

## RESUMEN TEORIA (1 a 6):

### Clase 1

#### Tipos de datos:

Las computadoras manejan 4 tipos básicos de datos binarios

- Números enteros sin/con signo
- Números reales con signo
- Números decimales codificados en binario (BCD)
- Caracteres

#### Representación de números enteros:

- Sin signo
- Modulo y signo
- Complemento a uno (Ca1) (Complemento a la base reducida)
- Complemento a dos (Ca2) (Complemento a la base)
- Exceso

#### Números enteros sin signo:

Si el numero tiene N bits, puedo representar:

- $2^n$  = Números distintos.
- El rango va desde 0 a  $(2^n - 1)$

#### Sistemas Posicionales:

$N^{10}$  es el valor decimal de una cantidad expresada en base B y con  $(n+1+m)$  dígitos en posiciones i

$$N^o = \sum_{i=-m}^n (\text{dígito})_i \times (\text{base})^i$$

$$... + x_4 \times B^4 + x_3 \times B^3 + x_2 \times B^2 + x_1 \times B^1 + x_0 \times B^0 + x_{-1} \times B^{-1} + x_{-2} \times B^{-2} + ...$$

#### Números en punto fijo (1)

- Se considera que todos los números a representar tienen exactamente la misma cantidad de dígitos y la coma fraccionaria esta siempre ubicada en el mismo lugar.

- En sistema decimal: 0,23 o 5,12 o 9,11: En los ejemplos anteriores cada número tiene tres dígitos y la coma está a la derecha del más significativo

### Números en punto fijo (2)

- La diferencia principal entre la representación en el papel y su almacenamiento en computadora, es que no se guarda coma alguna, se supone que está en un lugar determinado.

### Punto Fijo: Rango y Resolución:

- **Rango:** diferencia entre el número mayor y el menor
- **Resolución:** diferencia entre dos números 19 consecutivos

### Operaciones aritméticas:

- **Suma en binario:** Al ser un sistema posicional la suma es como en decimal con acarreo entre posiciones al superar el máximo valor representable con un dígito.
- **Resta en binario:** Las restas se podrán realizar si acomodamos los operandos de modo tal que resultado sea mayor que cero, sino deberemos 'pedir prestado'.

### Bits de condición (banderas):

- Son bits que el procesador establece de modo automático acorde al resultado de cada operación realizada.
- Sus valores permitirán tomar decisiones como:
  - Realizar o no una transferencia de control.
  - Determinar relaciones entre números (mayor, menor, igual).

### Banderas aritméticas:


- Z(cero): vale 1 si el resultado de la operación son todos bits 0.
- C(Carry) en la suma vale 1 si hay acarreo del bit más significativo; en la resta vale 1 si hay "borrow" hacia el bit más significativo. CUANDO LA OPERACION INVOLUCRA NUMEROS SIN SIGNO, " C=1 " INDICA UNA CONDICION FUERA DE RANGO.

### Sistema Hexadecimal:

- Base 16.
- Dígitos: {0,1,2,3,4,5,6,7,8,9, A, B, C, D, E, F}

### Sistema hexadecimal codificado en binario (BCH):

- Los dígitos hexadecimales se convierten uno a uno en binario
- Para representar un dígito hexadecimal se utilizará siempre 4 bits
- Se asocia cada dígito con su valor en binario puro



Dígito hexadecimal	Código BCH
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Notas de Clase 1

### Sistema decimal codificado en binario (BCD):

- Los dígitos decimales se convierten uno a uno en binario.
- Para representar un dígito decimal se requerirán 4 bits
- Se asocia cada dígito con su valor en binario puro.



Dígito decimal	Código BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Notas de Clase 1

#### BCD: BCD tiene dos ámbitos de aplicación:

- E/S y periféricos, los números se codifican usando un byte por dígito. Se dice que el número está desempaketado
- En calculo, se reservan 4 bits por digito. Se dice que el numero está empaquetado.

#### BCD (Decimal codificado en binario) EJEMPLOS:

##### Desempaquetado SIN SIGNO:

- 834 = 11111000 11110011 11110100 = F8 F3 F4
- Por cada digito se usan 8 bits, 4 para el binario puro y 4 se completan con "1 "

##### BCD: Desempaquetado CON SIGNO:

- Con 4 bits hay  $2^4=16$  combinaciones posibles de unos y ceros:
- Diez usamos para los dígitos 0 al 9
- Nos quedan seis sin usar
- C= 1100 representa al SIGNO " + "
- D= 1101 representa al SIGNO " - "

### Ejemplo: desempaquetado con signo:


- $+ 834 = 1111 \text{ } 1000 \text{ } 1111 \text{ } 0011 \text{ } 1100 \text{ } 0100 = F8 \text{ } F3 \text{ } C4$
- Los 4 bits que acompañan AL ULTIMO DIGITO son reemplazados por el SIGNO.
- $- 834 = 1111 \text{ } 1000 \text{ } 1111 \text{ } 0011 \text{ } 1101 \text{ } 0100 = F8 \text{ } F3 \text{ } D4$

### Ejemplo; EMPAQUETADO CON SIGNO:

- $+ 834 = 1000 \text{ } 0011 \text{ } 0100 \text{ } 1100 = 834C$

### Suma en BCD (Decimal codificado en binario):

- Cuando la suma de los dos dígitos da  $>9$  hay que generar el “acarreo” porque hay seis combinaciones no usadas
- Entonces: cuando la suma de los dígitos es  $>9$  hay que sumar 6 en ese dígito
- Ejemplo:

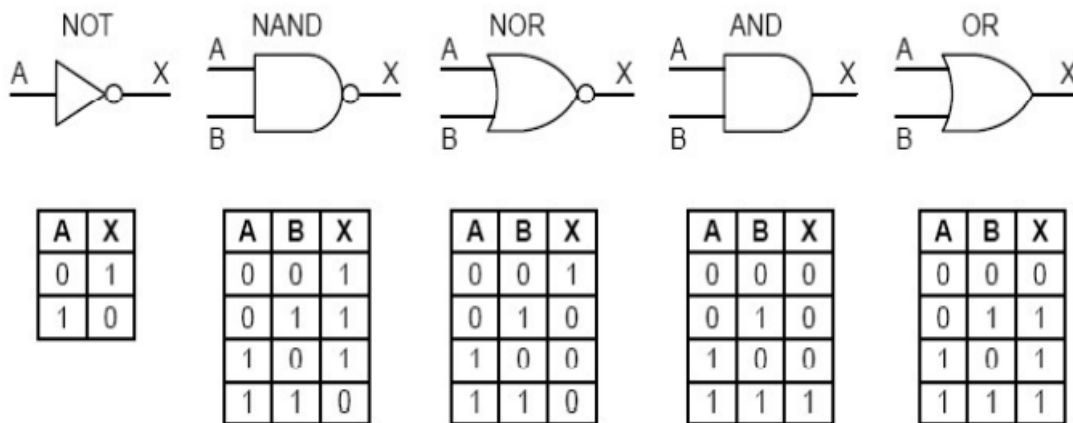
 Suma en BCD

1	111
+ 15	+ 0001 0101
27	0010 0111
42	111 1
	+ 0011 1100
	0110 ← 6
Resultado correcto	0100 0010

### Nivel de lógica digital:

- Un circuito digital es en el que están presentes dos valores lógicos
- Compuertas son dispositivos electrónicos que pueden realizar distintas funciones con estos dos valores lógicos
- Como vimos en el Ingreso las compuertas básicas son: AND, OR, NOT, NAND, NOR y XOR.

