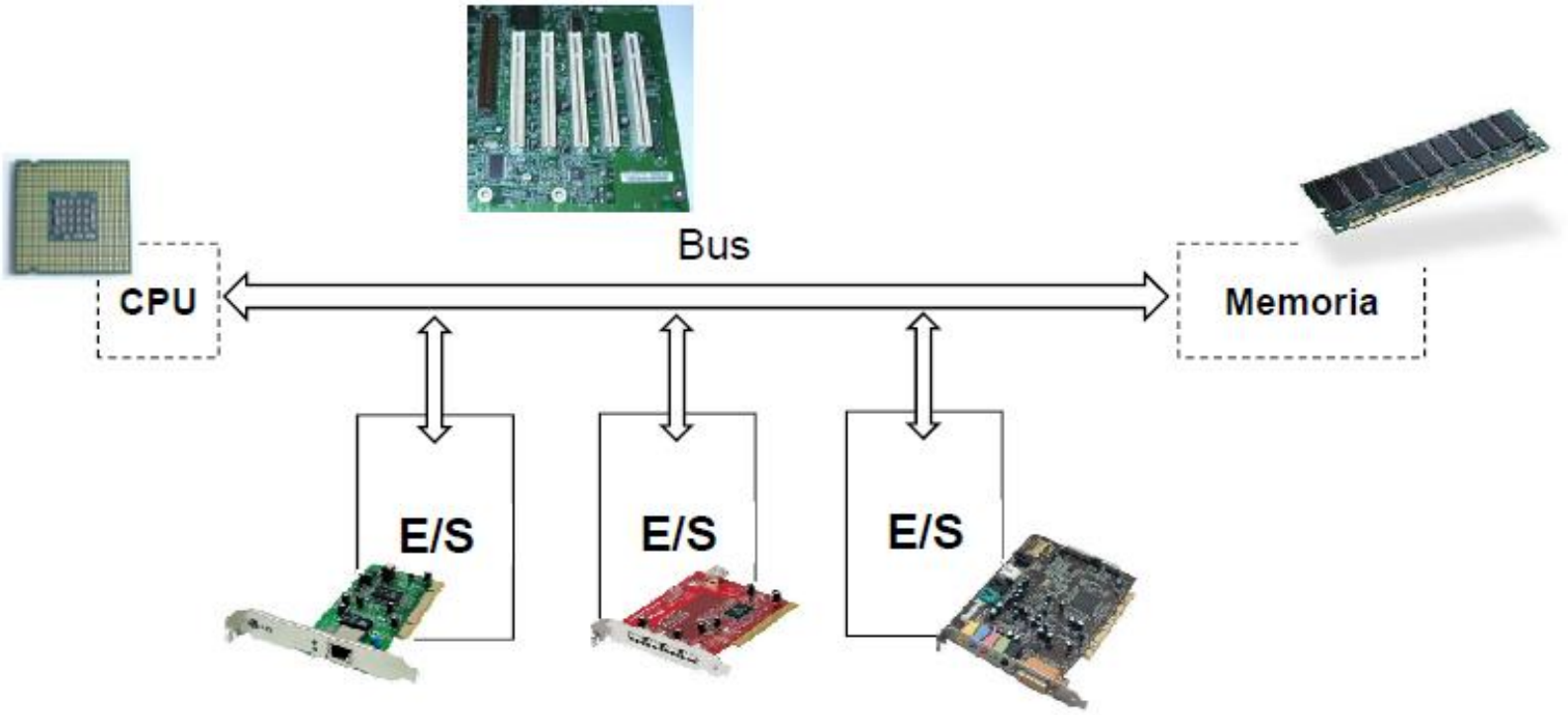
UCA
Universidad de Cádiz

6. Entrada salida, buses y periféricos.

Contenidos

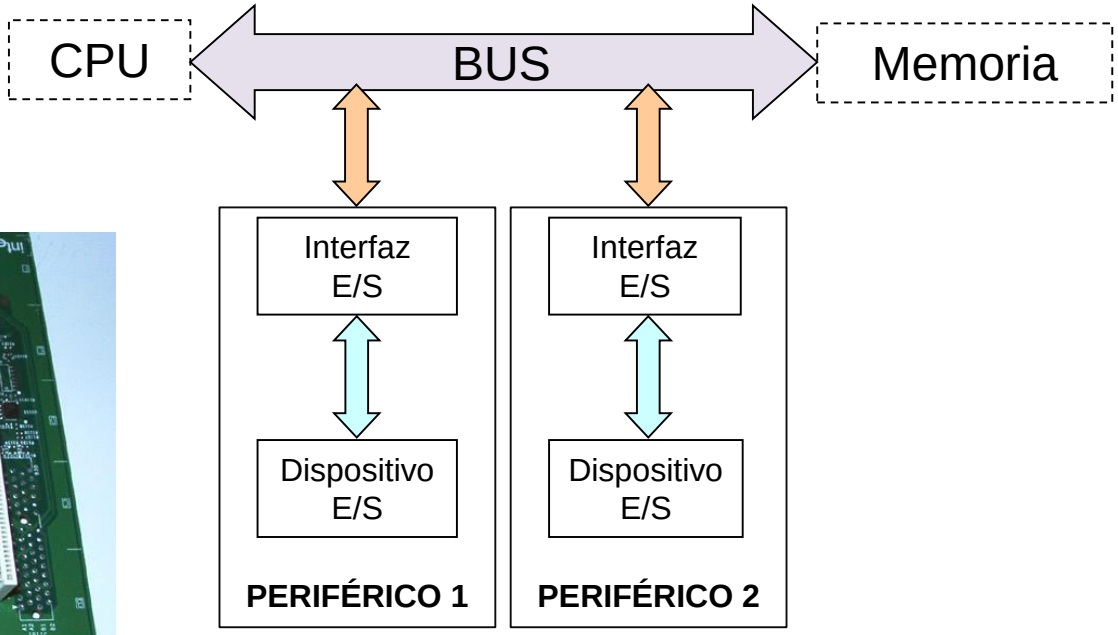
- Buses
  - Estructura y funcionamiento
  - Jerarquía de buses
- Periféricos
  - Concepto de periférico
  - Clasificación y tipos de periféricos
  - Estructura general de periféricos. Estudio de casos.
- Interfases de E/S
  - Estructura
  - Funcionamiento general
  - Características

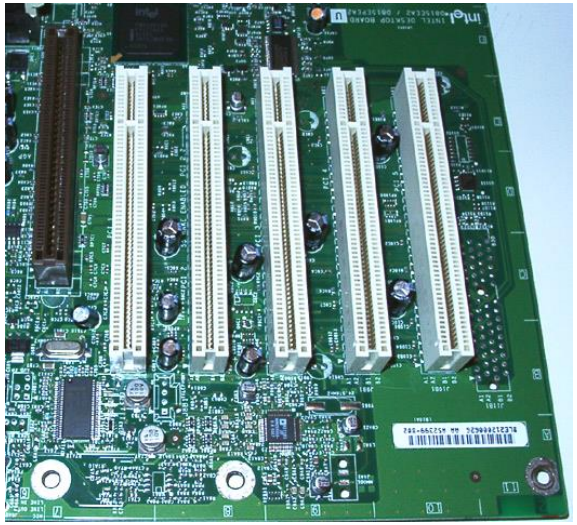
Introducción



Introducción

¿Qué es un bus?

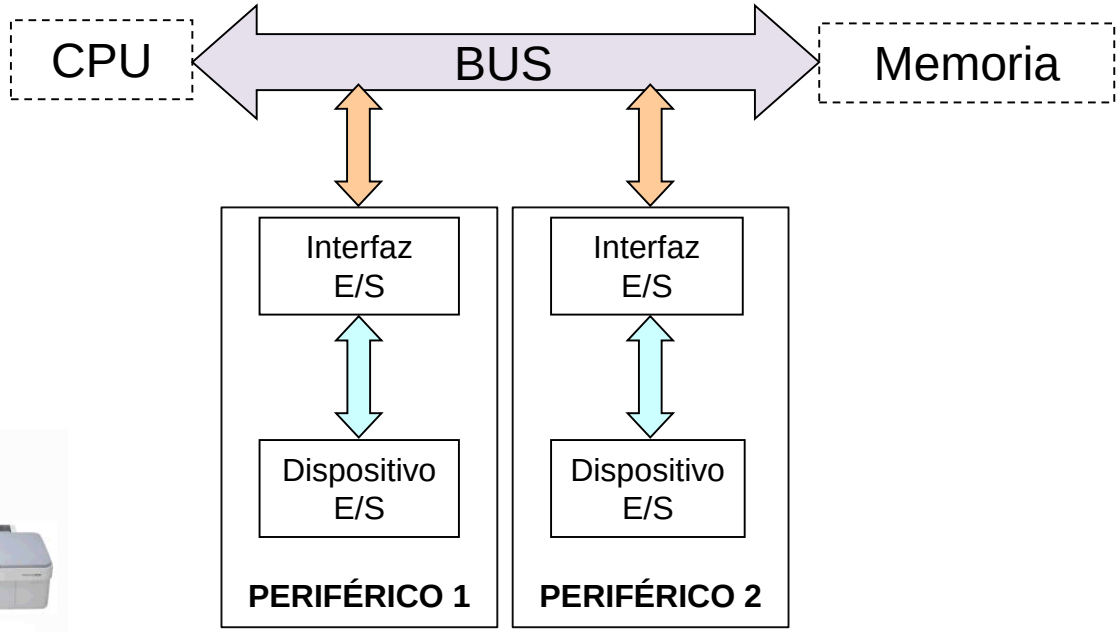




Conjuntos de líneas que interconectan dos o más elementos funcionales. Transfieren información.

