

# The MIPS computer architecture

## Lecture B08



Lecture notes section B08

2002-01-14

CSEI 303 Part B lecture notes

1

## Last time

- Interpreters
- Machine language
- Assembly language
  - structure
- Compilers
  - what a compiler does
- Stages of code generation
  - compiling, assembling, linking
  - compiling multi-file programs

2002-01-14

CSEI 303 Part B lecture notes

2

## In this lecture

- MIPS R2000 architecture
  - memory organization
  - registers
- Running programs
  - fetch-execute cycle
  - SPIM simulator

2002-01-14

CSEI 303 Part B lecture notes

3

## MIPS computer

- Name
  - “Microcomputer without Interlocking Pipeline Stages”
  - also a pun on “Millions of Instructions Per Second”
- Developed in 1985 by MIPS Technologies
  - now a subsidiary of Silicon Graphics, Inc.
- R2000 model
  - first and simplest of MIPS processors
  - later MIPS models extend basic architecture

2002-01-14

CSEI 303 Part B lecture notes

4

## MIPS computer

- Why MIPS?
  - a real processor
  - used in many computers
    - Silicon Graphics Indy
  - also used in many embedded systems
    - Sony Aibo
    - Sony Playstation
    - Nintendo 64

2002-01-14

CSEI 303 Part B lecture notes

5

## MIPS computer

- Why MIPS?
  - MIPS architecture is ancestor of many modern computers
    - IBM/Motorola PowerPC (Macintosh)
    - Digital Alpha (Alpha)
    - ARM (3Com Palm)
  - knowledge of MIPS can be easily carried over to these other architectures

2002-01-14

CSEI 303 Part B lecture notes

6

## MIPS computer

- Why not Intel 80x86?
  - MIPS is a simple, clean architecture
    - easier to learn
    - x86 architecture is cumbersome with many confusing addressing modes and exceptions
  - MIPS is representative of modern computer architecture
    - more useful to learn
    - x86 is complicated by backward compatibility with Intel 8086 (made in 1980)
  - MIPS has a good simulator (SPIM)
    - makes it easier to write/test MIPS programs

2002-01-14

CSE1303 Part B lecture notes

7

## MIPS computer

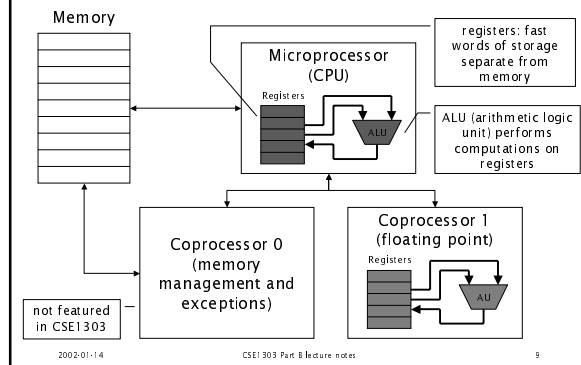
- MIPS was first computer to use term RISC
  - Reduced Instruction Set Computer
  - all instructions are
    - same length (4 bytes)
    - of similar complexity (simple)
    - (mostly) able to run in same time (1 clock cycle)
    - easily decoded and executed by computer hardware
- Intel x86 is considered CISC
  - Complex Instruction Set Computer
  - instructions vary in length, complexity and execution time
  - decoding and running instructions requires hardware-embedded program (microcode)

2002-01-14

CSE1303 Part B lecture notes

8

## MIPS architecture



2002-01-14

CSE1303 Part B lecture notes

9

## MIPS architecture

- Components of the MIPS processor
  - 32 general-purpose registers
    - each 32 bits in size
  - several special-purpose registers
    - PC (program counter)
    - HI, LO (multiplication/division results)
    - others available only to hardware
  - Arithmetic Logic Unit (ALU)
    - performs computations with registers

2002-01-14

CSE1303 Part B lecture notes

10

## MIPS architecture: registers

- General-purpose registers (GPRs)
  - can theoretically be used in any way by program
  - conventions assign certain uses to certain GPRs
    - adhering to conventions means your program can cooperate with others

2002-01-14

CSE1303 Part B lecture notes

11

## MIPS architecture: registers

- General-purpose registers (GPRs)
  - prefixed with \$ in assembly language
  - numbered \$0 to \$31
  - also given names based on register usage conventions
    - \$0 ⇔ \$zero
    - \$4 ⇔ \$a0
    - \$8 ⇔ \$t0
    - \$29 ⇔ \$sp
  - names usually used in assembly language programs to assist readability

2002-01-14

CSE1303 Part B lecture notes

12

## MIPS architecture: registers

used in CSE1303	Register name	Register number	Typical use
✓	\$zero	\$0	constant zero, cannot change
×	\$at	\$1	temporary storage for assembler
✓	\$v0, \$v1	\$2, \$3	function return values; system call number
✓	\$a0 - \$a3	\$4 - \$7	function and system call arguments
✓	\$t0 - \$t7, \$t8, \$t9	\$8 - \$15, \$24, \$25	temporary storage (caller-saved)
✓	\$s0 - \$s7	\$16 - \$23	temporary storage (callee-saved)
×	\$k0, \$k1	\$26, \$27	reserved for kernel trap handler
×	\$gp	\$28	pointer to global area
✓	\$sp	\$29	top-of-stack pointer
✓	\$fp	\$30	stack frame pointer
✓	\$ra	\$31	function return address