### Compuertas: símbolo y descripción funcional:



## Clase 2:

### Resumen: BCS

- El intervalo es simétrico
- El primer bit sólo indica el signo
- Los positivos empiezan con cero (0)
- Los negativos empiezan con uno (1)
- Hay dos ceros
- Números distintos:  $2^n$

### Complemento a uno (Ca 1):

- Si el número es positivo, los  $n$  bits tienen la representación binaria del número (como siempre)
- Si el número es negativo, los  $n$  bits tienen el Ca1 del valor deseado.
- El Ca1 de un número en base 2 se obtiene invirtiendo todos los bits
- Los positivos empiezan con cero (0)
- Los negativos empiezan con uno (1)
- El rango va desde  $-(2^{(n-1)} - 1)$  a  $+(2^{(n-1)} - 1)$  con dos ceros

### Complemento a dos (Ca2):

- Si el número es positivo, los  $n$  bits tienen la representación binaria del número (como siempre)
- Si el número es negativo, los  $n$  bits tienen el Ca2 del valor deseado.
- El Ca2 de un número (en base 2) se obtiene invirtiendo todos los bits (Ca1) y luego sumándole 1.
- Los positivos empiezan con cero (0)

- Los negativos empiezan con uno (1)
- El rango es asimétrico y va desde  $-(2^{(n-1)})$  a  $+(2^{(n-1)} - 1)$
- Hay un solo cero

### Resumen Ca2:

- El intervalo es asimétrico, hay un - más
- Los n bits representan al número
- Los positivos empiezan con cero (0)
- Los negativos empiezan con uno (1)
- Hay un solo cero
- Números distintos  $2^n$

### Nuevas banderas aritméticas:

- **N (negativo):** igual al bit más significativo del resultado. Es 1 si el resultado es negativo
- **V (overflow):** en 1 indica una condición de fuera de rango (desborde) en Ca2 (El resultado no se puede expresar con el número de bits utilizado).
  - Si sumamos dos números "+" y el resultado es "-" o si sumamos dos "-" y el resultado es "+" hay overflow, en otro caso no lo hay.
  - Si los números son de distinto signo nunca puede haber overflow
  - Si a un N "+" le restamos un N "-" y el resultado es "-" O si a un N "-" le restamos un "+" y el resultado es "+" hay overflow en la resta.
  - Si son del mismo signo (en la resta) nunca hay overflow.

### Clase 3:

#### Punto flotante:

- En el sistema de punto flotante el rango es mayor.
- Podemos representar números más grandes o más pequeños que en un sistema de punto fijo (para igual cantidad de bits), pero pagamos el precio que los Nos no están igualmente espaciados, como en punto fijo. (en punto flotante la resolución no es constante a lo largo del intervalo)

#### Normalización y Mantisas fraccionarias:

Con el objetivo de tener un único par de valores de mantisa y exponente para un número, se introduce la normalización

- Con el objetivo anterior, las mantisas fraccionarias se definen de la forma: 0,1DDDD..DD (Donde "D" puede ser 0 o 1).

- Cuando está NORMALIZADA la mantisa empieza SI O SI con 0,1 (El 1 siendo parte de la mantisa).
- **NOTA: EL BIT DE SIGNO NO VA DENTRO DE MI MANTISA.**

Formula punto flotante con mantisa fraccionaria:

$$0, M * B^E$$

### **Mantisa normalizada CON BIT IMPLICITO:**

Como todos los números comienzan con 0,1 ¿es necesario almacenar el 1?

En la representación de un numero normalizado con bit implícito, no se representa el bit implícito, pero si se tiene en cuenta para hacer los cálculos.

**Formula:**

$$0,1M * B^E$$

El “0,1” no forma parte de mi mantisa.

- En punto flotante, la resolución no es constante, sino que cambia según a que extremos nos acerquemos.

### **Error ABSOLUTO y Error RELATIVO:**

- El error ABSOLUTO(EA): es la diferencia entre el valor representado y el valor que QUIERO representar.
- El error RELATIVO es la división entre el EA y el número que quiero representar. EA/Numero a representar.

### **Estándar IEEE 754:**

**Simple precisión:**

- El estándar IEEE-754 para la representación en simple precisión de números en coma flotante exige una cadena de 32 bits. El primer bit es el bit de signo (S), los siguientes 8 son los bits del exponente (E) representado en exceso  $2^{(n-1)-1}$  y los restantes 23 son la mantisa (M):



### Doble precisión:

- El estándar IEEE-754 para la representación en doble precisión de números en coma flotante exige una cadena de 64 bits. El primer bit es el bit de signo (S), los siguientes 11 son los bits del exponente (E) representado en exceso y los restantes 52 son la mantisa (M)

### Cuadro IMPORTANTE DE IEEE-754:

IEEE 764			
Cálculo con 32 bits			
Exponente	Mantisa	Signo	Valor
$0 < E < 255$	Cualquiera	1	$-1 * 2^{E-127} * 1,M$ donde M es la mantisa
$0 < E < 255$	Cualquiera	0	$2^{E-127} * 1,M$ donde M es la mantisa
255	No nulo	Cualq.	NaN ("Not a number"). Se aplica para señalar varias condiciones de error.
255	0	1	-Infinito
255	0	0	Infinito
0	No nulo	1	$-(2^{-126}) * (0,M)$ (Números sin normalizar)
0	No nulo	0	$2^{-126} * (0,M)$ (Números sin normalizar)
0	0	1	-0
0	0	0	0

### Clase 4:

#### Circuitos Combinacionales o Combinatorios:

- Responden a los valores lógicos en las entradas, la salida está determinada exclusivamente por los valores de las entradas en ese instante.

- Si cambia la entrada, cambia la salida.
- Los valores pasados de las entradas no influyen en los valores de las salidas.

### **Circuitos Secuenciales:**

- Las salidas dependen tanto de las entradas como del estado interno del circuito.
- Tienen la característica de “almacenar” valores lógicos internamente.
- Estos valores se almacenan, aunque las entradas no estén.

### **Memoria:**

- Se puede construir con un flip-flop una memoria de 1 bit.
- Se llama biestable porque el circuito posee solo 2 estados posibles de funcionamiento, se queda en cada uno de ellos, salvo que las entradas provoquen un cambio.

### **Secuenciales – Clasificación:**

- Según la manera en que las salidas respondan a las señales lógicas presentes en la entrada, los biestables se clasifican en:
  - SR
  - J-K
  - D (sirve para almacenar)
  - T (sirve para )

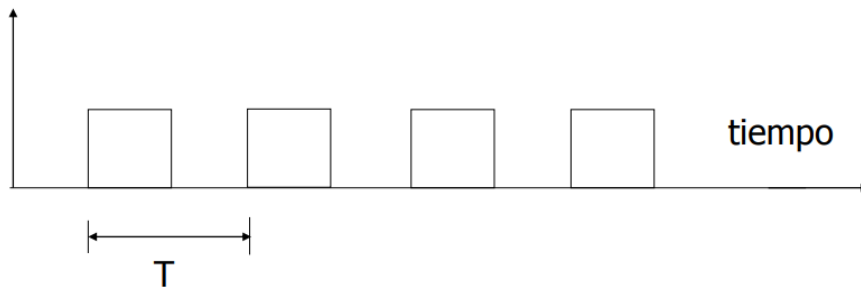
### **Respecto del instante en que pueden cambiar dichas salidas, pueden ser:**

- Asincrónicos: cuando en la entrada se establece una combinación, las salidas cambiarán
- Sincrónicos: la presencia de una entrada especial, determina “cuando” cambian las salidas acordes a las entradas

### **Reloj(Clock o CLK): “Señal especial”:**

- El orden en que ocurren los sucesos es importante.
- A veces los sucesos deben ocurrir simultáneamente.
- Reloj: es una señal de tiempo precisa que determina cuando se producen eventos.

