Tema 4. Acceso directo a memoria, buses y almacenamiento

Índice

1. Entrada/Salida por acceso directo a memoria (DMA)

Necesidad del *DMA*

- **E/S controlada por programa**: la *CPU* está dedicada exclusivamente a la *E/S*, transfiriendo los datos a la velocidad determinada por el periférico.

- **Interrupciones**: liberan en buena medida a la *CPU* de la gestión de las transferencias, pudiéndose dedicar a ejecutar otras tareas. La velocidad del periférico se puede ver disminuida si las interrupciones no se atiende con la frecuencia suficiente.

- **En las dos alternativas las transferencias de datos deben pasar a través de la *CPU***.
  => velocidad de transferencia limitada por la velocidad a que la *CPU* atiende el periférico

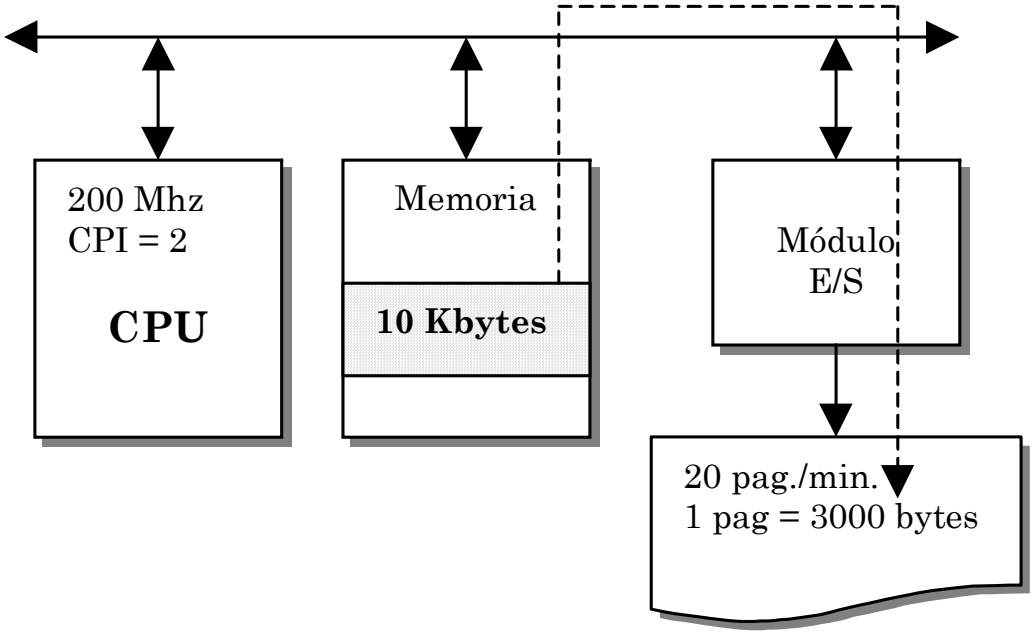
- Ambos procedimientos manifiestan, pues, limitaciones que afectan a la actividad de la *CPU* y a la velocidad de transferencia.

- Para poner de manifiesto las limitaciones de la E/S programada y por interrupción vamos a examinar en las siguientes transparencias dos supuestos de transmisión, una lenta y otra rápida.

1. Entrada/Salida por acceso directo a memoria (DMA)

Supuesto 1: Conexión de un periférico lento (impresora láser)

- La *CPU* opera a *200 Mhz*. y su *CPI* vale *2*.
- La impresora opera a *20* páginas/minuto, con *3.000* caracteres (*bytes*) por página.
- Se trata de imprimir un bloque de *10 Kbytes* ubicado en la memoria.



1. Entrada/Salida por acceso directo a memoria (DMA)

Supuesto 1: tiempo de CPU dedicado a la E/S

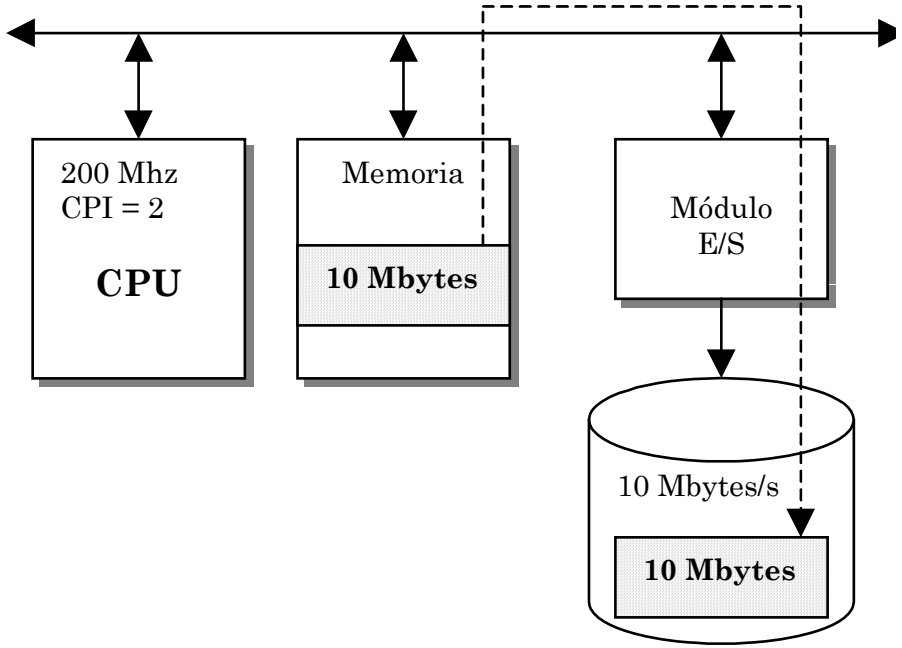
- **Conexión por E/S programada**
  - Una instrucción tarda en ejecutarse *2 * 5 ns = 10 ns*
    (*Tc = 1/Frecuencia = 1/200*10^6 seg. = 1/200*10^6 * 10^{-9} ns = 5 ns*.)
  - La impresora opera a 20*3.000 = 60.000 caract./min. = 1.000 caract./seg. = 1 KB/s.
  - Los 10 KB los imprimirá en 10 segundos. ➔ **La *CPU* está ocupada durante *10* seg.**

- **Conexión por interrupción**
  - Supondremos que son *10* las instrucciones que ejecuta la Rutina de servicio
  - La impresora genera una interrupción cuando está preparada para recibir un carácter.
  - Al transferir *10 KB* se producen *10.000* interrupciones➔ejecuta *100.000* instrucciones.
  - Luego **la *CPU* se ocupa durante** *10^5*10 ns = 10^6 ns=10^6*10^{-9} s. = 0,001 s. =* ***1 miliseg***.
  - **La *E/S* por interrupción ha reducido en 10.000 veces el tiempo que la CPU** está ocupada en atender la impresora.
  - Sin embargo, **la velocidad de la operación de E/S no ha cambiado**, como era de esperar al estar dominada por la velocidad del periférico.

1. Entrada/Salida por acceso directo a memoria (DMA)

Supuesto 2: Conexión de un periférico rápido (disco magnético)

- La *CPU* opera igual que en caso anterior, es decir, a *200 Mhz* y su *CPI* vale *2*.
- El disco opera a una velocidad de transferencia de *10 Mbytes/s*.
- Se trata en este caso de transmitir un bloque de *10 Mbytes* de la memoria al disco



1. Entrada/Salida por acceso directo a memoria (DMA)

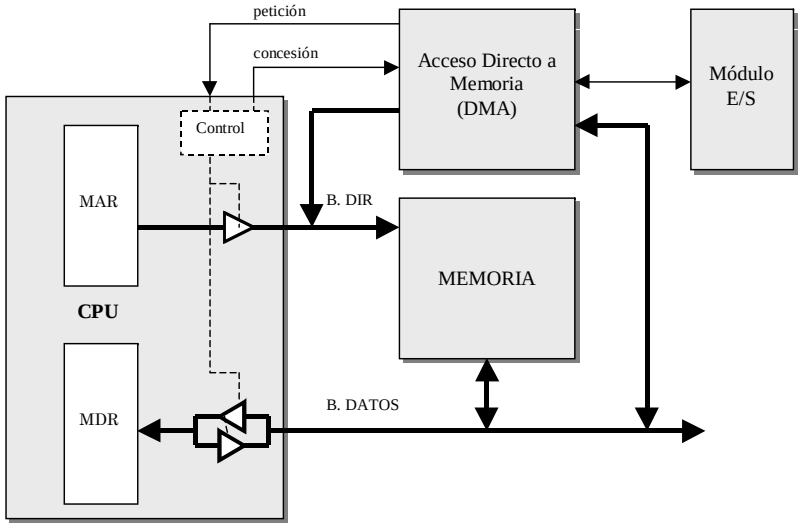
Supuesto 2: tiempo de CPU dedicado a la E/S

- **E/S programada**
  - A la velocidad de *10 Mbytes/s* el disco tarda *1* segundo en recibir los *10 Mbytes*,
  - En *E/S* programada la *CPU* envía un byte cada vez que el disco está preparado para recibirlo ➔ **la *CPU* está ocupada en la tranferencia durante *1 segundo***.
- **E/S por interrupción**
  - Seguimos suponiendo que la rutina de tratamiento ejecuta *10* instrucciones.
  - Como ahora se transmiten 10^7 *bytes* se producirán otras tantas interrupciones ➔ CPU ejecuta en toda la operación $10^7 * 10 = 10^8$ instrucciones.
  - Tiempo de **ocupación *CPU*** = 10^8 instrucciones * *10* ns/instrucción = 10^9 *ns* = ***1 seg.***
- **Conclusiones**
  - En el supuesto 2 las interrupciones no ahorran tiempo de CPU frente a la *E/S* programada porque se ha llegado al límite de velocidad de la línea de interrupción (*10 Mbytes/s*), por encima de esta velocidad la *E/S* por interrupción perdería datos
  - La E/S programada todavía admitiría más velocidad puesto que no tendría que ejecutar las instrucciones de guarda y restauración del contexto ni la instrucción *RTI*.
  - **Para dispositivos rápidos se hace necesario un procedimiento de E/S con menor intervención de la CPU**

1. Entrada/Salida por acceso directo a memoria (DMA)

Acceso Directo a Memoria (DMA: Direct Memory Access)

