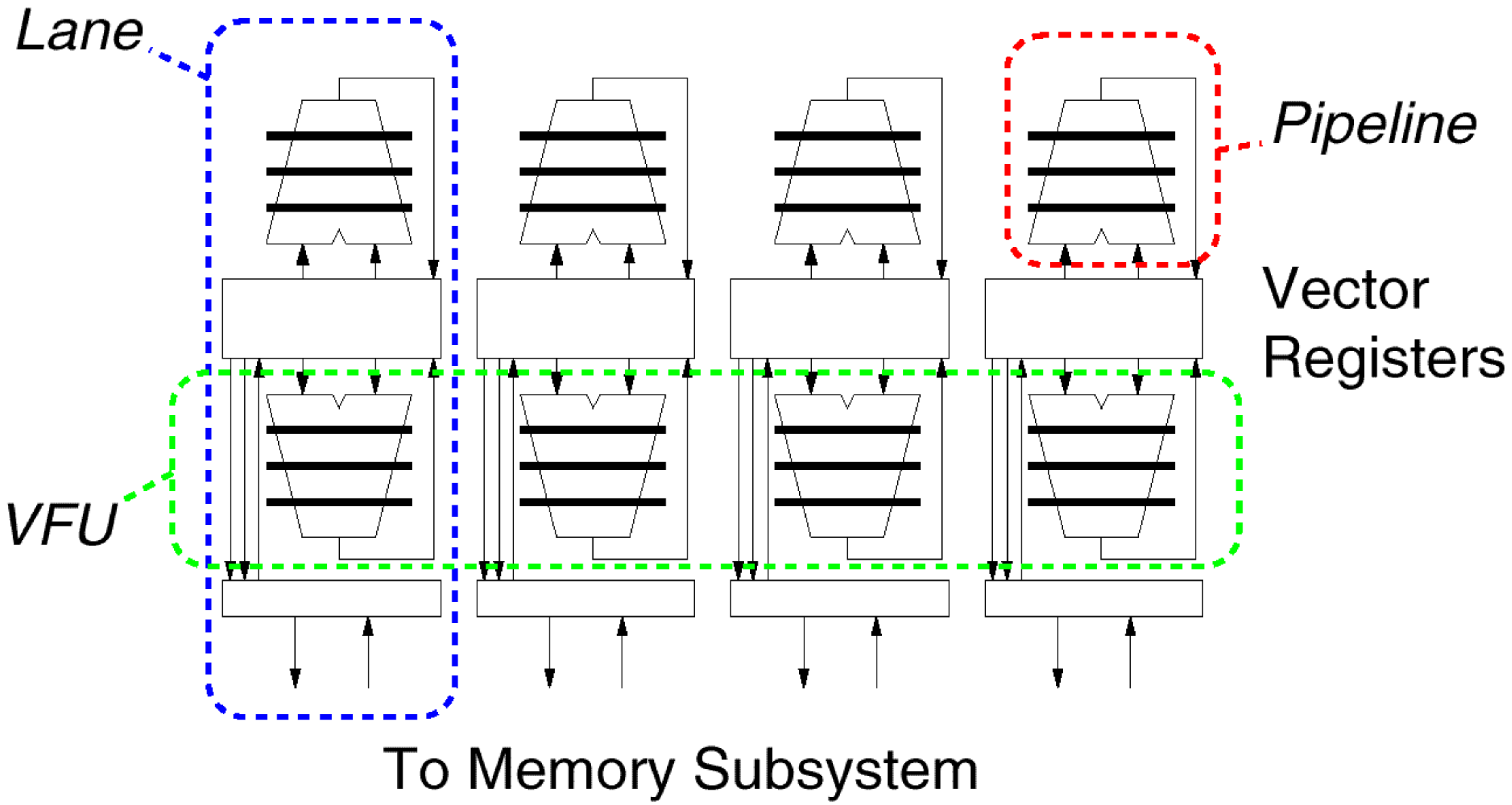
Memory operations

- Load/store operations move groups of data between registers and memory
- Three types of addressing
  - Unit stride
    - Fastest
  - Non-unit (constant) stride
  - Indexed (gather-scatter)
    - Vector equivalent of register indirect
    - Good for sparse arrays of data
    - Increases number of programs that vectorize

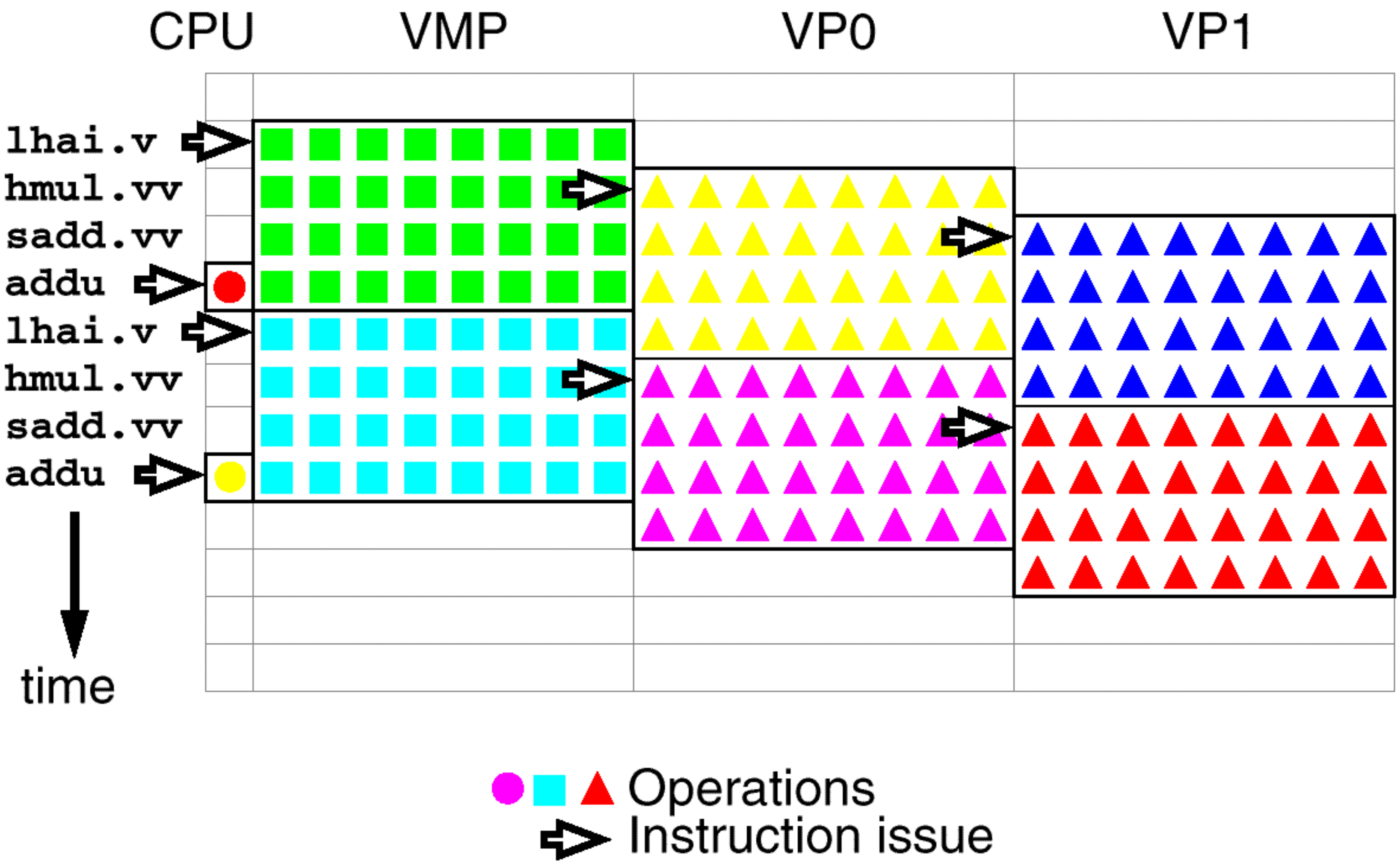
Vector Implementation

- Vector register file
  - Each register is an array of elements
  - Size of each register determines maximum vector length
  - Vector length register determines vector length for a particular operation
- Multiple parallel execution units = “lanes”
- Chaining = forwarding for vector operations

Vector Terminology



Example Execution of Vector Code



Aren’t Vectors Dead?

- High Cost: ~$1M each?
- Low latency, high BW memory system?
- >≈1M xtors for vector processor?
- High power, need elaborate cooling?
- Poor scalar performance?
- No virtual memory?
- Limited to scientific applications?

- Single-chip CMOS µP/IRAM
- IRAM = low latency,
  high bandwidth memory
- Small % in future
  + scales to Bxtor chips
- Vector micro can be low power
  + more energy eff. than superscalar
- Include modern, modest CPU
  Þ OK scalar (MIPS 5K v. 10k)
- Include demand-paged VM
- Multimedia apps vectorizable too:
  N * 64b, 2N * 32b, 4N * 16b

More Vector IRAM Advantages

● Real-time?	● Fixed pipeline, eliminate traditional caches and speculation Þ repeatable speed as vary input
● Vector Performance?	● Easy to scale with technology
● Code density?	● Much smaller than VLIW / EPIC
● Compilers?	● For sale, mature (> 20 years)

Increasing Scalar Complexity

MIPS μPs	R5000	R10000	10K/5K
Clock Rate	200 MHz	195 MHz	1.0x
On-Chip Caches	32K/32K	32K/32K	1.0x
Instructions / Cycle	1(+ FP)	4	4.0x
Pipe stages	5	5-7	1.2x
Model	In-order	Out-of-order	---
Die Size (mm^2)	84	298	3.5x
without cache, TLB	32	205	6.3x
Development (person-yr.)	60	300	5.0x
SPECint_base95	5.7	8.8	1.6x

Vectors Are Inexpensive

Scalar

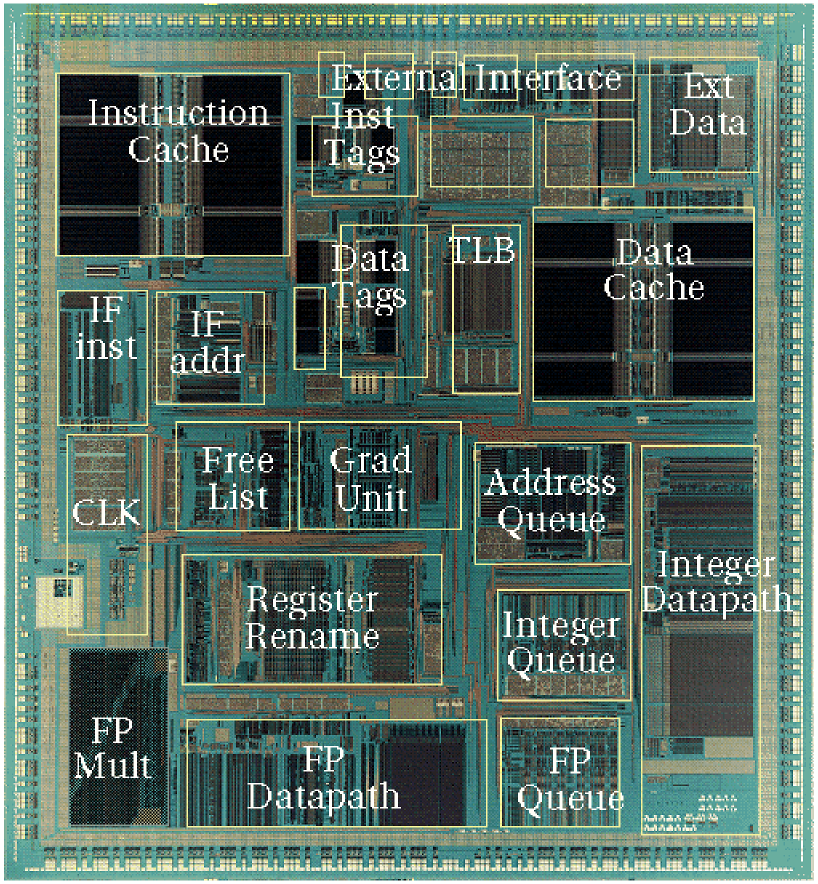
- N ops per cycle
  $\Rightarrow$ O(N^2) circuitry
- HP PA-8000
  - 4-way issue
  - reorder buffer: 850K transistors
    - incl. 6,720 5-bit register number comparators

