Introduction

CMSC 611: Advanced Computer Architecture

Some material adapted from Mohamed Younis, UMBC CMSC 611 Spr 2003 course slides
Some material adapted from David Culler, UC Berkeley CS252, Spr 2002 course slides, © 2002 UC Berkeley
Some material adapted from Hennessy & Patterson / © 2003 Elsevier Science

Overview

- Resources, syllabus, work load
- Grade structure and policy
- Expected background
- An introduction to computer architecture
- Why study computer architecture?
- Organization and anatomy of computers
- Impact of microelectronics technology on computers
- The evolution of the computer industry and generations

Course Resources

- Instructor: Marc Olano / ITE 354
  - Office Hours: Tue Thu 2:45 – 3:45
- TA: Yifang Liu / ITE 340
  - Office Hours: Wed 4:00 – 6:00
- Web Page:
  - **`www.cs.umbc.edu/~olano/611`**
- Book
  - Hennessy and Patterson, *Computer Architecture: A Quantitative Approach*, 3rd Edition

Syllabus

- Quantitative Design Principles
- Instruction Set Principles
- Pipelining and Instruction Parallelism
- Memory Hierarchy Design
- Storage and I/O
- Multiprocessor Systems
- Interconnection Networks

Workload

- Assignments
  - Approximately 2 hours, every other week
  - Mostly from book
- Exams
  - Midterm in class, Thursday October 16th
  - Final December 11th, 3:30 – 5:30
- Project

Project

- Teams of two
- You choose application area
- Design architecture for your application
- Final written report / architecture manual

Grades

- Breakdown
  - 20% Homework
  - 25% Midterm
  - 25% Final
  - 30% Project
- Homework late policy
  - Up to 1-week late, -20% of total points
  - One penalty-free late, requested in advance
  - >1 week late scores zero

Expected Background

- CMSC 411: Computer Architecture
  - Design of computer systems
    - Information representation
    - Floating point arithmetic
    - Hardwired & micro programmed control
    - Pipelining
    - Cache
    - Bus control & timing
    - I/O mechanisms
    - Parallel processing
- 411 focus on design and implementation (how)
- We focus on design decisions (why)

Introduction & Motivation

- Computer systems are responsible of 5-10% of the gross national product of the US
- WWW, ATM, DNA mapping, … are among the applications that were economically infeasible suddenly became practical
- You can be a part of this!
- Even if you don't want to do computer architecture, this class will
  - Help you understand the limits & capabilities of computing
  - Help you understand why
  - Tools of computer architecture apply everywhere!

Recent Developments

- Manipulating the instruction set abstraction
  - itanium: translate ISA64 -> micro-op sequences
  - transmeta: continuous dynamic translation of IA32
  - tinsilica: synthesize the ISA from the application
  - reconfigurable HW
- Virtualization
  - vmware: emulate full virtual machine
  - JIT: compile to abstract virtual machine, dynamically compile to host

Slide: David Culler, UCB

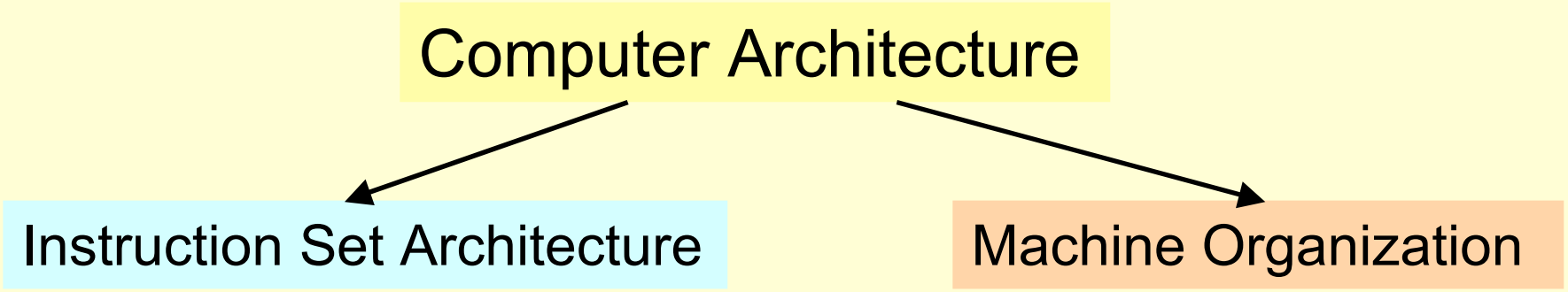
More Recent Developments

- Parallelism
  - wide issue, dynamic instruction scheduling, EPIC
  - multithreading (SMT)
  - chip multiprocessors
- Communication
  - network processors, network interfaces
- Exotic explorations
  - nanotechnology, quantum computing

What is "Computer Architecture"?

- Instruction set architecture
  - functional behavior of a computer system as viewed by a programmer (like the size of a data type – 32 bits to an integer).
- Computer organization
  - Structural relationships that are not visible to the programmer (like clock frequency or the size of the physical memory).
- The Von Neumann model is the most famous and common computer organization
  - Not the only (e.g. Harvard Architecture)

What is "Computer Architecture"?



- Interfaces
- Compiler/System View
- "Building Architect"

- Hardware Components
- Logic Designer's View
- "Construction Engineer"

Instruction Set Architecture

... the attributes of a [computing] system as seen by the programmer, i.e. the conceptual structure and functional behavior, as distinct from the organization of the data flows and controls the logic design, and the physical implementation.
– Amdahl, Blaaw, and Brooks, 1964

- Organization of Programmable Storage
- Data Types & Data Structures: Encoding & Representation
- Instruction Set
- Instruction Formats
- Modes of Addressing and Accessing Data Items and Instructions
- Exceptional Conditions

The instruction set architecture distinguishes the semantics of the architecture from its detailed hardware implementation

Slide: David Patterson, UCB

The Instruction Set: a Critical Interface

DEC Alpha	(v1, v3)	1992-1997
HP PA-RISC	(v1.1, v2.0)	1986-1996
Sun Sparc	(v8, v9)	1987-1995
MIPS	(MIPS I, II, III, IV, V)	1986-1996
Intel	(8086,80286,80386, 80486,Pentium, MMX, ...)	1978-2000

The instruction set can be viewed as an abstraction of the HW that hides the details and the complexity of the HW