Introducción

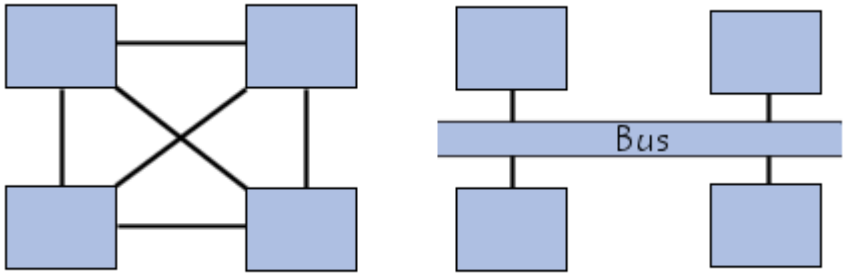
¿Qué es un periférico?



Elemento funcional que transmite información del exterior a la CPU y/o de la CPU al exterior.

Buses. Estructura y funcionamiento

- Se denomina **bus**, al conjunto de conexiones físicas (cables, pistas de circuito impreso, etc.) que pueden compartirse con múltiples componentes de hardware, con el objetivo de comunicar entre información.



- El propósito de un bus compartido es reducir el número de rutas necesarias para la comunicación entre los distintos componentes.

Buses. Estructura y funcionamiento

- Un bus se caracteriza por la cantidad de información que se transmite en un segundo. Este parámetro se llama velocidad de transferencia del bus y depende de dos factores:

  - **Número de líneas físicas de datos** mediante las cuales se envía la información en forma simultánea. Un cable plano de 32 hilos permite la transmisión de 32 bits en paralelo. El término "**ancho**" se utiliza para designar el número de bits que un bus puede transmitir simultáneamente.

  - **Velocidad del bus** (expresada en Hercios, 1 Hz = 1 s^{-1}) es la frecuencia del bus, es decir el número de paquetes de datos que pueden ser enviados o recibidos por segundo. Cada vez que se envían o reciben estos datos podemos hablar de ciclo.

Buses. Estructura y funcionamiento

- **Velocidad de transferencia o ancho de banda:** cantidad de datos que puede transportar por unidad de tiempo (expresada en baudios o bits por segundo)

$$V_t = \text{frecuencia} \times \text{nº bits}$$

Ejemplo: bus con un ancho de 16 bits y una frecuencia de 133 MHz

$$V_t = 16 \times 133 \times 10^6 = 2128 \times 10^6 \text{ bits/s}$$
$$= 2128 \times 10^6 / 8 \text{ Bytes/s} =$$
$$= 266 \times 10^6 \text{ Bytes/s} =$$
$$= 266 \times 10^6 / 2^{20} \text{ MB/s} =$$
$$= 253{,}67 \text{ MB/s}$$

Buses. Estructura y funcionamiento

- Ejemplos de buses:

Norma	Ancho del bus (bits)	Velocidad del bus (MHz)	Ancho de banda (MB/seg.)
PCI 32 bits	32	33	127,2
PCI 64 bits	64	66	508,6
AGP	32	66	254,3

Buses. Estructura y funcionamiento

Norma	Ancho del bus (bits)	Velocidad del bus (MHz)	Ancho de banda (MB/seg.)
ATA133	16	66	133
ATA serial (S-ATA,)	1		180
ATA serial II (S-ATA2)	2		380
USB	1		1,5
USB 2.0	1		60
USB 3.0	1		600
FireWire	1		100
FireWire 2	1		200

Buses. Jerarquía

- A más dispositivos conectados al bus, mayor es el retardo de propagación.

- La diferencia de velocidad de los dispositivos afecta negativamente al rendimiento global, ya que mientras los dispositivos lentos realizan una única transferencia, otro dispositivo más rápido podría haber realizado muchas más.

- A medida que aumenta el número de peticiones de transferencia, se puede producir un cuello de botella.

Buses. Jerarquía

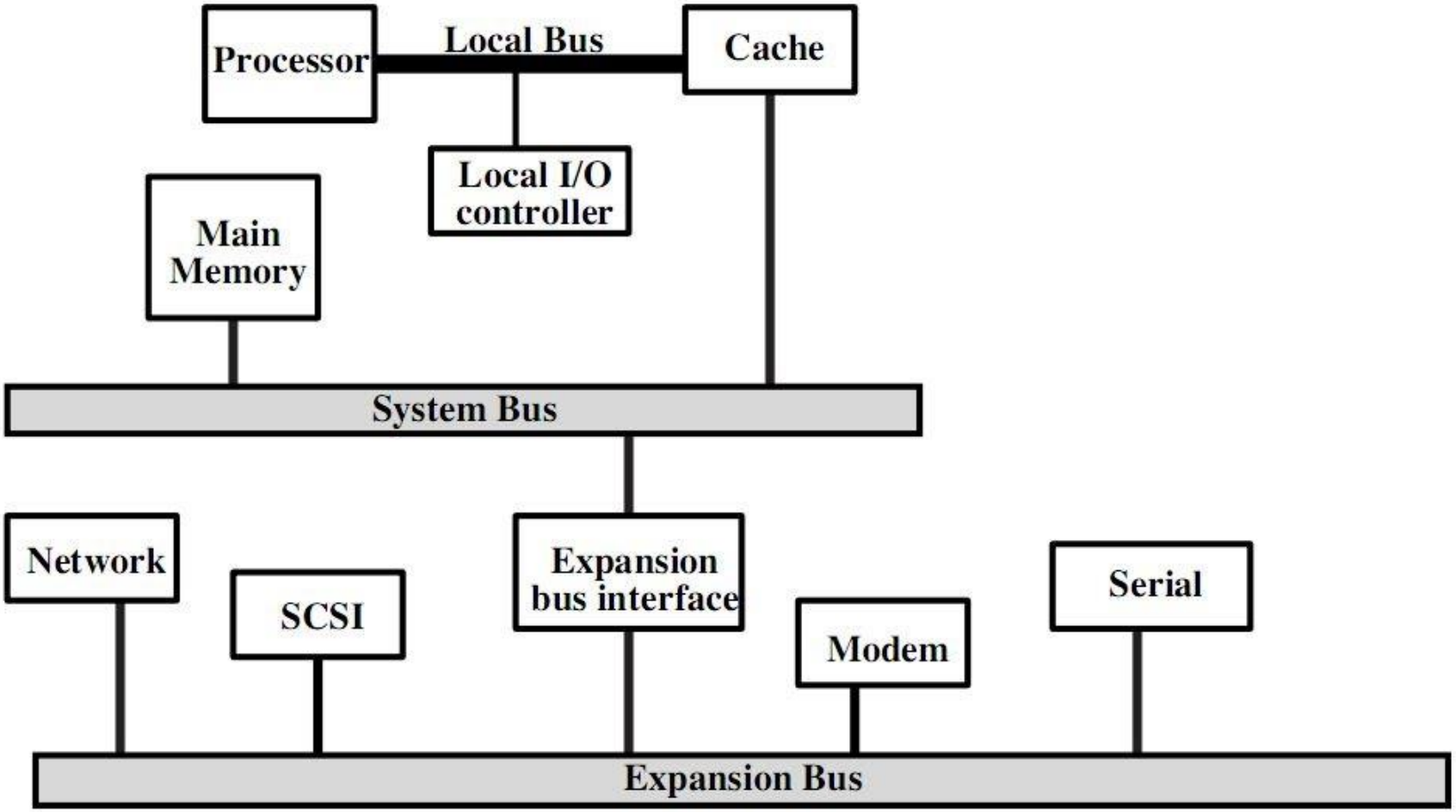
- Solución: dividir el bus de sistema en una serie de sub-buses adaptados a los dispositivos que están conectados a cada uno de ellos: **jerarquía de Buses**

- ¿Cómo hacer la división? Depende de cada sistema.

