



# *Fundamentos de Hardware*

2020

**CIFP RODOLFO UCHA PIÑEIRO**

# Fundamentos de Hardware

## Contenido

Tema 1: Introducción a los Sistemas Informáticos. Arquitectura de computadores .....	3
Introducción .....	3
Historia.....	3
Encapsulado.....	3
Esquema y estructura de un ordenador. Elementos funcionales y subsistemas .....	5
Ejecución de una instrucción .....	7
Buses: de control, de direcciones y de datos .....	8
Puerta Lógica .....	9
Ejemplo de secuencia de acciones en la ejecución de una instrucción .....	9
Memoria cache.....	12
Conversión de un número decimal a binario y binario a decimal .....	12
Suma de números Binarios.....	12
Resta de números binarios .....	13
Producto de números binarios .....	13
Actividades.....	14

## Tema 1: Introducción a los Sistemas Informáticos. Arquitectura de computadores

### Introducción

#### Historia

El primer procesador comercial, el Intel 4004, fue presentado el 15 de noviembre de 1971. Los diseñadores fueron Ted Hoff y Federico Faggin de Intel, y Masatoshi Shima de Busicom (más tarde ZILOG).



Ilustración 1: Microcontrolador Zilog Z8 Super-8

Los microprocesadores modernos están **integrados por** millones de **transistores** y otros componentes empaquetados en una cápsula cuyo tamaño varía según las necesidades de las aplicaciones a las que van dirigidas, y que van desde el tamaño de un grano de lenteja hasta el de casi una galleta. Las partes lógicas que componen un microprocesador son, entre otras: unidad aritmético-lógica, registros de almacenamiento, unidad de control, Unidad de ejecución, memoria caché y buses de datos control y dirección.

Existen una serie de fabricantes de microprocesadores, como **IBM, Intel, Zilog, Motorola, Cyrix y AMD.**

Llevó décadas para que llegáramos a los modelos actuales de procesadores. En realidad, demoramos algunos años para llegar también a la idea que tenemos hoy de cómo funciona un procesador. Antiguamente, los programas no eran compatibles con todos los modelos de computadoras, ya que ellos eran desarrollados específicamente para cada máquina.

#### Encapsulado

En manufactura de circuitos integrados e Ingeniería Electrónica, **el encapsulado es el resultado de la etapa final del proceso de fabricación de dispositivos con semiconductores**, en la cual un semiconductor o un circuito integrado; se ubica en una carcasa para protegerlo de daño físico, de la corrosión, evacuar el calor generado y a su vez permitirle la comunicación con el exterior mediante contactos eléctricos.

Simplificando, el encapsulado es la parte externa y visible de un microprocesador y tiene tres funciones básicas:

- I. Proteger al núcleo de la oxidación y cualquier elemento ambiental como el polvo.
- II. Enfriar el núcleo o ayudar a disipar el calor generado en él.
- III. Dar soporte a las patillas de conexión, pines o contactos (E/S)

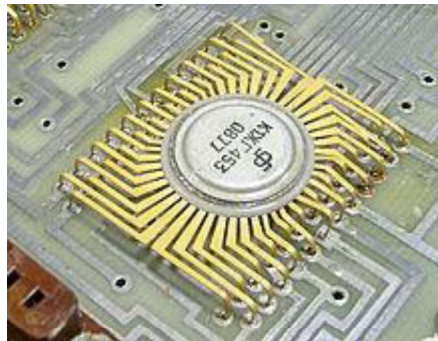


Ilustración 2: Encapsulado de antiguo circuito integrado fabricado en la URSS.

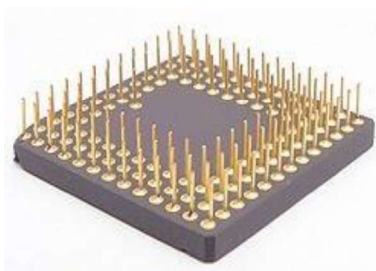
**Tipos de encapsulado** que podemos encontrarnos:

**PGA** (Pin grid array): El PGA es un tipo de empaquetado usado para los circuitos integrados, particularmente microprocesadores.

