University of Arizona

CSc 340

Foundations of Computer Systems

Christian Collberg

January 25, 2001

MIPS Instruction Set

MIPS Instructions

The MIPS instruction set is covered in detail in the SPIM handout. In all instructions Src2 can either be a register or a 16-bit immediate value. I left out the immediate forms because the MIPS assembler takes care of converting the general forms into the immediate forms when appropriate.

Logical operations `and, or, xor, nor`.

Syntax: `op Rdest, Rsrc1, Src2`.

Unary operations `abs, neg, not`.

Syntax: `op Rdest, Rsrc`.

MIPS Instructions ...

Shift/rotate `sll, srl, sra, rol, ror`.

Syntax: `op Rdest, Rsrc1, Src2`.

Arithmetic `add, addu, sub, subu`.

Syntax: `op Rdest, Rsrc1, Src2`.

MIPS Instructions ...

Multiplication

`mult, multu`. Syntax: `op Rsrc1, Rsrc2`.

- Multiplies the contents of the two registers, putting the low-order word of the result in the register ⌜`lo`⌝, and the high-order word in ⌜`hi`⌝.

`mul, mulo, mulou`.

Syntax: `op Rdest, Rsrc1, Rsrc2`.

- Multiplies `Rsrc1` by `Rsrc2` and stores the result in `Rdest`. These are synthetic instruction.

MIPS Instructions . . .

Division

`div, divu`. Syntax: `op Rsrc1, Rsrc2`.

- Divides the contents of Rsrc1 by Rsrc2, and leaves the quotient in ⌜`lo`⌝ and the remainder in ⌜`hi`⌝.

`div, divu`. Syntax: `op Rdest, Rsrc1, Rsrc2`.

- Divides the contents of `Rsrc1` by `Rsrc2`, and leaves the quotient in `Rdest`.

Slide 6–4

Load/Store Instructions

- `lw` Syntax: `ld Rdest, address`
- address can be one of:
  1. (register)
  2. imm
  3. imm(register)
  4. symbol
  5. symbol±imm
  6. symbol±imm(register).

  We'll cover addressing modes in more detail in a later topic.

Slide 6–5

Load/Store Instructions . . .

- The `lw` instruction, and its variants, load a value from memory into register `Rdest`. The memory address can be specified in one of the ways shown above.
- A load instruction affects the entire register, even if you only load a byte into it (for example). The high-order bytes are either
  1. cleared to zero (for an unsigned load), or
  2. sign-extended (for a signed load).

Slide 6–6

Load/Store Instructions . . .

- `sw`
  Syntax: `ld Rsrc, address`
- Stores the value in register `Rsrc` into memory at the specified address.
- Note that if you store one byte, only that byte of memory is affected. Stores have different behavior from loads in that sense.

Slide 6–7

Branch and Jump Instructions

- The syntax varies for different branch instructions:
  1. Some, like ⌜`b`⌝, have a single label argument and always branches to that label.
  2. Others compare two registers and branch if the comparison is true.
  3. Some test a single register and branch if the test is true.

Misc. Instructions

- ⌜`move Rdest, Rsrc`⌝ sets Rdest to Rsrc.
- ⌜`la Rdest, address`⌝ Loads the value of address into `Rdest`, not the contents of memory at that address.
- The following instructions Moves into and out of the ⌜`hi`⌝ and ⌜`lo`⌝ registers:
  1. ⌜`mfhi Rdest`⌝
  2. ⌜`mflo Rdest`⌝
  3. ⌜`mthi Rsrc`⌝
  4. ⌜`mtlo Rsrc`⌝

Misc. Instructions ...

- ⌜`li Rdest, imm`⌝ loads the immediate value imm into `Rdest`.
- ⌜`lui Rdest, imm`⌝ loads the low-order halfword of imm into the high-order halfword of Rdest. The low-order bits of Rdest are set to zero.

Readings and References

- SPIM Manual.