
HENRY

Intro to CS

MAP OF COMPUTER SCIENCE

COMPUTATIONAL COMPLEXITY

DOES P≠NP?

NXN CHESS PSPACE NXN GO

NP COMPLETE — TRAVELLING SALESMAN, MAP COLOURING

NP — GRAPH ISOMORPHISM

BQP — DISCRETE LOGARITHM, FACTORING

EFFICIENT FOR A QUANTUM COMPUTER

P — TESTING IF PRIME, GRAPH CONNECTIVITY

EFFICIENT FOR A COMPUTER

INFORMATION THEORY

0100110100011001

COMPRESSION — ENTROPY

ERROR CORRECTION — PARITY CHECKING — CODING THEORY

CRYPTOGRAPHY

PUBLIC KEY

ALL THE SECRETS → 7bh;"%&mP$avR

7bh;"%&mP$avR → ALL THE SECRETS

PRIVATE KEY

THEORETICAL COMPUTER SCIENCE

ALGORITHMS

BUBBLESORT(A)
1: GO FROM LEFT TO RIGHT.
2: COMPARE EACH PAIR.
3: IF LEFT ONE HIGHER, SWITCH.
4: DO UNTIL NO MORE SWITCHES.

BUBBLE SORT $O(n^2)$

1 2 3 4 5 6 7 8

MERGE SORT $O(n \log n)$

1 2 3 4 5 6 7 8

ANALYSIS OF ALGORITHMS — ALGORITHMIC COMPLEXITY

COMPUTABILITY THEORY

LOGIC — ZERO, ONE, NOT, OR, AND, XOR, NAND

ALAN TURING

TURING MACHINE

STATE REGISTER

1: MOVE LEFT
2: MOVE RIGHT
3: FLIP DIGIT
...
123: STOP

INFINITELY LONG TAPE

HEAD

LAMBDA CALCULUS

GRAPH THEORY

COMPUTATIONAL GEOMETRY

AUTOMATA THEORY

QUANTUM COMPUTATION

AND MORE

PARALLEL PROGRAMMING

MAIN JOB → JOB 1, JOB 2, JOB 3, JOB 4 → SMALLER JOBS

DATA STRUCTURES

ARRAY, LINKED LIST, LIST, TREE, GRAPH, STACK, HASHING

FORMAL METHODS

START, POSSIBLE STATES, ALERT

MONITOR

SOUND

HARDWARE

GPU, SSD, RAM, POWER, INPUT

CPU MOTHERBOARD

SCHEDULING

PROCESSES → SCHEDULER → CPU 1, CPU 2, CPU 3, CPU 4 → MULTIPROCESSING

COMPUTER ARCHITECTURE

CPU: CENTRAL PROCESSING UNIT, CONTROL UNIT, ARITHMETIC/LOGIC UNIT, MEMORY UNIT — INPUT, OUTPUT

GPU — MULTIPROCESSORS

FPGA — LOGIC BLOCK, INTERCONNECTION, INPUT/OUTPUT, SWITCH BOX, CONNECT BLOCK

COMPUTER ENGINEERING

SOFTWARE AND PROGRAMMING LANGUAGES

PYTHON, PHP, SWIFT, JAVA, C#, C++, C, ASSEMBLY, JAVASCRIPT, PERL, SQL, WEB APPS, BROWSER, APPLICATIONS, OPERATING SYSTEM, BIOS, SILICON, MACHINE CODE

JAVASCRIPT → COMPILER → C++ → COMPILER → C → COMPILER → ASSEMBLY → COMPILER → MACHINE CODE

COMPILERS

DATA MANAGEMENT

DATABASES, SQL, DATACENTRES

NETWORKING

CONCURRENT/DISTRIBUTED/PARALLEL SYSTEMS

PERFORMANCE

COMPUTER ANALYSIS — BENCHMARKING

OPERATING SYSTEMS

iOS, macOS

SOFTWARE ENGINEERING

FORMAL METHODS, UNIT TESTING, VERSION CONTROL, OBJECT ORIENTED DESIGN

MACHINE LEARNING

SUPERVISED, UNSUPERVISED, REINFORCEMENT, NEURAL NETWORK, CAT

OPTIMISATION

FINANCE, LEAGUE OF LEGENDS, AMAZON WAREHOUSE

BOOLEAN SATISFIABILITY

x_1 OR x_2 OR $\bar{x}_3$ (SAT)
$\bar{x}_1$ OR $\bar{x}_2$ OR x_3
$\bar{x}_1$ OR $\bar{x}_2$ OR $\bar{x}_3$
$\bar{x}_1$ OR x_2 OR x_3

SUPER COMPUTING

COMPUTER GRAPHICS

COMPUTATIONAL SCIENCE

COMPUTATIONAL PHYSICS, NUMERICAL ANALYSIS, BIOINFORMATICS, COMPUTATIONAL CHEMISTRY

HACKING

ARTIFICIAL INTELLIGENCE

A.I.