# Reloj (Clock) (CLK)



Cada tiempo  $T$ , la señal se repite

Tablas de comportamiento:

Sincronizables				Asincronizables		
CLK	S	R	$Q_{n+1}$	S	R	$Q_{n+1}$
1	0	0	$Q_n$	0	0	$Q_n$
1	0	1	0	0	1	0
1	1	0	1	1	0	1
1	1	1	Prohibido	1	1	Prohibido
0	x	x	$Q_n$			

CLK	J	K	$Q_{n+1}$	CLK	D	$Q_{n+1}$
1	0	0	$Q_n$	1	0	0
1	0	1	0	1	1	1
1	1	0	1	0	x	$Q_n$
1	1	1	$\overline{Q_n}$			
0	x	x	$Q_n$			

CLK	$Q_{n+1}$
1	$\overline{Q_n}$
0	$Q_n$

$M \times 2^E$   
 $0, M \times 2^E$   
 $0, 1 M \times 2^E$

## Clase 5:

¿Qué es una computadora?:

- Máquina
- Digital
- Sincrónica
- Cálculo numérico
- Cálculo lógico

- Controlada por programa
- Comunicación con el mundo exterior

### **Arquitectura y Organización:**

- Arquitectura son aquellos atributos visibles al programador
  - Conjunto de instrucciones, número de bits usados para representación de datos, mecanismos de E/S, técnicas de direccionamiento.
- Organización es cómo son implementados
  - Señales de control, interfaces, tecnología de memoria

### **Estructura y Función:**

- Estructura es el modo en el cual los componentes se relacionan entre sí.
- Función es la operación de los componentes individuales como parte de la estructura.

### **Función:**

- Las funciones de todas las computadoras son:
  - Procesamiento de datos
  - Almacenamiento de datos
  - Movimientos de datos
  - Control

### **Modelo de Von Neumann:**

Consta de 5 componentes principales:

- Unidad de entrada: provee las instrucciones y los datos
- Unidad de memoria: donde se almacenan datos e instrucciones
- Unidad aritmético-lógica: procesa los datos
- Unidad de control: dirige la operación
- Unidad de salida: se envían los resultados

### **Aspectos más importantes de Von Neumann:**

- Utilización del sistema binario:
  - Simplifica la implementación de funciones.
  - Disminuye la probabilidad de fallos.
- Instrucciones y datos residen en memoria:
  - Ejecución del programa en forma secuencial.
  - Aumenta la velocidad.

- La memoria es direccionable por localidad sin importar el dato almacenado.

### ¿Qué es un ciclo de instrucción?

- El procesamiento requerido para una sola instrucción se llama ciclo de instrucción.
- Búsqueda y Ejecución:

El secuenciador de la UC emite una dirección que va al PC.

El PC apunta a una dirección que es donde se encuentra la instrucción a buscar.

-Una operación de lectura sobre la celda de memoria apuntada produce que el contenido de esta, es decir una instrucción, viaje a través del bus de datos hacia el procesador y se aloje en el registro de instrucciones.

-Desde el IR, la instrucción pasa a la UC donde es decodificada así el procesador sabe que se debe hacer.

-Luego, a través del bus de direcciones, se busca la celda de memoria en donde se encuentra el primero de los operandos. Luego, a través del bus de datos, el operando se ubica en el registro temporal 1 de la ALU

-Lo mismo sucede con el operando 2.

-En el siguiente paso, el secuenciador le indica a la ALU que debe realizar la operación entre los dos operandos y guardarla en el RT 3

-Luego el secuenciador pone una dirección en el contador de programa --> Es emitida por el bus de direcciones hacia la memoria --> Se direcciona la celda donde hay que almacenar el resultado.

-Se realiza una operación de escritura en memoria del valor generado por la ALU a la celda direccionada (Para guardar los datos en los RE se realizan lecturas y para guardar el resultado en memoria se hace por turnos)

-El MAR funciona como la puerta de salida desde el CPU al bus de direcciones de las direcciones. Si el procesador lee datos, estos son obtenidos de la memoria a --> Se ubican en el MBR antes de llegar a su destino.

### ¿Qué es un programa?

- Es una secuencia de pasos.
- Se hace una operación aritmética/lógica por cada paso.
- Diferentes señales de control se necesitan para cada operación:
- la UC saca información de cada instrucción.

EDVAC (1946)

- Electronic Discrete Variable Automatic Computer

EDSAC (Cambridge, 1949)

- Electronic Delay Storage Automatic Calculator

### IAS CARACTERISTICAS:

- Memoria con 4096 palabras de 40 bits
  - Números Binarios
  - 2 instrucciones de 20 bits
- Set de registros (almacenamiento en CPU)
  - Registro Buffer de Memoria (MBR)
  - Registro de Direcciones de Memoria (MAR)
  - Registros de Instrucción y Buffer de Instrucción
  - Registro Contador de Programa (Program Counter)
  - Registros Acumulador y Multiplicador/Cociente

### UNIVAC i (Universal Automatic Computer):

- Primera computadora comercial (1949)
  - (Eckert-Mauchley Computer Corporation).
- Primera en utilizar un compilador para traducir idioma de programa en idioma de máquinas.
- Máquina decimal con 12 dígitos por palabra.
- Principal avance:
  - Sistema de cintas magnéticas que podían leerse hacia adelante y hacia atrás

- Procedimientos de comprobación de errores.
- Memoria de líneas de retardo de mercurio y tecnología a válvulas de vacío.

### SEGUNDA GENERACION: TRANSISTORES:

- Sustituyen a los tubos de vacío
- Mas pequeños
- Mas baratos
- Disipan menos el calor
- Dispositivos de estado solido
- Hechos con silicio
- Inventados en 1947 en los laboratorios Bell
- William Shockley y colaboradores

### TERCERA Y SIGUIENTE GENERACIONES: CIRCUITOS INTEGRADOS

- Integración a pequeña escala: desde 1965
- Mas de 100 componentes en un chip
- Integración a media escala: desde 1971
- 100-3000 componentes por chip
- Integración a gran escala: 1971-1977
- 3000 – 100.000 componentes por chip
- Integración a muy gran escala: desde 1978
- 100.000 - 100 millones de componentes por chip

### Interconexión de un sistema de cómputo:

- Sistema de cómputo está constituido por 3 subsistemas:
  - CPU
  - MEMORIA
  - E/S
- Los componentes deben poder comunicarse entre sí.

### ¿Qué es un Bus?

- **Un bus es un camino de comunicación entre dos o mas dispositivos. Una característica clave de un bus es que se trata de un medio de transmisión compartido. Al bus se conectan varios dispositivos conectados al bus puedan acceder a ella.**

- Si dos dispositivos transmiten durante el mismo periodo de tiempo, sus señales pueden solaparse y distorsionarse, de modo que solo un dispositivo puede transmitir con éxito en un momento dado.
- El bus que conecta los componentes principales del computador se denomina bus de sistema.
- Usualmente “broadcast”
- A menudo agrupadas
  - Un numero de canales en un bus
  - Bus de 32 bits con 32 canales separados de un solo bit cada uno.
- Las líneas de energía pueden no mostrarse.

