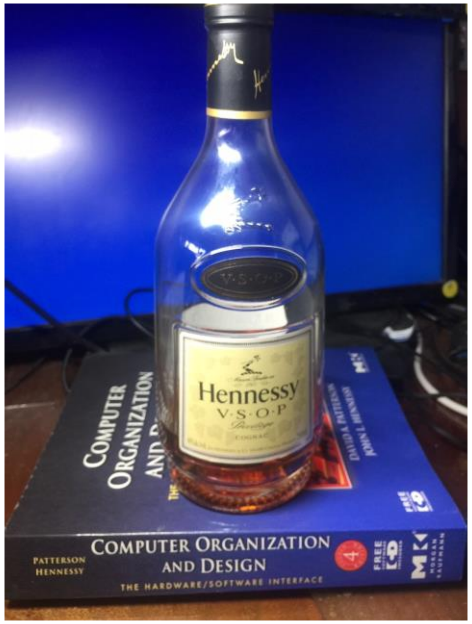
MIPS ISAs & Other Features
History, why's, mistakes & omissions
but one of the longer survivors!


Hennessy
V·S·O·P
COMPUTER ORGANIZATION AND DESIGN
THE HARDWARE/SOFTWARE INTERFACE
PATTERSON
HENNESSY

John R. Mashey JohnMashey@yahoo.com, @JohnMashey

February 24, 2017 (updated from 02/24/17, 11/20/97)

UC Santa Barbara – CMPSC 154 – special edition

Speaker – John R. Mashey - en.wikipedia.org/wiki/John_Mashey

PA

- Small farm in (hilly) Western Pennsylvania N of Pittsburgh, founded ~1850
  - Climate matters; keep topsoil; fix things; recycle; Liebig's Law. *smog, oil, coal, Marcellus shale*
- **Pennsylvania State University, 1964-1973, BS Math, MS/PhD CMPSC**

New Jersey

- **Bell Labs 1973-1983, early UNIX, MTS → Supervisor**
  - Programmer's Workbench, shell programming, text processing, workload measurement/tuning in first UNIX computer center, etc
  - Created & managed group with software + cognitive psychologists

Silicon Valley

- **Convergent Technologies 1983-1984 ($400M), MTS → Director Software**
- **MIPS Computer Systems 1985-1992, ($150M) Mgr. OS → VP Systems Technology**
  - System coprocessor, TLB, interrupt-handling; byte addressing(!);64-bit; Hot Chips 1989-2016
  - MIPS Performance Brief editor; a SPEC benchmarking group founder 1988- *(science, statistics)*
- **Silicon Graphics 1992-2000 ($3B),Dir. Systems Technology→ VP & Chief Scientist**
  - MIPS R10000 & later architecture, including performance counters & software
  - ccNUMA system architecture (NUMAflex in Origin3000, Altix); supercomputers for NCAR, etc
  - Performance issues in HPC, DBMS; technology forecasting, software strategy, **Big Data**
  - Evangelist work with sales/marketing, business development, alliances, scientists ♡ ☹**1995**
- **2001- Typical Silicon Valley "Semi-retired"** some consulting for high-tech co's, VCs

  Computer History Museum Trustee; Travel; ski in B.C.; hike; bike; occasionally write articles, do talks

  Technical advisory boards … interest in climate 2001-, and then a weird hobby 2007-

  Help climate scientists, expose machinery of doubt, blog, rocks in gears. 2012- CTCRE @ UCSF
- Committee for Skeptical Inquiry (CSI) Scientific/Technical Consultant

A few references

Stanford

- Hennessy, J. L., Jouppi, N., Baskett, F. and Gil, J., (1981) “MIPS: A VLSI Processor Architecture,” Proc. CMU Conference on VLSI Systems and Computations, pp.337-346, Computer Science Press, http://i.stanford.edu/pub/cstr/reports/csl/tr/81/223/CSL-TR-81-223.pdf
  John Hennessy, Norman Jouppi, Steven Przybylski, Christopher Rowen, Thomas Gross, (Feb 1983) “Design of a High Performance VLSI Processor”, Technical Report No. 236, Stanford University. http://i.stanford.edu/pub/cstr/reports/csl/tr/83/236/CSL-TR-83-236.pdf
  2-micron, 1-metal nMOS, 4MHz, word-addressing, no halfwords, **no “real” MMU**
- Hennessy, J. L., (1984), VLSI Processor Architecture, *IEEE Transactions on Computers,* C-33, no 12, pp. 1221-1246. http://ieeexplore.ieee.org/abstract/document/1676395/

MIPSco

- J. R. Mashey, RISC, MIPS, and the Motion of Complexity, *UniForum February1986 Proceedings*, Anaheim, CA pp. 116-124 https://books.google.com/books/about/UniForum_1986.html?id=koY_AQAAIAAJ
- C.Rowen,L.Crudele,D.Freitas,C.Hansen,E.Hudson, J.Kinsel, J.Moussouris, S.Prybylksi, T. Riordan, RISC VLSI Design for System-Level Performance, *VLSI Systems Design* March 1986 pp.81-88.
- J. Moussouris, L. Crudele, D. Freitas, C. Hansen, E. Hudson, R. March, S. Prybylski, T. Riordan, C. Rowen, D. Van’t Hof, A CMOS RISC Processor with Integrated System Functions, *COMPCON, page 126-131. IEEE Computer Society,* (March 4-6 1986) http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1663029
- F. Chow, M. Himelstein, E. Killian, L. Weber, Engineering a RISC Compiler System, *COMPCON, page 132-137. IEEE Computer Society,* (March 4-6 1986)
- M. DeMoney, J. Moore, J. Mashey, Operating Systems Support on a RISC, *COMPCON, page 138-143. IEEE Computer Society,* (March 4-6 1986)
- David A. Patterson, John L. Hennessy Computer Organization and Design – The Hardware/Software Interface (2013, 4th Edition) or MIPS 5th Edition(2017)

Section

Introduction

- **Overview of MIPS Chip History, especially 1984-2000**
- **In the beginning – design issues in a crazy year**
- **Primary Instruction Set Architectures (ISAs)** ***(~2 too many)***
  - **Stanford MIPS (~1982) 4MHz**
    - 32-bit ISA, 16 registers, word-addressed, no halfwords, MMU (ugh)
  - **MIPS-I: R2000 (1986, 8=>16.7Mhz) R3000 (1988, 25=>40Mhz)**
    - 32 bit ISA, 32-bit datapaths; SMP, fault-tolerant (FT) possible
  - **MIPS-II: R6000 (1990, 60, 80Mhz, ECL)**
    - 32-bit ISA, more instructions, with 64-bit datapaths
  - **MIPS-III: R4000 (1992), R4400(1993), R4300i (1995, low-cost, N64)**
    - 64/32-bit ISAs, 32 FP regs, 64-bit datapaths; SMP, FT much better
  - **MIPS-IV: R8000 (SGI 1994), R10000 (1996, 4-issue speculative)**
    - FP MADD, etc; Prefetching; supercomputing; scalability; latency
- **Mistakes, maybes, or wishes had there been more time**
- **ISA design very tough game, many try, few survive even 1 decade**

