New Directions in Computer Architecture

David A. Patterson

http://iram.cs.berkeley.edu/papers/direction/paper.html

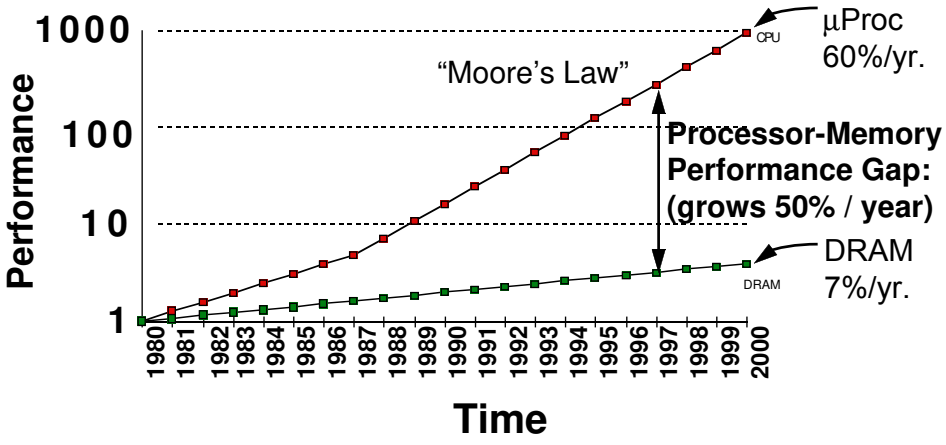
http://cs.berkeley.edu/~patterson/talks

patterson@cs.berkeley.edu
EECS, University of California
Berkeley, CA 94720-1776

Outline

- Desktop/Server Microprocessor State of the Art
- Mobile Multimedia Computing as New Direction
- A New Architecture for Mobile Multimedia Computing
- A New Technology for Mobile Multimedia Computing
- Berkeley's Mobile Multimedia Microprocessor
- Radical Bonus Application
- Challenges & Potential 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

- 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
- Benchmarks: SPEC, TPC-C, TPC-D
  - Benchmark highest optimization, ship lowest optimization?
  - Applications of past to design computers of future?

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

Challenge for Future Microprocessors

- "...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)

Billion Transitor Architectures and "Stationary Computer" Metrics

	SS++	Trace	SMT	CMP	IA-64	RAW
SPEC Int	+	+	+	=	+	=
SPEC FP	+	+	+	+	+	=
TPC (DataBse)	=	=	+	+	=	–
SW Effort	+	+	=	=	=	–
Design Scal.	–	=	–	=	=	=
Physical Design Complexity	–	=	–	=	=	+

(See *IEEE Computer* (9/97), Special Issue on Billion Transistor Microprocessors)

Desktop/Server State of the Art

- Primary focus of architecture research last 15 years
- Processor performance doubling / 18 months
  - assuming SPEC compiler optimization levels
- Growing MPU-DRAM performance gap & tax
- Cost fixed at ≈$500/chip, power whatever can cool
  - 10X cost, 10X power => 2X integer performance?
- Desktop apps slow at rate processors speedup?
- Consolidation of stationary computer industry?

PA-RISC MIPS PowerPC Alpha **SPARC** IA-64

Outline

- Desktop/Server Microprocessor State of the Art
- Mobile Multimedia Computing as New Direction
- A New Architecture for Mobile Multimedia Computing
- A New Technology for Mobile Multimedia Computing
- Berkeley's Mobile Multimedia Microprocessor
- Radical Bonus Application
- Challenges & Potential Industrial Impact

Intelligent PDA (2003?)

Pilot PDA (todo,calendar, calculator, addresses,...)

+ Gameboy (Tetris, ...)

+ Nikon Coolpix (camera)

+ Cell Phone, Pager, GPS, tape recorder, TV remote, am/fm radio, garage door opener, ...

+ Wireless data (WWW)

+ Speech, vision recog.