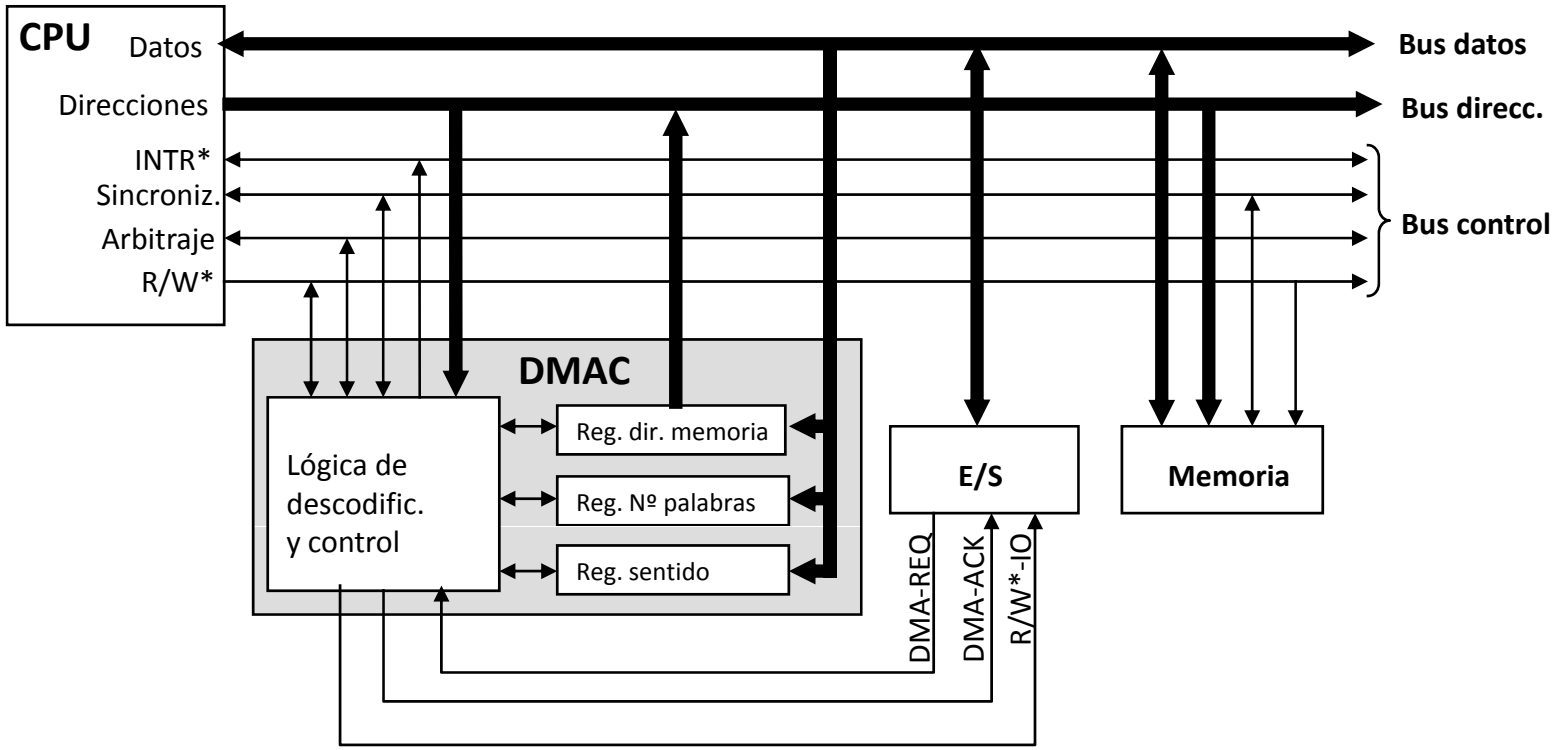
- Se trata de un **módulo con capacidad para leer/escribir directamente en la memoria** los datos procedentes/enviados de/a los dispositivos periféricos.
- **Un DMA es inicializado por la CPU cuando tiene que realizar una operación de E/S.**
- Una vez inicializado opera de la siguiente manera:
  - Solicita a la *CPU* la petición para acceder a memoria.
  - La *CPU* pone en estado de alta impedancia su conexión a los buses del sistema.
  - El DMA accede a la memoria
  - Cuando finaliza la operación el *DMA* la *CPU* vuelve a tomar control
  - La velocidad de transferencia sólo está limitada por el ancho de banda de la memoria.



1. Entrada/Salida por acceso directo a memoria (DMA)

Estructura de un controlador de DMA

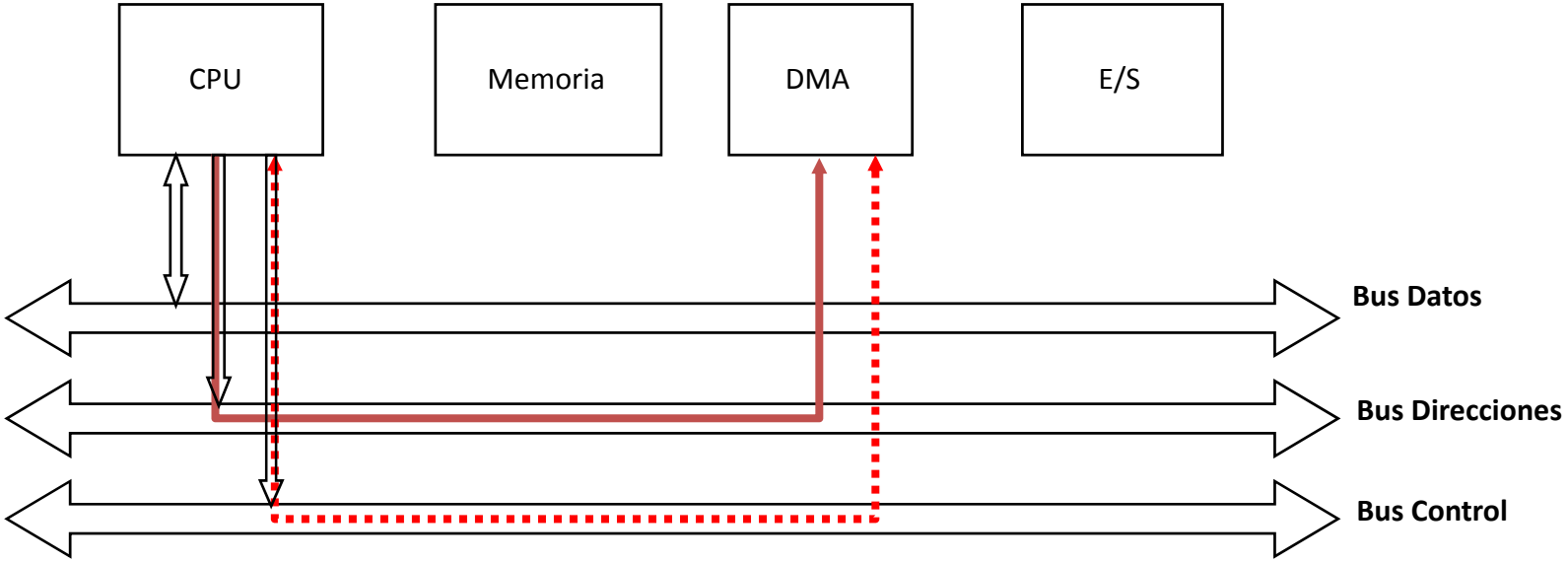
- **Reg. dir. memoria:** almacena la dir. inicial de memoria y se incrementa/decrementa después de transferir cada palabra
- **Reg. Nº palabras:** almacena el número de palabras a transferir y se decrementa después de transferir cada palabra
- **Reg. sentido:** almacena el sentido de la transferencia (lectura o escritura)



1. Entrada/Salida por acceso directo a memoria (DMA)

Inicialización de una transferencia DMA (1)

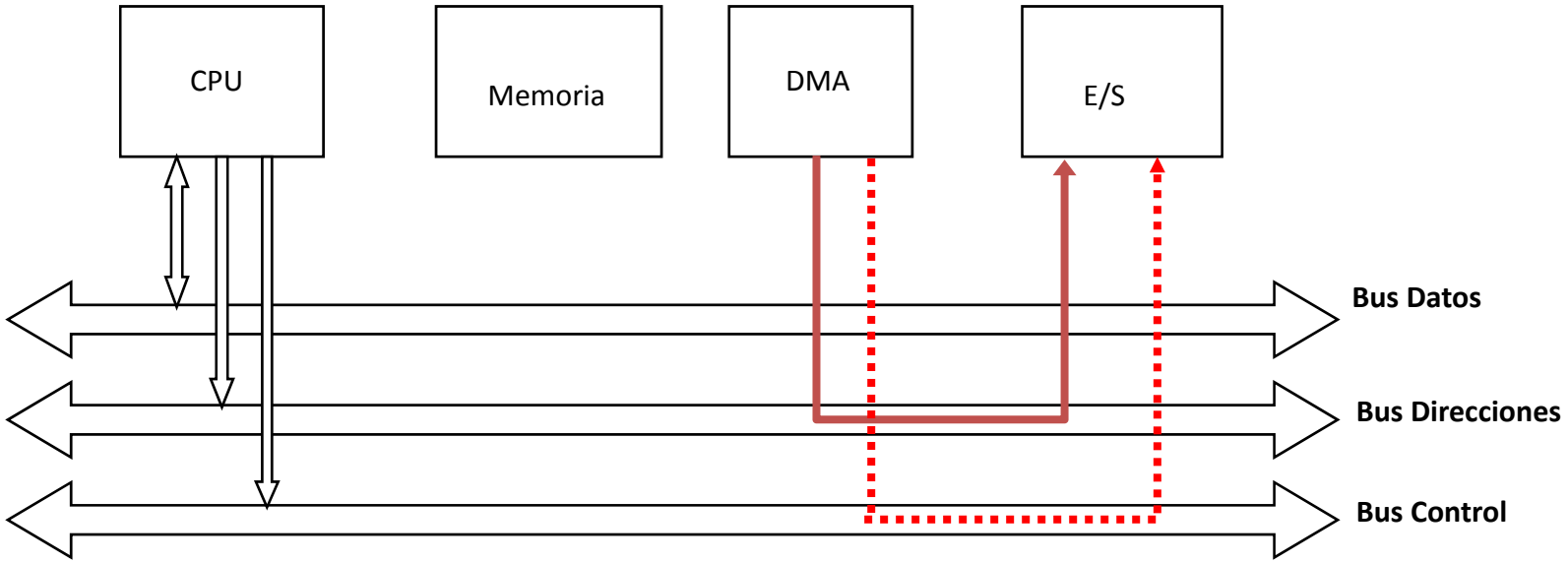
- ✓ **Paso1 : La CPU programa al controlador de DMA** para que ejecute la operación de E/S completa (todas las transferencias)
  - **Dirección inicial de memoria** → Reg. dir. memoria
  - **Número de palabras a transferir** → Reg. Nº palabras
  - **Sentido de la transferencia** → Reg. Sentido



1. Entrada/Salida por acceso directo a memoria (DMA)

Inicialización de una transferencia DMA (2)

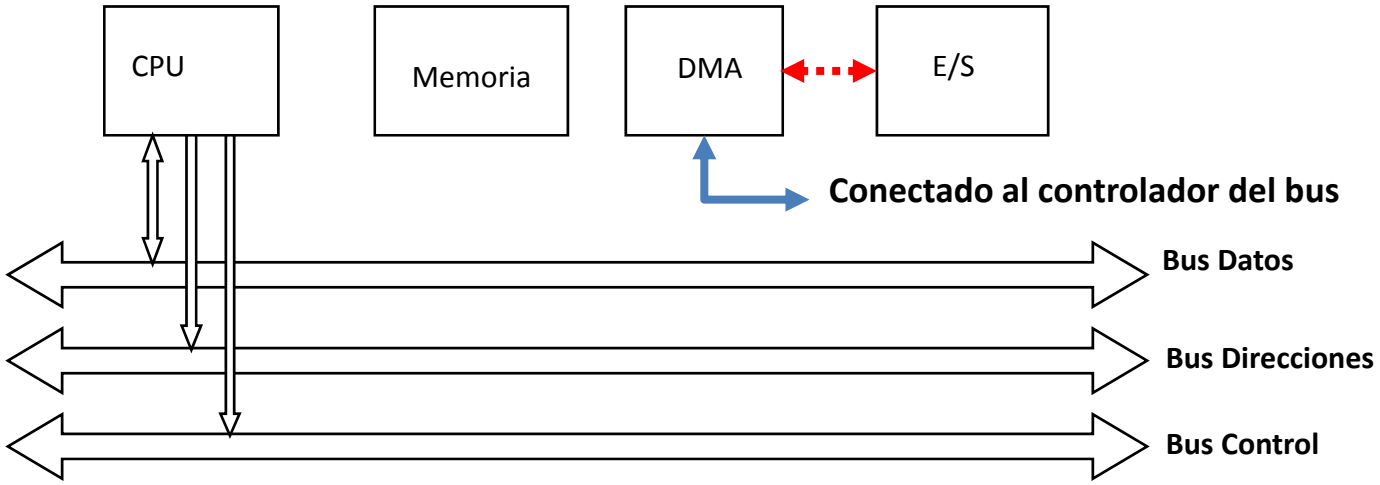
- ✓ **Paso2:** El controlador de DMA programa al controlador de dispositivo para realizar una transferencia
  - **Como en E/S programada**



1. Entrada/Salida por acceso directo a memoria (DMA)

Realización de la transferencia (1)

