

## Introducción a la Arquitectura de Sistemas – Ayuda de Memoria Para la Lección

### Representación de Numero Reales:

32 Bits	}	➤ <b>IBM 360:</b>	<table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td style="width: 20px;">S</td> <td style="width: 40px;">exponente CD(2,7)</td> <td style="width: 40px;">mantisa SVA(16,6)</td> </tr> <tr> <td>31 30</td> <td>24 23</td> <td>0</td> </tr> </table>	S	exponente CD(2,7)	mantisa SVA(16,6)	31 30	24 23	0
		S	exponente CD(2,7)	mantisa SVA(16,6)					
		31 30	24 23	0					
Normalización 0,X Frontera equilibrada									
➤ <b>PDP/11:</b>	<table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td style="width: 20px;">S</td> <td style="width: 40px;">exponente CD(2,8)</td> <td style="width: 40px;">mantisa SVA(2,24)</td> </tr> <tr> <td>31 30</td> <td>23 22</td> <td>0</td> </tr> </table>	S	exponente CD(2,8)	mantisa SVA(2,24)	31 30	23 22	0		
S	exponente CD(2,8)	mantisa SVA(2,24)							
31 30	23 22	0							
Normalización 0,1X Frontera equilibrada									
➤ <b>IEEE754 Corto:</b>	<table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td style="width: 20px;">S</td> <td style="width: 40px;">exponente CD(2,8)</td> <td style="width: 40px;">mantisa SVA(2,24)</td> </tr> <tr> <td>31 30</td> <td>23 22</td> <td>0</td> </tr> </table>	S	exponente CD(2,8)	mantisa SVA(2,24)	31 30	23 22	0		
S	exponente CD(2,8)	mantisa SVA(2,24)							
31 30	23 22	0							
Normalización 1,X Frontera NO equilibrada $\frac{b^d}{2} - 1$									
64 Bits	}	➤ <b>IEEE754 Largo:</b>	<table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td style="width: 20px;">S</td> <td style="width: 40px;">exponente CD(2,11)</td> <td style="width: 40px;">mantisa SVA(2,53)</td> </tr> <tr> <td>63 62</td> <td>52 51</td> <td>0</td> </tr> </table>	S	exponente CD(2,11)	mantisa SVA(2,53)	63 62	52 51	0
		S	exponente CD(2,11)	mantisa SVA(2,53)					
63 62	52 51	0							
Normalización 1,X Frontera NO equilibrada $\frac{b^d}{2} - 1$									

### Representación de Imágenes:

P: profundidad

C: cantidad de colores :  $2^P \rightarrow P = \lceil \log_2 C \rceil$

$$\text{Tamaño en Bytes} = \frac{\text{Alto} \times \text{Ancho} \times P}{8}$$

### Representación de Videos:

FPS: frames por segundo

$$\text{Tamaño en Bytes} = \frac{\text{Alto} \times \text{Ancho} \times P \times \text{FPS} \times \text{Tiempo}}{8}$$

### Representación de Sonido:

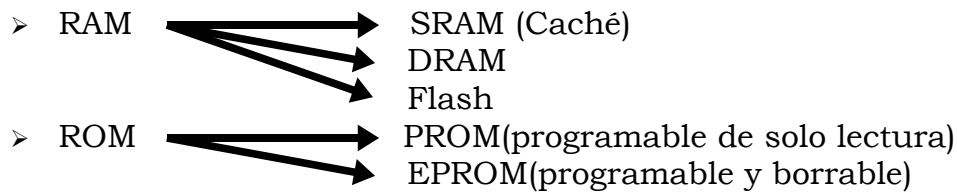
BMP: bits por muestra

C: canales

$F_m$ : frecuencia de muestreo

$$\text{Tamaño en Bytes} = \frac{BMP \times C \times F_m \times \text{Tiempo}}{8}$$

### Memorias:



### Discos Duros:

$$\text{Tiempo Total} = T_{\text{busqueda}} + T_{\text{espera}} + T_{\text{transferencia}}$$

