CS 354 - Machine Organization & Programming
Tuesday, November 5, 2019

Midterm Exam (~18%): Thursday, November 7th, 7:15 - 9:15 pm

- **Lec 1 (2:30 pm):** room 3650 of Humanities
- **Lec 2 (4:00 pm):** room B10 of Ingraham Hall
- ◆ UW ID required
- ◆ #2 pencils required
- ◆ closed book, no notes, no electronic devices (e.g., calculators, phones, watches)
- ◆ see "Midterm Exam 2" on course site Assignments for topics

Project p4A (~2%): DUE TODAY at 10 pm on Tuesday, November 5th

Homework hw5 (1.5%): DUE TOMORROW at 10 pm Wednesday, November 6th

Project p4b (~4%): DUE at 10 pm on Wednesday, November 13th

Last Time

Instructions - MOV, PUSH, POP
Operand Specifiers
Operands Practice
Operand/Instruction Caveats
Instruction - LEAL
Instructions - Arithmetic and Shift

Today

Instructions - Arithmetic and Shift (from last time)
Instructions - CMP and TEST, Condition Codes
Instructions - SET
Instructions - Jumps
Encoding Targets
Converting Loops
---------- END of Exam 2 Material -----------
Midterm 2 Reference Page

Next Time

Stack Frames
Read: B&O 3.7 intro - 3.7.3
Exam Mechanics

Instructions - CMP and TEST, Condition Codes

What?

-
-

Why?

How?

```
CMP S2,S1            CC <-- S1 - S2



TEST S2,S1           CC <-- S1 & S2
```

➢ What is done by `testl %eax, %eax`

Condition Codes (CC)

`total = a + b` assume variables are `int`s in 2's complement representation

ZF: zero flag
    set if `total == 0`

CF: carry flag
    set if `(unsigned) total  <  (unsigned) a`

SF: sign flag
    set if `total < 0`

OF: overflow flag

    set if `(a < 0 == b < 0)`
        `&& (total < 0 != a < 0)`

Instructions - SET

What?

How?

```
sete D          D <-- ZF
setne D         D <-- ~ZF
sets D          D <-- SF
setns D         D <-- ~SF
```

Unsigned Comparisons

```
setb D          D <-- CF
setbe D         D <-- CF | ZF
seta D          D <-- ~CF & ~ZF
setae D         D <-- ~CF
```

Signed (2's Complement) Comparisons

```
setl D          D <-- SF ^ OF
setle D         D <-- (SF ^ OF) | ZF
setg D          D <-- ~(SF ^ OF) & ~ZF
setge D         D <-- ~(SF ^ OF)
```

Example: a < b (assume int a is in %eax, int b is in %ebx)

1. `cmpl %ebx,%eax`
2. `setl %cl`
3. `movzbl %cl,%ecx`

Instructions - Jumps

What?

target:

Why?

How? Unconditional Jump

indirect jump:

```
jmp *Operand
```

direct jump:

```
jmp Label
```

How? Conditional Jumps

-
-

```
both:       je Label      jne Label     js Label      jns Label
unsigned:   jb Label      jbe Label     ja Label      jae Label
signed:     jl Label      jle Label     jg Label      jge Label
```

Encoding Targets

What?

Absolute Encoding

Problems?

- code is not
- code cannot be

Solution?

IA-32:

→ What is the distance (in hex) encoded in the `jne` instruction?

```
      Assembly Code           Address    Macnine Code
      cmpl  %eax, %ecx
      jne .L1                 0x_B8      75 ??
      movl  $11, %eax         0x_BA
      movl  $22, %edx         0x_BC
 .L1:                         0x_BE
```

→ If the `jb` instruction is 2 bytes in size and is at 0x08011357 and the target is at 0x8011340 then what is the distance (hex) encoded in the `jb` instruction?

Converting Loops

→ Which kind of C loop does each goto code fragment correspond?

```
loop1:
      loop_body
      t = loop_condition
      if (t) goto loop1:
```

```
      t = loop_condition
      if (!t) goto done:
loop2:
      loop_body
      t = loop_condition
      if (t) goto loop2
done:
```

```
      loop_init
      t = loop_condition
      if (!t) goto done:
loop3:
      loop_body
      loop_update
      t = loop_condition
      if (t) goto loop3
done:
```

✵ *Most compilers (gcc included)*

Exam 2 Reference Page

Powers of 2

$2^5 = 32$, $2^6 = 64$, $2^7 = 128$, $2^8 = 256$, $2^9 = 512$, $2^{10} = 1024$
$2^{10} = K$, $2^{20} = M$, $2^{30} = G$
$2^A \times 2^B = 2^{A+B}$
$2^A / 2^B = 2^{A-B}$

Hexadecimal Digits

$9_{16} = 9_{10} = 1001_2$
$A_{16} = 10_{10} = 1010_2$
$B_{16} = 11_{10} = 1011_2$
$C_{16} = 12_{10} = 1100_2$
$D_{16} = 13_{10} = 1101_2$
$E_{16} = 14_{10} = 1110_2$
$F_{16} = 15_{10} = 1111_2$

Registers

32 bit	16 bit	8 bit
`%eax`	`%ax`	`%ah, %al`
`%ecx`	`%cx`	`%ch, %cl`
`%edx`	`%dx`	`%bh, %bl`
`%ebx`	`%bx`	`%dh, %dl`
`%edi`	`%di`	
`%esi`	`%si`	
`%ebp`	`%bp`	
`%esp`	`%sp`	

Assembly

Most instructions with two operands have the order: Source, Destination
e.g., `subl s,d` means d = d - s; `imull s,d` means d = d * s

Comparison (`cmp`) and test instructions have operand order: Source2, Source1
e.g., `cmpl s2,s1` means s1 - s2; `test s2,s1` means s1 & s2

Suffixes for set, jump, and conditional move instructions are:
e.g., `jl` means jump if less, `setns` means set not signed
general - `e`: equal, `ne`: not equal, `s`: signed, `ns`: not signed
unsigned - `b`: below, `be`: below or equal, `a`: above, `ae`: above or equal
signed - `l`: less, `le`: less or equal, `g`: greater, `ge`: greater or equal