# Resumen Parcial 2 AyOC

## Índice:

[Resumen Parcial 2 AyOC 1](#_Toc168430308)

[Discos 2](#_Toc168430309)

[¿Qué es un disco duro? 2](#_Toc168430310)

[Estructura física de un disco duro 2](#_Toc168430311)

[Velocidad de giro del disco 3](#_Toc168430312)

[Velocidad interna 4](#_Toc168430313)

[Tamaño de Buffer o Cache 4](#_Toc168430314)

[Velocidad externa 4](#_Toc168430315)

[Tipos de acceso 4](#_Toc168430316)

[Tiempos 4](#_Toc168430317)

[Tiempo de búsqueda medio 4](#_Toc168430318)

[Latencia rotacional media: 4](#_Toc168430319)

[Tiempo medio de acceso 4](#_Toc168430320)

[Cálculo de la capacidad del disco 5](#_Toc168430321)

[Tipos de interfaces 5](#_Toc168430322)

[Interfaz ATA/IDE 5](#_Toc168430323)

[Interfaz SCSI 5](#_Toc168430324)

[Interfaz Serial ATA (SATA) 6](#_Toc168430325)

[Formato 6](#_Toc168430326)

[SSD 6](#_Toc168430327)

[Ventajas 6](#_Toc168430328)

[Desventajas 6](#_Toc168430329)

[Sistema de interconexión de BUSES 7](#_Toc168430330)

[Funcionamiento del bus: 7](#_Toc168430331)

[Jerarquía de buses múltiples: 8](#_Toc168430332)

[Elementos de diseño de un bus 9](#_Toc168430333)

[Tipos 9](#_Toc168430334)

[Tipo dedicado 9](#_Toc168430335)

[Tipo Multiplexado: 9](#_Toc168430336)

[Métodos de arbitraje 10](#_Toc168430337)

[Temporización 10](#_Toc168430338)

[Temporización síncrona: 10](#_Toc168430339)

[Temporización asíncrona: 10](#_Toc168430340)

[Anchura del bus 10](#_Toc168430341)

[Tipo de transferencia de datos 10](#_Toc168430342)

[Memorias: 11](#_Toc168430343)

[Tipos de memoria de computadora: 11](#_Toc168430344)

[Memoria RAM (Random Access Memory – Memoria de Acceso Aleatorio) 12](#_Toc168430345)

[Memoria DRAM: 12](#_Toc168430346)

[Celdas de la memoria DRAM: 13](#_Toc168430347)

[Proceso de Lectura y escritura 13](#_Toc168430348)

[Memoria SRAM 13](#_Toc168430349)

[Nomenclatura para las memorias DDR 13](#_Toc168430350)

[Memoria Cache 14](#_Toc168430351)

[¿Qué es la memoria cache L1 L2 L3 y cómo funciona? 14](#_Toc168430352)

[Memoria Cache L1: 14](#_Toc168430353)

[*Cache L1 Unificado y Cache L1 Dividido* 14](#_Toc168430354)

[Memoria Cache L2: 14](#_Toc168430355)

[Memoria Cache L3: 15](#_Toc168430356)

[Principio de localidad: Impacto en el rendimiento 15](#_Toc168430357)

[Funcionamiento básico 15](#_Toc168430358)

[Elementos de diseño de la memoria caché 16](#_Toc168430359)

[*Direcciones del cache* 16](#_Toc168430360)

[*Longitud* 17](#_Toc168430361)

[*Función de Mapeo de direcciones (Mapping function)* 17](#_Toc168430362)

[Tipos de memoria caché 19](#_Toc168430363)

[*Según su uso* 19](#_Toc168430364)

[*Según el algoritmo de reemplazo* 19](#_Toc168430365)

[*Según la política de emplazamiento* 19](#_Toc168430366)

[*Según la política de escritura* 19](#_Toc168430367)

[Latencia, ancho de bus y falta de caché 20](#_Toc168430368)

[Subsistema de Entrada/Salida (input/output) 20](#_Toc168430369)

[Modulo I/O 20](#_Toc168430370)

[Funciones de un módulo de I/O 21](#_Toc168430371)

[Control y timing 21](#_Toc168430372)

[Comunicación con el procesador. 21](#_Toc168430373)

[Comunicación con el dispositivo 21](#_Toc168430374)

[Buffering de datos 21](#_Toc168430375)

[Detección de errores 21](#_Toc168430376)

[Manejador de interrupciones de I/O 22](#_Toc168430377)

[DMA – Acceso directo a memoria. 22](#_Toc168430378)

[Funcionamiento 22](#_Toc168430379)

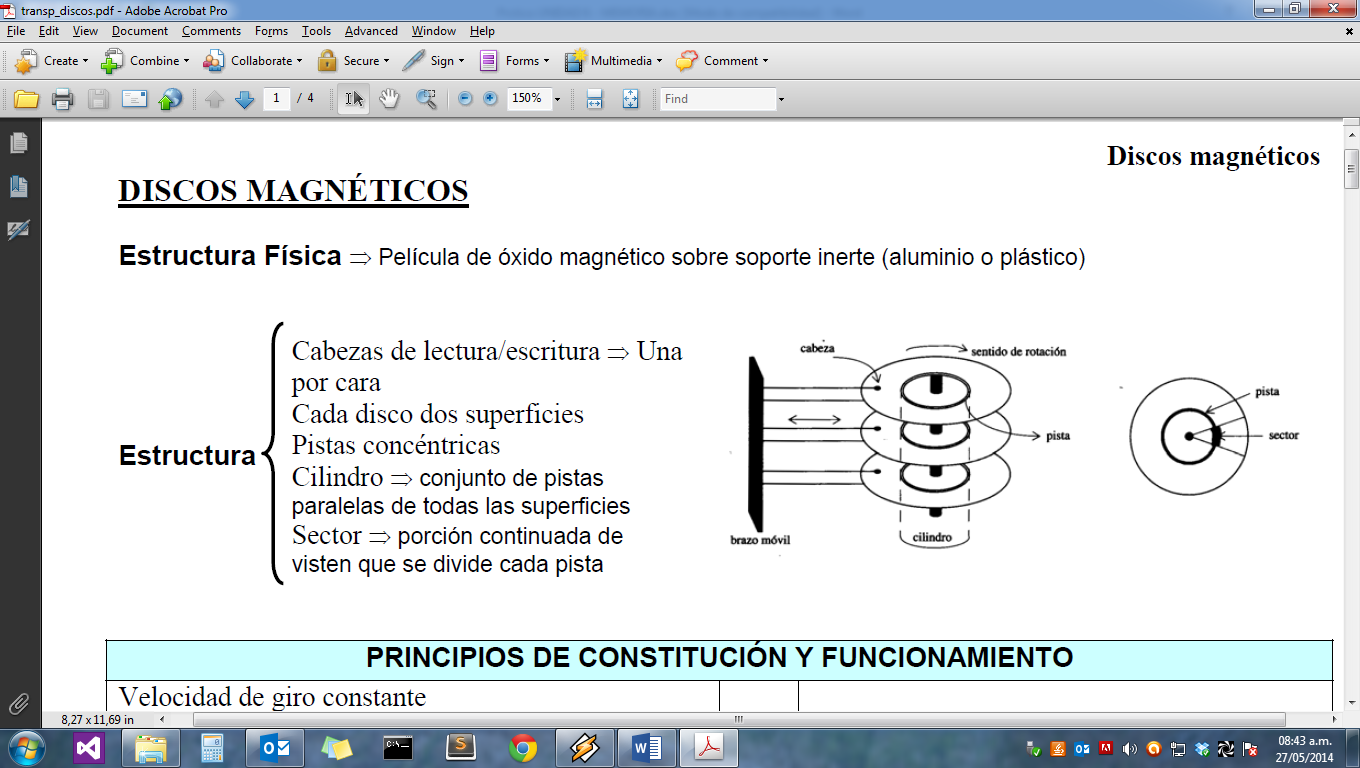
# Discos

## ¿Qué es un disco duro?

Un disco duro es un dispositivo en el que el almacenamiento de la informacion es permanente (almacenamiento no volátil), no necesita de un aporte constante de energía para conservar la informacion, y que puede alterarse en cualquier momento para ser reutilizado, dado que posee miles de ciclos antes de la aparición de errores.

