An Introduction to Intelligent RAM (IRAM)

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

patterson@cs.berkeley.edu

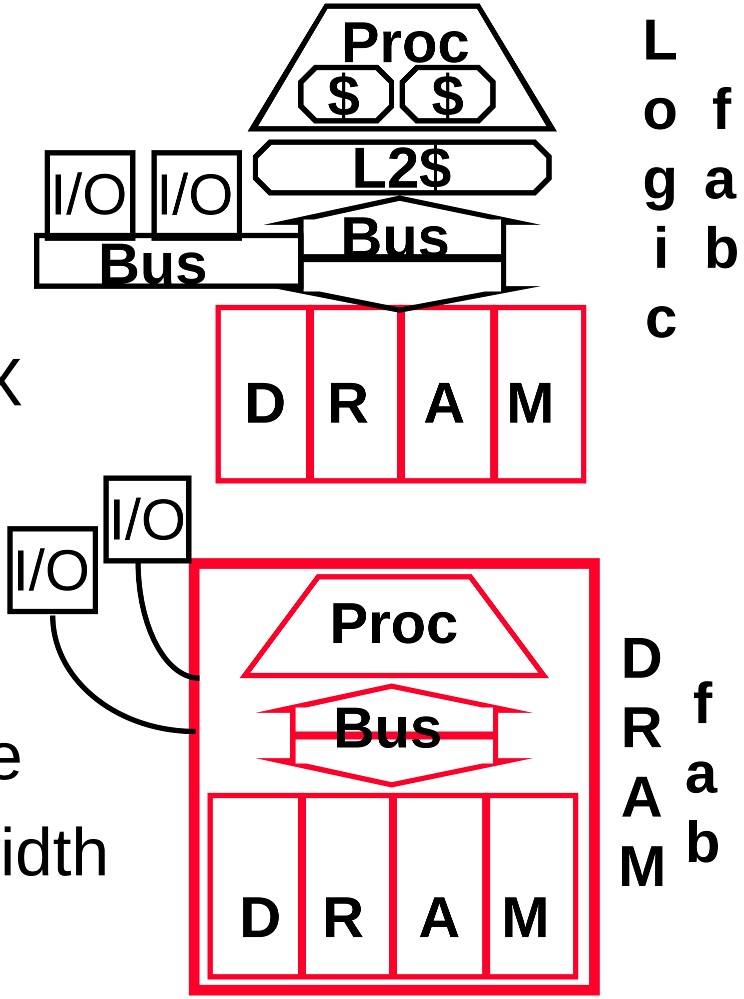
http://iram.cs.berkeley.edu/

EECS, University of California
Berkeley, CA 94720-1776

IRAM Vision Statement

Microprocessor & DRAM on a single chip:

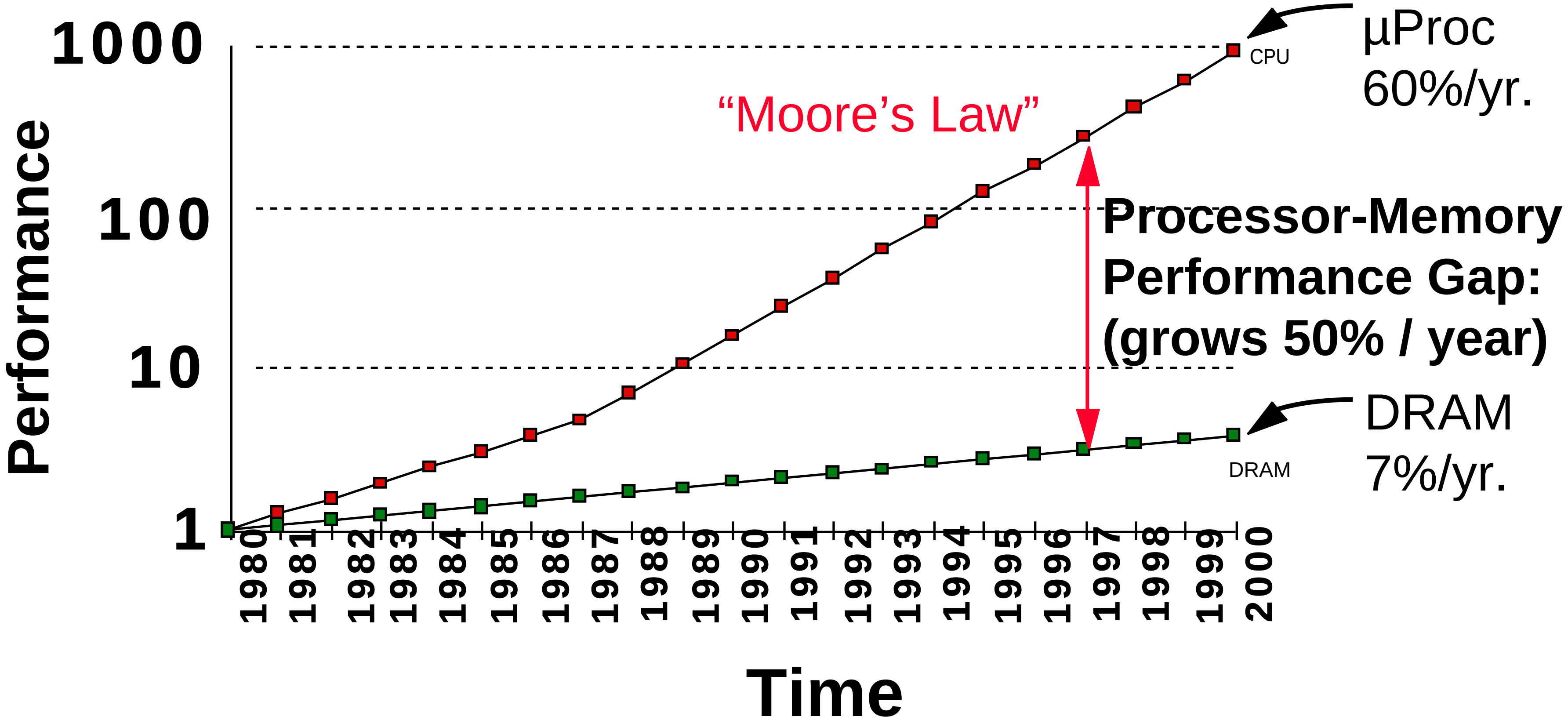
- on-chip memory latency 5-10X, bandwidth 50-100X
- improve energy efficiency 2X-4X (no off-chip bus)
- serial I/O 5-10X v. buses
- smaller board area/volume
- adjustable memory size/width



Outline

- Today’s Situation: Microprocessor & DRAM
- Potential of IRAM
- Applications of IRAM
- Grading New Instruction Set Architectures
- Berkeley IRAM Instruction Set Overview
- Berkeley IRAM Project Plans
- Related Work and Why Now?
- IRAM Challenges & Industrial Impact

Processor-DRAM Gap (latency)



µProc
60%/yr.

"Moore's Law"

Processor-Memory
Performance Gap:
(grows 50% / year)

DRAM
7%/yr.

Processor-Memory Performance Gap “Tax”

Processor	% Area	%Transistors
	(≈cost)	*(≈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

MIPS MPUs	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 (man yr.)	60	300	5.0x
■ SPECint_base95	5.7	8.8	1.6x

Today's Situation: Microprocessor

- 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

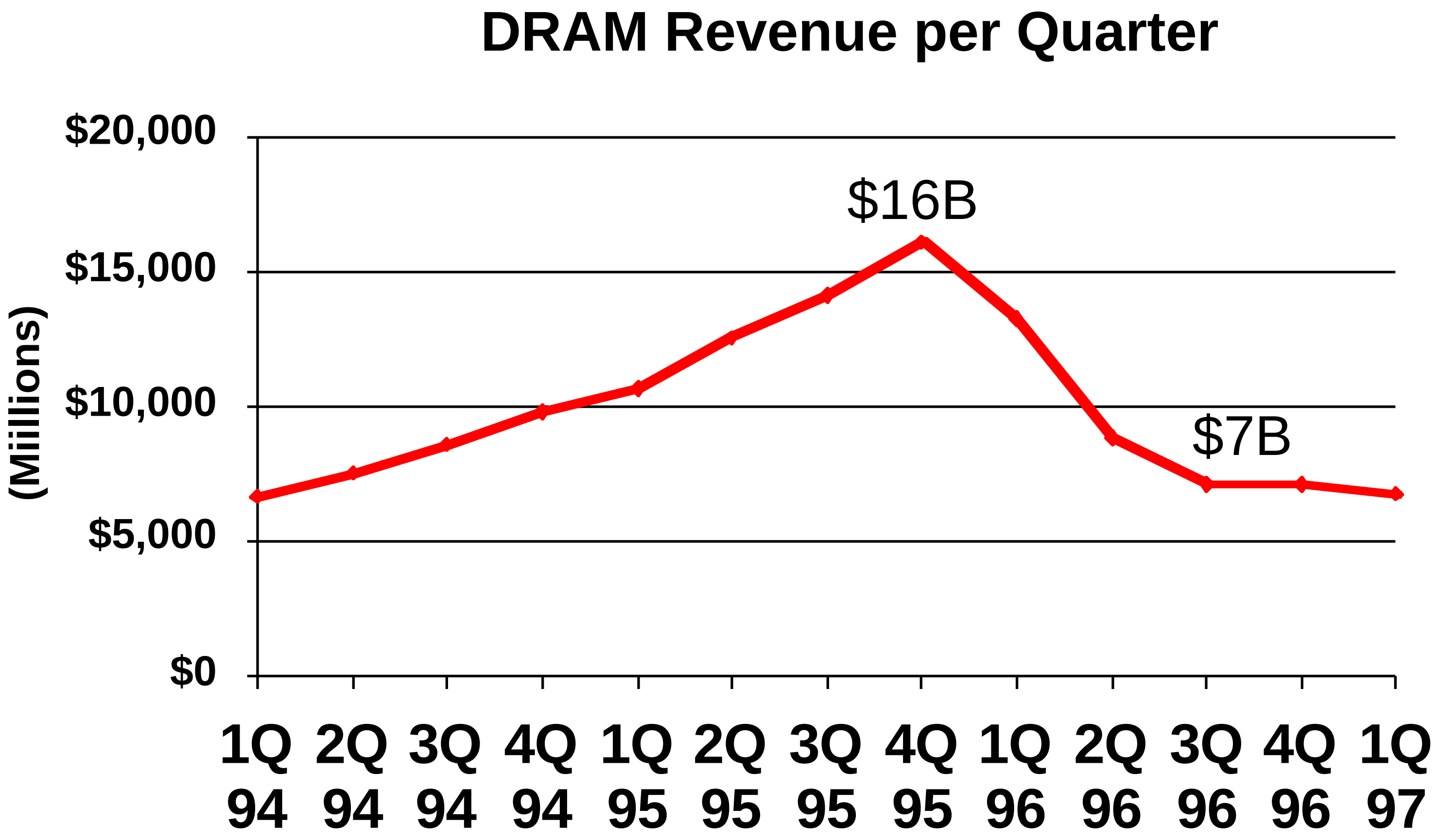
  2nd Alpha (8400): 266 ns/3.3 ns = 80 clks x 4 or 320

  3rd Alpha (t.b.d.): 180 ns/1.7 ns =108 clks x 6 or 648

  - 1/2X latency x 3X clock rate x 3X Instr/clock ⇒ ≈5X
- Power limits performance (battery, cooling)
- Shrinking number of desktop MPUs?