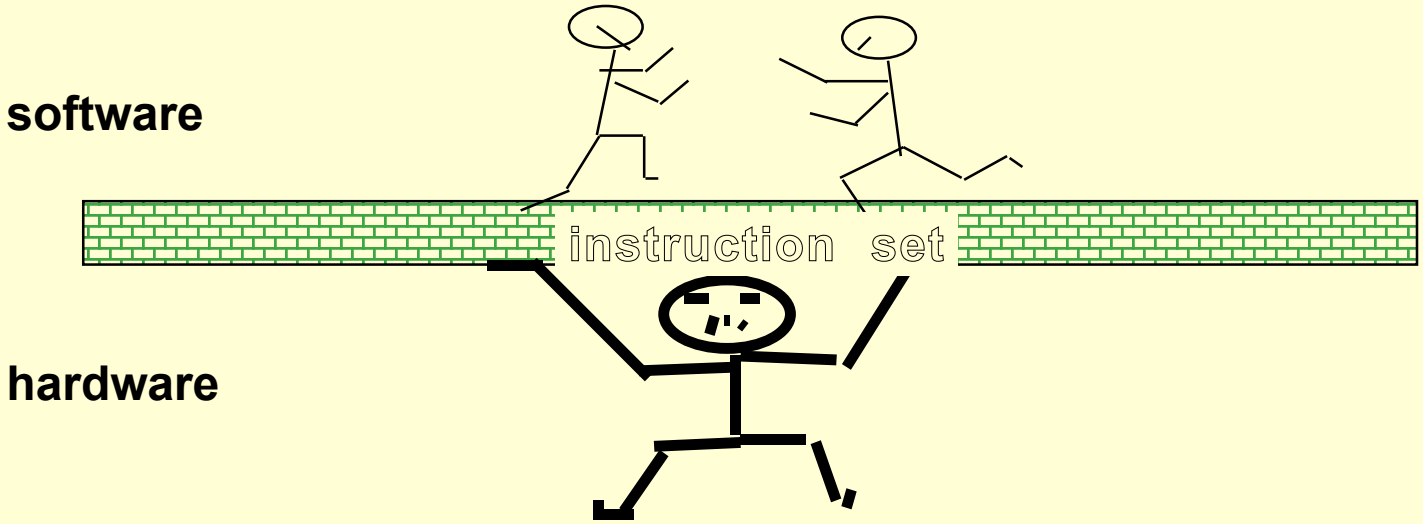
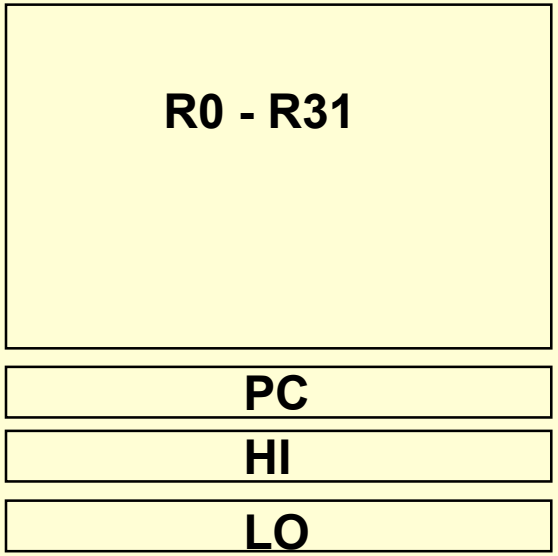


Figure: David Patterson, UCB

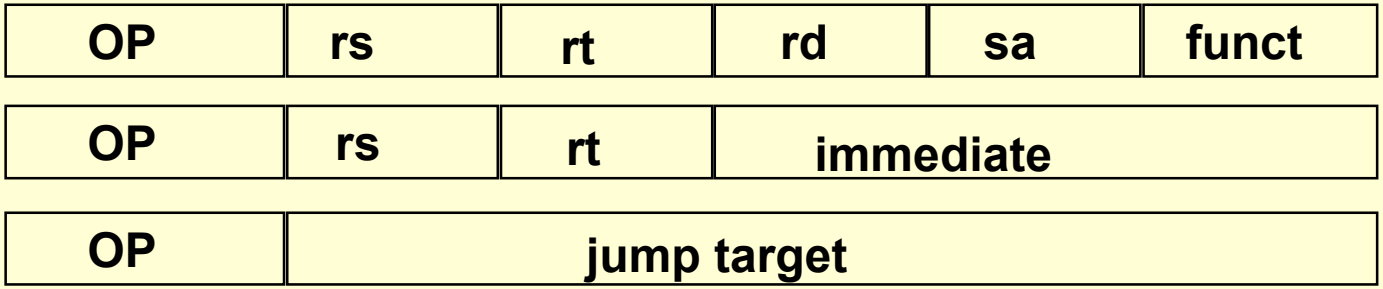
MIPS R3000 ISA (Summary)

- Instruction Categories
  - Load/Store
  - Computational
  - Jump and Branch
  - Floating Point
    - coprocessor
  - Memory Management
  - Special

Registers



3 Instruction Formats: all 32 bits wide



Slide: David Patterson, UCB

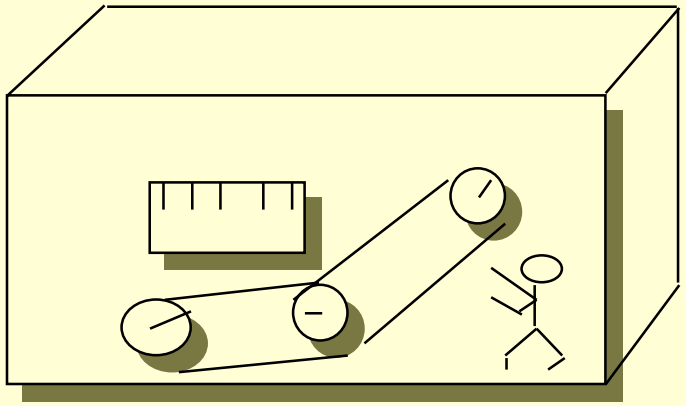
Machine Organization

- Capabilities & performance characteristics of principal functional units (e.g., Registers, ALU, Shifters, Logic Units, ...)
- Ways in which these components are interconnected
- Information flows between components
- Logic and means by which such information flow is controlled
- Choreography of functional units to realize the instruction set architecture
- Register Transfer Level Description

