MIPS Examples, Other ISAs

CPS 104

MIPS Mul/Div Instructions

- **Multiply**
  - Two 32-bit numbers multiplied requires 64 bits
  - $2^{32} \times 2^{32} = 2^{64}$
  - Lo and Hi Registers contain result of mul R1, R2
  - High order 32 bits in Hi, Low order in Lo
  - Pseudoinstruction: mul Rdest, R1, R2
  - Puts low order of result into Rdest
    - `mul R1, R2
     mflo Rdest`
- **Divide**
  - Quotient put into Lo and Remainder into Hi
  - Pseudoinstruction: div Rdest, R1, R2
  - Puts quotient into Rdest
    - `div R1, R2
     mflo Rdest`

NiosII Mul/Div Instructions

- **Multiply**
  - Two 32-bit numbers multiplied requires 64 bits
  - $2^{32} \times 2^{32} = 2^{64}$
  - `mul rC, rA, rB`
  - Only low 32 bits stored in rC
  - Need additional instructions to detect overflow
- **Divide**
  - `mul rC, rA, rB`
  - Quotient put into rC
  - Need additional instructions to obtain remainder

Assembly Programming

- Traverse array of structures
  - 2 integers: ID, score
- Loop traversal of Linked List
  - 2 integers: ID, score
  - 1 pointer: next
- Generic Recursion
- Recursive traversal of Linked List
- Others
- Let’s work some examples

PowerPC ISA

- Very similar to MIPS
- Indexed Addressing (register+register)
  - `lw $t1, $a0, $s3
     $t1 = mem[$a0+$s3]`
- Update Addressing
  - `lw $t1, 4($a0)
     $t1 = mem[$a0+4]; $a0 += 4;`
- Load/Store Multiple
- Counter Register
  - `bc loop, ctr != 0`
  - decrement ctr, if ctr != 0 go to loop

Administivia

- Homework #2 Due Wednesday
- Midterm: Monday February 12 in class
  - Covers up through Wednesday’s lecture
- After Midterm
  - Logic Design
  - Reading Appendix B.1-B.3, B.5
**Intel 80x86 ISA**

- Long history
- Binary compatibility
- 1978: 8086, 16-bit, registers have dedicated uses
- 1980: 8087, added floating point (stack)
- 1982: 80286, 24-bit
- 1985: 80386, 32-bit, new insts -> GPR almost
- 1989-95: 80486, Pentium, Pentium II
- 1997: Added MMX
- 1999: Pentium III
- 2002: Pentium 4

**80x86 Registers and Addressing Modes**

- eight 32-bit GPRs
  - EAX, ECX, EDX, EBX, ESP, EBP, ESI, EDI
- six 16-bit Registers for code, stack, & data
- 2 address ISA
  - one operand is both source and destination
- Not Load/Store
  - One operand can be in memory

**80x86 Addressing Modes**

- Register Indirect
  - mem[reg]
  - not ESP or EBP
- Base + displacement (8 or 32 bit)
  - mem[reg + const]
  - not ESP or EBP
- Base + scaled Index
  - mem[reg + (2^scale x index)]
  - scale = 0,1,2,3
  - base any GPR, Index not ESP
- Base + scaled Index + displacement
  - mem[reg + (2^scale x index) + displacement]
  - scale = 0,1,2,3
  - base any GPR, Index not ESP

**Condition Codes**

- Both PowerPC and x86 ISA have condition codes
- Special HW register, that has values set as side effect of instruction execution
- Example conditions
  - Zero
  - Negative
- Example use
  - subi $t0, $t0, 1
  - bz loop

**80x86 Instruction Encoding**

- Variable Size 1-byte to 17-bytes
- Jump (JE) 2-bytes
- Push 1-byte
- Add Immediate 5-bytes
- W bit says 32-bits or 8-bits
- D bit indicates direction
  - memory -> reg or reg -> memory
  - movw EBX, [EDI + 45]
  - movw [EDI + 45], EBX

**Summary**

- Procedure calls
- powerPC, Intel 80x86 ISA
- Next Time
- Compilers, linkers, loaders
- Reading
- Start Appendix B -- logic design