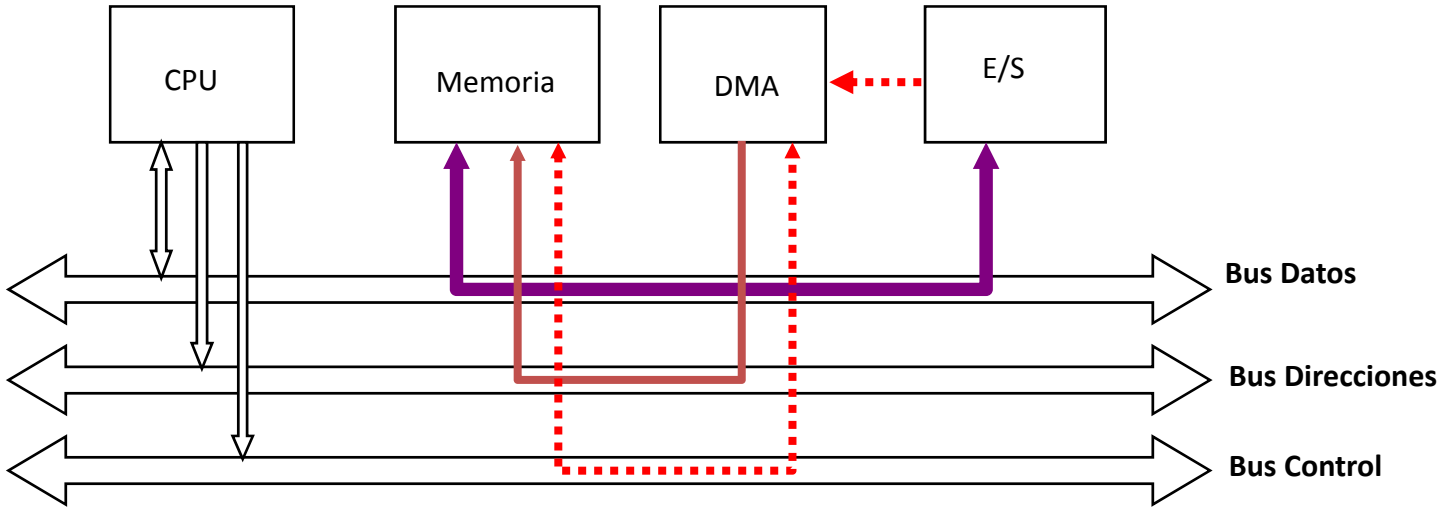
- **Paso 3:**
  - **El controlador del dispositivo avisa al controlador de DMA** de que está listo para realizar la transferencia
  - El **DMA solicita el control del bus** mediante las líneas de arbitraje
  - El **DMA recibe la concesión del bus** e indica al periférico que **puede iniciar la transferencia**



1. Entrada/Salida por acceso directo a memoria (DMA)

Realización de la transferencia (2)

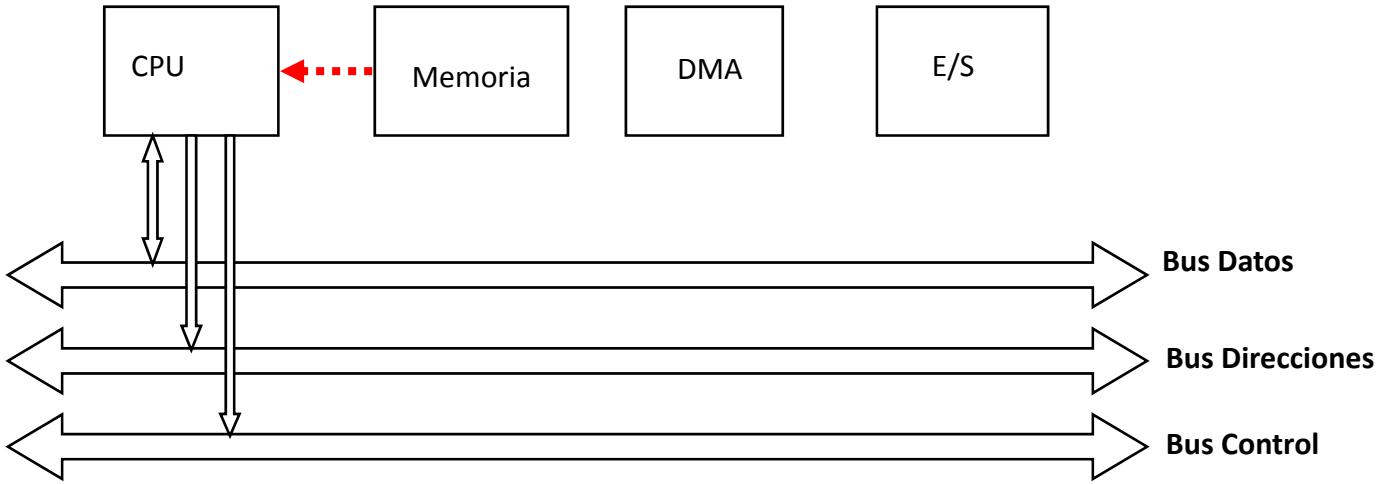
- ✓ **Paso 4:**
  - **El DMA debe generar y procesar las señales del bus adecuadas**
    - Dirección de memoria sobre la que se realiza la transferencia
    - Señales de sincronización de la transferencia
    - Señales de lectura/escritura
  - **Después de la transferencia de cada palabra se actualizan los registros del DMA**
    - Decrementar el registro de nº de palabras
    - Incrementar/decrementar el reg. de direcciones de mem. (según sean direcciones crecientes o decrecientes
  - **Si el tipo de transferencia permite enviar más de una palabra** antes de devolver el control al bus **se repite el paso 4** hasta que se hayan realizado la transferencia de todo el bloque
    - **El controlador del dispositivo avisa al controlador de DMA** de que está listo para realizar la transferencia de la siguiente palabra



1. Entrada/Salida por acceso directo a memoria (DMA)

Finalización de la transferencia

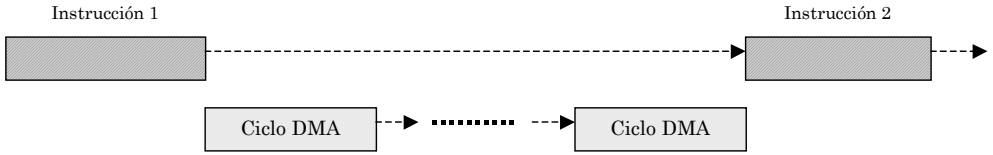
- **Paso 5**: Cuando el registro nº de palabras llega a cero.
  - El **DMAC libera el bus** y **devuelve el control a la CPU**
    - El DMAC suele activar una señal de interrupción para indicar a la CPU la finalización de la operación de E/S solicitada



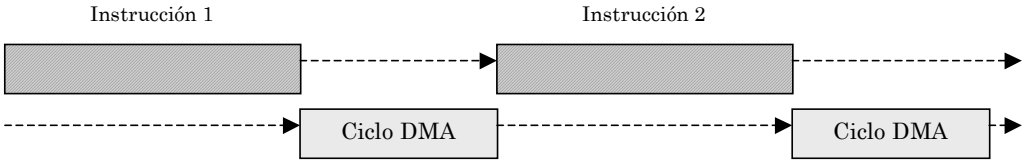
1. Entrada/Salida por acceso directo a memoria (DMA)

Control del bus en operaciones de DMA

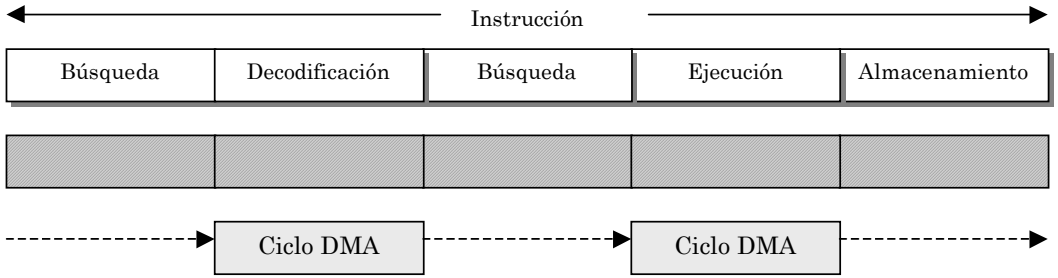
- **Ráfagas**
  - El *DMA* toma control del bus durante la transmisión de un bloque de datos completo.
  - Consigue alta velocidad de transferencia, dejando inactiva la *CPU* durante la operación.



- **Robo de ciclo**
  - El DMA toma control del bus y lo retiene durante un solo ciclo para transmitir una palabra
  - Es el modo más usual de transferencia. Se dice que el DMA roba ciclos a la CPU.



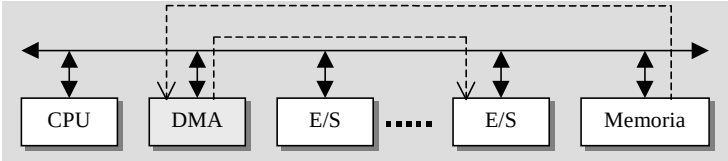
- **Modo transparente**
  - El *DMA* accede al bus sólo en los ciclos en los que la *CPU* no lo utiliza.
  - Esto ocurre en diferentes fases de ejecución de las instrucciones, p.e. decodificación
  - El programa no se ve afectado en su velocidad de ejecución



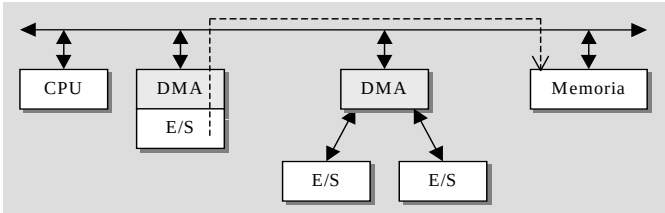
1. Entrada/Salida por acceso directo a memoria (DMA)

Configuraciones de DMA

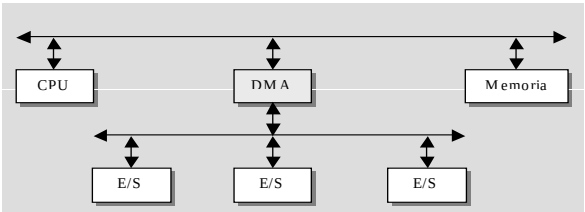
- ***DMA* independiente**
  - Todos los módulos comparten el bus del sistema.
  - El *DMA* hace de intermediario entre la memoria y el periférico.
  - Esta configuración es poco eficaz, ya que cada transferencia consume 2 ciclos del bus.



- **Integración *DMA-E/S***
  - Proporciona un camino entre el *DMA* y uno o más controladores de E/S
  - Reduce a 1 el número de ciclos de utilización del bus.