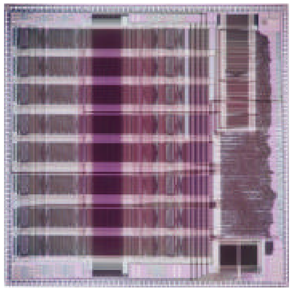
Vector

- N ops per cycle
  $\Rightarrow$ O($N + \varepsilon N^2$) circuitry
- T0 vector micro*
  - 24 ops per cycle
  - 730K transistors total
    - only 23 5-bit register number comparators

*See `http://www.icsi.berkeley.edu/real/spert/t0-intro.html`

MIPS R10000 vs. T0





Vectors Lower Power

Single-issue Scalar	Vector
● One instruction fetch, decode, dispatch per operation	● One instruction fetch,decode, dispatch per vector
● Arbitrary register accesses, adds area and power	● Structured register accesses
● Loop unrolling and software pipelining for high performance increases instruction cache footprint	● Smaller code for high performance, less power in instruction cache misses
● All data passes through cache; waste power if no temporal locality	● Bypass cache
● One TLB lookup per load or store	● One TLB lookup per group of loads or stores
● Off-chip access in whole cache lines	● Move only necessary data across chip boundary

Superscalar Energy Efficiency Worse

Superscalar

- Control logic grows quadratically with issue width
- Control logic consumes energy regardless of available parallelism
- Speculation to increase visible parallelism wastes energy

Vector

- Control logic grows linearly with issue width
- Vector unit switches off when not in use
- Vector instructions expose parallelism without speculation
- Software control of speculation when desired:
  - Whether to use vector mask or compress/expand for conditionals

Applications

Limited to scientific computing? ***NO!***

- Standard benchmark kernels (Matrix Multiply, FFT, Convolution, Sort)
- Lossy Compression (JPEG, MPEG video and audio)
- Lossless Compression (Zero removal, RLE, Differencing, LZW)
- Cryptography (RSA, DES/IDEA, SHA/MD5)
- Multimedia Processing (compress., graphics, audio synth, image proc.)
- Speech and handwriting recognition
- Operating systems/Networking (`memcpy`, `memset`, parity, checksum)
- Databases (hash/join, data mining, image/video serving)
- Language run-time support (stdlib, garbage collection)
- even SPECint95

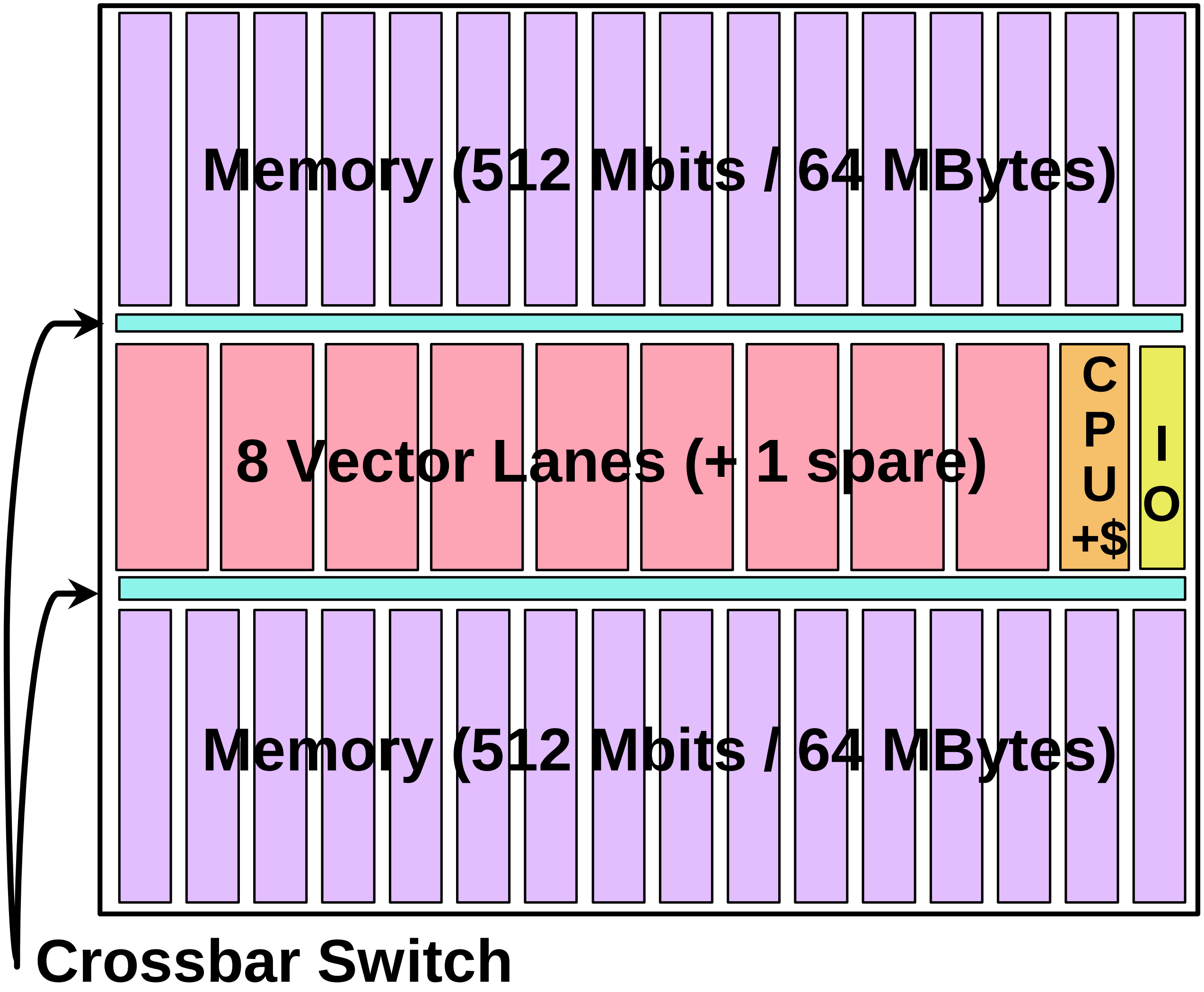
significant work by Krste Asanovic at UCB, other references available

Mediaprocesing Vector Lengths

Kernel	Vector length
● Matrix transpose/multiply	# vertices at once
● DCT (video, comm.)	image width
● FFT (audio)	256-1024
● Motion estimation (video)	image width, iw/16
● Gamma correction (video)	image width
● Haar transform (media mining)	image width
● Median filter (image process.)	image width
● Separable convolution (img. proc.)	image width

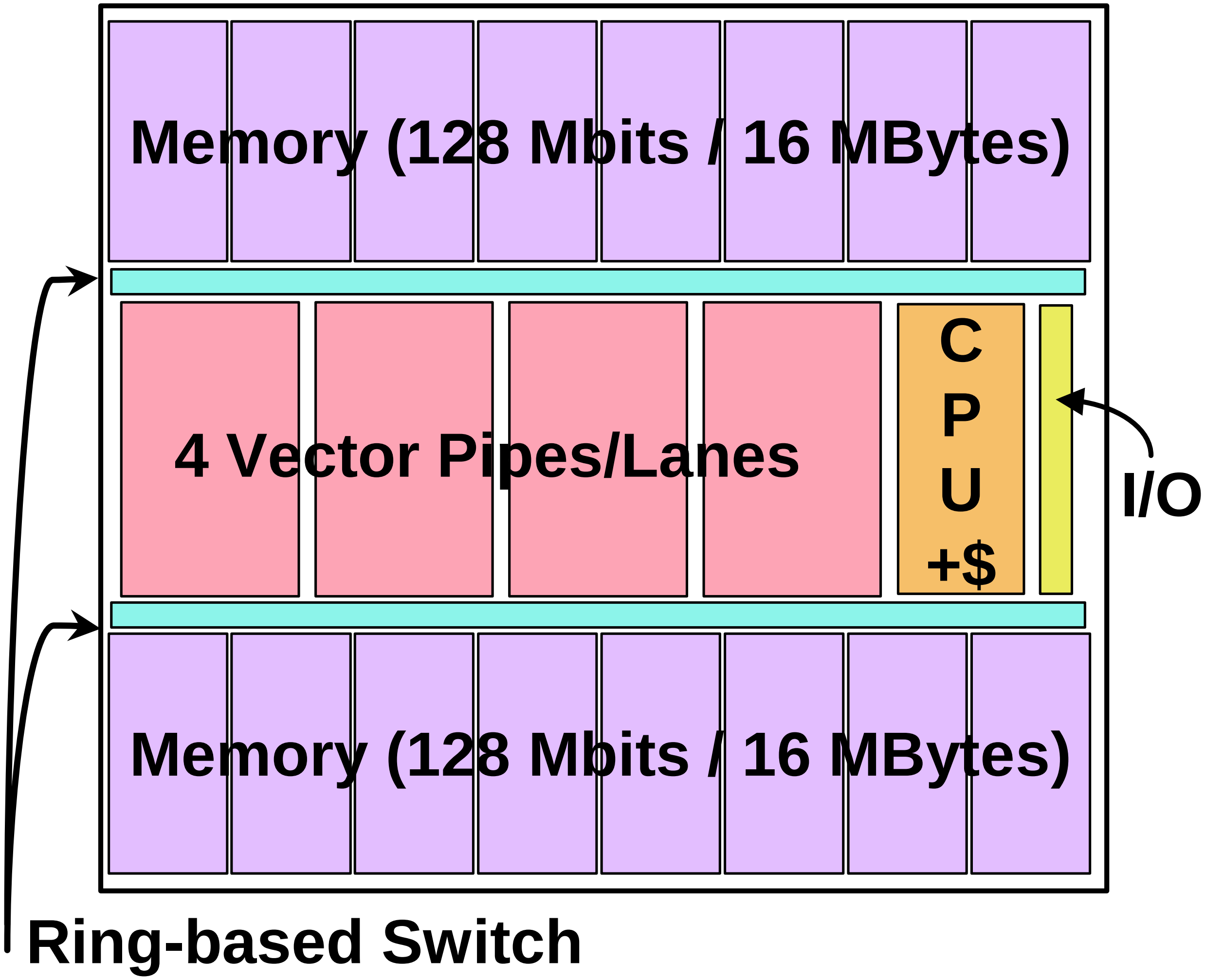
(from Pradeep Dubey - IBM,
`http://www.research.ibm.com/people/p/pradeep/tutor.html`)

V-IRAM-2 Floorplan


Memory (512 Mbits / 64 MBytes)
8 Vector Lanes (+ 1 spare)
CPU +$
IO
Memory (512 Mbits / 64 MBytes)
Crossbar Switch

- 0.13 µm, 1 Gbit DRAM
- >1B Xtors: 98% Memory, Xbar, Vector Þ regular design
- Spare Lane & Memory Þ 90% die repairable
- Short signal distance Þ speed scales <0.1 µm

Tentative V-IRAM-1 Floorplan


Memory (128 Mbits / 16 MBytes)
4 Vector Pipes/Lanes
C
P
U
+$
I/O
Memory (128 Mbits / 16 MBytes)
Ring-based Switch

- 0.18 µm DRAM,
  32 MB in
  16 banks x 256b,
  128 subbanks
- 0.25 µm,
  5 Metal Logic
- 200 MHz CPU
  4K I$, 4K D$
- 4 Float. Pt./Integer vector units
- die: 16 x 16 mm
- xtors: 270M
- power: ~2 Watts

What about I/O?