### **Direcciones:**

- Si el bus es compartido por diferentes elementos, estos deben tener identidades: direcciones.
- La dirección de memoria identifica una celda de memoria en la que almacena información.
- Lectura y escritura se plantean respecto de la CPU.

### **Bus de datos:**

- Transporta datos
  - No hay diferencia entre “dato” e “instrucción” en este nivel.
- El “ancho” es un valor determinante de las prestaciones
  - 8,16,32,64 bits.

### **Bus de direcciones:**

- Identifica el origen o el destino de los datos.
  - La CPU necesita leer una instrucción (dato) de una dada ubicación en memoria.
- El ancho del Bus determina la máxima capacidad de memoria del sistema.

### **Bus de control:**

- Información de control y temporizado
  - Señales de lectura/escritura de Memoria o E/S
  - Señales de selección o habilitación
  - Señales de Reloj (Clock)



- Señales de pedido de interrupción.

### **Componentes de hardware dedicados a cada función:**

- Dispositivos de E/
  - Teclado
  - Mouse
  - Joystick
- Dispositivos de S/
  - Monitor
  - Impresora

### **Componentes de hardware:**

- Para procesamiento
  - CPU
  - Memoria
- Para almacenamiento
  - Memoria
  - Discos (Rígidos, diskettes)
  - Cintas, CD, DVD

### **CPU – ALU:**

- La instrucción se almacena temporalmente en un registro de la CPU llamado IR
- El bloque control puede “leer” IR y así saber que hacer, donde están los operandos y donde poner el resultado.
- ¿Cómo sabe la CPU donde encontrar la próxima instrucción?

Hay un registro en la CPU llamado PC, Contador de Programa o Program Counter.

Cuando un programa va a ser ejecutado, el PC contiene la dirección de la primera instrucción.

Alcanzada la primera instrucción, el PC es incrementado para apuntar a la siguiente instrucción.

### **CPU:**

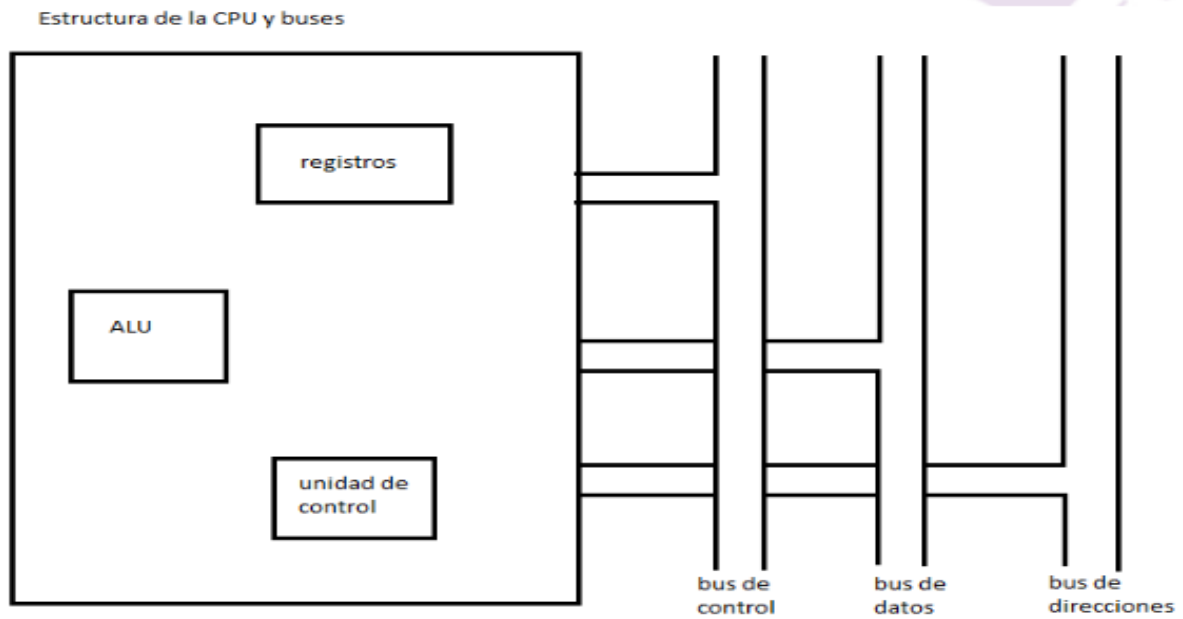
- Todas las CPU tienen registros internos de propósito general que pueden ser referenciados por el programador, como fuente o destino (o ambos) en una instrucción.
- “Como si” fuesen memoria, pero mucho más rápidos. Son lugares de almacenamiento temporario: D0, D1, D2, ...
- La CPU interactúa con la memoria a través de un par de registros que están “ocultos” al programador.
- MAR= registro de dirección de memoria
- MBR= registro de dato de memoria.
- Estos registros están conectados a los buses.
- Además, la CPU tiene otros registros que permiten almacenar direcciones; para poder brindar flexibilidad.

Es el encargado de ejecutar los programas, desde el sistema operativo hasta las aplicaciones de usuario; sólo ejecuta instrucciones programadas en lenguaje de bajo nivel, realizando operaciones aritméticas y lógicas simples, tales como sumar, restar, multiplicar, dividir, las lógicas binarias y accesos a memoria. Esta unidad central de procesamiento (CPU) está constituida, esencialmente, por registros, una unidad de control, una unidad aritmético-lógica (ALU) y una unidad de cálculo en coma flotante (conocida antiguamente como «coprocesador matemático»).

Las cosas que debe hacer una CPU son:

- Captar instrucciones
- Interpretar instrucciones
- Captar datos
- Procesar datos

CUADRO CPU:



## Clase 7:

### Elementos de una instrucción de maquina:

- **Código de operación**
- Especifica la operación a realizar (Ej: suma)
- Es un código binario
- **Referencia del operando fuente**
- Establece donde se encuentra el operando
- La operación puede involucrar uno o más operando fuente (o, de entrada)

### Elementos de una instrucción de maquina (2):

- **Referencia del operando resultado**
- Establece donde almacenar el resultado
- **Referencia de la siguiente instrucción**
- Le dice a la CPU donde buscar la siguiente instrucción después de la ejecución de la instrucción anterior
- **En la mayoría de los casos se ubica a continuación de la instrucción actual.**

### Elementos de una instrucción de maquina (3):

- Los operandos fuente y resultado pueden estar en tres lugares:
- Memoria
- Registro de la CPU
- Dispositivos de E/S

### Representación de instrucciones:

- Dentro de la computadora cada instrucción está representada mediante una secuencia de bits
- La secuencia se divide en campos en correspondencia a los elementos que la componen.
- Este esquema se conoce como formato de la instrucción
- Es difícil para el programador tratar con las representaciones binarias de las instrucciones de máquina. Por lo tanto, se usa una representación simbólica.
- Los códigos de operación se representan por medio de abreviaturas, llamadas **mnemónicos** que indican la operación

### Representación de instrucciones: Los ejemplos más comunes:

- ADD adición (suma)
- SUB sustracción (resta)
- MOV movimiento de datos
- AND, OR, XOR operaciones lógicas

### Tipos de instrucciones:

- El lenguaje de máquina expresa las operaciones en forma “básica” involucrando movimiento de datos y uso de registros.
- Cualquier programa escrito en lenguaje de alto nivel se debe convertir a un lenguaje de máquina para ser ejecutado.

### Podemos categorizar las instrucciones de maquina como de:

- **Procesamiento de datos**
- Operaciones aritméticas y lógicas.

