Lecture 1: Introduction to Computer Systems

CS 105

Fall 2024

https://cs.pomona.edu/classes/cs105/

Abstraction

Correctness

- **Example 1: Is $x^2 \geq 0$?**

- **Example 2: Is $(x + y) + z = x + (y + z)$?**

Performance

```
void copyij(int src[2048][2048],
            int dst[2048][2048]){
  int i,j;
  for (i = 0; i < 2048; i++){
    for (j = 0; j < 2048; j++){
      dst[i][j] = src[i][j];
    }
  }
}
```

```
void copyji(int src[2048][2048],
            int dst[2048][2048]){
  int i,j;
  for (j = 0; j < 2048; j++){
    for (i = 0; i < 2048; i++){
      dst[i][j] = src[i][j];
    }
  }
}
```

Security

```
int buggy_authenticate(){
  char password[4]; // allocate space to store a string
  gets(password);   // initialize string from user input

  return 0;         // always returns False
}

void example3(){
  if(buggy_authenticate()){ // equivalent to if False
    printf("The answer is 42\n"); // should never happen
  } else {
    printf("Unauthenticated User (correct behavior)\n");
  }
}
```

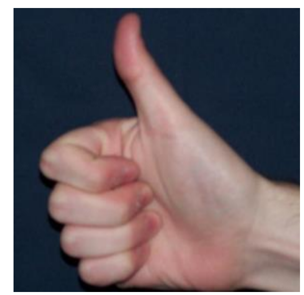
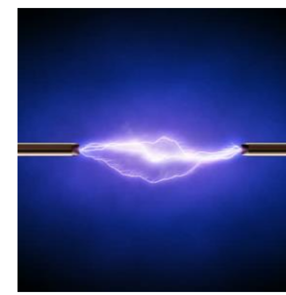
BITS

Bits

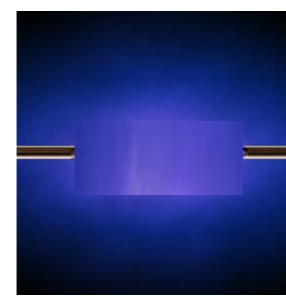
- a **bit** is a binary digit that can have two possible values

- can be physically represented with a two state device



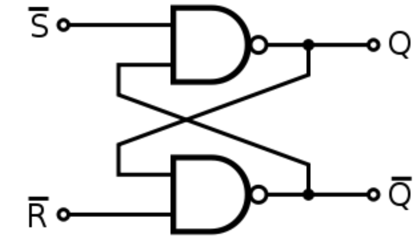






Storing bits

- Static random access memory (SRAM): stores each bit of data in a flip-flop, a circuit with two stable states
- Dynamic Memory (DRAM): stores each bit of data in a capacitor, which stores energy in an electric field (or not)
- Magnetic Disk: regions of the platter are magnetized with either N-S polarity or S-N polarity
- Optical Disk: stores bits as tiny indentations (pits) or not (lands) that reflect light differently
- Flash Disk: electrons are stored in one of two gates separated by oxide layers







Boolean Algebra

- Developed by George Boole in 19th Century
- Algebraic representation of logic---encode “True” as 1 and “False” as 0

And

&	0	1
0	0	0
1	0	1

Or

|	0	1
0	0	1
1	1	1

Not

~	
0	1
1	0

Exclusive-Or (Xor)

^	0	1
0	0	1
1	1	0

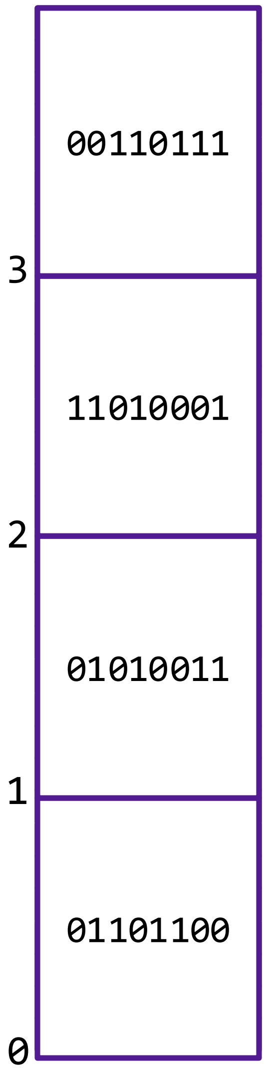
- How does this map to set operations?