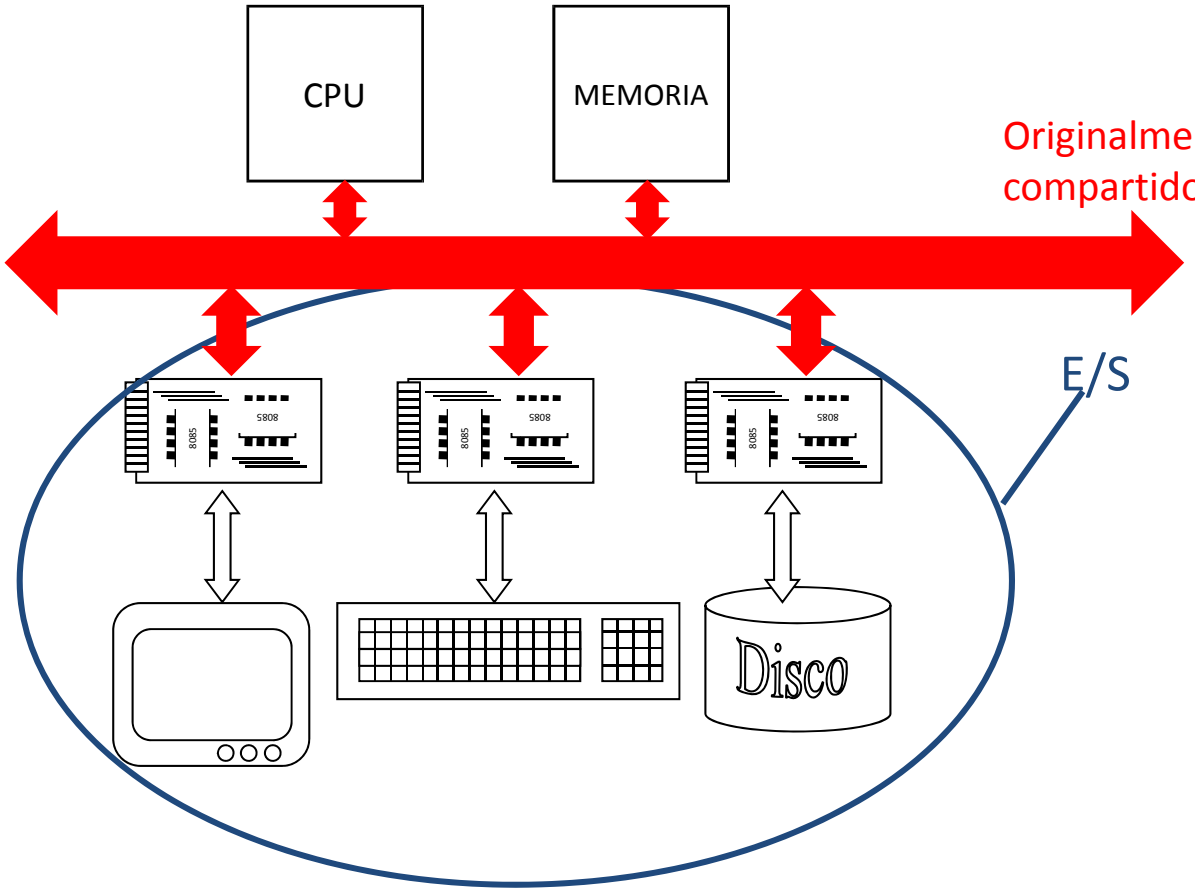
- **Bus de *E/S* conectado al *DMA***
  - Generaliza el caso anterior con un bus de *E/S* para todos los controladores de E/S.
  - Proporciona una configuración fácilmente ampliable.



2. Buses de interconexión E/S

Necesidad de los buses

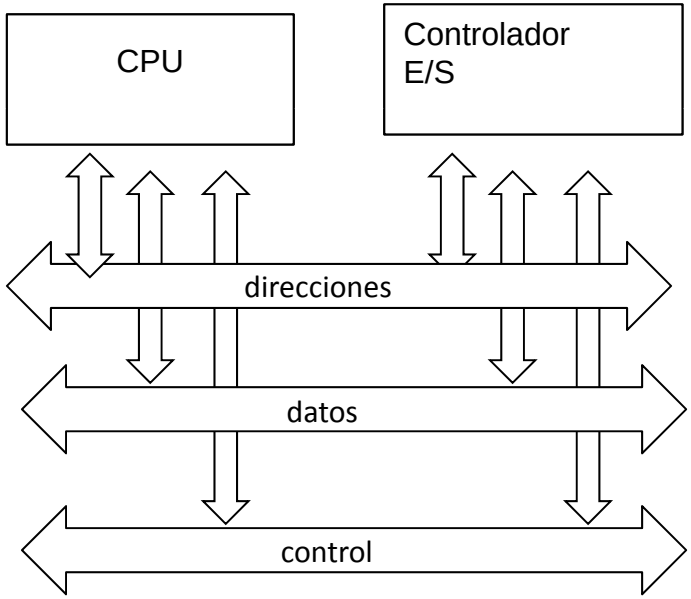
- Para que la CPU pueda intercambiar información con el sistema de memoria y los dispositivos de E/S es necesario un medio de comunicación denominado BUS



2. Buses de interconexión E/S

Organización general de un bus

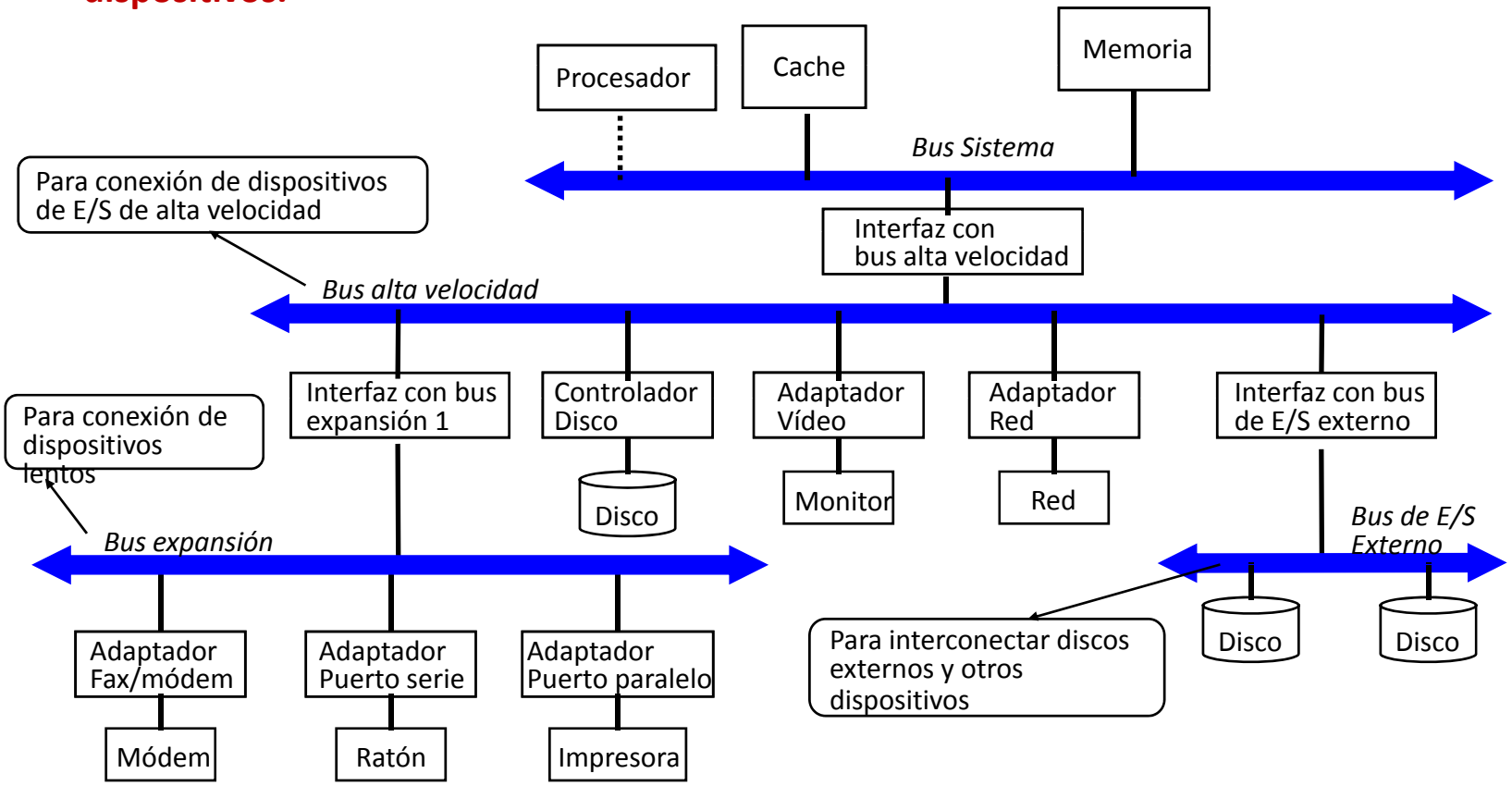
- **Líneas de control**
  - Utilizadas para implementar el protocolo de comunicación entre componentes
- **Líneas de datos**
  - Transmite la información entre emisor y receptor
  - Anchura de bus: número de bits que se puede transmitir
- **Líneas de dirección**
  - Contienen la información de direccionamiento
  - Habitualmente, coinciden físicamente con las de dato



2. Buses de interconexión E/S

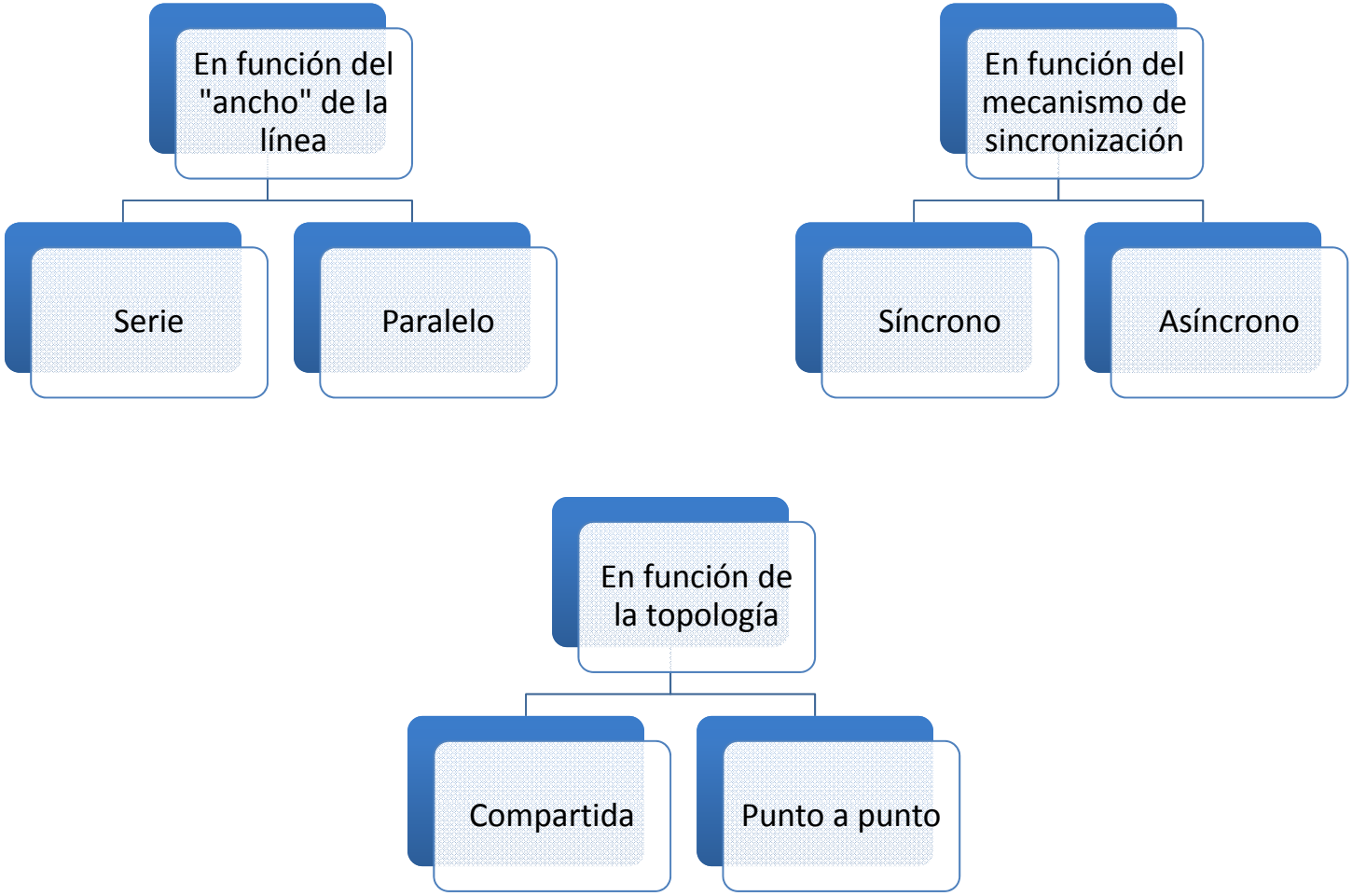
Jerarquía de los buses

- El uso de un único bus resultó inadecuado cuando aumentaron el número de dispositivos a conectar con velocidades de operación muy diferentes.
- Se estableció una **Jerarquía de buses para compatibilizar las velocidades de los diferentes dispositivos.**