PA-RISC PowerPC MIPS Alpha **SPARC** 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 Myhrvold M/S: 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)

DRAM Generation

Minimum Memory Size	'86 1 Mb	'89 4 Mb	'92 16 Mb	'96 64 Mb	'99 256 Mb	'02 1 Gb
4 MB	32 →	8				
8 MB		16 →	4			
16 MB			8 →	2		
32 MB				4 →	1	
64 MB				8 →	2	
128 MB					4 →	1
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, ...

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”?
- AlphaSta. 600: 180 ns=128b, 270 ns= 512b Next generation (21264): 180 ns for 512b?

Potential IRAM Bandwidth: 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

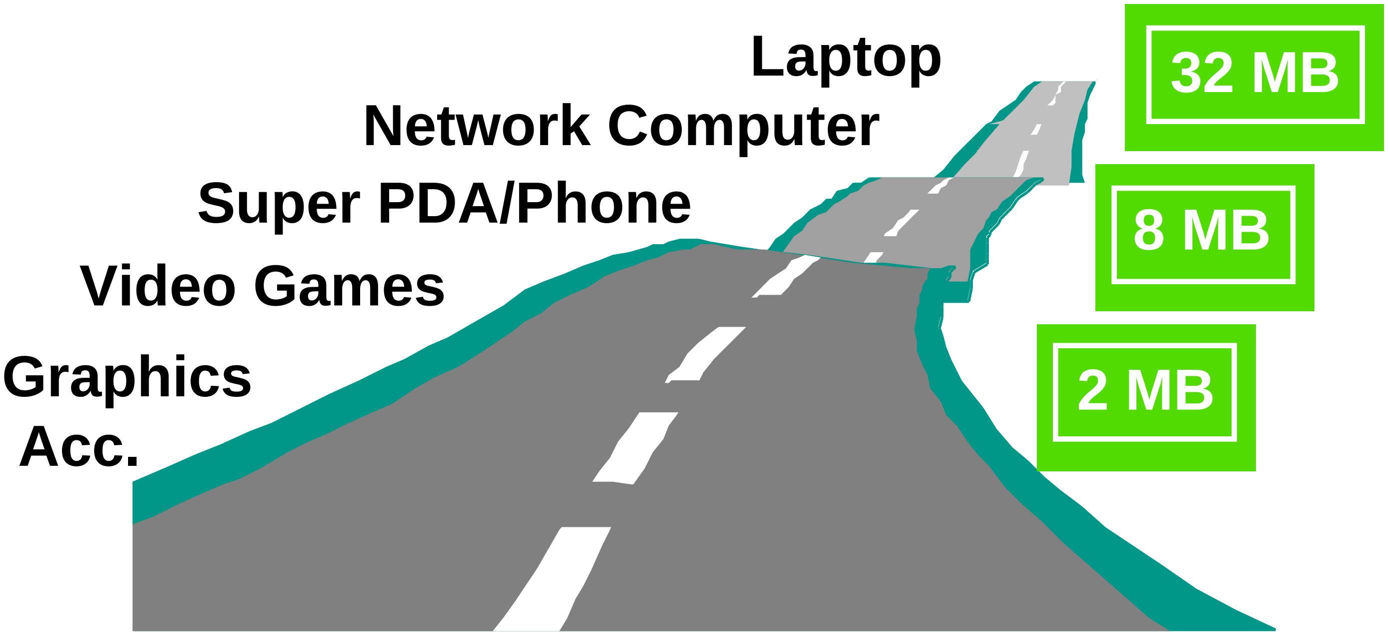
Potential Energy Efficiency: 2X-4X

- Case study of StrongARM memory hierarchy vs. IRAM memory hierarchy
  - 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
- Memory cell area ratio/process: P6, α ‘164,SArm
  cache/logic : SRAM/SRAM : DRAM/DRAM
  20-50 : 8-11 : 1

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?
  - “Map out” bad memory modules to improve yield?
  - Improve yield with variable refresh rate?
  - Reduce test cases/testing time during manufacturing?
- IRAM advantages even greater if innovate inside DRAM memory interface?

Commercial IRAM highway is governed by memory per IRAM?



Laptop

Network Computer

Super PDA/Phone

Video Games

Graphics Acc.

32 MB

8 MB

2 MB

Near-term IRAM Applications

- “Intelligent” Set-top
  - 2.6M Nintendo 64 (≈ $150) sold in 1st year
  - 4-chip Nintendo ⇒ 1-chip: 3D graphics, sound, fun!
- “Intelligent” Personal Digital Assistant
  - 0.6M PalmPilots (≈ $300) sold in 1st 6 months
  - Handwriting + learn new alphabet (α = K, ⏋ = T, ∟ = 4) v. Speech input

App #1: PDA of 2003?

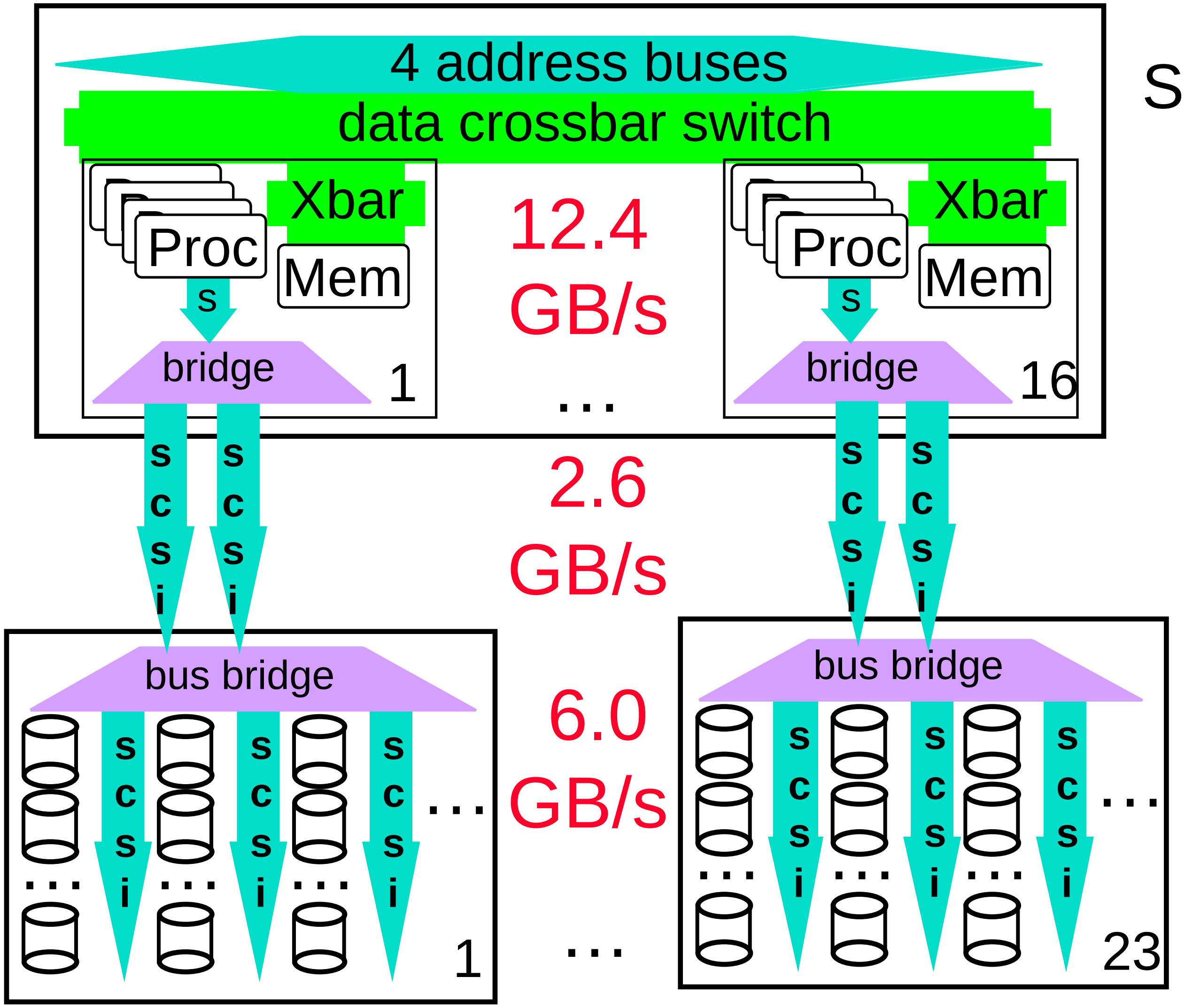
- Pilot PDA (calendar, notes, address book, calculator, memo, ...)
- + Gameboy
- + Nikon Coolpix (camera, tape recorder, notes ...)
- + Cell Phone,Pager, GPS
- + Speech, vision recognition
- + wireless data (WWW)

– Vision to see surroundings, scan documents

– Voice output for conversations

– Play chess with PDA on plane?

Revolutionary App: Decision Support?

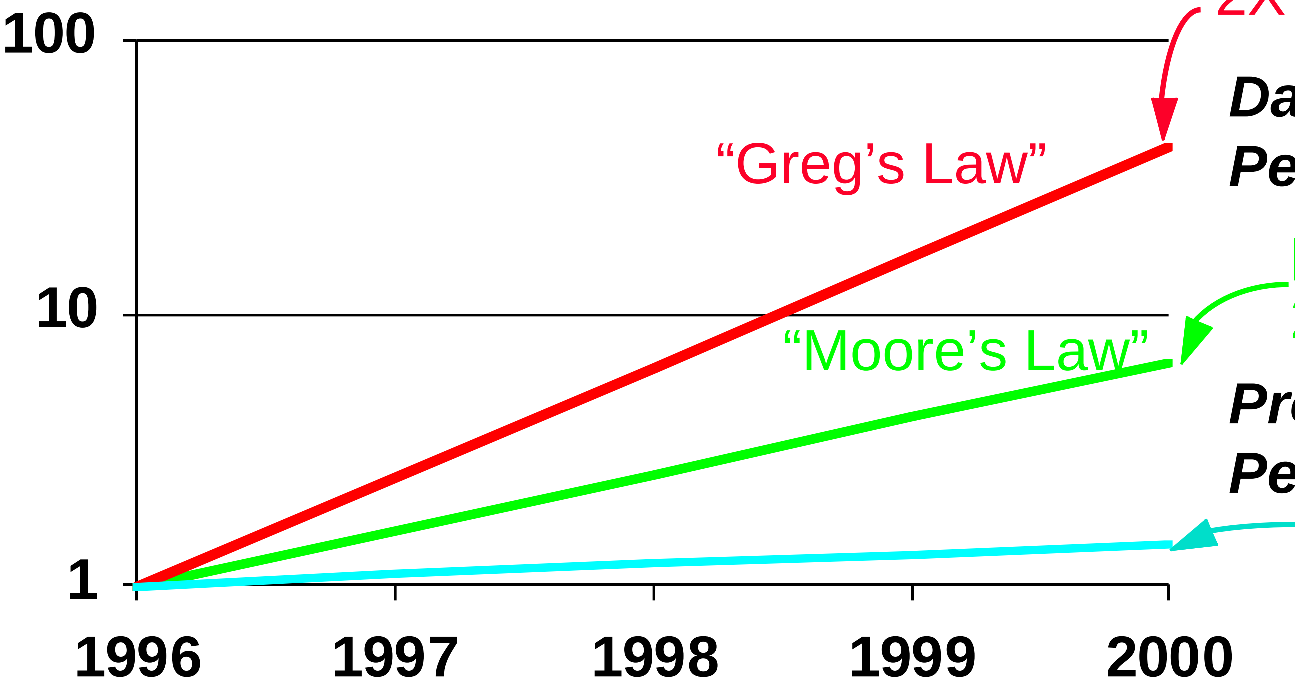


Sun 10000 (Oracle 8):