- Current system architectures have limitations
- I/O bus performance lags that of other system components
- Parallel I/O bus performance scaled by increasing clock speed and/or bus width
  - Eg. 32-bit PCI: ~50 pins; 64-bit PCI: ~90 pins
  - Greater number of pins ⇒ greater packaging costs
- Are there alternatives to parallel I/O buses for IRAM?

Serial I/O and IRAM

- Communication advances: fast (Gbps) serial I/O lines [YankHorowitz96], [DallyPoulton96]
  - Serial lines require 1-2 pins per unidirectional link
  - Access to standardized I/O devices
    - Fiber Channel-Arbitrated Loop (FC-AL) disks
    - Gbps Ethernet networks
- Serial I/O lines a natural match for IRAM
- Benefits
  - Serial lines provide high I/O bandwidth for I/O-intensive applications
  - I/O bandwidth incrementally scalable by adding more lines
    - Number of pins required still lower than parallel bus
- How to overcome limited memory capacity of single IRAM?
  - SmartSIMM: collection of IRAMs (and optionally external DRAMs)
  - Can leverage high-bandwidth I/O to compensate for limited memory

Example I/O-intensive Application: External (disk-to-disk) Sort

- Berkeley NOW cluster has world record sort: 8.6 GB disk-to-disk using 95 processors in 1 minute
- Balanced system ratios for processor:memory:I/O
  - Processor: N MIPS
  - Large memory: N Mbit/s disk I/O & 2N Mb/s Network
  - Small memory: 2N Mbit/s disk I/O & 2N Mb/s Network
- Serial I/O at 2-4 GHz today (v. 0.1 GHz bus)
- IRAM: 2-4 GIPS + 2 * 2-4Gb/s I/O + 2 * 2-4Gb/s Net
- ISIMM: 16 IRAMs + net switch + FC-AL links (+ disks)
- 1 IRAM sorts 9 GB, SmartSIMM sorts 100 GB

See “IRAM and SmartSIMM: Overcoming the I/O Bus Bottleneck”, *Workshop on Mixing Logic and DRAM, 24th Int’l Symp. on Computer Architecture*, June 1997

Why IRAM now?
Lower risk than before

- Faster Logic + DRAM available now/soon?
- DRAM manufacturers now willing to listen
  - Before not interested, so early IRAM = SRAM
- Past efforts memory limited ⇒ multiple chips
  ⇒ 1st solve the unsolved (parallel processing)
  - Gigabit DRAM ⇒ ~100 MB; OK for many apps?
- Systems headed to 2 chips: CPU + memory
- Embedded apps leverage energy efficiency, adjustable memory capacity, smaller board area
  ⇒ 115M embedded 32b RISC in 1996 [Microproc. Report]

IRAM Challenges

- Chip
  - Good performance and reasonable power?
  - Speed, area, power, yield, cost in DRAM process?
  - Testing time of IRAM vs. DRAM vs. microprocessor?
  - Bandwidth / Latency oriented DRAM tradeoffs?
  - Reconfigurable logic to make IRAM more generic?
- Architecture
  - How to turn high memory bandwidth into performance for real applications?
  - Extensible IRAM: Large program/data solution? (e.g., external DRAM, clusters, CC-NUMA, IDISK ...)

IRAM Conclusion

- IRAM potential in memory BW and latency, energy, board area, I/O
  - 10X-100X improvements based on technology shipping for 20 years (not JJ, photons, MEMS, ...)
- Challenges in power/performance, testing, yield
- Apps/metrics of future to design computer of future
- V-IRAM can show IRAM's potential
  - multimedia, energy, size, scaling, code size, compilers
- Shift semiconductor balance of power?

  Who ships the most memory? Most microprocessors?

Interested in Participating?

- Looking for more ideas of IRAM enabled apps
- Looking for possible MIPS scalar core
- Contact us if you're interested:

  http://iram.cs.berkeley.edu/

  rfromm@cs.berkeley.edu

  patterson@cs.berkeley.edu

- Thanks for advice/support:
  - DARPA, California MICRO, ARM, IBM, Intel, LG Semiconductor, Microsoft, Neomagic, Samsung, SGI/Cray, Sun Microsystems

Backup Slides

The following slides are used

to help answer questions,

and/or

go into more detail as time permits...

Today’s Situation: Microprocessor

MIPS μPs	R5000	R10000	10K/5K
Clock Rate	200 MHz	195 MHz	1.0x
On-Chip Caches	32K/32K	32K/32K	1.0x
Instructions/Cycle	1(+ FP)	4	4.0x
Pipe stages	5	5-7	1.2x
Model	In-order	Out-of-order	---
Die Size (mm^2)	84	298	3.5x
without cache, TLB	32	205	6.3x
Development (person-yr.)	60	300	5.0x
SPECint_base95	5.7	8.8	1.6x

Speed Differences Today...

- Logic gates currently slower in DRAM vs. logic processes
- Processes optimized for different characteristics
  - Logic: fast transistors and interconnect
  - DRAM: density and retention
- Precise slowdown varies by manufacturer and process generation

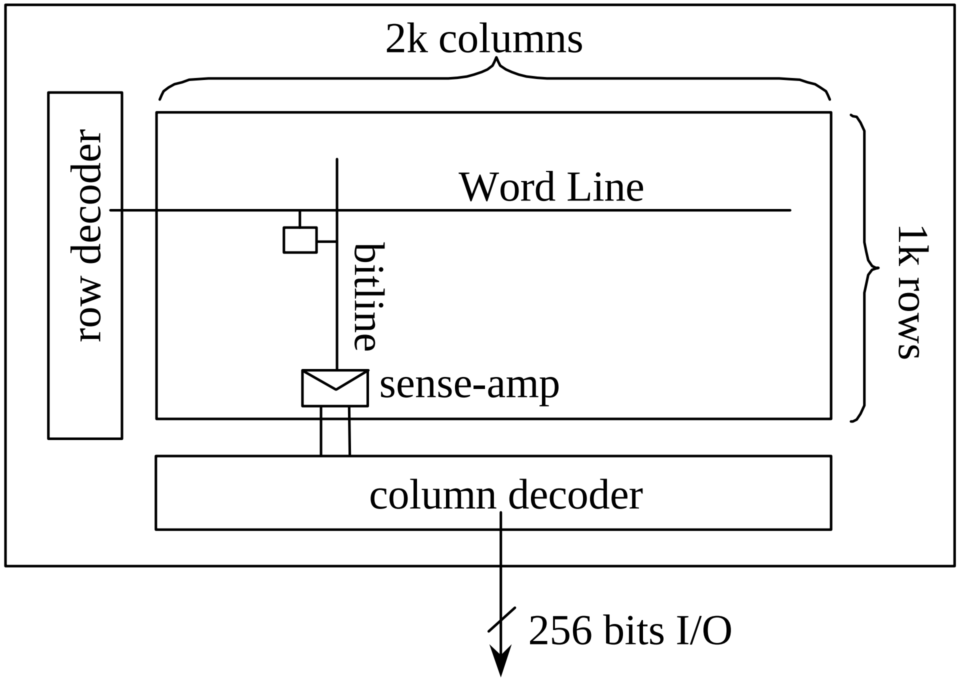
… and in the Future

- Ongoing trends in DRAM industry likely to alleviate current disadvantages
  - DRAM processes adding more metal layers
    - Enables more optimal layout of logic
  - Some DRAM manufacturers (Mitsubishi, Toshiba) already developing merged logic and DRAM processes
  - 1997 ISSCC panel predictions
    - Equal performance from logic transistors in DRAM process available soon
    - Modest (20-30%) increase in cost per wafer

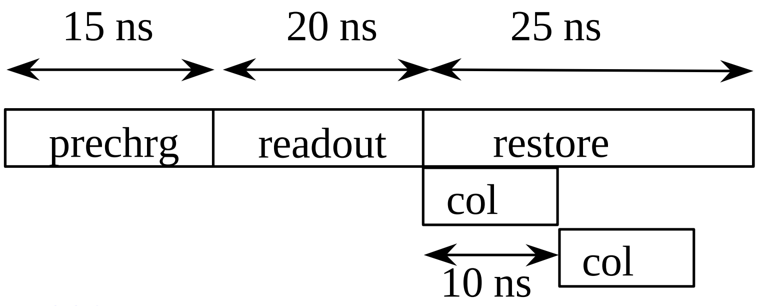
DRAM Access

Steps:

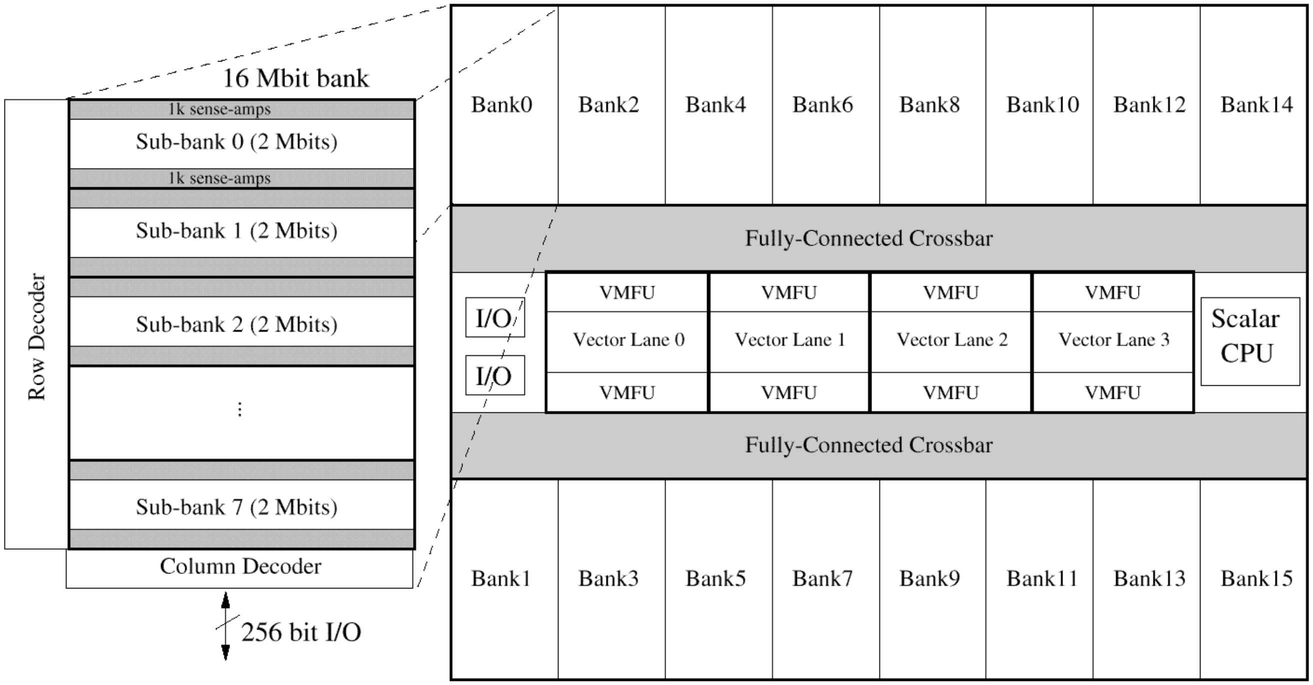
1. Precharge
2. Data-Readout
3. Data-Restore
4. Column Access



$$\text{energy}_{\text{row access}} = 5 \times \text{energy}_{\text{column access}}$$