- Almacenamiento de datos
- Transferencias dentro del sistema.
- Instrucciones de E/S
- Transferencia de datos entre la computadora y los mecanismos externos.
- Control.

#### Máquina 4 direcciones:

- Direcciones explícitas para operandos, resultado y próxima instrucción.
- Son “raras”, cada campo de dirección tiene que tener bits para “acomodar” una dirección completa.
- Ej. si dirección = 24 bits, la instrucción tiene 96 bits de referencias.
- Add, DirRes, DirOp1, DirOp2, DirProxIns

#### Máquina de 3 direcciones:

- Add, DirRes, DirOp1, DirOp2
- Dirección de la próxima instrucción está almacenada en un registro de la CPU, llamado Contador de Programa PC.
- Referencias = 72 bits. Todavía larga.

#### Máquina de 2 direcciones:

- Reduce el tamaño de la instrucción.
- 48 bits de referencias.
- Hay que mover el Op1 a un registro temporal.
- Menos elección donde guardar el resultado.

#### Máquina de 1 dirección:

- Add, DirOp1
- Registros especiales en la CPU (acumulador).
- Instrucciones para cargar y descargar el acumulador.
- Un operando y resultado en lugar predefinido.

- Instrucción más corta (24 bits de referencias).

### Diseño del conjunto de instrucciones:

- El conjunto de instrucciones es el medio que tiene el programador para controlar la CPU

### Tipos de operaciones:

- **Transferencia de datos:** Mov (load/store)
- **Aritméticas:** Add, Sub, Inc, Dec, Mul
- **Lógicas:** And, Or, Xor, Not
- Conversión
- **E/S:** In, Out
- **Transferencia de control:** salto, bifurcación
- **Control del sistema:** usadas por S.O.

### Tipos de datos: LOS MAS IMPORTANTES:

- Direcciones
- Números: enteros, Punto Fijo, Punto Flotante
- Caracteres: ASCII, BCD
- Datos lógicos

### Modos de direccionamiento:

Hay 2 métodos generales:

- 1) Si un operando va a usarse varias veces puede colocarse en un registro
  - **Usar un registro para una variable tiene 2 ventajas:**
    - El acceso es más rápido
    - Se necesitan menos bits.
    - Ej. si hay 32 reg. se necesitan 5 bits para especificar c/u de ellos (menos bits que las dir. de mem.).
- 2) Especificar uno o más operandos en forma implícita:
  - Ejemplos: reg2= reg2+fuentes1; el acumulador.

Los mdd (métodos de direccionamiento) tienen como objetivo:

- disminuir la cantidad de bits en la instrucción
- la dirección puede que no se conozca hasta el momento de ejecutar el programa
- manejo más eficiente de datos (arreglos)

Modos de direccionamiento (4):

- Inmediato
- Directo
- Por Registro
- Indirecto por memoria
- Indirecto por registro
- Por desplazamiento
- Del stack

Mdd inmediato:

- El operando se obtiene automáticamente de la memoria al mismo tiempo que la instrucción.
- No requiere una referencia extra a memoria de datos
- Se utiliza para definir constantes y para inicializar variables.
- Desventaja: tamaño del operando limitado por el tamaño del campo de direccionamiento.

Mdd Directo:

- El campo de dirección tiene la dirección efectiva del operando.
- Es simple, pero tiene un espacio limitado de direcciones por cantidad de bits del campo.
- Uso: acceder a variables globales, cuya dirección se conoce en el momento de compilación.

Mdd Por registro:

- Conceptualmente igual al directo, pero se especifica un registro en lugar de una posición de memoria.

- La referencia a registro usa menos bits que la especificación de la dirección y no requiere acceso a memoria de datos.
- Desventaja: los registros no son muchos y es un recurso preciado.

#### Mdd Indirecto por memoria:

- En la instrucción está la dirección de la dirección del operando. Trata de solucionar el problema del directo. Así, con una dirección de menos bits en la instrucción, se apunta a una dirección de más bits.
- Ventaja: espacio de direccionamiento mayor
- Desventaja: múltiples accesos a memoria.

#### Mdd Indirecto por registro:

- En la instrucción se especifica el registro que tiene almacenada la dirección.
- Ventaja: menos bits para especificar el registro que la posición de memoria. Espacio de direccionamiento grande, accede una vez menos a memoria que el indirecto. La dirección así usada se llama apuntador.

#### Mdd Por desplazamiento:

- Combina capacidades de indirecto y directo. Requiere que la instrucción tenga dos campos de dirección. Estos dos campos se suman para producir la dirección efectiva. Los más comunes:
  - Relativo
  - De registro base
  - Indexado

#### Mdd Del Stack:

- El stack o pila es un arreglo lineal de localidades de memoria. Es una lista o cola donde el último en entrar es el primero en salir. Es una zona de memoria reservada.



- Asociado con la pila o stack hay un registro apuntador (o registro puntero de pila), cuyo valor es la dirección tope de pila o stack.

## Clase 8:

### Organización de registros:

- **Registros visibles al usuario:** son utilizados por el programador.
- **Registros de control y estado:** son utilizados por la UC para controlar la operación de la CPU (no son visibles por el programador).

### Registros visibles al usuario:

- Propósito general
- Datos
- Dirección
- Códigos de condición

### Registros visibles al usuario (2):

- **Pueden ser asignados a una variedad de funciones:**
- cualquier registro de propósito general puede contener el operando para cualquier código de operación (verdadero propósito)
- pueden existir restricciones (ej. registros dedicados a operaciones en PF)
- se pueden utilizar para direccionamiento (ej. indirecto de registro)
- sólo para datos o sólo para direcciones
- los registros de dirección pueden ser asignados para un mdd (ej. reg. índice para direccionamiento auto indexado)

### Números de registros:

- Afecta al tamaño de la instrucción.

- Mayor número de registros, más bits para especificarlos en la instrucción.
- **Pocos registros:** más referencias a memoria
- **Numero optimo:** entre 8 y 32 reg. Mas, no hay gran mejora (aumenta tamaño de la instrucción).

### Registros de control y estado:

- Empleados para controlar la operación de la CPU. En la mayoría de las máquinas no son visibles al usuario.
- Los 4 esenciales para la ejecución de instrucciones:
- Contador de programa (PC)
- Registro de instrucción (IR)
- Registro de dirección de memoria (MAR)
- Registro buffer de memoria (MBR)

### Clase 9:

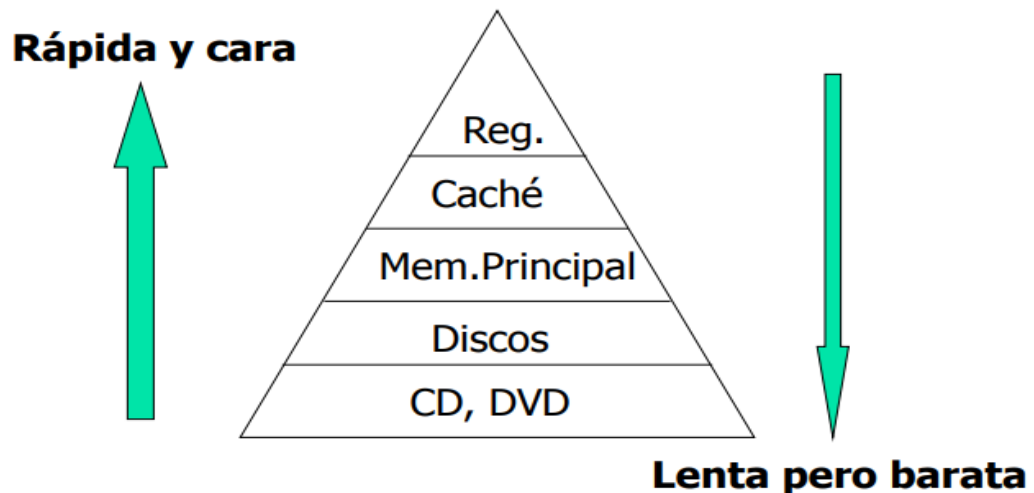
### Memoria:

