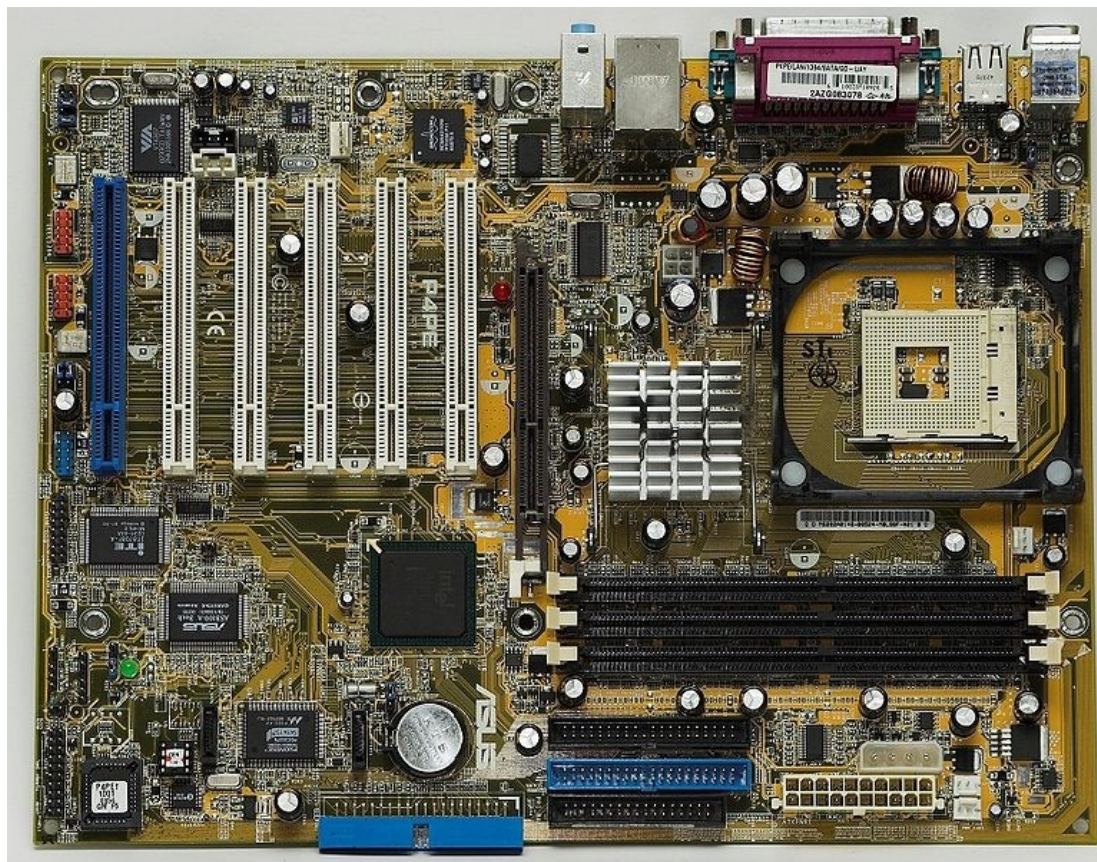


Introducción a los Computadores



-
1. Introducción
 2. Niveles de descripción de un computador
 3. Modelo Von Neumann
 4. Representación binaria
 5. Ejemplo arquitectura Von Neumann

1. Introducción

¿Qué es un computador?

[Carl Hamacher 96, Prof. de arquitectura del computador (Toronto)]:

“Un computador es una **máquina** de cálculo **electrónica** de alta velocidad que acepta información **digitalizada**, la **procesa** atendiendo a una lista de **instrucciones** que **almacena** internamente, y produce la correspondiente información de **salida**.”

- **Funciones** de un computador
 - Procesamiento de datos
 - Almacenamiento de datos
 - Transferencias de datos entre el computador y el exterior
 - Control de las anteriores operaciones
- La excesiva generalidad de estas funciones se debe a que la especialización funcional de un computador ocurre cuando se programa y no cuando se diseña.

1. Introducción

El computador **NO es una máquina que razone** de forma semejante al ser humano.

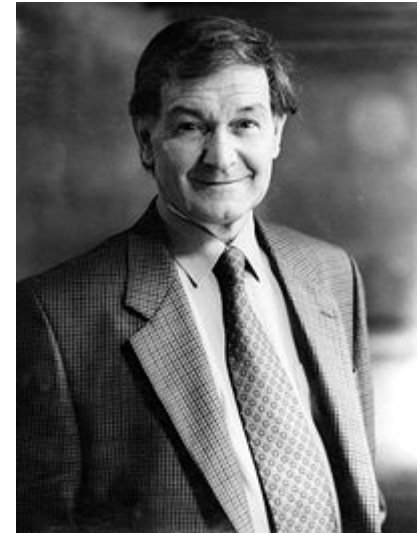
R2-D2 no existe



R2-D2 es un personaje de ficción
de
la Guerra de las Galaxias (1977)

Roger Penrose: "debe haber algo de naturaleza no computable en las leyes físicas que describen la actividad mental. La mente y el cerebro son dos entidades separables"

Penrose sugiere: "ninguna máquina de computación podrá ser inteligente como un ser humano, ya que los sistemas formales algorítmicos; o sea, los sistemas de instrucciones secuenciadas sobre los cuales están construidas las computadoras; **nunca les otorgarán la capacidad de comprender y encontrar verdades** que los seres humanos poseen"

