CMSC 611

Introduction

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 2:00 – 4:00
- TA: Neelofer Tamboli / ITE 334
  - Office Hours: Wed 2:00 – 4:00
- Web Page:
  - **`www.umbc.edu/~olano/611`**
- Book
  - Hennessy and Patterson, *Computer Architecture: A Quantitative Approach*, 3rd or 4th 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, Wednesday October 22nd
  - Final December 12th, 1:00 – 3:00
- Project

Project

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

Grades

- Breakdown
  - 30% Homework
  - 25% Midterm
  - 25% Final
  - 20% Project
- Homework late policy
  - Up to 1-week late, -20% of total points
  - >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

- Multi-core
  - IBM/Sony Cell
  - Intel Core 2, Nehalem
  - Intel Larrabee
- GPUs
- Virtualization
  - vmware: emulate full virtual machine
  - JIT: compile to abstract virtual machine, dynamically compile to host

Slide: David Culler, UCB

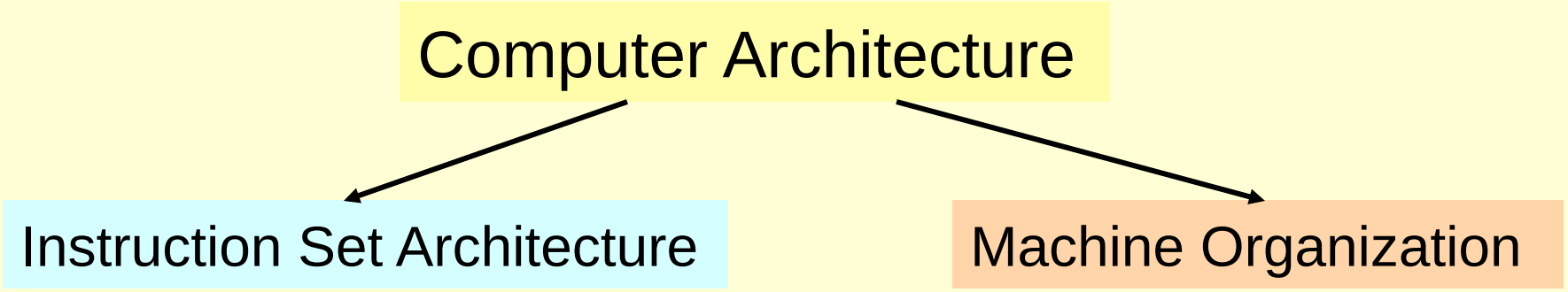
More Recent Developments

- 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-

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

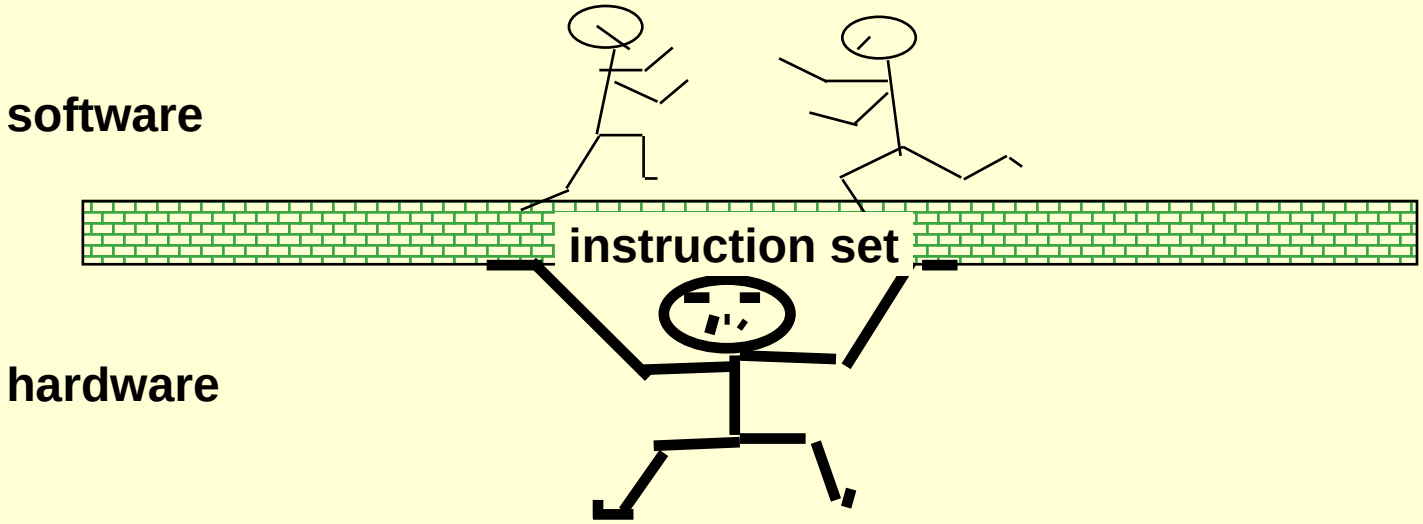
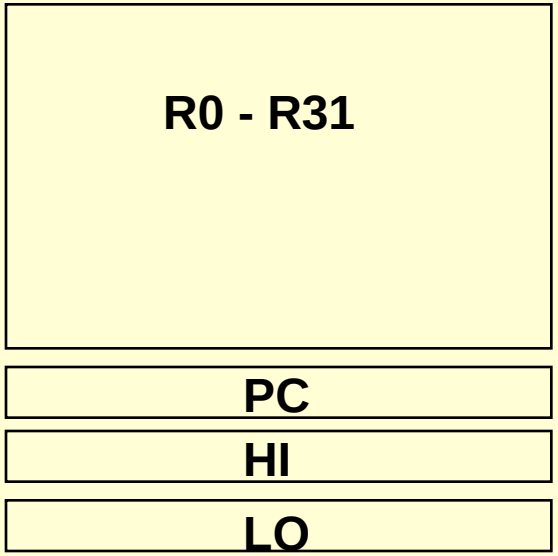