Logic Designer's View

ISA Level

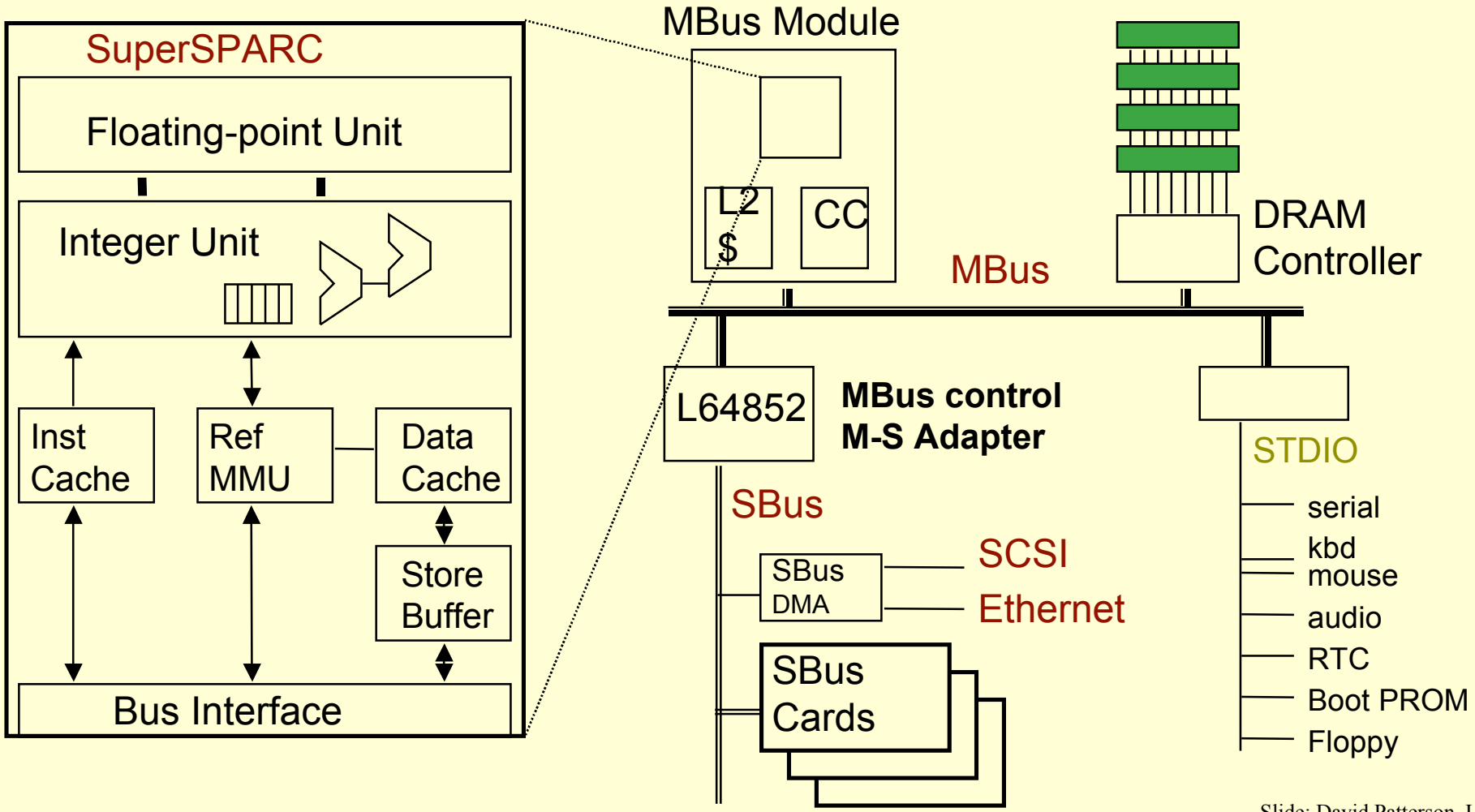
Functional Units & Interconnect



Slide: David Patterson, UCB

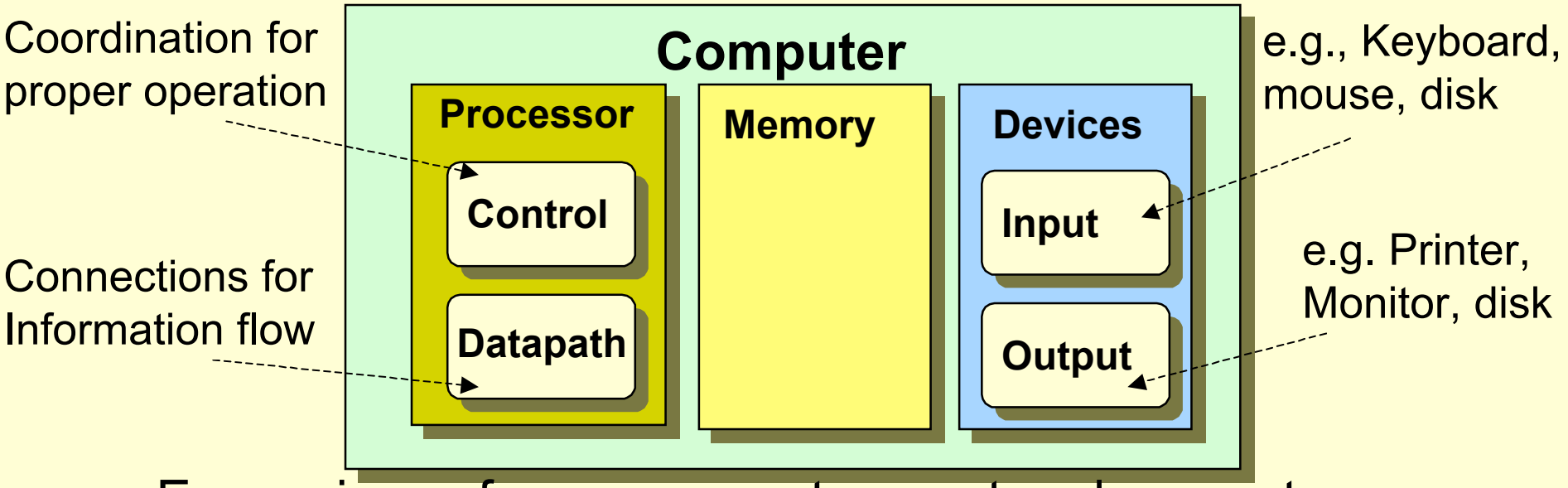
Example Organization

- TI SuperSPARCtm TMS390Z50 in Sun SPARCstation20



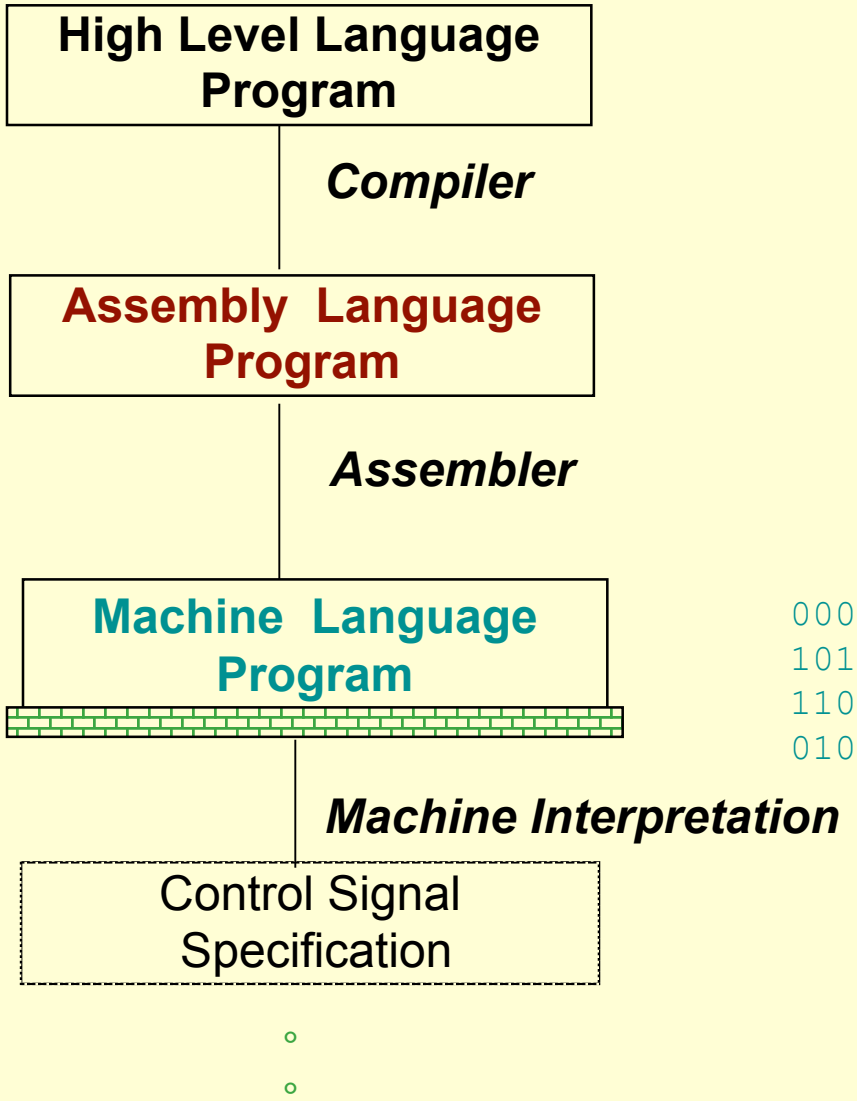
Slide: David Patterson, UCB

General Comp Organization



- Every piece of every computer, past and present: input, output, memory, datapath and control
- The design approach is constrained by the cost and size and capabilities required from every component
- An example design target can be 25% of cost on Processor, 25% of cost on minimum memory size, rest on I/O devices, power supplies, and chassis

Levels of Behavior Representation




```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
lw $15, 0($2)
lw $16, 4($2)
sw $16, 0($2)
sw $15, 4($2)
```

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