2. Buses de interconexión E/S

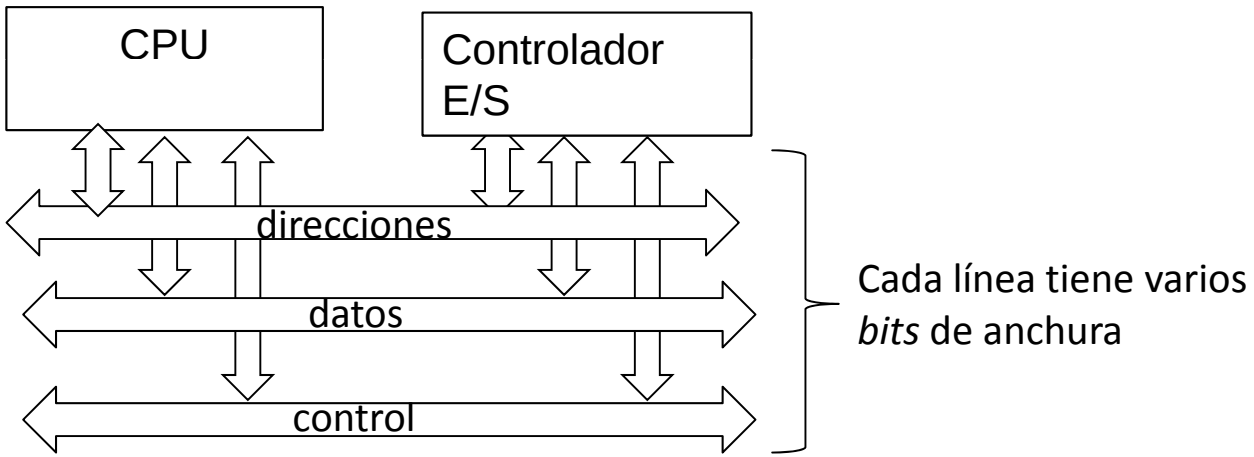
Clasificación de los buses



2. Buses de interconexión E/S

Transmisión paralela

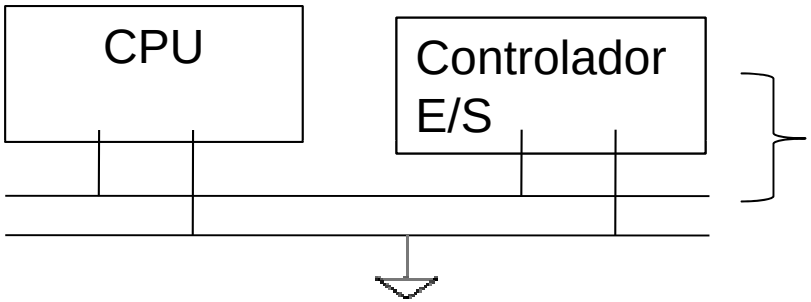
- Utiliza varias líneas de comunicación (cables) a través de las cuales **se envían varios bits de información de forma simultánea**
  - ✓ Ancho de banda elevado (en teoría…)
  - ✗ Para conectar dispositivos a distancias medias o largas resulta muy costosa
  - ✗ Frecuencia de funcionamiento limitada por factores físicos
  - ✗ Los dispositivos de baja velocidad no aprovechan el potencial de la transmisión paralela (ratón, módem…



2. Buses de interconexión E/S

Transmisión serie

- Utiliza una única línea de comunicación (2 cables: dato y tierra) a través de las cual se **se envían los bits de información de forma secuencial**
  - ✓ Es menos costosa que la E/S paralela
  - ✓ Permite frecuencias de funcionamiento mayores
  - ✗ A igualdad de frecuencia que el caso paralelo, el ancho de banda es menor



Toda la información (datos, control…) se emite secuencialmente

Es posible **aumentar el ancho de banda** entre dos dispositivos **usando varias conexiones serie**

2. Buses de interconexión E/S

Comunicación síncrona vs asíncrona

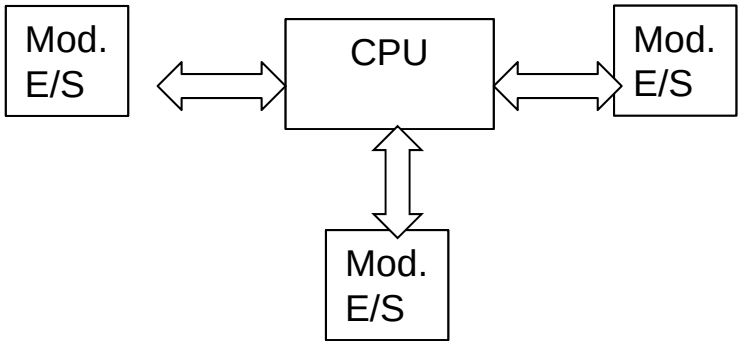
- ¿Cuándo empieza/acaba el envío de un dato?
  - **Asíncrona**
    - La señal de reloj NO se envía por la línea de comunicación
      - Emisor y receptor utilizan sus propias señales de reloj.
      - Es necesario establecer un **protocolo** para la sincronización entre ambos
    - ✓ No es imprescindible que emisor y receptor funcionen a la misma frecuencia
  - **Síncrona**
    - La señal de reloj viaja con el resto de señales
    - ✓ Más sencillo y, en general, más eficiente
    - ✗ Exige que emisor y receptor funcionen a la misma frecuencia
    - ✗ Debe ser corto si queremos que sea rápido (para que no le afecte el clock skew)

2. Buses de interconexión E/S

Punto a punto vs. Compartida

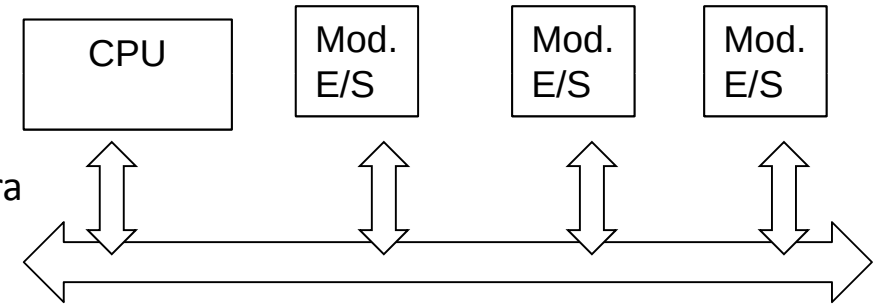
Comunicación Punto a punto

- ✓ Gran rendimiento
- ✗ Exige un gran número de interfaces



Compartida

- ✓ Más versátil y barato
- ✗ Cuello de botella en la comunicación
- ✗ Exige sincronización entre dispositivos para evitar escrituras simultáneas



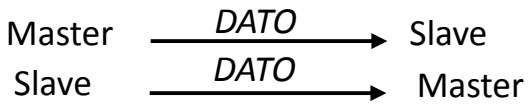
2. Buses de interconexión E/S

Transferencia de datos en un bus

- **Tipos básicos de transferencia**
  - Escritura
  - Lectura

- **Fases de una transferencia**
  - 1. Direccionamiento (identificación) del *slave*
  - 2. Especificación del tipo de operación (lectura o escritura)
  - 3. Transferencia del dato
  - 4. Finalización del ciclo de bus

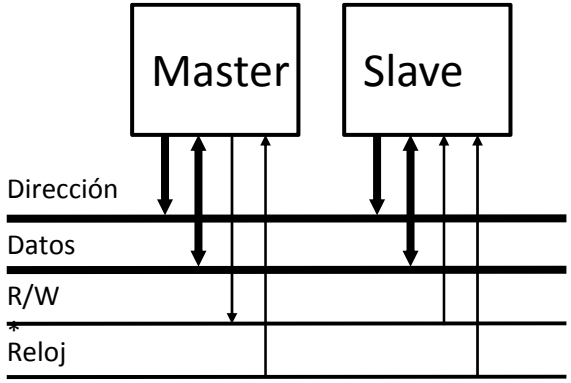
- **Control de la transferencia**
  - Sincronización: determinar el inicio y el final de cada transferencia
  - Arbitraje: controlar del acceso al bus en caso de varios masters

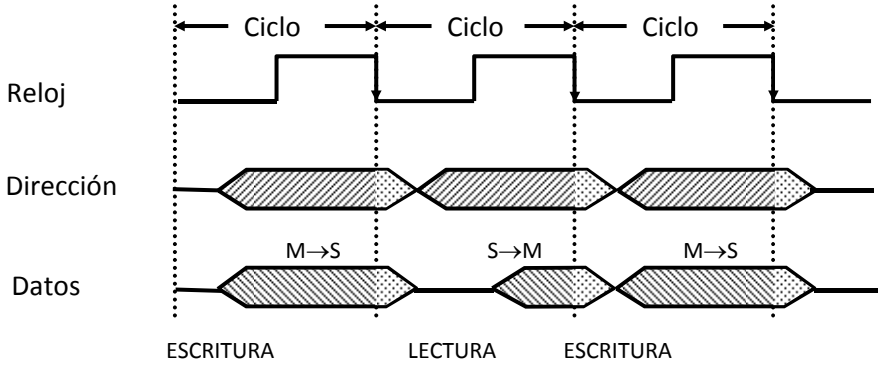


2. Buses de interconexión E/S

Sincronización: protocolo síncrono

- **Existe un reloj que gobierna todas las actividades del bus**, las cuales tienen lugar en un número entero de ciclos de reloj (1 en el ejemplo)
- Los **flancos del reloj** (de bajada en el ejemplo) determinan el comienzo de un nuevo ciclo de bus y el final del ciclo anterior





tiempo que debe ser superior a:
t. decodificación + t. setup + t. skew

tiempo que debe ser superior a:
t. setup + t. skew

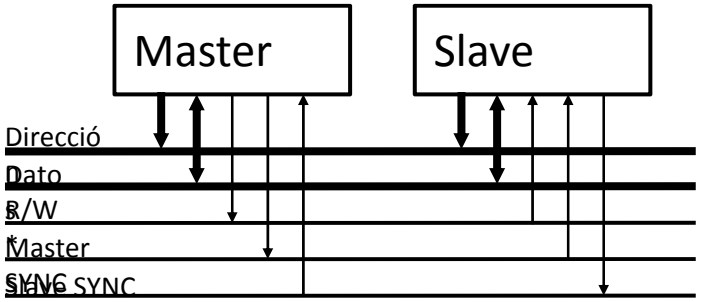
tiempo que debe ser superior a:
t. hold

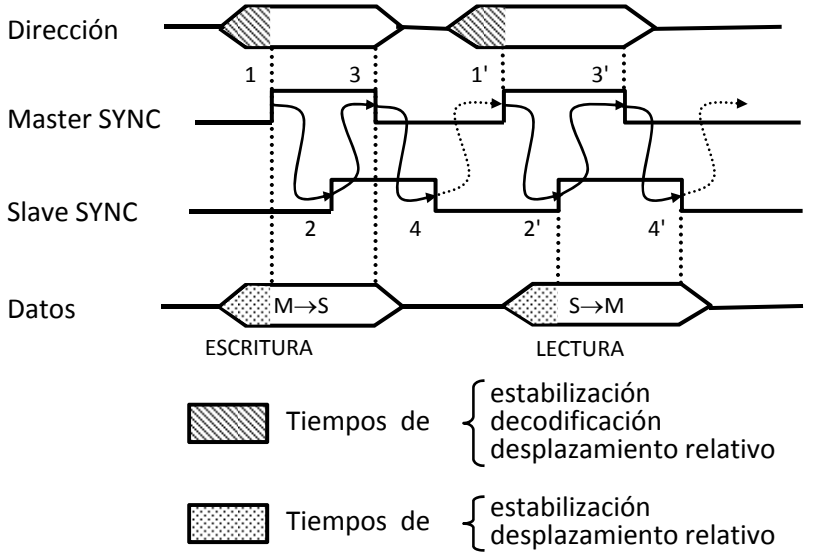
- ***Ventajas*:**
  - Simplicidad (de diseño y de uso)
  - Sólo se necesita una señal (reloj) para llevar a cabo la sincronización
  - Mayor velocidad (en relación a protocolo asíncrono)
- ***Desventajas*:**
  - El periodo de reloj se tiene que adaptar a la velocidad del dispositivo más lento (por lo que suele usarse para conectar dispositivos homogéneos)
  - No existe confirmación de la recepción de los datos
  - La necesidad de distribuir la señal de reloj limita la longitud del bus