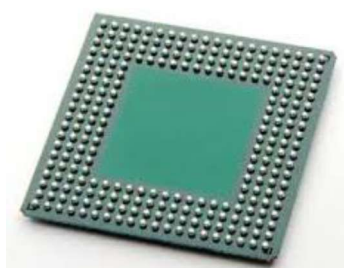
Originalmente el PGA consiste en un cuadrado de conectores en forma de agujero donde se insertan los pines del chip por medio de presión. Según el chip, tiene más o menos agujeros (uno por cada patilla).

**BGA**: Los terminales externos, en realidad esferas de soldadura, se sitúan en formato de tabla en la parte inferior del encapsulado.

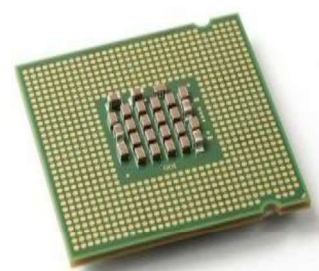
**LGA**: Es un encapsulado con electrodos alineados en forma de array (vector) en su parte inferior. Es adecuado para las operaciones donde se necesita alta velocidad debido a su baja inductancia. Además, en contraste con el BGA, no tiene esferas de soldadura por lo cual la altura de montaje puede ser reducida.



**PGA**



**BGA**

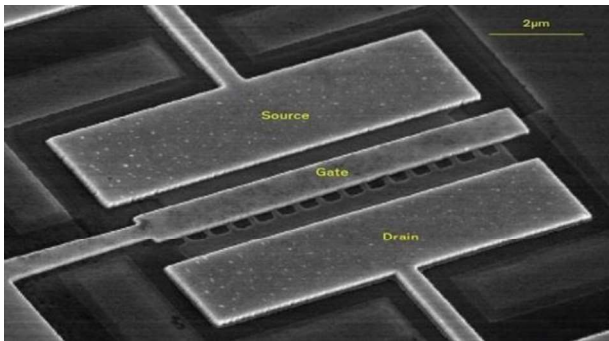


**LGA**

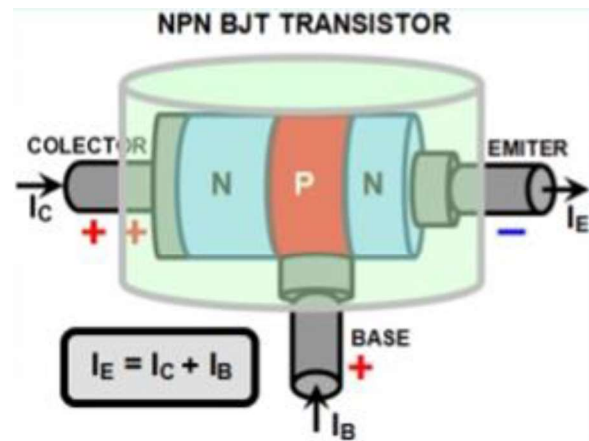
Los procesadores no existirían, al menos tan pequeños, si no fuera por los **transistores** (El transistor es un dispositivo electrónico semiconductor utilizado para entregar una señal de salida en



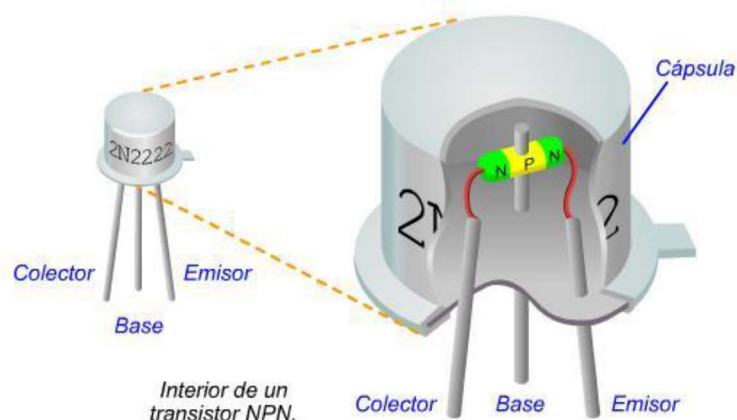
respuesta a una señal de entrada.). **Son la unidad básica por así decirlo, de todo procesador y circuito integrado.** Se trata de un **dispositivo semiconductor que cierra o abre un circuito eléctrico** o amplifica una señal. De esta forma podemos crear unos y ceros, el **lenguaje binario** que entiende la CPU.