Instruction Set Architectures - History from 1989

- **Hot Chips 1 – 1989 https://www.hotchips.org/archives/hc01/**
  - ~~**National Semiconductor NS32GX32 (NS32032, etc)**~~
  - **Intel 486 (X86: CISC with increasingly RISC-like internals)**
  - **Motorola 68040 (Freescale→NXP ColdFire, RISCized controllers)**
  - **Sun SPARC – CMOS, ~~ECL~~, ~~GaAs~~ (Oracle quit, Fujitsu still doing)**
  - **MIPS (1986 R2000, 1988 R3000)**
  - ~~**Motorola 88K**~~
  - ~~**Intel i860, Intel i960**~~
  - ~~**AMD 29000 (word-addressed)**~~
  - ~~**Intergraph Clipper**~~

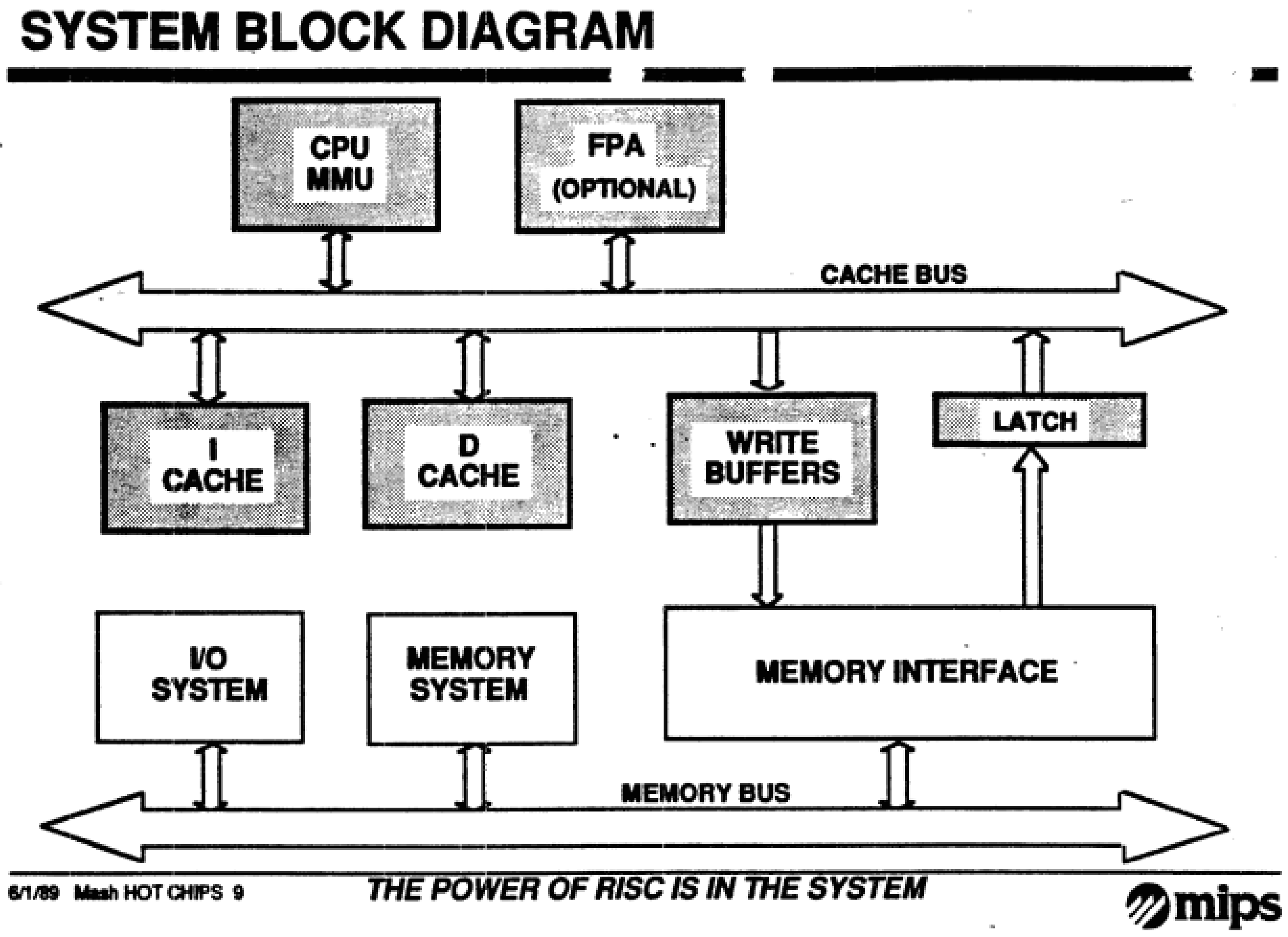
RISC vs CISC from comp.arch **https://yarchive.net/comp/risc_definition.html**

Active in 1989, but not at Hot Chips: ~~HP PA-RISC~~, early ARM

Later

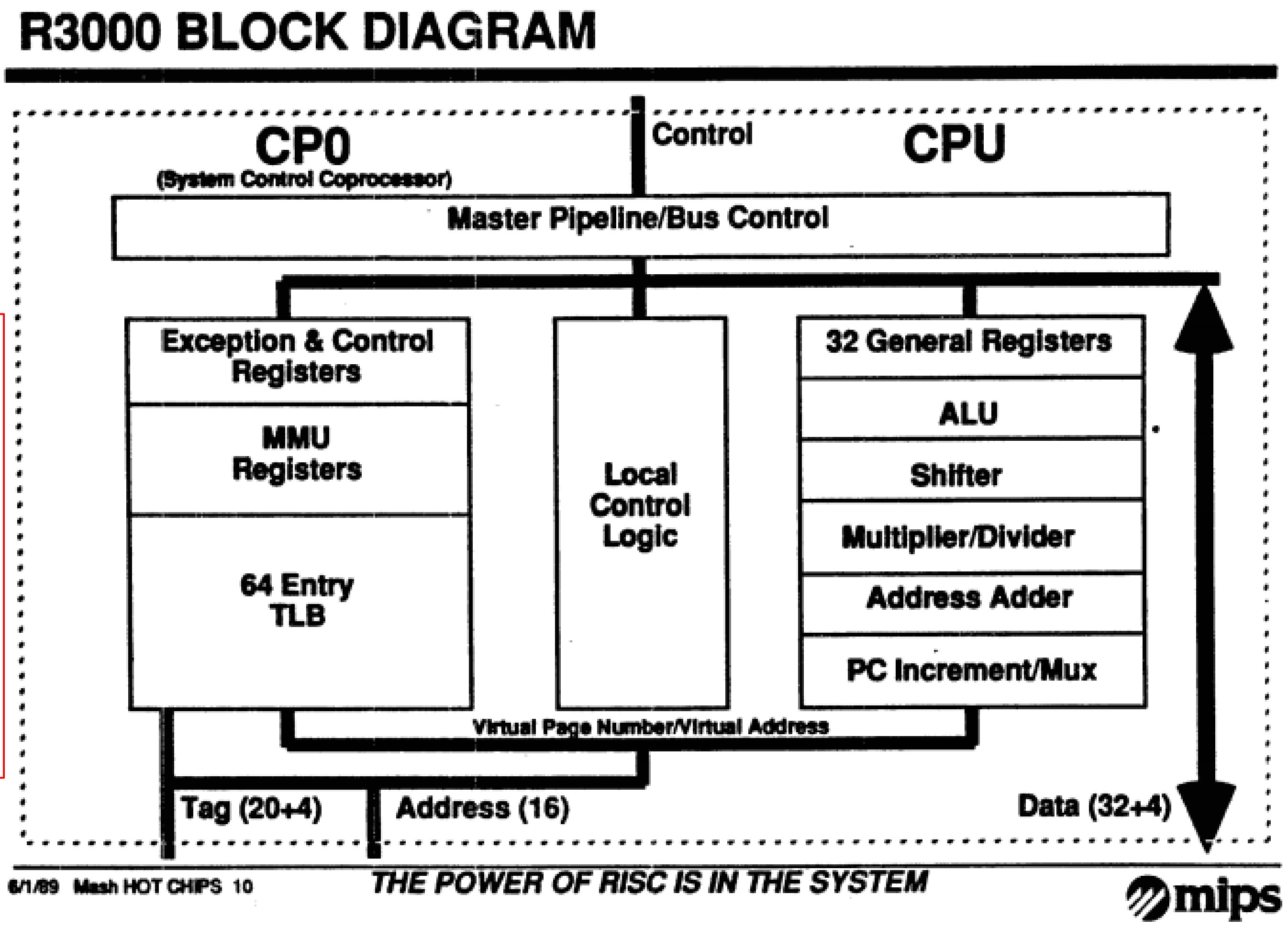
- **POWER / PowerPC; ~~Alpha (word addressed at first), Intel Itanium~~**
- **~~Mainframe (~~except ~1964 IBM S/360) & ~~minicomputer ISAs ~gone~~**
- **Tensilica (1997), part of Cadence since 2013**
- **RISC V (2010-)**