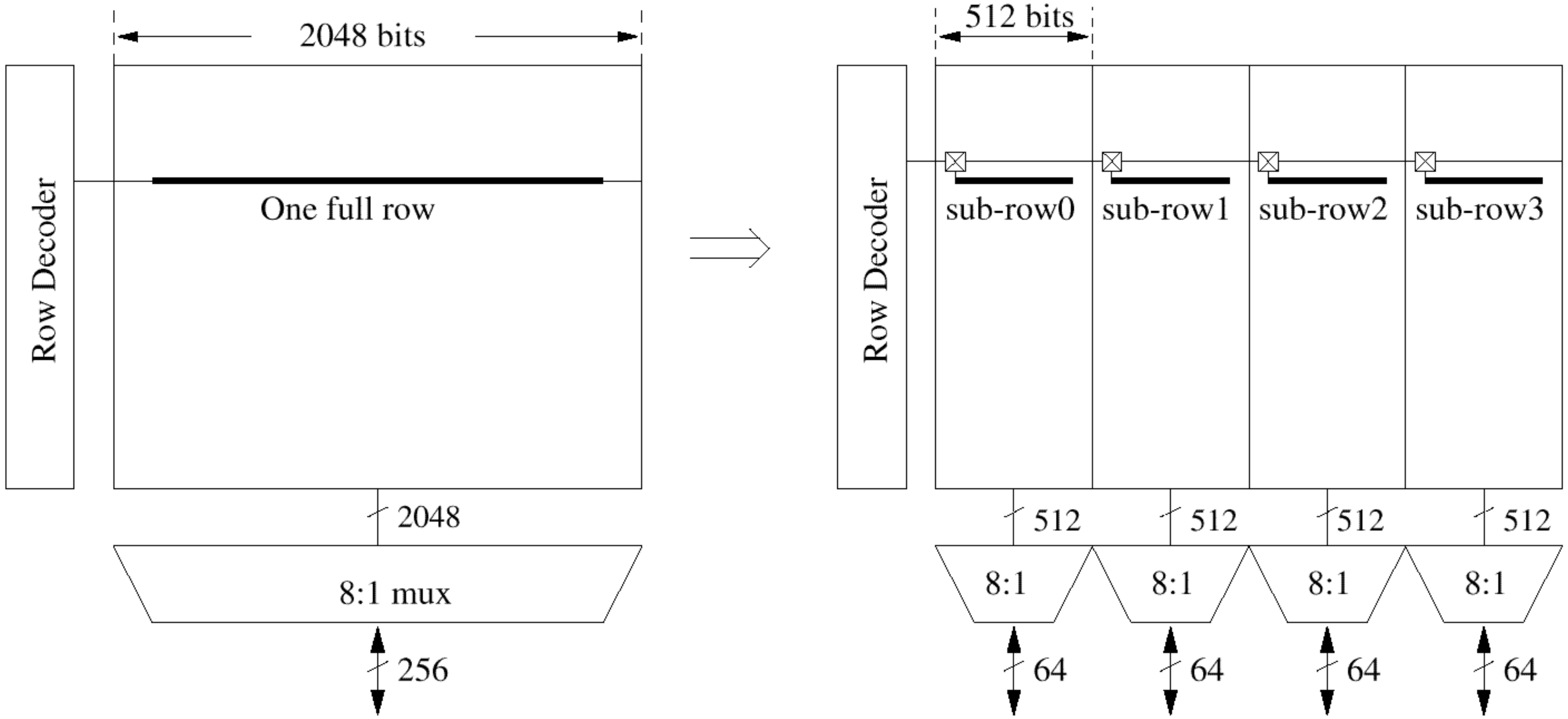


Possible DRAM Innovations #1



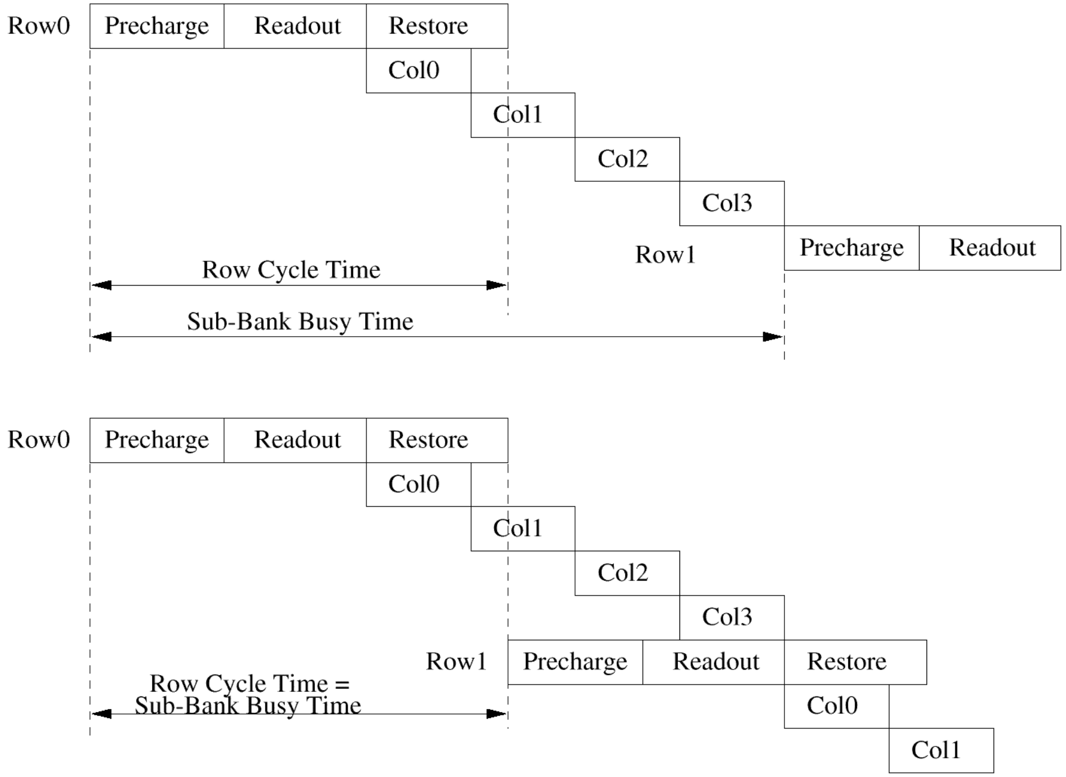
- More banks
  - Each bank can independently process a separate address stream
- Independent Sub-Banks
  - Hides memory latency
  - Increases effective cache size (sense-amps)

Possible DRAM Innovations #2



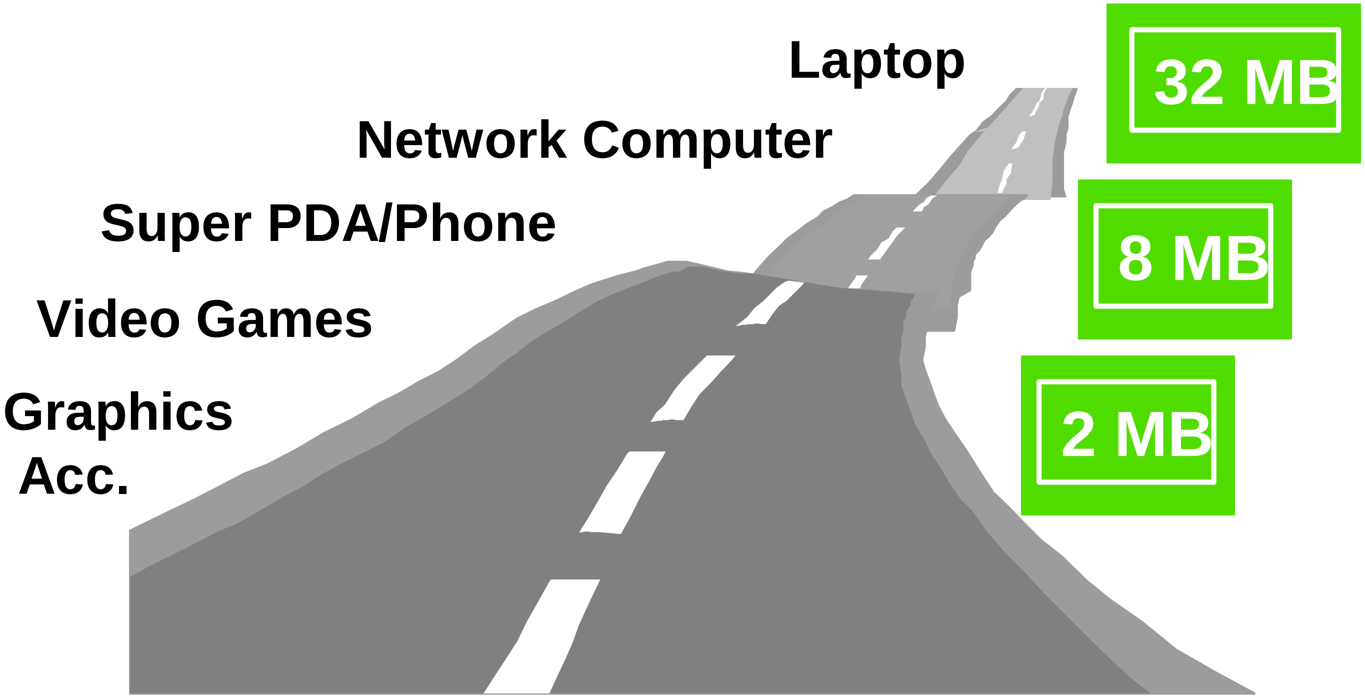
- Sub-rows
  - Save energy when not accessing all bits within a row

Possible DRAM Innovations #3



- Row buffers
  - Increase access bandwidth by overlapping precharge and read of next row access with col accss of prev row

Commercial IRAM highway is governed by memory per IRAM?



“Vanilla” Approach to IRAM

- Estimate performance IRAM implementations of conventional architectures
- Multiple studies:
  - #1: “Intelligent RAM (IRAM): Chips that remember and compute”, *1997 Int’l Solid-State Circuits Conf.*, Feb. 1997.
  - #2 & #3: “Evaluation of Existing Architectures in IRAM Systems”, *Workshop on Mixing Logic and DRAM, 24th Int’l Symp. on Computer Architecture*, June 1997.
  - #4: “The Energy Efficiency of IRAM Architectures”, *24th Int’l Symp. on Computer Architecture*, June 1997.