Exercise 1: Boolean Operations

- Evaluate each of the following expressions
  1. `1 | (~1)`
  2. `~( 1 | 1)`
  3. `(~1) & 1`
  4. `~( 1 ^ 1)`

Bytes and Memory

- **Memory** is an array of ~~bits~~ bytes
- A **byte** is a unit of eight bits
- An index into the array of memory is an **address**, **location**, or **pointer**
- We speak of the *value* in memory at an address
  - The value may be a single byte …
  - … or a multi-byte quantity starting at that address



General Boolean algebras

- Bitwise operations on bytes

```
  01101001      01101001      01101001
& 01010101    | 01010101    ^ 01010101    ~ 01010101
----------    ----------    ----------    ----------
  01000001      01111101      00111100      10101010
```

Exercise 2: Bitwise Operations

- Assume: `a = 01101100, b = 10101010`

- What are the results of evaluating the following Boolean operations?

  - `~a`
  - `a & b`
  - `a | b`
  - `a ^ b`

Bitwise vs Logical Operations

- Bitwise Operators &, |, ~, ^
  - View arguments as bit vectors
  - operations applied bit-wise in parallel
- Logical Operators &&, ||, !
  - View 0 as “False”
  - View anything nonzero as “True”
  - Always return 0 or 1
  - Early termination

Exercise 3: Bitwise vs Logical Operations

- `~01101100`
- `~00000000`
- `~~01101100`

- `!01101100`
- `!00000000`
- `!!01101100`

- `01101100 & 10101010`
- `01101100 | 10101010`

- `01101100 && 10101010`
- `01101100 || 10101010`

Bit Shifting

- Left Shift: **`x << y`**
  - Shift bit-vector **`x`** left **`y`** positions
  - Throw away extra bits on left
  - Fill with 0's on right

Undefined Behavior if you shift amount < 0 or ≥ word size

- Right Shift: **`x >> y`**
  - Shift bit-vector **`x`** right **`y`** positions
  - Throw away extra bits on right
  - Logical shift: Fill with 0's on left
  - Arithmetic shift: Replicate most significant bit on left

Choice between logical and arithmetic depends on the type of data

Example: Bit Shifting

- 01101001 << 4 10010000
- 01101001 $>>_l$ 2 00011010
- 01101001 $>>_a$ 4 00000110

Exercise 4: Bit Shifting

- 10101010 << 4
- 10101010 $>>_l$ 4
- 10101010 $>>_a$ 4

Bits and Bytes Require Interpretation

10001100 00001100 10101100 00000000

might be interpreted as

- The integer 3,485,745
- A floating point number close to 4.884569×10^{-39}
- The string “105”
- A portion of an image or video
- An address in memory

Information is Bits + Context

LOGISTICS

The Course in a Nutshell

- Textbooks (not required)
    - Bryant and O'Halloran, *Computer Systems: A Programmer's Perspective*, **third edition**, Pearson, 2016
    - Arpaci-Dusseau and Arpaci-Dusseau, *Operating Systems: Three Easy Pieces,* online, 2018
- Classes
    - Monday and Wednesday, 2:45-4pm in Edmunds 101
- Labs
    - Wednesdays 7-8:15pm in Edmunds 229/219
    - **Starts this Wednesday**!
- Office Hours TBA (M 4:15-5:15pm today)
- Mentor Sessions TBA

Grading

- Assignments (9)
  - Introduced during labs, Due Tuesdays at 11:59pm
  - Tremendous fun, work in pairs
  - 10 late days

- Check-ins (5)
  - three-question quizzes (13 topics total)
  - September 18, October 9, October 30, November 20, December 4
  - Can improve grade on any topics(s) with "Extra Chance Check-in" (may take after any later check-in or during final exam time Dec 9 @2-5pm)

- Grades
  - Must successfully complete all the assignments
  - Beyond that, 45% assignments, 50% check-ins, 5% participation

Course website

https://cs.pomona.edu/classes/cs105

- All information is on the course website
- All course materials get posted on the course website
- Links from the course page:
  - Slack (#cs105-2024fa), for questions and discussion
  - Gradescope, for submitting assignments and seeing grades
  - Additional resources