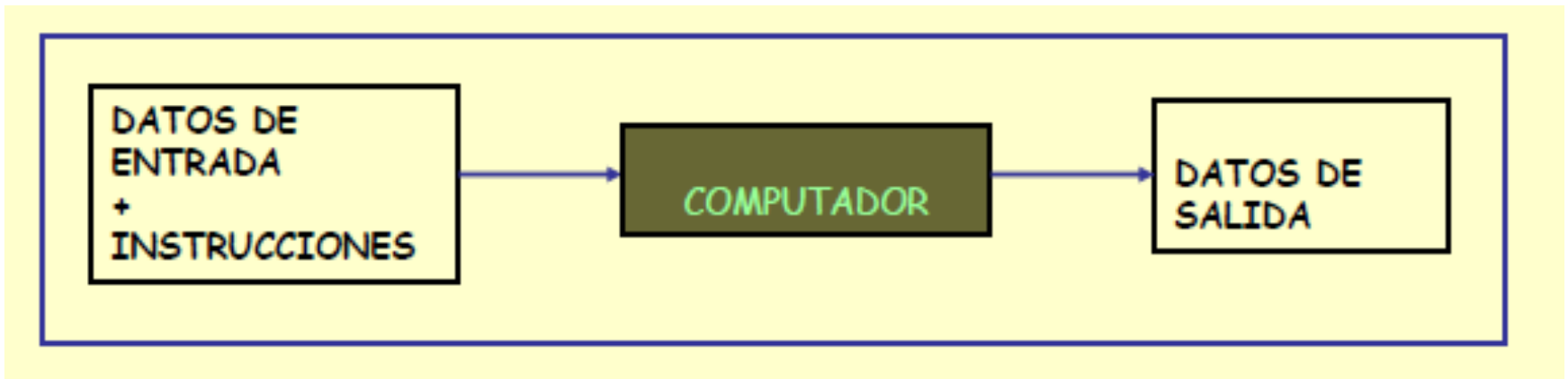


Roger Penrose
(físico-matemático Inglés)

1. Introducción

La **estructura básica** de una computadora incluye: **microprocesador** (CPU), **memoria** y **dispositivos de entrada/salida** (E/S), junto a los **buses** que permiten la comunicación entre ellos.

La computadora es una dualidad entre **hardware** (parte física) y **software** (parte lógica), que interactúan entre sí para una determinada función.



DATOS: elementos sobre los que actúan las instrucciones del programa

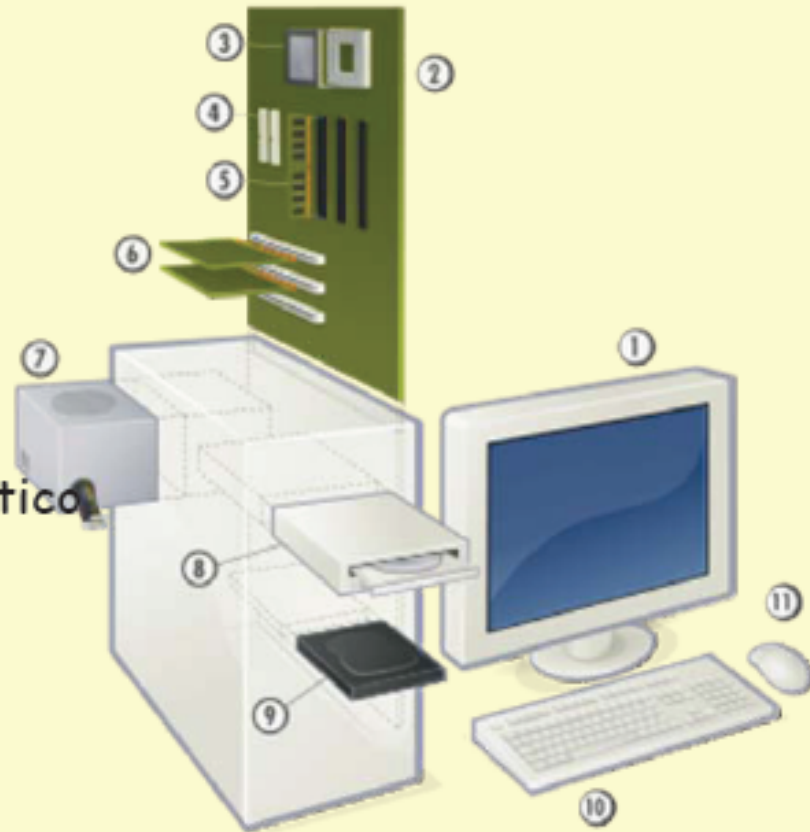
INSTRUCCIONES: indican al computador qué es lo que tienen que hacer con los datos