- 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

IRAM Application Inspiration: Database Demand vs. Processor/DRAM speed



Database demand:
2X / 9 months

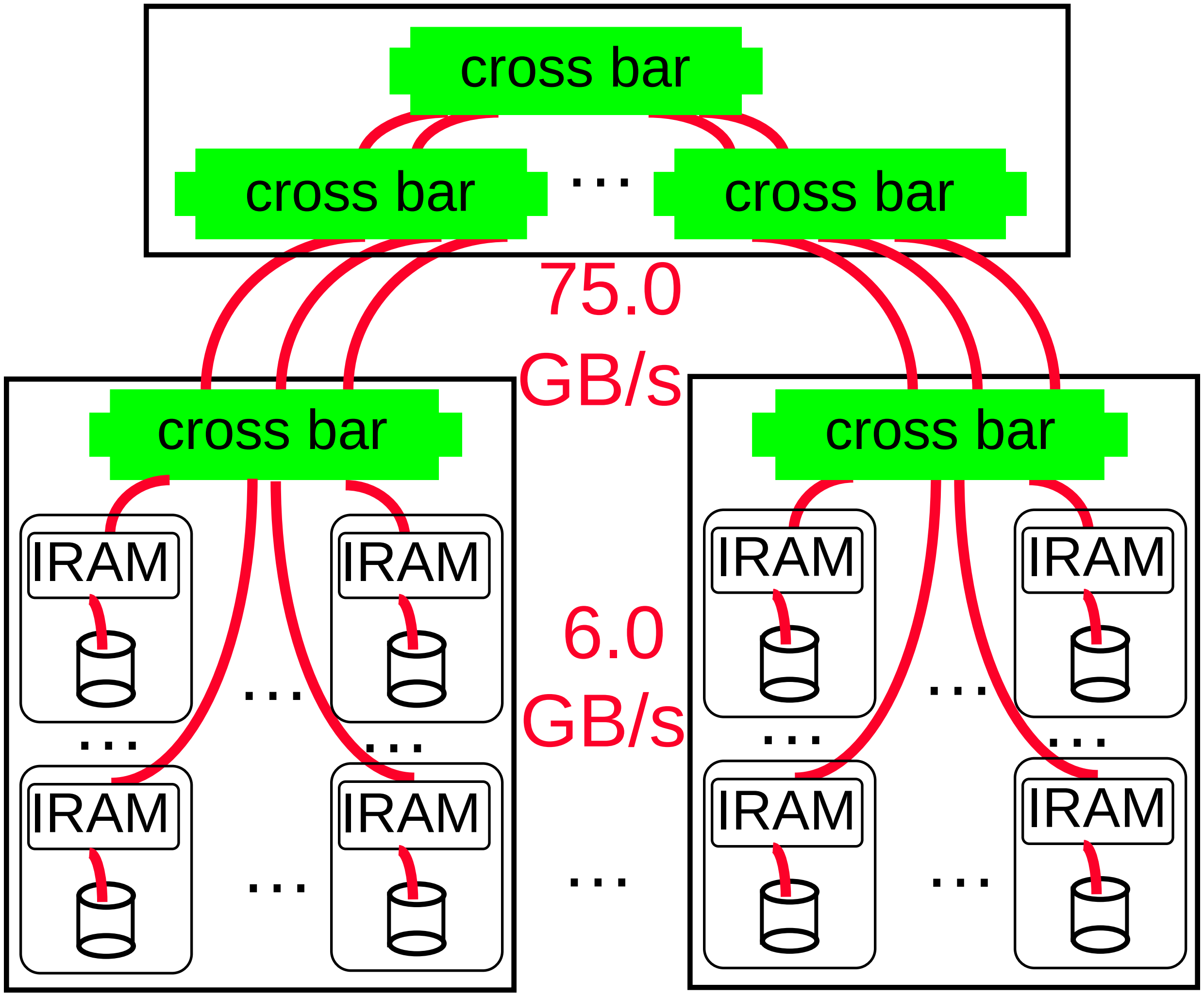
Database-Proc. Performance Gap:

µProc speed
2X / 18 months

Processor-Memory Performance Gap:

DRAM speed
2X /120 months

App #2: “Intelligent Disk”(IDISK): Scaleable Decision Support?



- 1 IRAM/disk + xbar + fast serial link v. conventional SMP
- Network latency = f(SW overhead), not link distance
- Move function to data v. data to CPU (scan, sort, join,...)
- Cheaper, faster, more scalable (≈1/3 $, 3X perf)

“Vanilla” Approach to IRAM

- Estimate performance IRAM version of Alpha (same caches, benchmarks, standard DRAM)
  - Used optimistic and pessimistic factors for logic (1.3-2.0 slower), SRAM (1.1-1.3 slower), DRAM speed (5X-10X faster) for standard DRAM
  - SPEC92 benchmark ⇒ 1.2 to 1.8 times slower
  - Database ⇒ 1.1 times slower to 1.1 times faster
  - Sparse matrix ⇒ 1.2 to 1.8 times faster
- Conventional architecture/benchmarks/DRAM not exciting performance; energy,board area only

"Vanilla" IRAM - Performance Conclusions

- IRAM systems with existing architectures provide moderate performance benefits
- High bandwidth / low latency used to speed up memory accesses, not computation
- Reason: existing architectures developed under assumption of 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

A More Revolutionary Approach: DRAM

- Faster logic in DRAM process
  - DRAM vendors offer faster transistors + same number metal layers as good logic process? @ ≈ 20% higher cost per wafer?
  - As die cost ≈ f(die area^4), 4% die shrink ⇒ equal cost

A More Revolutionary Approach: New Architecture Directions

- "...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)

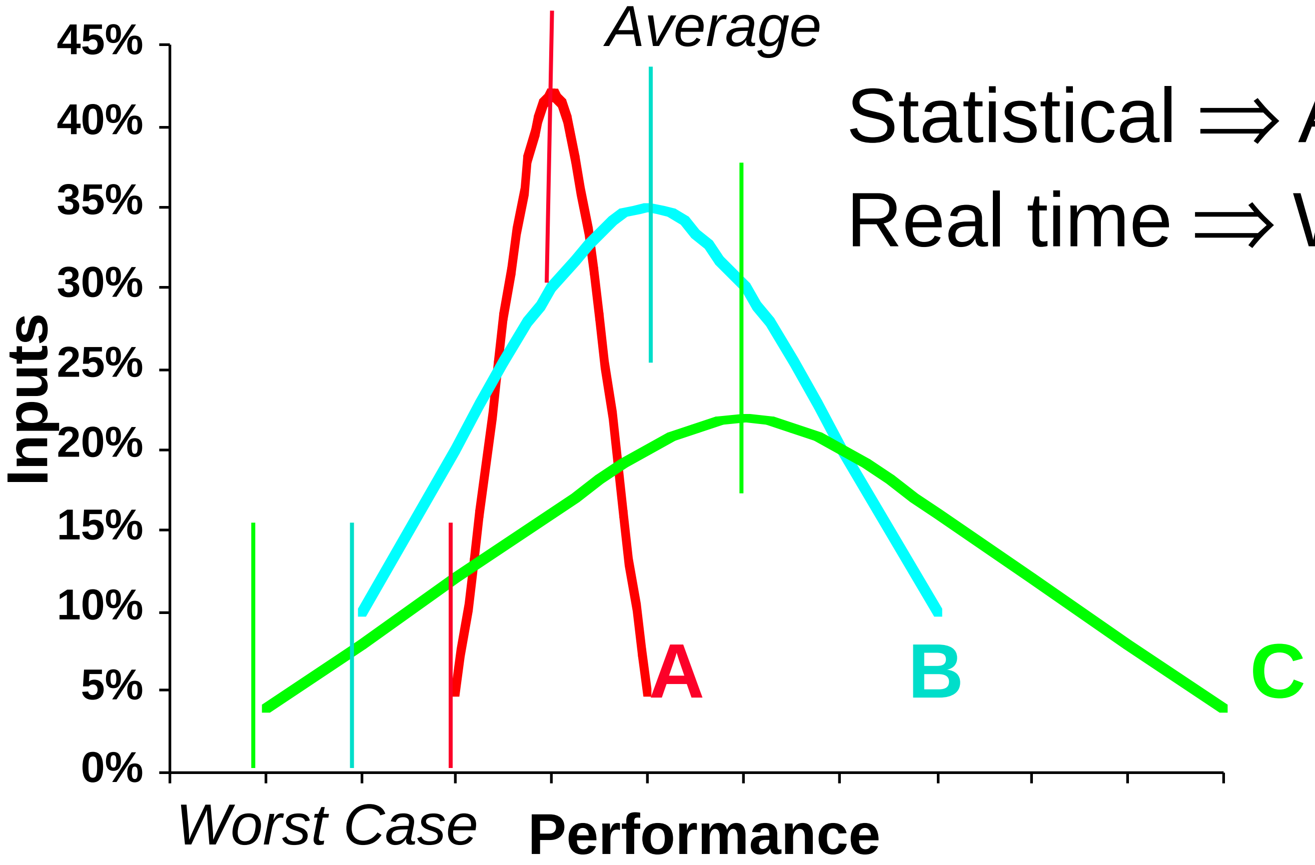
New Architecture Directions

- "…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 Fl. 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)

Grading Architecture Options

	OOO/SS++	IA-64	microSMP
Technology scaling	C	C+	A
Fine grain parallelism	A	A	A
Coarse grain (n chips)	A	A	B
Compiler maturity	B	C	B
MIPS/transistor (cost)	C	B–	B
Programmer model	D	B	B
Energy efficiency	D	C	A
Real time performance	C	B–	B
Grade Point Average	C+	B–	B+

Which is Faster?
Statistical v. Real time Performance



Statistical ⇒ Average ⇒ C

Real time ⇒ Worst ⇒ A

Potential IRAM Architecture

- “New” model: VSIW=Very Short Instruction Word!
  - Compact: Describe N operations with 1 short instruct.
  - Predictable (real-time) perf. vs. statistical perf. (cache)
  - Multimedia ready: choose N*64b, 2N*32b, 4N*16b
  - Easy to get high performance; N operations:
    - » are independent
    - » use same functional unit
    - » access disjoint registers
    - » access registers in same order as previous instructions
    - » access contiguous memory words or known pattern
    - » hides memory latency (and any other latency)
  - Compiler technology already developed, for sale!

Operation & Instruction Count: RISC v. "VSIW" Processor

(from F. Quintana, U. Barcelona.)

Spec92fp	Operations (M)			Instructions (M)		
Program	RISC	VSIW	R / V	RISC	VSIW	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

VSIW reduces ops by 1.2X, instructions by 20X!

Revive Vector (= VSIW) Architecture!

■ Cost: ≈ $1M each?	■ Single-chip CMOS MPU/IRAM
■ Low latency, high BW memory system?	■ IRAM = low latency, high bandwidth memory
■ Code density?	■ Much smaller than VLIW/EPIC
■ Compilers?	■ For sale, mature (>20 years)
■ Vector Performance?	■ Easy scale speed with technology
■ Power/Energy?	■ Parallel to save energy, keep perf
■ Scalar performance?	■ Include modern, modest CPU ⇒ OK scalar (MIPS 5K v. 10k)
■ Real-time?	■ No caches, no speculation ⇒ repeatable speed as vary input
■ Limited to scientific applications?	■ Multimedia apps vectorizable too: N*64b, 2N*32b, 4N*16b

Mediaprocesing Functions (Dubey)

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

(from http://www.research.ibm.com/people/p/pradeep/tutor.html)

Vector Surprise

- 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

Software Technology Trends Affecting V-IRAM?