(tiempo de acceso)

$$T_{\text{busqueda}} = T_0 + T_p \times \#P$$

(tiempo de arranque del motor) (tiempo en alcanzar una pista)

$$T_{\text{espera}} = \frac{1}{2w}$$

w es la velocidad angular (rps)

$$T_{\text{transferencia}} = \frac{\# \text{bytes a leer}}{V_{\text{transferencia}}} = \frac{\# \text{bytes a leer}}{C_p w}$$

$$\text{Capacidad} = \frac{\# \text{bytes}}{\text{sector}} \frac{\# \text{sectores}}{\text{pista}} \frac{\# \text{pistas}}{\text{cara}} \# \text{caras}$$

### Discos Raid: (redundant array of inexpensive/independent disk)

Ventajas: tolerantes a fallos (mediante la redundancia)  
mayor velocidad (mediante paralelismo)

- Nivel 0 → Stripping
- Nivel 1 → Espejo
- Nivel 2 → No se estudia
- Nivel 3 → No se estudia
- Nivel 4 → = a Nivel 0 + un disco de paridad
- Nivel 5 → = a Nivel 4 pero con la paridad distribuida
- Nivel 6 → No se estudia

**CODE-2:** (Computadora didáctico-elemental)

Procesador sencillo de uso educativo

Ventajas: maximizar velocidad

encontrar la mejor manera de hacer algo

desventajas: bajo nivel de abstracción

mucha repetición de código

Direccionamiento:

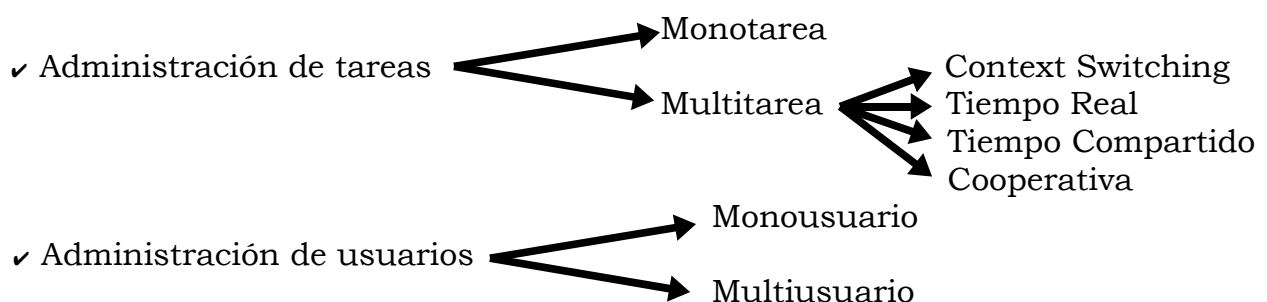
- Implícito: el operando se encuentra en un registro implícito, B-, CALL, RET.
- Inmediato: el operando forma parte de la instrucción misma, el campo v en instrucciones como el LLO, LHI, LD y ST.
- De registro: la instrucción guarda los bits correspondientes al numero de registro que contiene el operando a utilizar, ADDS, SUBS, NAND.
- Directo: la instrucción tiene la dirección de memoria o el numero de registro en la que se encuentra el operando. CODE lo emula en F1, F4 y LLO, LLI. (no la posee CODE)
- Indirecto: la instrucción indica la dirección de memoria o el numero de registro en la que se encuentra la dirección de memoria del operando. CODE lo emula en ST y LD. (no la posee CODE)
- Base indexado: la dirección base se encuentra almacenada en un registro y en la instrucción se encuentra el desplazamiento (o viceversa), LD y ST.
- De pila: caso particular de direccionamiento implícito. Se accede al operando a través del registro de pila RE, CALL, RET

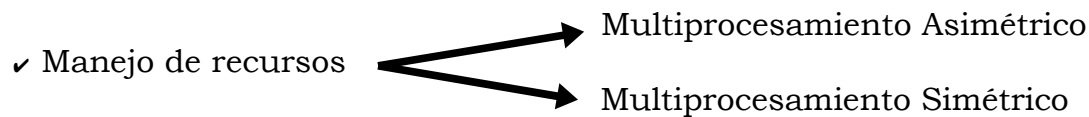
**Sistemas Operativos:** (conjunto de programas destinados a permitir la gestión eficaz de los recursos de una computadora)

Funciones:

1. Interpretar comandos.
2. Coordinar y manipular el hardware.
3. Organizar los archivos en dispositivos de almacenamiento.
4. Gestión de errores de hardware.
5. Servir de base para la creación de software.
6. Configurar el entorno para el uso de software y periféricos.
7. Controlar recursos compartidos.