INSTRUCCIONES + DATOS EN LA **MEMORIA**. EL **PROCESADOR** RECOGE Y EJECUTA LAS INSTRUCCIONES

1. Introducción

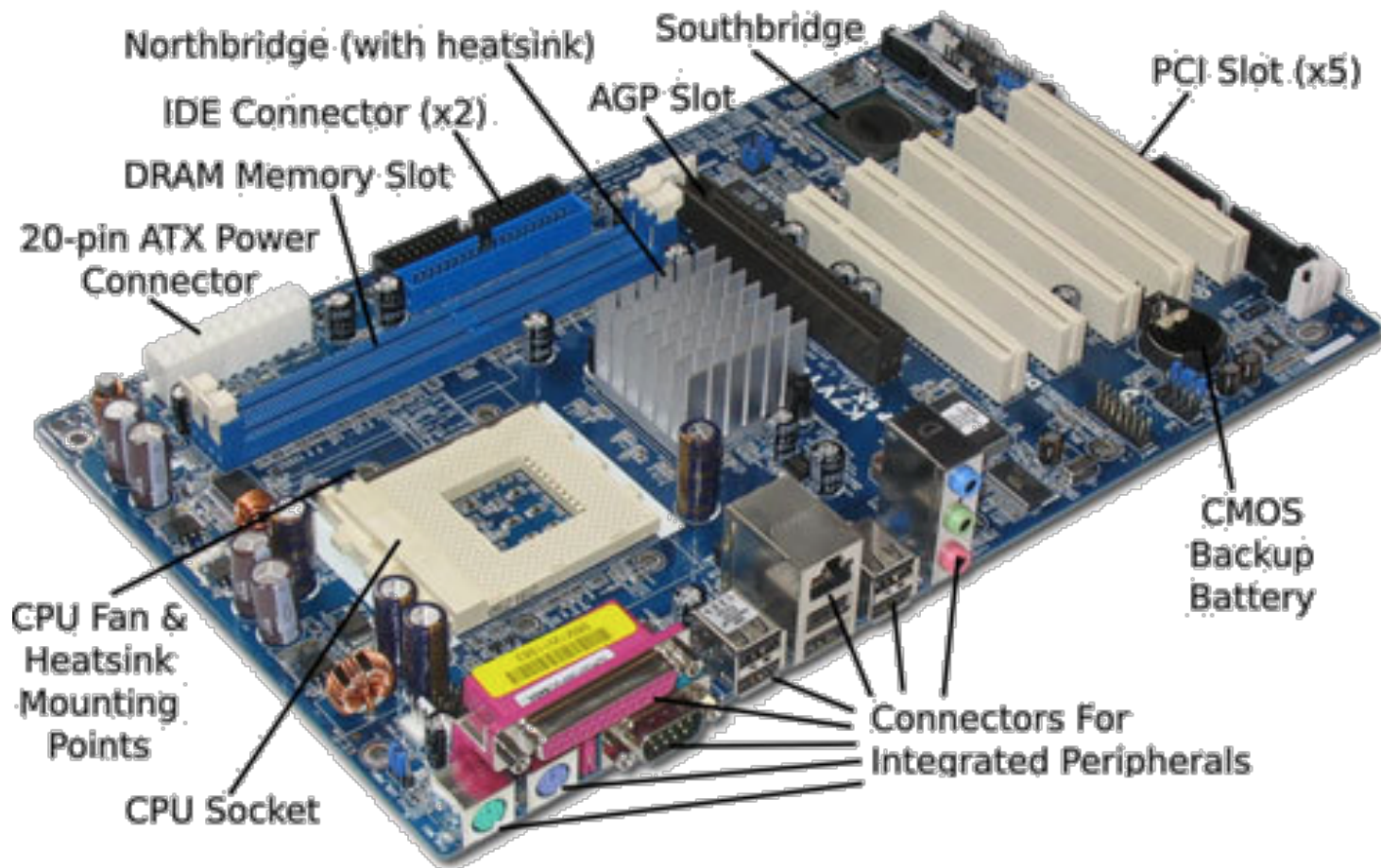
Elementos de una computadora

- 1: Monitor
- 2: Placa base
- 3: Procesador
- 4: Puertos ATA
- 5: Memoria principal (RAM)
- 6: Placas de expansión
- 7: Fuente eléctrica
- 8: Unidad de almacenamiento óptico
- 9: Disco duro
- 10: Teclado
- 11: Ratón



Placa Base

La "placa base": **Elemento principal de todo ordenador**, en el que se encuentran o al que se conectan todos los demás aparatos y dispositivos. Está formada por un circuito impreso que se asienta sobre múltiples capas de cobre aisladas entre sí mediante resina. Sobre las láminas de cobre se graban **fotoquímicamente** los circuitos. A estos circuitos se conectan los diversos elementos que constituyen su configuración. Las **empresas** que fabrican placas base siguen unas **pautas generales** para la ubicación de los conectores, zócalos de la CPU, los puertos, las conexiones del ventilados de la CPU, a estas pautas estándar se le denominan **factores de forma**. El factor de forma **ATX** (*Advanced Technology Extended*, para IBM y sus clones)/BTX domina el mercado actual de los PC.