ALUOP[0:3] <= InstReg[9:11] & MASK

Slide: David Patterson, UCB

Levels of Abstraction

- S/W and H/W consists of hierarchical layers of abstraction, each hides details of lower layers from the above layer
- The instruction set arch. abstracts the H/W and S/W interface and allows many implementation of varying cost and performance to run the same S/W

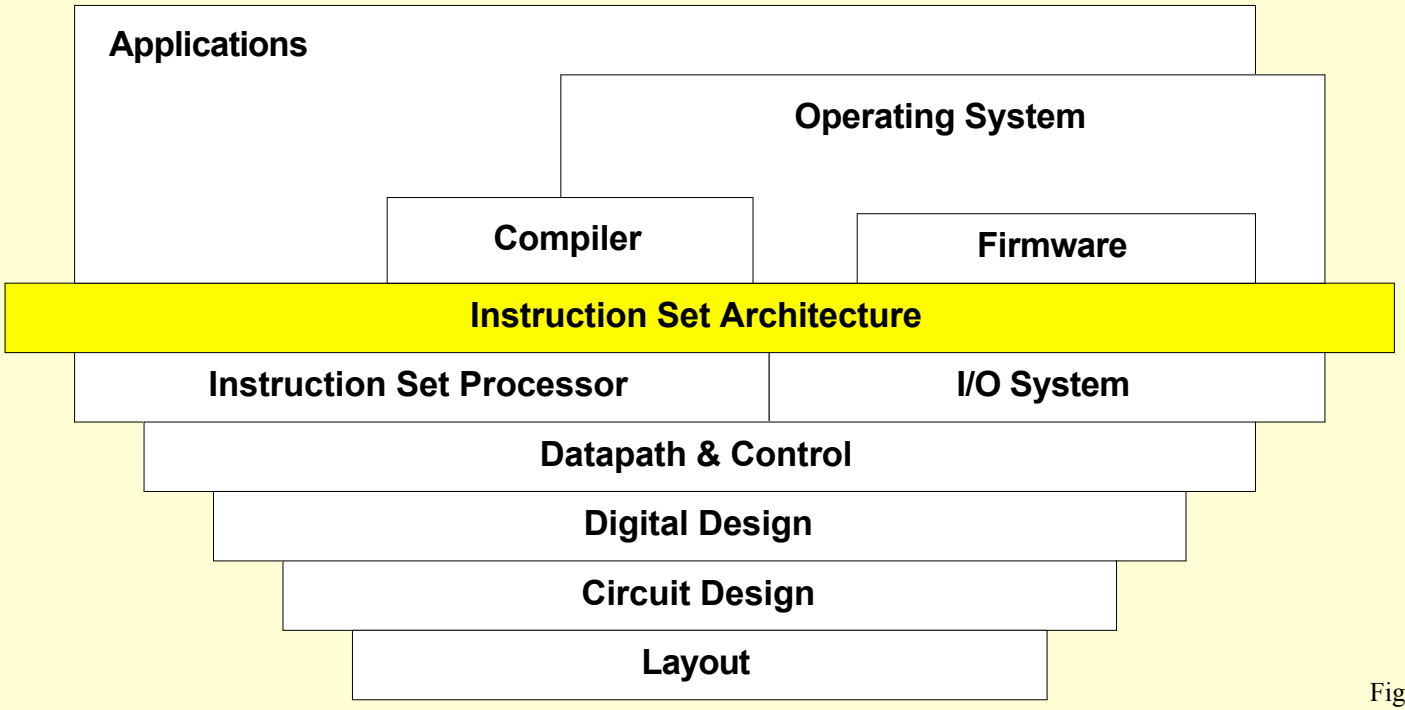


Figure: David Patterson, UCB

Forces on Computer Architecture

- Programming languages might encourage architecture features to improve performance and code size, e.g. Fortran and Java
- Operating systems rely on the hardware to support essential features such as semaphores and memory management
- Technology always raises the bar for what could be done and changes design's focus
- Applications usually derive capabilities and constrains
- History provides the starting point, filters out mistakes

