Bits, Bytes and Integers – Part 1

15-213/14-513/15-513: Introduction to Computer Systems
2nd Lecture, September 1, 2022

Instructors:

Dave Andersen (15-213)

Zack Weinberg (15-213)

Brian Railing (15-513)

David Varodayan (14-513)

Waitlist questions

- 15-213: Mary Widom (marwidom@cs.cmu.edu)
- 15-513: Mary Widom (marwidom@cs.cmu.edu)
- 14-513: INI Enrollment (ini-academic@andrew.cmu.edu)

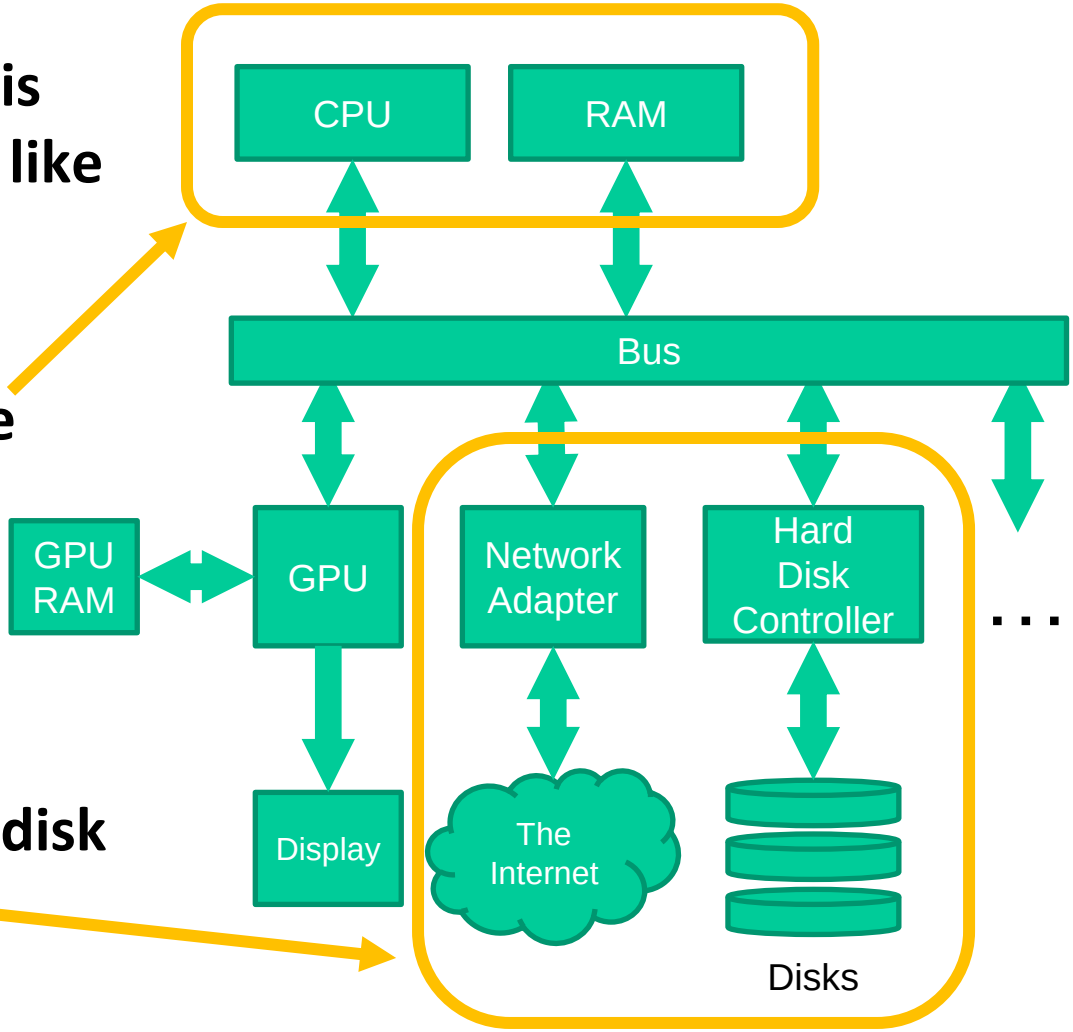
- Please don't contact the instructors with waitlist questions.