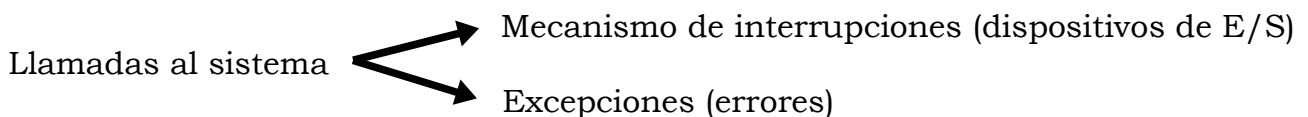
Característica:





*Shell*: es el programa arrancador del sistema operativo.

*El modo kernel o núcleo* es la parte fundamental de un sistema operativo. Es el software responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora mediante llamadas al sistema.



### **Comunicación de Datos:**



OSI propone resolver el problema de la comunicación de datos dividiéndolo en 7 niveles.

- 1- Físico → Parte topológica, transmisión de bits puros.
- 2- Enlace → Se encarga de que todas las maquinas conectadas por cable puedan comunicarse. Ethernet posee el protocolo CSMA/CD.
- 3- Red → Algoritmo y tablas de ruteo.
- 4- Transporte → control de flujo, los protocolos mas conocidos UDP y TCP.
- 5- Sesión → Aparece el concepto de numero de puerto.
- 6- Presentación → Se encarga de la Sintaxis y Semántica. (TCP-IP no tiene).
- 7- Aplicación → Contiene los protocolos mas conocidos como el HTTP.

### **Detección de Errores:**

- Técnica de la paridad.
- Comprobación de Redundancia Cíclica.

### **Corrección de Errores:**

- Hamming.

**Paradigmas de la Programación:** (ideas que indican una forma de programar)

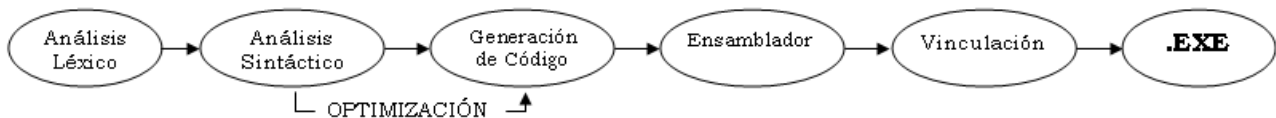
- Imperativo
  - ➔ Orientado a flujo → BASIC.
  - ➔ Procedural → Pascal, C.
- Declarativos
  - ➔ Programación funcional → LISP.
  - ➔ Programación lógica → Prolog.
- De Objeto
  - ➔ Programas orientados a objetos → java y C++.  
(Herencia y Polimorfismo)

Lenguajes 

- ➔ Alto Nivel → Facilita el trabajo.
- ➔ Bajo Nivel → Difícil de escribir, pero más rápidos.

**Pasaje de Alto Nivel a Bajo Nivel:**

- Interpretación: lee y compila línea por línea en el orden de ejecución. Es más flexible ya que si modifico el programa el binario se genera de inmediato
- Compilación: compila todo el programa de una vez. Es más rígido, ya que cualquier modificación en el programa implica realizar una nueva compilación.



**Análisis Léxico:** Busca símbolos y los identifica, operadores, constantes, primitivas.

**Análisis Sintáctico:** crea un árbol de sintaxis.

**Análisis semántico:** comprueba que las construcciones tengan significado, a partir del árbol.

**Optimizador:** a partir del árbol de sintaxis trata de mejorar el código

**Generador de código:** Traduce a assembler a partir del árbol.

**Ensamblador:** Traduce a lenguaje máquina (binario).

**Vinculación:** une cada variable con un lugar en la memoria. Utiliza dll's.