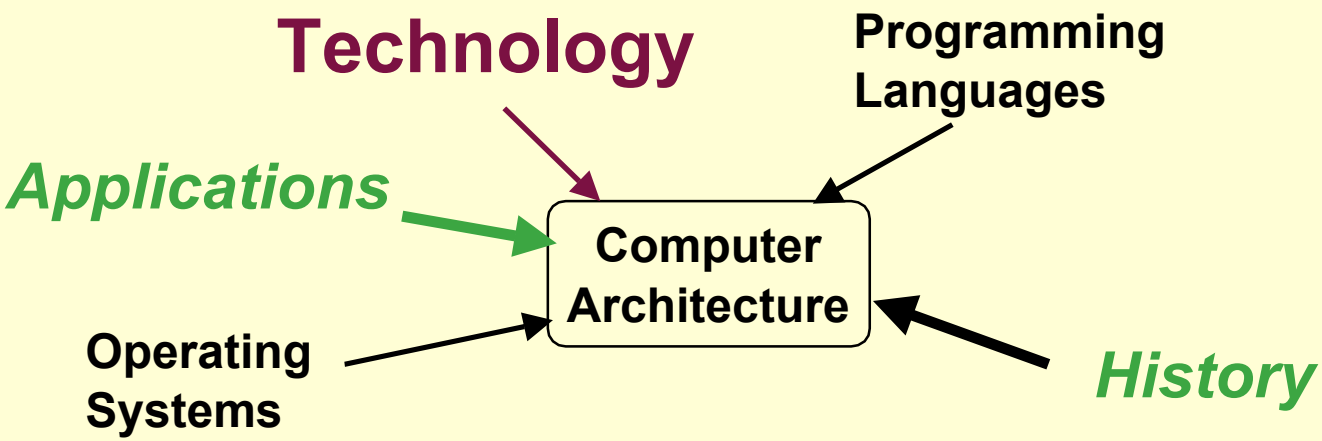


Figure: David Patterson, UCB

Technology – dramatic change

- Processor
  - logic capacity: about 30% increase per year
  - clock rate: about 20% increase per year

Higher logic density gave room for instruction pipeline & cache

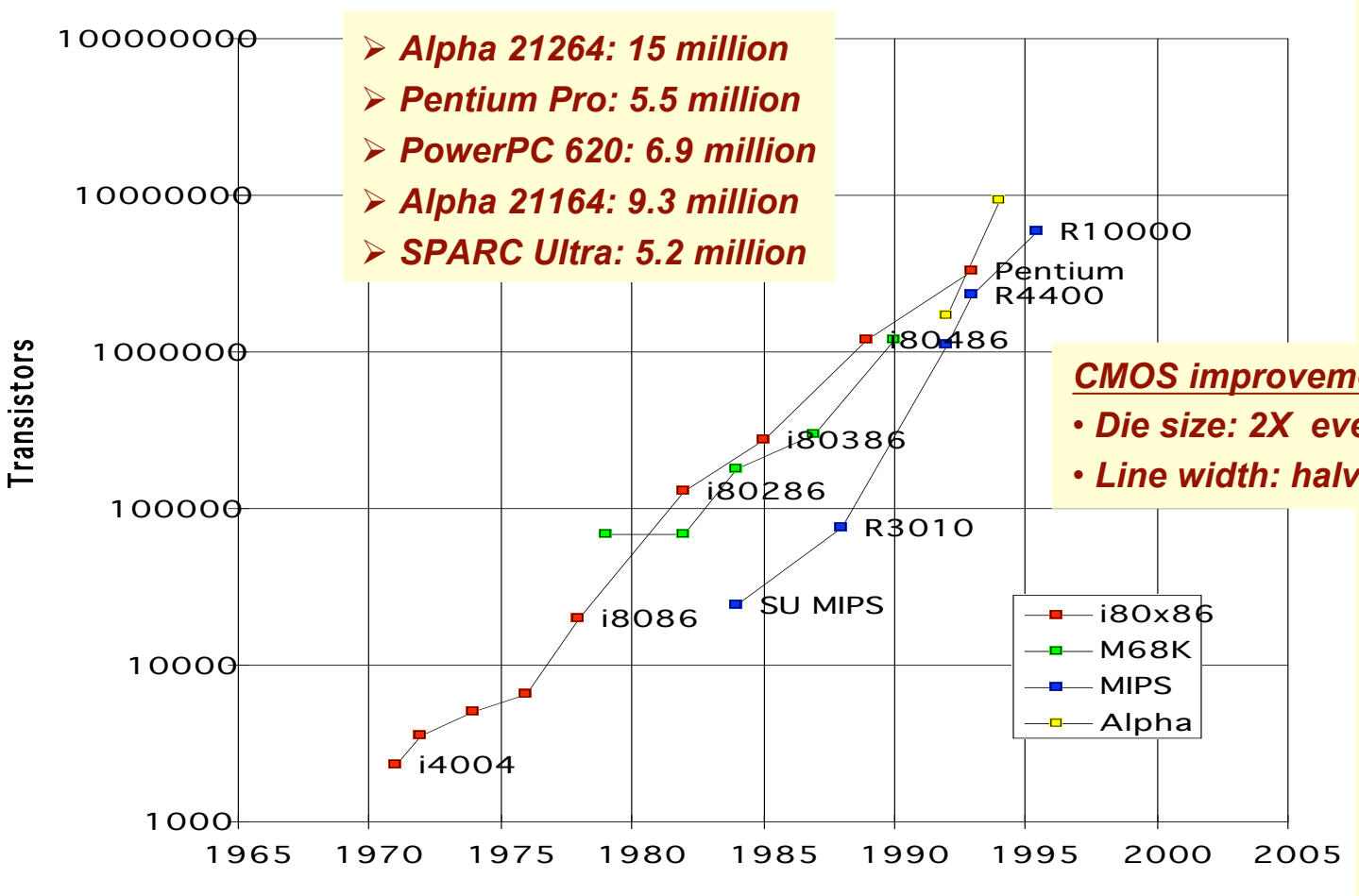
- Memory
  - DRAM capacity: about 60% increase per year (4x / 3 years)
  - Memory speed: about 10% increase per year
  - Cost per bit: about 25% improvement per year