Supercomputadora o Superordenador

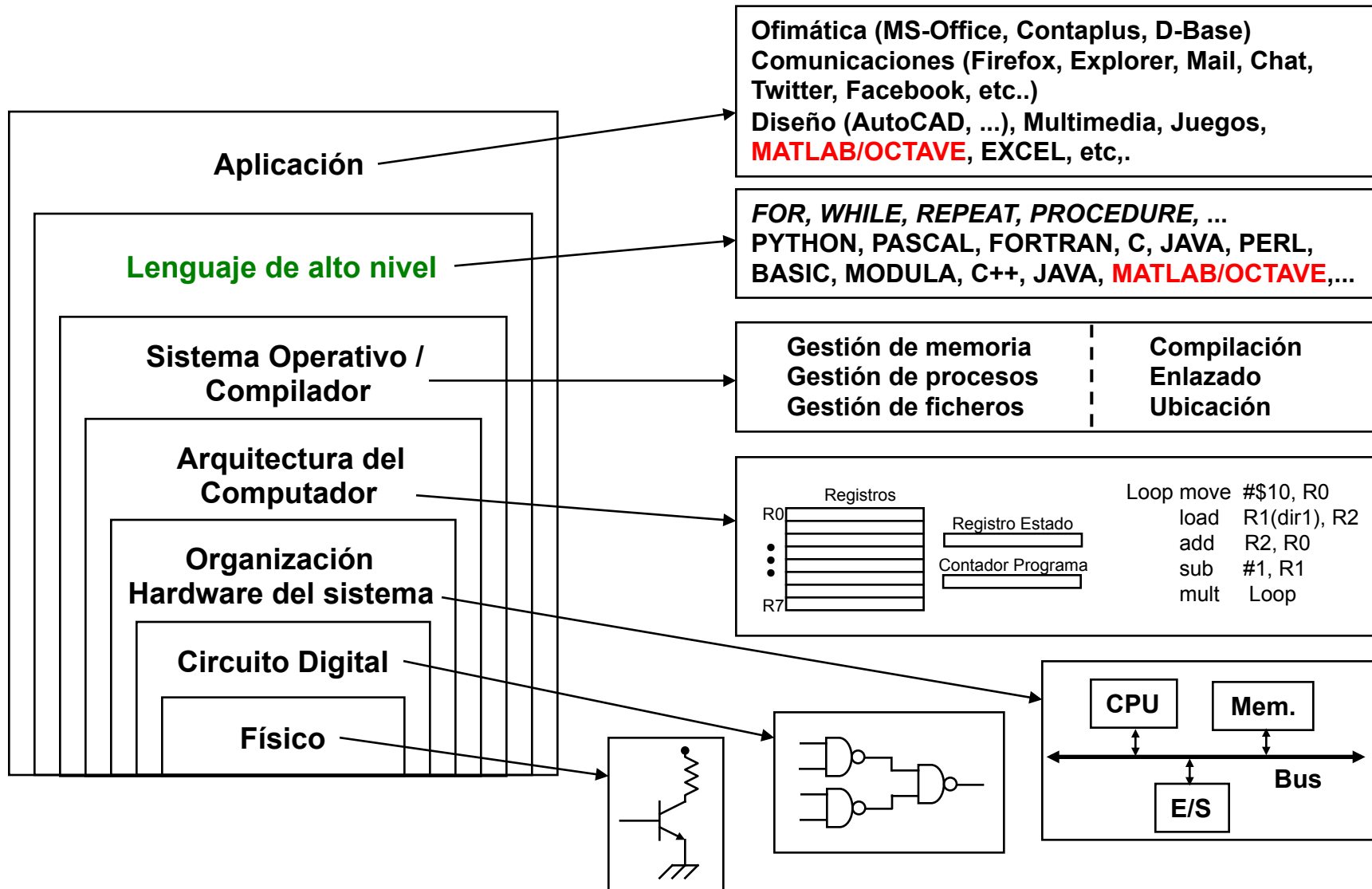
Computadora con capacidades de cálculo muy superiores a las comúnmente disponibles de las máquinas de escritorio de la misma época en que fue construida.

FLOPS (FLoating-point Operations Per Second; *tera*-FLOPS 10^{12} → *exa*-FLOPS 10^{18})

4 importantes tecnologías:

- La tecnología de registros vectoriales, creada por *Seymour Cray*, considerado el padre de la Supercomputación. Esta tecnología permite la ejecución de innumerables operaciones aritméticas en paralelo.
- El sistema conocido como M.P.P. por las siglas de *Massively Parallel Processors* o Procesadores Masivamente Paralelos, que consiste en la utilización de cientos y a veces miles de microprocesadores estrechamente coordinados.
- La tecnología de computación distribuida: los clusters de computadoras de uso general y relativo bajo costo, interconectados (a través del sistema M.P.I., *Message Passing Interface*) por redes locales de baja latencia y el gran ancho de banda.
- Cuasi-Supercómputo: Recientemente, con la popularización de la Internet, han surgido proyectos de computación distribuida en los que softwares especiales aprovechan el tiempo ocioso de miles de ordenadores personales para realizar grandes tareas por un bajo costo. A diferencia de las tres últimas categorías, el software que corre en estas plataformas debe ser capaz de dividir las tareas en bloques de cálculo independientes que no se ensamblaran ni comunicarán por varias horas. En esta categoría destacan *BOINC* y *Folding@home*.