2. Buses de interconexión E/S

Sincronización: protocolo asíncrono

- **No existe señal de reloj**
- Los dispositivos implicados en la transmisión fijan el comienzo y el final de la misma mediante el intercambio de señales de control (**handshake**)
- Se utilizan 2 señales de sincronización:
  - Master SYNC (procedente del master)
  - Slave SYNC (procedente del slave)
- Protocolo completamente interbloqueado:
  - A cada flanco de master le sigue uno del slave
- **Ciclo de escritura**:
  1: (M a S) Hay un dato en el bus
  2: (S a M) He tomado el dato
  3: (M a S) Veo que lo has tomado
  4: (S a M) Veo que lo has visto (Bus libre)
- ***Ciclo de lectura***:
  1': (M a S) Quiero un dato
  2': (S a M) El dato está en el bus
  3': (M a S) He tomado el dato
  4': (S a M) Veo que lo has tomado (Bus libre)
- ***Ventajas:***
  Facilidad para conectar elementos de diferentes velocidades
  Fiabilidad: la recepción siempre se confirma.
- ***Desventajas:***
  El intercambio de señales de control introduce retardos adicionales
  A igualdad de velocidades de los dispositivos, menos eficiente que el síncrono

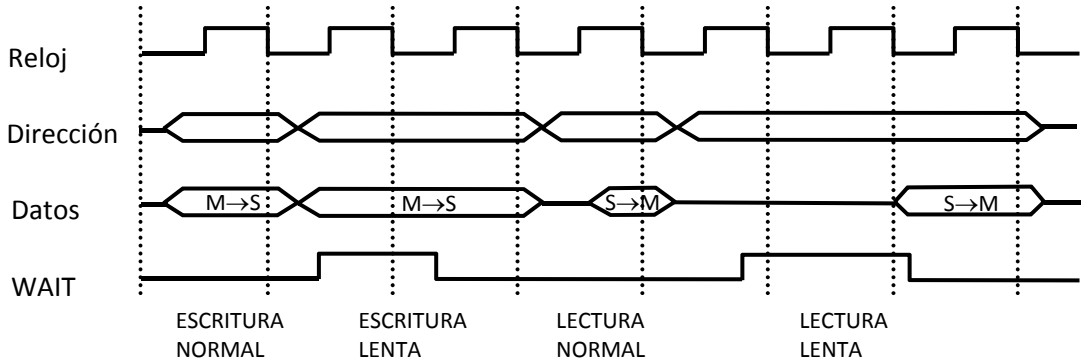


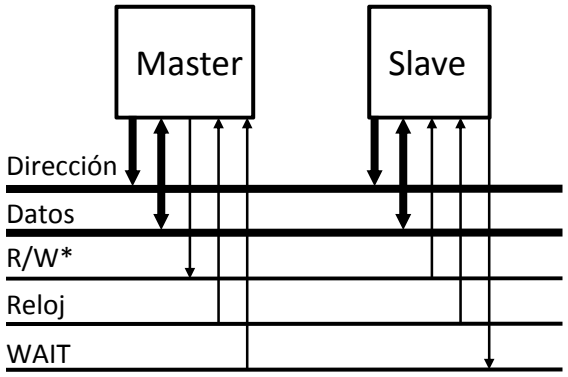


2. Buses de interconexión E/S

Sincronización: protocolo semisíncrono

- Las transferencias se rigen por una **única señal de reloj**
- Cada transferencia puede ocupar uno o varios periodos de reloj
- Señal de WAIT (puede activarla cualquier Slave si no es capaz de realizar la transf. en 1 solo ciclo)
  - **Dispositivos rápidos**: operan como en un bus síncrono (una transferencia por ciclo)
  - **Dispositivos lentos**: activan la señal de WAIT y congelan actuación del Master (una transferencia puede ocupar varios ciclos)





2. Buses de interconexión E/S

Conflictos en el acceso: arbitraje

- Si puede haber varios emisores (Masters), es necesario que el acceso al bus sea *libre de conflictos*
  - Se debe evitar que dos módulos escriban simultáneamente en el bus
  - Cada dispositivo solicita permiso para poder tomar el control del bus

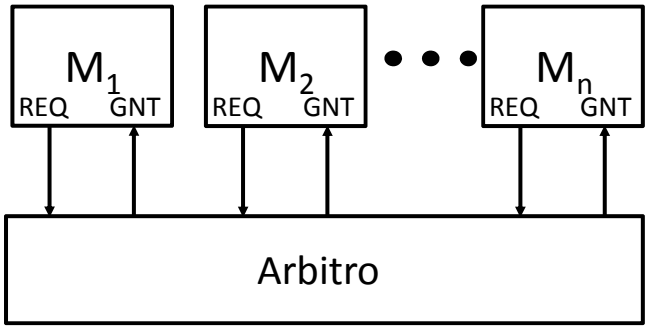
- **Protocolos centralizados**: existe un **árbitro** del bus o **master principal** que controla el acceso al bus
  - En estrella
  - Daisy-chain de 3 y 4 hilos

- **Protocolos distribuidos**: el control de acceso al bus se lleva a cabo entre todos los posibles masters de una forma cooperante
  - Protocolo de líneas de identificación
  - Protocolo de códigos de identificación

2. Buses de interconexión E/S

Protocolos centralizados: en estrella

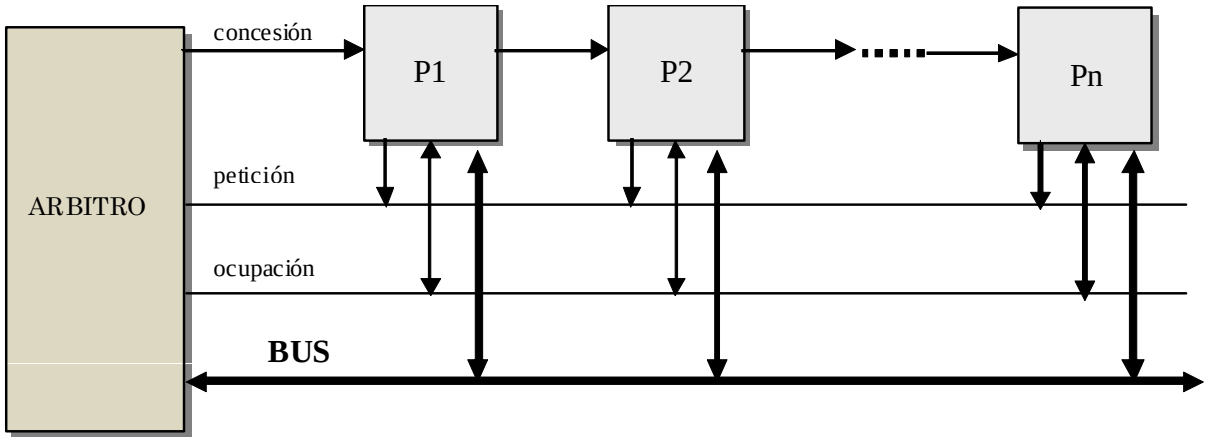
- Cada master se conecta al árbitro mediante dos líneas individuales:
  - **BUS REQUEST** (REQ): línea de petición del bus
  - **BUS GRANT** (GNT): línea de concesión del bus
- Varias peticiones de bus pendientes: el árbitro puede aplicar distintos algoritmos de decisión
  - FIFO, Prioridad fija, Prioridad variable
- **Ventajas:**
  - Algoritmos de arbitraje simples
  - Pocos retardos de propagación de las señales (en comparación con daisy-chain)
- **Desventajas:**
  - Número elevado de líneas de arbitraje en el bus (dos por cada posible master)
  - Número de masters alternativos limitado por el número de líneas de arbitraje
- Ejemplo: PCI



2. Buses de interconexión E/S

Protocolo de encadenamiento (daisy chaining) de 3 señales

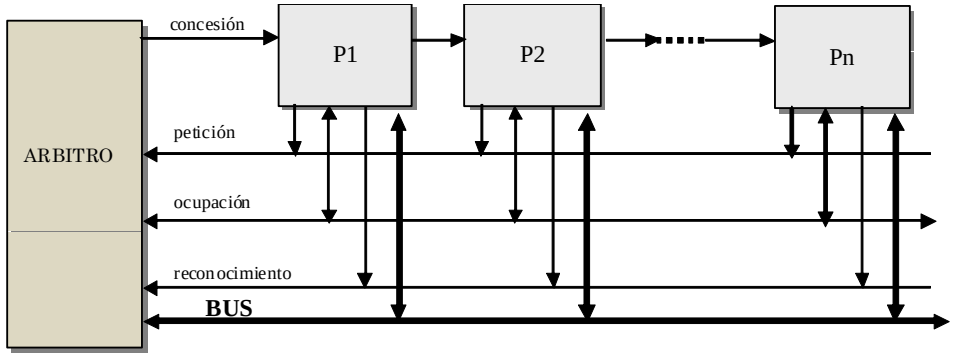
- **Utiliza tres líneas para gestionar el arbitraje**: **petición, ocupación y concesión**.
- **La línea de petición es compartida** por todos los procesadores a través de una entrada al árbitro con capacidad de O-cableada.
- El árbitro activa la concesión cuando recibe una petición y la de ocupación está desactivada.
- Cuando un procesador toma el control del bus activa la línea de ocupación.
- Si un procesador recibe la concesión y no ha solicitado el bus, transmite la señal al siguiente.
- Un procesador toma el control del bus si tiene una petición local pendiente, la línea de ocupación está desactivada y recibe el flanco de subida de la señal de concesión.

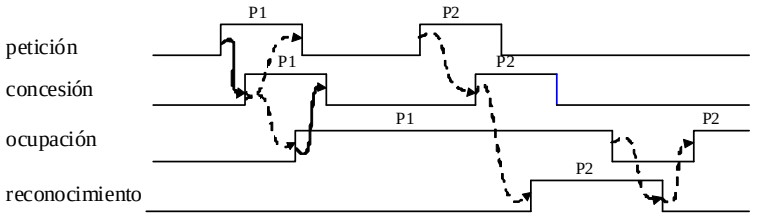


2. Buses de interconexión E/S

Protocolo de encadenamiento (daisy chaining) de 4 señales

- **Permite simultanear el uso del bus por un procesador con el arbitraje para seleccionar el procesador que utilizará el bus en la siguiente transacción.**
- Cuando el primer procesador abandona el bus, el arbitraje para el siguiente ya se ha realizado.
- La línea nueva de reconocimiento la activa un procesador que solicitó el bus (activó petición) y recibió la concesión pero la línea de ocupación estaba activa (bus ocupado).
- Cuando el árbitro recibe la activación de reconocimiento inhibe su actuación, es decir, deja de atender la señal de petición y generar la de concesión.
- El procesador queda en espera de ocupar el bus cuando lo abandone su actual usuario desactivando la señal ocupación.
- Cuando esto ocurre, el procesador ocupa el bus y desactiva la señal de reconocimiento.