Figure: David Patterson, UCB

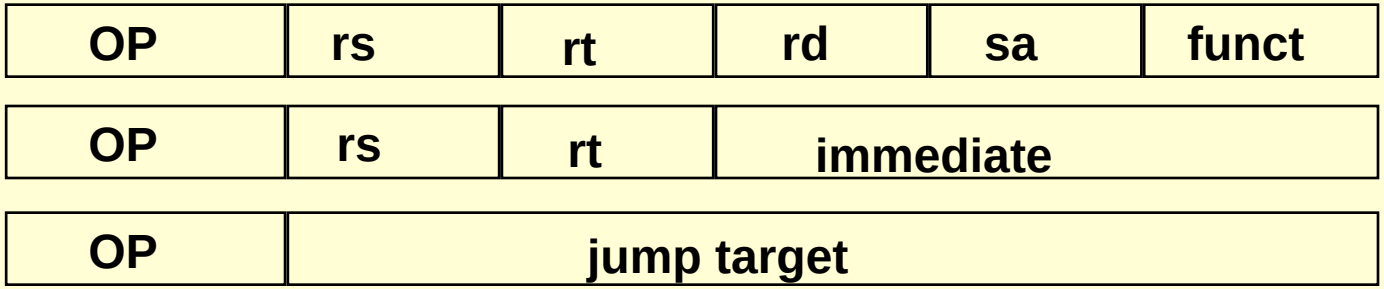
MIPS R3000 ISA (Summary)