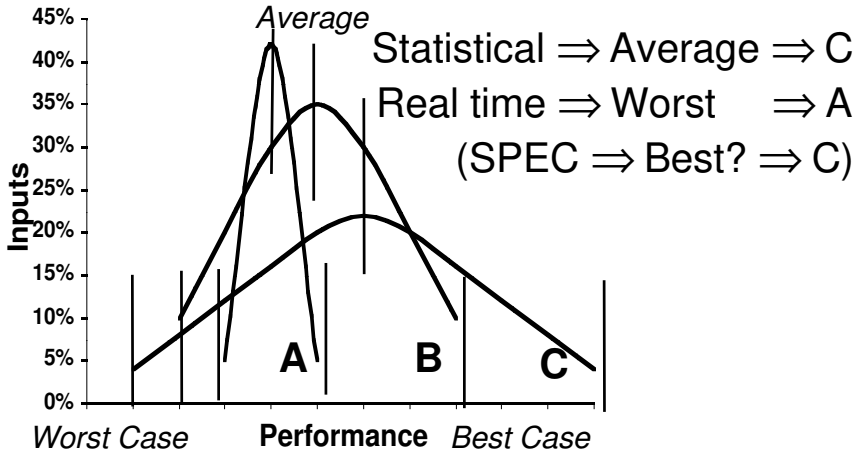
+ Voice output for conversations

- Speech control of all devices
- Vision to see surroundings, scan documents, read bar code, measure room, ...

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 real-time response, continuous media data types (no temporal locality), fine grain parallelism, coarse grain parallelism, memory BW
  - "How Multimedia Workloads Will Change Processor Design", Diefendorff & Dubey, *IEEE Computer* (9/97)

Which is Faster? Statistical v. Real time v. SPEC



Billion Transitor Architectures and "Mobile Multimedia" Metrics

	SS++	Trace	SMT	CMP	IA-64	RAW
Design Scal.	–	=	–	=	=	=
Energy/power	–	–	–	=	=	–
Code Size	=	=	=	=	–	=
Real-time	–	–	=	=	=	=
Cont. Data	=	=	=	=	=	=
Memory BW	=	=	=	=	=	=
Fine-grain Par.	=	=	=	=	=	+
Coarse-gr.Par.	=	=	+	+	=	+

Outline

- Desktop/Server Microprocessor State of the Art
- Mobile Multimedia Computing as New Direction
- A New Architecture for Mobile Multimedia Computing
- A New Technology for Mobile Multimedia Computing
- Berkeley's Mobile Multimedia Microprocessor
- Radical Bonus Application
- Challenges & Potential Industrial Impact

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

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

Vector Multiply with dependency

```
/* Multiply a[m][k] * b[k][n] to get
  c[m][n] */
for (i=1; i<m; i++)
{
    for (j=1; j<n; j++)
    {
        sum = 0;
        for (t=1; t<k; t++)
        {
            sum += a[i][t] * b[t][j];
        }
        c[i][j] = sum;
    }
}
```

Novel Matrix Multiply Solution

- You don't need to do reductions for matrix multiply
- You can calculate multiple independent sums within one vector register
- You can vectorize the outer (j) loop to perform 32 dot-products at the same time
- Or you can think of each 32 Virtual Processors doing one of the dot products
  - (Assume Maximum Vector Length is 32)
- Show it in C source code, but can imagine the assembly vector instructions from it

Optimized Vector Example

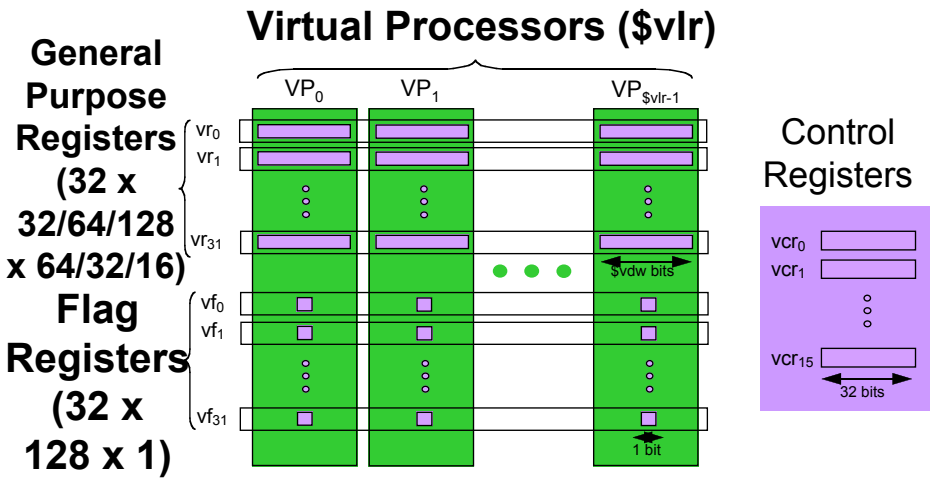
```
/* Multiply a[m][k] * b[k][n] to get c[m][n] */
for (i=1; i<m; i++)
{
 for (j=1; j<n; j+=32)/* Step j 32 at a time. */
    {
        sum[0:31] = 0; /* Initialize a vector
                          register to zeros. */
        for (t=1; t<k; t++)
        {
         a_scalar = a[i][t]; /* Get scalar from
                                a matrix. */
         b_vector[0:31] = b[t][j:j+31];
                          /* Get vector from
                             b matrix. */
         prod[0:31] = b_vector[0:31]*a_scalar;
         /* Do a vector-scalar multiply. */
```