- V-IRAM: any CPU + vector coprocessor/memory
  - scalar/vector interactions are limited, simple
  - Example V-IRAM architecture based on ARM 9, MIPS
- Vectorizing compilers built for 25 years
  - can buy one for new machine from The Portland Group
- Microsoft "Win CE"/ Java OS for non-x86 platforms
- Library solutions (e.g., MMX); retarget packages
- Software distribution model is evolving?
  - New Model: Java byte codes over network?
    + Just-In-Time compiler to tailor program to machine?

V-IRAM1 Instruction Set

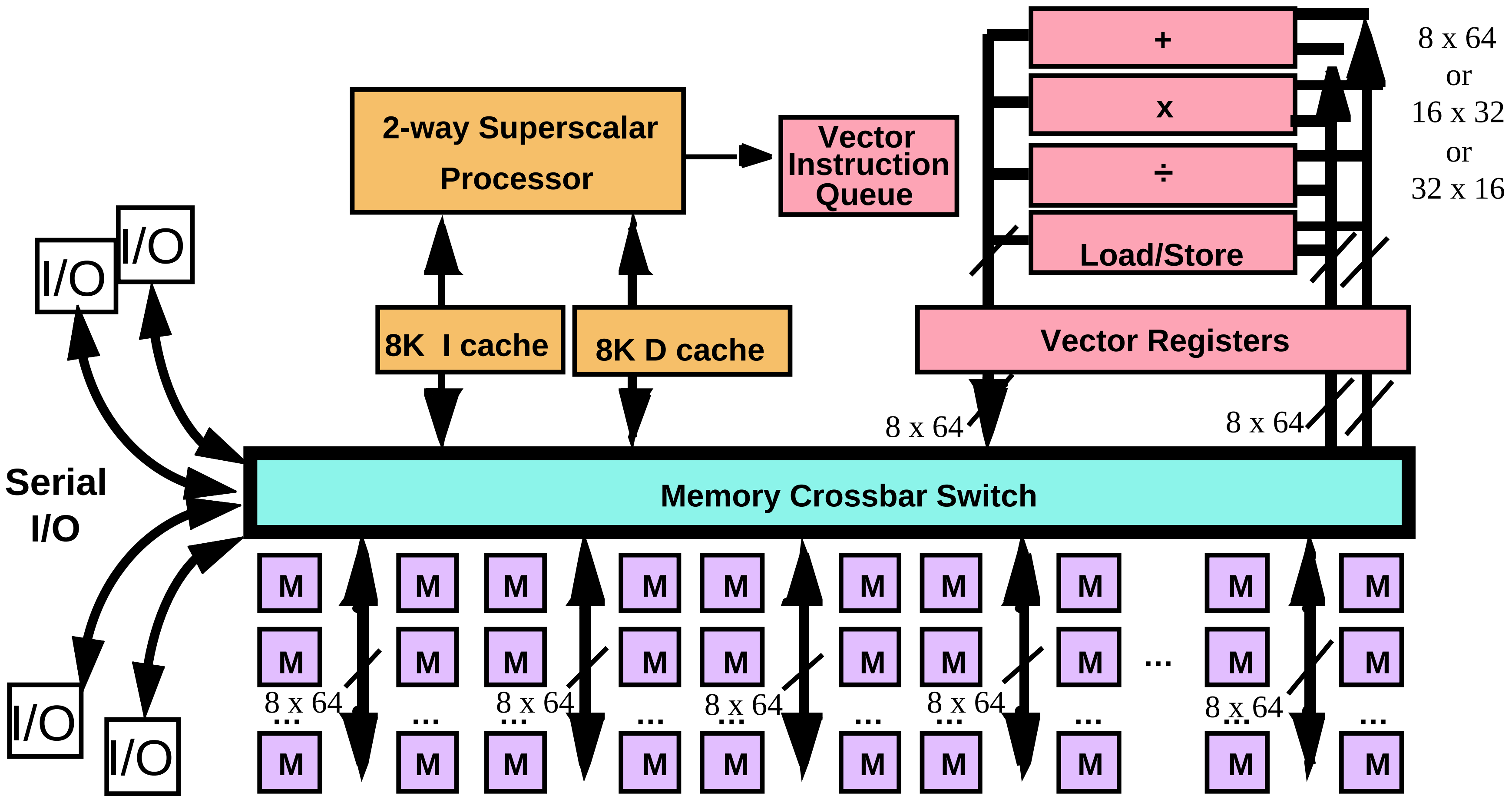
Scalar **Standard scalar instruction set (e.g., ARM, MIPS)**

Vector ALU	**+** **–** **x** **÷** **&** **|** **shl** **shr**	**s.int** **u.int** **s.fp** **d.fp**	**8** **16** **32** **64**	**.vv** **.vs** **.sv**	**saturate** **overflow**	**masked** **unmasked**
Vector Memory	**load** **store**	**s.int** **u.int**	**8** **16** **32** **64**	**8** **16** **32** **64**	**unit** **constant** **indexed**	**masked** **unmasked**

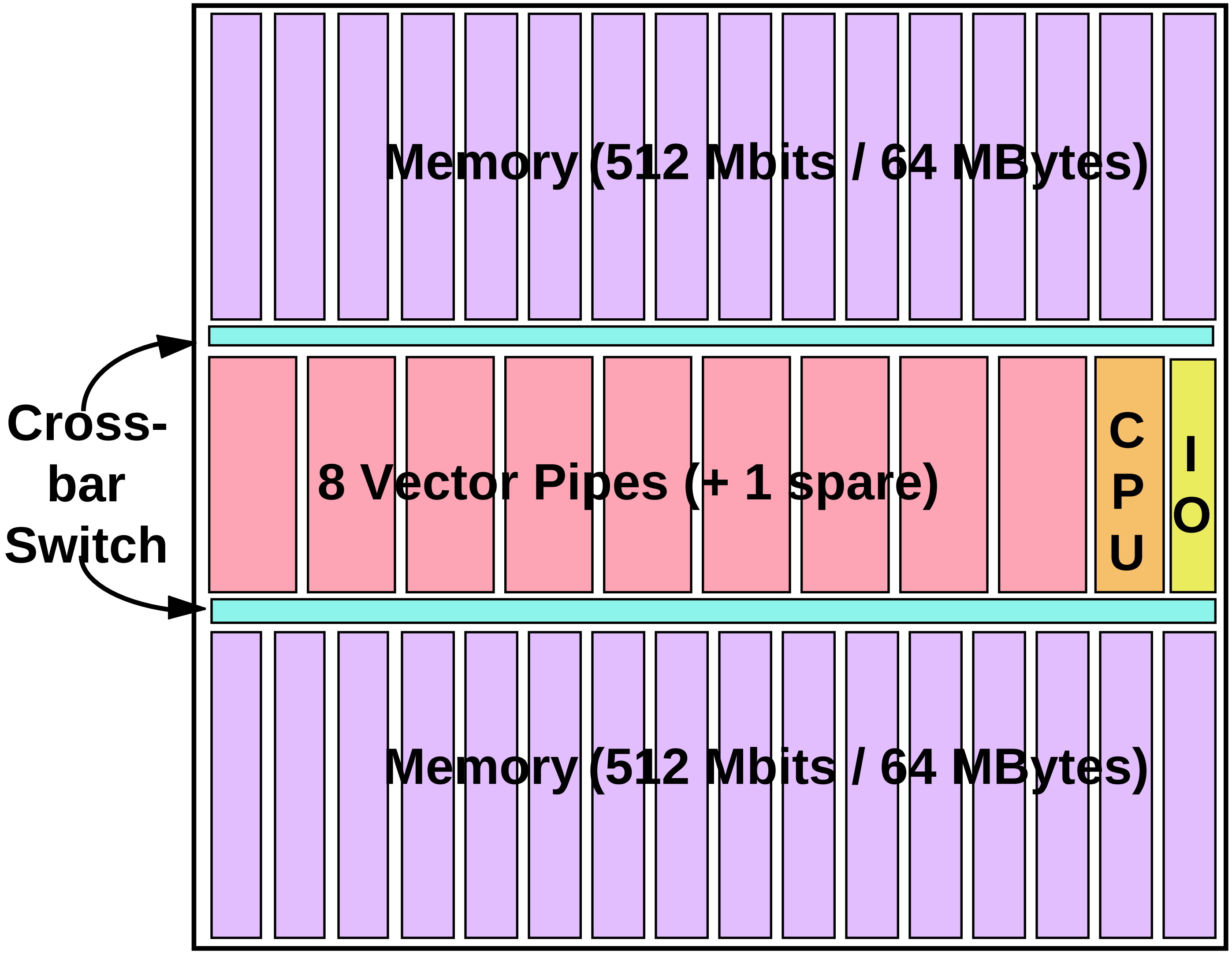
Vector Registers **32 x 32 x 64b (or 32 x 64 x 32b or 32 x 128 x 16b)**
+ 32 x128 x 1b flag

Plus: **flag**, **convert**, **DSP**, and **transfer** operations

V-IRAM-2: 0.13 µm, Fast Logic, 1GHz 16 GFLOPS(64b) / 64 GOPS(16b) / 128MB

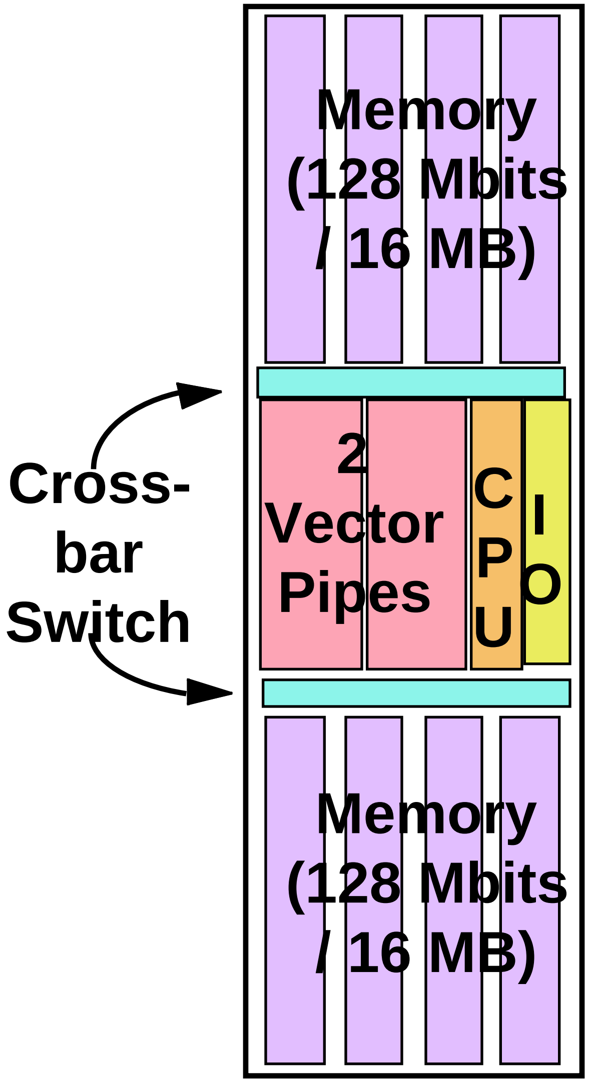


V-IRAM-2 Floorplan


Memory (512 Mbits / 64 MBytes)
Cross-bar Switch
8 Vector Pipes (+ 1 spare)
C P U
I O
Memory (512 Mbits / 64 MBytes)

