Num ations

There are 10 types of this world

Those who understand binary and those who don't

Agenda

- Bits, Bytes, and Words
- Number bases and base conversion
  - Positional notation

- Binary arithmetic an[illegible]ion
  - Signed numbers
  - Arithmetic and overflow
  - Packed Decimal, ASCII, Parity…

From Lecture 1: Below Your Program

- High-level language program (in C)

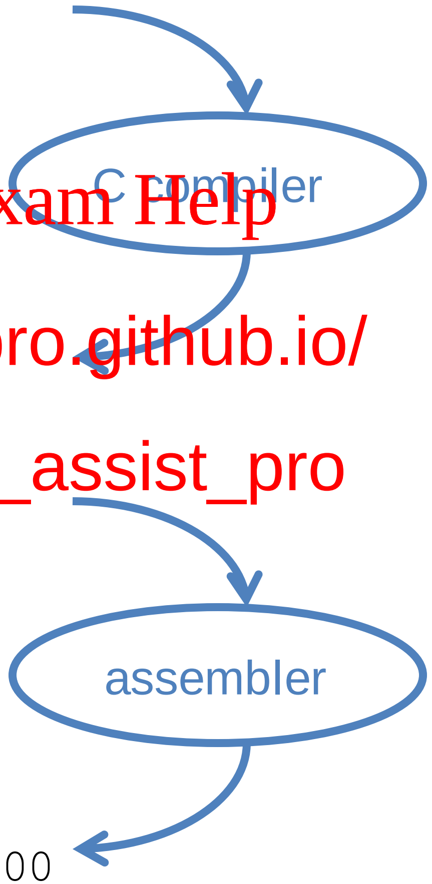
```
swap (int v[], int k) {
    int temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```

- Assembly language pr

```
swap:  sll
       add    $2, $4,$2
       lw     $15, 0($2)
       lw     $16, 4($2)
       sw     $16, 0($2)
       sw     $15, 4($2)
       jr     $31
```

- Machine (object) code (for MIPS)