Optimized Vector Example cont'd

```
            /* Vector-vector add into results. */
            sum[0:31] += prod[0:31];
        }

        /* Unit-stride store of vector of
           results. */
        c[i][j:j+31] = sum[0:31];
    }
}
```


Vector Multimedia Architectural State



Vector Multimedia 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

DSP vs. General Purpose MPU

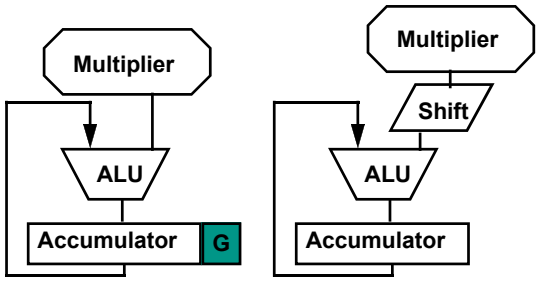
- "MIPS/MFLOPS" = Multiply-Accumulate (MAC) rate
  - DSPs are judged if keep the multipliers busy 100%
- DSPs have 16-bit to 24-bit data words
  - Rounding modes for "fixed point" numbers vs. integers
- The "SPEC" of DSPs is 4 algorithms:
  - Inifinite and Finite Impule Response (IIR, FIR) filters
  - FFT, and Convolvers
- Software is not (yet) king in DSPs.
  - People still write in assembly language
  - In DSPs, algorithms are king!

DSP Data Path: Overflow?

- DSP are descended from analog : what should happen to output when "peg" an input? (e.g., turn up volume control knob on stereo)
  - Modulo Arithmetic???
- Set to most positive ($2^{N-1}-1$) or most negative value(-2^{N-1}) : "saturation"
- Many algorithms were developed in this model

DSP Data Path: Accumulator

- Don't want overflow or have to scale accumulator
- Option 1: accumalator wider than product: "guard bits"
  - Motorola : 24b x 24b => 48b product, 56b Accumulator
- Option 2: shift right and round product before adder



DSP Memory

- DSPs want multiple data ports
  - FIR Tap implies multiple memory accesses
  - Many DSPs have multiple data memories
  - No DSPs have data caches
- Addressing: Autoincrement/Autodecrement
  - Want to keep MAC datapath busy
  - Don't use datapath to calculate fancy address
  - Complex addressing is good

How are DSPs different?

- Narrow data widths, special overflow, rounding
- High Speed Multiply-Accumulate
- Multiple memory ports
- Specialized memory addressing
- Zero overhead loops and repeat instructions
- I/ O support – Serial and parallel ports

Vector support for DSP

- Narrow data?
- High Speed MAC?
- Multiple memory ports?
- Zero overhead loops?
- Autoincrement addressing?
- Width part of vector register: 32 x 64 or 64 x 32 or 128 x 16 (width orthogonal to ISA)
- Can chain vector multiply, add => 1 MAC/clock/lane
- Vectors have banked memory, multiple pipes, Load/Store units
- Vector instructions imply whole loops
- Vector load, store do this: access 32 sequential words

Special Vector Instructions for DSP

- 16b / 32b / 64b vector DSP ops: +,–,x, shl, shr
- Wider accumulate?
  - Option 2 on MAC: Round 2nd operand before add
  - All 3 DSP rounding modes
- Overflow?
  - saturate result if overflow
- Surprising conclusion: vector architecture + narrow data width ops + modest DSP support => excellent DSP support + real compilers!
- Doing DSP benchmarks to compare

Software Technology Trends Affecting New Direction?

- any CPU + vector coprocessor/memory
  - scalar/vector interactions are limited, simple
  - Example 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?

Outline

- Desktop/Server Microprocessor State of the Art
- Mobile Multimedia Computing as New Direction
- A New Architecture for Mobile Multimedia Computing
- A New Technology for Mobile Multimedia Computing
- Berkeley's Mobile Multimedia Microprocessor
- Radical Bonus Application
- Challenges & Potential Industrial Impact