2. Niveles de descripción de un computador



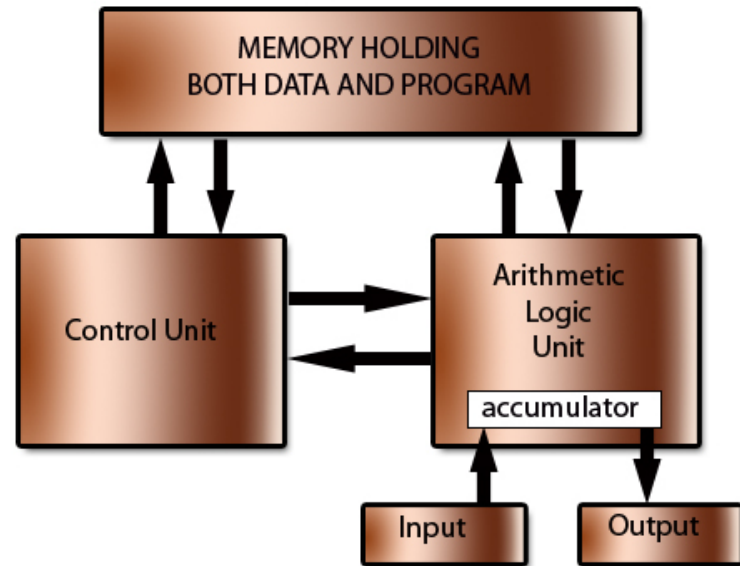
Un **lenguaje de alto nivel** se caracteriza por expresar el algoritmo de una manera adecuada a la **capacidad cognitiva humana**, en lugar de la capacidad ejecutora de las máquinas.

3. Modelo Von Neumann



John Von Neumann (1903-1957)
(Matemático húngaro)

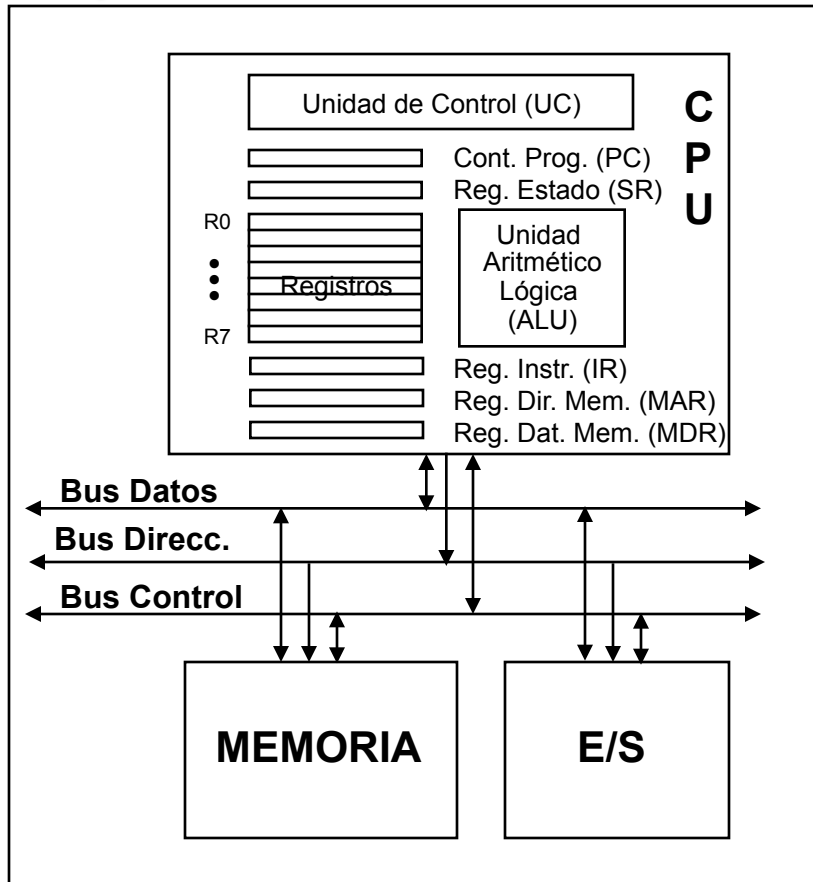
The Von Neumann or Stored Program architecture



(c) www.teach-ict.com

Arquitectura Von Neumann: arquitecturas de computadoras que utilizan el mismo dispositivo de almacenamiento tanto para las instrucciones como para los datos.

Esquema



Módulos básicos

CPU (Unidad Central de Proceso)

- Realiza la ejecución de las instrucciones

Unidad de Memoria

- Almacena las instrucciones y los datos

Unidad de E/S

- Transfiere información entre el computador y los dispositivos periféricos

Elementos de interconexión: BUSES (canales)

Bus de datos

- Para transferencia de datos entre la CPU y memoria o E/S

Bus de direcciones

- Para especificar la dirección de memoria o la dirección del registro de E/S

Bus de control

- Señales de control de la transferencia (reloj, lectura/escritura, etc.)

3. Modelo Von Neumann

Características principales del modelo Von Neumann

- Su funcionamiento se basa en el concepto de **programa almacenado en memoria**. La memoria principal almacena
 - **Instrucciones**: programa que controla el funcionamiento del computador
 - **Datos**: datos que procesa y genera dicho programa
- **Las palabras** en memoria siguen una **organización lineal**
 - Todas las palabras de memoria tienen el mismo tamaño
 - No hay distinción explícita entre instrucciones y datos
- **La ejecución** de las instrucciones es **secuencial**
 - El secuenciamiento de las instrucciones es implícito, y viene determinado por el orden en que han sido almacenadas en la memoria. Este secuenciamiento sólo puede ser modificado por instrucciones específicas de salto
 - El **contador de programa (PC)** indica en cada instante cual es la siguiente instrucción a ejecutar
- Las fases que se distinguen en la ejecución de una instrucción son
 - **Búsqueda de la instrucción** en memoria (*Fetch*) y cálculo de la direcc. de instrucción siguiente
 - **Descodificación** de la instrucción por parte de la CPU
 - **Búsqueda de los operandos** de la instrucción
 - **Ejecución** de la instrucción
 - **Escritura** del resultado