Diálogo de señales:
- Ocupación del bus por P1,
- Arbitraje a favor de P2 mientras P1 realiza su transacción.
- P2 ocupa el bus cuando P1 finaliza

2. Buses de interconexión E/S

Puentes de comunicación

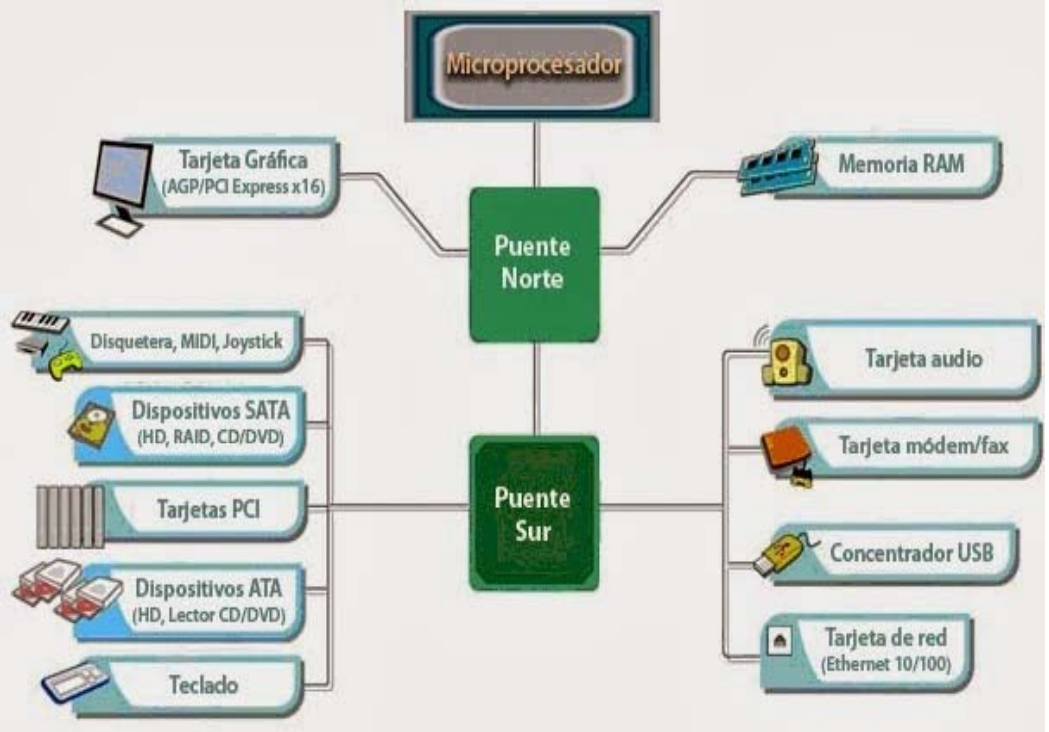
Los actuales computadores suelen articular el sistema de intercomunicación entre sus componentes en torno a dos dispositivos (CIs) denominados Puente Norte (Northbridge) o Centro Controlador de Memoria (Memory Controller Hub) y Puente Sur (Southbridge) o Centro Controlador de E/S (E/S Controller Hub)

- **El Puente norte** controla la conexión de la CPU con los componentes de alta velocidad del computador:
  - Memoria RAM
  - Buses de alta velocidad
  - Puente sur

- **El Puente sur (Southbridge)** se encarga de coordinar los diferentes dispositivos de entrada/salida y algunas otras funcionalidades de baja velocidad. **Se comunica indirectamente con la CPU a través del puente norte.**

Un puentes sur actual puede incluir soporte para:

- Bus PCI
- Bus ISA
- Controlador DMA
- Controladores de interrupción
- Ethernet
- RAID
- USB
- Codec de Audio

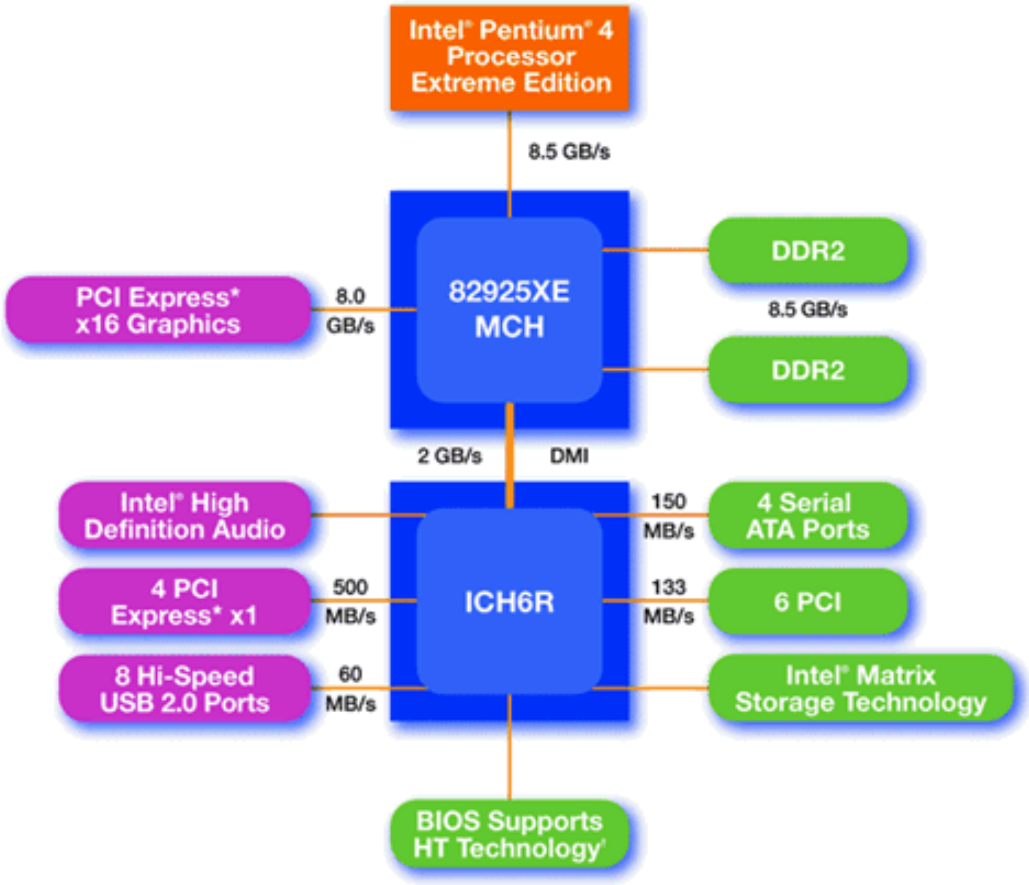


2. Buses de interconexión E/S

Ejemplo: Chipsets de Intel. 2004

- **Controlador de memoria** (puente norte, Memory Controller Hub):
  - Controlador DMA que conecta el procesador con memoria, el bus gráfico y el chip del puente sur

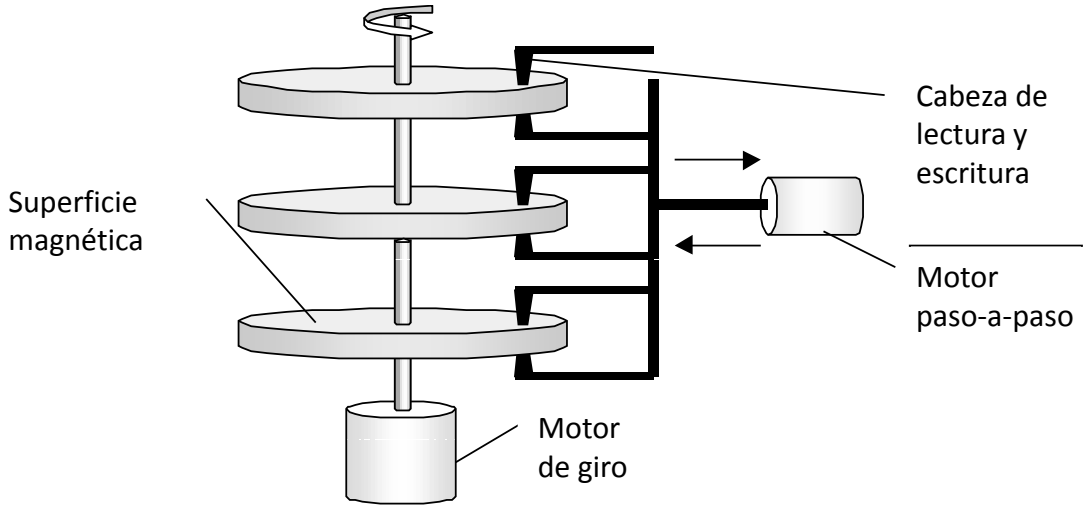
- **Controlador de E/S** (puente sur, I/O Controller Hub):
  - Conecta el puente norte a los buses de E/S



4. Dispositivos de almacenamiento

Discos magnéticos

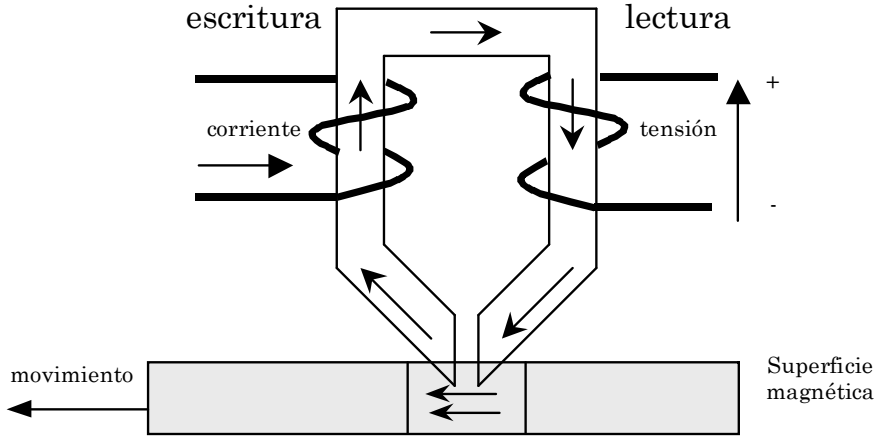
- Son memorias secundarias que se conectan al computador como dispositivos periféricos
- Forman parte de la jerarquía de memoria del computador.
- Están constituidos por superficies circulares recubiertas por un material ferromagnético.
- La información se asocia al sentido de la magnetización de pequeñas áreas de su superficie.
- El conjunto de superficies gira a una velocidad constante por la acción de un motor.
- La información se escribe y lee a través de un conjunto de cabezas que se mueve radialmente.



4. Dispositivos de almacenamiento

Cabeza de lectura y escritura

- Los procesos de lectura y escritura se realizan a través de una cabeza de grabación
- La cabeza es un núcleo de hierro y dos bobinas, una para escritura y otra para lectura.
- La cabeza opera sobre el colchón de aire formado por su propio movimiento.



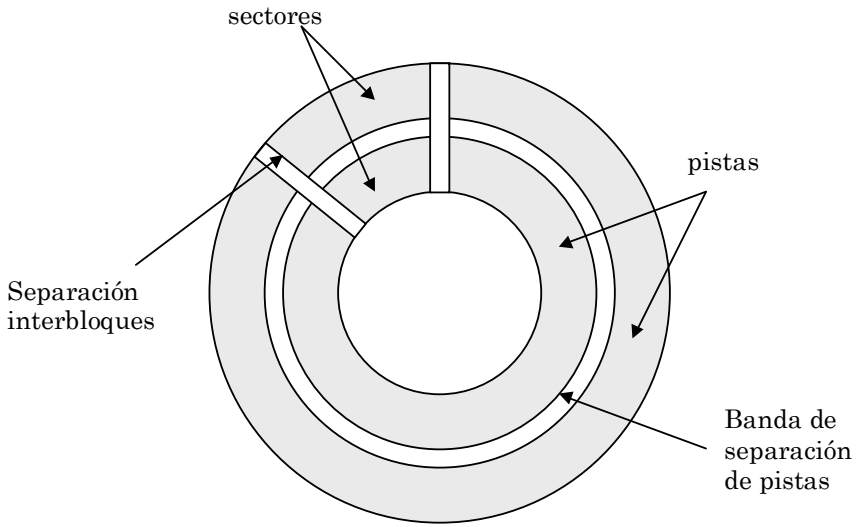
Escritura: Se hace pasar por la bobina de escritura una corriente que crea un campo magnético en el núcleo de hierro que se cierra a través de la pequeña región de la superficie magnética que en ese momento está bajo la cabeza, dejando una magnetización remanente del mismo sentido que el campo de la bobina.