- En general, los buses más rápidos serán los más cercanos a la CPU y los más lentos los más alejados

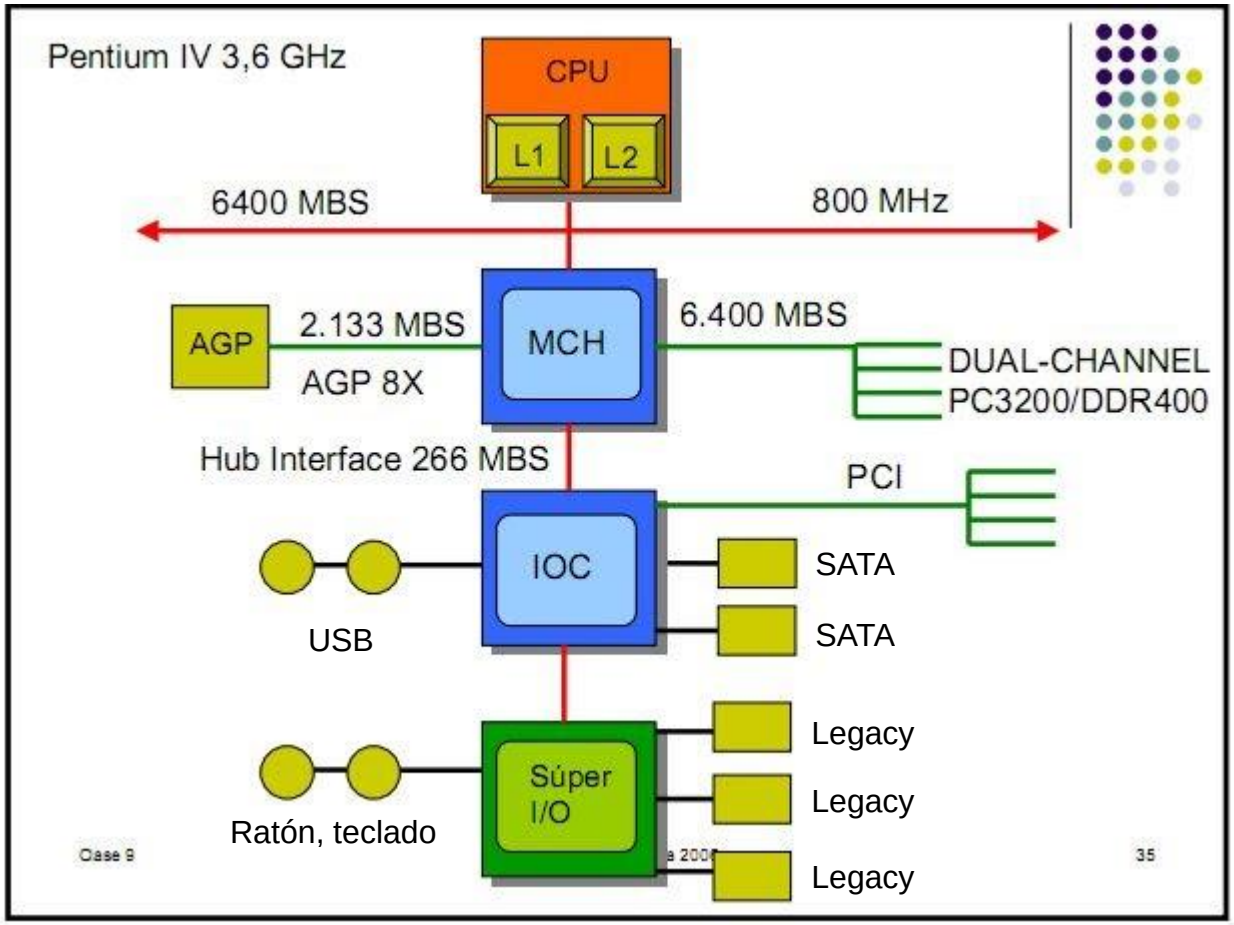
- Los buses se interconectan mediante adaptadores que permiten al bus más rápido considerar al más lento como un periférico más

Buses. Jerarquía



(a) Traditional Bus Architecture

Buses. Jerarquía



MCH:

Memory Control Hub

IOC:

I/O controler

Periféricos. Concepto

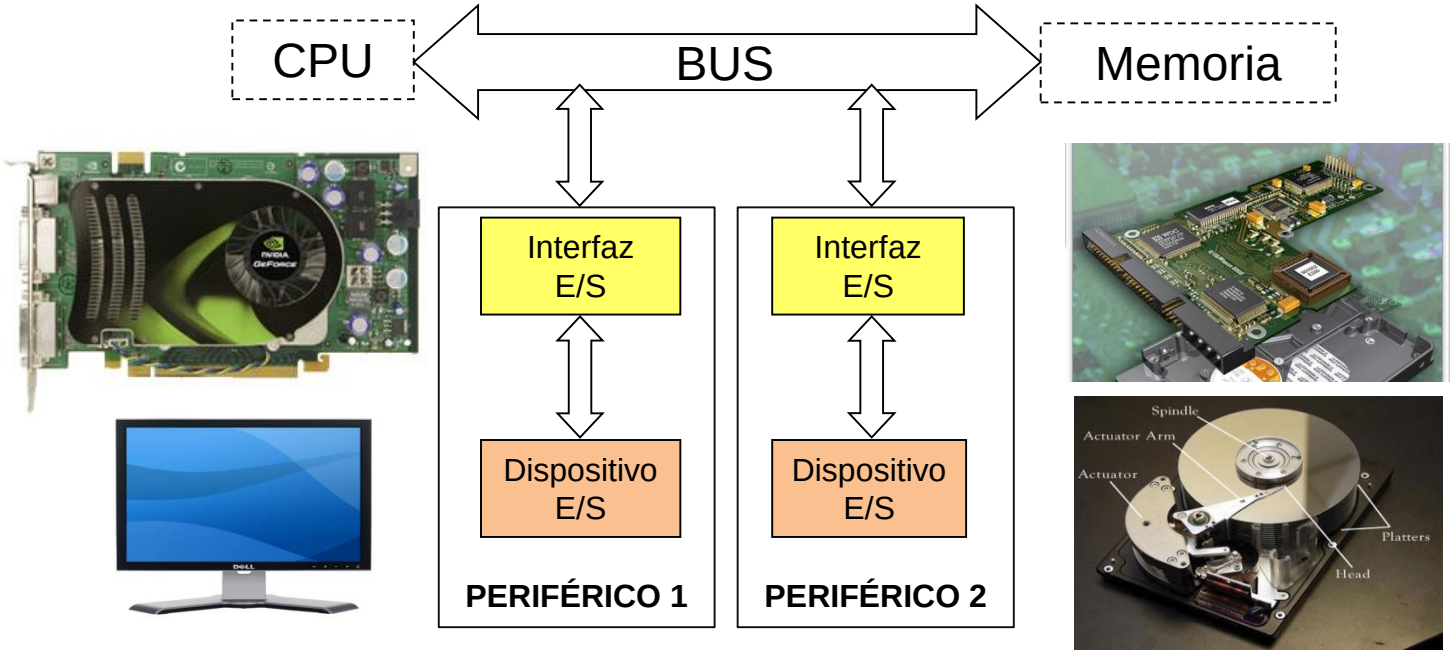
Elemento funcional que transmite información del exterior a la CPU y/o de la CPU al exterior.

Tipos:

- **Comunicación** (conectan el computador con el mundo exterior)
  - Hombre-máquina: Teclado, ratón, impresora, plotter, escáner, …
  - Máquina-máquina: módem, tarjeta de red…

- **Almacenamiento** (permiten almacenar información)
  - Acceso directo: Discos, DVD, …
  - Acceso secuencial: Cintas
  - Acceso aleatorio: memoria USB, …