- **Velocidad del procesador:** se duplica cada 18 meses (sin variar su precio) la cantidad de instrucciones ejecutadas por segundo.
- **Memoria:** se cuadruplica su tamaño cada 36 meses (al mismo precio). Velocidad aumenta a razón de un 10% anual.
- **A medida que aumenta** la brecha entre las velocidades del procesador y de la memoria, las distintas arquitecturas buscan tender un puente sobre esta brecha. Una computadora típica suele tener distintos tipos de memoria, desde una rápida y cara (registros) hasta una barata y lenta (discos).
- **La interacción entre los diferentes** tipos de memoria se aprovecha de forma tal que se logra un comportamiento, por parte de la computadora, equivalente al que tendría con una memoria única, grande y rápida, cuando en realidad tiene distintos tipos de memoria trabajando en forma coordinada.

### Jerarquía de memorias:

- La forma en que se organizan estos distintos tipos de memoria es lo que se conoce como **jerarquía de memoria**.
- En la cima de la jerarquía están los registros. En la base, las memorias secundarias (discos magnéticos) y de almacenamiento “off line” (CD, DVD, cintas)

## Jerarquía de memorias (2)



- **A medida que ascendemos** tenemos mayor rendimiento y más costo por bit. Entre la memoria principal y la secundaria hay otro tipo de memoria para salvar la brecha.
- **Cuando ascendemos**, también aumenta la frecuencia de accesos a ese tipo de memoria.

<b>Tipos de memoria</b>	<b>Tiempo de acceso</b>	<b>Tamaño típico</b>
Registros	1 ns	1 KB
Caché	5-20 ns	1 MB
Mem. Principal	60-80 ns	1 GB
Discos	10 ms	160 GB

**Características:**

- **Memorias semiconductoras:**

- Tiempo de acceso: tiempo máximo que transcurre desde que se inicia la operación de lec/esc hasta obtener/almacenar el dato.
- Tiempo de ciclo: tiempo mínimo que tiene que haber entre dos operaciones sucesivas sobre la memoria  
 $t(\text{ciclo}) > t(\text{acceso})$

- **Memorias magnéticas:**

- Tiempo de acceso: tiempo de posicionar el cabezal + tiempo de latencia (+ tiempo de lectura)
- Velocidad de transferencia: bytes/seg

- **Métodos de acceso:**

- **Acceso aleatorio:** el tiempo para acceder a una locación dada es independiente de la secuencia de accesos anteriores y es constante. Ejemplo la memoria principal.
- **Acceso secuencial:** el acceso debe hacerse en una secuencia lineal específica. Variable. Ejemplo son las unidades de cinta.
- **Acceso directo:** los bloques ó registros individuales tienen una dirección única que se basa en la localización física. Variable. Ejemplo los discos magnéticos.
- **Acceso asociativo:** memoria caché.
- **Memoria de acceso ALEATORIO:**
- RAM (Random Access Memory). Aleatorio significa que se puede acceder a cualquier celda de memoria en el mismo tiempo, independientemente de la posición en la estructura de la memoria.
- **Nota:** Memorias estáticas y dinámicas se cargan en transistores y capacitores.

❖ DRAM almacena más información que SRAM en la misma superficie.

❖ Los 'capacitores' son más chicos que los flip flop

❖ DRAM hay que "refrescarla"

❖ Los 'capacitores' se descargan

❖ Usada como memoria principal

❖ SRAM más rápida

❖ Usada como memoria caché

### Organización:

El elemento básico de una memoria de semiconductor es la celda de memoria.

- **Todas** las celdas de memoria de semiconductor comparten 3 propiedades:
  - 1) Dos estados estables: para representar al uno (1) y al cero (0).
  - 2) Se puede escribir en ellas, al menos una vez
  - 3) Se pueden leer para conocer el estado.
- **En general** la celda tiene 3 terminales funcionales capaces de llevar una señal eléctrica:
  - Selección: selecciona una celda de memoria
  - Control: especifica lectura ó escritura
  - Escritura/Lectura de datos

### Organización del chip:

- Cada chip contiene un arreglo de celdas de memoria.
- En las memorias de semiconductor se han empleado dos enfoques organizacionales: 2D y 2½D.

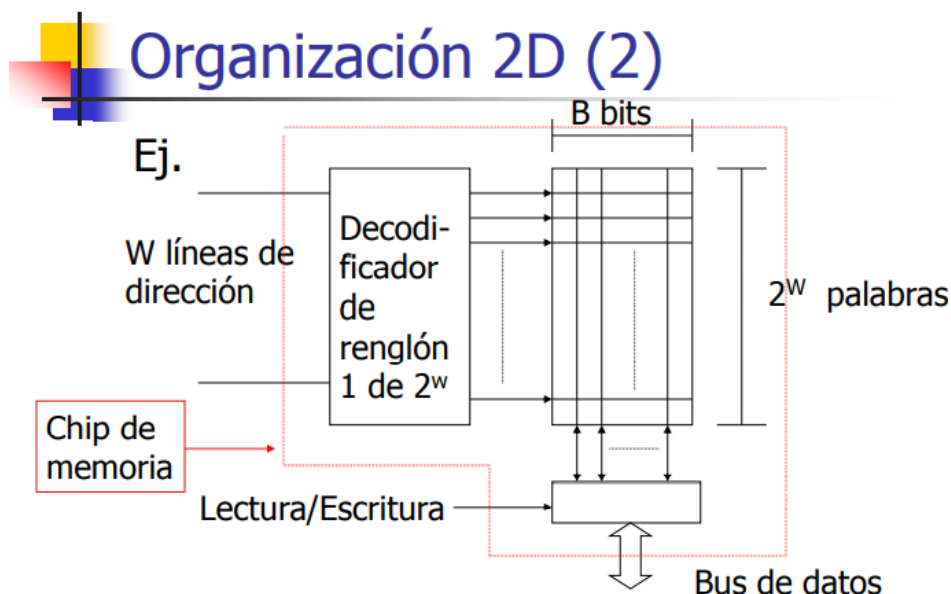
#### - Organización 2D:

- **Es la forma más sencilla de organización de memoria.**
- Las celdas forman una matriz de 2 n filas y M columnas, siendo 2 n el número de palabras del chip y m la cantidad de

bits de cada palabra. Cada fila es seleccionada por la decodificación de una configuración diferente de los  $N$  bits de dirección. Esta organización tiene el inconveniente de que el selector (decodificador) crece exponencialmente con el tamaño de la memoria. Igual ocurre al número de entradas de las compuertas que generan las salidas.

- El arreglo está organizado en  $2^W$  palabras de  $B$  bits cada una. Cada línea horizontal (una de  $2^W$ ) se conecta a cada posición de memoria, seleccionando un renglón.
- Las líneas verticales conectan cada bit a la salida.
- El decodificador que está en el chip, tiene  $2^W$  salidas para  $W$  entradas (bits del bus de direcciones).