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

Alternative Goal: Low Cost V-IRAM-2



- Scaleable design, 0.13 generation
- Reduce die size by 4X by shrinking vector units (25%), memory (25%), CPU cache (50%)
- ≈80 mm^2, 32 MB
- High Perf. version: 2.5 w, 1000 MHz, 4 - 16 GOPS
- Low Power version: 0.5 w, 500 MHz, 2 - 8 GOPS

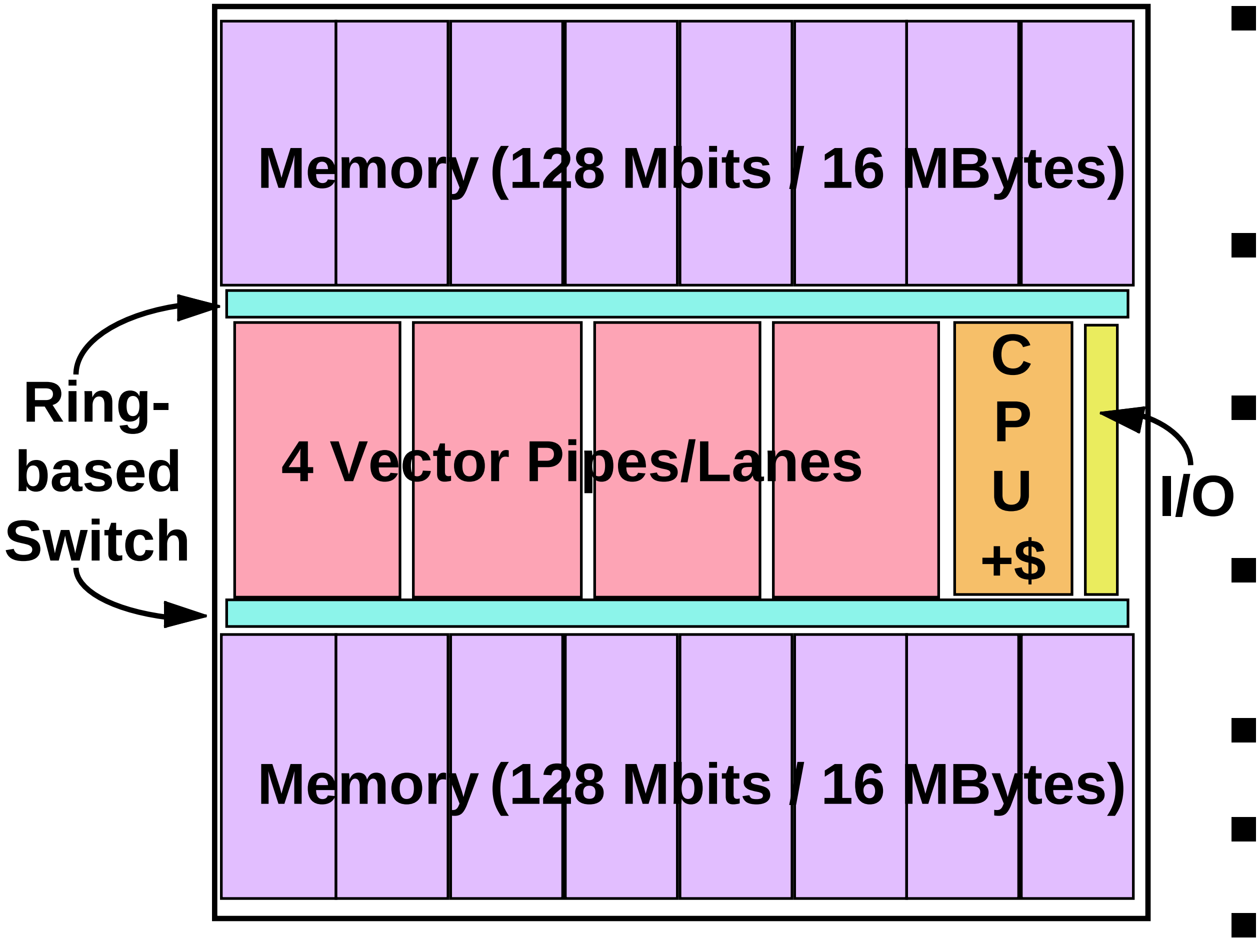
Grading Architecture Options

	OOO/SS++	IA-64	µSMP	VIRAM
Technology scaling	C	C+	A	A
Fine grain parallelism	A	A	A	A
Coarse grain (n chips)	A	A	B	A
Compiler maturity	B	C	B	A
MIPS/transistor (cost)	C	B–	B	A
Programmer model	D	B	B	A
Energy efficiency	D	C	A	A
Real time performance	C	B–	B	A
Grade Point Average	C+	B–	B+	A

VIRAM-1 Specs / Goals

Technology	**0.18-0.20 micron, 5-6 metal layers, fast xtor**	
Memory	**32 MB**	
Die size	≈ **250 mm^2**	
Vector pipes/lanes	**4 64-bit (or 8 32-bit or 16 16-bit or 32 8-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$_{\text{univers.}}$	**200scalar/200vector MHz**	**300sc/300vector MHz**
Perf$_{\text{university}}$	**1.6 GFLOPS$_{64}$-6 GFLOPS$_{16}$**	**2.4 GFLOPS$_{64}$-10 GFLOPS$_{16}$**
Clock$_{\text{industry}}$	**400scalar/400vector MHz**	**600s/600v MHz**
Perf$_{\text{industry}}$	**3.2 GFLOPS$_{64}$-12 GFLOPS$_{16}$**	**4 GFLOPS$_{64}$-16 GFLOPS$_{16}$**

Tentative VIRAM-1 Floorplan


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

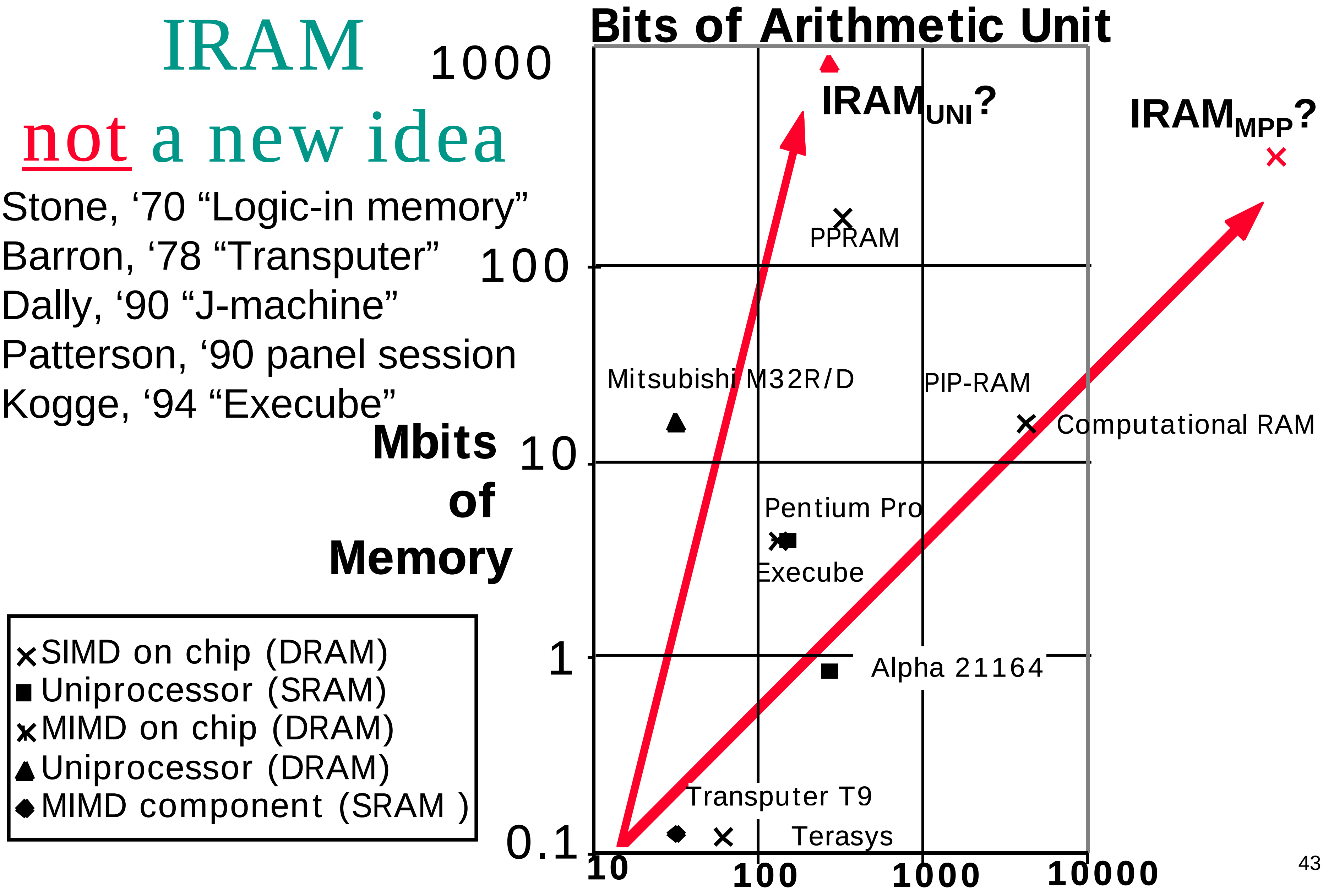
- 0.18 µm DRAM 32 MB in 16 banks x 256b, 128 subbanks
- 0.25 µm, 5 Metal Logic
- ≈ 200 MHz CPU, 4K I$, 4K D$
- ≈ 4 100 MHz FP/int. vector units
- die: ≈ 16x16 mm
- xtors: ≈ 270M
- power: ≈2 Watts

V-IRAM-1 Tentative Plan

- Phase I: Feasibility stage (≈H1'98)
  - Test chip, CAD agreement, architecture defined
- Phase 2: Design & Layout Stage (≈H2'98)
  - Simulated design and layout
- Phase 3: Verification (≈H2'99)
  - Tape-out
- Phase 4: Fabrication,Testing, and Demonstration (≈H1'00)
  - Functional integrated circuit
- First microprocessor ≥ 0.25B transistors!

IRAM
not a new idea

Stone, ‘70 “Logic-in memory”
Barron, ‘78 “Transputer”
Dally, ‘90 “J-machine”
Patterson, ‘90 panel session
Kogge, ‘94 “Execube”



“Architectural Issues for the 1990s” (From Microprocessor Forum 10-10-90):

- Given:
  Superscalar, superpipelined RISCs and
  Amdahl's Law will not be repealed
  => High performance in 1990s is not limited by CPU

- Predictions for 1990s:
  "Either/Or" CPU/Memory will disappear *(“nonblocking cache”)*

  Multipronged attack on memory bottleneck
  cache conscious compilers
  lockup free caches / prefetching

  All programs will become I/O bound; design accordingly

  Most important CPU of 1990s is in DRAM: "IRAM"
  (Intelligent RAM: 64Mb + 0.3M transistor CPU = 100.5%)
  => CPUs are genuinely free with IRAM

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 mem. capacity, smaller board area ⇒ OK market v. desktop (55M 32b RISC '96)

IRAM Challenges

- Chip
  - Good performance and reasonable power?
  - Speed, area, power, yield, cost in DRAM process?
  - Testing time of IRAM vs DRAM vs microprocessor?
  - BW/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 mem/IO BW, energy, board area; challenges in power/performance, testing, yield
- 10X-100X improvements based on technology shipping for 20 years (not JJ, photons, MEMS, ...)
- Apps/metrics of future to design computer of future
- V-IRAM can show IRAM's potential
  - multimedia, energy, size, scaling, code size, compilers
- Revolution in computer implementation v. Instr Set
  - Potential Impact #1: turn server industry inside-out?
- Potential #2: shift semiconductor balance of power?