```
000000 00000 00101 0001000010000000
000000 00100 00010 0001000000100000
 . . .
```



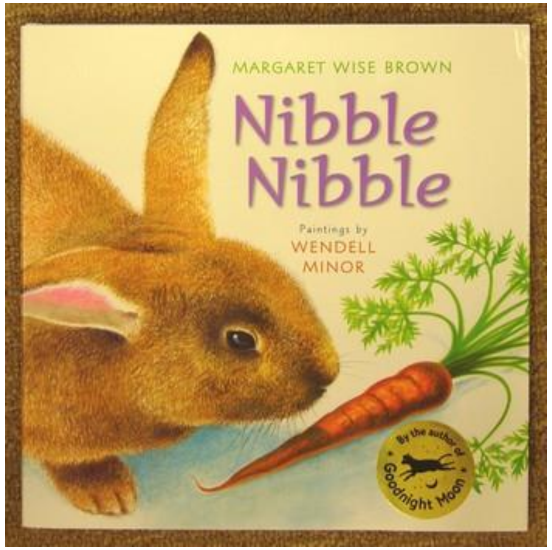
How do peo ers represent

Why Base 10? Why Base 2?

- **Decimal: Base 10**, a single number from 0-9
- **Binary: Base 2**, a single digit is called a ***bit*** (binary digit)
- A bit is the smallest unit of information, and can represent…
  - 1 / 0
  - True / False
  - Yes / No
  - On / Off
  - used in a two-state (Boolean) logic
- Can represent anything with a sequence of binary bits, but the bit patterns have *no* intrinsic meaning by themselves!

Nibbles to Words

- Typically store information in groups
  - a **byte** is a group of 8 bits
    - e.g. 01100101
  - a **nibble/nybble** (a s 4 bits,
    - e.g. 0110
  - a **word** (MIPS) is a group of 4 bytes, or 32 bits
    - e.g. 01100101011001010110010101100101
- Least significant bit right most



Numbers and Positional Notation

$\mathbf{a}_n\mathbf{a}_{n-1}\ldots\ldots\ldots\mathbf{a}_1\mathbf{a}_0\ .\ \mathbf{a}_{-1}\mathbf{a}_{-2}\ldots\ldots\ldots\mathbf{a}_{-m}$

$i=-m$

$N = \mathbf{a}_n b^n + \mathbf{a}_{n-1} b^{n-1} + \ldots\ldots + \mathbf{a}_1 b + \mathbf{a}_0 + \mathbf{a}_{-1} b^{-1} + \ldots\ldots + \mathbf{a}_{-m} b^{-m}$

Examples

- 238_{10}

- 10110_2

Common Bases

Name of Base	Base	Digits used
Decimal	10	0,1,2,3,4,5,6,7,8,9
Binary	2	[illegible]
Octal	8	[illegible],6,7
Hexadecimal	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

- We often write hex numbers preceded by 0x

$1011_2 = 11_{10} = 13_8 = B_{16} = 0xB$

How many bits are needed to represent a decimal number?

- What is the largest decimal number with **n** digits?

  $10^n - 1$

- What is the largest digits?

  $2^m - 1$

- For base b, the largest number is $b^k - 1$

How many bits are needed to represent a decimal number?

How many digits necessary for numbers up to one million?

How to Conve ase to Another?

Base Conversion – Decimal to Another Base

- Approach 1: Make a table
  - Divide by **b**^{i}
  - The quotient is the most significant bit
  - Repeat with the rem

- Approach 2:
  - Divide by **b**
  - Remainder of result is least significant bit
  - Repeat with the quotient

Base Conversion – Decimal to Another Base

Example: What is 523_{10} in binary?

Base Conversion – Decimal to Another Base

Example: What is 53241_{10} in hexadecimal?

Dec	Hex	Bin
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

Base Conversion – Other Base to Decimal

- Basic Approach: Direct expansion with positional weights

  $N = \mathbf{a}_n b^n + \mathbf{a}_{n-1} b^{n-1} + \ldots\ldots + \mathbf{a}_1 b + \mathbf{a}_0$

- Advanced Approach

  $N = a_n \mathbf{b}^n + a_{n-1} \mathbf{b}^{n-1} + \ldots\ldots + a_1 \mathbf{b} + a_0$

  $= (\ldots((a_n \mathbf{b} + a_{n-1})\mathbf{b} + a_{n-2})\mathbf{b} + \ldots + a_1)\mathbf{b} + a_0$

Base Conversion – Other Base to Decimal

- Examples: What is 1010101_2 in Decimal

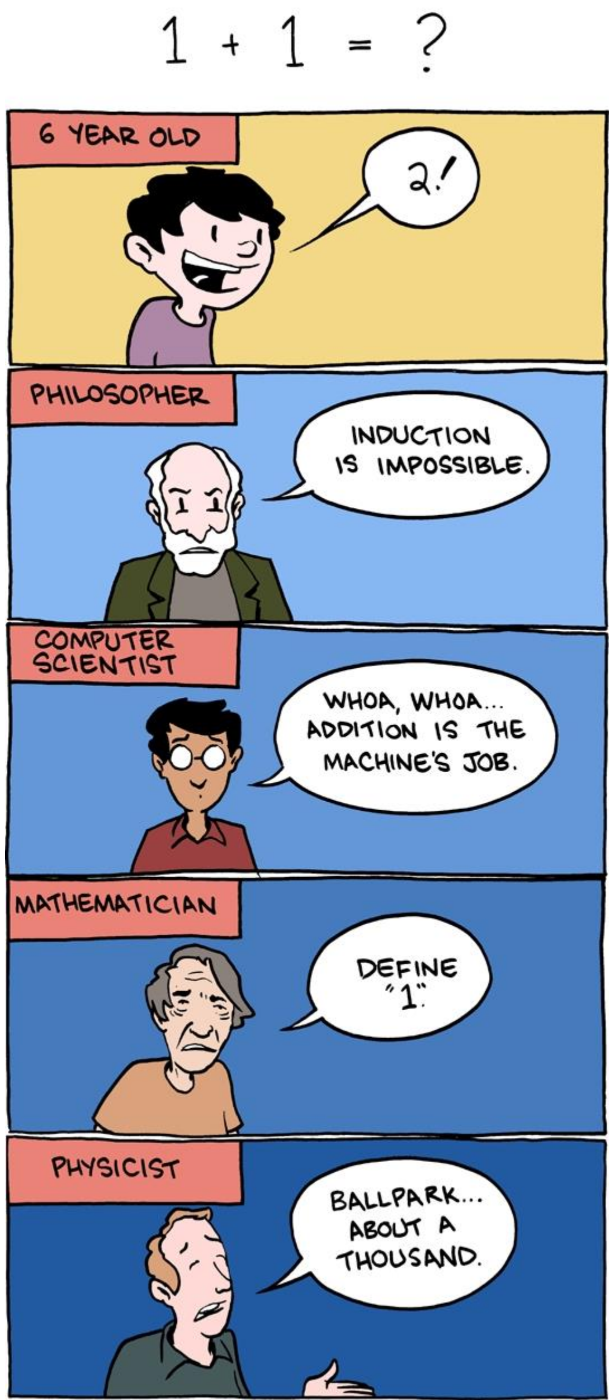
Base Conversion – Other Base to Decimal

Examples: What is $1AB_{16}$ in Decimal?

Base Conversion – Base A to Base B

- Conversion from base a to base b
  - First convert base a to decimal
  - Then convert decimal to base b
- Special cases (easier
  - Binary to hexadecimal and back
    - Example:
      11010101101_2
      $= 011010101101_2$
      $= 6AD_{16}$
  - Binary to octal and back
    - Example: 760_8 becomes 111110000_2

Dec	Hex	Bin
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



Questions

- What is the largest binary number with n bits?
- How do we add binary numbers?

- How ... ary numbers?

- Why do program... ays mix up Halloween and Christmas?

- How should we represent negative numbers?