Lectura: Se mueve la superficie magnética por debajo de la cabeza. La variación de flujo producida por las áreas magnetizadas induce una tensión en la bobina de lectura, que una vez amplificada permite identificar los valores grabados en la superficie del disco.

4. Dispositivos de almacenamiento

Formato de grabación

- Los datos se organizan en un conjunto de anillos concéntricos denominados *pistas*.
- Las pistas adyacentes están separadas por bandas vacías.
- Para simplificar la electrónica, se suele almacenar el mismo número de bits en cada pista.
- Los datos se transfieren en bloques y se almacenan en regiones denominada sectores.
- Suele haber entre 10 y 100 sectores por pista, y estos pueden ser de longitud fija o variable.
- Los sectores adyacentes se separan con regiones vacías.
- Se graban algunos datos extra utilizados sólo por el controlador del disco (no accesibles al usuario).



4. Dispositivos de almacenamiento

Parámetros de rendimiento de un disco magnético (1)

➢ Tiempo de búsqueda (*Ts*)

- Tiempo que tarda la cabeza en posicionarse en la pista.
- Se compone de dos partes:
  - Tiempo de comienzo *s* (necesario para que la cabeza adquiera una cierta velocidad radial)
  - Tiempo de pista (necesario para atravesar las pistas que preceden a la que se accede)
- Aproximación de Ts: ***Ts = m * n + s*** *n* = número de pistas atravesadas
  m = constante que depende del disco
- Valores típicos de m y s:
  - Discos de gama baja **m = *0,3 ms*** ***s = 20 ms***
  - Discos de gama alta ***m = 0,1 ms*** ***s* de *3 ms***

➢ Retardo rotacional (*Tr*)

- Tiempo que tarda el sector en alcanzar la cabeza.
- Valor medio de *Tr* : ***Tr = 1/2r*** (*r* = velocidad angular del disco en revoluciones/segundo)
- Valores típicos
  - Discos duros a *3.600 rpm* (una revolución tardará *16,7 ms*) ***8,3 ms*.**
  - Disquetes a *100 - 600 rpm* .....***100* a *200 ms.***

4. Dispositivos de almacenamiento

Parámetros de rendimiento de un disco magnético (2)

- **Tiempo de acceso (*Ta*)**
  - Suma del tiempo de búsqueda y el retardo rotacional:

$$Ta = Ts + Tr = m * n + s + 1/2r$$

- **Tiempo de transferencia (*Tt*)**
  - Tiempo de transferencia de datos una vez accedido el inicio de los mismos :

$$Tt = b/rN$$

b = número de bytes a transferir
N = número de bytes de una pista
r = velocidad de rotación en rps

- **Tiempo de operación (*To*)**
  - Suma de las componentes anteriores: $To = m * n + s + 1/2r + b/rN$

- En un computador para realizar una operación de E/S, a estos tiempos habrá que añadir:
  - *tiempo de espera por un canal*, si el disco no dispone del suyo propio (SO)
  - *tiempo de espera en la cola* hasta que el dispositivo esté disponible (SO)

5. Conjuntos redundantes de discos independientes

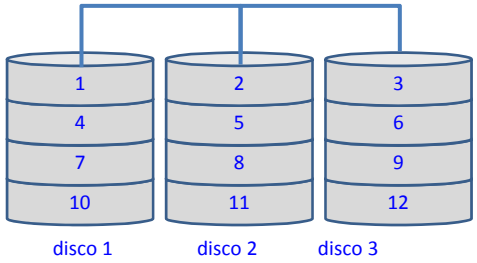
RAID (*Redundant Array of Independent/Inexpensive Disks*)

- **Es un método de combinación de dos o más discos para formar una única unidad lógica**
- Utilizan dos principios básicos: la **redundancia y el paralelismo**
- Aumentando el grado de redundancia se consigue mayor fiabilidad (tolerancia de fallos)
- Grabando un bloque de datos en varios discos se consigue acelerar su acceso (paralelismo)
- Existen diferentes opciones denominadas niveles de RAID
- Cada nivel proporciona un equilibrio distinto entre tolerancia a fallos, rendimiento y coste
- Cada nivel de RAID se ajusta mejor a determinadas aplicaciones y entornos
- Existen siete niveles de RAID (0 al 6) definidos y aprobados por el RAID *Advisory Board*

5. Conjuntos redundantes de discos independientes

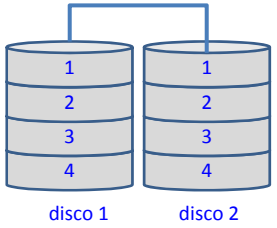
RAID Nivel 0

- Distribuye los datos a través de varios discos y **no proporciona redundancia**
- Maneja acceso simultáneo a varios discos, lo que **proporciona mayor velocidad de acceso**
- **No es tolerante a fallos**: si un disco falla el sistema falla
- Si un archivo ocupa las divisiones 1, 2 y 3 (ver dibujo) se puede recuperar en la tercera parte del tiempo que se tardaría si las tres divisiones estuvieran en el mismo disco.
- Se utiliza en tratamiento de imagen, sonido, vídeo y en general en aplicaciones que requieran gran velocidad de acceso y puedan tolerar fallos



RAID Nivel 1

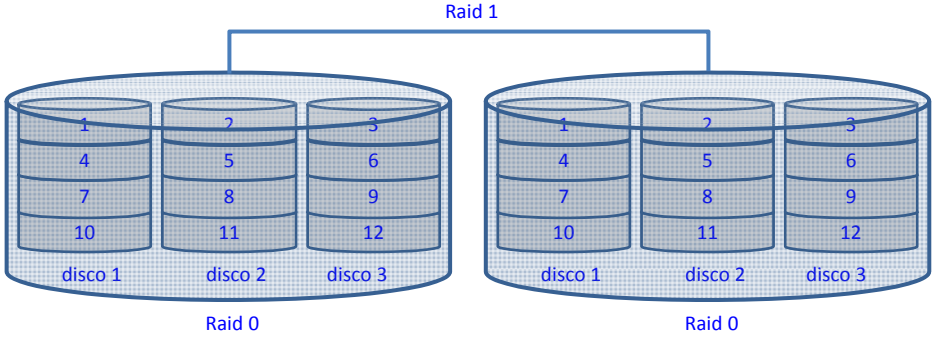
- Utiliza una copia ***espejo (mirroring)*** para **proporcionar redundancia tolerante a fallos**
- Los discos guardan exactamente la misma información por parejas
- Cuando un disco falla, su espejo puede recuperarlo, pero aumenta el coste del sistema



5. Conjuntos redundantes de discos independientes

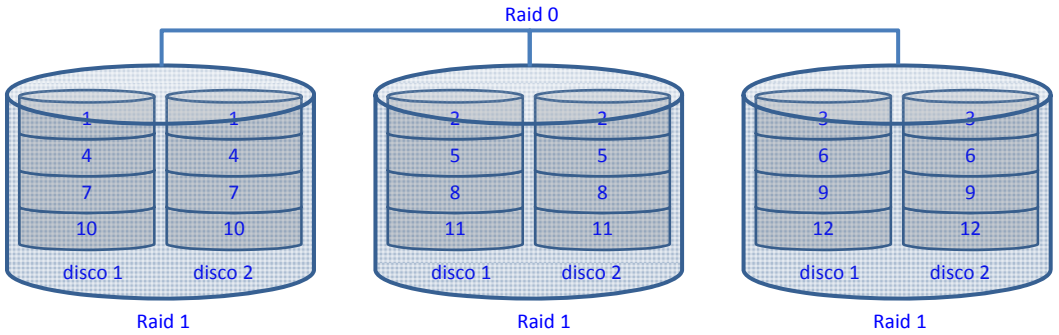
RAID 0 - 1

- Cada conjunto RAID 0 posee una copia espejo (RAID 1)
  - **Incorpora a la alta velocidad del RAID 0 la tolerancia de fallos de RAID 1**
  - **Coste elevado**



RAID 1 - 0

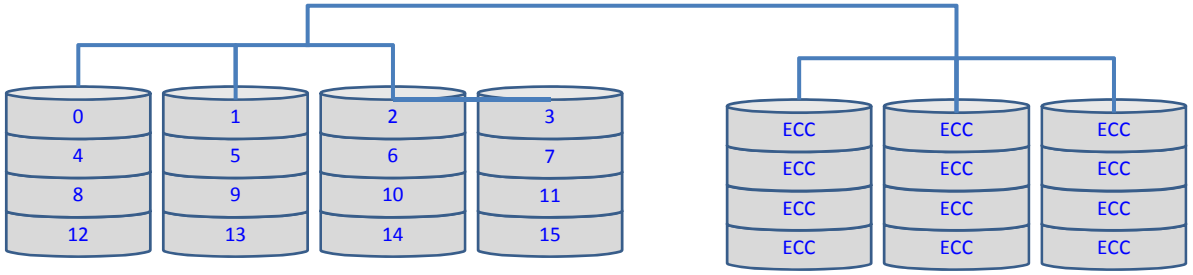
- Conjuntos RAID 1 (con su redundancia y tolerancia fallos) se organizan como un RAID 0



5. Conjuntos redundantes de discos independientes

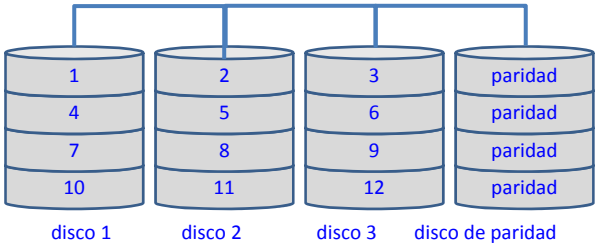
RAID Nivel 2

- **Utiliza códigos de corrección de Hamming** (los estudiaremos en el tema de memoria).
- Realiza acceso paralelo y sincronizado a todos los discos, datos y corrección de errores (ECC)



RAID Nivel 3

- Distribuye los datos por múltiples discos a nivel de bytes e **introduce el *chequeo de paridad***
- Añade redundancia utilizando un ***disco de paridad*** que permite recuperar errores producidos por un fallo en uno cualquier disco
- Si el fallo se produce en el disco de paridad, se pierde la redundancia, pero se mantiene intacta la información original.



La paridad se calcula con la función O-exclusiva: $X_p = X_1 \oplus X_2 \oplus X_3$

Si se produce un error en el disco 1 se puede recuperar su contenido con el contenido de los otros dos y el de paridad: $X_1 = X_p \oplus X_2 \oplus X_3$

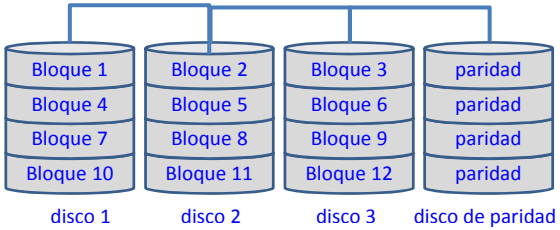
Análogamente para los discos 2 y 3: $X_2 = X_1 \oplus X_p \oplus X_3$

$X_3 = X_1 \oplus X_2 \oplus X_p$

5. Conjuntos redundantes de discos independientes

RAID Nivel 4

- Distribuye los datos por bloques entre varios discos, con la paridad en un disco.
- La paridad permite la recuperación de cualquier disco en caso de fallo.
- **El rendimiento es muy bueno para lecturas** (similar al nivel 0).
- **Penaliza las escrituras** porque requiere actualizar siempre los datos de paridad



RAID Nivel 5

- Crea datos de paridad, distribuyéndolos a través de todos los discos, excepto en aquel disco en que se almacena la información original. **Es la alternativa más popular.**
- Es el más completo de todos los niveles de redundancia, si un disco falla la información de paridad en los otros permite la reconstrucción del fallo.
- Escribe datos en los discos al nivel de bloques, siendo más apropiado para múltiples transacciones pequeñas como e-mail, procesadores de palabras, hojas electrónicas, etc.