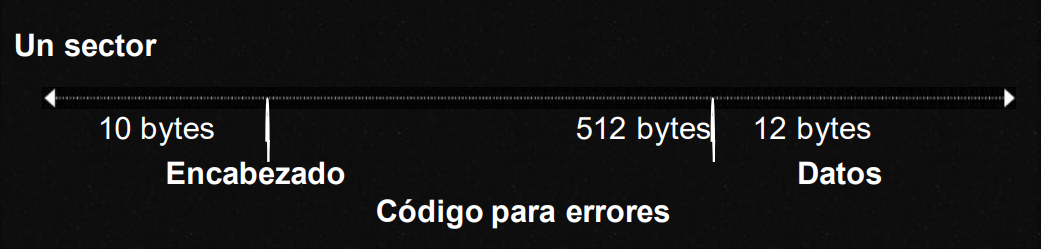
Su funcionamiento se basa en el sistema de grabación magnética.

## Estructura física de un disco duro



* El disco duro es un dispositivo magnético y mecánico, con partes móviles, siendo por tanto más delicado que otros sistemas del almacenamiento.
* Los datos se almacenan sobre una serie de discos o platos (entre 2 y 4 normalmente) que están recubiertos de una fina capa de material magnético.
* Estos platos están situados en un eje común e impulsados por un motor a grandes velocidades, 7200 rpm en los modelos actuales.
* La superficie de los platos debe de estar absolutamente libre de todo elemento contaminante.
* Los datos se distribuyen sobre los platos en miles de círculos concéntricos que se denominan **pistas.**
* A su vez, cada una de estas pistas está dividida en cientos de zonas contiguas de igual capacidad denominadas sectores, el estándar actual es de 512 bytes por sector.

**Sector típico:**



* **Encabezado** con informacion para sincronizar la lectura, identifica r el sector.
* **Datos** con longitud en bytes expresada usualmente como potencia de 2.
* **Código para errores** con informacion para detectar y/o corregir posibles errores.

## Velocidad de giro del disco

Velocidad angular a la que giran los platos, medida en revoluciones por minuto (rpm)

Los discos duros modernos de gama media tienen una velocidad de 7200 rpm. Los discos duros portátiles ofrecen una velocidad menor, de 5400 rpm.

Cuanto mayor sea la velocidad de rotación tardaremos menos en situarnos en el sector en el que se encuentra la informacion requerida.

### Velocidad interna

Este parámetro esta influenciado por la velocidad de giro y por la velocidad de las partes mecánicas. Corresponde al flujo máximo de datos que se puede leer/escribir en un determinado momento

Hay que tener en cuenta que a medida que el disco se llena este se vuelve más lento.

Tenemos en cuenta dos velocidades:

* **Velocidad máxima de transferencia interna**: valor en condiciones teóricas y para cantidades mínimas de datos.
* **Velocidad sostenida**: indica el rendimiento al leer contenido en cantidades apreciables de datos,

### Tamaño de Buffer o Cache

Cumple la función de memoria intermedia, entre la parte “rápida” del disco (controladora) y la parte “lenta” (partes internas, mecánicas y magnéticas), almacenando temporalmente la informacion mientras que está esperando ser procesada.

### Velocidad externa

Velocidad a la que la interfaz transmite los datos entre la “salida” del disco duro (buffer) y el resto de los componentes del PC.

## Tipos de acceso

* Aleatoria: El disco graba la informacion en cualquier parte del disco
* Secuencial: El cabezal del disco mecánico una vez que se ubicó dónde queremos la informacion, va a grabar la informacion uno al lado del otro

## Tiempos

Al tratarse de un dispositivo mecánico, el acceso a los datos lleva cierto tiempo, milisegundos (ms) frente a los nanosegundos (ns) de un dispositivo sin partes móviles.

### Tiempo de búsqueda medio

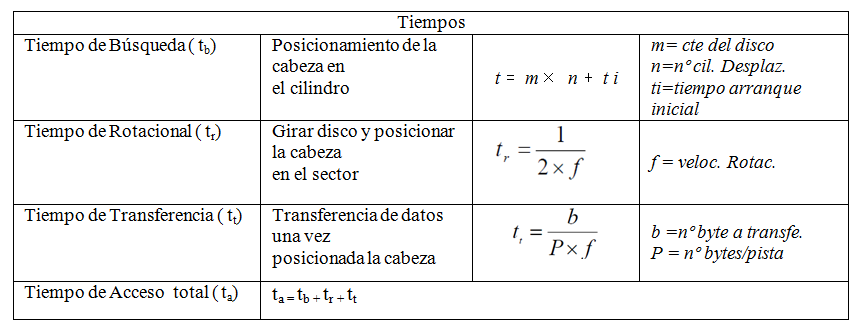
Tiempo medio que tarda la aguja en situarse en la pista deseada. Ronda los 8.5ms.

