

DATA REPRESENTATION (CONTD)

Problem Solving with Computers-I

<https://ucsb-cs16-wi17.github.io/>

C++

```
#include <iostream>
using namespace std;

int main(){
    cout<<"Hola Facebook\n";
    return 0;
}
```



Recap: Representation of non-negative numbers

- Positional encoding
- External representation
- Internal representation
- Binary representation:
 - Only two symbols: 0 and 1
 - Each position is called a *bit*
 - 8 bits makes a byte
 - Bits take up space

$$\begin{array}{cccc} 1 & 0 & 2 & A \\ \hline 16^3 & 16^2 & 16^1 & 16^0 \end{array}$$

weight \rightarrow

$$\begin{array}{cccc} 1 & 1 & 0 & 10 \\ \hline 16 & 8 & 4 & 2 \end{array}$$
$$= 16 + 8 + 2 = 26_{10}$$

Converting between binary and decimal

Binary to decimal: $1\ 0\ 1\ 1\ 0_2 = ?_{10}$

16 8 4 2 1

$$16 + 4 + 2 = 22_{10}$$

Decimal to binary: $34_{10} = ?_2$

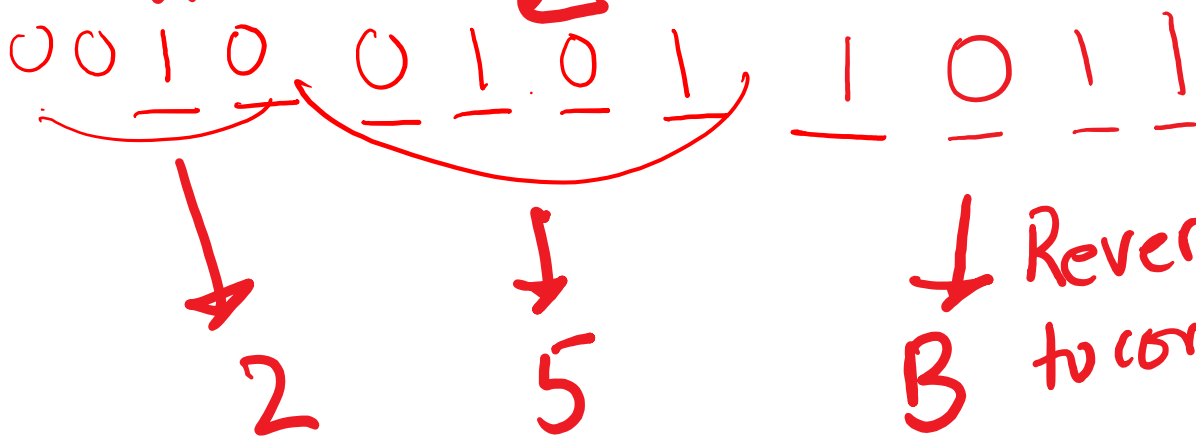
$\begin{array}{r} 100010 \\ - 32 \\ \hline 2 \end{array}$

$\begin{array}{r} 1\ 0\ 0\ 0\ 1\ 0 \\ \hline 32\ 16\ 8\ 4\ 2\ 1 \end{array}$

Hex to binary

- Each hex digit corresponds directly to four binary digits
- Programmers love hex, why? *Represent large constants in a fast way & easy conversion to binary*

$25B_{16} = ?$ In binary



Reverse the process to convert back to hex

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Hexadecimal to decimal

$$25B_{16} = ? \text{ Decimal}$$

$$\underline{256} \quad \underline{16} \quad \underline{1}$$

$$2 * 256 + 5 * 16 + 11 * 1$$
$$= 603$$

Hexadecimal to decimal

- Use polynomial expansion
- $25B_{16} = 2 \cdot 256 + 5 \cdot 16 + 11 \cdot 1 = 512 + 80 + 11$
 $= 603$

0X stands for hex representation

- Decimal to hex: $36_{10} = ?_{16}$

$$\begin{array}{r} 32 \\ \hline 4 \end{array}$$

$$\begin{array}{r} 0 \\ \hline 256 \end{array} \quad \begin{array}{r} 2 \\ \hline 16 \end{array} \quad \begin{array}{r} 4 \\ \hline 1 \end{array}$$

0X24

Decimal vs. Hexadecimal vs. Binary

Examples:

1010 1100 0011 (binary)
= 0xAC3

10111 (binary)
= 0001 0111 (binary)
= 0x17

0x3F9
= 11 1111 1001 (binary)

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Binary to hex: ⁰⁰100011100

A. 8F0

2 3 (12)

☒ B. 23C

Adding leading zeros is okay because it doesn't change the value of the number

C. None of the above

BIG IDEA: Bits can represent anything!!