3. Modelo Von Neumann (La CPU)

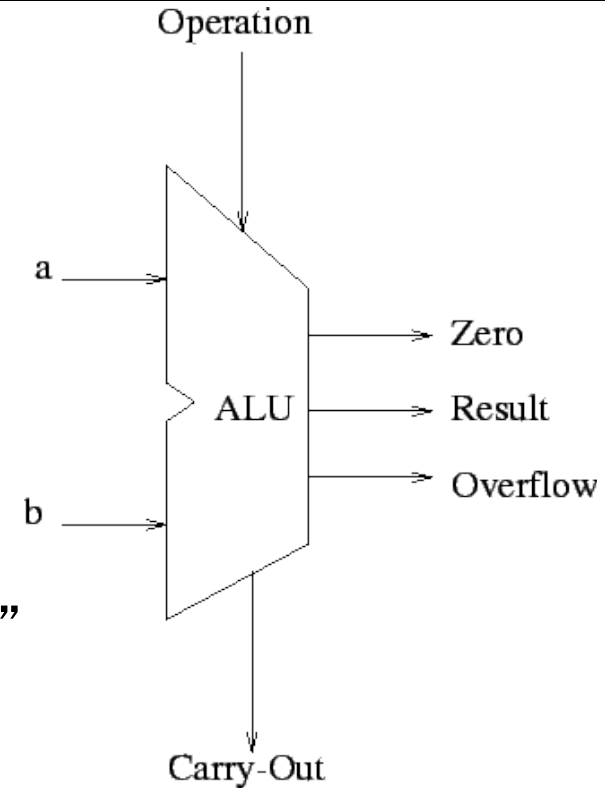
Unidad de proceso o ruta de datos (*data-path*)

- **Unidad Aritmético-Lógica (ALU)**
 - Realiza las **operaciones aritméticas y lógicas (NOT, AND, OR...)** que indican las instrucciones del programa
- **Banco de Registros**
 - Conjunto de registros visibles al usuario
 - Almacena los datos y los resultados con los que trabaja la ALU
- **Registros especiales**
 - Contador de programa, registro de estado, dirección del dato en memoria, etc.
- **Buses internos**
 - Caminos de interconexión entre los elementos anteriores

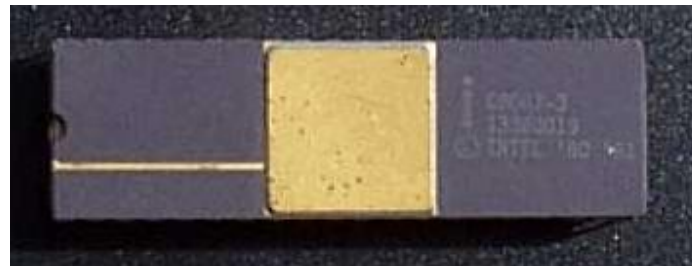
Unidad de control o ruta de control (*control-path*)

- **Genera las señales** necesarias para que la unidad de proceso ejecute las instrucciones de forma adecuada
- **Es un sistema secuencial.** Su complejidad depende de
 - La complejidad de la unidad de proceso y del número y tipo de instrucciones a ejecutar

- **Realiza las operaciones aritméticas y lógicas** que indican las instrucciones del programa
 - **Aritméticas**
 - *Enteras*
 - *Reales* -> Punto Flotante
 - **Lógica binaria**
 - *Comparación*
 - *Álgebra de Boole* (AND, OR, etc)
- La velocidad de cálculo se mide con “Benchmark”
 - MIPS: millones de instrucciones por segundo
 - MFLOPS: millones de instrucciones en punto flotante por segundo (*Floating point operations per second*)
- Con el 80386 aparece el coprocesador matemático 8087