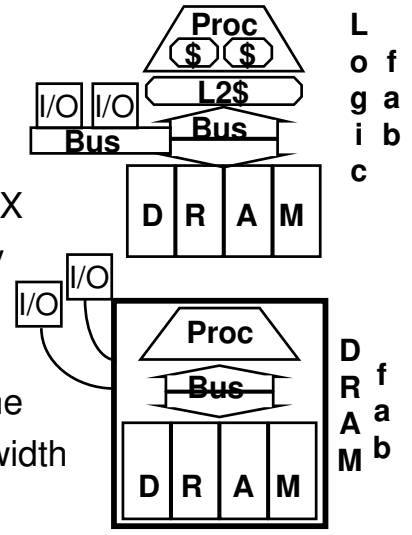
A Better Media for Mobile Multimedia MPUs: Logic+DRAM

- Crash of DRAM market inspires new use of wafers
- 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
- Called Intelligent RAM ("IRAM") since most of transistors will be DRAM

IRAM Vision Statement

Microprocessor & DRAM on a single chip:

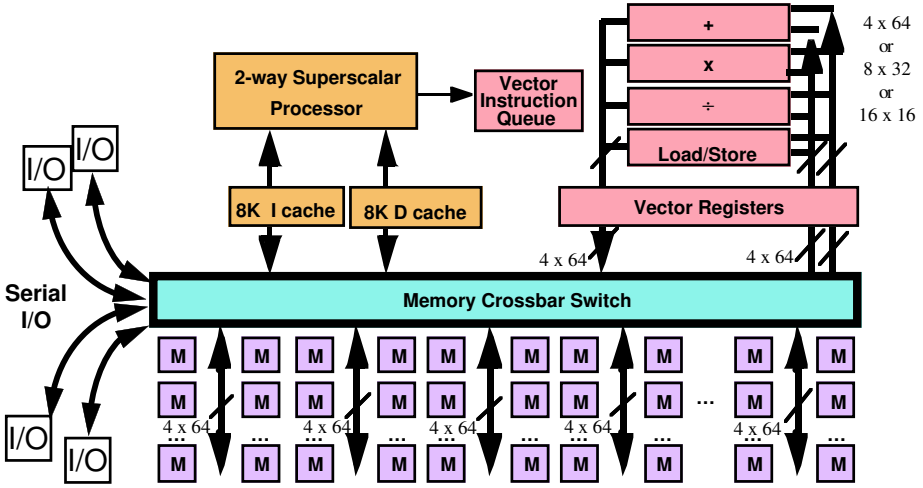
- on-chip memory latency 5-10X, bandwidth 50-100X
- improve energy efficiency 2X-4X (no off-chip bus)
- serial I/O 5-10X v. buses
- smaller board area/volume
- adjustable memory size/width



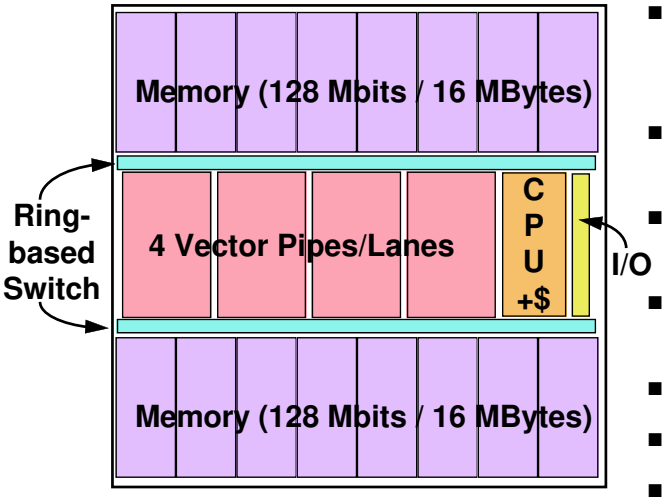
Outline

- Desktop/Server Microprocessor State of the Art
- Mobile Multimedia Computing as New Direction
- A New Architecture for Mobile Multimedia Computing
- A New Technology for Mobile Multimedia Computing
- Berkeley's Mobile Multimedia Microprocessor
- Radical Bonus Application
- Challenges & Potential Industrial Impact

V-IRAM1: 0.18 µm, Fast Logic, 200 MHz
1.6 GFLOPS(64b)/6.4 GOPS(16b)/32MB

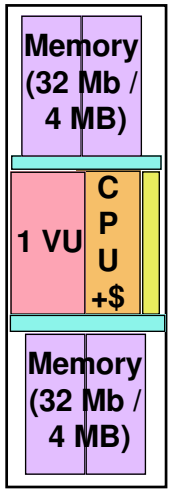


Tentative VIRAM-1 Floorplan



- 0.18 µm DRAM 32 MB in 16 banks x 256b, 128 subbanks
- 0.25 µm, 5 Metal Logic
- ≈ 200 MHz ARM 9, 4K I$, 4K D$
- ≈ 4 100 MHz FP/int. vector units
- die: ≈ 16x16 mm
- xtors: ≈ 270M
- power: ≈2 Watts