“Vanilla” IRAM - #1

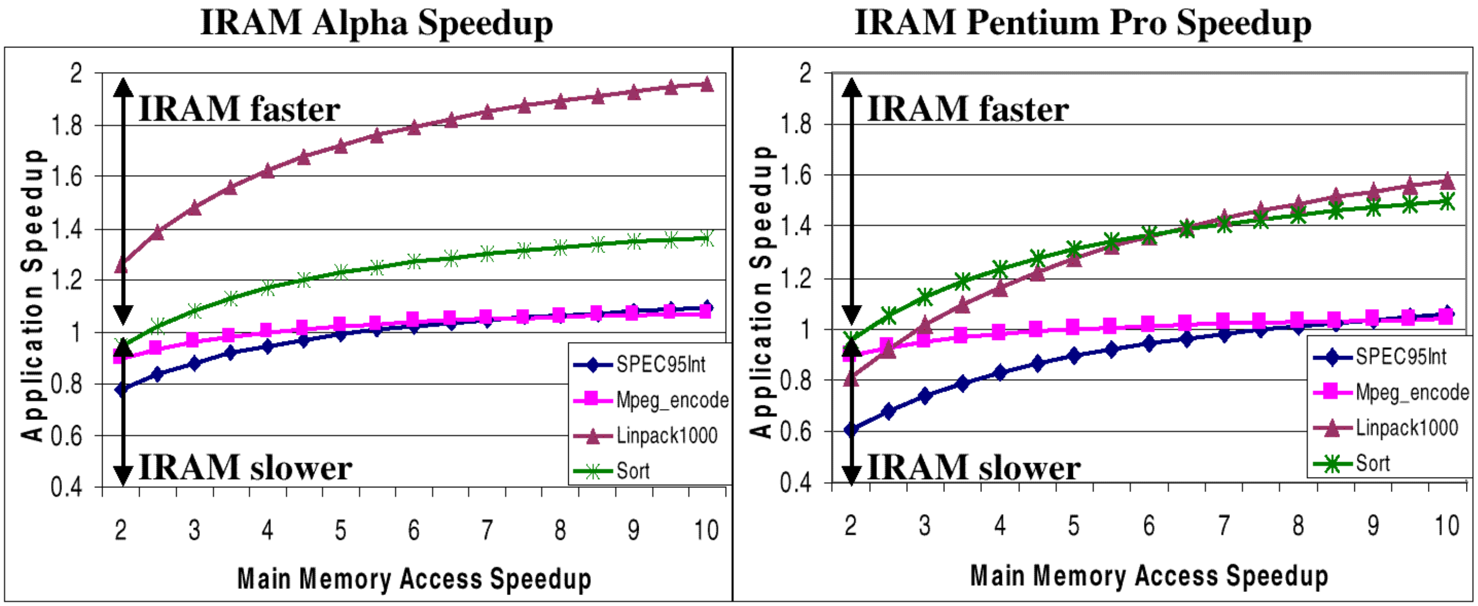
- Methodology
  - Estimate performance of IRAM implementation of Alpha architecture
    - Same caches, benchmarks, standard DRAM
  - Used optimistic and pessimistic factors for logic (1.3-2.0X slower), SRAM (1.1-1.3X slower), DRAM speed (5-10X faster) for standard DRAM
- Results
  - Spec92 benchmark Þ 1.2 to 1.8 times slower
  - Database Þ 1.1 times slower to 1.1 times faster
  - Sparse matrix Þ 1.2 to 1.8 times faster

“Vanilla” IRAM - Methodology #2

- Execution time analysis of a simple (Alpha 21064) and a complex (Pentium Pro) architecture to predict performance of similar IRAM implementations
- Used hardware counters for execution time measurements
- Benchmarks: spec95int, mpeg_encode, linpack1000, sort
- IRAM implementations: same architectures with 24 MB of on-chip DRAM but no L2 caches; all benchmarks fit completely in on-chip memory
- IRAM execution time model:

$$\text{Execution time} = \frac{\text{computation time}}{\text{clock speedup}} + \frac{\text{L1 miss count} * \text{memory access time}}{\text{memory access speedup}}$$

“Vanilla” IRAM - Results #2

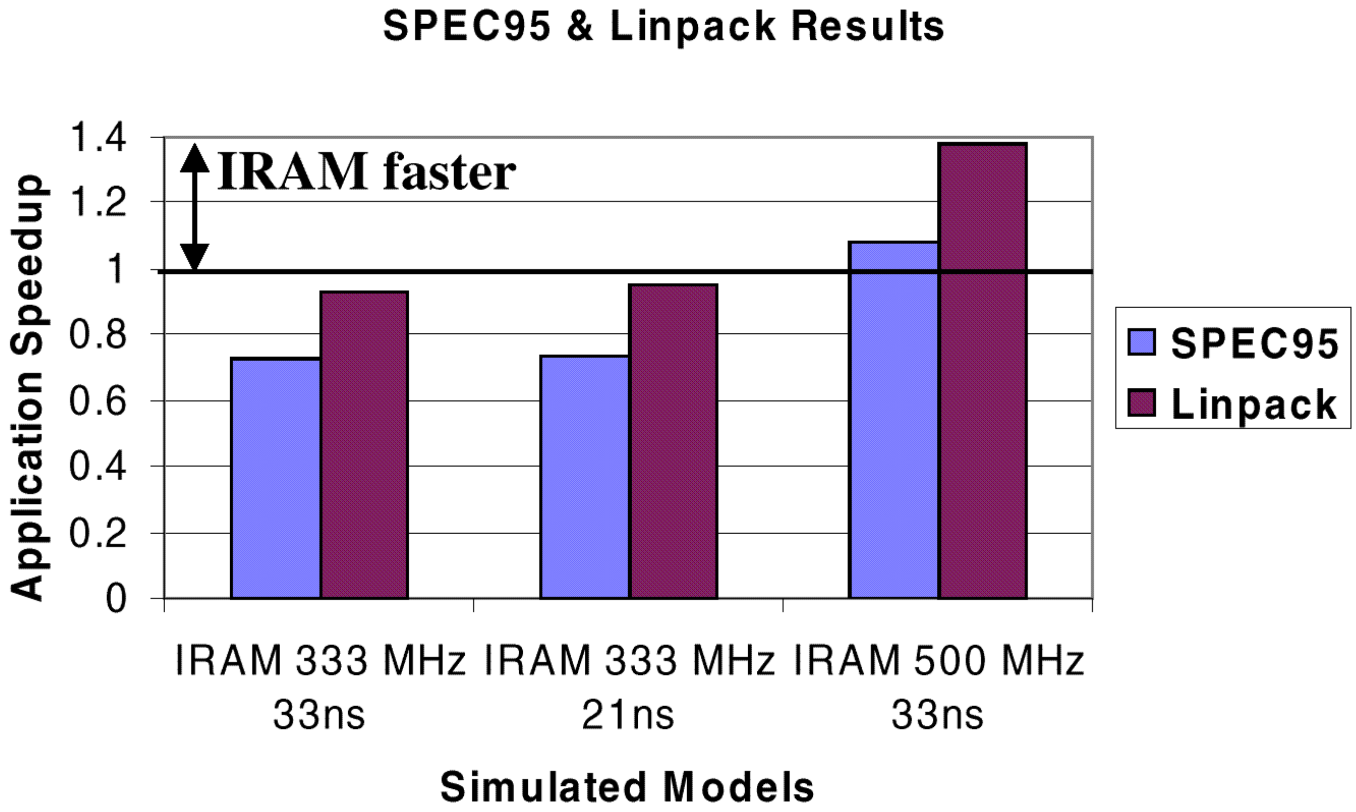


- Equal clock speeds assumed for conventional and IRAM systems
- Maximum IRAM speedup compared to conventional:
  - Less than 2 for memory bound applications
  - Less than 1.1 for CPU bound applications

“Vanilla” IRAM - Methodology #3

- Used SimOS to simulate simple MIPS R4000-based IRAM and conventional architectures
- Equal die size comparison
  - Are for on-chip DRAM in IRAM systems same as area for level 2 cache in conventional system
- Wide memory bus for IRAM systems
- Main simulation parameters
  - On-chip DRAM access latency
  - Logic speed (CPU frequency)
- Benchmarks: spec95int (compress, li, ijpeg, perl, gcc), spec95fp (tomcatv, su2cor, wave5), linpack1000

“Vanilla” IRAM - Results #3



- Maximum speedup of 1.4 for equal clock speeds
- Slower than conventional for most other cases

“Vanilla” IRAM - Methodology #4

- Architectural models
  - Simple CPU
  - IRAM and Conventional memory configurations for two different die sizes (Small ≈ StrongARM; Large ≈ 64 Mb DRAM)
- Record base CPI and activity at each level of memory hierarchy with shade
- Estimate performance based on access CPU speed and access times to each level of memory hierarchy
- Benchmarks: hsfsys, noway, nowsort, gs, ispell, compress, go, perl

“Vanilla” IRAM - Results #4

- Speedup of IRAM compared to Conventional
  - Higher numbers mean higher performance for IRAM

Model	Small-IRAM		Large-IRAM	
Clock rate	**(0.75 X)**	**(1.0 X)**	**(0.75 X)**	**(1.0 X)**
hsfsys	0.81	1.08	0.77	1.02
noway	0.89	1.19	0.82	1.09
nowsort	0.95	1.27	0.81	1.08
gs	0.90	1.20	0.78	1.04
ispell	0.78	1.04	0.77	1.03
compress	1.13	1.50	0.82	1.09
go	0.99	1.31	0.76	1.02
perl	0.78	1.04	0.76	1.01

- Performance is comparable: IRAM is 0.76 to 1.50 times as fast as conventional
  - Dependent on memory behavior of application

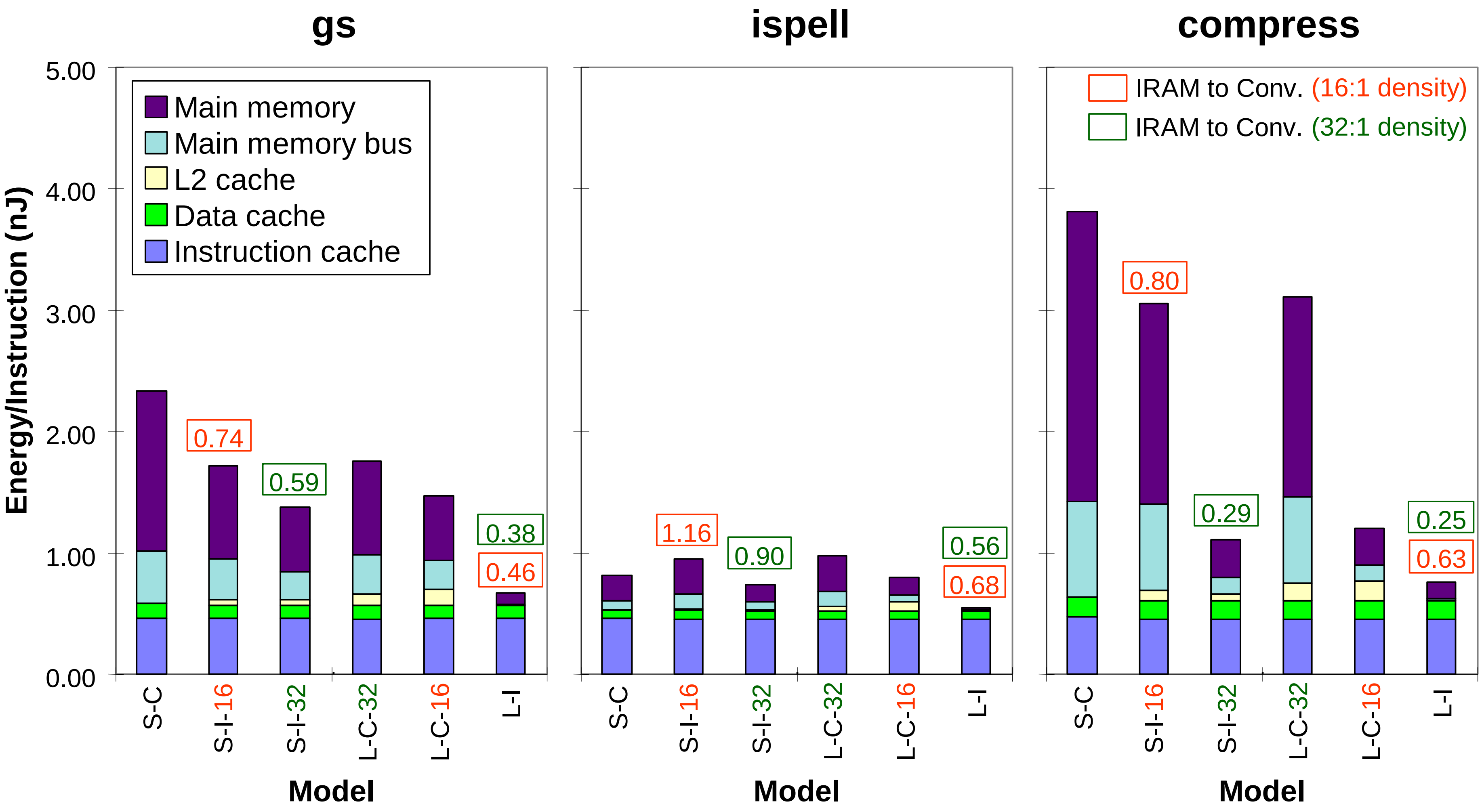
Frequency of Accesses

- On-chip DRAM array has **much** higher capacity than SRAM array of same area
- IRAM reduces the frequency of accesses to lower levels of memory hierarchy, which require more energy
  - On-chip DRAM organized as L2 cache has lower off-chip miss rates than L2 SRAM, reducing the off-chip energy penalty
  - When entire main memory array is on-chip, high off-chip energy cost is avoided entirely