Brief Review (from my 1989 Hot Chips talk on R3000)



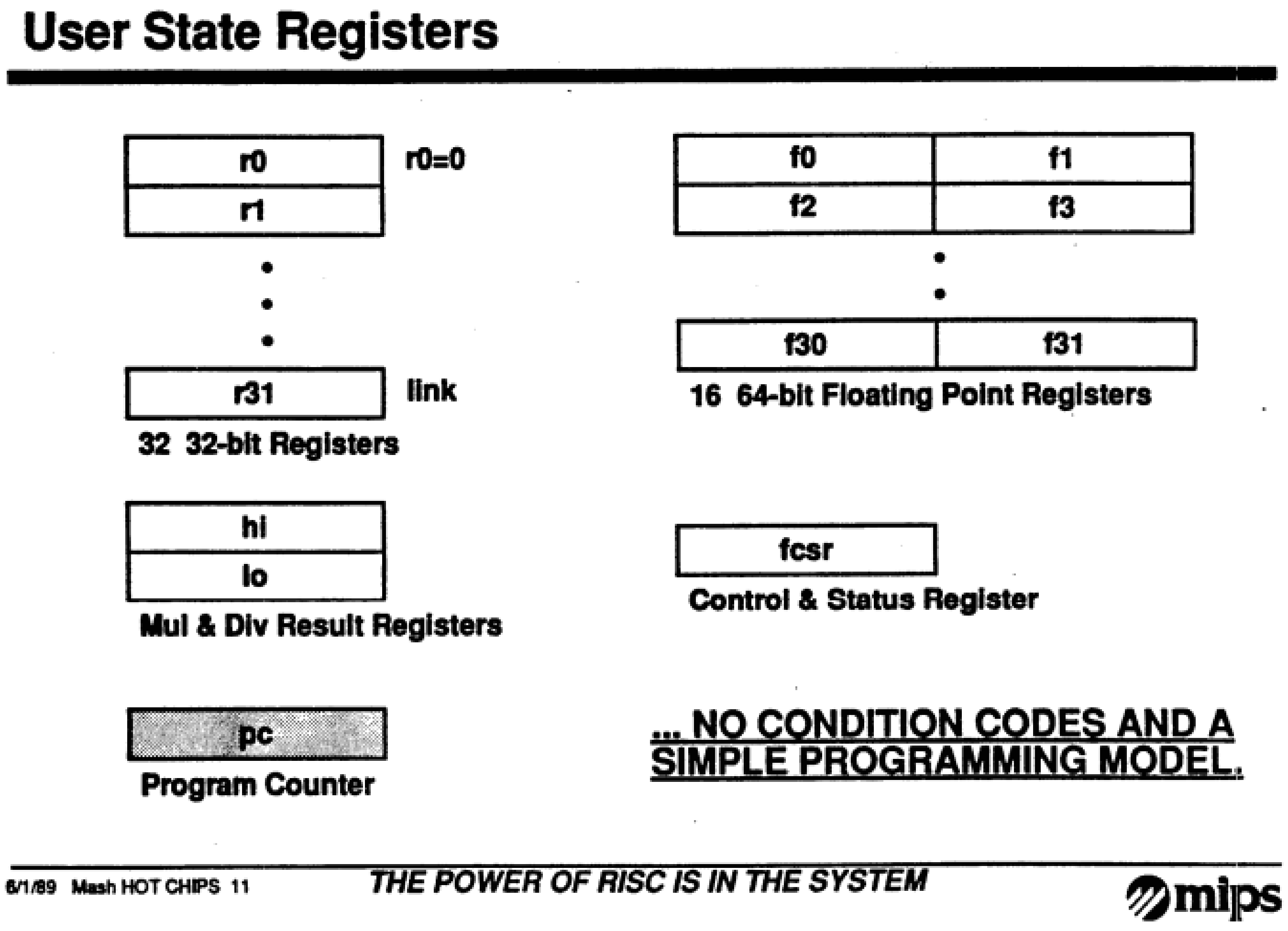
5.2
Direct-mapped
SRAM Caches,
32-bit access
D-cache
Write-through

Brief Review (from my 1989 Hot Chips talk on R3000)