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

VIRAM-1 Specs/Goals

Technology	**0.18-0.20 micron, 5-6 metal layers, fast xtor**
Memory	**16-32 MB**
Die size	**≈ 250-300 mm^2**
Vector pipes/lanes	**4 64-bit (or 8 32-bit or 16 16-bit)**
Serial I/O	**4 lines @ 1 Gbit/s**
$Power_{university}$	**≈2 w @ 1-1.5 volt logic**
$Clock_{university}$	**200scalar/200vector MHz**
$Perf_{university}$	**1.6 GFLOPS$_{64}$ – 6 GOPS$_{16}$**
2X	
$Power_{industry}$	**≈1 w @ 1-1.5 volt logic**
$Clock_{industry}$	**400scalar/400vector MHz**
$Perf_{industry}$	**3.2 GFLOPS$_{64}$ – 12 GOPS$_{16}$**

V-IRAM-1 Tentative Plan

- Phase I: Feasibility stage (≈H1'98)
  - Test chip, CAD agreement, architecture defined
- Phase 2: Design & Layout Stage (≈H2'98)
  - Test chip, 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?

Grading VIRAM

Stationary Metrics	VIRAM	Mobile Multimedia Metrics	VIRAM
SPEC Int	–	Energy/power	+
SPEC FP	+	Code Size	+
TPC (DataBse)	–	Real-time response	+
SW Effort	=	Continous Data-types	+
Design Scal.	+	Memory BW	+
Physical Design Complexity	=	Fine-grain Parallelism	+
		Coarse-gr. Parallelism	=

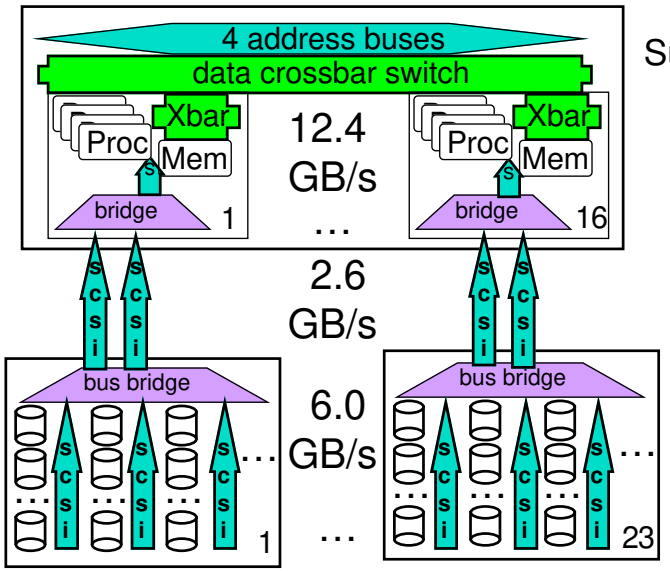
IRAM Challenges

- 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 ...)

Outline

- Desktop/Server Microprocessor State of the Art
- Mobile Multimedia Computing as New Direction
- A New Architecture for Mobile Multimedia Computing
- A New Technology for Mobile Multimedia Computing
- Berkeley's Mobile Multimedia Microprocessor
- Radical Bonus Application
- Challenges & Potential Industrial Impact

Revolutionary App: Decision Support?

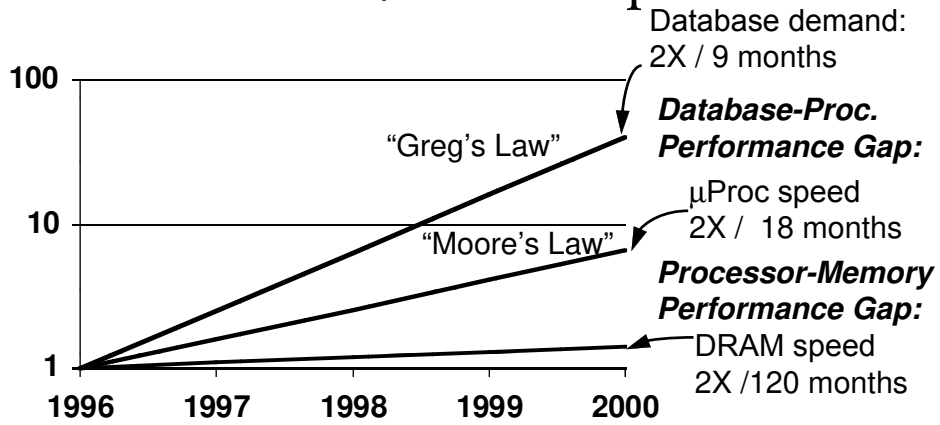


Sun 10000 (Oracle 8):