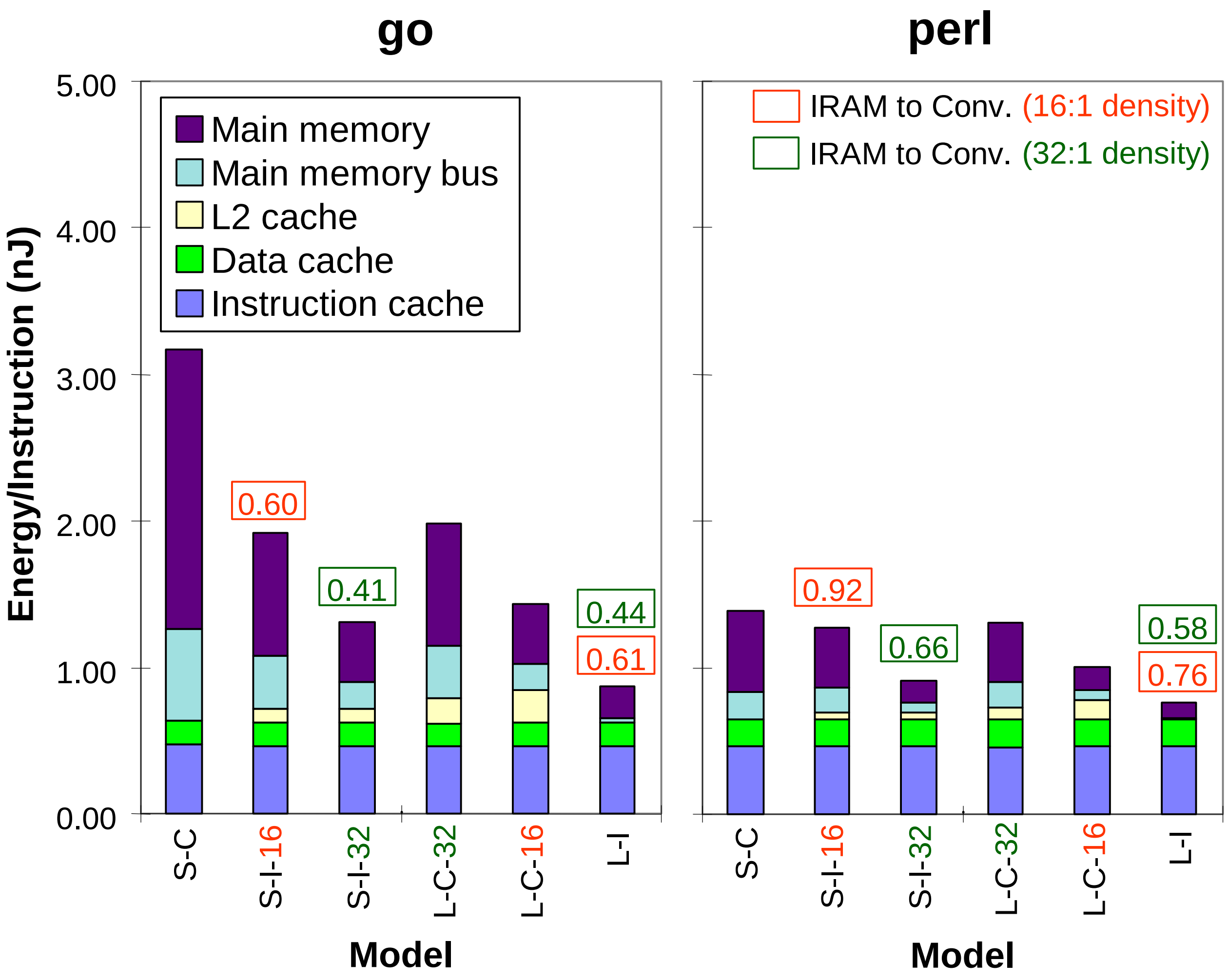
Energy of Accesses

- IRAM reduces energy to access various levels of the memory hierarchy
  - On-chip memory accesses use less energy than off-chip accesses by avoiding high-capacitance off-chip bus
  - Multiplexed address scheme of conventional DRAMs selects larger number of DRAM arrays than necessary
  - Narrow pin interface of external DRAM wastes energy in multiple column cycles needed to fill entire cache block