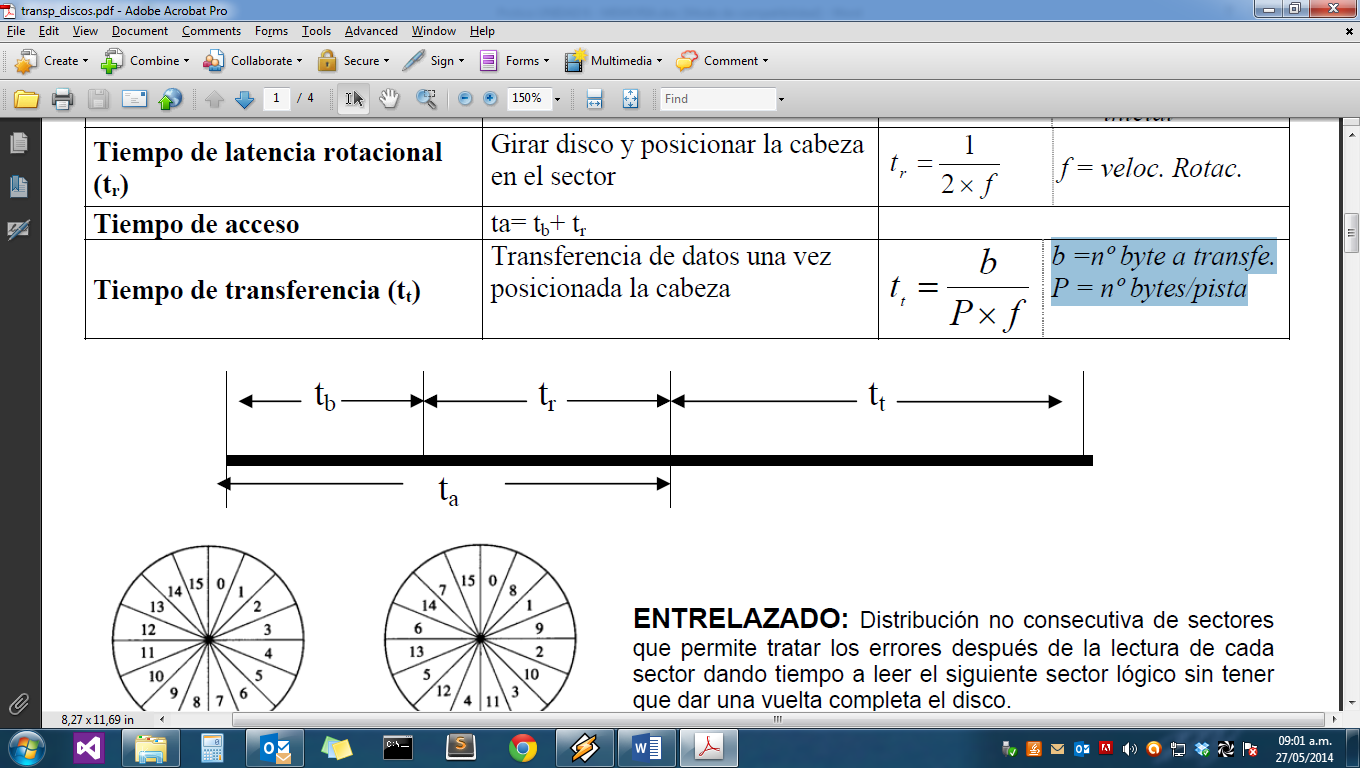
### Latencia rotacional media:

Al llegar la cabeza a la **pista** que contiene el sector que buscamos, esta tiene que encontrarlo. Este valor representa el tiempo medio que tarda en alcanzarse el sector requerido.

### Tiempo medio de acceso

Tiempo que tarda la cabeza en situarse en la **pista y el sector** adecuado. Es la suma del tiempo de búsqueda medio, la latencia rotacional media y del tiempo de escritura/lectura.





## Cálculo de la capacidad del disco

*Caras = Superficie.*

## Tipos de interfaces

### Interfaz ATA/IDE

Es la más utilizada de la historia del PC.

Las siglas IDE hacen referencia a una de las características más importantes de esta tecnología, gran parte de la circuitería lógica de control del disco se encuentran en el propio disco, haciendo que la compatibilidad este casi garantizada.

Los discos duros ATA/IDE se distribuyen en canales, cada uno de los cuales emplea un cable plano con un máximo de dos dispositivos por canal.

**Habitualmente existen tres maneras de configurar un disco duro:**

1. Maestro (master): Dispositivo principal, tiene preferencia a la hora del arranque del sistema operativo. Si hay otro dispositivo, tiene que ser esclavo.
2. Esclavo (Slave): Dispositivo secundario. Debe haber otro dispositivo como maestro.
3. Selección por cable (cable select): El dispositivo será maestro o esclavo en función de su posición en el cable. Si el dispositivo es el único en el cable, debe estar situado en la posición como maestro. Tiene el inconveniente de que mientras se accede a un dispositivo el otro no se puede usar.

### Interfaz SCSI

Contemporánea a la interfaz ATA/IDE, se reservó para equipos de gama media/alta debido a que era bastante más avanzada y, por tanto, más costosa.

Tenía problemas de compatibilidad puesto que necesita de una placa base especial con el controlador SCSI.

### Interfaz Serial ATA (SATA)

La velocidad de transferencia de este interfaz es de 150 MB/s en el caso de SATA/150 o SATA I, de 300 MB/s en el caso de SATA/300 o SATA II, y de 600 MB/s en el caso de SATA/600 o SATA III.

En cuanto a las **conexiones**, la interfaz SATA simplifica bastante la instalación del dispositivo, ya que cada disco posee su propio cable de datos evitando así la necesidad de los jumpers, puesto que todos los discos duros se comportan siempre como maestros.

Los dispositivos SATA emplean dos cables, un conector eléctrico y un conector de datos de 7 hilos:

* 2 para mandar datos en uno y otro sentido
* 2 para indicar la recepción
* 3 de tierra

## 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.

## SSD

Son dispositivos de almacenamiento de datos que usa memoria no volátil como las memorias Flash para almacenar los datos, en lugar de los platos giratorios magnéticos de los discos duros convencionales.

Las unidades SSD son compatibles con las interfaces de disco duro tradicionales, como SATA o SAS, y tienen un formato de disco duro familiar, como 3.5m; 2,5 o 1,8 pulgadas.

### Ventajas

* Son muy rápidos ya que no tienen partes móviles
* Consumen menos energía
* Menor ruido
* Resistente a las caídas gracias a carecer de partes móviles
* Borrado de datos más seguro

### Desventajas

* Vida limitada a un determinado número de ciclos de escritura
* La capacidad, que a día de hoy es menor que en los discos duros convencionales, y sobre todo su precio.
* Falta de estándares: los estándares utilizados por las unidades de disco duro, cuya eficiencia se ha probado durante largo tiempo, no se aplican de igual forma cuando se trabaja con la tecnología flash NAND.

# Sistema de interconexión de BUSES

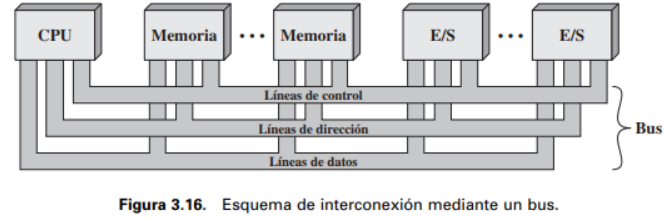
Un bus es un camino de comunicación entre dos o más dispositivos. Una característica clave de un bus es que se trata de un medio de transmisión compartido. Usualmente, un bus está constituido por varios caminos de comunicación, o líneas.

Cada línea es capaz de transmitir señales binarias representadas por 1 y 0. En un intervalo de tiempo, se puede transmitir una secuencia de dígitos binarios a través de una única línea. Se pueden utilizar varias líneas del bus para transmitir dígitos binarios simultáneamente (en paralelo)

El bus que conecta los componentes principales del computador (procesador, memoria, E/S) se denomina bus del sistema (system bus).

Se pueden clasificar en tres grupos funcionales:

* Bus de datos
* Bus de direcciones
* Bus de control.

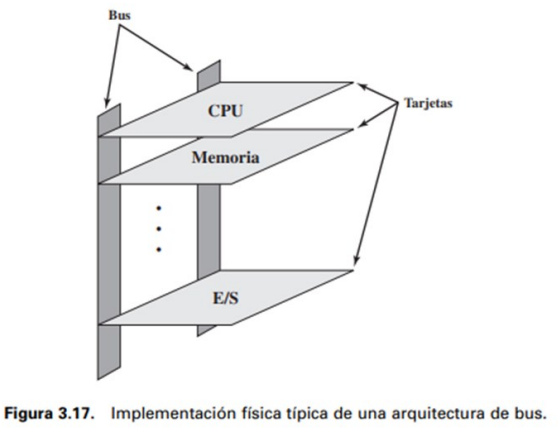


## Funcionamiento del bus:

1. Obtener el uso del bus.
2. Transferir el dato a través del bus.

Si un módulo desea pedir un dato a otro modulo, debe:

1. Obtener el uso del bus
2. Transferir la petición al otro modulo mediante las líneas de control y dirección apropiadas. Después debe esperar a que el segundo modulo envié el dato.



