CS152
Computer Architecture and Engineering
Lecture 2

Review of MIPS ISA and Performance

January 25, 1999

John Kubiatowicz (http.cs.berkeley.edu/~kubitron)

lecture slides: http://www-inst.eecs.berkeley.edu/~cs152/

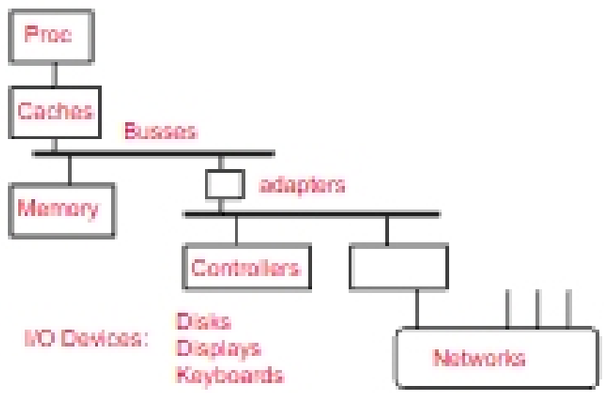
Overview of Today's Lecture: Review Instruction Sets, Performance

- Review from Last Lecture (1 min)
- Classes, Addressing, Format (20 min)
- Administrative Matters (3 min)
- Operations, Branching, Calling conventions (25 min)
- Break (5 min)
- MIPS Details, Performance (25 min)

Review: Organization

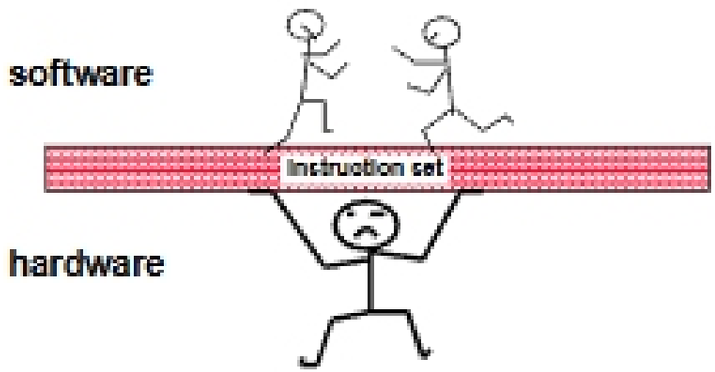
- All computers consist of five components
  - Processor: (1) datapath and (2) control
  - (3) Memory
  - I/O: (4) Input devices and (5) Output devices
- Not all "memory" is created equally
  - Cache: fast (expensive) memory are placed closer to the processor
  - Main memory: less expensive memory--we can have more
- Input and output (I/O) devices have the messiest organization
  - Wide range of speed: graphics vs. keyboard
  - Wide range of requirements: speed, standard, cost ...
  - Least amount of research (so far)

Summary: Computer System Components

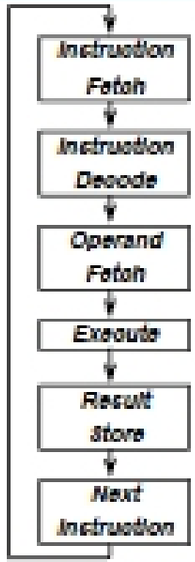


- All have interfaces & organizations

Review: Instruction Set Design



Instruction Set Architecture: What Must be Specified?



- Instruction Format or Encoding
  - how is it decoded?
- Location of operands and result
  - where other than memory?
  - how many explicit operands?
  - how are memory operands located?
  - which can or cannot be in memory?
- Data type and Size
- Operations
  - what are supported
- Successor Instruction
  - jumps, conditions, branches
  - *fetch-decode-execute is implicit!*

Basic ISA Classes

Accumulator (1 register):

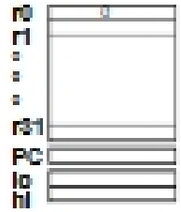
1 address

General Purpose Registers Dominate

- 1975-1988: all machines use general purpose registers
- Advantages of registers
  - registers are faster than memory
  - registers are easier for a compiler to use
    - e.g., (A*B) – (C*D) – (E*F) can do multiplies in any order vs. stack
  - registers can hold variables
    - memory traffic is reduced, so program is sped up (since registers are faster than memory)
    - code density improves (since register named with fewer bits than memory location)

MIPS I Registers

- Programmable storage
  - 2^32 x bytes of memory
  - 31 x 32-bit GPRs (R0 = 0)
  - 32 x 32-bit FP regs (paired DP)
  - HI, LO, PC

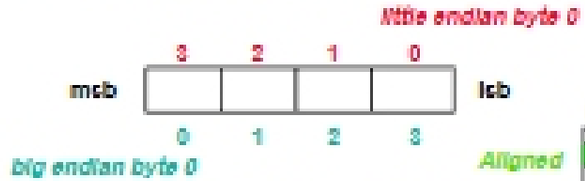


Memory Addressing

- Since 1980 almost every machine uses addresses to level of 8-bits (byte)
- 2 questions for design of ISA:
  - Since could read a 32-bit word as four loads of bytes from sequential byte addresses or as one load word from a single byte address,
    How do byte addresses map onto words?
  - Can a word be placed on any byte boundary?

Addressing Objects: Endianess and Alignment

- Big Endian: address of most significant byte = word address (xx00 = Big End of word)
  - IBM 360/370, Motorola 68k, MIPS, Sparc, HP PA
- Little Endian: address of least significant byte = word address (xx00 = Little End of word)
  - Intel 80x86, DEC Vax, DEC Alpha (Windows NT)



Alignment: require that objects fall on address that is multiple of their size.