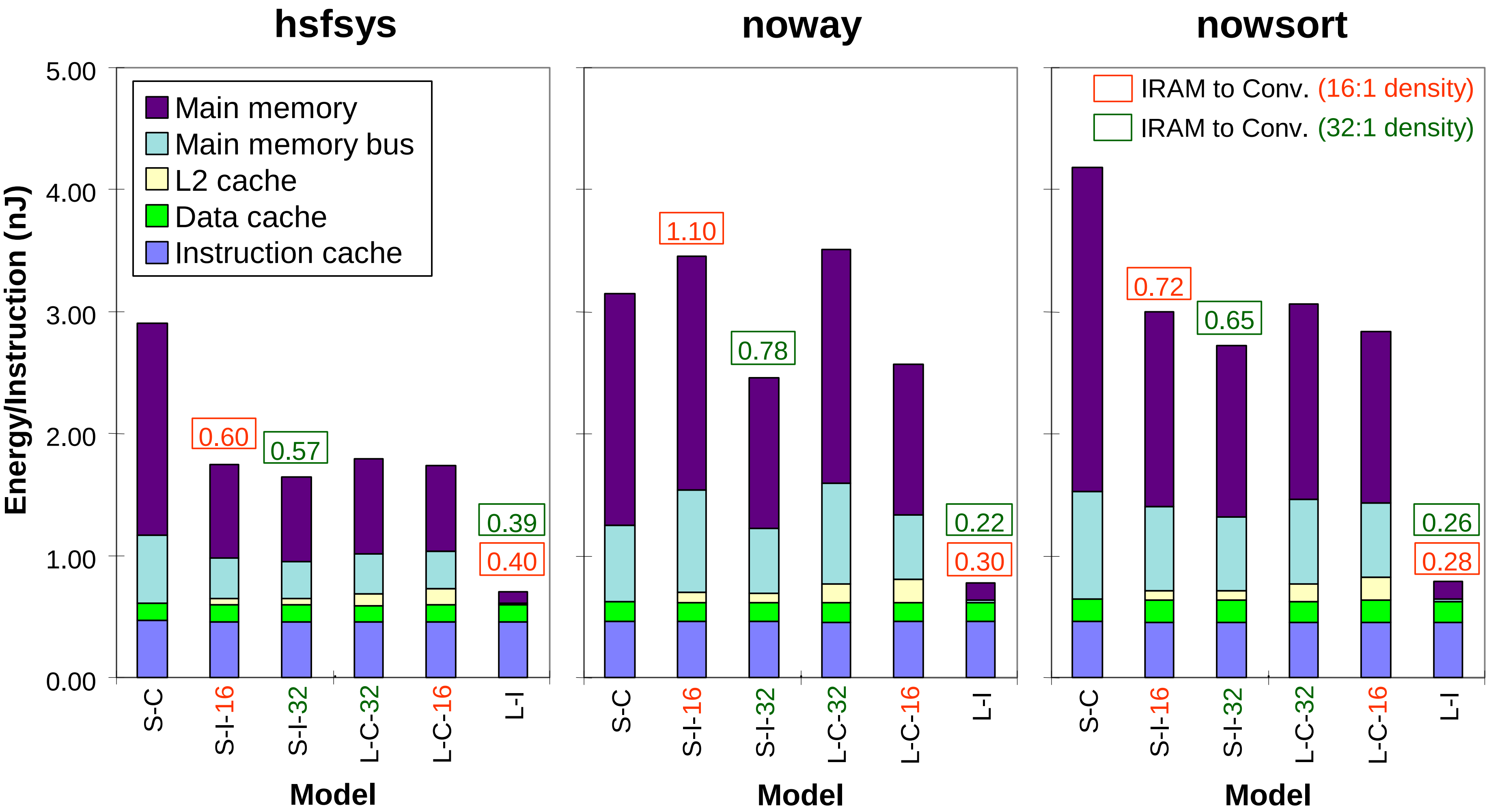
Energy Results 1/3



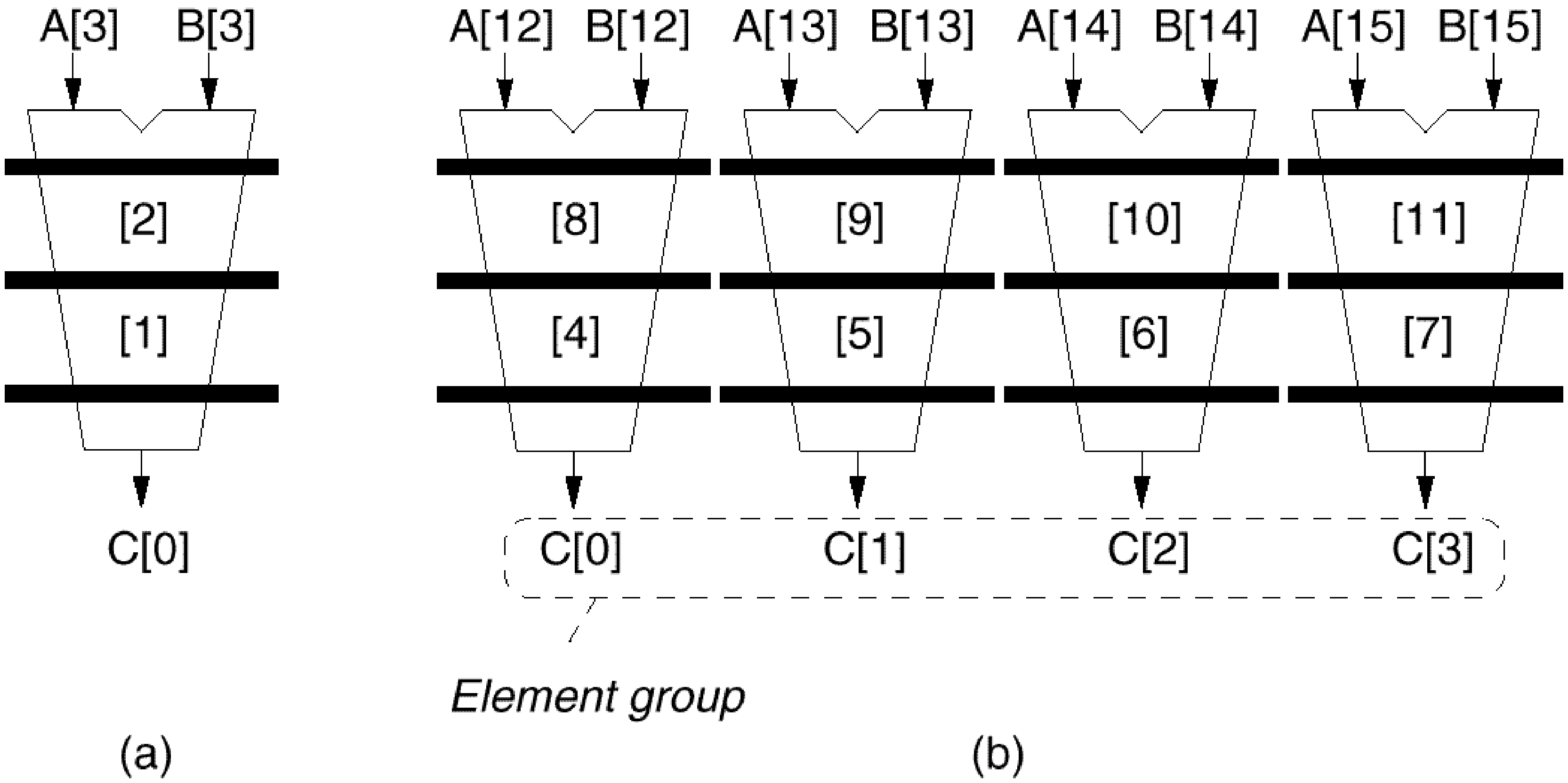
Energy Results 2/3



Energy Results 3/3



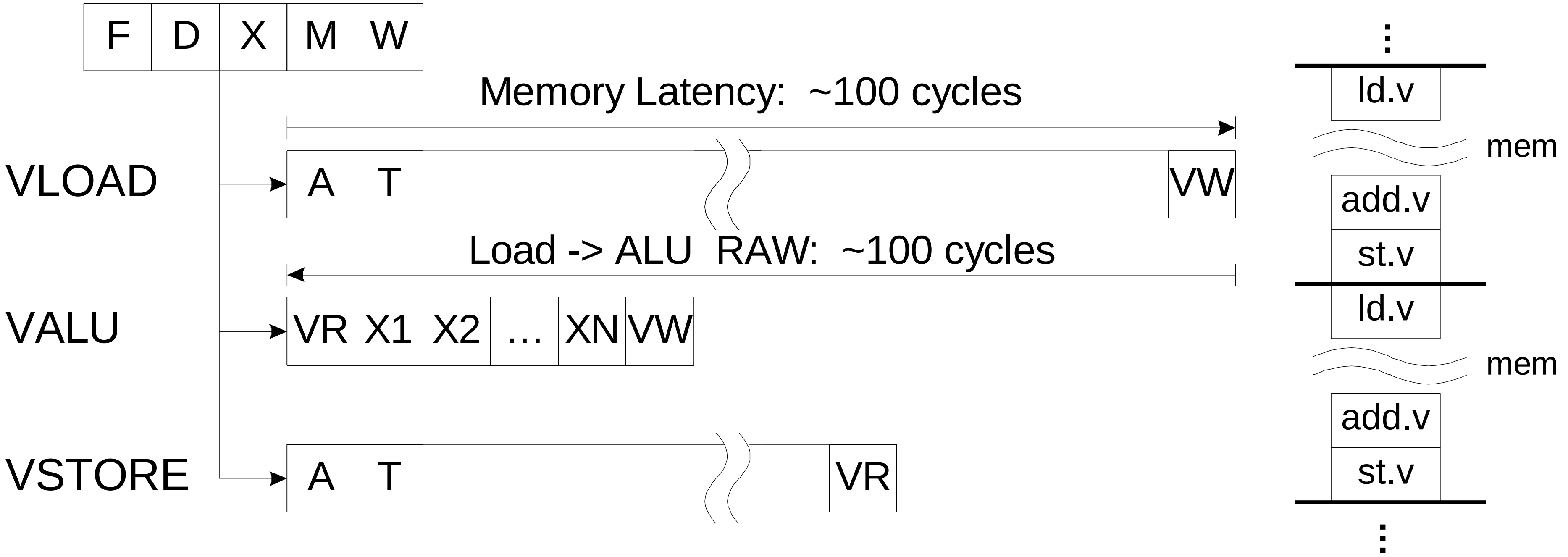
Parallel Pipelines in Functional Units



One per cycle

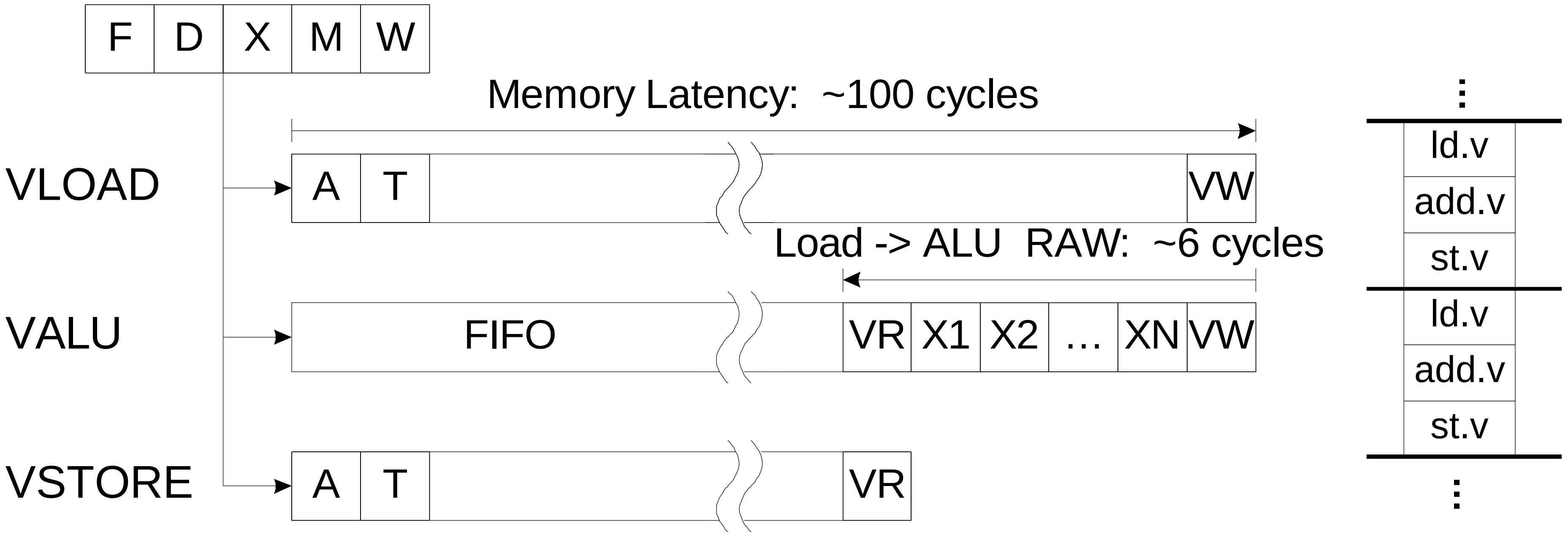
Four per cycle

Tolerating Memory Latency Non-Delayed Pipeline



- Load → ALU sees full memory latency (large)

Tolerating Memory Latency
Delayed Pipeline



- Delay ALU instructions until memory data returns
- Load → ALU sees functional unit latency (small)

Latency not always hidden...

- Scalar reads of vector unit state
  - Element reads for partially vectorized loops
  - Count trailing zeros in flags
  - Pop count of flags
- Indexed vector loads and stores
  - Need to get address from register file to address generator
- Masked vector loads and stores
  - Mask values from end of pipeline to address translation stage to cancel exceptions

Standard Benchmark Kernels

- Matrix Multiply (and other BLAS)
  - "Implementation of level 2 and level 3 BLAS on the Cray Y-MP and Cray-2", Sheikh et al, *Journal of Supercomputing*, 5:291-305
- FFT (1D, 2D, 3D, ...)
  - "A High-Performance Fast Fourier Transform Algorithm for the Cray-2", Bailey, *Journal of Supercomputing*, 1:43-60
- Convolutions (1D, 2D, ...)
- Sorting
  - "Radix Sort for Vector Multiprocessors", Zagha and Blelloch, *Supercomputing 91*

Compression

- Lossy
  - JPEG
    - source filtering and down-sample
    - YUV ↔ RGB color space conversion
    - DCT/iDCT
    - run-length encoding
  - MPEG video
    - Motion estimation (Cedric Krumbein, UCB)
  - MPEG audio
    - FFTs, filtering

- Lossless
  - Zero removal
  - Run-length encoding
  - Differencing
  - JPEG lossless mode
  - LZW

Cryptography

- RSA (public key)
  - Vectorize long integer arithmetic
- DES/IDEA (secret key ciphers)
  - ECB mode encrypt/decrypt vectorizes
  - IDEA CBC mode encrypt doesn't vectorize (without interleave mode)
  - DES CBC mode encrypt can vectorize S-box lookups
  - CBC mode decrypt vectorizes

IDEA mode	ECB	CBC enc.	CBC dec.
	(MB/s)	(MB/s)	(MB/s)
T0 (40 MHz)	14.04	0.70	13.01
Ultra-1/170 (167 MHz)	1.96	1.85	1.91
Alpha 21164 (500 MHz)	4.01		

- SHA/MD5 (signature)
  - Partially vectorizable

Multimedia Processing

- Image/video/audio compression (JPEG/MPEG/GIF/png)
- Front-end of 3D graphics pipeline (geometry, lighting)
  - Pixar Cray X-MP, Stellar, Ardent, Microsoft Talisman MSP
- High Quality Additive Audio Synthesis
  - Todd Hodes, UCB
  - Vectorize across oscillators
- Image Processing
  - Adobe Photoshop

Speech and Handwriting Recognition

- Speech recognition
  - Front-end: filters/FFTs
  - Phoneme probabilities: Neural net
  - Back-end: Viterbi/Beam Search
- Newton handwriting recognition
  - Front-end: segment grouping/segmentation
  - Character classification: Neural net
  - Back-end: Beam Search
- Other handwriting recognizers/OCR systems
  - Kohonen nets
  - Nearest exemplar

Operating Systems / Networking

- Copying and data movement (`memcpy`)
- Zeroing pages (`memset`)
- Software RAID parity XOR
- TCP/IP checksum (Cray)
- RAM compression (Rizzo ‘96, zero-removal)

Databases

- Hash/Join (Rich Martin, UCB)
- Database mining
- Image/video serving
  - Format conversion
  - Query by image content

Language Run-time Support

- Structure copying
- Standard C libraries: `mem*, str*`
  - Dhrystone 1.1 on T0: 1.98 speedup with vectors
  - Dhrystone 2.1 on T0: 1.63 speedup with vectors
- Garbage Collection
  - “Vectorized Garbage Collection”, Appel and Bendiksen, *Journal Supercomputing, 3:151-160*
  - Vector GC 9x faster than scalar GC on Cyber 205

SPECint95

- `m88ksim` - 42% speedup with vectorization
- `compress` - 36% speedup for decompression with vectorization (including code modifications)
- `ijpeg` - over 95% of runtime in vectorizable functions
- `li` - approx. 35% of runtime in mark/scan garbage collector
  - Previous work by Appel and Bendiksen on vectorized GC
- `go` - most time spent in linke list manipulation
  - could rewrite for vectors?
- `perl` - mostly non-vectorizable, but up to 10% of time in standard library functions (`str*, mem*`)
- `gcc` - not vectorizable
- `vortex` - ???
- `eqntott` (from SPECint92) - main loop (90% of runtime) vectorized by Cray C compiler

V-IRAM-1 Specs/Goals

Technology	0.18-0.20 micron, 5-6 metal layers, fast transistor	
Memory	32 MB	
Die size	~250 mm^2	
Vector lanes	4 x 64-bit (or 8 x 32-bit or 16 x 16-bit)	
Target	Low Power	High Performance
Serial I/O	4 lines @ 1 Gbit/s	8 lines @ 2 Gbit/s
Power	~2 W @ 1-1.5 Volt logic	~10 W @ 1.5-2 Volt logic
Clock$_{\text{university}}$	200 scalar / 100 vector MHz	250 scalar / 250 vector MHz
Perf$_{\text{university}}$	0.8 GFLOPS$_{64}$-3 GFLOPS$_{16}$	2 GFLOPS$_{64}$-8 GFLOPS$_{16}$
Clock$_{\text{industry}}$	400 scalar / 200 vector MHz	500 scalar / 500 vector MHz
Perf$_{\text{industry}}$	1.6 GFLOPS$_{64}$-6 GFLOPS$_{16}$	4 GFLOPS$_{64}$-16 GFLOPS$_{16}$

How to get Low Power, High Clock rate IRAM?