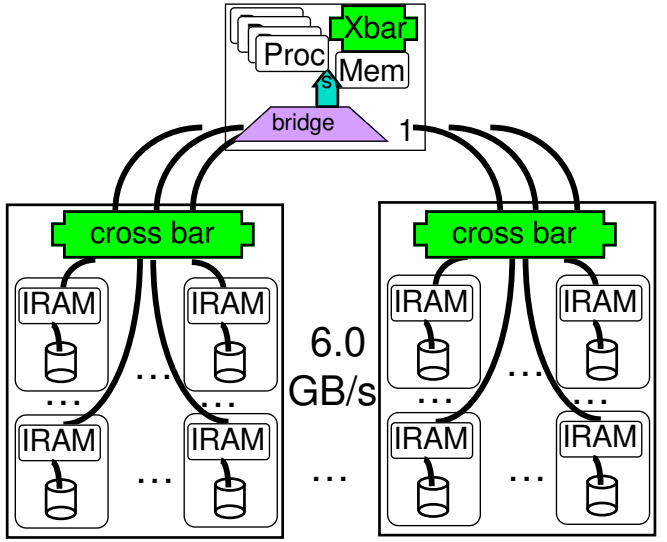
- TPC-D (1TB) leader
- SMP 64 CPUs, 64GB dram, 603 disks

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



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, more scalable (≈1/3 $, 3X perf)

Mobile Multimedia Conclusion

- 1000X performance increase in "stationary" computers, consolidation of industry
  => time for architecture/OS/compiler researchers declare victory, search for new horizons?
- Mobile Multimedia offer many new challenges: energy efficiency, size, real time performance, ...
- VIRAM-1 one example, hope others will follow
- Apps/metrics of future to design computer of future!
  - Suppose PDA replaces desktop as primary computer?
  - Work on FPPP on PC vs. Speech on PDA?

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, ...)
- Suppose IRAM is successful
- 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 VIRAM enabled apps
- Looking for possible MIPS scalar core
- Contact us if you're interested:
  email: patterson@cs.berkeley.edu
  http://iram.cs.berkeley.edu/
  - iram.cs.berkeley.edu/papers/direction/paper.html
- Thanks for advice/support: DARPA, California MICRO, Hitachi, IBM, Intel, LG Semicon, Microsoft, Mitsubishi, Neomagic, Samsung, SGI/Cray, Sun Microsystems, TI

Backup Slides

(The following slides are used to help answer questions)

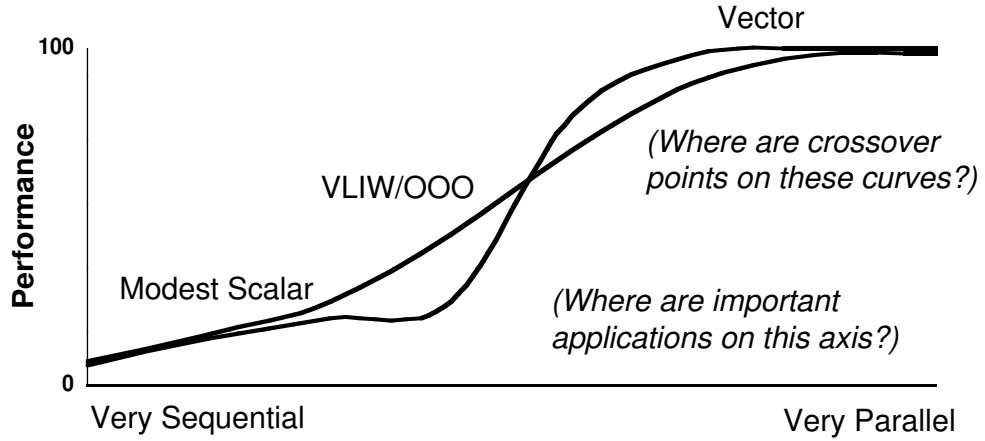
Vectors Lower Power

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 Even Worse

Superscalar	Vector
■ Control logic grows quad-ratically with issue width	■ Control logic grows linearly with issue width
■ Control logic consumes energy regardless of available parallelism	■ Vector unit switches off when not in use
■ Speculation to increase visible parallelism wastes energy	■ Vector instructions expose parallelism without speculation
	■ Software control of speculation when desired:
	– Whether to use vector mask or compress/expand for conditionals

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



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?