Reminder: Lab Deadlines

- C Programming Lab: Came out Tuesday, due Sep. 6
- Data Lab: Comes out today, due Sep. 15
- Start early, start often

Roadmap – Inside a Computer

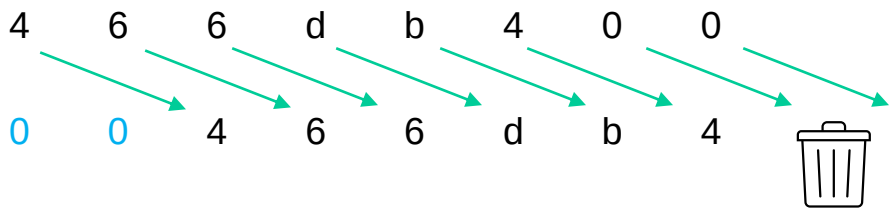
- You may have seen this block diagram, or one like it, before.
- For the first half of the course we'll be concentrating on the CPU and RAM
- Second half discusses disk and network I/O
- Graphics? Take 462



Course Teaser (Things You May Know Already)

What value do you get if you arithmetic right-shift the hexadecimal number `0x466db400` by eight bits?

- **Hexadecimal is just like decimal … if you have sixteen fingers.**
- **"Eight bits" is two hex digits.**
- **"Arithmetic right shift" is division by a power of two.**



Course Teaser (Things You May Know Already)

On an x86-64 machine, how much space does this C struct take? That is, what is the value of `sizeof(struct S)`?

```
struct S {
    char c[2];
    char *p;
    int z;
}
```

2 bytes

6 bytes wasted space

8 bytes

4 bytes

4 bytes wasted space

24 bytes total

Course Teaser (Things You May Know Already)

```
/* a.c */
#include <stdio.h>
extern long x;
int main(void)
{
  printf("%ld\n", x);
  return 0;
}
```

```
/* b.c */

double x = 3.14159;
```

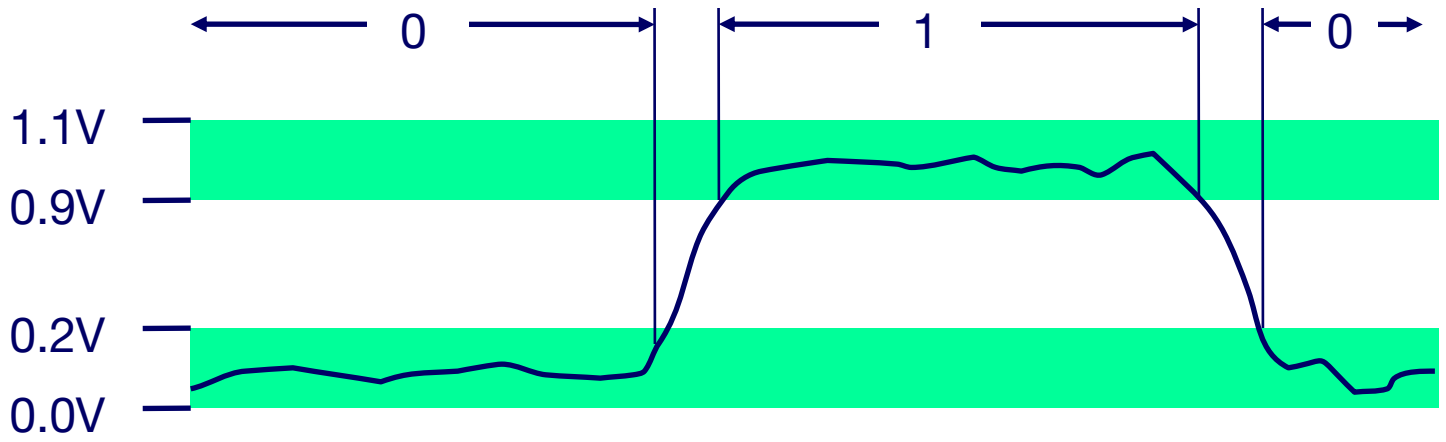
What's wrong with this program?