Transistores presentes en un procesador



Esquema de un transistor (I)



Interior de un transistor NPN.

Esquema de un transistor (II)

## Esquema y estructura de un ordenador. Elementos funcionales y subsistemas

Un **ordenador** se divide fundamentalmente en dos partes: el **Hardware** y el **Software**.

- ✓ El **hardware** es la parte física del computador, la parte tangible; es decir aquello que podemos tocar del computador.
- ✓ El **software** es la parte lógica del computador, es decir el conjunto de instrucciones que le ordenan al **hardware** que tarea debe realizar.

La **arquitectura von Neumann**, también conocida como modelo de von Neumann o arquitectura *Princeton*, es una arquitectura de computadoras basada en la descrita en 1945 por el matemático y físico John von Neumann y otros. Este describe una arquitectura de diseño para un computador digital electrónico con partes que constan de una **unidad de procesamiento** que contiene una unidad aritmético-lógica y registros del procesador, **una unidad de control** que contiene un registro de instrucciones y un contador de programa, una **memoria** para almacenar tanto datos como instrucciones, almacenamiento masivo externo, y **mecanismos de entrada y salida**.

### Características:

- ✓ Una **memoria principal** que almacenaba tanto datos como instrucciones
- ✓ Una **unidad aritmético-lógica** que operaba con datos binarios
- ✓ Una **unidad de control** que interpretaba las instrucciones y generaba las señales para su ejecución.
- ✓ Un equipo de entrada y salida controlado por la unidad de control.

### Definiciones:

- ✓ **Unidad central de proceso (CPU):** Se compone de unidad de control, registros (elementos de memoria que contienen información relativa al programa que se está ejecutando y al control del propio procesador) y la unidad aritmético y lógica de un computador, esto es, el bloque que descodifica y ejecuta las instrucciones que se encuentran ubicadas en memoria a través de los Buses (camino a través de los cuales las instrucciones y los datos circulan entre las distintas unidades del ordenador).
- ✓ **Unidad aritmético y lógica (UAL o ALU):** Permite realizar una serie de operaciones elementales aritméticas y lógicas; tales como suma, resta, y lógico (and), o lógico (or), etc. Los datos sobre los que opera esta unidad provienen de la memoria principal y pueden estar almacenados de forma temporal en alguno de los registros de la propia unidad aritmético-lógica. La mayoría de las acciones de la computadora son realizadas por la ALU. La ALU toma datos de los registros del procesador. Estos datos son procesados y los resultados de esta operación se almacenan en los registros de salida de la ALU. Otros mecanismos mueven datos entre estos registros y la memoria.