Performance optimization no longer implies smaller programs

- Disk
  - Capacity: about 60% increase per year

Computers became lighter and more power efficient

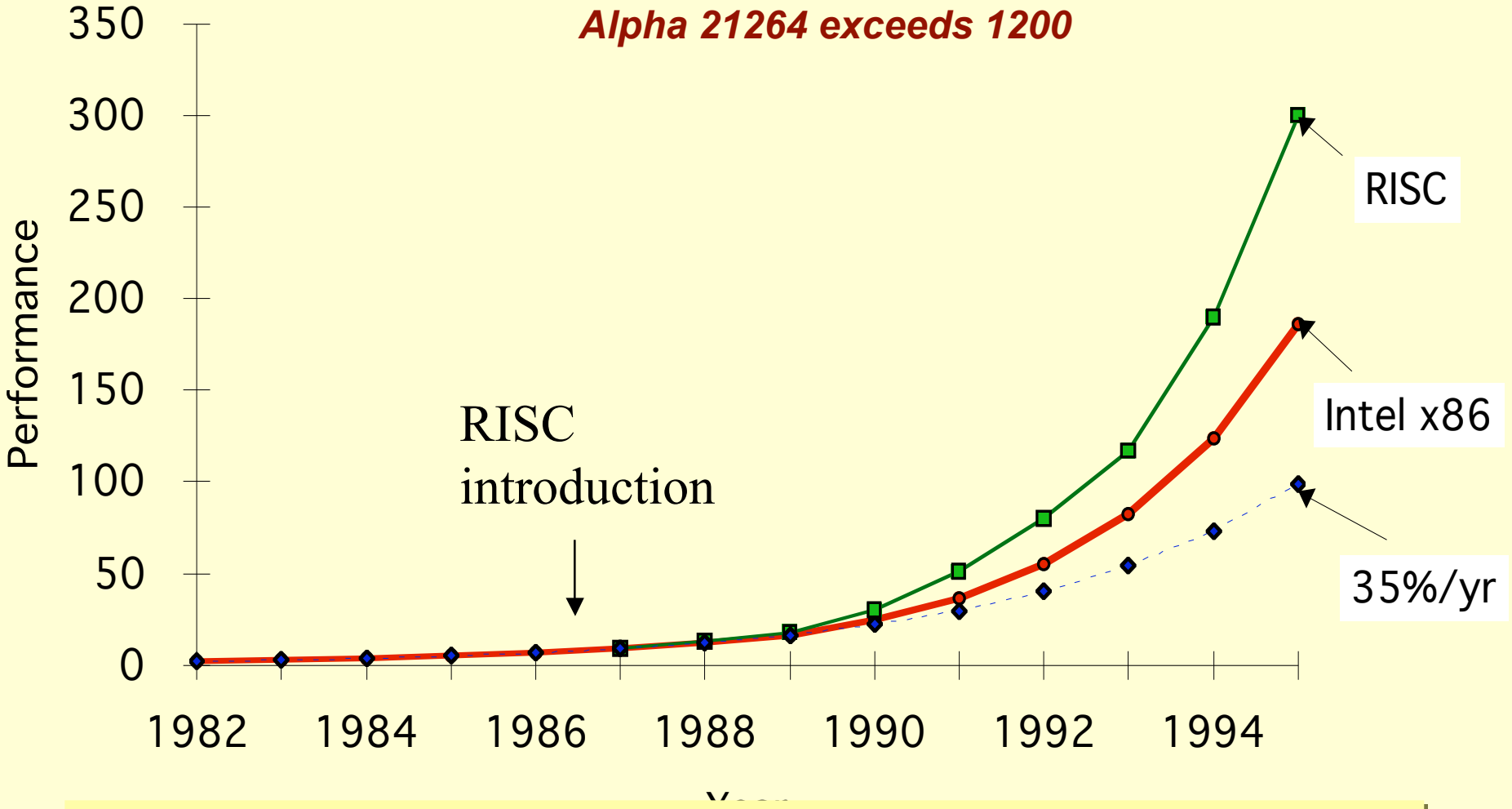
Technology Impact



In ~1985 the single-chip processor and the single-board computer emerged
In the 2004+ timeframe, today's mainframes may be a single-chip computer

Figure: David Patterson, UCB

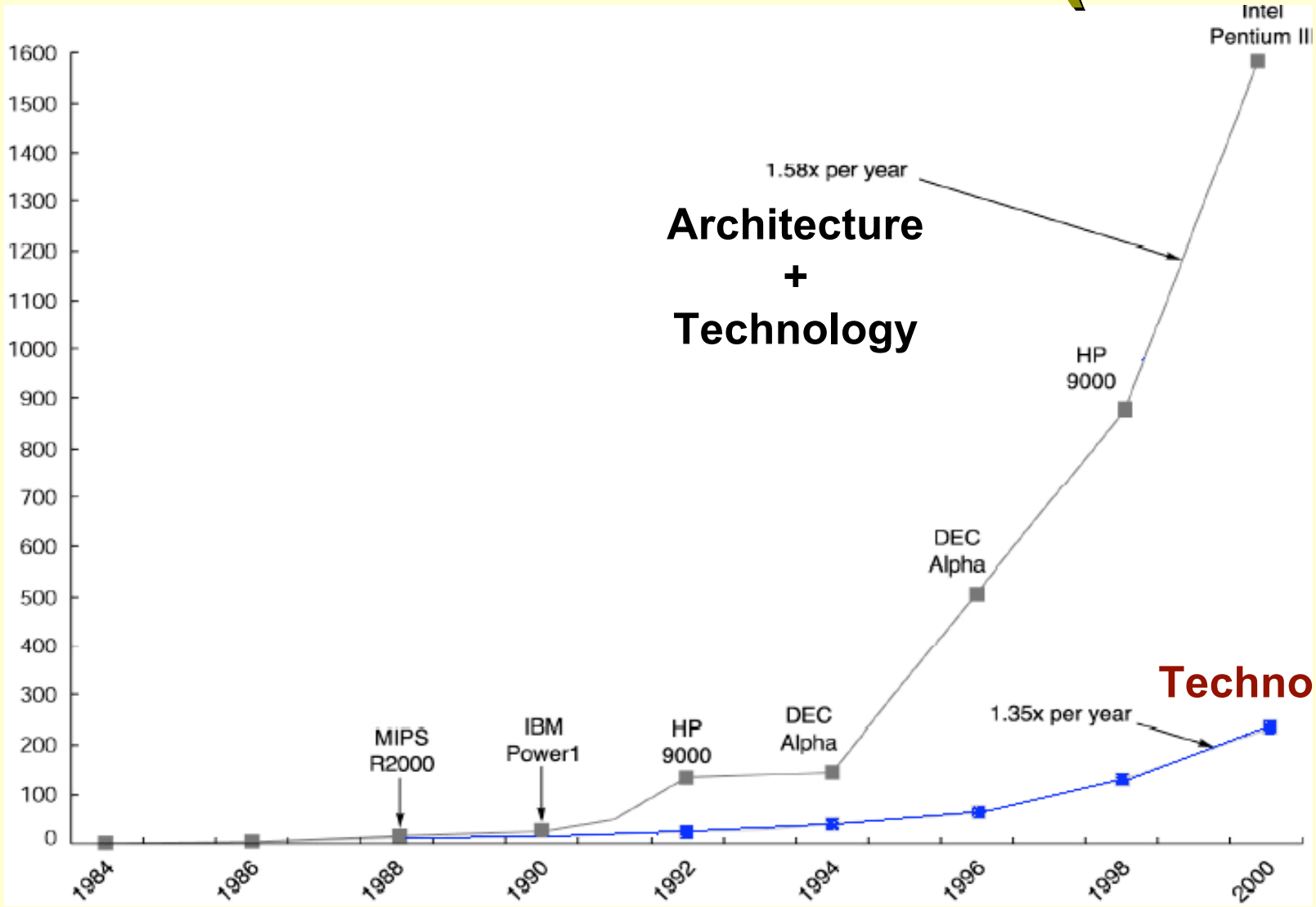
Processor Performance (SPEC)



