Warm up! Compute:

```
0x1a ^ 0x72
```

= 0x68

0x1a = 0001 1010

0x72 = ^ 0111 0010

0110 1000

6 8

Bit-wise Operators, Git

CS 2130: Computer Systems and Organization 1

September 7, 2022

Announcements

- Homework 1 due Monday 9/12/2022
- TA office hours
  - **In-person**: Olsson 001, Wed-Sun, 5-7pm
  - **Online**: Discord, Wed-Sun, varies
  - Discord is now available
- My office hours
  - Tuesday, 4-5pm, Discord/Zoom
  - Wednesday, 4:30-6pm, Rice 210 (masks requested)
  - Thursday, 11am-12pm, Discord/Zoom

Quiz Review

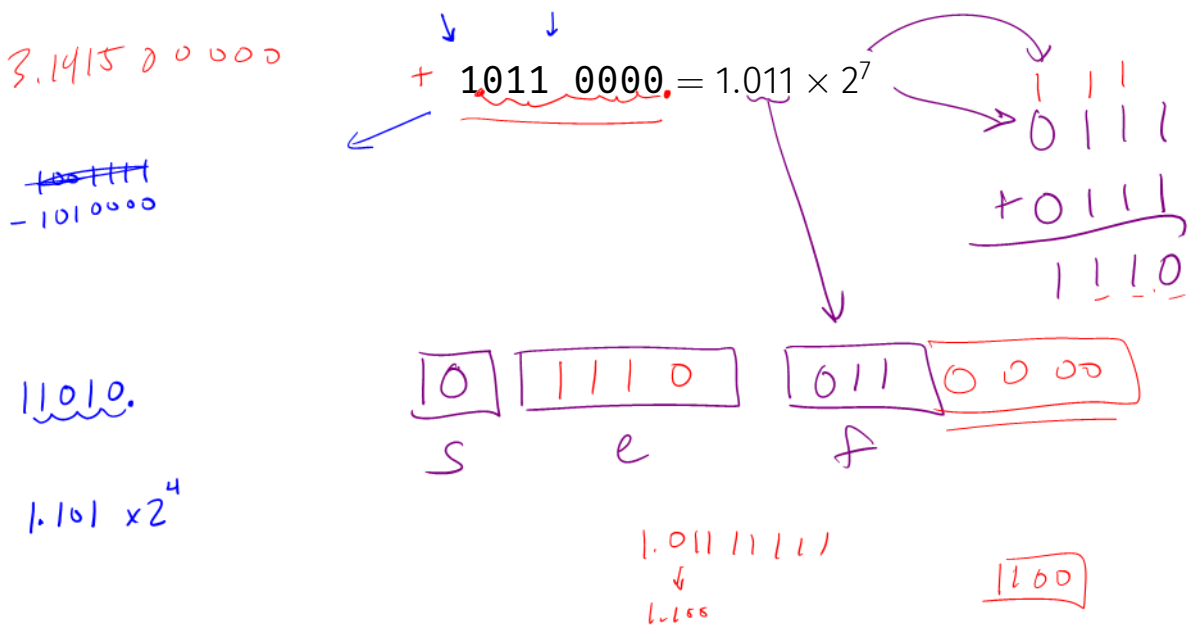
$0x12_{16} > 12_{10}$

18

$0x8$ 8

10

Quiz Review


3.1415 0 0 0 0
+ 1011 0000. = 1.011 × 2^7
1 1 1
0 1 1 1
+ 0 1 1 1
1 1 1 0
- 101 0000
1101 0.
1.101 x 2^4
0 1 1 1 0 0 1 1 0 0 0 0
s e f
1.011 1 1 1 1
1.155
1100

Operations (on Integers)

Bit vector: fixed-length sequence of bits (ex: bits in an integer)

- Manipulated by bitwise operations

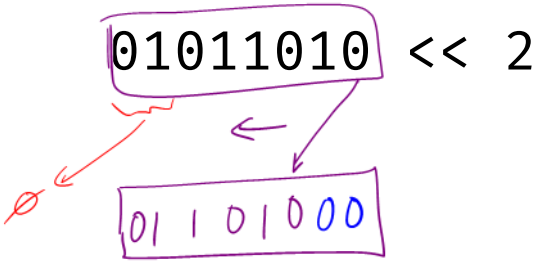
Bitwise operations: operate over the bits in a bit vector

- Bitwise not: $\sim x$ - flips all bits (unary)
- Bitwise and: $x\&y$ - set bit to 1 if x, y have 1 in same bit
- Bitwise or: $x|y$ - set bit to 1 if either x or y have 1
- Bitwise xor: $x\hat{}y$ - set bit to 1 if x, y bit differs

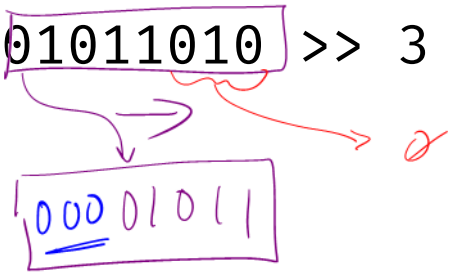
Operations (on Integers)

- Logical not: $!x$
  - $!0 = 1$ and $!x = 0, \forall x \neq 0$
  - Useful in C, no booleans
  - Some languages name this one differently
- Left shift: $x << y$ - move bits to the left
  - Effectively multiply by powers of 2
- Right shift: $x >> y$ - move bits to the right
  - Effectively divide by powers of 2
  - Signed (extend sign bit) vs unsigned (extend 0)