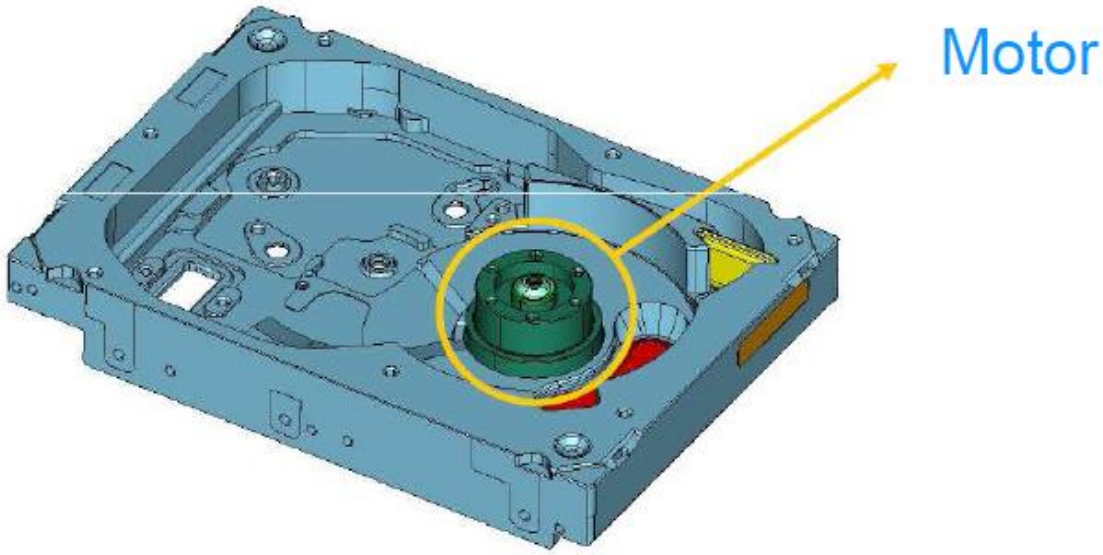
Periféricos. Estructura



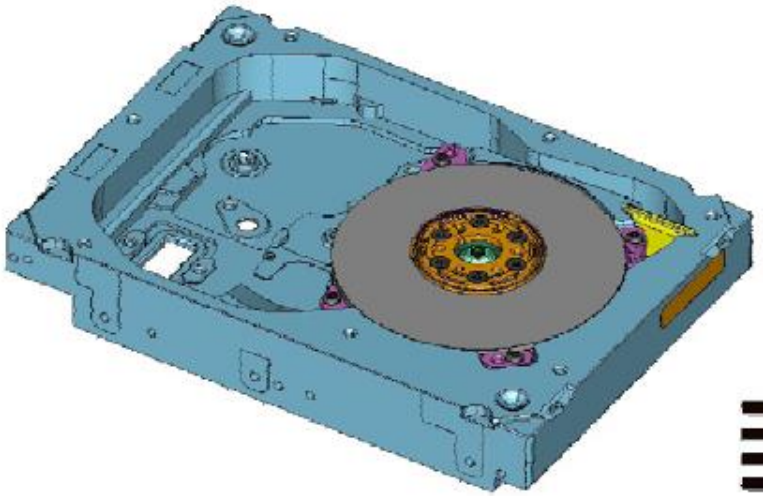
El periférico está formado por:

- Dispositivo E/S: hardware que interactúa con el entorno.
- Interfaz E/S (unidad E/S, controlador): elemento que adapta la información para que pueda ser transmitida/recibida por los buses del sistema y también pueda ser recibida/transmitida por los dispositivos E/S.

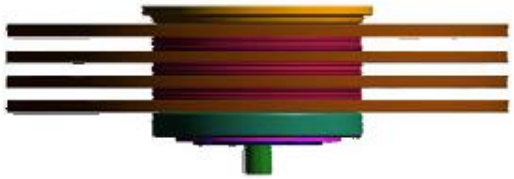
Periférico. Ejemplo Disco duro



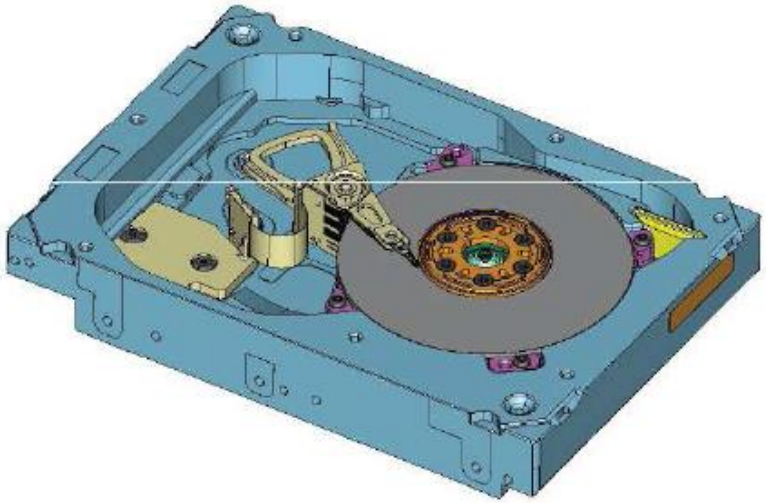
Periférico. Ejemplo Disco duro



Discos

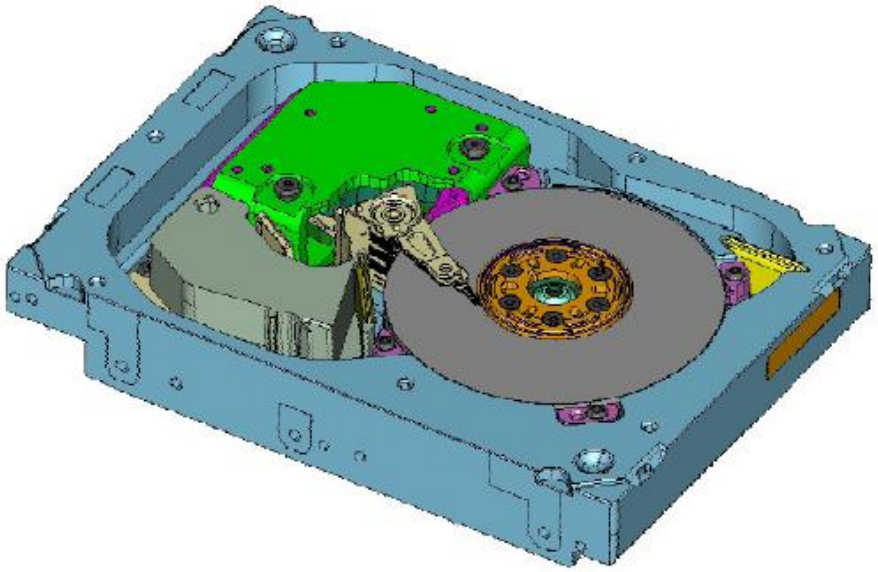


Periférico. Ejemplo Disco duro



Cabezas
lectoras/escritoras

Periférico. Ejemplo Disco duro



Módulo de control
y mecánica

Periférico. Ejemplo Disco duro

Electrónica de control:

Planificación de comandos

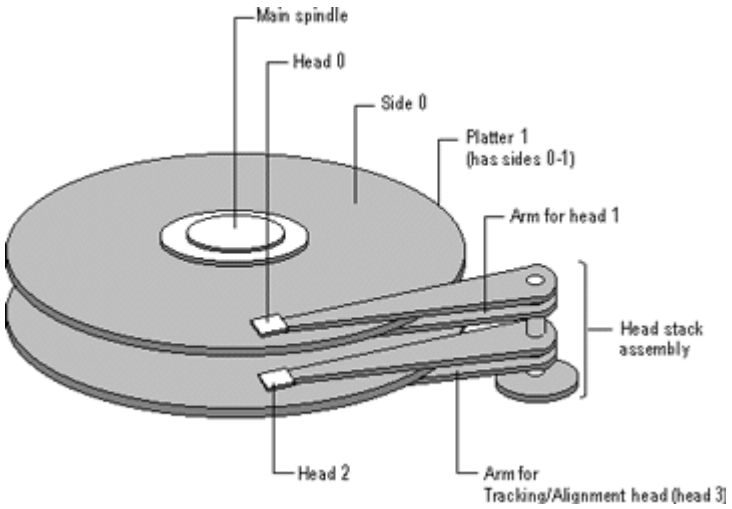
Corrección de errores

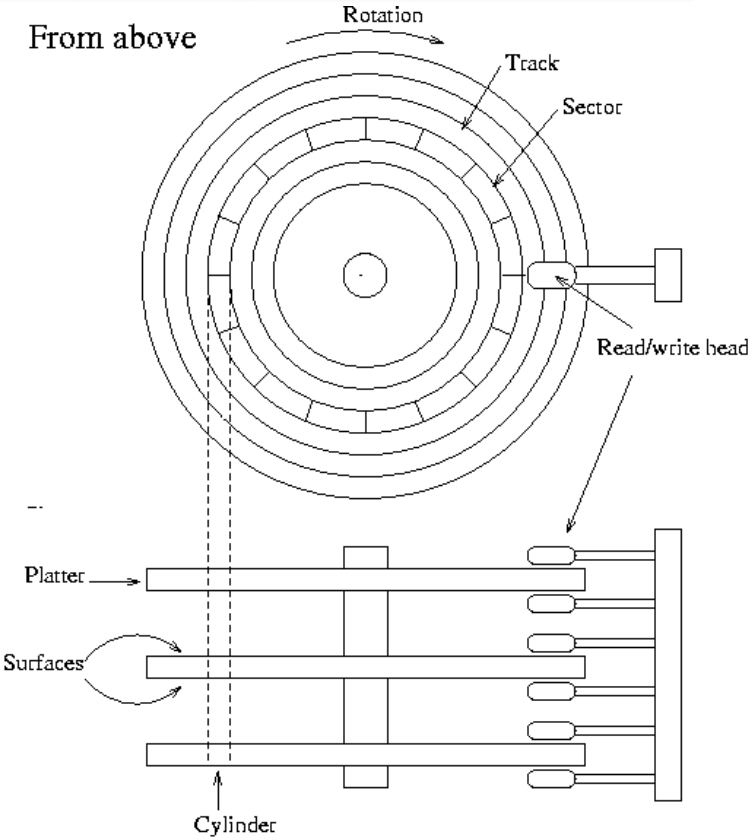
Optimización de transferencia

Comprobación de integridad

Control de las revoluciones por minuto (RPM)

Periférico. Ejemplo Disco duro





Pista (Track): Un anillo del plato

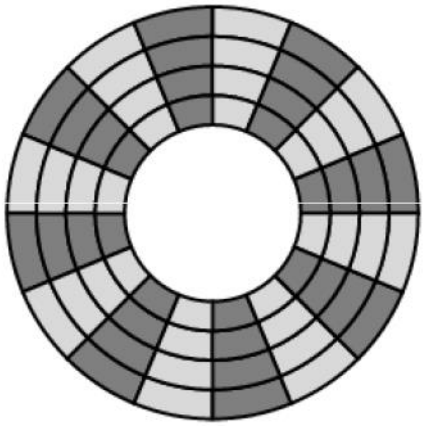
Sector: División de la superficie del disco realizada en el formateo (típicamente 512bytes)