ROBOTICS

COMPUTER VISION

FIND THE HUMANS

APPLICATIONS

VIRTUAL REALITY

SIMULATION

BIG DATA

NATURAL LANGUAGE PROCESSING

CHATBOTS
-HEY
-ARE U A ROBOT? HELLO THERE-
-YES- WAIT, I MEAN NO-
-PROVE IT! CALCULATING... I MADE YOU CAKE-

SCONES, TEA, FLOUR, BUTTER, BIRTHDAY, CAKE, FOOD, BACON, CELEBRATION, PANCAKE, BREAKFAST

KNOWLEDGE REPRESENTATION

TELEPRESCENCE

AUGMENTED REALITY

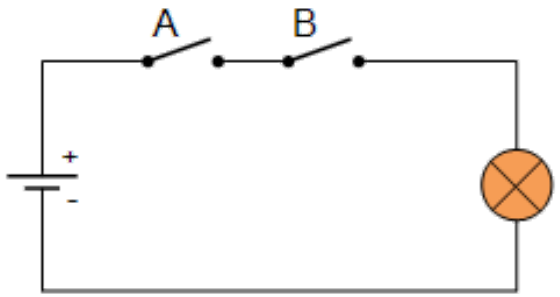
HUMAN COMPUTER INTERACTION

IMAGE PROCESSING

INTERNET OF THINGS

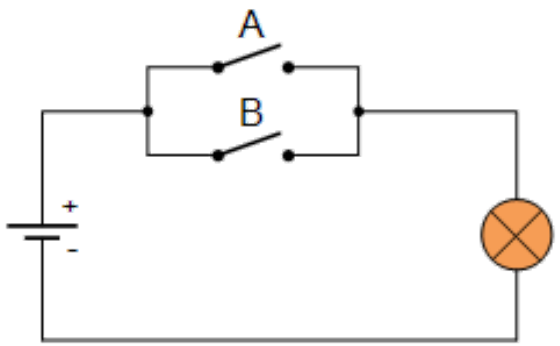
YOUTUBE
DOMAIN OF SCIENCE
MAP OF COMPUTER SCIENCE

BY DOMINIC WALLIMAN ©2017



Lamp – ON = "1"
Lamp – OFF = "0"

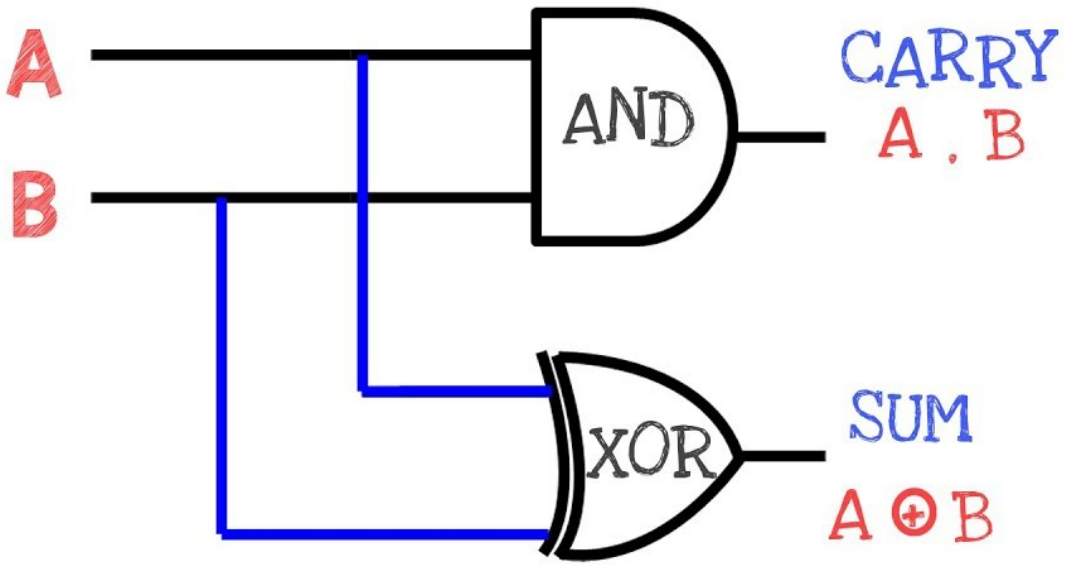
Switch A – Open = "0", Closed = "1"
Switch B – Open = "0", Closed = "1"