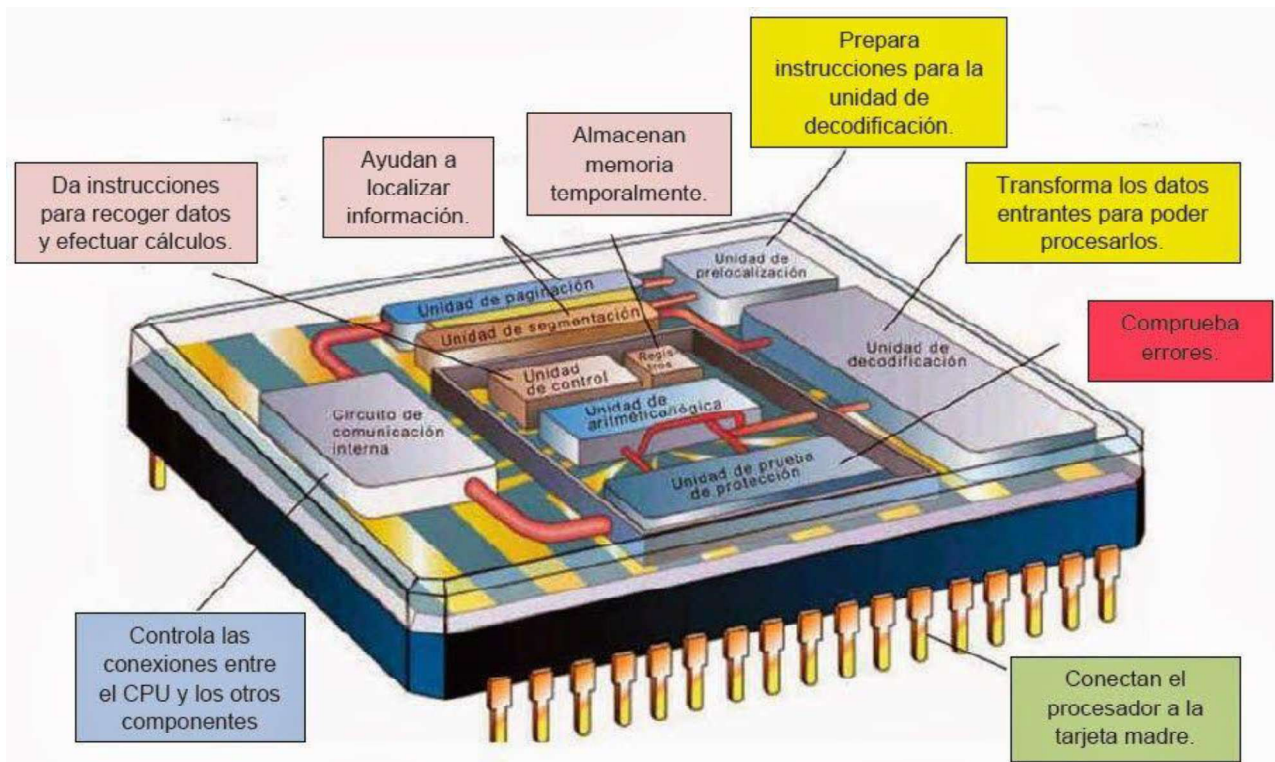


Ilustración 3: Estructura interna de un microprocesador

- ✓ **Unidad de control (UC):** Se encarga de leer, una tras otra, las instrucciones máquina almacenadas en la memoria principal y de generar las señales de control necesarias para que todo el ordenador funcione y ejecute las instrucciones leídas. Para conocer en todo momento la posición de memoria en la que está almacenada la instrucción que corresponde ejecutar, existe un registro apuntador llamado contador de programa, que contiene esta información.
- ✓ **Memoria principal:** Es una unidad dividida en celdas que se identifican mediante una dirección. Cada celda suele estar formada por un conjunto de puntos de memoria o bits que son el elemento básico de información, y cuyos valores, cero o uno, corresponden a estados de tensión bien diferenciados. Todas las celdas son del mismo tamaño (mismo número de bits) y se emplean para almacenar tanto datos como instrucciones de máquina.
- ✓ **Unidad de prueba de protección:** Unidad de prueba de protección (Protección Test Unit, PTU) Esta parte de la CPU trabaja con la unidad de control para vigilar si las funciones se realizan correctamente o no. Si detecta que algo no se ha hecho apropiadamente, genera una señal de error.