Gráfico del 2D:



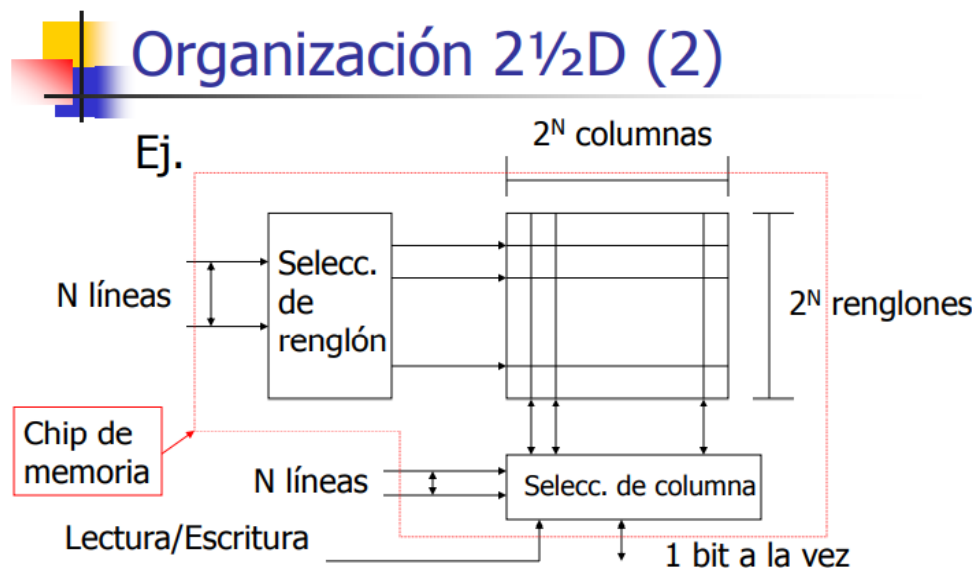
- Organización  $2\frac{1}{2}D$ :

- En lugar de una sola selección de  $2^n$  salidas, en esta organización se utilizan dos decodificadores de  $2^{n/2}$ , operando en coincidencia. Las líneas de direcciones se reparten entre los decodificadores. Para una configuración dada de las líneas de dirección se selecciona un único bit de

la matriz. Por ello también se la denomina organización por bits.

- En esta organización se necesitan múltiples matrices, tantas como bits deba tener la palabra. De este modo, para armar una palabra, se selecciona un bit de cada matriz en la misma exacta posición de cada matriz, actuando de forma paralela.
- El arreglo es 'cuadrado' y funciona igual que 2D.
- Los bits de una misma palabra están dispersos en distintos chips.
- La dirección se divide en dos partes: una selección de renglón y una selección de columna. Hay 2 decodificadores.

Gráfico de 2½D:



Comparación:

- En 2D todos los bits están en el mismo chip.
- En 2½D los bits de una misma palabra estarán en distintos chips.
- 2D es muy larga y estrecha, No grande de palabras de pocos bits. Cada línea de selección de palabra tiene que tener un

manejador y conectarse al decodificador. Ocupan mucha superficie.

- **2D** dificulta el uso eficaz de los circuitos correctores de error. En 2½D al estar los bits dispersos en distintos chips hay menor probabilidad de error.
- **En 2½D** al usar decodificación separada de filas y columnas, reduce la complejidad de los decodificadores.

#### Problema 1:

- Cada módulo de memoria cubre el espacio de direccionamiento requerido, pero sólo cubre una parte de la palabra.
- Solución: usar varios módulos 'en paralelo'.

#### Problema 2:

- La longitud de la palabra es la deseada, pero los módulos no tienen la capacidad deseada.
- Solución: cubrir un cierto rango de direcciones con módulos de memoria 'en serie'

#### Nuevas tecnologías RAM:

- **La DRAM** básica es la misma desde los primeros chips de RAM
- **Enhanced DRAM**
  - Contiene pequeña SRAM
  - La SRAM guarda la última línea leída (¡como una Cache!)
  - **Cache DRAM**
  - Contiene una SRAM más grande
  - Se usa la SRAM como cache o como buffer serial
  - **Synchronous DRAM (SDRAM)**
  - Actualmente en DIMMs
  - Acceso sincronizado con un reloj externo
    - Se presenta una dirección a la RAM
    - RAM encuentra los datos (y CPU esperaría la DRAM)



- SDRAM mueve datos en tiempo del reloj del sistema, la CPU conoce cuando los datos estarán listos
- CPU puede hacer otra cosa mientras tiene que esperar
- Modo Burst permite SDRAM trabajar en bloques

## Clase 10

### Memoria Caché:

- Históricamente CPU han sido más rápidas que las memorias.
- El aumento de circuitos que es posible incluir en un chip
- Diseñadores de CPU lo usaron para hacerla más veloz (ej. pipeline).
- Los diseñadores de memoria lo usaron para aumentar la capacidad del chip (más memoria, más grandes decodificadores).
- **Esta diferencia implica:** después que la CPU 'emite' una solicitud de lectura a la memoria (bus de direcciones, bus de control) pasan muchos ciclos de reloj antes que reciba la palabra que necesita, por el bus de datos.
- **En todos** los ciclos de instrucción, la CPU accede a memoria al menos una vez, para buscar la instrucción y muchas veces accede a buscar operandos.
- **La velocidad** a la cual la CPU ejecuta instrucciones está limitada por el tiempo del ciclo de memoria.
- **El problema** no es tecnológico sino económico. Se pueden construir memorias tan rápidas como la CPU, pero para obtener la máxima velocidad tiene que estar dentro del chip de la CPU
- Llegar a la memoria por el bus del sistema es 'lento'.
- **Solución:** Técnicas para combinar una cantidad pequeña de memoria rápida con una cantidad grande de memoria lenta, para obtener la velocidad de memoria 'casi' rápida.

### Principios:

- El uso de la memoria caché se sustenta en dos principios ó propiedades que exhiben los programas:

#### 1) Principio de localidad espacial de referencia

- cuando se accede a una palabra de memoria, es “muy probable” que el próximo acceso sea en la vecindad de la palabra anterior.

#### 2) Principio de localidad temporal de referencia:

- cuando se accede a una posición de memoria, es “muy probable” que un lapso de “tiempo corto”, dicha posición de memoria sea accedida nuevamente.

#### Localidad espacial:

- Localidad espacial, se sustenta en:
- Ejecución secuencial del código
- Tendencia de los programadores a hacer próximas entre si variables relacionadas
- Acceso a estructuras tipo matriz o pila

#### Localidad temporal:

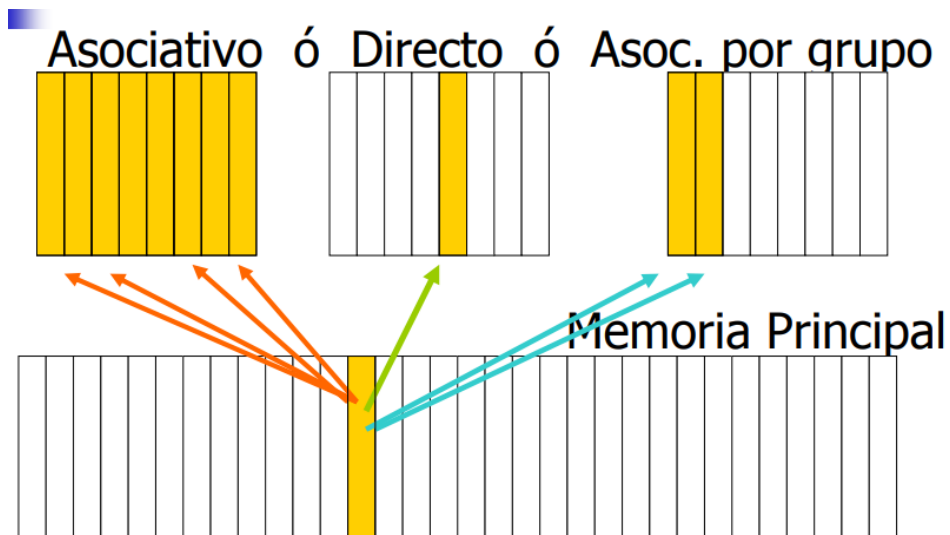
- Localidad temporal, se sustenta en:
- Formación de ciclos o bucles
- Subrutinas (Procedimientos o Funciones)
- Pilas

#### Caché:

- La idea general es que cuando se hace referencia a una palabra, ella y alguna de las vecinas se traen de la memoria grande y lenta a la caché, para que en el siguiente acceso la palabra buscada se encuentre en el caché.