  Who ships the most memory? Most microprocessors?

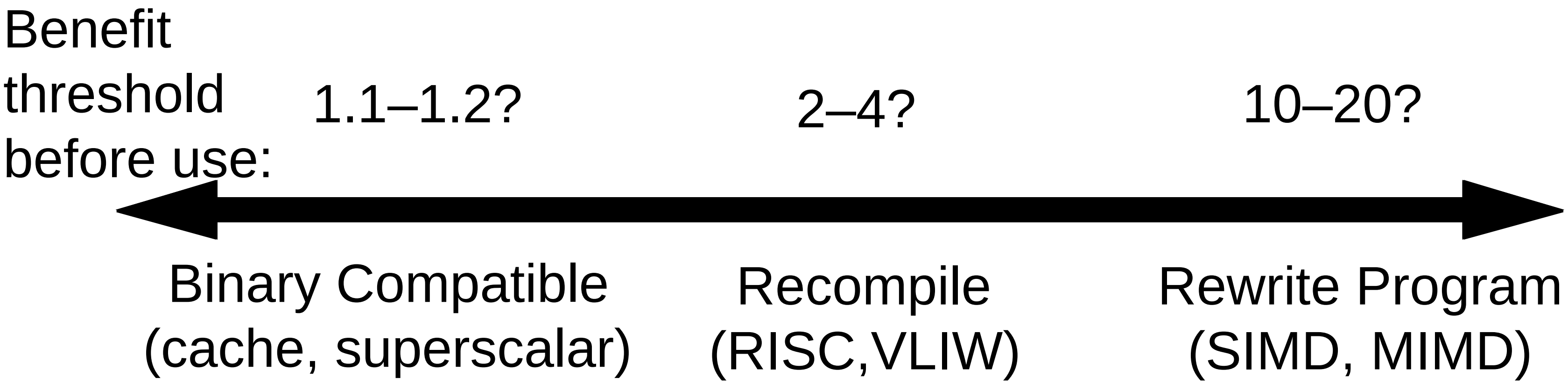
Interested in Participating?

- Looking for ideas of IRAM enabled apps
- Looking for possible MIPS scalar core
- Contact us if you’re interested:
  `http://iram.cs.berkeley.edu/`
  `email: patterson@cs.berkeley.edu`
- Thanks for advice/support: DARPA, California MICRO, ARM, IBM, Intel, LG Semiconductor, Microsoft, Mitsubishi, Neomagic, Samsung, SGI/Cray, Sun Microsystems

Backup Slides

(The following slides are used to help answer questions)

New Architecture Directions



Benefit threshold before use:

1.1–1.2?

2–4?

10–20?

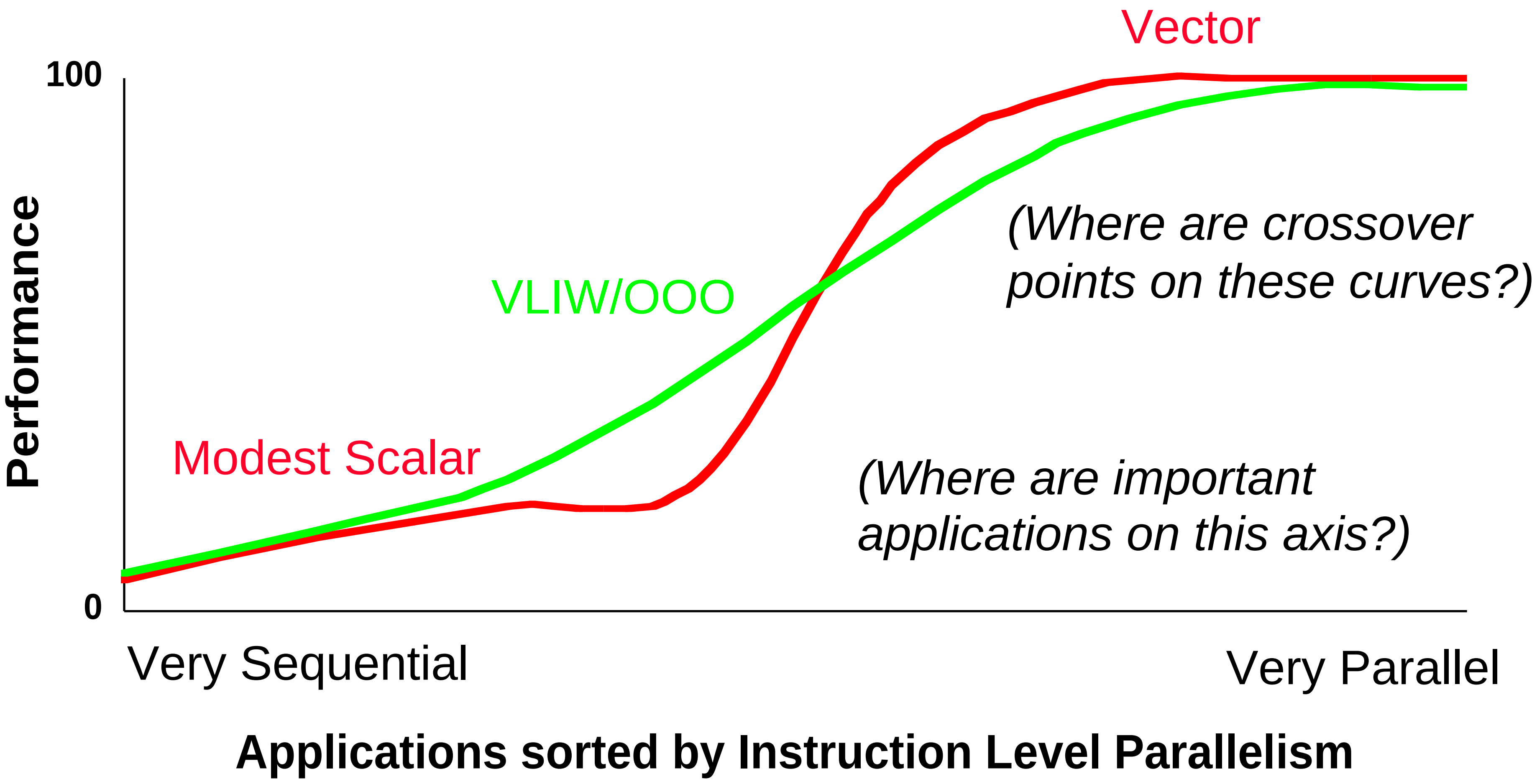
Binary Compatible (cache, superscalar)

Recompile (RISC,VLIW)

Rewrite Program (SIMD, MIMD)

- More innovative than “Let’s build a larger cache!”
- IRAM architecture with simple programming to deliver cost/performance for many applications
  - Evolve software while changing underlying hardware
  - Simple ⇒ sequential (not parallel) program; large memory; uniform memory access time

VLIW / Out-of-Order vs. Modest Scalar+Vector



(Where are crossover points on these curves?)

(Where are important applications on this axis?)

Vector 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

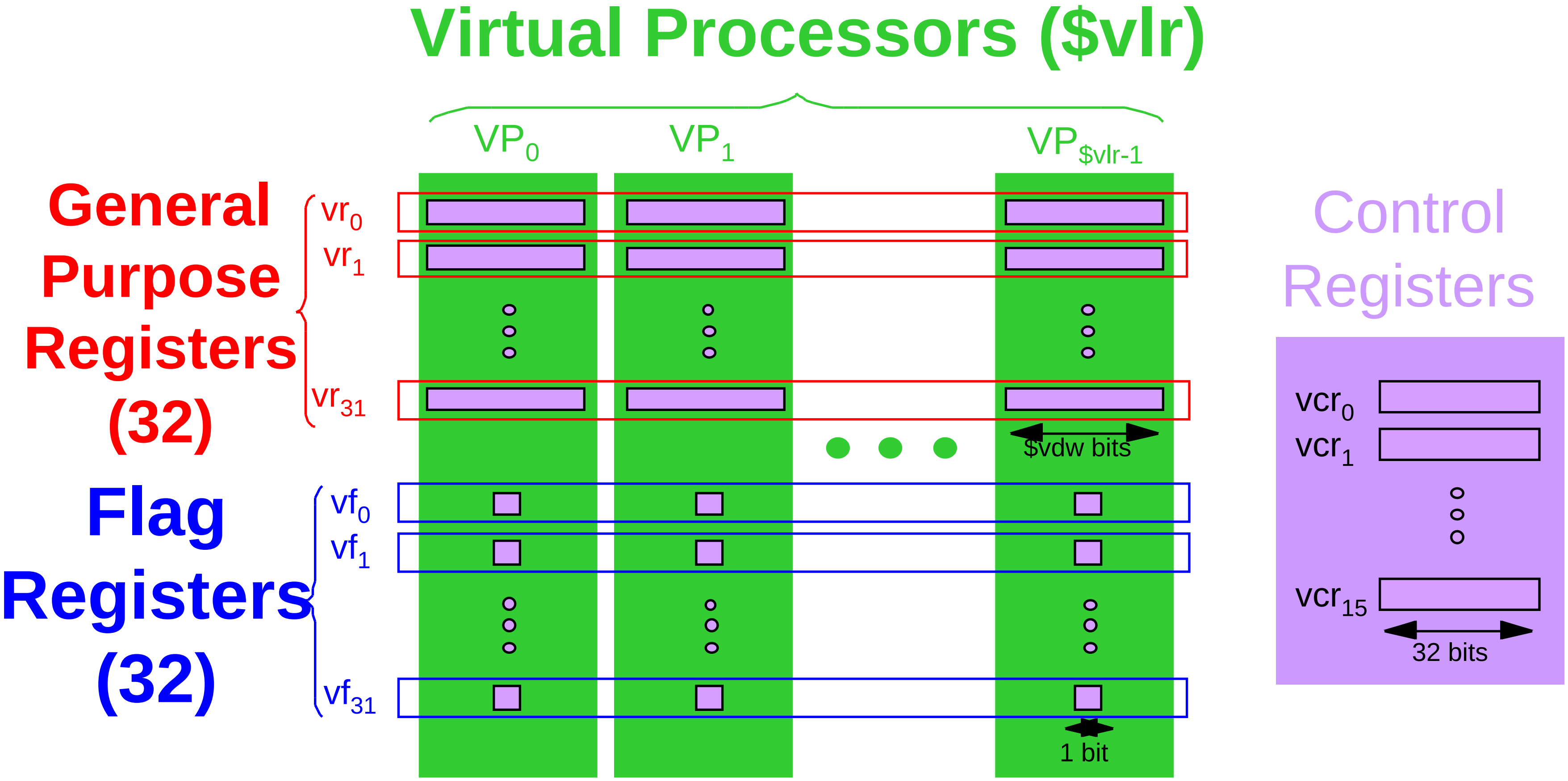
Variable Data Width

- Programmer thinks in terms of vectors of data of some width (8, 16, 32, or 64 bits)
- Good for multimedia
  - More elegant than MMX-style extensions
- Shouldn’t have to worry about how it is stored in memory
  - No need for explicit pack/unpack operations