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

Registers



3 Instruction Formats: all 32 bits wide

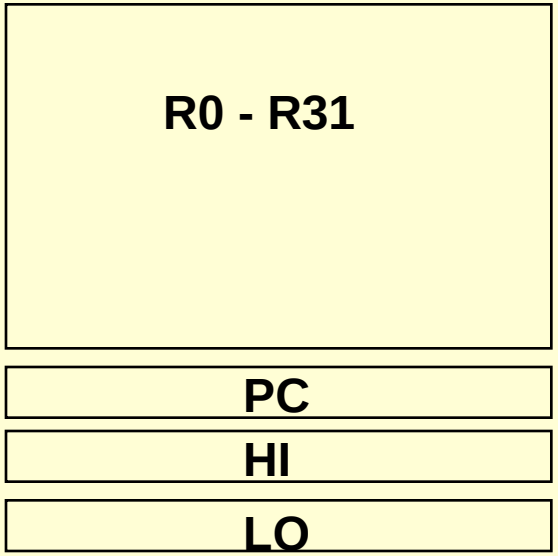


Slide: David Patterson, UCB

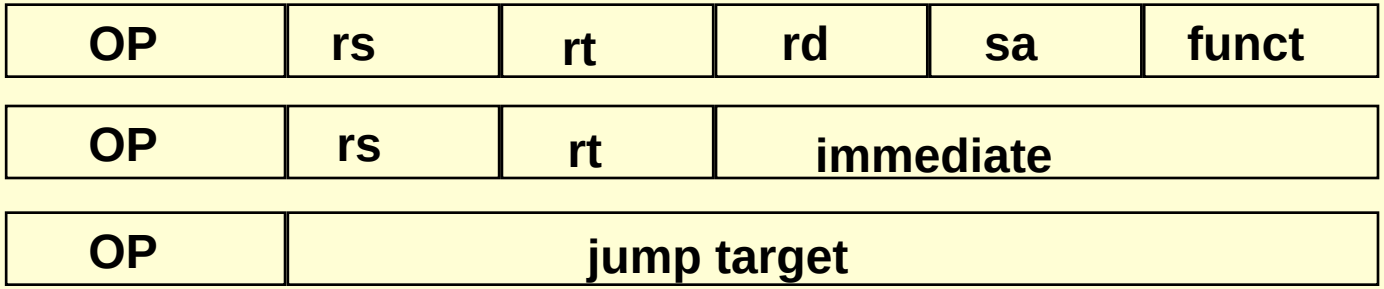
MIPS R3000 ISA (Summary)