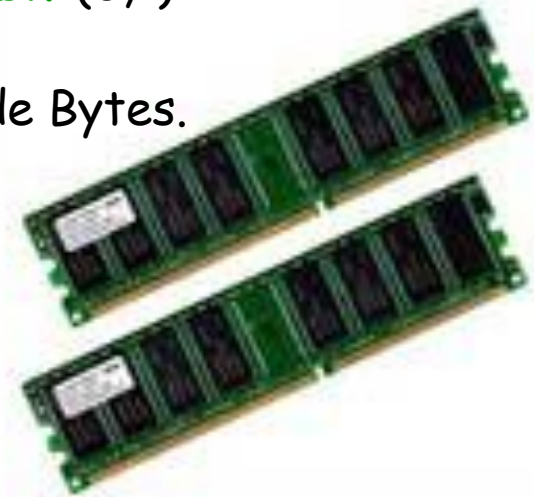


www.specbench.org



3. Modelo Von Neumann (La memoria)

- La **unidad elemental de información** digital es el **bit** (0,1) ← **binary digit**
- La **capacidad de almacenamiento** se mide en **Bytes** (1 byte → 8 *bits u octecto*) o múltiplos de Bytes.
 - 1 Byte = 8 bits (**octecto**)
 - 1 KB = 2^{10} B = 1024 B
 - 1 MB = 2^{20} B = 1024 KB = 1048576 B
 - 1 GB = 2^{30} B = $1.0737 \cdot 10^9$ B
 - 1 TB = 2^{40} B = $1.0995 \cdot 10^{12}$ B
- Longitud de Palabra:** N° de bits que se pueden leer con un solo acceso (8, 16, 32, 64, 128 bits).
- Algunas arquitecturas utilizan diferente longitud de palabra para los registros internos que para la memoria.
 - Pentium L=32/64
 - Algunas videoconsolas L=128
- La velocidad de acceso a los datos depende de la longitud de palabra y del tiempo de acceso (60, 70 ns)
- ROM, RAM, EPROM, EEPROM, FLASH**



Read-Only Memory Erasable-Programmable Read-Only Memory (Memoria ROM **no-volátil**) Electrically-Erasable Programmable Read-Only Memory (Memoria ROM **no-volátil**)

4. Representación binaria

Representación base binaria (Base 2)

- Sólo dispone de 2 símbolos: **0** y **1** (Decimal: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9)

Punto decimal

Base 10

$$d_m d_{m-1} \cdots d_1 d_0 . d_{-1} d_{-2} \cdots d_{-n}$$

$$d = \sum_{i=-n}^m 10^i \times d_i$$

Punto binario

Base 2

$$b_m b_{m-1} \cdots b_1 b_0 . b_{-1} b_{-2} \cdots b_{-n}$$

$$b = \sum_{i=-n}^m 2^i \times b_i$$

- Representa todos los números con combinaciones de esos dos:

$$0 = 0$$

$$1 = 1$$

$$2 = 10$$

$$3 = 11$$

$$4 = 100$$

$$5 = 101$$

$$6 = 110$$

$$7 = 111 \quad \leftarrow 2^3 - 1 = 7$$

...

- Rango Representable con n bits: $[0, 2^n - 1]$

4. Transformaciones números enteros

Transformaciones Binario-→Decimal y Decimal-→Binario

- **Binario-→Decimal:**

1º) ¿Qué representan los números que estamos acostumbrados a ver?

P.ej: $186_{10} = 1 \cdot 10^2 + 8 \cdot 10^1 + 6 \cdot 10^0$

2º) En cualquier otra base, la representación es igual, utilizando en lugar de 10 la base correspondiente

P.ej: $11010_2 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 16 + 8 + 2 = 26_{10}$

Ejemplos: 1101_2 , 10101_2

$$1101_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 8 + 4 + 1 = 13_{10}$$

$$10101_2 = 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 16 + 4 + 1 = 21_{10}$$

- **Decimal-→Binario:**

Se emplea el siguiente algoritmo (**Divisiones sucesivas**):

Tenemos *decimal* (*d*), buscamos *binario* (*b*)

While (cociente ≥ 1)

d/2

bit menos significativo de *b* = resto de división

end

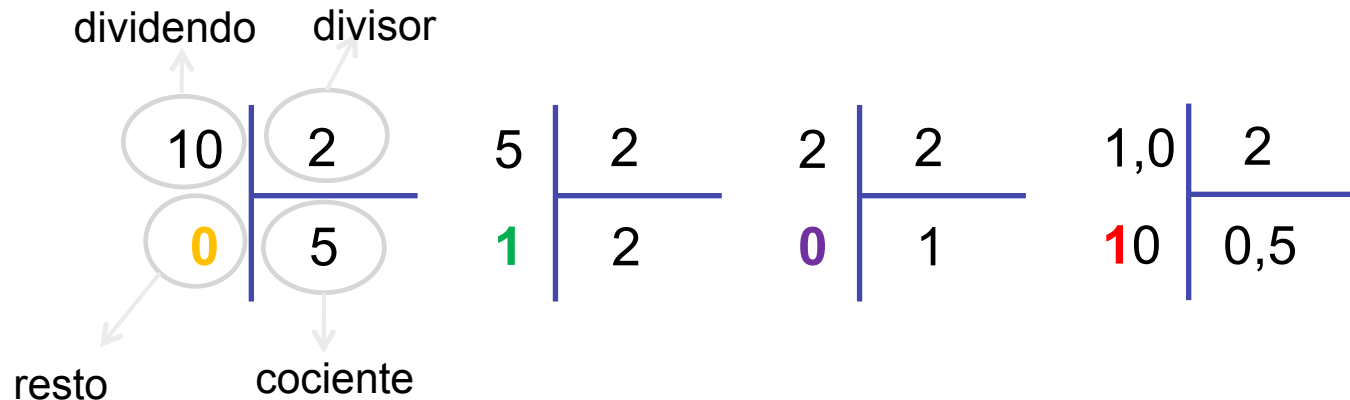
Ejemplos: 10_{10}

4. Transformaciones números enteros (Decimal $\rightarrow$ Binario)

Ejemplos: 10_{10}

$$10_{10} = 1010_2$$

Manualmente:



Con el ordenador:

divisor cociente resto

10 = 2x5 + 0; $b_0 = 0$

5 = 2x2 + 1; $b_1 = 1$

2 = 2x1 + 0; $b_2 = 0$

1 = 2x0 + 1; $b_3 = 1$

Con **MatLab**:

cociente = **fix**(10/2)

resto = **rem**(10,2)