Positional encoding scheme

Numbers	Binary Code
---------	-------------

0	→ 00
1	→ 01
2	→ 10
3	→ 11

01 (1)
+ 10 (2)

11 (3)
Makes sense!

+ $\begin{array}{r} 01 \\ 01 \\ \hline 10 \end{array}$ $\begin{array}{l} (1) \\ (1) \\ (2) \end{array}$ $\left[\begin{array}{l} \text{Digit by digit} \\ \text{addition works out.} \\ \text{Good for making efficient} \\ \text{circuits} \end{array} \right]$

How many (minimum) bits are required to represent the numbers 0 to 3?

Arbitrary scheme

Numbers	Code
0	→ 11
1	→ 10
2	→ 00
3	→ 01

+ $\begin{array}{r} 10 \\ 10 \\ \hline 00 \end{array}$ $\begin{array}{l} (1) \\ (1) \\ (2) \end{array}$ $\left[\begin{array}{l} + 10 (1) \\ + 00 (2) \\ \hline 10 (1) \end{array} \right]$ $\begin{array}{l} \text{#2} \\ = 1? \end{array}$
Lucky coincidence

BIG IDEA: Bits can represent anything!!

Arbitrary mapping is okay here

Colors

Binary code



Red



0 0

1 1



Green



0 1

0 1



Blue



1 0

1 0

How many (minimum) bits are required to represent the three colors?

*2 bits
(can't have fractional bits)*

BIG IDEA: Bits can represent anything!!

Characters

Code

'a'  000

'b'  001

'c'  010

'd'  011

'e'  100

6 characters needs 3 bits to encode

N bits can represent at most 2^N things

What is the minimum number of bits required to represent all the letters in the English alphabet?

A. 3

B. 4

☒ C. 5

D. 6

E. 26

26 letters in the alphabet

$2^5 = 32$ (5 bits gives us enough unique bit patterns)

$2^4 = 16$ (4 bits is not enough)

BIG IDEA: Bits can represent anything!!

- Logical values?

- 0 \Rightarrow False, 1 \Rightarrow True

- colors ?

- Characters?

- 26 letters \Rightarrow 5 bits ($2^5 = 32$)
 - upper/lower case + punctuation \Rightarrow 7 bits (in 8) (“ASCII”)

- standard code to cover all the world’s languages \Rightarrow 8,16,32 bits (“Unicode”)

www.unicode.com

- locations / addresses? commands?

- **MEMORIZE:** N bits \Leftrightarrow at most 2^N things

Red

Green

Blue



What is the maximum positive value that can be stored in a byte?

A. 127

B. 128

☒ C. 255

D. 256

1 byte \longrightarrow 8 bits

\downarrow

2^8

$= 256$

unsigned char x;

\downarrow

use 1 byte to store x

\rightarrow Intend to store only non-negative numbers

In that case, the range of values that can be stored are 0-255

Generalize to N bits

BIG IDEA: Bits can represent anything!!

[Positive & Negative nos]

Signed numbers

Binary Code

(This representation is called 2's complement)

-3	→	1 0 1	
-2	→	1 1 0	
-1	→	1 1 1	
0	→	0 0 0	
1	→	0 0 1	
2	→	0 1 0	
		<u>-4</u> <u>2</u> <u>1</u>	

+

is

1	1 0	(-2)
0	1 0	(2)
<hr/>		
0	0 0	(0)

How many (minimum) bits are required to represent the numbers -3 to 2? (3)