Superficie (Surface): caras de los discos

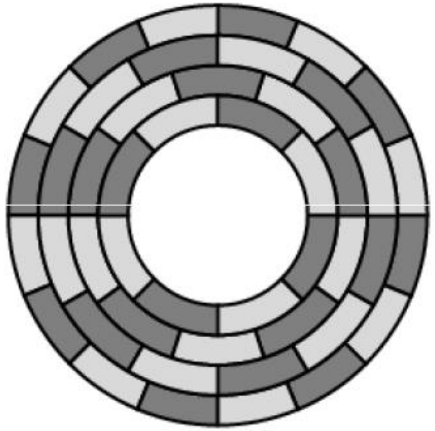
Cabeza lectora (Head). Graba y lee la información que está almacenada

Cilindro: Conjuntos de pistas colocada coincidentes en todos los discos

Periférico. Ejemplo Disco duro



(a) Constant angular velocity



(b) Multiple zoned recording

Para discos con velocidad angular constante

ns: número de superficies

p: número de pistas por superficie

s: número de sectores por pista

ts: bytes por sector

$$Capacidad = n_s \times p \times s \times t_s$$

Para discos con múltiples zonas

z: número de zonas

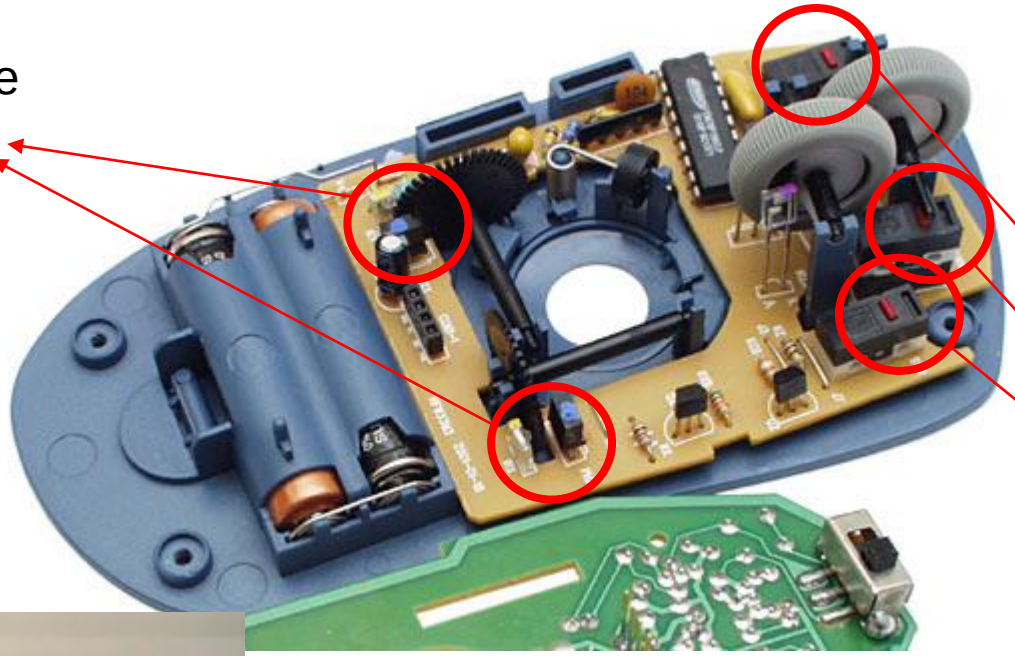
pi: pistas de la zona i

si: sectores por pista de la zona i

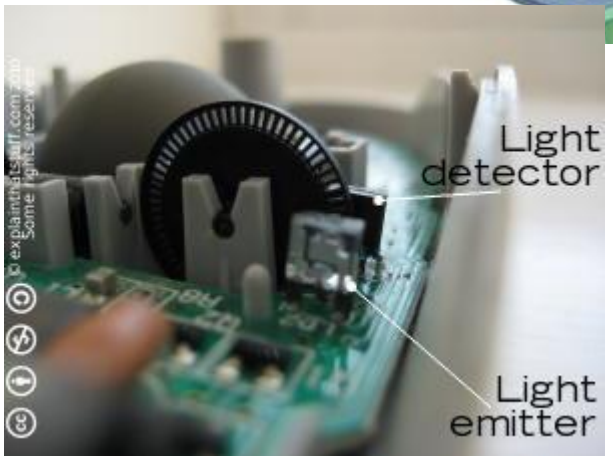
$$Capacidad = n_s \times t_s \times \sum_{i=1}^{z} (p_i \times s_i)$$

Periférico. Ejemplo Ratón

Transductores de movimiento X-Y



Pulsadores

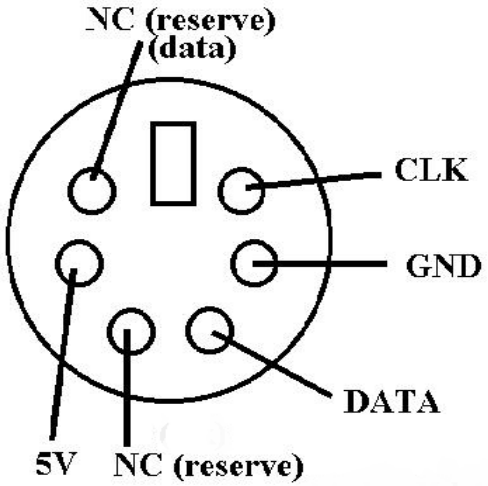


Cada transductor cuenta cuantos cambios se producen durante un tiempo determinado y manda los datos de X e Y para actualizar la visualización. Junto con estos datos se envía el estado los pulsadores.

Periférico. Ejemplo Ratón

Los datos se transmite por medio un bus serie: Rs232, PS2 o USB. También admite comunicación inalámbrica Bluetooth o propia del fabricante.

Nosotros nos centraremos en PS2.



El conector tiene los siguiente pines:

Clock- señal de reloj para transferencias síncronas

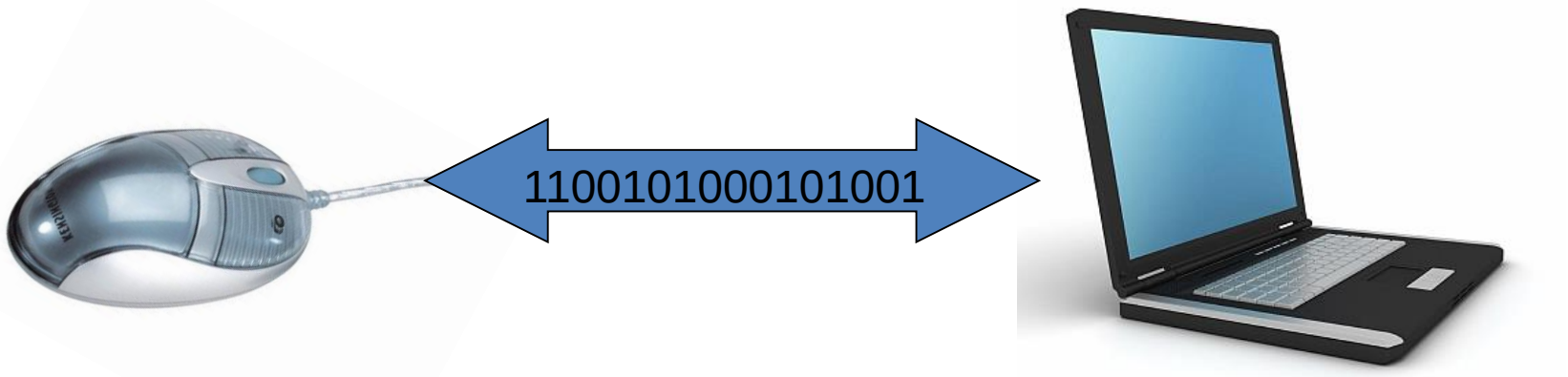
Alimentación: **GND**, **+5V**

DATA: línea de transmisión serie

Periférico. Ejemplo Ratón

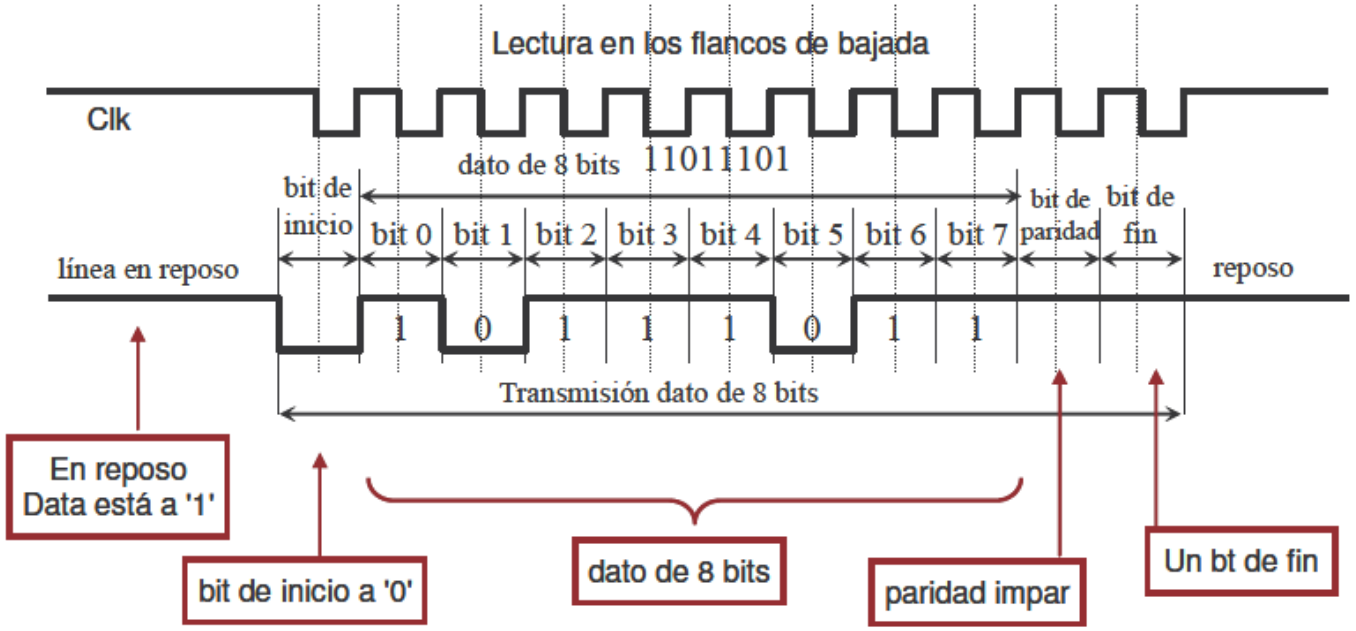
El teclado y ratón PS/2 implementan un protocolo serie bidireccional síncrono con el ordenador.

El bus esta en espera (idle) cuando ambas líneas (dato y reloj) están a ‘1’. La computadora tiene control total sobre el bus, pudiendo inhibir la comunicación en cualquier momento poniendo un cero en la línea de reloj.



Periférico. Ejemplo Ratón

Las tramas enviadas por el protocolo PS/2 son de 11 bits, teniendo un bit de inicio, 8 bits de datos, un bit de paridad impar y un bit de fin.

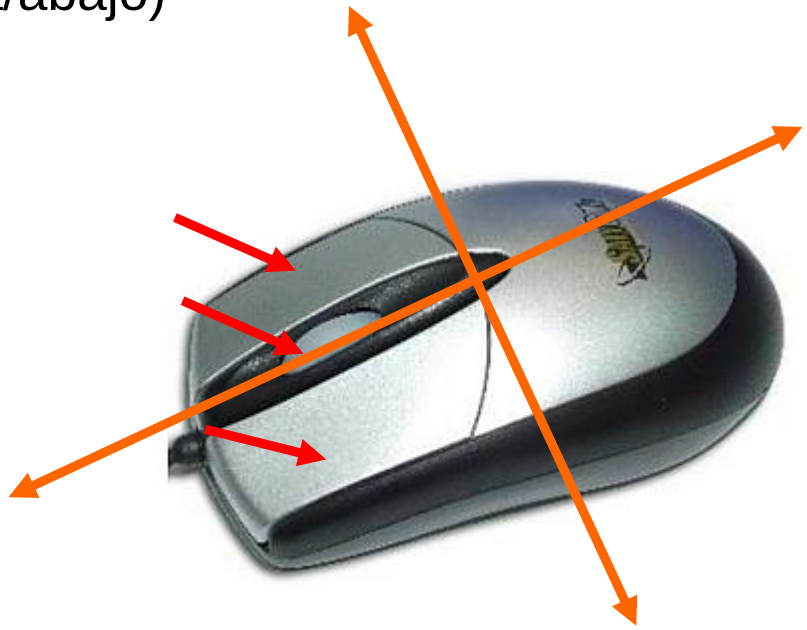


Periférico. Ejemplo Ratón

El interfaz estandar PS/2 de ratón admite las siguientes entradas:

- Movimiento en el eje X (izquierda/derecha)
- Movimiento en el eje Y (arriba/abajo)
- Botón izquierdo
- Botón central
- Botón derecho

El ratón lee estas entradas a una frecuencia determinada y actualiza los contadores internos e indicadores para reflejar los estados de movimiento y botones.



Periférico. Ejemplo Ratón

El ratón estándar tiene dos contadores que recogen la información del movimiento: los contadores de movimiento en X y en Y.

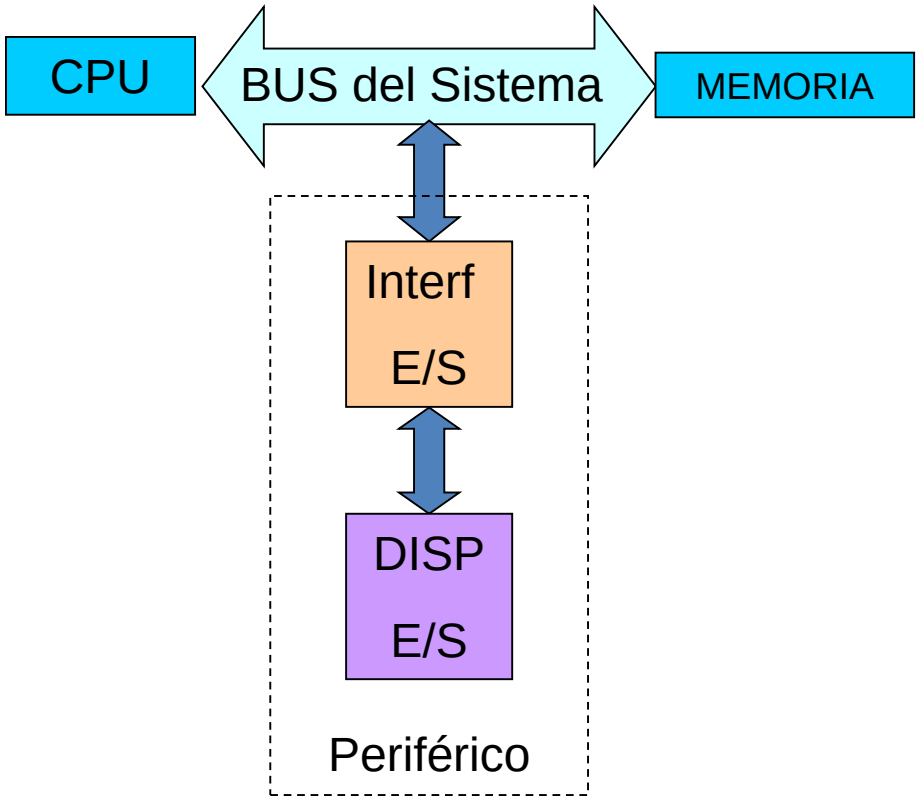
Estos valores se codifican con 8 bits en complemento a 2, teniendo cada uno asociado un bit de desbordamiento. El contenido de estos contadores, así como el estado de los tres botones, se envían a la computadora en un paquete de tres bytes.

Los contadores de movimiento representan el movimiento que el ratón ha experimentado desde que se envió el ultimo paquete, por tanto representan posiciones relativas.

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
Byte 1	Overfl.Y	Overfl. X	Signo Y	Signo X	'1'	bot. Medio	bot. dcha	bot. izq.
Byte 2	Movimiento X							
Byte 3	Movimiento Y							

Interfaces de E/S. Definición

Interfases de E/S: Son los elementos funcionales que deben comunicar la CPU con los dispositivos E/S.