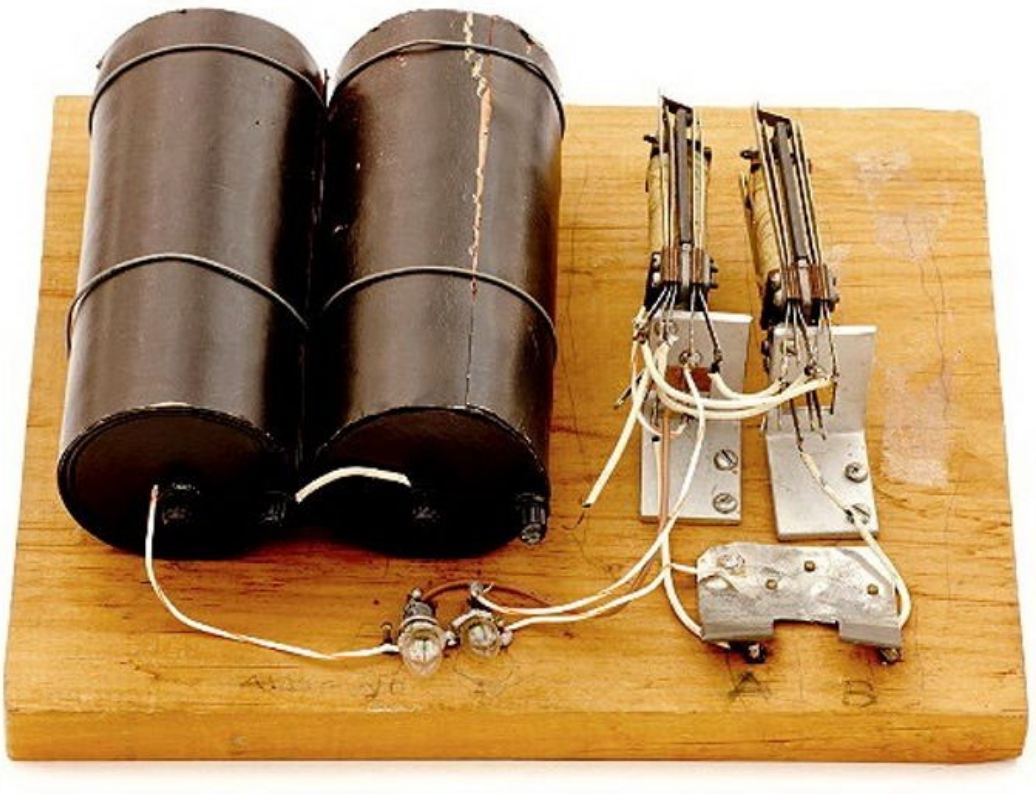


Lamp – ON = "1"
Lamp – OFF = "0"

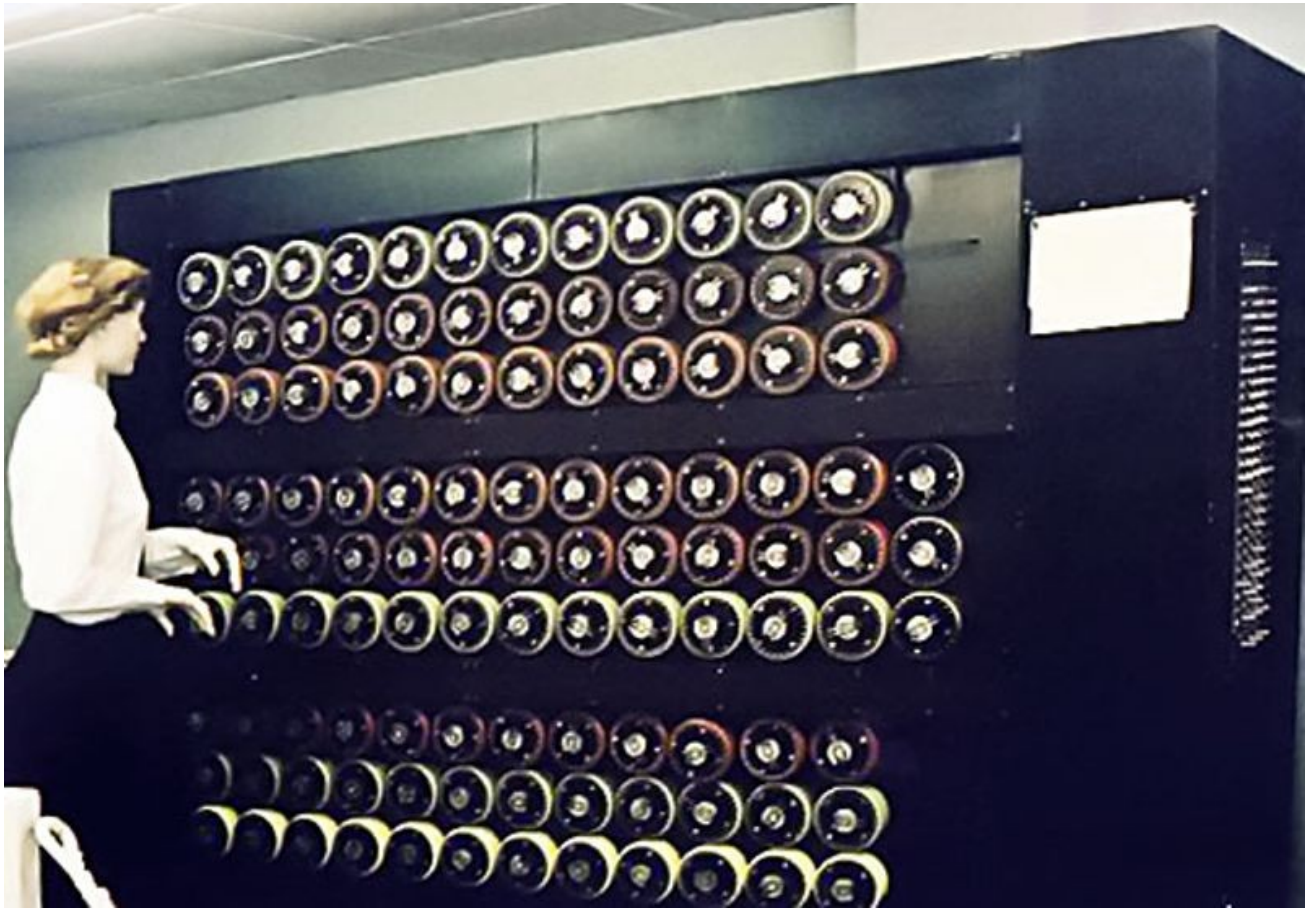
Switch A – Open = "0", Closed = "1"
Switch B – Open = "0", Closed = "1"

