CS305
Computer Architecture

Arithmetic in MIPS

Bhaskaran Raman
Room 406, KR Building
Department of CSE, IIT Bombay

http://www.cse.iitb.ac.in/~br

Recall Integer Representation

- Possibilities
  - Sign-magnitude
  - 1’s complement
  - 2’s complement
- Which representation does MIPS use? Why?

Implications of Number Representation for ISA

- Unsigned versions of **`lb, lh: lbu, lhu,`** (**`lwu`** ?)
- Many instructions have unsigned versions
  - **`add`** → **`addu, addi`** → **`addiu, sub`** → **`subu`**
  - **`slt`** → **`sltu, slti`** → **`sltiu`**
- Questions:
  - Is result if **`add`** and **`addu`** always the same?
  - So why separate **`addu`** instruction ?
  - Using single **`sltu`** or **`sltiu`** to check array bounds

Integer Multiplication and Division in MIPS

- Use of special registers **`hi, lo`**
  - **`mult, multu`** store 64-bit result in **`hi, lo`**
  - **`div, divu`** store quotient in lo, remainder in hi
- How to get result from **`hi, lo`** ?
  - Special instructions: **`mfhi <reg>, mflo <reg>`**
  - Why not instructions for moving values to **`hi, lo`** ?

Recall Floating Point Representation

- Normalized scientific notation
  - Sign, significand, exponent (biased 2's complement)
  - Single precision: 32 = 1 + 8 + 23
  - Double precision: 64 = 1 + 11 + 52
  - Subnormal numbers, NaN
  - IEEE 754 standard
    - Supported in MIPS, same as in C float/double

MIPS Floating Point Instructions

- Arithmetic: **+, -, x, /** add.s, add.d sub.s/d mul.s/d div.s/d
- Comparisons: **eq, ne, lt, le, gt, ge**
  - Set a special bit c.eq.s c.le.d
  - To be used in next instruction: **bc1t, bc1f** co-processor
- Memory operations: **lwc1, ldc1, swc1, sdc1**
- MIPS has separate 32 x 32-bit FP registers
  - Why don't we need additional bit in instruction encoding?
  - Can be considered as 16 x 64-bit FP registers for double

Common Programming Bugs

- Signed versus unsigned
- Not handling overflow
- Assuming FP associativity
- Assuming FP precision

$a \times (b \times c) \neq (a \times b) \times c$

$x == y$