Left Bit-shift Example



Right Bit-shift Example



Bit-shift

Computing bit-shift effectively multiplies/divides by powers of 2

Consider decimal:

$$2130 <<_{10} 2 = 213000 = 2130 \times 100$$

$$2130 >>_{10} 1 = 213 = 2130 / 10$$

Right Bit-shift Example 2

```
11001010 >> 1
```

unsigned

202

0 1100101 → 101

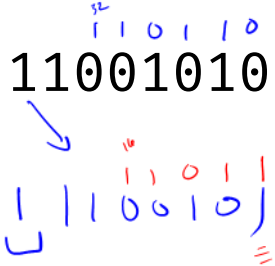
Right Bit-shift Example 2

For **signed** integers, extend the sign bit (1)

- Keeps negative value (if applicable)
- Approximates divide by powers of 2

Signed

```
11001010 >> 1
```



-54

-27

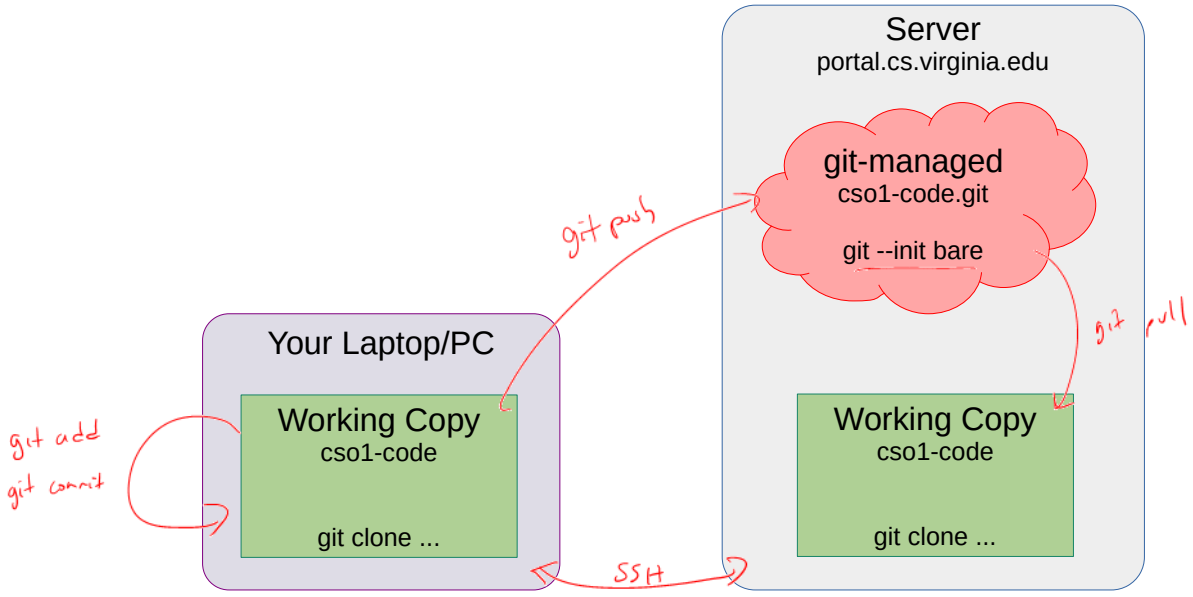
git

git

git: distributed version control

- Created by Linus Torvalds (Linux)
- Free and open source software
- Separate from GitHub/GitLab/... which use git
- Website: git-scm.com

git in this Class



Review

- Transistors
- Information modeled by voltage through wires (1 vs 0)
- Gates

- Examples of and, not gates
- Multi-bit values: representing integers
  - Signed and unsigned
- Floating point

How to do the work of multi-bit?

Multi-bit Mux

Our first multi-bit example: mux

Adder

Add 2 1-bit numbers: a, b

Adder

What is missing? Consider:

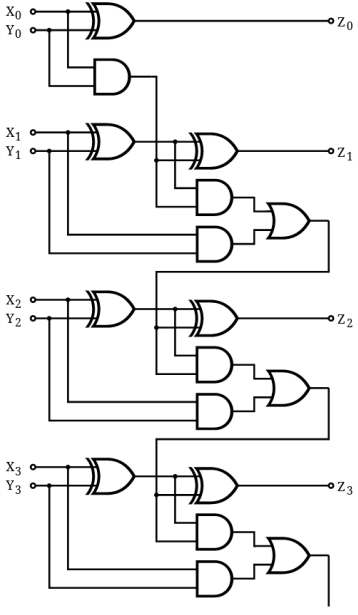
```
  11
+01
----
```

3-input Adder

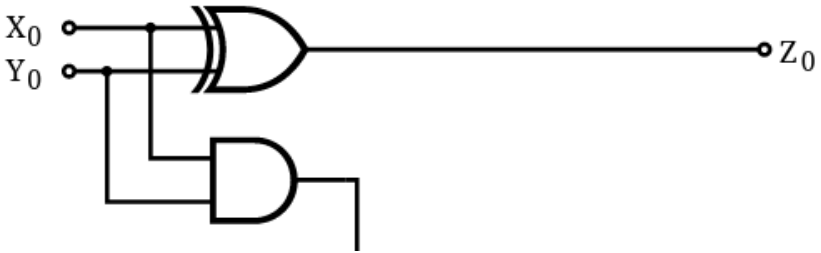
Add 3 1-bit numbers: a, b, c

Aside: 3-input AND / XOR

Ripple-Carry Adder



Ripple-Carry Adder



Ripple-Carry Adder