Half Adder

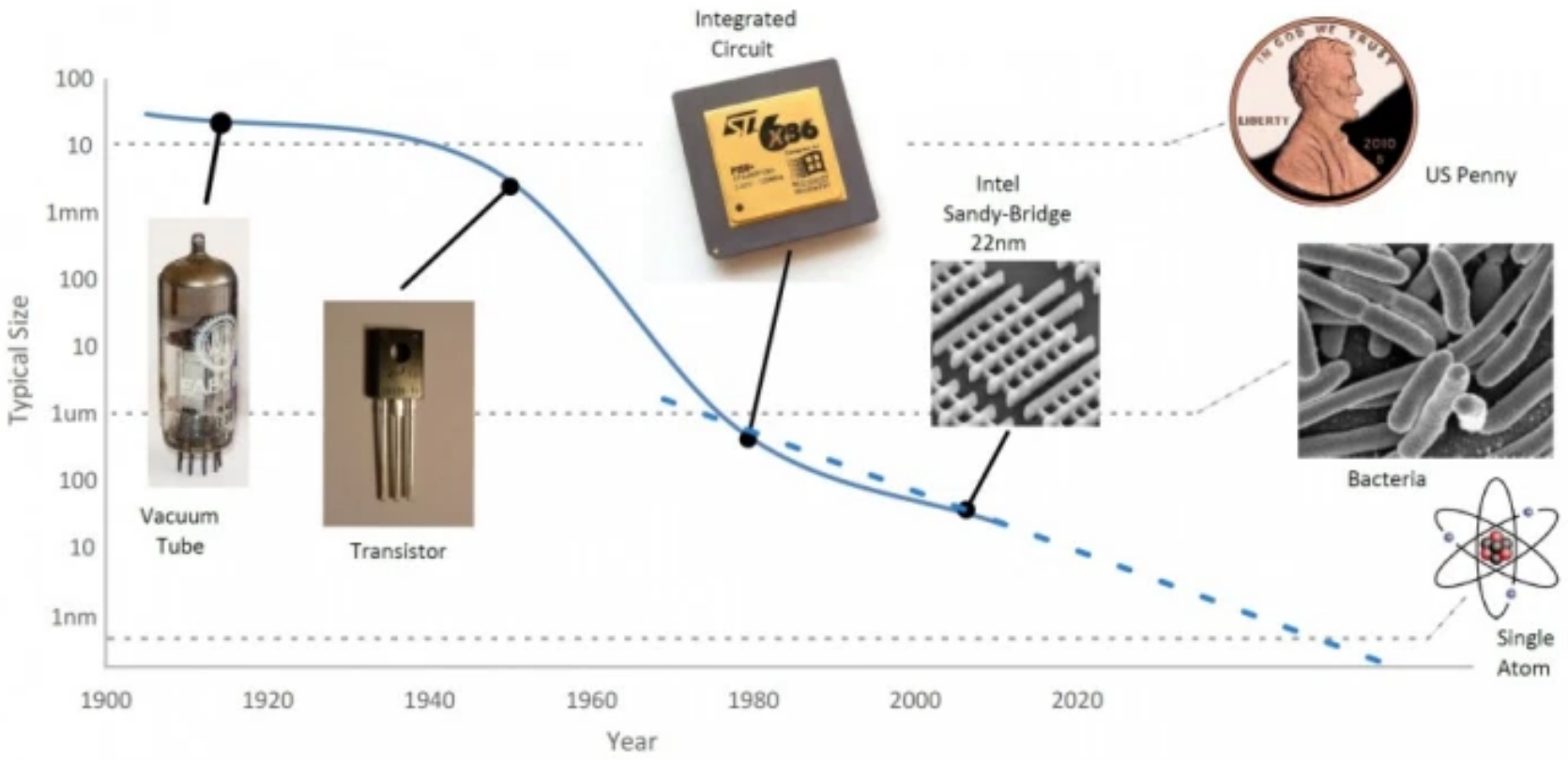




1937 - Adder



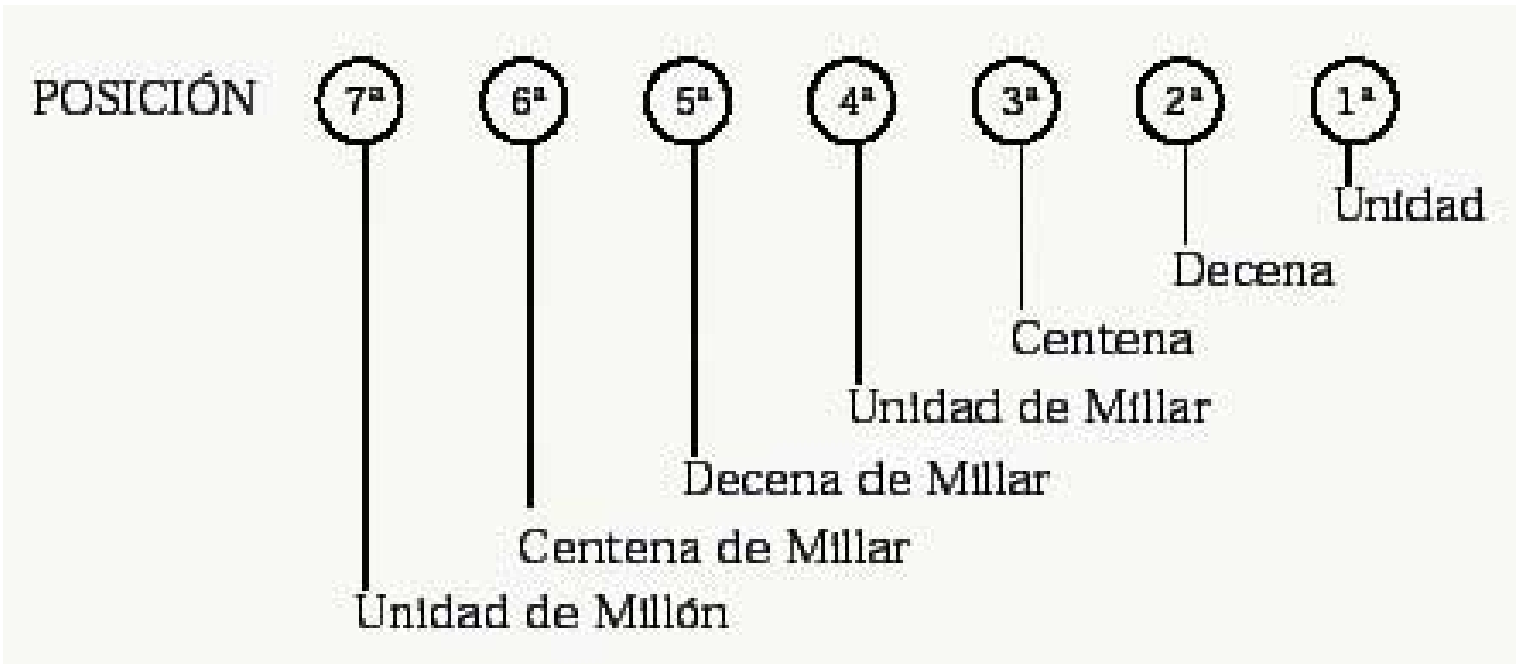
1937 - British Bombe
Alan Turing


Integrated
Circuit
US Penny
Intel
Sandy-Bridge
22nm
Bacteria
Vacuum
Tube
Transistor
Single
Atom
Typical Size
100
10
1mm
100
10
1um
100
10
1nm
1900
1920
1940
1960
1980
2000
2020
Year

Ley de Moore.