V-IRAM1 DSP ISA Features

- 16b / 32b / 64b vector DSP ops: +,–,x, shl, shr
  + shift and round 2nd operand (3 rounding modes)
  + saturate result if overflow (optional)

Vector Architectural State



Vectors Are Inexpensive

Scalar

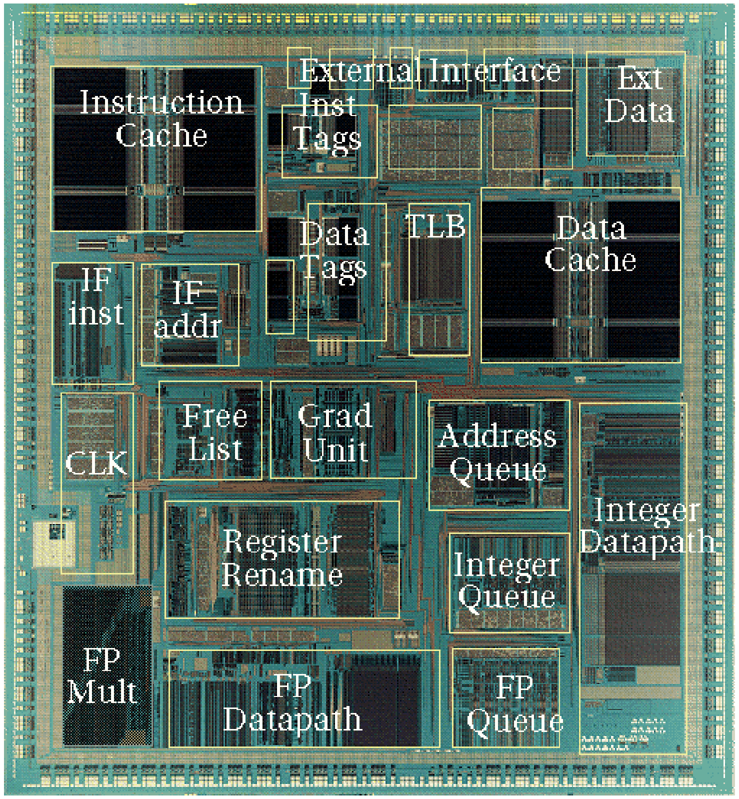
- N ops per cycle
  $\Rightarrow O(N^2)$ circuitry
- HP PA-8000
  - 4-way issue
  - reorder buffer:
    850K transistors
    - incl. 6,720 5-bit register
      number comparators

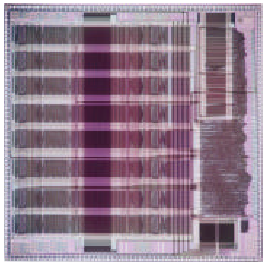
Vector

- 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
  - No floating point

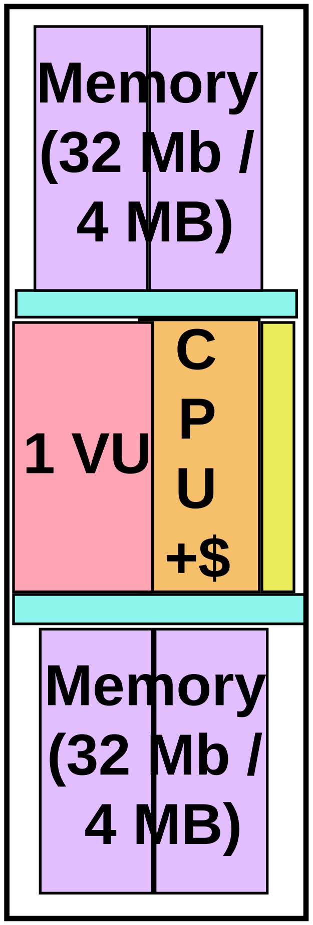
***See** `http://www.icsi.berkeley.edu/real/spert/t0-intro.html`

MIPS R10000 vs. T0





Tentative VIRAM-”0.25” Floorplan



- Demonstrate scalability via 2nd layout (automatic from 1st)
- 8 MB in 4 banks x 256b, 32 subbanks
- ≈ 200 MHz CPU, 4K I$, 4K D$
- 1 ≈ 200 MHz FP/int. vector units
- die: ≈ 5 x 16 mm
- xtors: ≈ 70M
- power: ≈0.5 Watts

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

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

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

What about I/O?

- Current system architectures have limitations
- I/O bus performance lags other 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

ISIMM / IDISK Example: Sort

- Berkeley NOW cluster has world record sort: 8.6GB 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, Smart SIMM sorts 100 GB

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
- 12/97: 233 MHz, 268 MIPS, 0.36W typ., $49

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 v. DRAM)	2.4	1.6
L1 to Memory (off- v. on-chip)	98.5	4.6
L2 to Memory (off-chip)	316.0	*(n.a.)*

» Based on Digital StrongARM, 0.35 µm technology

» See "The Energy Efficiency of IRAM Architectures," *24th Int'l Symp. on Computer Architecture*, June 1997

Vectors Lower Power

Single-issue Scalar

- One instruction fetch, decode, dispatch per operation
- Arbitrary register accesses, adds area and power
- Loop unrolling and software pipelining for high performance increases instruction cache footprint
- All data passes through cache; waste power if no temporal locality
- One TLB lookup per load or store
- Off-chip access in whole cache lines

Vector

- One instruction fetch,decode, dispatch per vector
- Structured register accesses
- Smaller code for high performance, less power in instruction cache misses
- Bypass cache
- One TLB lookup per group of loads or stores
- 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

Characterizing IRAM Cost/Performance

- Cost ≈ embedded processor + memory
- Small memory on-chip (25 - 100 MB)
- High vector performance (2 -16 GFLOPS)
- High multimedia performance (4 - 64 GOPS)
- Low latency main memory (15 - 30ns)
- High BW main memory (50 - 200 GB/sec)
- High BW I/O (0.5 - 2 GB/sec via N serial lines)
  - Integrated CPU/cache/memory with high memory BW ideal for fast serial I/O

IRAM Cost

- Fallacy: IRAM must cost ≥ Intel chip in PC (≈ $250 to $750)
  - Lower cost package for IRAM:
    - » IRAM: 1 chip with ≈ 30-40 pins, 1-5 watts
    - » Intel Pentium II module (242 pins): 1 chip with ≈ 400 pins, + 512KB cache, graphics/memory controller = 43 watts
  - Cost of whole IRAM applications < $300
  - Mitsubishi M32R with 2MB memory < 2-4X memory
- Smaller footprint, lower power ⇒ IRAM cluster cost ≈ “DRAM cluster” (SIMM)

Example IRAM Architecture Options

- (Massively) Parallel Processors (MPP) in IRAM
  - Hardware: best potential performance / transistor, but less memory per processor
  - Software: few successes in 30 years: databases, file servers, dense matrix computations, ... delivered MPP performance often disappoints
  - Successes are in servers, which need more memory than found in IRAM
  - How get 10X-20X benefit with 4 processors?
  - Will potential speedup justify rewriting programs?

Goal for Vector IRAM Generations

- V-IRAM-1 (≈2000)
- 256 Mbit generation (0.20)
- Die size = 1.5X 256 Mb die
- 1.5 - 2.0 v logic, 2-10 watts
- 100 - 500 MHz
- 4 64-bit pipes/lanes
- 1-4 GFLOPS(64b)/6-16G (16b)
- 30 - 50 GB/sec Mem. BW
- 32 MB capacity + DRAM bus
- Several fast serial I/O

- V-IRAM-2 (≈2003)
- 1 Gbit generation (0.13)
- Die size = 1.5X 1 Gb die
- 1.0 - 1.5 v logic, 2-10 watts
- 200 - 1000 MHz
- 8 64-bit pipes/lanes
- 2-16 GFLOPS/24-64G
- 100 - 200 GB/sec Mem. BW
- 128 MB cap. + DRAM bus
- Many fast serial I/O

DRAM v.
Desktop Microprocessors

Standards	pinout, package, refresh rate, capacity, ...	binary compatibility, IEEE 754, I/O bus
Sources	Multiple	Single
Figures of Merit	1) capacity, 1a) $/bit 2) BW, 3) latency	1) SPEC speed 2) cost
Improve Rate/year	1) 60%, 1a) 25%, 2) 20%, 3) 7%	1) 60%, 2) little change

DRAM Design Goals

- Reduce cell size 2.5, increase die size 1.5
- Sell 10% of a single DRAM generation
  - 6.25 billion DRAMs sold in 1996
- 3 phases: engineering samples, first customer ship(FCS), mass production
  - Fastest to FCS, mass production wins share
- Die size, testing time, yield => profit
  - Yield >> 60%
    (redundant rows/columns to repair flaws)

DRAMs over Time

	DRAM Generation					
1st Gen. Sample	**‘84**	**‘87**	**‘90**	**‘93**	**‘96**	**‘99**
Memory Size	**1 Mb**	**4 Mb**	**16 Mb**	**64 Mb**	**256 Mb**	**1024 Mb**
Die Size (mm^2)	**55**	**85**	**130**	**200**	**300**	**450**
Memory Area (mm^2)	**30**	**47**	**72**	**110**	**165**	**250**
Memory Cell Area (μm^2)	**28.8**	**11.1**	**4.28**	**1.64**	**0.61**	**0.23**

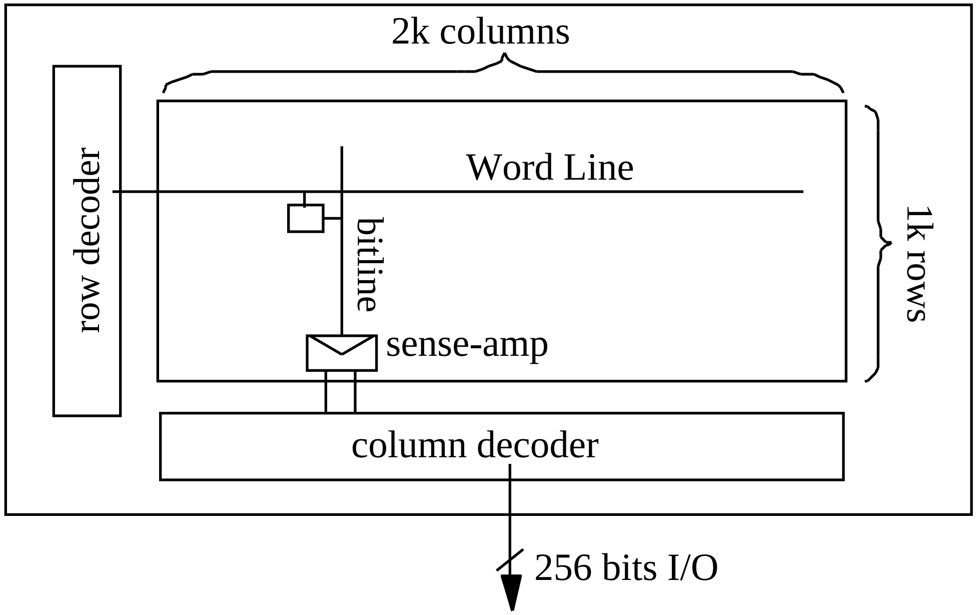
(from Kazuhiro Sakashita, Mitsubishi)