5.4 Software-Managed Translation Lookaside Buffer (TLB) … unusual, many thought crazy

Brief Review (from my 1989 Hot Chips talk on R3000) 2



User State Registers

32 32-bit Registers

16 64-bit Floating Point Registers

Mul & Div Result Registers

Control & Status Register

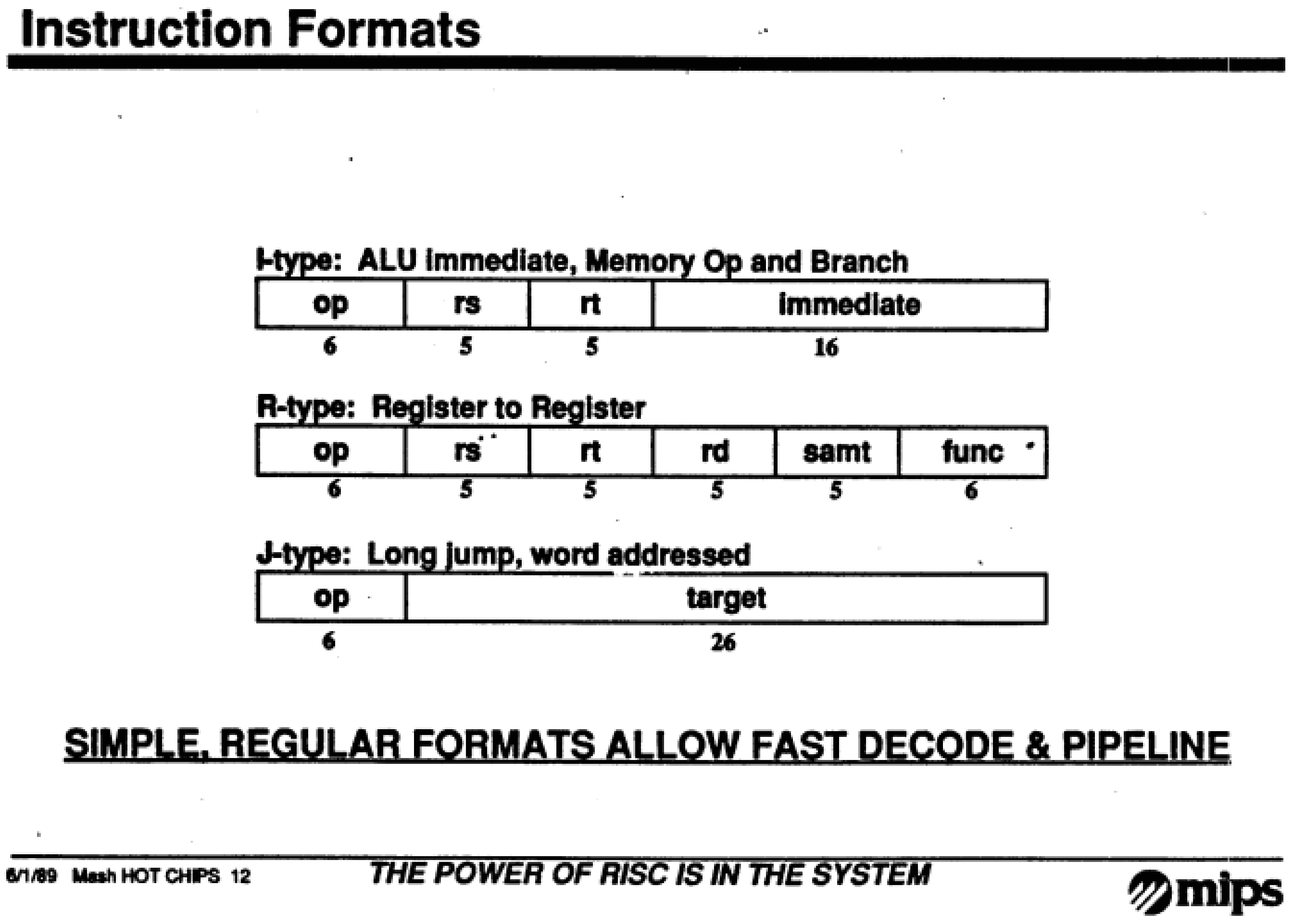
Program Counter

... NO CONDITION CODES AND A SIMPLE PROGRAMMING MODEL.

6/1/89 Mash HOT CHIPS 11 *THE POWER OF RISC IS IN THE SYSTEM* mips

Brief Review (from my 1989 Hot Chips talk on R3000)

2



Instruction Formats

I-type: ALU Immediate, Memory Op and Branch

op	rs	rt	immediate
6	5	5	16

R-type: Register to Register

op	rs	rt	rd	samt	func
6	5	5	5	5	6

J-type: Long jump, word addressed

op	target
6	26

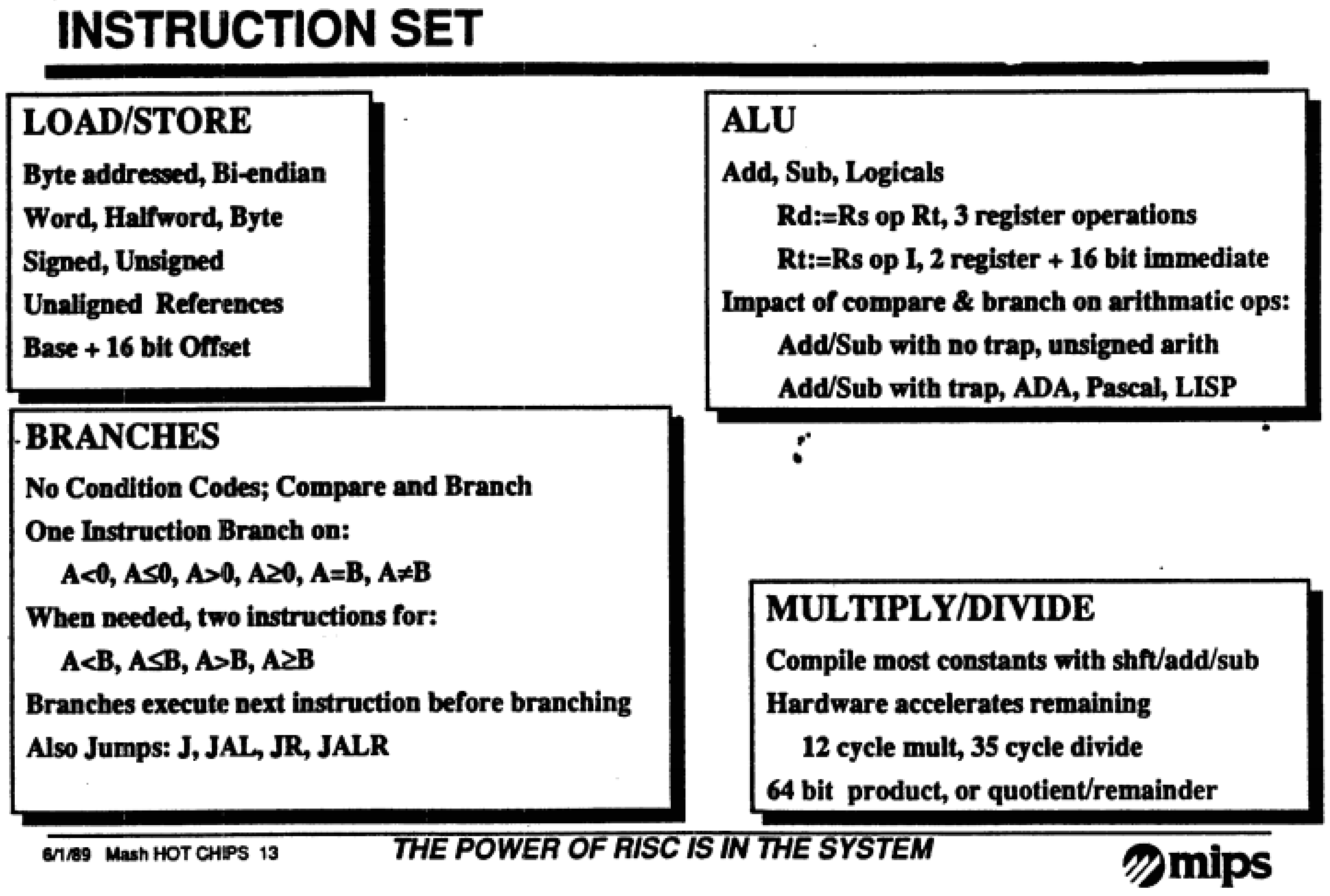
SIMPLE, REGULAR FORMATS ALLOW FAST DECODE & PIPELINE

6/1/89 Mash HOT CHIPS 12

THE POWER OF RISC IS IN THE SYSTEM

mips

Brief Review (from my 1989 Hot Chips talk on R3000) 2



INSTRUCTION SET

LOAD/STORE

Byte addressed, Bi-endian
Word, Halfword, Byte
Signed, Unsigned
Unaligned References
Base + 16 bit Offset