**La anchura del bus es un factor clave a la hora de determinar las prestaciones del conjunto del sistema.**

Por ejemplo, si el bus de datos tiene una anchura de 8 bits y las instrucciones son de 16bits, entonces el procesador debe acceder al módulo de memoria dos veces por cada ciclo de instrucción.

La anchura del bus de direcciones determina la máxima capacidad de memoria posible en el sistema. Además, las líneas de direcciones generalmente se utilizan también para direccionar los puertos de E/S.

**Las líneas de control** se utilizan para controlar el acceso y el uso de las líneas de datos y de direcciones. Las señales de control transmiten tanto ordenes como informacion de temporización entre los módulos del sistema. Algunas líneas de control típicas son:

**Escritura en memoria (Memory write):** hace que el dato del bus se escriba en la posición direccionada.

**Lectura de memoria (Memory read):** hace que el dato de la posición direccionada se sitúe en el bus.

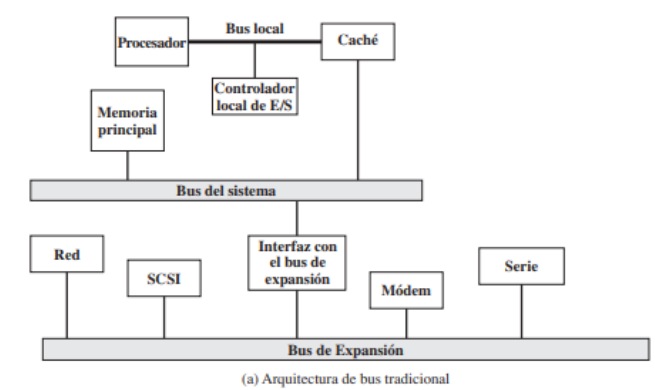
## Jerarquía de buses múltiples:

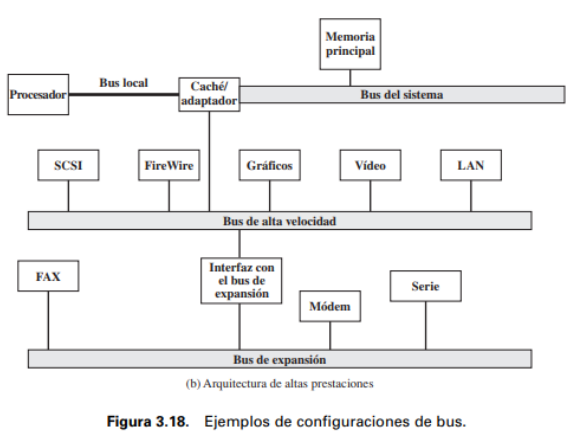
Si se conecta un gran número de dispositivos al bus, las prestaciones pueden disminuir. Hay dos causas principales:

1. En general, a mas dispositivos conectados al bus, mayor es el retardo de propagación.
2. El bus puede convertirse en un cuello de botella a medida que las peticiones de transferencia acumuladas se aproximan a la capacidad del bus.

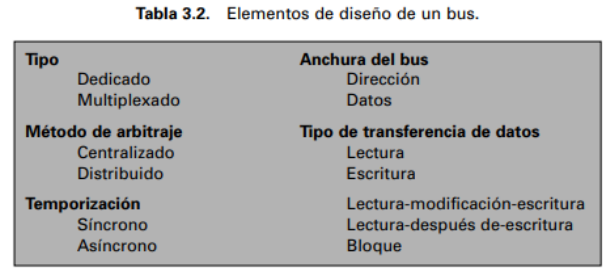
Este problema se puede resolver en alguna medida incrementando la velocidad a la que el bus puede transferir los datos y utilizando buses mas anchos (por ejemplo, incrementando el bus de datos de 32 a 64bits).

El uso de caché alivia la exigencia de soportar los accesos frecuentes del procesador a la memoria principal.





## Elementos de diseño de un bus



### *Tipos*

#### Tipo dedicado

Una línea de bus dedicada esta permanentemente asignada a una función o a un subconjunto físico de componentes del computador. Un ejemplo de dedicación funcional, común en muchos buses, es el uso de líneas separadas para direcciones y para datos.

#### Tipo Multiplexado:

La ventaja del multiplexado en el tiempo es el uso de menos líneas, cosa que ahorra espacio y, normalmente, costes. La **desventaja** es que se necesita una circuitería mas compleja en cada módulo. Además, existe una posible reducción en las prestaciones debido a que los eventos que deben compartir las mismas líneas no pueden producirse en paralelo.

### *Métodos de arbitraje*

En un esquema **centralizado**, un único dispositivo hardware, denominado controlador del bus o arbitro, es responsable de asignar tiempos en el bus. El dispositivo puede estar en un modulo separado o ser parte del procesador.

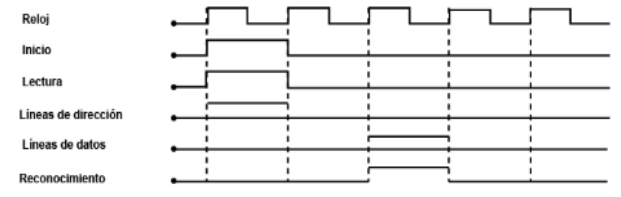
En un esquema **distribuido**, no existe un controlador central. En su lugar, cada módulo dispone de lógica para controlar el acceso y los módulos actúan conjuntamente para compartir el bus.

En ambos métodos de arbitraje, el propósito es designar un dispositivo, el procesador o un modulo de E/S como maestro del bus.

### *Temporización*

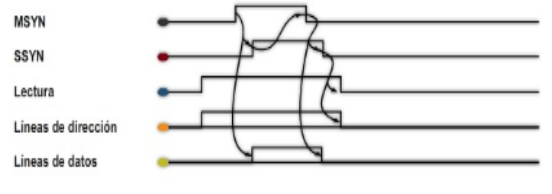
#### Temporización síncrona:

Los eventos en el bus están determinados por el ciclo de reloj (un pulso de reloj, un evento). El bus incluye una línea de reloj a través de la que se transmite una señal de sincronización.



#### Temporización asíncrona:

Con la temporización asíncrona, al no haber una señal de reloj que determine el principio y fin del tiempo de actuación, la presencia de un evento en el bus es consecuencia, y depende, de que se produzca un evento previo que indique que una fase se ha terminado y se puede proceder a la siguiente fase de la acción (siguiente evento del bus)

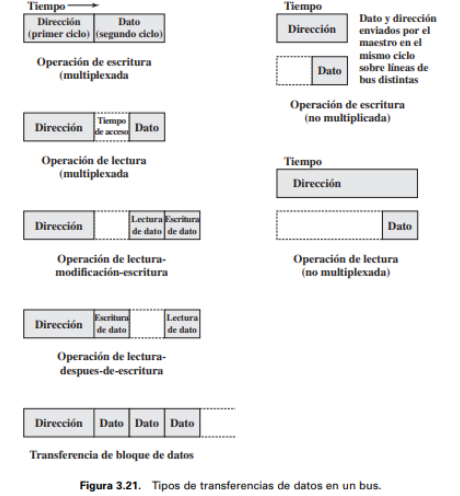


### *Anchura del bus*

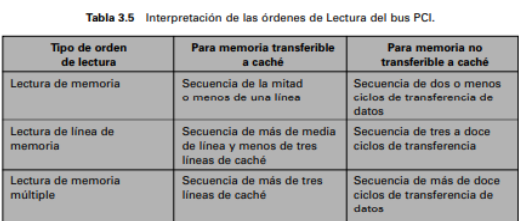
La anchura del bus de datos afecta a las prestaciones del sistema: cuanto mas ancho es el bus de datos, mayor es el numero de bits que se transmiten a la vez.