Today: Bits, Bytes, and Integers

- **Representing information as bits**
- **Bit-level manipulations**
- **Integers**
  - Representation: unsigned and signed
  - Conversion, casting
  - Expanding, truncating
  - Addition, negation, multiplication, shifting
  - Summary
- **Representations in memory, pointers, strings**

Everything is bits

- **Each bit is 0 or 1**
- **By encoding/interpreting sets of bits in various ways**
  - Computers determine what to do (instructions)
  - … and represent and manipulate numbers, sets, strings, etc…
- **Why bits? Electronic Implementation**
  - Easy to store with bistable elements
  - Reliably transmitted on noisy and inaccurate wires



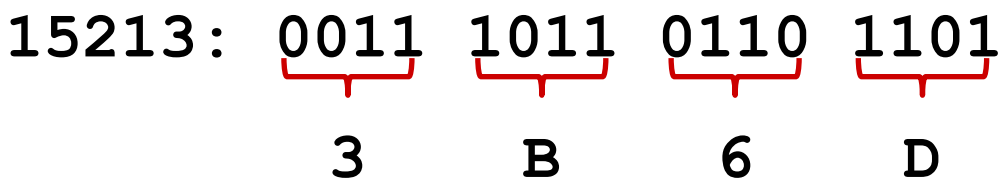
For example, can count in binary

- **Base 2 Number Representation**
  - Represent 15213_{10} as 11101101101101_2
  - Represent 1.20_{10} as $1.0011001100110011[0011]\ldots_2$
  - Represent 1.5213×10^4 as $1.1101101101101_2 \times 2^{13}$

Encoding Byte Values

- **Byte = 8 bits**
  - Binary 00000000_2 to 11111111_2
  - Decimal: 0_{10} to 255_{10}
  - Hexadecimal 00_{16} to FF_{16}
    - Base 16 number representation
    - Use characters '0' to '9' and 'A' to 'F'
    - Write $FA1D37B_{16}$ in C as
      - 0xFA1D37B
      - 0xfa1d37b

Hex	Decimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111



Preview: Combining bytes…

C Data Type	Typical 32-bit	Typical 64-bit
`char`	1	1
`short`	2	2
`int`	4	4
`long`	4	8
`float`	4	4
`double`	8	8
pointer	4	8

Preview: … to make integers

	W			
	8	16	32	64
UMax	255	65,535	4,294,967,295	18,446,744,073,709,551,615
TMax	127	32,767	2,147,483,647	9,223,372,036,854,775,807
TMin	-128	-32,768	-2,147,483,648	-9,223,372,036,854,775,808

- UMax = $2^w - 1$ where *w* is the number of bits ("word size")
- UMin = 0
- TMax = $2^{w-1} - 1$
- TMin = -2^{w-1}
  - Asymmetric!
  - Because of zero

Activity: binary, hexadecimal, twos complement

https://canvas.cmu.edu/courses/30386/assignments/524495

https://www.cs.cmu.edu/~213/activities/bits-and-bytes.pdf

- **Form groups of three or four people**
  - Three is probably easier in this room, since the chairs don't move
- **Person in the middle of each group, bring up the PDF in a program that lets you scribble on it**
  - If you can't do that, swap chairs with someone in your group who can
- **Work through the activity *as a group***
  - Today, we just go start to finish at your own pace
  - Faculty and TAs will be walking around to help
- **When you're done, upload your annotated PDF to Canvas**