ALU

Add, Sub, Logicals
- Rd:=Rs op Rt, 3 register operations
- Rt:=Rs op I, 2 register + 16 bit immediate

Impact of compare & branch on arithmatic ops:
- Add/Sub with no trap, unsigned arith
- Add/Sub with trap, ADA, Pascal, LISP

BRANCHES

No Condition Codes; Compare and Branch
One Instruction Branch on:
- A<0, A≤0, A>0, A≥0, A=B, A≠B

When needed, two instructions for:
- A<B, A≤B, A>B, A≥B

Branches execute next instruction before branching
Also Jumps: J, JAL, JR, JALR

MULTIPLY/DIVIDE

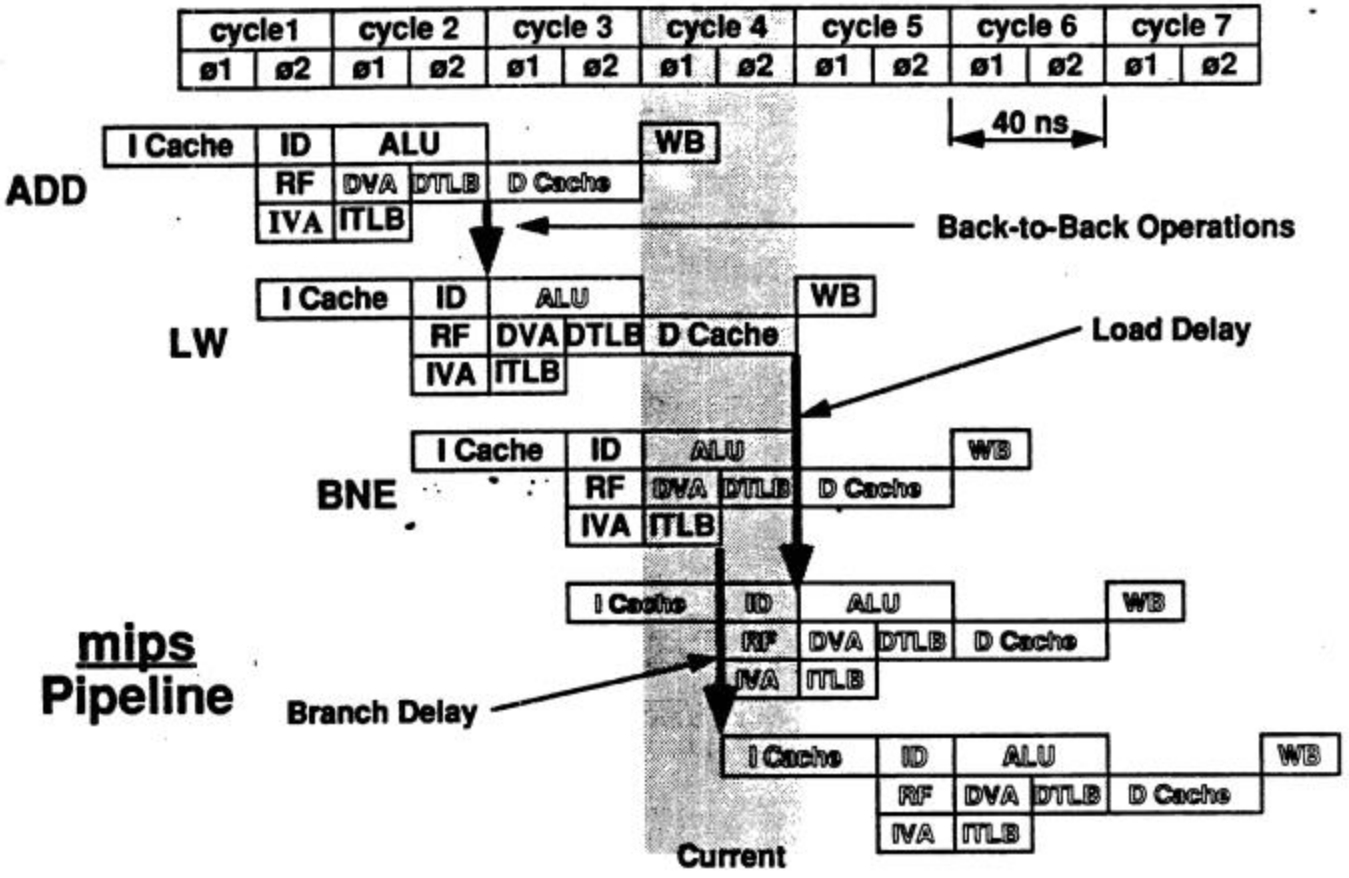
Compile most constants with shft/add/sub
Hardware accelerates remaining
- 12 cycle mult, 35 cycle divide

64 bit product, or quotient/remainder

6/1/89 Mash HOT CHIPS 13 *THE POWER OF RISC IS IN THE SYSTEM* mips

Brief Review (from my 1989 Hot Chips talk on R3000) 4

Single-issue
5-stage pipeline
Forwarding, but
No hazard-stall
for Loads,
Assembler tried
To fill load-delay
Slot or insert NOP.



Guidelines

- When in doubt, leave it out … usually
- Most additions had to be justified by
  Performance … 1% from simulations
  Functionality, like SYSCALL
- MIPS ISA designed from C, Fortran, Pascal statistics, with a little thought about PL/I, COBOL. Nothing special for Lisp, Smalltalk.
  Contrast: HP PA RISC had a few instructions to help COBOL, given market.
- Fallacy:Write in assembly language to obtain the highest performance. **2.18**
  In early MIPS days, I wrote assembly versions of *strcpy, strcmp*, etc… because one always did that … and threw them out. C compiler code as good.
- Always constrained by previous choices: **path dependency**
- Try to think ahead to likely future implementations
  Minimize implementation artifacts (we didn't always do that)
  Load-delays … not so good, should have had hazard detect and stall **3.5**
  Implementation of multiply & divide
  32x32-bit FP registers was worst mistake, should have been 32x64-bit

In the beginning – Design Issues - R2000 (1986), R3000 (1988)

- Design requirements
  - Good for multiple UNIX versions, other OS (embedded)
  - CPU + efficient FPU connection + SRAM cache
  - Crazed schedule for R2000 (4Q84 – 3Q85):
    create new architecture … systems … software < 2 years
  - Manufacturable die in 2 micron (2,000 nm!) CMOS, 144-pin package
- Design implications
  - Integer unit, big full-associative TLB, on-chip cache control
  - External FPU, direct access to/from SRAM & DRAM
  - Minimalist design “only just barely works”
  - Die space and pins limited
  - 110K transistors in CPU (1986), 70K in R2010 FPU (1987)

RISC, MIPS and the Motion of Complexity (1986)

- Movement of complexity among chips, hardware system, compilers & OS

Figure 4 shows the design style that we actually employ. Everyone owns a part of the complexity that they find tolerable. A small part has disappeared (to the left) because groups have found cases where they could work together to lessen the total complexity.