Números Binarios

```
// Sistema UNARIO

I I I I I I I

// utiliza un sólo tipo de símbolo. Su desventaja es que no permite simbolizar
// cómoda y rápidamente conjuntos con muchos elementos.

// Numeros ROMANOS

// Parecido al Unario, pero disponemos de más simbolos
// y operaciones implicitas

CXVII = cien + diez + cinco + uno + uno

MCMV = mil + (mil – cien) + cinco

// SISTEMAS POSICIONALES

// En estos sistemas, cada símbolo, además del número de unidades que representa
// considerado en forma aislada, tiene un significado o peso distinto
// según la posición que ocupa en el grupo de caracteres del que forma parte.
```


POSICIÓN
7ª
6ª
5ª
4ª
3ª
2ª
1ª
Unidad
Decena
Centena
Unidad de Millar
Decena de Millar
Centena de Millar
Unidad de Millón

Símbolos del sistema

Cada sistema utiliza sus propios símbolos:

- Decimal: 0, 1, 2, 3, 4, 5, 6, 7, 8 y 9.
- Binario: 0 y 1.
- Quinario: 0, 1, 2, 3 y 4.
- Octal: 0, 1, 2, 3, 4, 5, 6 y 7.
- Hexadecimal: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E y F.

$$N = \sum_{i=-k}^{n-1} d_i \cdot 10^i$$

$$N = \sum_{i=-k}^{n} d_i \cdot 2^i$$

Representaciones

Physical State	OFF	ON	ON	OFF	OFF	ON	ON	OFF
Binary Notation	0	1	1	0	0	1	1	0
Order of Magnitude	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Decimal Value	128	64	32	16	8	4	2	1
Applicable Value	0	64	32	0	0	4	2	0
Total Decimal Value	102 = 64 + 32 + 4 + 2							

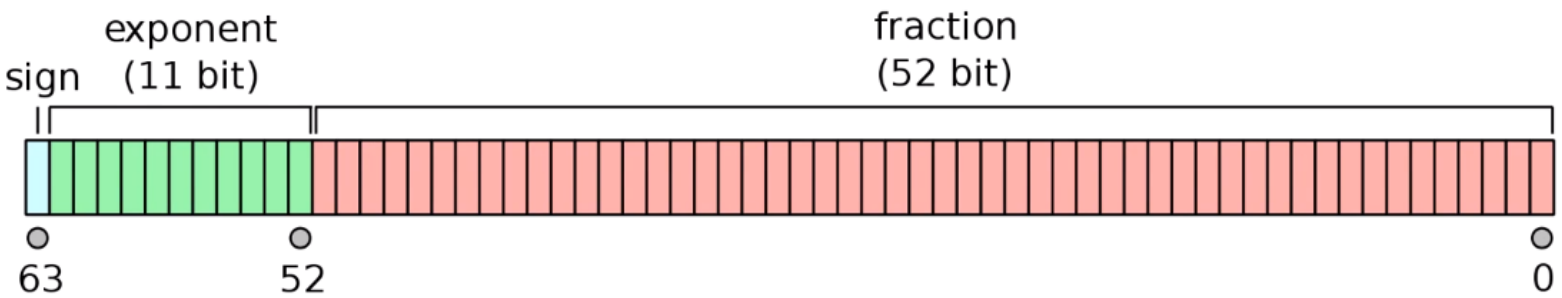
ASCII

Binary	Decimal	Glyph	Binary	Decimal	Glyph
0010 1110	46	.	0011 1010	58	:
0010 1111	47	/	0011 1011	59	;
0011 0000	48	0	0011 1100	60	<
0011 0001	49	1	0011 1101	61	=
0011 0010	50	2	0011 1110	62	>
0011 0011	51	3	0011 1111	63	?
0011 0100	52	4	0100 0000	64	@
0011 0101	53	5	0100 0001	65	A
0011 0110	54	6	0100 0010	66	B
0011 0111	55	7	0100 0011	67	C
0011 1000	56	8	0100 0100	68	D
0011 1001	57	9	0100 0101	69	E