Performance now improves ~ 50% per year (2x every 1.5 years)

Slide: David Patterson, UCB

Processor Performance (SPEC)



Relying on technology alone would have kept us 8 years behind

One Architectural Factor

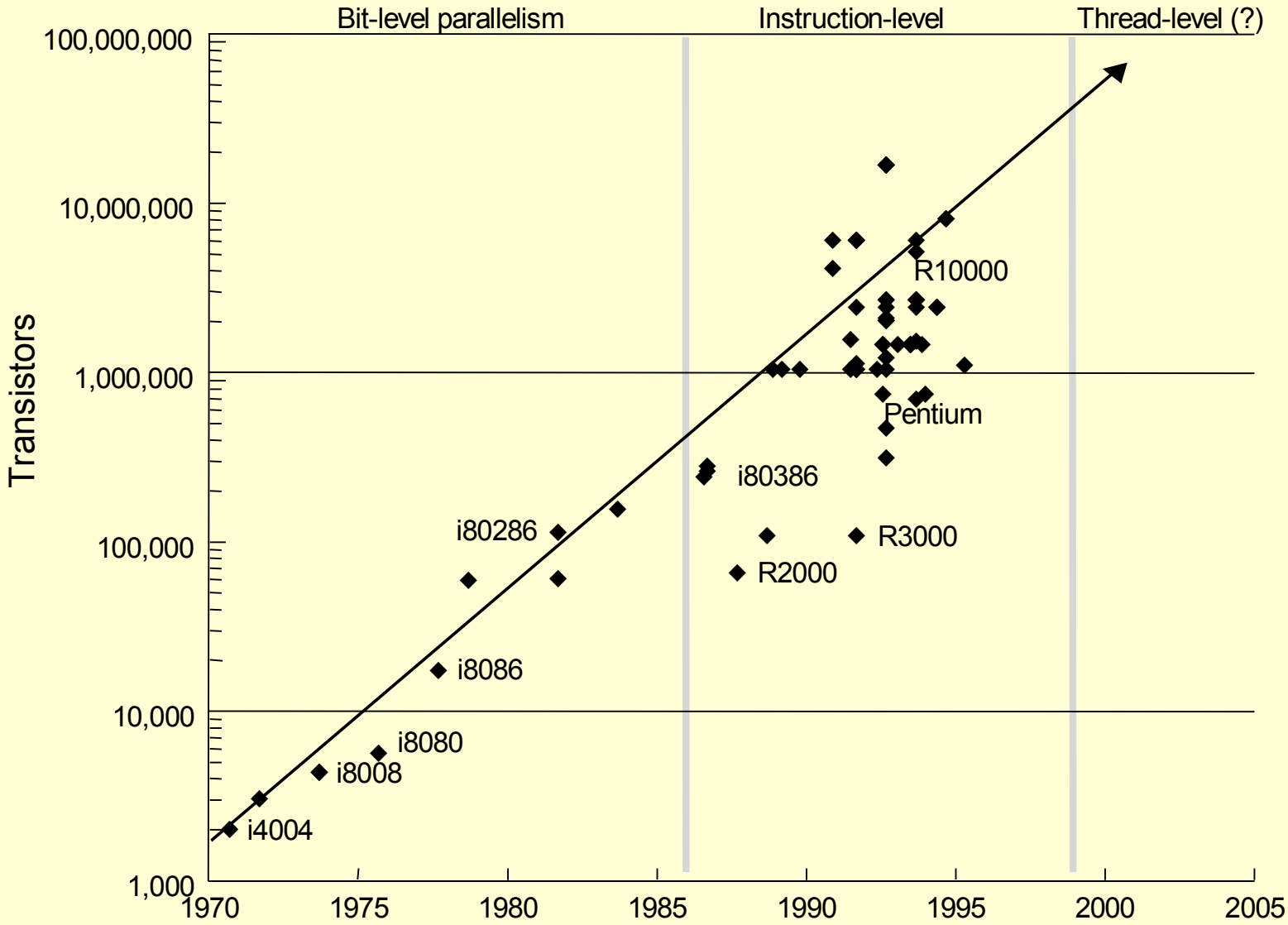
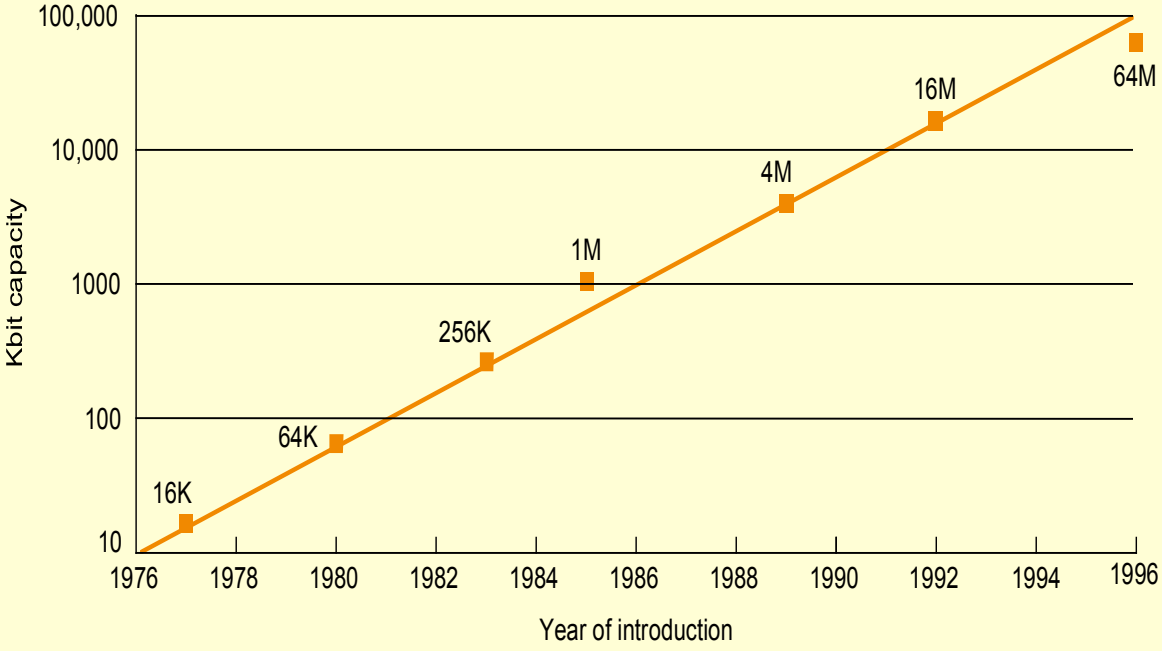