Two's Complement

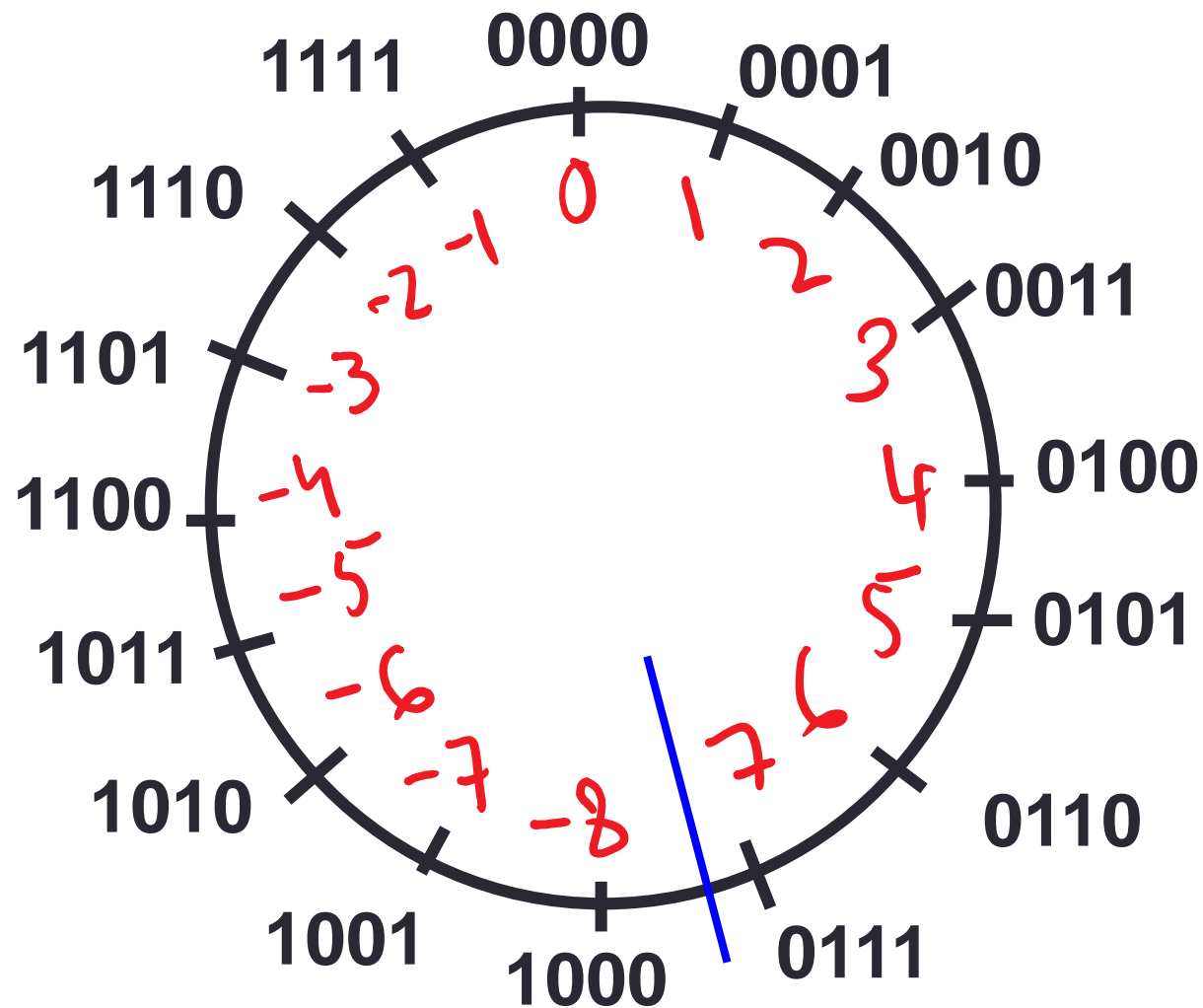
- Most significant bit represents a large negative weight:

$$\frac{1}{-2^3} \quad \frac{0}{2^2} \quad \frac{0}{2^1} \quad \frac{1}{2^0} = -8 + 1 = -7$$

- To find the 2's complement representation
 - Write unsigned representation of the number saving one bit for sign
 - Flip all the bits
 - Add 1

Two's Complement

- Flip all the bits of unsigned representation and add 1



$$2 - 3 = ?$$

$$\begin{array}{r}
 0010 \text{ (2)} \\
 + 1101 \text{ (-3)} \\
 \hline
 1111 \text{ (-1)}
 \end{array}$$

Two's Complement: $1101_2 = ?_{10}$

A. -2

☒ B. -3

C. -4

D. -5

$$\begin{array}{cccc} \overline{1} & \overline{1} & \overline{0} & \overline{1} \\ \downarrow & \downarrow & \downarrow & \downarrow \\ -8 & +4 & +1 & \\ \hline & & & -3 \end{array}$$

Addition and Subtraction

- Positive and negative numbers are handled in the same way.
- The carry out from the most significant bit is ignored.
- To perform the subtraction $A - B$, compute $A + (\text{two's complement of } B)$

Data types

Binary numbers in memory are stored using a finite, fixed number of bits typically:

- 8 bits (byte)
- 16 bits (half word)
- 32 bits (word)
- 64 bits (double word or quad)

Data type of a variable determines the:

- exact representation of variable in memory
- number of bits used (fixed and finite)
 - range of values that can be correctly represented

Next time

- Under the hood of program compilation
- Separate compilation with makefiles