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)

DSP vs. General Purpose MPU

- DSPs tend to be written for 1 program, not many programs.
  - Hence OSes are much simpler, there is no virtual memory or protection, ...
- DSPs sometimes run hard real-time apps
  - You must account for anything that might happen
  - All possible interrupts or exceptions must be accounted for and their collective time be subtracted from the time interval.
  - Therefore, exceptions are BAD!
- DSPs have an infinite continuous data stream

DSP Data Path: Rounding

- Even with guard bits, will need to round when store accumulator into memory
- 3 DSP standard options
- Truncation: chop results => biases results up
- Round to nearest:
  < 1/2 round down, ≥ 1/2 round up (more positive)
  => smaller bais
- Convergent:
  < 1/2 round down, > 1/2 round up (more positive),
  = 1/2 round to make lsb a zero (+1 if 1, +0 if 0)
  => no bais (IEEE 754 round to nearest even)

DSP Data Path: Precision

- Word size affects precision of fixed point numbers
- DSPs have 16-bit, 20-bit, or 24-bit data words
- Floating Point DSPs cost 2X - 4X vs. fixed point, slower than fixed point
- DSP programmers will scale values inside code
  - SW Libraries
  - Seperate explicit exponent
- "Blocked Floating Point" single exponent for a group of fractions

DSP Data Path: Multiplier

- Specialized hardware performs all key arithmetic operations in 1 cycle
- ≥ 50% of instructions can involve multiplier
  => single cycle latency multiplier
- Need to perform multiply-accumulate (MAC)
- n-bit multiplier => 2n-bit product

DSP Addressing: Buffers

- DSPs dealing with continuous I/O
- Often interact with an I/O buffer (delay lines)
- To save memory, often organized as circular buffer
- What can do to avoid overhead of address checking instructions for circular buffer?
  - Keep start register and end register per address register for use with autoincrement addressing, reset to start when reach end of buffer
- Every DSP has "modulo" or "circular" addressing

DSP Addressing: FFT

- FFTs start or end with data in bufferfly order

0 (000)	=>	0 (000)
1 (001)	=>	4 (100)
2 (010)	=>	2 (010)
3 (011)	=>	6 (110)
4 (100)	=>	1 (001)
5 (101)	=>	5 (101)
6 (110)	=>	3 (011)
7 (111)	=>	7 (111)

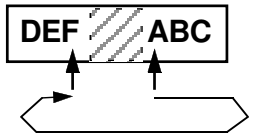
- What can do to avoid overhead of address checking instructions for FFT? "bit reverse" address
- Many DSPs have "bit reverse" addressing for radix-2 FFT

DSP Instructions

- May specify multiple operations in single instruction
- Must support Multiply-Accumulate (MAC)
- Need parallel move support
- Usually have special loop support to reduce branch overhead

VIRAM support for DSP, cont'd

- Autoincrement addressing?
- Circular addressing?



- FFT bit reverse addr?
- Vector load, store do this: access 32 sequential words
- Can match to vector load just with duplicate buffer:

  - Write twice on input
  - Use vector load, set maximum vector length to buffer size
- Do FFT differently