### *Tipo de transferencia de datos*

Por último, un bus permite varios tipos de transferencia de datos. Todos los buses permiten tanto transferencia de escritura (dato de maestro a esclavo) como de lectura (dato de esclavo a maestro)



* El uso de las tres ordenes de lectura de memoria se resume en la siguiente tabla:



# Memorias:

## Tipos de memoria de computadora:

1. [Memoria RAM (Random Access Memory – Memoria de Acceso Aleatorio):](#_Memoria_RAM_(Random)
2. Memoria ROM (Read Only Memory)
3. [Memoria Cache](#_Memoria_Cache)
4. [Memoria RAM Dinámica o DRAM](#_Memoria_DRAM:)
5. [Memoria RAM Estática o SRAM](#_Memoria_SRAM)
6. Memoria Flash
7. [Memoria Virtual](#_Funcionamiento_básico)
8. Memoria de video o VRAM

### Memoria RAM (Random Access Memory – Memoria de Acceso Aleatorio)

La memoria RAM es el tipo de memoria más importante de la computadora, su nombre representa las siglas de Random Access Memory; la razón de dicho nombre es porque la misma está dividida en celdas de memoria donde se almacenan cada uno de los bits o pulsos eléctricos (que representan los 0 y 1) y a las cuales se puede acceder directamente indistintamente de su posición o dirección.

La memoria RAM está dividida en celdas en donde se almacenan temporalmente cada uno de los bits que componen los bytes de la informacion con la que trabaja el microprocesador. Cada una de las celdas de la memoria que almacena un bit (1 o 0), se encuentra formada por un transistor y un capacitor. Mientras los capacitores sostienen los bits de informacion, los transistores actúan como interruptores que permiten a su controlador de memoria leer o modificar la informacion (los bits) que contienen cada una de las celdas. A este tipo de memoria con celdas formadas por un capacitor y un transistor se la denomina Dynamic Random Access Memory (Memoria de Acceso Aleatorio Dinámica) o por sus siglas **DRAM.**

El elemento básico de una memoria de semiconductor es la **celda de memoria (“Memory cell”)** cuyas características son:

* Presentan dos estados estables o (semiestables). Se usan para representar el 0 y el 1.
* Se puede escribir (al menos una vez), para setear el estado.
* Se los puede leer (se “siente” el estado)

#### Memoria DRAM:

Un capacitor, es un dispositivo que almacena carga eléctrica, funcionando como un medio para retener electrones. En el contexto del almacenamiento de datos, la carga de un capacitor se utiliza para representar un bit. Un bit con valor 1 se corresponde con un capacitor cargado con electrones, mientras que un bit con valor 0 se representa por un capacitor sin carga. Sin embargo, los capacitores presentan el inconveniente de la fuga de carga debido a su naturaleza física, lo que implica que no pueden mantener su estado de carga indefinidamente. Por esta razón, el controlador de memoria tiene que recargar los capacitores con 1 constantemente. Para eso el controlador tiene que primero leer la memoria y rellenar los capacitores aun cargados con electrones (o sea los que representan a los bits 1) antes de que se descarguen. Esta operación de carga ocurre automáticamente varias veces por segundo.

Este proceso de recarga constante es de donde proviene el nombre **Dynamic RAM (DRAM)**, ya que debe ser dinámicamente recargada todo el tiempo para no perder la informacion que almacena. El punto en contra de este tipo de tecnología es que la hace más lenta, pero a su vez mas económica y se puede tener en grandes cantidades a diferencia de la Memoria Cache.

##### Celdas de la memoria DRAM:

La memoria está formada por celdas de bits distribuidas en una grilla bidimensional. Las celdas de memoria (cada una compuesta por un capacitor y un transistor microscópicos) están grabadas en laminas u obleas de silicio distribuidas en una matriz de columnas llamadas bitlines (líneas de bits y filas llamadas wordlines (líneas de palabra), la intersección de una columna y una fila determinan la dirección de una celda de memoria.

###### Proceso de Lectura y escritura

La Memoria DRAM funciona enviando una señal o carga eléctrica RAS (Row Address Strobe - Señal de dirección de fila) a la fila donde se encuentran las celdas cuyos transistores se van a activar para poder permitir pasar a los electrones que se almacenaran en los capacitores de las celdas. Una vez que la fila ha sido seleccionada con la señal RAS; para escribir los bits que representan a los 1, se envían cargas eléctricas CAS (Column Address Strobe o Señal de dirección de columna) a través de las correspondientes columnas.

Por otro lado, para leer las celdas de memoria de una fila, una vez seleccionado, mediante la CAS, un detector de carga llamado amplificador sensor (sense amplifier) determina el nivel de carga del capacitor de la celda que se está leyendo; 1 o 0.

A su vez un controlador para mantener los capacitores de las celdas cargados, tiene que leer cuales son las celdas cargadas y luego las vuelve a cargar. Todo este proceso de lectura y escritura ocurre en periodos de nanosegundos.

Resumen: Es el tipo de memoria cuyas celdas de almacenamiento de bits están compuestas cada una por un capacitor y un transistor, requiriendo constante recarga o refrescamiento.

#### Memoria SRAM

La memoria estática RAM (SRAM) utiliza una tecnología diferente a aquella de la memoria DRAM. En la memoria estática cada celda de bit está compuesta por cuatro o seis transistores y algunos circuitos; logrando que no sea necesario refrescar la informacion constantemente como sucede con la memoria Dinámica. Eso hace que la memoria SRAM sea más rápida que la DRAM, sin embargo, por tener más partes o componentes en cada celda, las mismas ocupan mayor espacio que las celdas de la DRAM, obteniéndose menor cantidad de bits de almacenamiento en un chip del mismo tamaño y, por otro lado, encareciendo los costos por la mayor cantidad de partes.

Dado que la memoria SRAM es más costosa que la memoria DRAM, se la utiliza para la memoria cache del microprocesador

Resumen: Cuenta con celdas compuestas cada una por cuatro a seis transistores; sin capacitores, por lo que no requieren recarga o refrescamiento de datos constantes.

#### Nomenclatura para las memorias DDR

Tomando como ejemplo la siguiente memoria: **4GB DDR3-2133 PC3-17000**

Podemos decir que se trata de un modulo de 4GB de tipo DDR3, frecuencia aparente o efectiva de trabajo de 2133Mhz o para ser más correctos, 2133 megatransferencias/s y una tasa de transferencia de datos máxima de 17000MB/s o, en dual channel: 17000 x 2

### Memoria Cache

#### ¿Qué es la memoria cache L1 L2 L3 y cómo funciona?

La memoria cache está diseñada para hacer que el acceso a la informacion por parte del procesador sea lo más rápida posible. Está ubicada dentro del procesador, reduciendo significativamente el tiempo necesario para acceder a los datos en comparación con la memoria RAM.