2002-01-14

CSE1303 Part B lecture notes

13

## MIPS architecture: registers

- Special registers LO and HI
  - special registers do not begin with \$ sign
  - contain result of previous multiplication or division instruction
  - instructions available to move contents of LO or HI back to a GPR

2002-01-14

CSE1303 Part B lecture notes

14

## MIPS architecture: registers

- Special register PC
  - contains address of instruction currently being executed
  - to change instruction being executed, change value in PC
    - takes effect after current instruction is completed
    - happens automatically for most instructions
    - branch/jump instructions can change PC explicitly

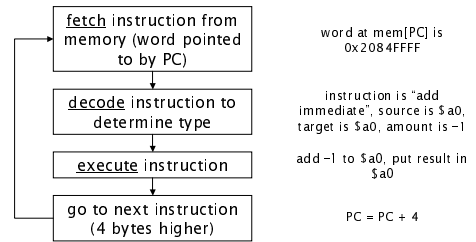
2002-01-14

CSE1303 Part B lecture notes

15

## MIPS execution

- Programs are run by MIPS hardware performing fetch-execute cycle



2002-01-14

CSE1303 Part B lecture notes

16

## MIPS architecture: memory

- Memory
  - addresses are 32 bits wide
  - addresses from 0x00000000 to 0xFFFFFFFF
    - about 4 billion bytes of addressable space
  - programs can access memory with load/store instructions
    - load (copy) a value from memory to GPR
    - store (copy) a value from GPR to memory
    - 1, 2 or 4 bytes at once
  - arithmetic/logic instructions cannot access memory
    - to perform arithmetic on memory contents, must copy via GPR

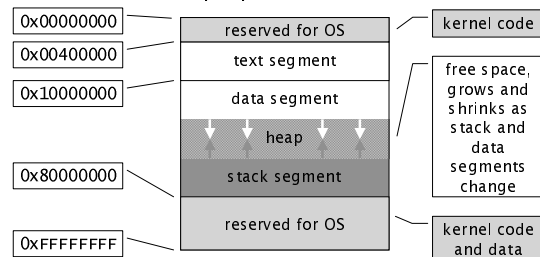
2002-01-14

CSE1303 Part B lecture notes

17

## MIPS architecture: memory

- Memory is organized into segments, each with its own purpose



2002-01-14

CSE1303 Part B lecture notes

18

## MIPS architecture: memory

- Text segment
  - starts at memory address 0x00400000
    - up to address 0x0FFFFFFF
  - contains user's executable program (code)

2002-01-14

CSEI 303 Part B lecture notes

19

## MIPS architecture: memory

- Data segment
  - starts at memory address 0x10000000
    - up towards stack
  - contains program's static data
    - global variables
    - string constants

2002-01-14

CSEI 303 Part B lecture notes

20

## MIPS architecture: memory

- Stack segment
  - starts at memory address 0x7FFFFFFF
    - grows down in memory towards data segment
  - contains system stack
  - used for temporary storage
    - function local variables
    - function arguments
    - function return address
    - saved registers

2002-01-14

CSEI 303 Part B lecture notes

21

## MIPS architecture: memory

- System heap
  - exists between data segment and stack segment
  - empty at start of program execution
  - memory allocated with malloc is taken from heap for program to use
    - freed memory is returned to heap

2002-01-14

CSEI 303 Part B lecture notes

22

## SPIM simulator

- A freely-distributable simulator of MIPS R2000 architecture
  - runs on many computers
    - Unix
    - MS Windows
    - Macintosh OS X
  - easy-to-use graphical user interface

2002-01-14

CSEI 303 Part B lecture notes

23

## SPIM simulator

The screenshot shows the SPIM simulator window with several panes. At the top, there are registers (PC, PC14, R0-R31) and their values. Below that are control buttons like 'quit', 'load', 'run', 'stop', 'clear', 'print', 'breakpoint', 'help', 'kernel', and 'more'. The 'Text Segments' pane shows memory addresses and hex values. The 'Data Segments' pane shows memory addresses and values. The 'Messages' pane at the bottom shows the simulator's output.

register values

commands

text segment (user code)

data and stack segments

messages

2002-01-14

CSEI 303 Part B lecture notes

24

## SPIM simulator: registers

PC      HI      LO

general-purpose registers

2002-01-14      CSE1303 Part B lecture notes      25

## SPIM simulator: code

address of instruction      bit pattern of instruction

meaning of instruction's bit pattern      source code of program

2002-01-14      CSE1303 Part B lecture notes      26

## SPIM simulator: data

name of segment

address of first byte of row      word-by-word contents of memory (16 bytes per row)

2002-01-14      CSE1303 Part B lecture notes      27

## Covered in this lecture

- MIPS R2000 architecture
  - memory organization
  - registers
- Running programs
  - fetch-execute cycle
  - SPIM simulator

2002-01-14      CSE1303 Part B lecture notes      28

## Going further

- Dynamic memory allocation
  - how to write the malloc function
- Operating system kernel programming
  - using the "reserved for OS" memory areas
  - introduced in CSE2324/CSE3324

2002-01-14      CSE1303 Part B lecture notes      29

## Next time

- MIPS instruction set
- Writing MIPS programs

2002-01-14      CSE1303 Part B lecture notes      30



Reading:  
Lecture notes section B09

## **Copyright**

Copyright © 2001 Deborah Pickett.  
No part of this presentation may be  
duplicated without permission from  
the author.