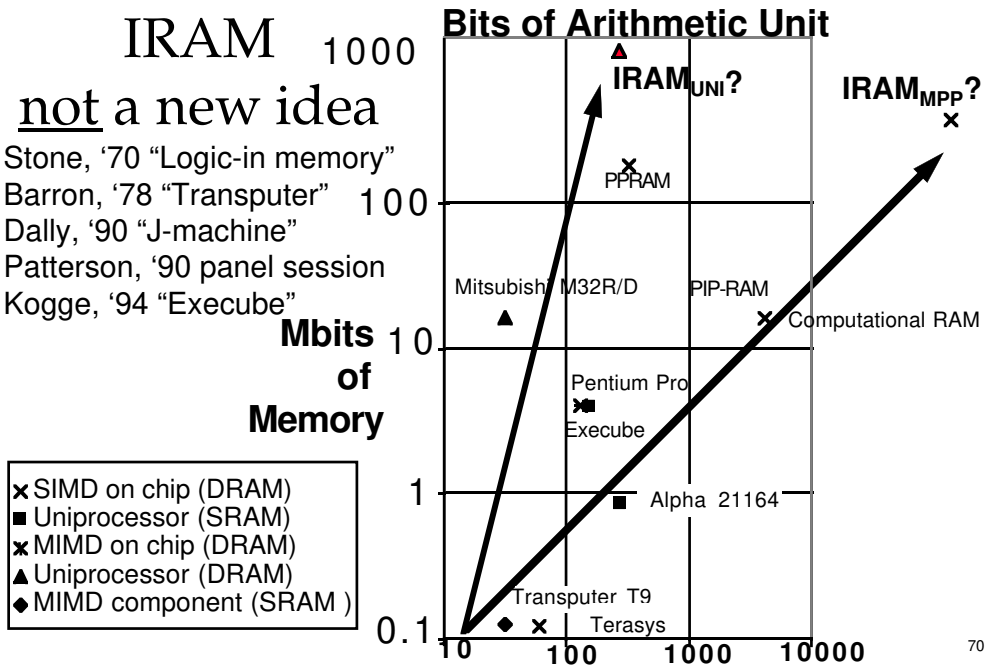
DSP Addressing

- Want to keep MAC datapath busy
- Assumption: any extra instructions imply clock cycles of overhead in inner loop
  => complex addressing is good
  => don't use datapath to calculate fancy address
- Autoincrement/Autodecrement register indirect
  - lw r1,0(r2)+ => r1 <- M[r2]; r2<-r2+1
  - Option to do it before addressing, positive or negative

IRAM

not a new idea

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



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)

"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

"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

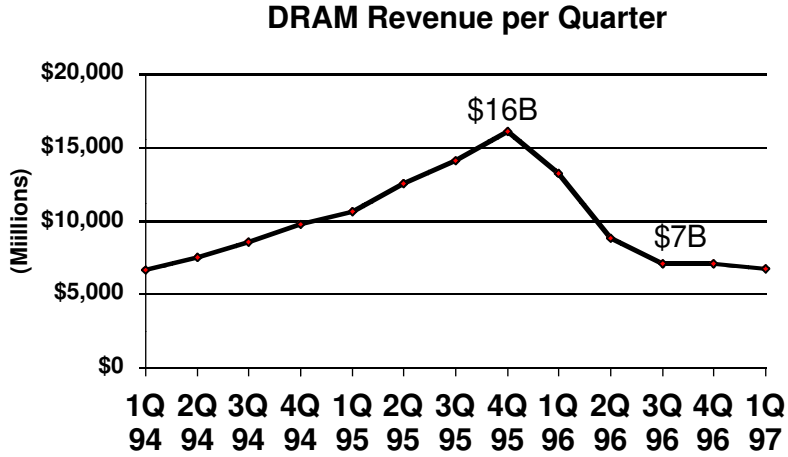
"Vanilla" Approach to IRAM

- 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

Today's Situation: DRAM

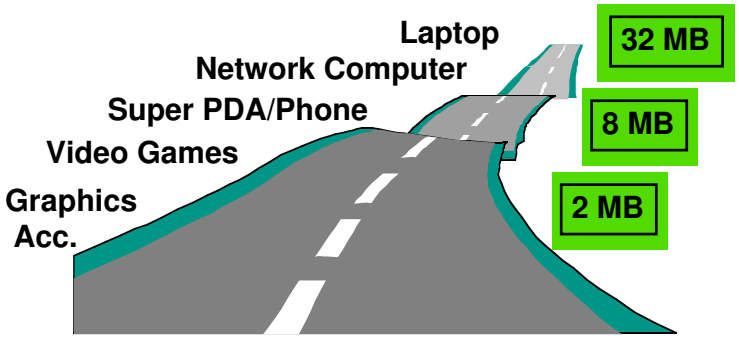
- Commodity, second source industry
  ⇒ high volume, low profit, conservative
  - Little organization innovation (vs. processors) in 20 years: page mode, EDO, Synch DRAM
- DRAM industry at a crossroads:
  - Fewer DRAMs per computer over time
    - » Growth bits/chip DRAM : 50%-60%/yr
    - » Nathan Myhrvold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

Today's Situation: DRAM



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

Commercial IRAM highway is governed by memory per IRAM?



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

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

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 (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

Vectors Are Inexpensive

Scalar

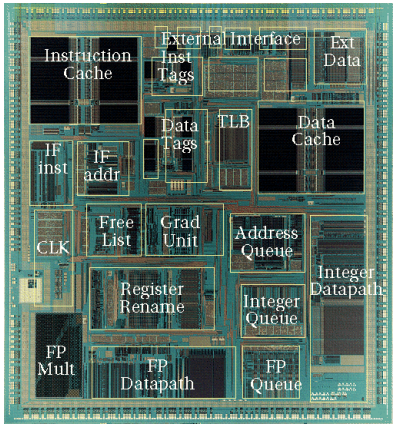
- N ops per cycle
  $\Rightarrow O(N^2)$ circuitry
- HP PA-8000
  - 4-way issue
  - reorder buffer: 850K transistors
    - incl. 6,720 5-bit register number comparators

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



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*

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

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?

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

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

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