Jerarquía: La CPU busca primero los datos en la cache L1. Si no los encuentra (un fallo o miss), busca en la cache L2, y luego en la L3. Si no están presentes en la L3, se accede a la memoria RAM, y como último recurso, a la memoria secundaria (SWAP o memoria virtual)

##### Memoria Cache L1:

La cache L1 es la más cercana a los núcleos de la CPU. Tiene una velocidad de alrededor de 1150 GB/s y una latencia de tan solo 0.9 ns. Suele tener un tamaño de aproximadamente 256KB en total, aunque puede variar dependiendo de la potencia y costo de la CPU (procesadores de gama alta pueden tener más capacidad)

Cada núcleo de la CPU tiene su propia cache L1.

###### *Cache L1 Unificado y Cache L1 Dividido*

* Cache L1 unificada: Una sola cache almacena tanto instrucciones como datos.
  + Ventajas:
    - Tasa de aciertos potencialmente mayor para un tamaño dado, ya que nivela automáticamente la carga entre instrucciones y datos.
    - Diseño e implementación más sencillos.
* Cache L1 dividida: Una cache dedicada para instrucciones y otra para datos.
  + Tendencia actual, especialmente en procesadores superescalares que enfatizan la ejecución paralela de instrucciones
  + Ventaja clave: elimina la competencia entre el procesador de instrucciones y la unidad de ejecución por acceder a la caché.
    - En caches unificadas, las peticiones de la unidad de ejecución para acceder a datos pueden bloquear temporalmente las peticiones del precaptador de instrucciones, degradando el rendimiento.
  + Las caches divididas evitan este cuello de botella, permitiendo un uso más eficiente del cauce segmentado de instrucciones.

##### Memoria Cache L2:

Es el siguiente nivel después de la cache L1. Tiene una velocidad de unos 470GB/s y una latencia de aproximadamente 2,8ns. Varía entre 256KB y 18MB dependiendo del procesador. No está dividida en instrucciones y datos como la L1, y cada núcleo de la CPU suele tener su propia cache L2.

Para el caso de una cache L2 externa, muchos diseños no usan el bus del sistema como camino para las transferencias entre la cache L2 y el procesador, sino que emplea un camino de datos aparte para reducir el trafico en el bus del sistema. Gracias a la continua reducción de dimensiones de los componentes de los procesadores, es fácil encontrar que lo incorporan en el propio chip.

Con la creciente disponibilidad de superficie para cache en el propio chip, en la mayoría de los microprocesadores modernos se ha llevado la caché L2 al procesador, y se añade una cache L3.

##### Memoria Cache L3:

Se encuentra en el chip del procesador, siendo el ultimo nivel de cache dentro de la CPU. Es la más lenta de las tres, con una velocidad de más de 200GB/s y una latencia de alrededor de 11ns. Los procesadores actuales tienen al menos 4MB de cache L3, y pueden llegar hasta 64MB. Normalmente, la L3 se distribuye con unos 2MB por cada núcleo, pero no está dentro de cada núcleo, sino que se comunica a través de un bus de datos.

##### Principio de localidad: Impacto en el rendimiento

La eficiencia de la caché se mide en términos de aciertos (hits) y fallos (misses). Un alto índice de aciertos mejora el rendimiento del sistema, ya que la CPU pasa menos tiempo esperando datos.

La cache explota los conceptos de localidad (ubicación cercana a los núcleos de procesamiento) y velocidad (tiempos de acceso muy rápidos) para mejorar el rendimiento de la CPU.

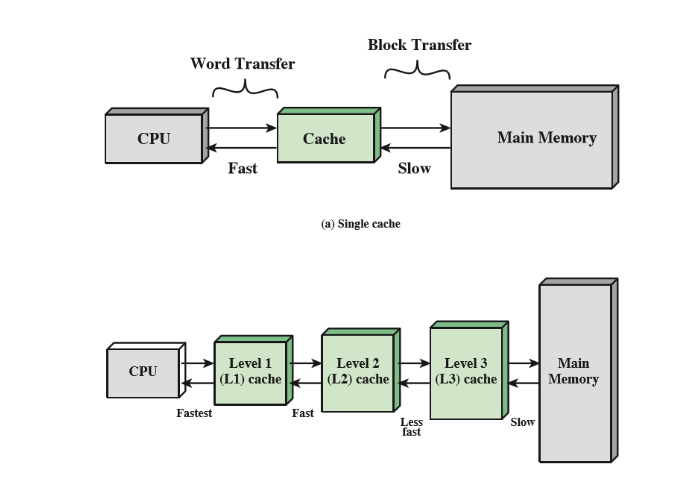
Aunque la cantidad de caché es muy pequeña en comparación con la RAM (medida en kilobytes o megabytes frente a gigabytes en la RAM), su cercanía y velocidad compensan el tamaño reducido.

##### Funcionamiento básico

Cuando el procesador necesita acceder a una informacion, solicita estos datos a la memoria RAM. El proceso de obtención de datos desde la RAM toma varios ciclos de reloj debido a la **latencia**, lo que puede significar nanosegundos de espera. La tasa a la cual el procesador puede ejecutar las instrucciones está claramente limitada por el tiempo de ciclo de la memoria. La solución es explotar el principio de localidad y proveer de una pequeña y rápida memoria entre el procesador y la memoria principal (cache).

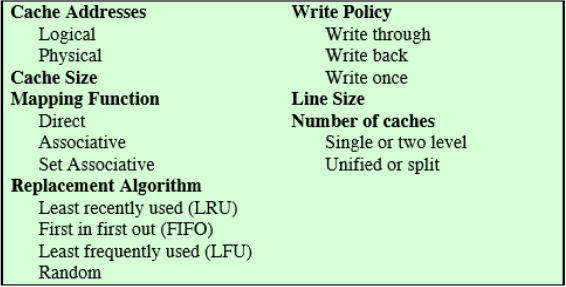
La memoria cache se introdujo para solucionar el problema del tiempo de espera del procesador. El cache contiene una copia de la memoria principal. Almacena temporalmente los datos e instrucciones mas utilizados para que el procesador pueda acceder a ellos rápidamente sin tener que esperar a que vengan desde la RAM.

La memoria cache está organizada jerárquicamente para maximizar la eficiencia. Si la informacion no se encuentra en ninguno de los niveles de caché, el procesador recurre a la memoria RAM, y si es necesario, a la memoria secundaria (swap o virtual)



##### Elementos de diseño de la memoria caché

* Direcciones del caché (Cache Addresses)
* Longitud (Cache size)
* Función de mapeo de direcciones (Mapping Function)
* Algoritmo de reemplazo (reemplacement algorithm)
* Política de escritura (Write Policy)
* Tamaño de la línea (line size)
* Numero de niveles (Number fo caches)



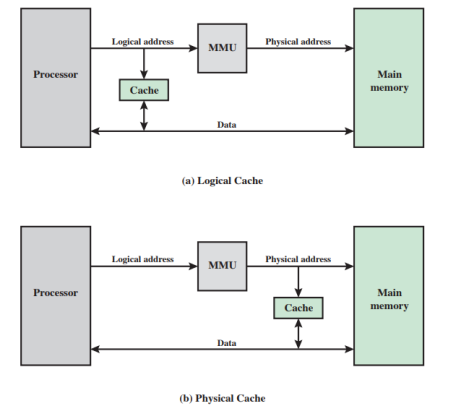
###### *Direcciones del cache*

Cuando se usa una memoria virtual, las instrucciones de la CPU contienen direcciones virtuales que deben traducirse a direcciones físicas por la Unidad de Gestión de Memoria (MMU) antes de acceder a la memoria principal.