Figure: David Culler, UCB

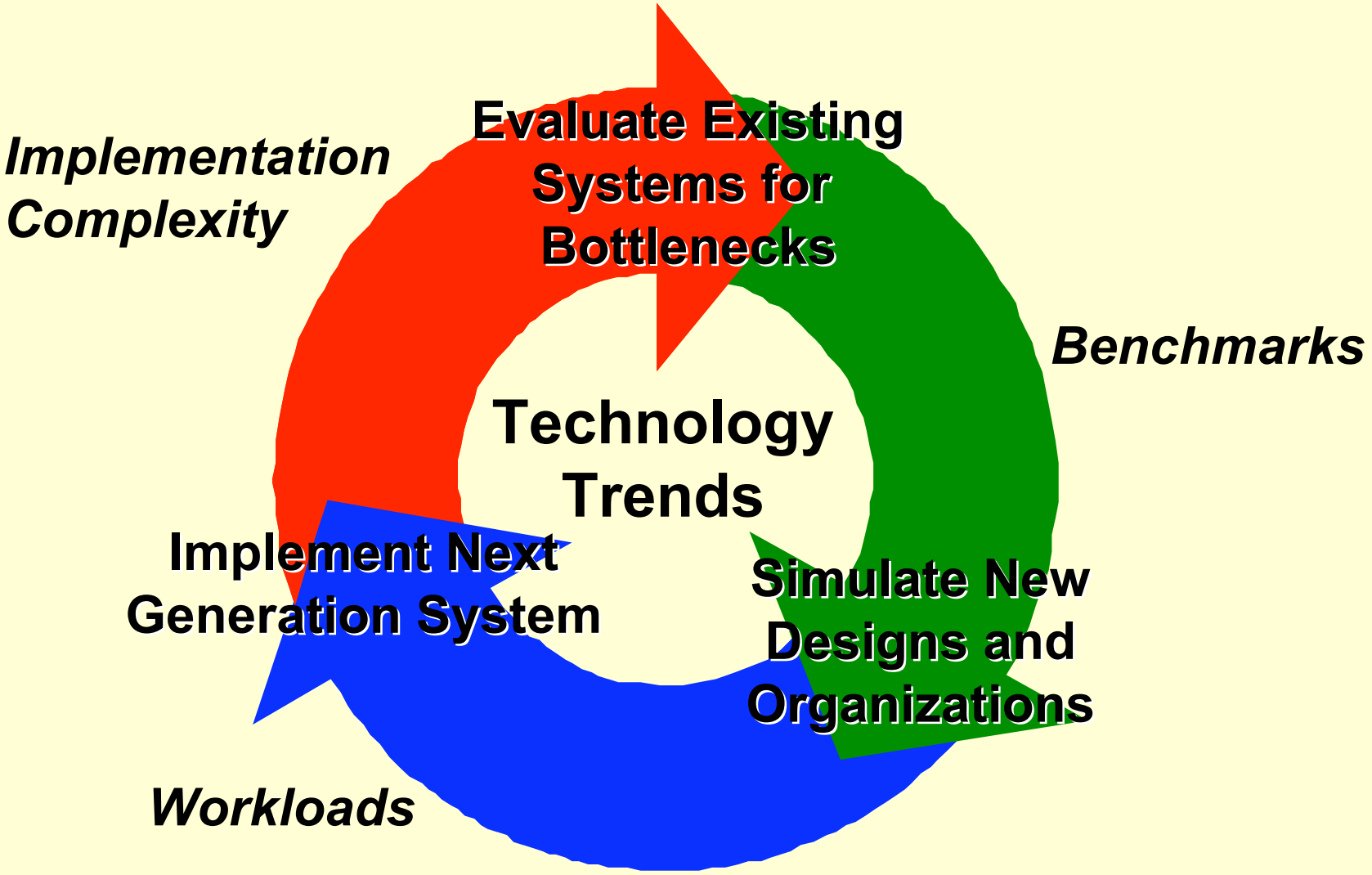
Technology Impact on Design

- DRAM capacity 4x / 3 yrs; 16,000x in 20 yrs!
- Programming concern: cache not RAM size
- Processor organization becoming main focus for performance optimization
- HW designer focus not only performance but functional integration and power consumption (e.g. system on a chip)



Year	Size(Mb)	Cyc time
1980	0.0625	250 ns
1983	0.25	220 ns
1986	1	190 ns
1989	4	165 ns
1992	16	145 ns
1996	64	120 ns
2000	256	100 ns

Computer Engineering Methodology



Slide: David Patterson, UCB

Computer Generations

- Computers were classified into 4 generations based on revolutions in the technology used in the development
- By convention, commercial electronic computers are take to be the first generation rather than the electromechanical machines that preceded them
- Today computer generations are not commonly referred to due to the long standing of the VLSI technology and the lack of revolutionary technology in sight

Gen	Dates	Technology	Principal new product
1	1950-1959	Vacuum tube	Commercial electronic computer
2	1960-1968	Transistor	Cheaper computers
3	1969-1977	Integrated circuits	Minicomputer
4	1978- ?	LSI and VLSI	Personal computers and workstations

Historical Perspective

Year	Name	Size ($Ft.^3$)	Power (Watt)	Perform. (add/sec)	Mem. (KB)	Price	Price/ Perform. vs. UNIVAC	Adjusted price 1996	Adjusted price/perform vs. UNIVAC
1951	UNIVAC 1	1000	124K	1.9K	48	$1M	1	$5M	1
1964	IBM S/360 model 50	60	10K	500K	64	$1M	263	$4.1M	318
1965	PDP-8	8	500	330K	4	$16K	10,855	$66K	13,135
1976	Cray-1	58	60K	166M	32,768	$4M	21,842	$8.5M	15,604
1981	IBM PC	1	150	240K	256	$3K	42,105	$4K	154,673
1991	HP 9000/ model 750	2	500	50M	16,384	$7.4K	3,556,188	$8K	16,122,356
1996	Intel PPro PC 200 Mhz	2	500	400M	16,384	$4.4K	47,846,890	$4.4K	239,078,908

After adjusting for inflation, price/performance has improved by about 240 million in 45 years (about 54% per year)

Conclusion

- So what's in it for you?
  - In-depth understanding of the inner-workings of modern computers, their evolution, and trade-offs present at the hardware/software boundary.
  - Experience with the design process in the context of a reasonable size hardware design
- Why should a programmer care?
  - In the 60's and 70's performance was constrained by the size of memory, not an issue today
  - Performance optimization needs knowledge of memory hierarchy, instruction pipeline, parallel processing, etc.
  - Systems' programming is highly coupled with the computer organization, e.g. embedded systems

Computer architecture is at the core of computer science & Eng.