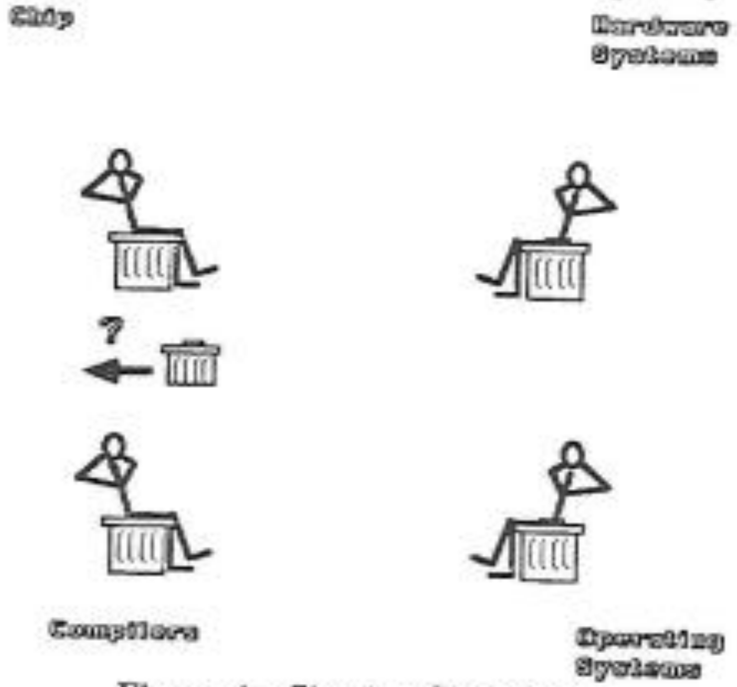


Figure 4 - Sharing Complexity

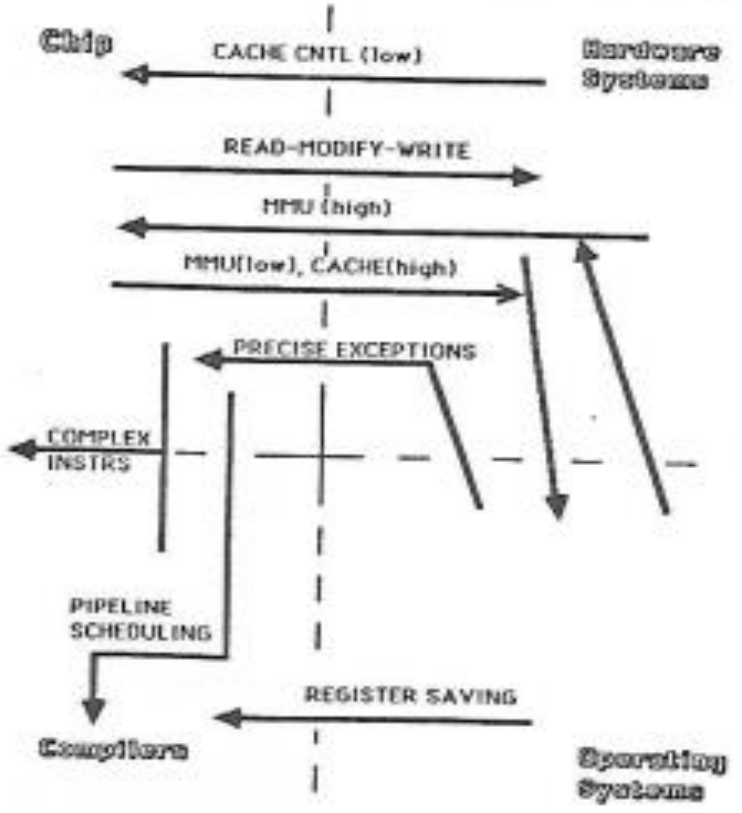


Figure 5 - Motion of Complexity

4.4. Sequential vs Pipeline Figure 6 illus-

MIPS-I (R2000, R3000)

2

4Q84 Start with Stanford MIPS experience, ~~original idea: commercialize that~~
- 16 registers, some 16-bit instructions
- Word-addressed to save gate delay, no halfword operations (eek!)
- MMU not from OS people!
- Academic chips can/should explore ideas, commercial needs 100%

12/84 Make competitive, think about IBM 801 papers, commercial realities
- 32 registers, 32-bit-only instructions
- Byte-addressed, words, halfwords (Hennessy changed mind!)

1/85 – 2/85: Fine-tune user ISA via compiler analysis & functional needs

3/4/85 Instructions

multiply & divide integer hardware added
load-byte/half unsigned kept
variable shifts kept
ADDIU added
LUI (yes)
ORUI (rejected)
Extends (byte, halfword) rejected, given signed loads, SLL, SRA
Opcode X may or may not be worthwhile, depending on existing ops

MIPS-I

2

3/85: byte order and alignment discussions; Bi-Endian for Daisy (!) 2

3/85-7/85: interfaces, kernel features; FP specs (COP1, trapping)

Some CPUs kept “FP registers dirty”, better to trap, lower overhead

4/85: MMU

COP0 eliminated special operations, same trapping mechanisms

4/21/85 register conventions tweaked

3Q85: last ISA tweaks

LWL, LWR, SWL, SWR – worry about unaligned Fortran variables

4Q85 Tapeout, first chips, Christmas present

1Q-2Q86 debug OS & system, 1st demoes

4Q86 ship systems to customers

2Q87 R2010 FPU

Insane schedule … and done with *MacProject* on my 128K Mac.

R2000 (1986), R3000 (1988)

- CPU – Compiler interactions
  - User-level Instruction Set Architecture (ISA)
    mostly spec'd by compiler writers + performance analysis
    Assumed global optimizing compiler (from Fred Chow's work @ Stanford)
  - UNIX kernel wish to be optimized (1Q86, 1st systems)
    CPU provided uncached attribute for devices
    ANSI C "volatile" attribute just in time!
    Horrific bug – needed binary search on optimized UNIX kernel
    Cache bugs with device drivers for tape drives
    New ISA, new chips, new system, new UNIX port, ~new compilers
    Don't do this unless you absolutely must!

R2000 (1986), R3000 (1988)

- CPU – Operating system interactions
  - Integrating Virtual Memory, TLBs, Caches (P&H) **5.4**
    Physical-indexed, physical-tagged caches to ease OS ports, no aliasing
    explicit cache management not widely present in 1985
  - MMU/exceptions/COP0 mostly spec'd by OS people (me+2 others)
    Software TLB refill for flexibility & paranoia of experienced OS people from past MMU/exception horrors … paranoia rewarded!
    Months of BSD UNIX, then AT&T UNIX port found new bug, S/W fix

  - I think only MIPS & HP did this, many thought nuts, but works fine. (P&H)
    I wrote TLB miss handler, then tuned with minimal hardware for speed
    CPU had no hardware for finding & accessing Page Tables in memory.
  - Omitted: many CPUs had hardware-set reference & dirty bits.
    UNIXes sometimes turned those off anyway, as in wanting to trap first write to a page for Copy-on-Write policies.
    We told logic designers we did not want those features (relief).
    Possible to do SMP OS, just barely (DeMoney, Moore, Mashey (1986))

  - Sometimes careful omission of features → better in every way

R2000 (1986), R3000 (1988)

6 or 7

- CPU – Hardware interactions for Symmetric MultiProcessing (SMP)
  - Cache-control on-chip, not outside logic
  - Physical-address, physical tags helped
    R2000: SMP possible, very tricky timing, SGI made it work, barely
    R3000: much easier, 2 more external signals
  - By 1988 feasible to ship cache-coherent SMP, even in workstation-sizes

MIPS-II

2

8/21/87: Additions to user ISA for R6000 (multichip ECL implementation)

Much performance modeling, analysis of real code

~~Y 1. Load double / store-double integer~~ (later removed when 64-bit known)

Y 2. Load double / store double floating coprocessor (Yes!)