- Digital Strong ARM 110 (1996): 2.1M Xtors
  - 160 MHz @ 1.5 v = 184 “MIPS” < 0.5 W
  - 215 MHz @ 2.0 v = 245 “MIPS” < 1.0 W
- Start with Alpha 21064 @ 3.5v, 26 W
  - Vdd reduction ⇒ 5.3X ⇒ 4.9 W
  - Reduce functions ⇒ 3.0X ⇒ 1.6 W
  - Scale process ⇒ 2.0X ⇒ 0.8 W
  - Clock load ⇒ 1.3X ⇒ 0.6 W
  - Clock rate ⇒ 1.2X ⇒ 0.5 W
- 6/97: 233 MHz, 268 MIPS, 0.36W typ., $49

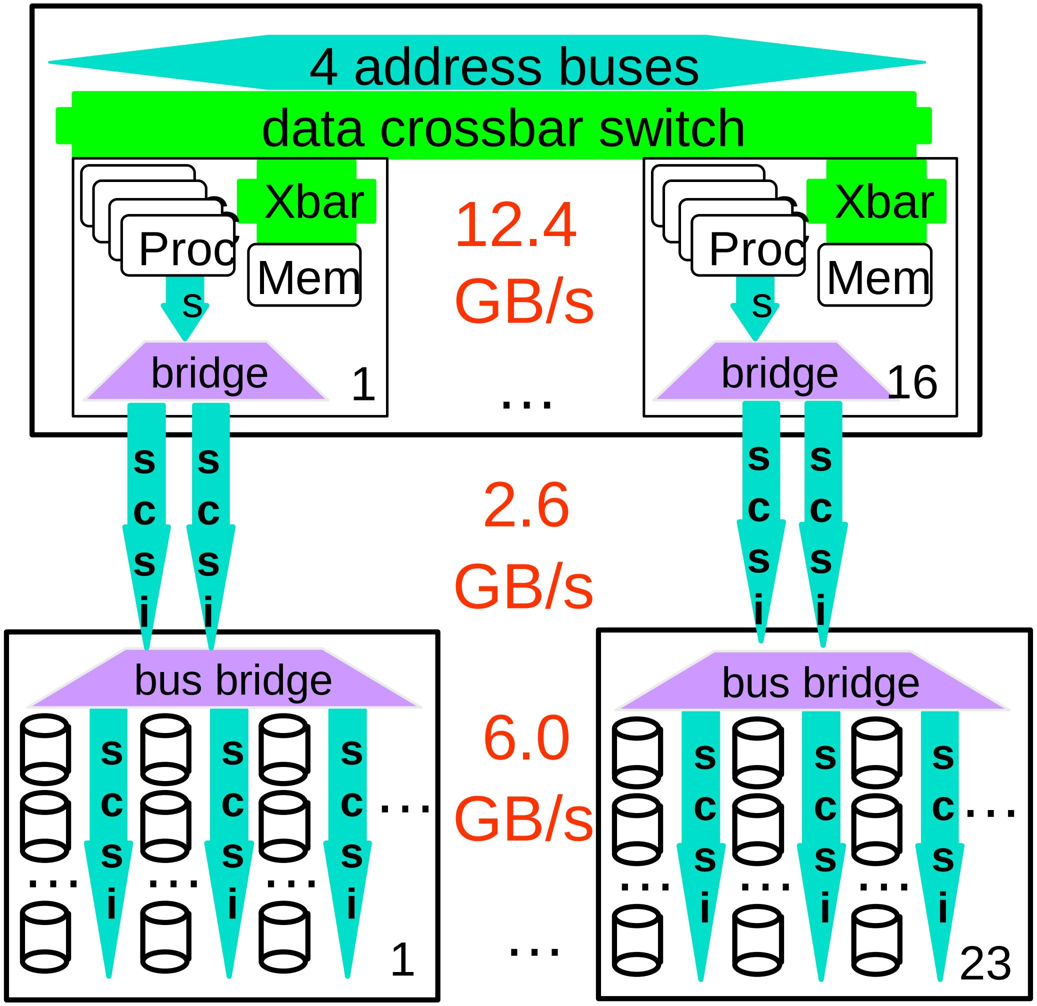
Serial I/O

- Communication advances: fast (Gbps) serial I/O lines [YankHorowitz96], [DallyPoulton96]
  - Serial lines require 1-2 pins per unidirectional link
  - Access to standardized I/O devices
    - Fiber Channel-Arbitrated Loop (FC-AL) disks
    - Gbps Ethernet networks
- Serial I/O lines a natural match for IRAM
- Benefits
  - Avoids large number of pins for parallel I/O buses
  - IRAM can sink high I/O rate without interfering with computation
  - “System-on-a-chip” integration means chip can decide how to:
    - Notify processor of I/O events
    - Keep caches coherent
    - Update memory

Serial I/O and IRAM

- How well will serial I/O work for IRAM?
  - Serial lines provide high I/O bandwidth for I/O-intensive applications
  - I/O bandwidth incrementally scalable by adding more lines
    - Number of pins required still lower than parallel bus
- How to overcome limited memory capacity of single IRAM?
  - SmartSIMM: collection of IRAMs (and optionally external DRAMs)
  - Can leverage high-bandwidth I/O to compensate for limited memory
- In addition to other strengths, IRAM with serial lines provides high I/O bandwidth

Another Application: Decision Support (Conventional)

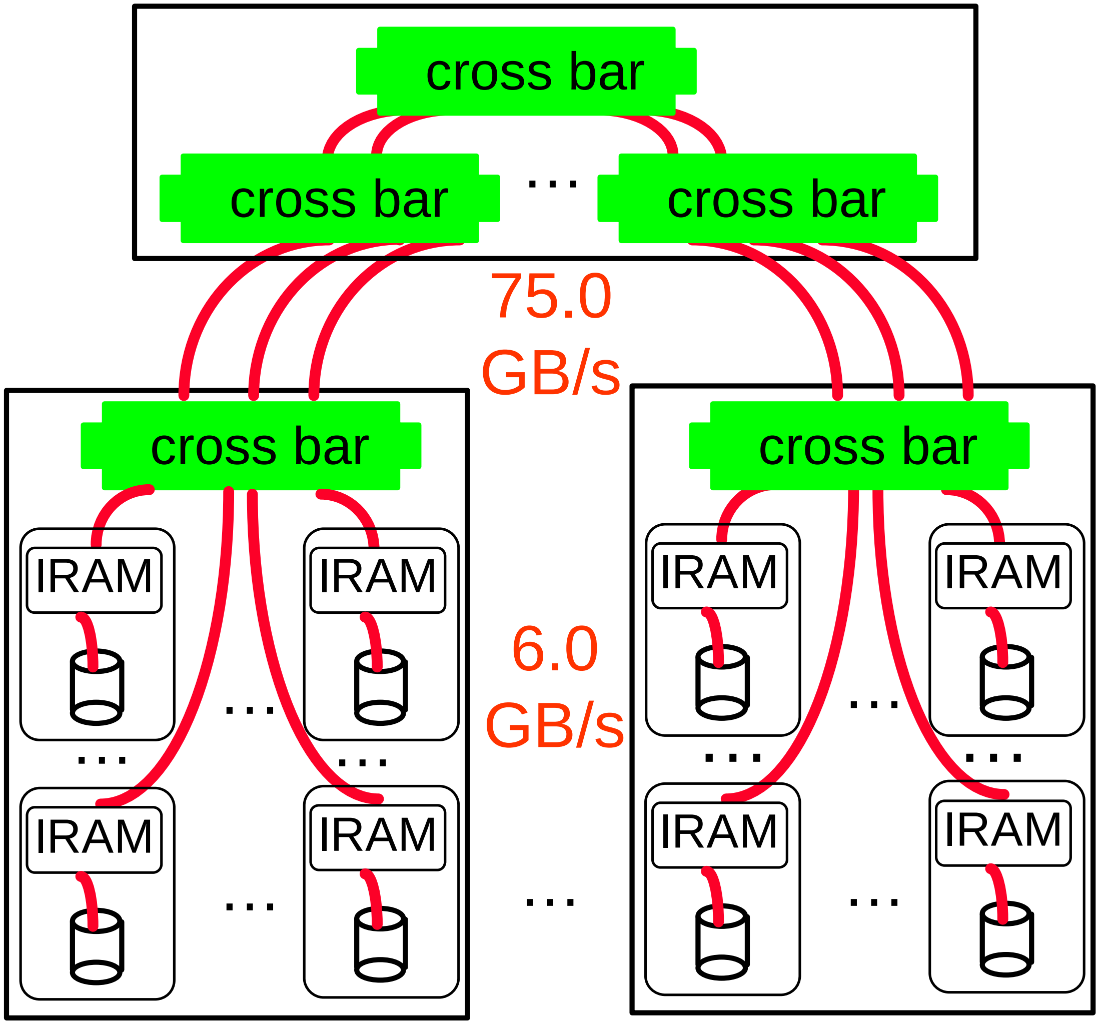


Sun 10000 (Oracle 8):

- TPC-D (1TB) leader
- SMP 64 CPUs, 64GB dram, 603 disks

Disks,encl.	$2,348k
DRAM	$2,328k
Boards,encl.	$983k
CPUs	$912k
Cables,I/O	$139k
Misc	$65k
HW total	$6,775k

“Intelligent Disk”: Scalable Decision Support



1 IRAM/disk + shared nothing database

- 603 CPUs, 14GB dram, 603 disks

Disks (market)	$840k
IRAM (@$150)	$90k
Disk encl., racks	$150k
Switches/cables	$150k
Misc	$60k
Subtotal	$1,300k
Markup 2X?	~$2,600k

~1/3 price, 2X-5X perf.

Richard Fromm, IRAM tutorial, ASP-DAC ‘98, February 10, 1998

Testing in DRAM

- Importance of testing over time
  - Testing time affects time to qualification of new DRAM, time to First Customer Ship
  - Goal is to get 10% of market by being one of the first companies to FCS with good yield
  - Testing 10% to 15% of cost of early DRAM
- Built In Self Test of memory:
  BiST v. External tester?
  Vector Processor 10X v. Scalar Processor?
- System v. component may reduce testing cost

Operation & Instruction Count: RISC v. Vector Processor

Spec92fp	Operations (M)			Instructions (M)		
Program	RISC	Vector	R / V	RISC	Vector	R / V
swim256	115	95	1.1x	115	0.8	142x
hydro2d	58	40	1.4x	58	0.8	71x
nasa7	69	41	1.7x	69	2.2	31x
su2cor	51	35	1.4x	51	1.8	29x
tomcatv	15	10	1.4x	15	1.3	11x
wave5	27	25	1.1x	27	7.2	4x
mdljdp2	32	52	0.6x	32	15.8	2x

Vectors reduce ops by 1.2X, instructions by 20X!

from F. Ouintana, University of Barcelona

V-IRAM-1 Tentative Plan

- Phase I: Feasibility stage (H1’98)
  - Test chip, CAD agreement, architecture defined
- Phase 2: Design Stage (H2’98)
  - Simulated design
- Phase 3: Layout & Verification (H2’99)
  - Tape-out
- Phase 4: Fabrication,Testing, and Demonstration (H1’00)
  - Functional integrated circuit
- First microprocessor ≈ 250M transistors!