How to Re https://eduassistpro.github.io/ d Numbers?

Sign-and-Magnitude

- The first approach
- Use the most significant bit to represent the sign

+13= 0000 1101

-13 = 1000 1101

- Problems
  - Two representations for zero: 0000 0000 and 1000 0000
  - Cannot add a positive number and a negative number together

One's Complement

- ***Invert each bit!***

  +13= 0000 1101

  -13 = 1111 0010

- ***Problems:***
  - *Still two representations for zero* 0 and 1111 1111
  - *Answer is off by 1*
  - Incorrect overflow
    - What is 16 + (-13)?

Two’s Complement

- **The gold standard**
- **Invert the bits and add one!**

  +13 = 0000 1101

  What is –13?

- Unique zero

  0000 0000

- MSB represents the sign.
  - 1 if negative
  - 0 if positive
- Negative numbers are defined as $-N = \mathbf{B}^n - N$

Two's Complement

- Easily implemented in hardware

- Range from -2^{n-1} to

- The complement of compleme , i.e., -(-Y) = Y.

Two’s Complement

- 16 - 13 = ?

The Hitchhiker's Guide to the Galaxy

What is the qu e universe, and everyt

Binary Arithmetic

Addition	Subtraction	Multiplication
0 + 0 = 0	0 - 0 = 0	0 * 0 = 0
0 + 1 = 1		* 1 = 0
1 + 0 = 1	1 - 0 = 1	0 = 0
1 + 1 = 0 carry 1	1 - 1 = 0	1 * 1 = 1

- Rules in base 10 are also valid in any other base
- Subtraction often done using addition and 2's complement
- Multiplication and division are similar. We will learn in a few lectures

Arithmetic overflow

- Typically use a fixed # of bits to represent numbers!
- *Arithmetic overflow* can occur during two's complement addition/subtracti

Arithmetic overflow

- Example: $6_{10} + 5_{10}$ using 4 bits

Arithmetic overflow

- Typically use a fixed # of bits to represent numbers!
- *Arithmetic overflow* can occur during two's complement

  addition/subtracti

  – Add 2 positive nu ve result

  – Add 2 negative numbers and ge result.
  – A minus B with A<0 and B>0 and getting positive result
  – A minus B with A>0 and B<0 and getting negative result
- Need to take extra care when implement it on the circuits

Packed Decimal

(Binary Coded Decimal, BCD)

- Replace each digit with its 4-bit equivalent

  5372_{10} in BCD is 0101 0011 0111 0010

- Good
  - User friendly? Yes!
  - BCD is easier for humans to parse
- Bad
  - Wastes storage space
  - BCD is harder to implement in hardware

Parity

- Used to check for corrupt data in storage or transmission, with two kinds: even and odd
  - Total # of bits with 1 [illegible] parity
  - Total # of bits with 1 [illegible] arity
- Examples:

Even Parity	[illegible]ity
0010101001	1110101000
0000000000	0011110111
1010101011	0000000001

- Advantage of odd parity?
- Detecting multiple errors? Correcting errors?

How to aracters?

Character Data Codes

Name	bits per symbol	# of symbols	Comments
IBM-BCD	6	64	Capital letters: A-Z, 0-9, $, etc. Not to confuse with Packed Decimal.
ASCII	7	128	All letters: a-z, A-Z, 0-9, $, BEL, TAB, etc. See link below.
USACII	8	[illegible]	[illegible] n).
EBCDIC	8	[illegible]	[illegible] s (odd parity)
UNICODE	16	[illegible]	[illegible] etters of all languages.

ASCII

American
Standard
Code for
Information
Interchange

UNICODE

The Martian

A ars.

The c roken.

Only one camera is wor und.

How can he communic arth?

48 4F 57 41 4C 49 56 45?
What was the Earth trying to say?

48 4F 57 41

4C 49 56 45?

Summary

- Definitions: Bits, Nibbles, Bytes, Words
- Representations: number bases, conversion

- Signed numbers w t

- Other data repres

  - Packed Decimal (BCD)
  - ASCII and other character data codes
  - Parity

Review and more information

- Textbook
  - Section 2.4, Signed and Unsigned Numbers

There are 10 types of people in this world…
Those who understand binary and those who don’t