CPSC 313: Computer Hardware and Operating Systems

Unit 1: The y86 (as a sequential processor)

2024 Winter Term 1

THE UNIVERSITY OF BRITISH COLUMBIA
Computer Science
Faculty of Science

Admin

- Lab 1 and Quiz 0 are out and due in the next ~1 week!
- Anyone registered with CfA who has not provided your accommodation letter yet: Send it to cpsc313-admin@cs.ubc.ca ASAP! Don't start Quiz 0 or register for Quiz 1 until we have entered your accommodation information.

Today

- Topics:
  - The ALU: Implementing Arithmetic and Logical Operators
    - Arithmetic: Add, Sub (Mul, Div, Mod)
    - Logical: Xor, And
  - Condition codes
- Learning outcomes:
  - Explain what the y86 condition codes ZF and SF really mean
  - Read/Write simple y86 programs

Pre-Class and In-Class Work

- Today and in general we expect you to have done all the pre-class work!
- In class exercises (small groups recommended!) will help you develop intuition about WHY things are as they are.
- This means that we may ask you to try to do things you don't already know how to do.

Bitwise Logical Operations: A Refresher

AND (&)

	0	**1**
0	0	0
1	0	1

XOR (^)

	0	**1**
0	0	1
1	1	0

Quick practice (we'll do more later):

- 0xEEEE & 0x1111
- 0xEEEE ^ 0x1111
- 0x9393 & 0xA0A0
- 0x9393 ^ 0xA0A0

What is 0xE? 14

What is 14 in binary? 8 + 4 + 2 = 1110

What is 0x1?? 1

What is 1 in binary? 1 = 0001

Now, what is 0xEEEE & 0x1111? 0000

0xEEEE & 0x1111

E == 1110

```
1110 1110 1110 1110
0001 0001 0001 0001

0000 0000 0000 0000   --> 0x0000
```

0xEEEE ^ 0x1111

```
1110 1110 1110 1110
0001 0001 0001 0001


1111 1111 1111 1111 == 0xFFFF
```

0x9393 & 0xA0A0

```
1001 0011 1001 0011
1010 0000 1010 0000

1000 0000 1000 0000  -> 0x8080
```

0x9393 ^ 0xA0A0

```
1001 0011 1001 0011
1010 0000 1010 0000


0011 0011 0011 0011 == 0x3333
```

Arithmetic Fundamentals

- Addition is pretty traditional. Assuming 16-bit values:

  0x5 + 0x3 = 0x8

  0xF + 0x1 = 0x10

  0xFFFF + 0x1 = 0x10000

- But, how does hardware perform subtraction?

  0x5 - 0x3 = 0x5 + (-0x3)

  -0x3 => 0011 -> 1100 -> 1101

```
 0101
+1101
 0010
```

Using Condition Codes to Jump

- The whole reason we have condition codes (ZF, SF, OF) is to implement control structures, such as if-then, if-then-else, and loops.
- Writing assembly involves a lot of figuring out which ALU instruction to use to set the condition flags so that you can use the right conditional jump – we'll give you lots of practice!

Inclass Exercises:

If there's time:
Playing with Some Programs in the Simulator

Coming Up

- Always: more pre- and in-class exercises!
- Lab 1 due Sunday
- Quiz 0 due Sep 18
- Our first scheduled tutorials Tuesday-Friday of this week!