- 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

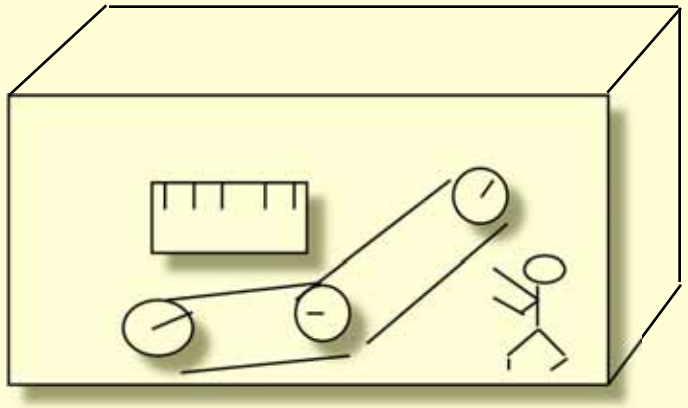
Machine Organization

- 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

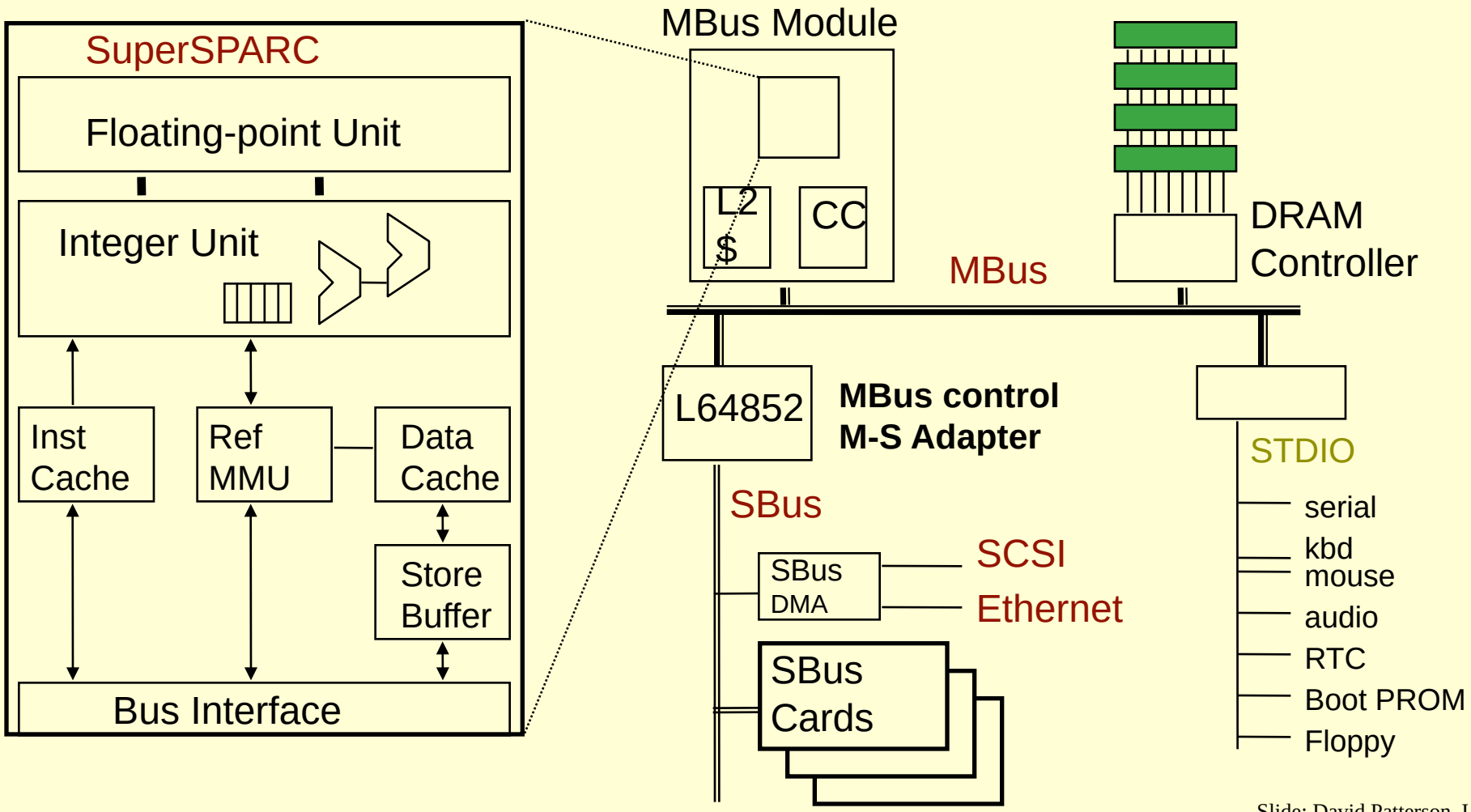
Functional Units & Interconnect



Slide: David Patterson, UCB

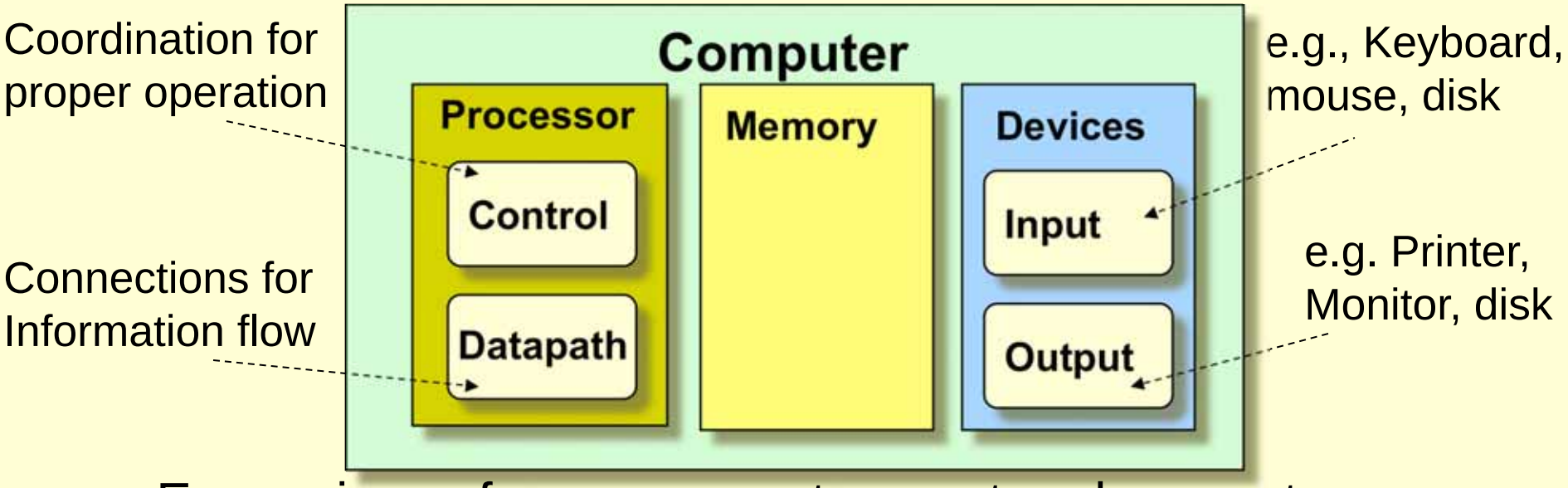
Example Organization

- 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