Y 3. Branch-likely

N 4. Branch-unlikely

Y 5. Convert with explicit rounding mode

N 6. Eliminate delay slot after loads

Y 7. Load-linked – Store Conditional (flexible synchronization) **2.11**

N 8. Floating-load immediate

Y 9. SQRT

Y 10.Trap instructions for ADA

MIPS-III

- 4Q88: Decide R4000 would be 64/32-bit extension of MIPS-II
- 1Q89: more details
- 4/89 C models, start of discussions for “long long”
  Later, 1991/1992 this got settled by industry, defacto standard
  John Mashey, The Long Road to 64 Bits (2006)
  http://queue.acm.org/detail.cfm?id=1165766 OR
  http://cacm.acm.org/magazines/2009/1/15667-the-long-road-to-64-bits/fulltext
- C 99 Rationale – long long p.37-41
  http://www.open-std.org/jtc1/sc22/wg14/www/C99RationaleV5.10.pdf

- 4Q91 – superpipelined, single-issue, on-chip L1 caches, control for L2,
  1st 64-bit microprocessor, SMP and FT features

  R4000 would have been 3-metal 2-issue superscalar if could have waited, which could have happened if we’d done an “R3500” with more pins, demuxed busses, to get to 50/66Mhz about the same time as HP Snakes, ~1990.
  Other ~1991 superscalar chips got better floating point, worrisome to SGI

MIPS-IV

R8000 (TFP) got started at SGI, multi-chip set, good at linear algebra

2Q92 SGI acquired MIPS

FP MADD, conditional moves, prefetch instructions, 4-issue superscalar

Very good for some kinds of floating point codes, but finicky

R1X000 much more flexible, lower cost, rapidly outperformed R8000
4-issue, speculative, out-of-order superscalar

Later designs, “Beast” & “Alien” never were completed, in favor of → Itanium.
Too bad: synchronization in big multiprocessors, Coherent-no-allocate caching.

Mistakes: my opinion, not too many bad ones

- **R2000/R2010 32-bit floating point registers & only 32-bit float ld/st** **3.5**
  - 32 singles = 16 doubles, agony for years
  - ALWAYS PLAN FOR BIGGER BUSSES
  - Hard call to have 64-bitters, given implied need for 64-bit float ld/st
  - But might have avoided wish for MIPS-II, a really good thing
  - INSTALLED BASE AND THIRD-PARTY SOFTWARE VENDORS
- Load delay slots (instead of hardware hazard stalls) **4.7**
  - Quickly fixed by R4000, but binaries last for long time
  - Bothersome NOPs for embedded footprint, UNIX kernel
  - INSTALLED BASE AND THIRD-PARTY SOFTWARE VENDORS
- Branch delay slots **4.8**
  - OK at the time, some pain later, and some odd bugs in R4000
  - Obviated by branch prediction, branch folding
- MIPS = <u>M</u>icroprocessor without <u>I</u>nterlocked <u>P</u>ipeline <u>S</u>tages
  - Maybe, but we had them for MFLO/MFHI & floating point
  - Over time, interlocks appeared …. might have been better if started
    Load stall, MUL / DIV operations that put results in GP registers, not HI/LO