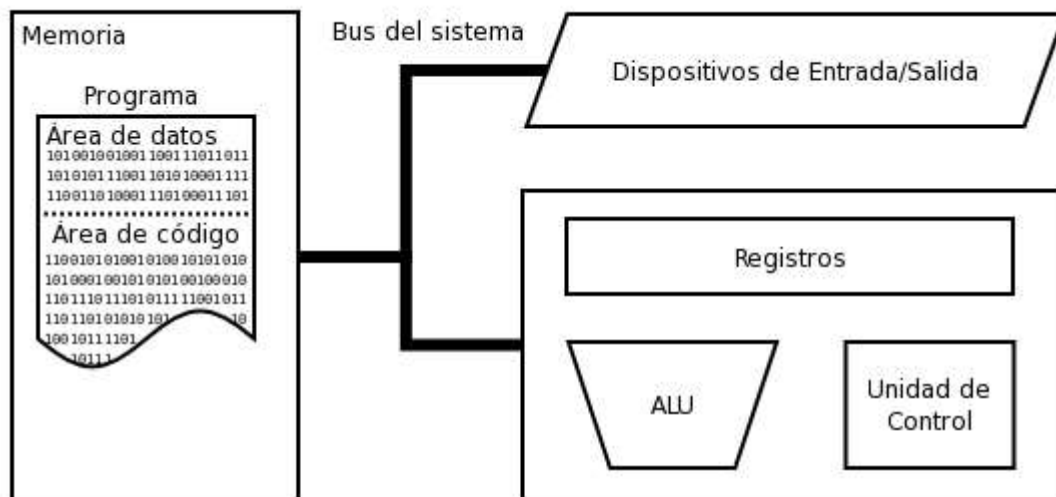


Ilustración 4: Dibujo esquemático del funcionamiento de un procesador

Nota: Lectura de interés

<https://www.profesionalreview.com/2019/06/30/partes-de-un-procesador/>

### Ejecución de una instrucción

El proceso de ejecución de una instrucción es el siguiente:

1. La **unidad de interfaz** con el bus lee la siguiente instrucción del programa y los datos asociados, que le llegan a través del FSB (Frontal Serial Bus).
2. La **unidad de decodificación** traduce la instrucción y se la pasa a la unidad de control para que decida qué hacer con ella.
3. Si la instrucción necesita ejecutar alguna operación matemática, se la pasa a la **ALU**.
4. La **ALU** realiza la operación y deja el resultado en un registro.

5. La **unidad de control** le pasa el resultado de la operación a la unidad de interfaz con el bus y le da la orden de guardarla en la memoria.
6. La **unidad de interfaz** con el bus escribe en la memoria **RAM** el resultado de la ejecución de la instrucción a través del **FSB**.

Este comportamiento se ejecuta de forma continua en un bucle hasta que se alcanza el final del programa.

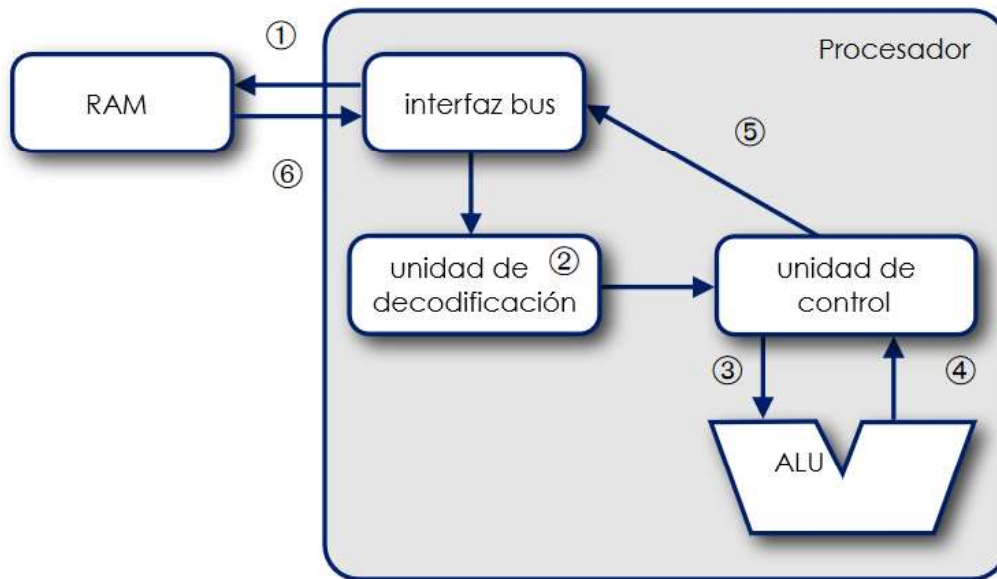


Ilustración 5: Dibujo esquemático del proceso de ejecución de una instrucción

### Buses: de control, de direcciones y de datos

En arquitectura de computadores, el bus (o canal) es un sistema digital que transfiere datos entre los componentes de una computadora. Está formado por cables o pistas en un circuito impreso, dispositivos como resistores y condensadores, además de circuitos integrados.