**Una caché lógica o virtual** almacena datos utilizando direcciones virtuales, permitiendo acceder al caché sin pasar por la MMU, lo que es más rápido.

**Una caché física** almacena tatos usando direcciones físicas de memoria principal.

La ventaja del caché lógico es mayor velocidad al no requerir traducción de direcciones. La desventaja, es que, en sistemas de memoria virtual, cada aplicación tiene el mismo espacio de direcciones virtuales comenzando en 0. Así, la misma dirección virtual en dos aplicaciones se refiere a dos direcciones físicas diferentes. Esto requiere vaciar la caché al cambiar de aplicación, o agregar bits adicionales para identificar el espacio de direcciones virtuales al que pertenece cada línea de caché.



###### *Longitud*

Se busca que la longitud sea lo suficientemente chica como para que el costo por bit sea cercano al costo por bit de la memoria principal. Y lo suficientemente grande como para que el tiempo promedio de acceso sea cercano al del cache en forma individual. Cuando mas grande es el cache, mayor numero de componentes (“puertas lógicas”) se requieren para el direccionamiento en el cache, lo cual genera que cache grandes tienden a ser un poco mas “lentos” que cache chico.

###### *Función de Mapeo de direcciones (Mapping function)*

Cada línea del cache almacena un numero de bloques de la memoria principal. Como hay muchos más bloques en la memoria principal que líneas en el cache, se necesita de una función o mecanismo que pueda mapear bloques de memoria en las líneas del cache y pueda determinar en qué línea de cache se encuentra un bloque determinado de memoria.

Las dos técnicas mas conocidas son el **MAPEO DIRECTO Y EL MAPEO ASOCIATIVO.**

Mapeo directo:

Se mapea cada bloque de memoria principal en solo una posible línea del cache. El mapeo se expresa como:

i = j modulo m

i = número de línea del cache

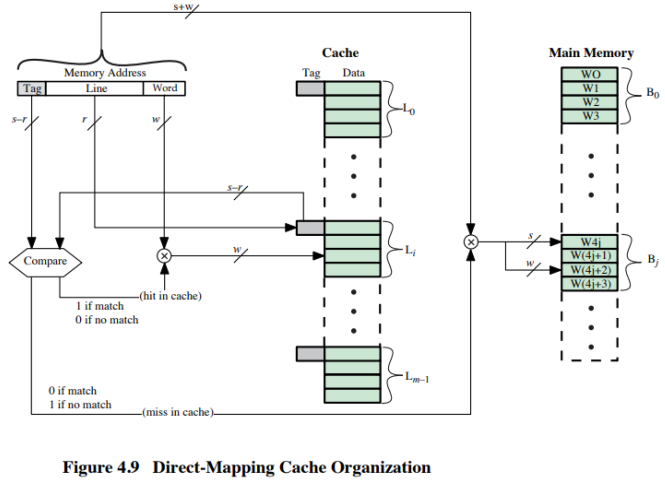
j = numero de bloques en memoria principal

m = cantidad de líneas en el cache

**La función de mapeo directo** se implementa usando la dirección de memoria principal.

Los **w bits** menos significantes de la dirección identifican a una única palabra de memoria dentro del bloque de palabras

Los **s bits** restantes de la dirección especifican una de los 2s bloques de memoria. Los bits mas significantes dentro de estos s bits, representan una **etiqueta o TAG de r bits que identifica una de las m=2r líneas del cache.**



El uso de una **parte de la dirección de memoria como numero de línea del cache** provee un único mapeo de cada bloque de memoria en el cache. Cuando se lee el bloque en su línea asignada, es necesario **TAGEAR (etiquetar) el DATO** para poder *diferenciarlo de los otros bloques* que pueden ocupar la misma línea del cache.

La desventaja que tiene el mapeo directo, es que asigna líneas fijas del cache a determinados bloques de memoria

Mapeo Asociativo (Correspondencia Asociativa)

Esta técnica permite que cada bloque de memoria principal pueda ser almacenado en cualquiera de las líneas del cache. La lógica de control del cache interpreta a la dirección de memoria principal dividida en dos campos:

**El campo de TAG o etiqueta, que identifica el bloque de memoria** principal (para saber si el bloque se encuentra en el cache, la lógica del cache **realiza una comparación simultanea entre todos los TAGS** o etiquetas de todas las líneas del cache)

[Algoritmo de reemplazo (tipos de memoria cache](#_Según_el_algoritmo)):

[Política de escritura](#_Según_la_política) (tipos de memoria cache)

##### Tipos de memoria caché

Los tipos de memoria caché se pueden catalogar teniendo en cuenta diversos parámetros, como, por ejemplo:

###### Según su uso

Según para que se esta usando la memoria caché se pueden dar los siguientes casos:

* **Scratchpad Memory:** Es un tipo de memoria local muy rápida para el almacenamiento temporal de algunos datos u operaciones de cálculo, etc. Esta memoria es similar a una L1 y se encuentra próxima a la ALU.
* **Buffer Victima o victim cache:** Es una pequeña caché completamente asociativa de tamaño pequeño que actúa como repositorio de datos e instrucciones descartados en la L1.
* **Assist cache:** similar a la anterior, completamente asociativa y con algoritmo de sustitución FIFO.
* **Trace cache o L0:** son memorias de muy pequeño tamaño, cercanas a la unidad de decodificación de instrucciones y con objetivo de almacenar micro-operacion generado por instrucciones recientes, para no tener que decodificarlas la próxima vez que se necesiten.

###### *Según el algoritmo de reemplazo*

Es la forma en la que van eliminando datos para dejar espacio para otros dentro de la memoria cache:

* **Aleatorio o RR:** se elige de una forma pseudoaleatoria.
* **LRU:** relacionado con el principio de localidad temporal, sustituyendo el menos reciente.
* **FIFO:** primero en llegar, primero en salir
* **MRU:** reemplaza el más recientemente accedido.
* **PLRU:** en cachés asociativas, generalmente con mas de 4 vías, se implementa este pseudo-LRU.
* **SLRU:** se divide en un segmento de prueba y otro protegido, donde las líneas del protegido están ordenadas desde la mas antigua hasta la mas reciente, y las líneas de un segmento se van pasando al final del protegido para que tengan una segunda oportunidad de acceso.
* **LFU:** con un contador va eliminando los menos accedidos primero.

###### *Según la política de emplazamiento*

Según el nivel de correspondencia u organización se tiene:

* **Correlación directa:** a cada bloque de memoria principal se le asigna un marco de memoria caché concreto
* **Asociativa total:** cualquier posición de memoria principal se puede almacenar en cualquier posición de la caché.
* **Asociativa de n vías (4-way, 8-way, etc.):** en caso de ser 1-way, cada posición de un bloque de memoria principal solo puede almacenarse en una posición concreta de la caché. Con mas vías se mezcla una correspondencia directa a nivel de conjuntos y una correspondencia asociativa a nivel de bloques.

###### *Según la política de escritura*

* **Write-through:** modifica tanto la cache como la memoria principal para que no existan datos caducos y mantener así una coherencia.
* **Write-back:** solo se escribe la caché, y si la línea escrita va a ser reemplazada, también se modificará en la memoria de niveles jerárquicos superiores antes de su reemplazo.