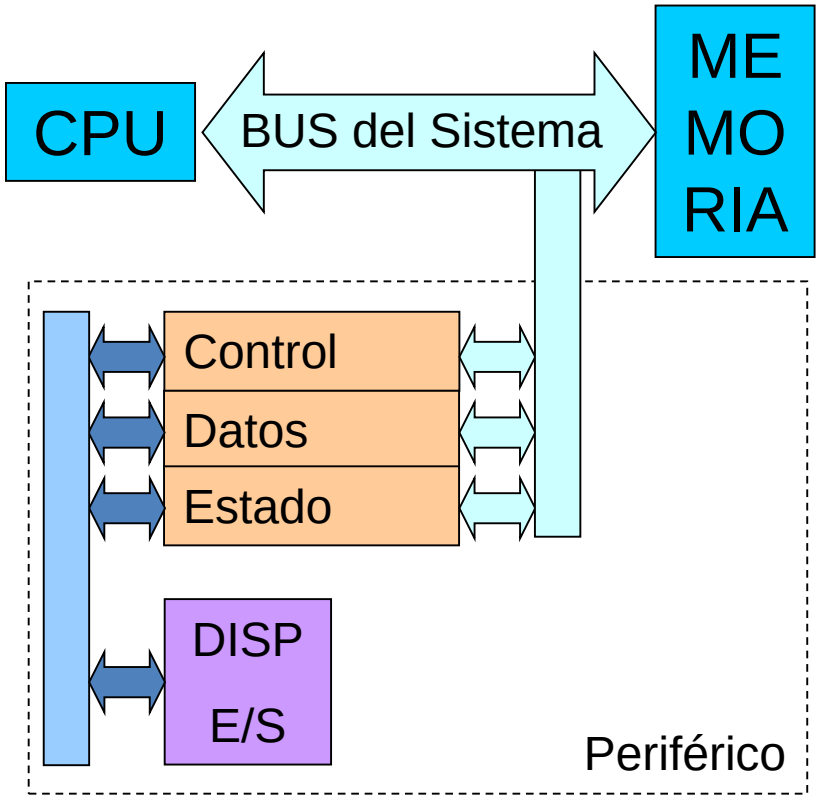


Son necesarias porque:

- Cada periférico es particular
- La velocidad de transferencia es menor que la empleada para transmitir entre la memoria y la CPU
- Los formatos, protocolos, especificaciones físicas y temporales son diferentes de las de las que emplea la CPU

Interfaces de E/S. Acceso

La CPU interacciona con la unidad de E/S por medio de tres registros



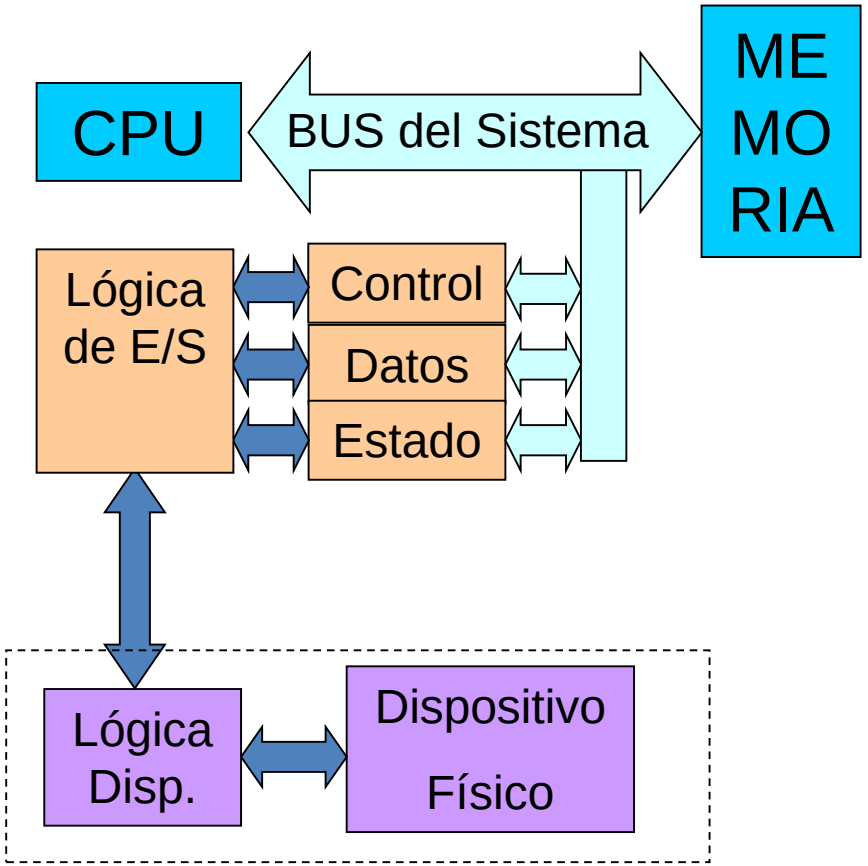
Control: Ordenes al periférico

Estado: Cómo está el periférico en cada momento

Datos: Información que se comunica

Interfaces de E/S. Acceso

La CPU interacciona con la unidad de E/S por medio de tres registros



Control: Ordenes al periférico
– Encender o apagar
– Saltar página en impresoras
– Posicionar el brazo de un disco
– …

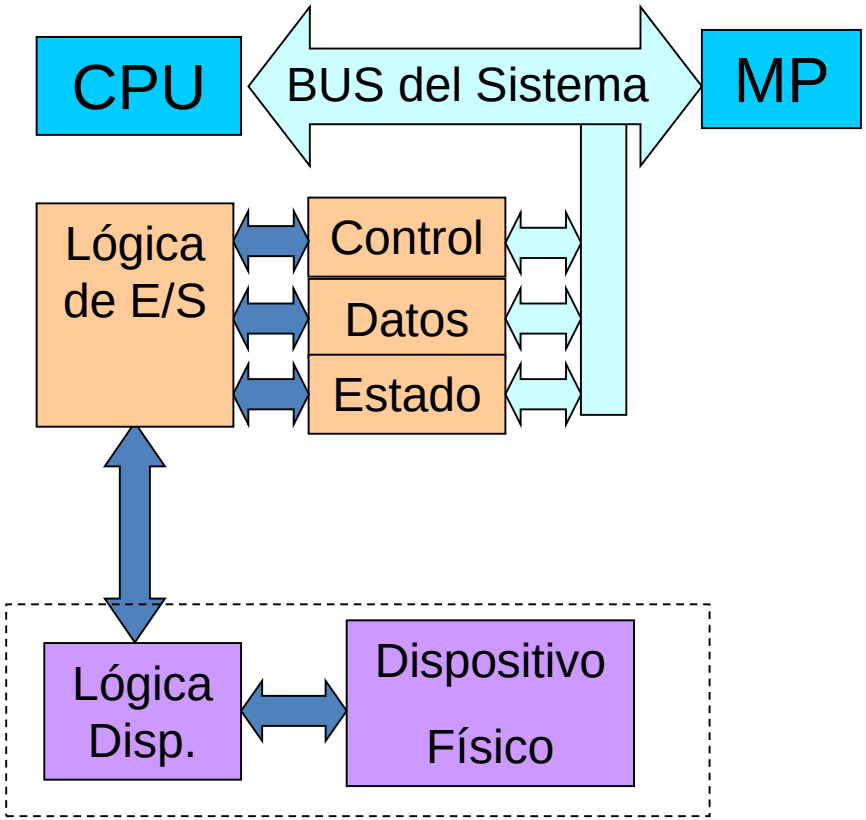
Estado: ¿Cómo está el periférico en cada momento?

– Nuevo dato
– Periférico encendido/apagado
– Periférico ocupado
– Periférico operativo o no
– Error de operación
– ….

Datos: Información que se comunica

Interfaces de E/S. Funciones

Los interfases de E/S tienen como funciones:



- **Atender al procesador:**
  - Decodificación de órdenes
  - Información de estado
  - Control y temporización
  - Ej.: datos a Memoria

- **Controlar periférico(s):**
  - Comunicación con dispositivos
  - Detección de errores
  - Almacenamiento temporal de datos, periférico->CPU

Interfaces de E/S. Tipos

Tipos de interfaces por complejidad:

Canal de E/S:
Contiene un microprocesador que se encarga de la mayoría de los detalles del procesamiento de E/S. Suele utilizar técnicas de acceso directo a memoria para desligarse completamente del uso de la CPU principal.

Controlador de E/S:
Es un módulo más simple, que requiere que la CPU tenga un control detallado del dispositivo. Normalmente utiliza técnicas de gestión basadas en interrupciones.

Puerto de E/S: es el modo de acceso más simple. La CPU lo controla normalmente en tiempo real de forma similar a la memoria

Interfaces de E/S. Características

Unidad de transferencia: puede ser por carácter, por bloque de bytes, por bloques de palabras, por bits, etc.