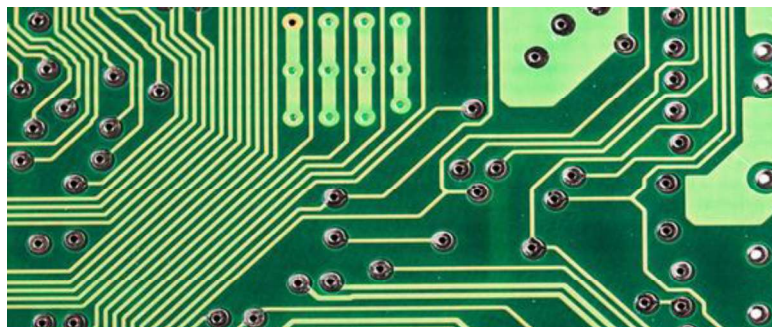


Ilustración 6: Pista impresas en una placa base.

#### Bus de control

El bus de control gobierna el uso y acceso a las líneas de datos y de direcciones. Como estas líneas están compartidas por todos los componentes, tiene que proveerse de determinados mecanismos que controlen su utilización.



### Bus de direcciones

La memoria RAM es direccionable, de forma que cada celda de memoria tiene su propia dirección. Las direcciones son un número que selecciona una celda de memoria dentro de la memoria principal o en el espacio de direcciones de la unidad de entrada/salida.

El bus de direcciones es un canal del microprocesador totalmente independiente del bus de datos donde se establece la dirección de memoria del dato en tránsito.

### Bus de datos

El bus de datos permite el intercambio de datos entre la CPU y el resto de las unidades.

Para ampliar (Funcionamiento de los buses):

[https://www.youtube.com/watch?v=4i5\\_07y5w6c&t=821s](https://www.youtube.com/watch?v=4i5_07y5w6c&t=821s)

### Puerta Lógica

Una puerta lógica es un bloque en la construcción de un circuito digital. La mayoría de las puertas lógicas tienen dos entradas junto a una salida y están basadas en álgebra booleana. En un momento dado, cada terminal está en una de las dos condiciones binarias: **false** (alto) o **true** (bajo). Falso representa 0, y verdadero representa 1. Dependiendo del tipo de puerta lógica que se utilice y de la combinación de entradas, la salida binaria será diferente.


Hay siete puertas lógicas básicas: **AND, OR, XOR, NOT, NAND, NOR** y **XNO**.

#### Puerta lógica AND

La puerta lógica Y, más conocida por su nombre en inglés AND, realiza la función booleana de producto lógico. Su símbolo es un punto ( $\cdot$ )

La ecuación característica que describe el comportamiento de la puerta es

$$S = A * B$$

Tabla de verdad	Función	Simbología															
<table border="1"> <tr> <th>a</th><th>b</th><th>S</th></tr> <tr> <td>0</td><td>0</td><td>0</td></tr> <tr> <td>0</td><td>1</td><td>0</td></tr> <tr> <td>1</td><td>0</td><td>0</td></tr> <tr> <td>1</td><td>1</td><td>1</td></tr> </table>	a	b	S	0	0	0	0	1	0	1	0	0	1	1	1	$S = a \times b$	
a	b	S															
0	0	0															
0	1	0															
1	0	0															
1	1	1															

### Ejemplo de secuencia de acciones en la ejecución de una instrucción

En este ejemplo vemos los pasos a realizar una suma de un valor contenido en el **registro acumulador (AC)** con otro que está en memoria:

Descripción de la operación		Operaciones con Registros
1	Llevar el <b>contenido del CP</b> al bus de direcciones	RD := CP