Unicode

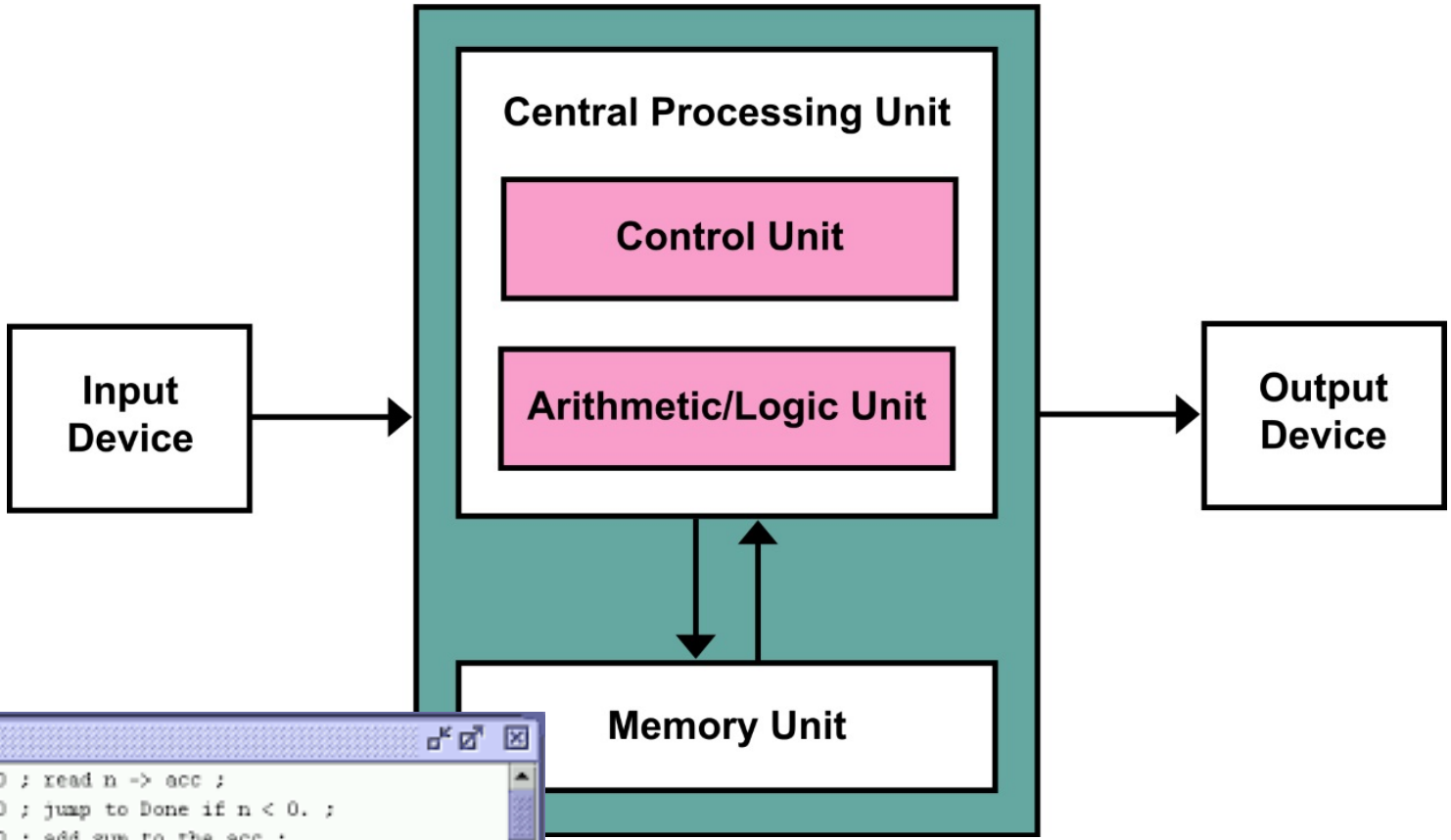
Number of bytes	Bits for code point	First code point	Last code point	Byte 1	Byte 2	Byte 3	Byte 4
1	7	U+0000	U+007F	0xxxxxxx			
2	11	U+0080	U+07FF	110xxxxx	10xxxxxx		
3	16	U+0800	U+FFFF	1110xxxx	10xxxxxx	10xxxxxx	
4	21	U+10000	U+10FFFF	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx

IEEE 754: Floating-Point Signed Double



https://www.h-schmidt.net/FloatConverter/IEEE754.html

Lenguaje de Máquina

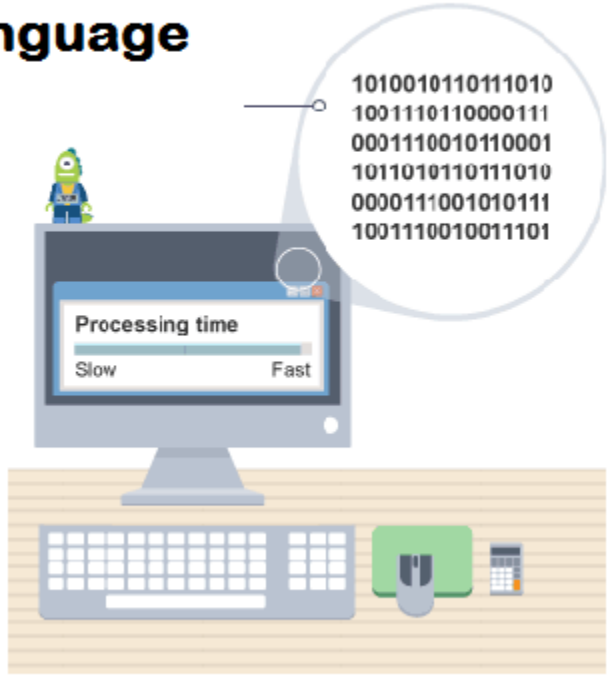


W1-0.ram

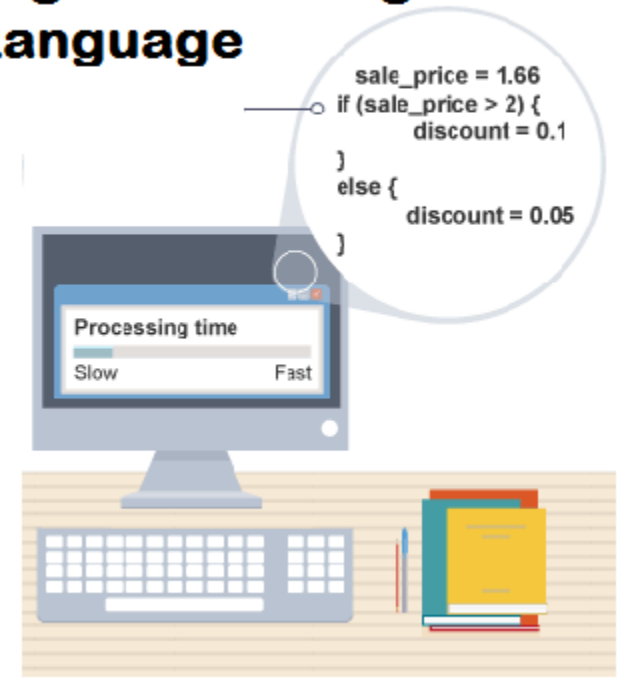
```
0011000000000000 ; read n -> acc ;
1011000000001010 ; jump to Done if n < 0. ;
0101000000010000 ; add sum to the acc ;
0010000000010000 ; store the new sum ;
1001000000000000 ; go back & read in next number ;
0001000000010000 ; load the final sum ;
0100000000000000 ; output the final sum ;
0000000000000000 ; stop ;
0000000000000000 ; 2-byte location where sum is stored ;
0000000000000000  ;
0000000000000000  ;
```

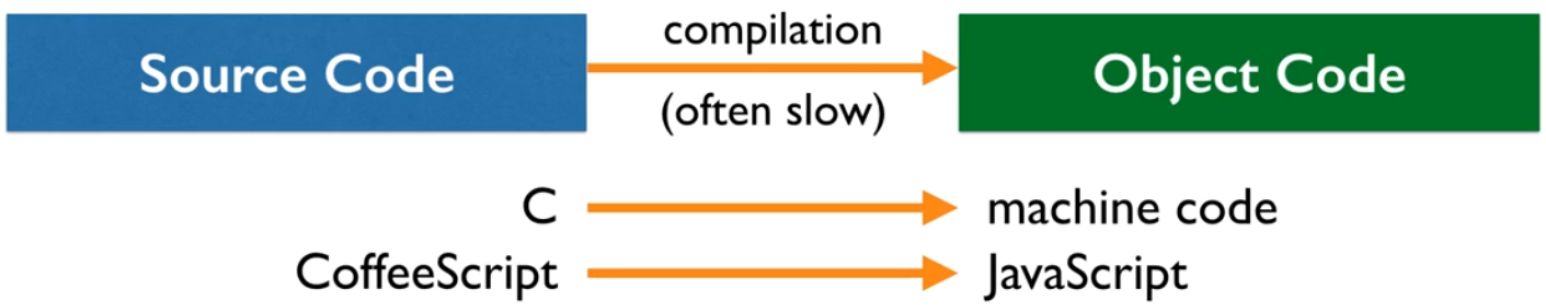
	op. code	source register	target reg.	destination reg.	shift amount	function type
instruction	000000	00001	00010	00110	00000	100000
hex	0x0	0x1	0x2	0x6	0x0	0x20
decimal	0	1	2	6	0	32
meaning	arithmetic	register 1	register 2	register 6	no offset	addition
English	*"take the value in register 1, add the value in register 2, place the result in register 6"*					
assembly	`add $t6, $t1, $t2`					

Low Level Programming Language



High Level Programming Language




Source Code
compilation
(often slow)
Object Code
C
machine code
CoffeeScript
JavaScript