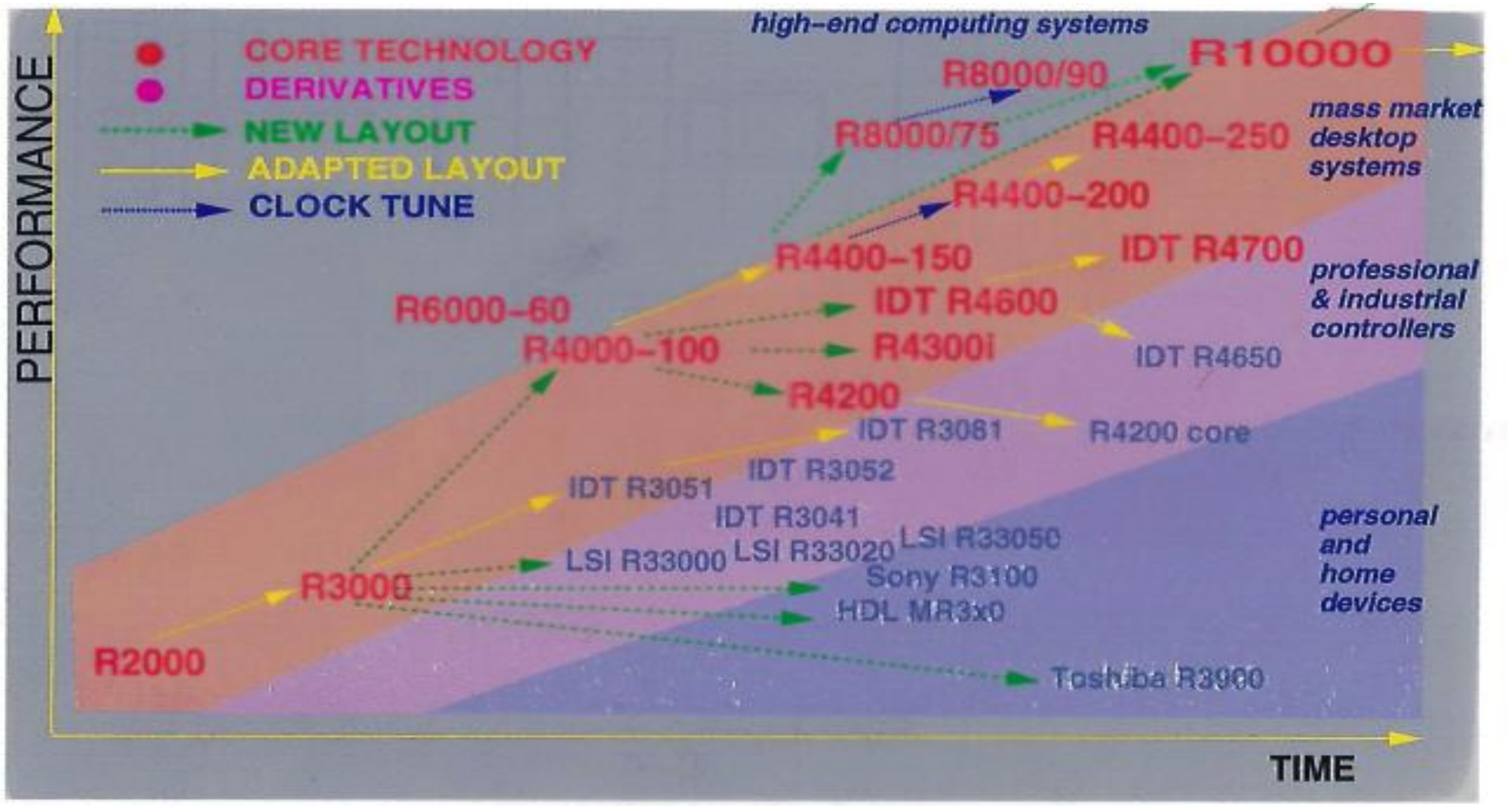
Not too bad, could have been better

- LWL/LWR SWL/SWR – {Load, store} Word {Left, Right}
  - Worry over legacy unaligned code, designers said byte networks existed
  - LWL/LWR oddly read destination register, merged data, unlike other Loads
  - SWL/SWR might need extra bit on bus for 3-byte cases
  - I should have fought harder for solution on next page, but no time
  - Unexpectedly useful for COBOL & PL/I … but not big markets for us
- Integer multiply/divide .. Good to have hardware, not just steps, but…
  - Violated normal rule of language use: 32 op 32 → 32, not 64 bits
  - 32x32 => 64-bit product and 32/64 => 32-quotient & 32-remainder
    Better (Alpha): Multiply Lower, Multiply Upper, Quotient, Remainder
    Interlock on use would have been better
    Irregularity & extra visible state caused pain,extra bit for R10000 later
- Lack of synchronization instructions in R2000/R3000
  - OK at time, and done on purpose (*nobody liked anything*)
  - Needed standard user libraries for test-and-set, etc, fast-path syscalls.
    I thought about this 2Q85, but life got hectic, and then it was too late.

Not mistakes, but really could have done better

- More interlocks lessen pipeline visibility – loads, eliminate LO & HI
- LWL/LWR/SWL/SWR use by compiler switches … wrong
  For mem*, str* functions, might have fought harder for other ops
- Third-party software vendors want one binary, hard to move
  Real problem for general-purpose systems, much less for embedded
- User-level intercept: if we'd had time, something like this for MIPS-I
  - Dedicate ~2-3 more integer registers
  - Intercept unimplemented opcode, transfer to emulation library
    Compile/link code with newest opcodes, but library to make work
  - Intercept unaligned references, and by run-time switch
    1) Fault, since it usually is (S/360 faulted, S/370 didn't, we asked!)
    2) Fix and return, just slow
    3) Record, fix and return, slow, but helps find bad code
  - May have been helpful for LISP, Smalltalk, ie.., tags
    normal operation fast, rare cases can be handled
- Ideally, just MIPS-I (R2000, R3000, "R3500") and
  MIPS-IV (2-issue superscalar "R4500") in 64/32 forms
  ECL R6000 decision caused cascade of challenges, unfortunately

History – proliferation, but ripple effects from decisions ~1996



19M+ units in 1996, 35M+ units in 1997!!!; 80+ versions
What once was high–end chip (R3000) ... now <$20 versions.

- **I've lost track since, but in 2010s, ~800 Million MIPS 32 or 64-bit cores were shipped / year. (Smaller than ARM, but still Billions out there.)**

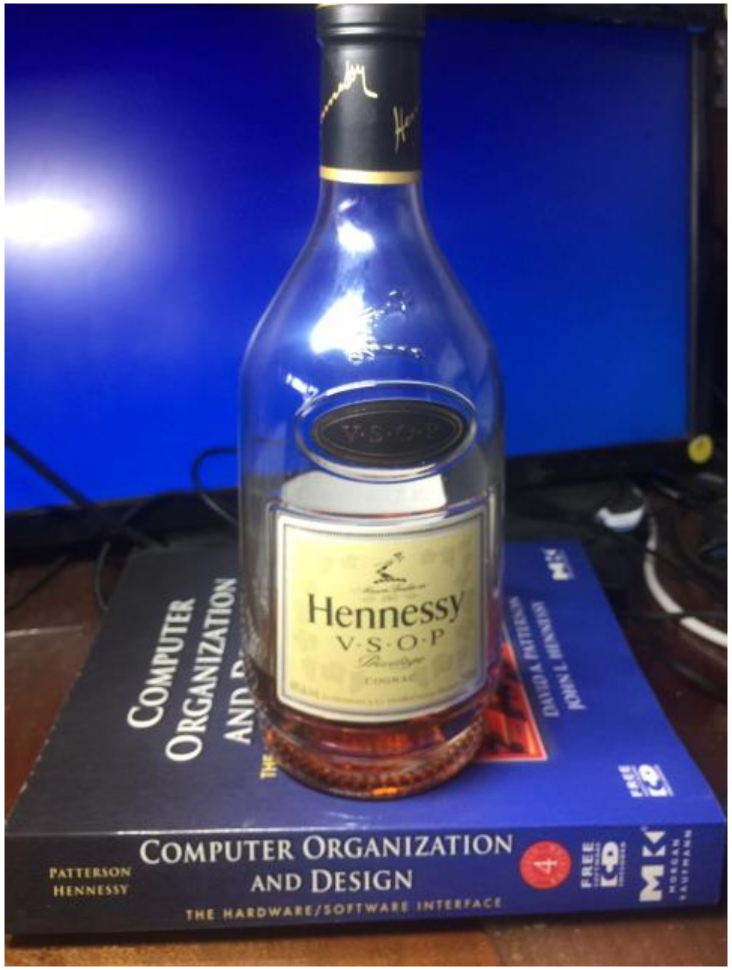
The irony of ARM’s embedded focus/success by Dave Jaggar

- **https://en.wikipedia.org/wiki/Dave_Jaggar**
  “**David Jaggar** …computer scientist who was responsible for the development of the ARM architecture between 1992 and 2000, redefining it from a low-cost workstation processor to the dominant embedded system processor. …
  Jaggar was born in 1967 in Christchurch, New Zealand … attended the University of Canterbury, where he gained a Bachelor of Science degree in Computer Science in 1987 and a Master of Science degree in Computer Science in 1991. His Master's thesis was titled *A Performance Study of the Acorn RISC Machine*, in which he exposed shortcomings of the early ARM designs.
  Jaggar is best known for creating the Thumb architecture to re-position ARM as an embedded processor.”
  My NZ lecture on RISC design inspired him, but also convinced him that ARM couldn’t match MIPS performance, so should aim for lower-power, denser code → great success in nascent market for personal devices!! Argh!!
  He told this story at recent Stanford talk and presented nice gift to me
- Lecture at Stanford – 05/29/19
  https://systemx.stanford.edu/events/seminar/20190529/bonus-lecture-arm-microprocessor-my-part-its-downfall

Dave Jaggar’s gift to me for inspiring him

- He said he’d also looked for Patterson cognac, could find none

V·S·O·P
Maison Fondée en 1765
Hennessy
V·S·O·P
Privilège
COGNAC
750ML
JOHN L HENNESSY & DAVID A PATTERSON
COMPUTER ARCHITECTURE A QUANTITATIVE APPROACH



Conclusions

- Baggage accumulates, so start small
- Academic computer architecture can/should explore features, commercial architecture must solve 100% of (sometimes boring) issues
- Seemingly-minor decisions can have decades-long effects
- In 1985, one never would have expected MIPS ISA to be a long survivor
- Look at RISC-V, some elements will be familiar

- Advertisement: visit Computer History Museum, Mountain View, CA
  You can see computers, chips, software, etc.
  https://computerhistory.org/

Extras