2	Introducir el contenido del bus de direcciones en el <b>registro de dirección</b> de memoria para descodificar la posición de la instrucción a ejecutar	
3	Hacer una <b>lectura en memoria</b> y cargar la instrucción	$RM := M[RD]$
4	Llevar el contenido del registro de memoria al bus de memoria.	$RI := RM$
5	El contenido del bus de memoria se introduce en el registro de instrucciones.	
6	Se incrementa el contador de programa quedando preparado para la ejecución siguiente.	$CP := CP+1$
7	A continuación, se hace una lectura en memoria del operando implicado (pasos 8-10)	
8	El contenido de la dirección del operando pasa al bus de direcciones.	$RD := RI.CDO$
9	El contenido del bus de direcciones se debe introducir en el registro de dirección de memoria para descodificar la posición del operando implicado en la suma.	
10	Lectura del operando direccionado	$RM := M[RD]$
11	El contenido del registro de memoria se debe llevar <b>al bus de memoria</b>	$RT := RM$
12	El contenido del bus de memoria se debe pasar al <b>registro temporal</b> para que acceda a la ALU.	
13	El contenido del acumulador debe entrar en la ALU para sumarse al operando llegado de memoria.	$AC := AC + RT$
14	Se debe indicar a la ALU la operación a realizar mediante una señal (señal suma).	
15	El resultado obtenido se debe guardar en el acumulador.	

Como comentamos anteriormente, **la memoria está formada por celdas, cada una de ellas con posibilidad de almacenar una información**. Para acceder a la celda de memoria se ha de hacer referencia a la dirección de la celda, bien para ver la información contenida en ella o para almacenar un dato en dicha celda.

El conjunto de 8 bits a los que se accede se denomina **byte, octeto o carácter**. A partir de aquí, la información se medirá como conjunto de bytes; es decir, como bloques de 8 bits.

Cada ordenador, agrupa estos bloques de 8 bits en lo que se denomina **palabra**. Cuando se dice que un ordenador es de 8, 16, 32 o 64 bits, nos estamos refiriendo al tamaño de los registros de la CPU y el tamaño de estos registros nos indica el tamaño de la palabra de ese ordenador.

La **longitud de la palabra** coincide con el número de bits que se transfieren entre las unidades del computador.

Así, un **Pentium IV** por ejemplo, que es un micro de 32 bits, usa una palabra de 32 bits, que agrupa 4 bytes.

Definiciones:

- ✓ **Instrucción de máquina:** una instrucción de máquina o instrucción-máquina es una operación elemental que un programa puede solicitar a un procesador para que la ejecute.

Una instrucción de máquina es pues una orden básica que el ordenador directamente puede interpretar y ejecutar, sin requerir ningún paso o traducción previos.

- La colección de instrucciones de máquina que puede interpretar y ejecutar un procesador, es lo que se denomina **su juego de instrucciones**.
  - Una instrucción completa de lenguaje de máquina contiene un **opcode** y, opcionalmente, **la especificación de uno o más operandos** - sobre los que la operación debe actuar.
- ✓ **Código de operación:** un *opcode* (operationcode) o *código de operación*, es la porción de una instrucción de lenguaje de máquina que especifica la operación a ser realizada.

**El direccionamiento** es una operación que se realiza cuando el procesador ejecuta o interpreta una instrucción. El modo de direccionamiento utilizado afecta directamente a la rapidez de ejecución de un programa. Para acceder a una dirección de memoria se pueden utilizar diferentes modos de direccionamiento:

- ✓ **Inmediato.** Se produce cuando las instrucciones contienen dentro sus propios datos, de modo que no se necesita acceder a la memoria para leerlo.
- ✓ **Directo.** Se produce cuando expresa la dirección real del objeto. Así la dirección de memoria 12000 se corresponde con la posición 12000 de memoria.
- ✓ **Indirecto.** Se produce cuando la dirección obtenida no es el objeto deseado, sino su dirección. Por tanto, para obtener el objeto deseado se requiere un acceso adicional a la memoria.
- ✓ **Relativo.** El direccionamiento se llama relativo cuando la dirección del dato que interviene en la instrucción se obtiene sumando a la dirección de la propia instrucción una cantidad fija, que normalmente está contenida en un registro de tipo especial.