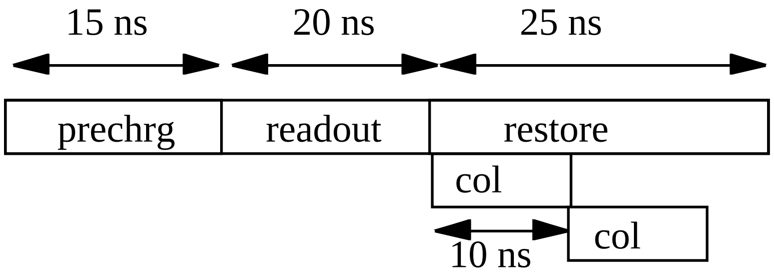
DRAM Access

Steps:

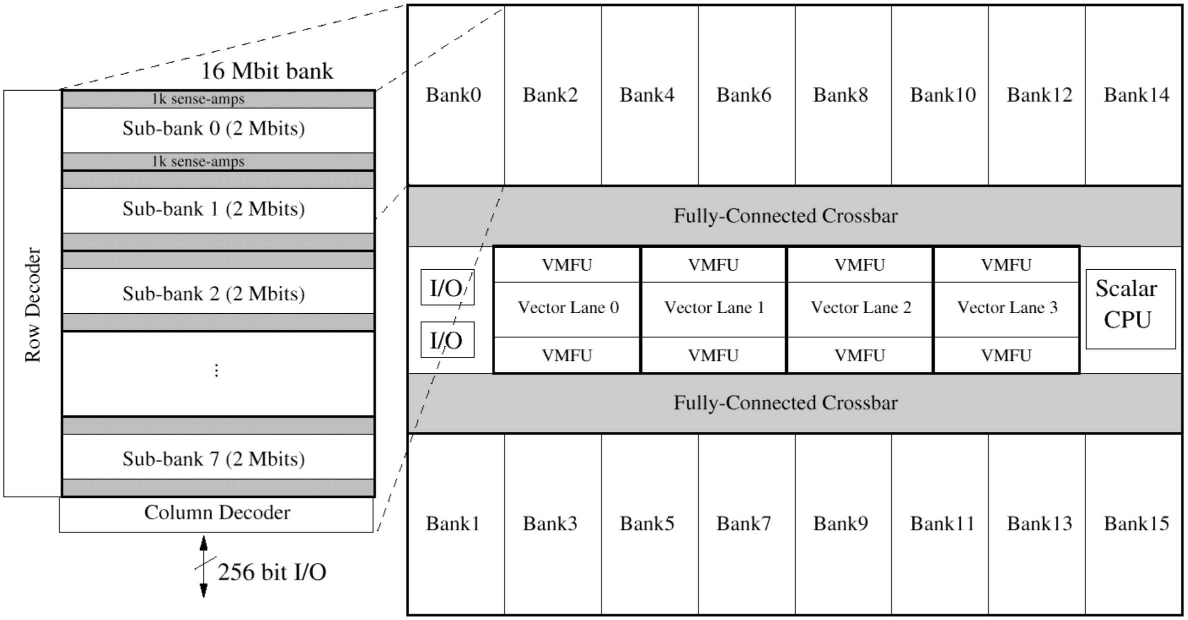
1. Precharge
2. Data-Readout
3. Data-Restore
4. Column Access

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



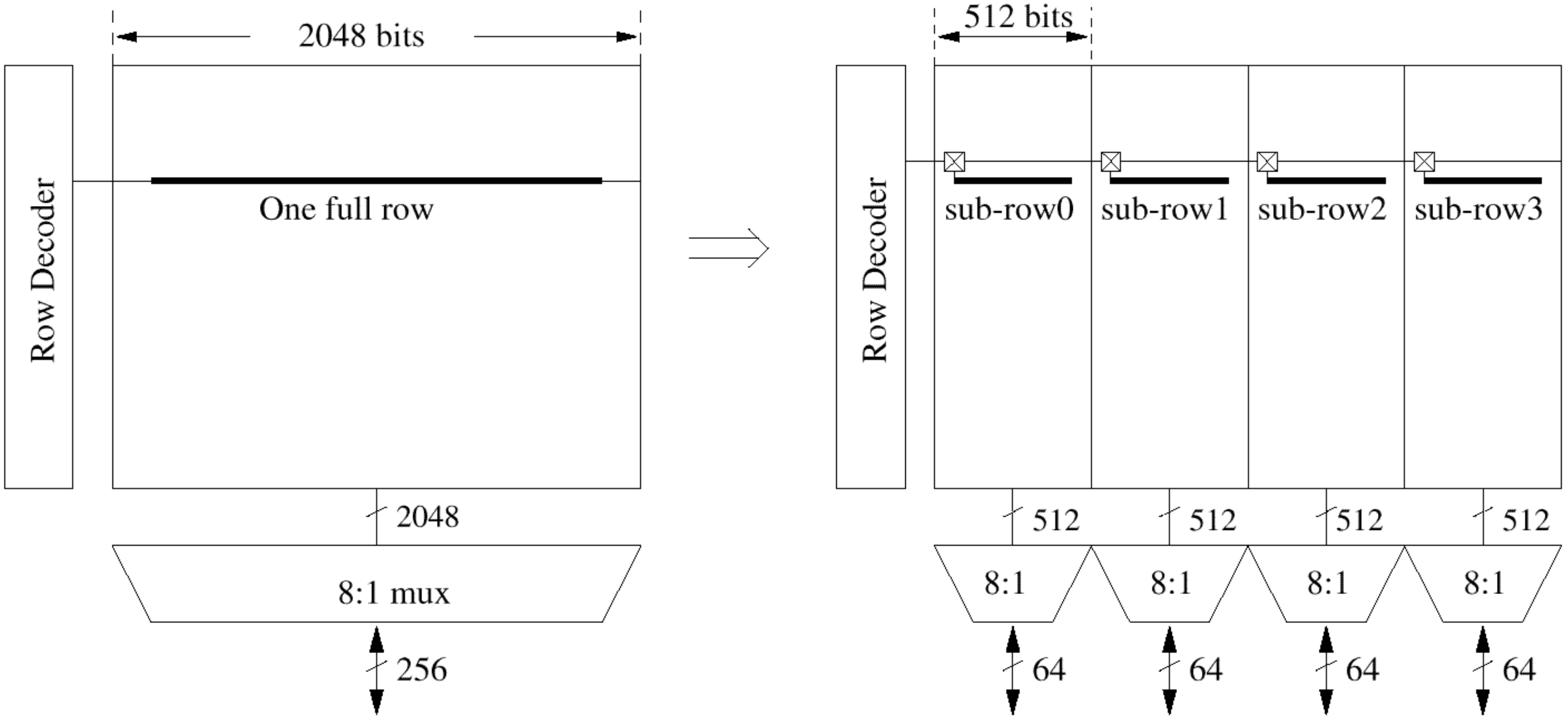


Possible DRAM Innovations #1



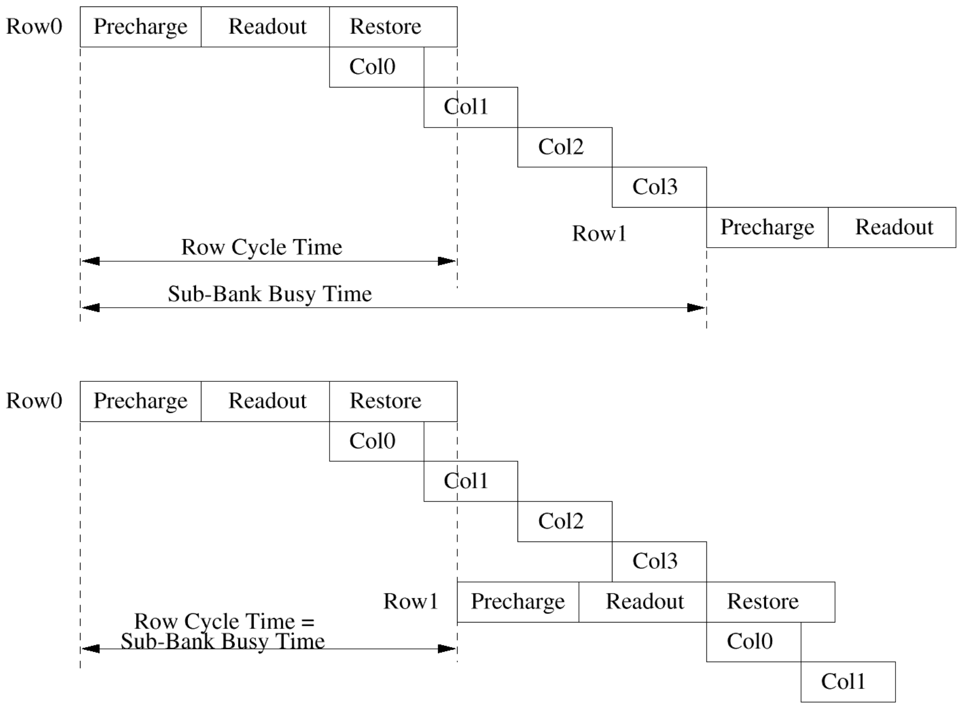
- More banks
  - Each bank can independently process a separate address stream
- Independent Sub-Banks
  - Hides memory latency
  - Increases effective cache size (sense-amps)

Possible DRAM Innovations #2



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

Possible DRAM Innovations #3



Row0 Precharge Readout Restore

Col0

Col1

Col2

Col3

Row1 Precharge Readout

Row Cycle Time

Sub-Bank Busy Time

Row0 Precharge Readout Restore

Col0

Col1

Col2

Col3

Row1 Precharge Readout Restore

Col0

Col1

Row Cycle Time = Sub-Bank Busy Time

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

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

How difficult to build and sell 1B transistor chip?

- Microprocessor only: ≈600 people, new CAD tools, what to build? (≈100% cache?)
- DRAM only: What is proper architecture/ interface? 1 Gbit with 16b RAMBUS interface? 1 Gbit with new package, new 512b interface?
- IRAM: highly regular design, target is not hard, can be done by a dozen Berkeley grad students?

Why a company should try IRAM

- If IRAM doesn’t happen, then someday:
  - $10B fab for 16B Xtor MPU (too many gates per die)??
  - $12B fab for 16 Gbit DRAM (too many bits per die)??
- This is not rocket science. In 1997:
  - 20-50X improvement in memory density;
    ⇒ more memory per die or smaller die
  - 10X -100X improvement in memory performance
  - Regularity simplifies design/CAD/validate: 1B Xtors “easy”
  - Logic same speed
  - < 20% higher cost / wafer (but redundancy improves yield)
- IRAM success requires MPU expertise + DRAM fab

Words to Remember

“...a strategic inflection point is a time in the life of a business when its fundamentals are about to change. ... Let's not mince words: A strategic inflection point can be deadly when unattended to. Companies that begin a decline as a result of its changes rarely recover their previous greatness.”

– *Only the Paranoid Survive*, Andrew S. Grove, 1996

Justification#2: Berkeley has done one "lap"; ready for new architecture?

- RISC: Instruction set /Processor design + Compilers (1980-84)
- SOAR/SPUR: Obj. Oriented SW, Caches, & Shared Memory Multiprocessors + OS kernel (1983-89)
- RAID: Disk I/O + File systems (1988-93)
- NOW: Networks + Clusters + Protocols (1993-98)
- IRAM: Instruction set, Processor design, Memory Hierarchy, I/O, Network, and Compilers/OS (1996-200?)