4. Otra forma de transformar números enteros (Decimal → Binario)

$$156_{10} \\ = 10011100_2$$

8-dígitos en base-2 → $2^7 \ 2^6 \ 2^5 \ 2^4 \ 2^3 \ 2^2 \ 2^1 \ 2^0$

128 64 32 16 8 4 2 1

156 = **1 0 0 1 1 1 0 0**

1^{er} paso: ¿128 entra en el 156? Si=1, No=0

2^{ndo} paso: restamos 128 a 156 (28) y nos preguntamos si 64 entra en el 28. Si=1, No=0.

etc...

$$\begin{array}{r} -128 \\ \hline 28 \\ -16 \\ \hline 12 \\ -8 \\ \hline 4 \\ -4 \\ \hline 0 \end{array}$$

4. Transformaciones números no enteros

Transformaciones Binario→Decimal y Decimal→Binario

- **Binario- >Decimal:**

1º) ¿Qué representan los números que estamos acostumbrados a ver?

P.ej: $186.75_{10} = 1 \cdot 10^2 + 8 \cdot 10^1 + 6 \cdot 10^0 + 7 \cdot 10^{-1} + 5 \cdot 10^{-2}$

P.ej: $12.34_{10} = 1 \cdot 10^1 + 2 \cdot 10^0 + 3 \cdot 10^{-1} + 4 \cdot 10^{-2}$

2º) En cualquier otra base, la representación es igual, utilizando en lugar de 10 la base correspondiente

P.ej: $10.11_2 = 1 \cdot 2^1 + 0 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} = 2 + 0.5 + 0.25 = 2.75_{10}$

P.ej: $101.11_2 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} = 4 + 0 + 1 + 0.5 + 0.25 = 5.75_{10}$

Ejemplos: 0.01_2 , 1.101_2

$$0.01_2 = 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = 0.25_{10}$$

$$1.101_2 = 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = 1 + 0.5 + 0.125 = 1.625_{10}$$

- **Decimal- >Binario:**

Se emplea el siguiente algoritmo (**Multiplicaciones sucesivas**):

Tenemos *decimal* (*d*), buscamos *binario*(*b*)

While (*d*>0)

d=*d**2

if (*d*>1): Siguiente bit tras el punto = 1

else: Siguiente bit tras el punto = 0

d=*d*-1

end

Ejemplos: 10.75_{10}

4. Transformaciones números no enteros (Decimal $\rightarrow$ Binario)

Ejemplos: 10.75_{10}

$$10.75_{10} = 1010.11_2$$

R_1 R_2

R_1 R_2

$R_1 = 10_{10}$

	divisor	cociente	resto
$10 = 2 \times 5 + 0;$			$b_0 = 0$
$5 = 2 \times 2 + 1;$			$b_1 = 1$
$2 = 2 \times 1 + 0;$			$b_2 = 0$
$1 = 2 \times 0 + 1;$			$b_3 = 1$

$R_2 = 0.75_{10}$

$$2 \times R_2 = 1.5 \quad d_1 = \text{int}(1.5) = 1$$

$$2 \times f_1 = 1.0 \quad d_2 = \text{int}(1.0) = 1$$

$$2 \times f_2 = 0.0 \quad \text{STOP (d=0)}$$

$$\text{p.e.} : (1.5 - 1.0) = 0.5$$

$$f_1 = \text{frac}(1.5) = 0.5$$

$$f_2 = \text{frac}(1.0) = 0.0$$

4. Aritmética

La **suma binaria** se realiza igual que en decimal:

Vamos a realizar la suma de 10 y 15:

$$10 = 1010$$

$$15 = 1111$$

La tabla para la suma es:

+	0	1
0	0	1
1	1	10

Ejemplo:

$$\begin{array}{r}
 111 \\
 1010 \\
 + 1111 \\
 \hline
 11001 = 25
 \end{array}$$

$$\begin{array}{r}
 \text{128} \quad \text{64} \quad \text{32} \quad \text{16} \quad \text{8} \quad \text{4} \quad \text{2} \quad \text{1} \\
 (25)_{10} = 0 \quad 0 \quad 0 \quad \boxed{1} \quad \boxed{1} \quad 0 \quad 0 \quad 1 \\
 \hline
 -16 \\
 \hline
 9 \\
 \hline
 -8 \\
 \hline
 1 \\
 \hline
 -1 \\
 \hline
 0
 \end{array}$$

5. Ejemplo computador Von Neumann

Características del computador:

$$2^0 + 2^0 + \dots + 2^7 = 255_{10}$$



- Computador de **8 bits**: Datos e instrucciones
- Rango de datos: $[0_{10}, 255_{10}] \rightarrow [00000000_2, 11111111_2]$
- **Memoria**: 16 bytes $\rightarrow$ 8 datos y 8 instrucciones
- 4 bits para direccionar la memoria
- 2 registros
- ALU: Suma, resta y comparación.
- Repertorio de instrucciones:

store	ST (000)	X,XXXX (Reg, Pos memoria)
load	LD (001)	X,XXXX
add	ADD (010)	X,X,X (R_1, R_2, R_3)
subtract	SUB (011)	X,X,X
compare	CMP (100)	X,X,X
- EJEMPLOS:
 - Suma
 - Comparación