## Memoria cache

Un **caché** es un tipo de memoria auxiliar de la que se puede recuperar a alta velocidad con una capacidad de almacenamiento relativamente pequeña. **Se encuentra entre la unidad central de procesamiento (CPU) y la memoria principal.**

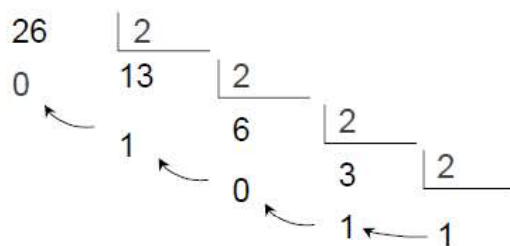
## Conversión de un número decimal a binario y binario a decimal

¿Cómo se pasa un número de base 10 a base 2?

1. Esta operación se realiza dividiendo el número de base 10 (dividendo) por 2 (divisor).
2. El cociente obtenido de la división se convertirá en dividendo, para volver a dividirlo por 2 (divisor).
3. Al nuevo cociente obtenido se le aplica la misma operación, y así sucesivamente hasta que aparezca un cociente igual a 0.

Si lo que se quiere hacer es la operación contraria, es decir, pasar de base 2 a base 10, procederemos multiplicando por potencias sucesivas de 2, empezando por  $2^0$  cada dígito binario de izquierda a derecha.

Pasar a binario el decimal 26



$$26_{10} = 11010_2$$

$$1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 16 + 8 + 0 + 2 + 0 = 26$$

## Suma de números Binarios

Las posibles combinaciones al sumar dos bits son

- $0 + 0 = 0$
- $0 + 1 = 1$
- $1 + 0 = 1$
- $1 + 1 = 10$

$$\begin{array}{r} 100110101 \\ + 11010101 \\ \hline 1000001010 \end{array}$$



Operamos como en el sistema decimal: comenzamos a sumar desde la derecha, en nuestro ejemplo,  $1 + 1 = 10$ , entonces escribimos 0 en la fila del resultado y *llevamos* 1 (este "1" se llama *arrastre*). A continuación, se suma el acarreo a la siguiente columna:  $1 + 0 + 0 = 1$ , y seguimos hasta terminar todas las columnas (exactamente como en decimal).

### Resta de números binarios

El algoritmo de la resta en binario es el mismo que en el sistema decimal. Pero conviene repasar la operación de restar en decimal para comprender la operación binaria, que es más sencilla. Los términos que intervienen en la resta se llaman minuendo, sustraendo y diferencia.

Las restas básicas 0-0, 1-0 y 1-1 son evidentes:

- $0 - 0 = 0$
- $1 - 0 = 1$
- $1 - 1 = 0$
- $0 - 1 =$  no cabe o se pide prestado al próximo.

La resta  $0 - 1$  se resuelve, igual que en el sistema decimal, tomando una unidad prestada de la posición siguiente:  $10 - 1 = 1$  y me llevo 1, lo que equivale a decir en decimal,  $2 - 1 = 1$ . Esa unidad prestada debe devolverse, sumándola, a la posición siguiente. Veamos algunos ejemplos:

Restamos  $17 - 10 = 7$

```

10001
-01010
-----
00111

```

Restamos  $217 - 171 = 46$

```

11011001
-10101011
-----
00101110

```

### Producto de números binarios

El algoritmo del producto en binario es igual que en números decimales; aunque se lleva cabo con más sencillez, ya que el 0 multiplicado por cualquier número da 0, y el 1 es el elemento neutro del producto.

Por ejemplo, multipliquemos 10110 por 1001:

```

  10110
*  1001
-----
  10110
 00000
 00000
 10110
-----
11000110

```

En sistemas electrónicos, donde se suelen utilizar números mayores, no se utiliza este método sino otro llamado **algoritmo de Booth**.