#### Mapeo de la memoria:

Asociativo Ó Directo Ó Asociativo por grupo.



### Aciertos y fallos:

- La efectividad de la caché se expresa a través de la **frecuencia de aciertos**: es decir el número de veces que la caché acierta direcciones.
- Un acierto de caché sucede cuando los datos que necesita el procesador están almacenados en la caché
- la CPU obtiene los datos a alta velocidad.
- Un **fallo de caché** ocurre cuando los datos buscados no se encuentran en la caché
- la CPU tiene que obtenerlos de la memoria principal, a una velocidad menor.

### Caché: L1 y L2

- ¿Por qué hay dos (ó +) niveles de caché?
  - L1 y L2
- Porcentaje de aciertos: 90%
- Porcentaje de fallos: 10%
- Conviene mejorar el 90% con el mismo razonamiento y no poner esfuerzo en el 10% restante.

### Memoria Externa:

- Tipos de memoria externa:
  - Discos magnéticos

- Discos Ópticos
  - CD-ROM
  - CD-R
  - CD-RW
  - DVD
- Cintas Magnéticas

### Discos Magnéticos:

Platos:

- Superficies de AL cubiertos con oxido de Fe, material magnético,
- Ahora también se usa vidrio
- Se dilata menos que el AL.
- Superficie más uniforme.
- Reducción de defectos superficiales.

### Principios Físicos:

- Pequeñas áreas del disco son magnetizadas en diferentes direcciones por un transductor.
- Debe existir un movimiento relativo entre disco y el transductor al momento de la lectura/escritura.
- Cambios en la dirección de magnetización es lo que se detecta en la lectura

### Mecánica de lectura y escritura:

- Lectura y escritura es a través de una cabeza transductora (bobina)
- Durante lectura/escritura, la cabeza es estacionaria y el plato gira
- Se almacenan ceros y unos por medio de la magnetización de pequeñas áreas del material.

### Pista y sector:

- Un número entero de sectores se graban en una pista.
- El sector es la unidad de transferencia de/hacia el disco.

### Sector típico:

- Sucesión o serie de bits divididos en campos:
- **Encabezado** con información para sincronizar la lectura e identificar el sector
- **Datos** con longitud en bytes expresada usualmente como potencia de 2
- **Código para errores** con información para detectar y/o corregir posibles errores.

### Características posibles:

- Cabeza fija (raro) o móvil.
- Disco removible o fijo.
- Simple o doble lado.
- Uno o múltiples platos.
- Mecanismo de cabeza:
- Contacto (Floppy)
- Distancia Fija
- Aerodinámica (Winchester)

### Estructura de un disco:

- Múltiples platos
- Una cabeza por cara
- Todas las cabezas se mueven solidariamente
- Pistas alineadas en cada plato forman cilindros
- Datos son almacenados por cilindros
  - Reduce movimientos de cabezas
  - Aumenta velocidad de respuesta

**Formato:** Define cantidad, tamaño y función de distintos campos en cada pista.

- **Hardware:** Tamaño de sector fijo por marcas físicas
- **Software:** tamaño de sector determinado por S.O.

### Clase 11:

#### CD-ROM:

- Basado en CD para audio
- Policarbonato revestido con capa altamente reflectiva, usualmente aluminio.
- Datos almacenados como “pits”
- Lectura por láser reflejado

#### Acceso al CD-ROM:

- Difícil
- Mover cabeza lectora a una posición cercana
- Establecer la velocidad correcta
- Leer la identificación (dirección)
- Ajustar a la posición requerida

#### Capacidad CD-ROM:

- 666000KB = 650 MB

#### CD-ROM pros y contras:

- Gran capacidad
- Fácil para producción en masa
- Removible
- Robusto
- Caro en pequeñas corridas
- Lento
- Solo lectura

#### Ópticos:

- CD-recordable
- CD-RW

### DVD:

- Digital Video Disk
- Dispositivo para films
  - Solo películas
- Digital Versatile Disk
- Dispositivo para computadoras
  - Puede leer disco de computadoras y discos de video

### DVD - Tecnología:

- Multi-capa
- Capacidad muy alta
- Toda una película
  - Compresión MPEG
- Estandarizado

### Diferencia entre blu ray y DVD:

- El formato **Blu Ray** utiliza el estándar de compresión H. 264, el cual ofrece la misma calidad que el MPEG-2, utilizando sin embargo menos espacio. Además, tiene más datos por segundo y mejores codificadores que el **DVD**.

### Cinta Magnética:

- Acceso en serie
- Lento
- Muy Económico
- Backup y archivo

### MODEM:

- **Convierte** señales “0” y “1” en tonos de audio.
- Sistema telefónico responde entre 50 y 3500Hz.

- **Tasa** de Bits/seg (bps) es el número de bits enviados por segundo.
- **Tasa Baudio** (Baud rate) es el número de cambios de señal por segundo (Por J. Baudot).
- Máxima tasa baudio para el sistema telefónico es 2400.
- **Es posible** enviar varios bits por baudio, señalando en frecuencias diferentes

#### MODEM (2):

- Amplitud:
  - Modulada
  - Frecuencia
  - Modulada
  - Fase
  - Modulada

#### Dispositivos de ENTRADA de Datos:

- Teclado y Mouse:
  - Tasas de entrada muy lentas
  - 10 caracteres de 8 bits por segundo en teclado
  - El mouse es más rápido: 1 cambio en los bits de la posición X e Y por milisegundo
  - Click de mouse: bit por 1/10 segundo
- **El desafío** del diseño de dispositivos de entrada de datos manual es reducir el número de partes móviles

#### Dispositivos de SALIDA de Datos:

- Monitores de video
  - Alfanuméricos
  - Gráficos
- Impresoras
  - Impacto
  - Laser



### Monitores de Video:

- Color O blanco y negro
- Se muestran 50/60 cuadros completos por segundo
- Resolución vertical: número de líneas = 500
- Resolución Horizontal: punto por línea = 700

### Dos tipos de Video: Terminal Y Mapeado en memoria:

- **Monitor** de video, memoria de visualización y teclados armados juntos para formar un terminal
- **Monitor** de video con memoria de visualización que esta mapeada en memoria
- **Terminales:** usualmente orientados a carácter
- Conexión con ancho de banda pequeño (serie)
- **Visualización** con memoria de video mapeada permite mostrar imágenes y movimiento
  - Conexión al bus de memoria permite cambios rápidos (ancho de banda grande)

### ROM de caracteres:

- Los bits de una línea son leídos serialmente
- Se accesa 9 veces a la misma posición horizontal y sucesivas posiciones verticales.