##### Latencia, ancho de bus y falta de caché

Donde primero busca el procesador la siguiente instrucción a procesar, es en la memoria caché. **La latencia** es entonces, el tiempo que se tarda en acceder a los datos desde la memoria. Mientras mas lejos y mas lenta, mayor latencia y más tiempo tendrá que esperar la CPU su siguiente instrucción. Así, cuando una instrucción no esta situada en la memoria caché, el procesador debe buscarla directamente en la memoria RAM, a esto se le denomina **falta de caché o cache miss**, es entonces cuando se experimenta un pc más lento.

El ancho de bus también es de gran importancia para la velocidad, ya que esta marca la capacidad para transferir mayores bloques de datos desde la memoria a la CPU.

# Subsistema de Entrada/Salida (input/output)

## Modulo I/O

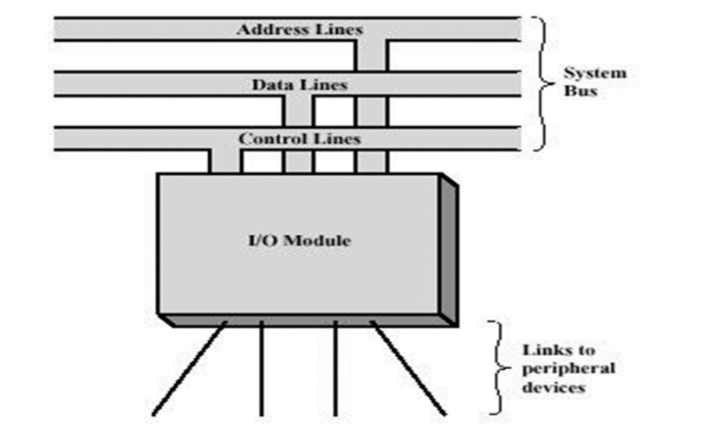
Un modulo de I/O es un componente que permite la comunicación entre la CPU y los dispositivos periféricos. Estos módulos son responsables de adaptar las señales de los dispositivos externos para que sean compatibles con el bus del sistema.

El módulo de I/O, además de tener partes mecánicas y conectores, tiene la lógica para realizar la comunicación entre los periféricos y el bus del sistema.

La tasa de transferencia de datos de los periféricos es mucho mas lenta que el procesador y la memoria. Seria impracticable usar el bus de alta velocidad del sistema para comunicarse directamente con el periférico.

Un módulo de I/O básicamente realiza la:

* Interface con el procesador y la memoria vía el bus del sistema
* Interface con uno o más dispositivos periféricos.



### Funciones de un módulo de I/O

* [Control y timing (temporización)](#_Control_y_timing)
* [Comunicación con el procesador](#_Comunicación_con_el)
* [Comunicación con el dispositivo](#_Comunicación_con_el_1)
* [Buffering de datos](#_Buffering_de_datos)
* [Detección de errores.](#_Detección_de_errores)

#### Control y timing

Los recursos internos como la memoria y el bus del sistema deben ser compartidos entre un numero de actividades entre las que se encuentran los datos de I/O. Se debe coordinar el flujo de trafico entre los recursos internos y los dispositivos externos.

Por ejemplo, el control de la transferencia de datos entre un dispositivo externo y el procesador:

1. El procesador interroga al modulo de I/O para chequear el estado del dispositivo conectado
2. El modulo I/O envía el estado del dispositivo
3. El procesador requiere la transferencia de los datos enviando un comando al módulo de I/O
4. El modulo de I/O obtiene una unidad de datos (ej. 8 o 16 bits) del dispositivo externo
5. Los datos son transferidos desde el modulo de I/O al procesador.

**Cada interacción entre el procesador y el modulo de I/O requiere de una arbitración del bus del sistema.**

#### Comunicación con el procesador.

Por lo general la comunicación con el procesador incluye:

* Decodificación de comandos: El modulo I/O acepta comandos del procesador, típicamente enviadas como señales en el bus de control. Algunos comandos incluyen **parámetros** que se envían sobre el **bus de datos**.
* Datos: los datos se envían por el bus de datos
* Reporte de estados: se usan señales de estados para indicar el estado del dispositivo. Por ejemplo, estado BUSY o READY.
* Reconocimiento de direcciones: Como cada palabra de memoria, también cada dispositivo de I/O tiene su dirección. **El modulo debe reconocer una dirección única para cada periférico que controla**.

#### Comunicación con el dispositivo

Esta comunicación incluye también comandos, informacion de estados, y datos.

#### Buffering de datos

Esto surge debido a la diferencia significativa que hay entre las tasas de transferencias entre el procesador / la memoria y los periféricos.

#### Detección de errores

Por ejemplo, errores mecánicos o eléctricos (ej. Atascamiento de papel de la impresora)

Se puede usar la técnica de detección de errores por control de paridad.

### Manejador de interrupciones de I/O

**Desde el punto de vista del módulo de I/O:**

* Para un input, el modulo de I/O recibe un comando READ del procesador.
* El modulo empieza a leer datos desde el periférico y lo guarda en los registros internos del módulo.
* Envía una señal de interrupción al procesador por las líneas de control del bus.
* El módulo espera hasta que el dato sea requerido por el procesador.
* Una vez recibido el requerimiento del procesador, el modulo pone los datos en el bus de datos y se pone en estado de READY para otra operación de entrada/salida.

**Desde el punto de vista del procesador:**

* El procesador envía un comando READ y se ocupa de otra tarea (por ejemplo, otro proceso o programa)
* Al final de cada ciclo de instrucción el procesador chequea si hubo una interrupción.
* Cuando ocurre una interrupción de I/O, el procesador “salva el contexto” (contador de programa y registros) actual, y procesa o atiende la interrupción
* El procesador LEE los datos del I/O modulo y lo almacena en memoria.
* Luego restaura el contexto del programa que estaba ejecutando previo a la interrupción por I/O

## DMA – Acceso directo a memoria.

Cuando se necesitan mover mayores volúmenes de datos, la técnica mas eficiente es DMA (Direct Access Memory)

DMA requiere de un módulo adicional en el sistema. El DMA toma el control del bus del sistema (el cual tenia el procesador) para poder transferir datos hacia y desde la memoria usando el bus del sistema.

El DMA usa el bus cuando el procesador no lo necesita usar, o podría forzar al procesador que suspenda temporariamente su operación. Esto se llama **ROBO DE CICLOS** (el DMA le esta robando ciclos del bus al procesador)

### Funcionamiento

* Cuando el procesador quiere escribir o leer un bloque de datos, envía un comando al DMA con lo siguiente:
  + READ O WRITE para indicar si es lectura o escritura, usando las líneas de control.
  + La dirección del dispositivo I/O vía líneas de datos.
  + La dirección de memoria inicial para leer o escribir los datos, almacenándola en el registro de direcciones del DMA.
  + La cantidad de palabras (unidades de datos) a transferir, almacenándola en el registro de contador de datos (Data Count) del DMA.
* Una vez configurado, el procesador delega las tareas de I/O al modulo DMA y continua con otras tareas.
* El DMA transfiere el bloque de datos desde o hacia la memoria, una palabra por vez, sin intervención del procesador.
* Cuando finaliza la transferencia, el DMA genera una interrupción para notificar al procesador.
* El procesador solo se involucra al inicio y al final de la transferencia de datos, delegando el trabajo principal al DMA.