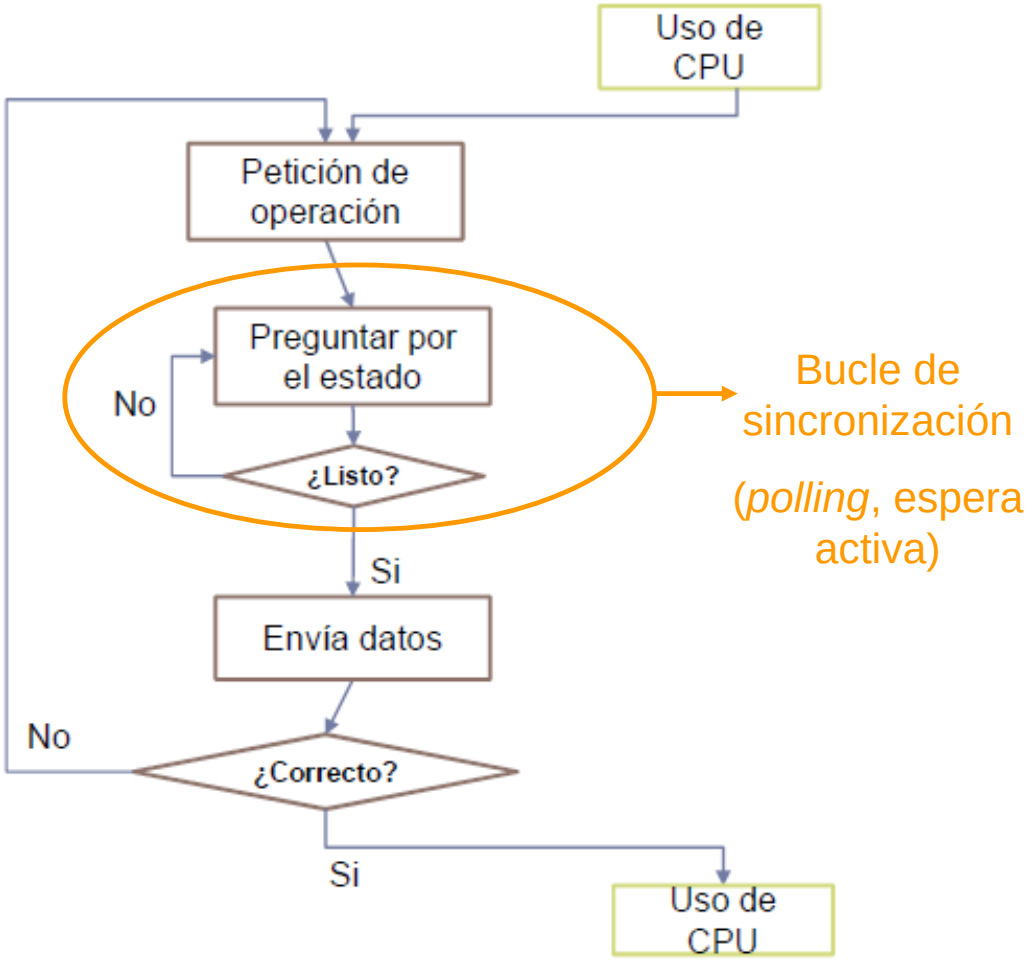
Modo de direccionamiento:

- Página coincidente: reúne en el mismo espacio memoria y E/S. Mismas instrucciones y señales.
- Páginas separadas. Memoria y E/S se direccionan es espacios diferentes. Deben existir instrucciones diferentes para cada tipo de acceso y alguna señales de selección diferenciadas.

Interacción CPU-controlador:

- E/S programada: La CPU controla directamente y en tiempo real. Opción menos eficaz y menos compleja de implementar.
- Interrupciones: El periférico es el que solicita que la CPU lo atienda
- DMA: la unidad E/S se desliga de la CPU. Los datos se transfieren directamente entre el controlador y memoria. Opción más eficaz, pero más compleja de implementar

E/S programada



E/S programada. Ejemplo

DATOS

Registros

Control:	0x300
Estado:	0x304
Datos:	0x308

Información de control (bit 0):

- 0 Leer, 1 escribir

Información de estado (bit 0)

- 0 ocupado, 1 Listo

Mapa de E/S en página coincidente. Instrucciones:

- LW y SW

E/S programada. Ejemplo

DATOS

Registros

Control:	0x300
Estado:	0x304
Datos:	0x308

Información de control (bit 0):

- 0 Leer, 1 escribir

Información de estado (bit 0)

- 0 ocupado, 1 Listo

Mapa de E/S en página coincidente. Instrucciones:

- LW y SW

Leer Dato:

1. Enviar la orden:

```
li $t0, 0
sw $t0, 0x300
```

2. Leer estado hasta que esté listo:

```
bucle:  lw, $t1 0x304
        beqz $t1, bucle
```

3. Leer el dato:

```
lw $t2 0x308
```

E/S programada. Ejemplo

DATOS

Registros

Control:	0x300
Estado:	0x304
Datos:	0x308

Información de control (bit 0):

- 0 Leer, 1 escribir

Información de estado (bit 0)

- 0 ocupado, 1 Listo

Mapa de E/S en página coincidente. Instrucciones:

- LW y SW

Escribir Dato:

1. Enviar la dato:

```
li $t0, 123
sw $t0, 0x308
```

2. Enviar orden:

```
li $t0, 1
sw $t0, 0x300
```

3. Leer estado hasta que esté listo

```
bucle:  lw $t1, 0x304
        beqz $t1, bucle
```

E/S programada. Ejemplo

DATOS

Registros

Control:	0x300
Estado:	0x304
Datos:	0x308

Información de control (bit 0):

- 0 Leer, 1 escribir

Información de estado (bit 0)

- 0 ocupado, 1 Listo

Mapa de E/S en página coincidente. Instrucciones:

- LW y SW

PROGRAMA:

Codifique un programa en ensamblador que lee 100 datos usando la unidad de E/S descrita, y los almacene En la dirección de memoria principal dada por la etiqueta ‘datos’.

E/S programada. Ejemplo

DATOS

Registros (mapeados como direcciones de memoria)

Control:	0x300
Estado:	0x304
Datos:	0x308

Información de control (bit 0):

- 0 Leer, 1 escribir

Información de estado (bit 0)

- 0 ocupado, 1 Listo

Mapa de E/S en página coincidente. Instrucciones:

- LW y SW

```
.data
    datos: .space 400
.text
.globl    main
main:     li $t3, 0
bucle1:   li $t0, 0
    sw $t0, 0x300
bucle2:   lw $t1, 0x304
    beqz $t1, bucle2
    lw $t2, 0x308
    sw $t2, datos($t3)
    add $t3, $t3 4
    bne $t3, 400 bucle1
```

E/S programada. Ejemplo

DATOS

Registros (mapeados como direcciones de memoria)

Control:	0x300
Estado:	0x304
Datos:	0x308

Información de control (bit 0):

- 0 Leer, 1 escribir

Información de estado (bit 0)

- 0 ocupado, 1 Listo

Mapa de E/S en página coincidente. Instrucciones:

- LW y SW

```
.data
    datos: .space 400
.text
.globl    main
main:     li $t3, 0
bucle1:   li $t0, 0
    sw $t0, 0x300
bucle2:   lw $t1, 0x304
    beqz $t1, bucle2
    lw $t2, 0x308
    sw $t2, datos($t3)
    add $t3, $t3 4
    bne $t3, 400 bucle1
```

Bucle de sincronización

E/S programada. Ejemplo

DATOS

Registros (mapeados como direcciones de memoria)

Control:	0x300
Estado:	0x304
Datos:	0x308

Información de control (bit 0):

- 0 Leer, 1 escribir

Información de estado (bit 0)

- 0 ocupado, 1 Listo

Mapa de E/S en página coincidente. Instrucciones:

- LW y SW

```
.data
     datos: .space 400
.text
.globl     main
main:      li $t3, 0
bucle1:    li $t0, 0
     sw $t0, 0x300
bucle2:    lw $t1, 0x304
     beqz $t1, bucle2
     lw $t2, 0x308
     sw $t2, datos($t3)
     add $t3, $t3 4
     bne $t3, 400 bucle1
```

Bucle de transferencia

E/S programada. Ejemplo

Problema:

Sea un computador con la capacidad de ejecutar 200 millones de instrucciones por segundo (200 MIPS)

Se conecta el módulo de E/S anteriormente descrito siendo el tiempo medio de espera de lectura de 5 ms

Calcule cuántas instrucciones se ejecutan en el bucle de sincronización y en el bucle de transferencia para el programa mostrado

```
.data
     datos: .space 400
.text
.globl      main
main:       li $t3, 0
bucle1:     li $t0, 0
     sw $t0, 0x300
bucle2:     lw $t1, 0x304
     beqz $t1, bucle2
     lw $t2, 0x308
     sw $t2, datos($t3)
     add $t3, $t3 4
     bne $t3, 400 bucle1
```

Bucle de transferencia

E/S programada. Ejemplo

Problema:

Bucle de sincronización:

De media dura 5 ms
Se ejecuta 200 MIPS de media
Ibs = $200*10^6 * 5*10^{-3} = 10^6$

Bucle de transferencia:

$1_{\text{(li \$t3 0)}} + 6 * 100 + 10^6{}_{\text{(Ibs)}}$

Como puede comprobarse, en el bucle se ejecuta 1.000.601 instrucciones, de las cuales 1.000.000 corresponden al bucle de espera (el 99,9%)

Es un desperdicio de ciclos de la CPU

```
.data
    datos: .space 400
.text
.globl    main
main:     li $t3, 0
bucle1:   li $t0, 0
    sw $t0, 0x300
bucle2:   lw $t1, 0x304
    beqz $t1, bucle2
    lw $t2, 0x308
    sw $t2, datos($t3)
    add $t3, $t3 4
    bne $t3, 400 bucle1
```

Bucle de transferencia

E/S programada. Ejemplo

Conclusiones:

- Se desperdicia mucho tiempo en el bucle de espera

- Sólo será lógico utilizarla para operaciones específicas:
  - Salida, cuando no se requieran espacios de espera: Actualizar un visualizador externo.
  - Entrada de forma asíncrona desligada del periférico.

- Es necesario buscar alternativas más eficaces.

Interrupciones. Definición

Bifurcación asíncrona de la secuencia normal de ejecución.

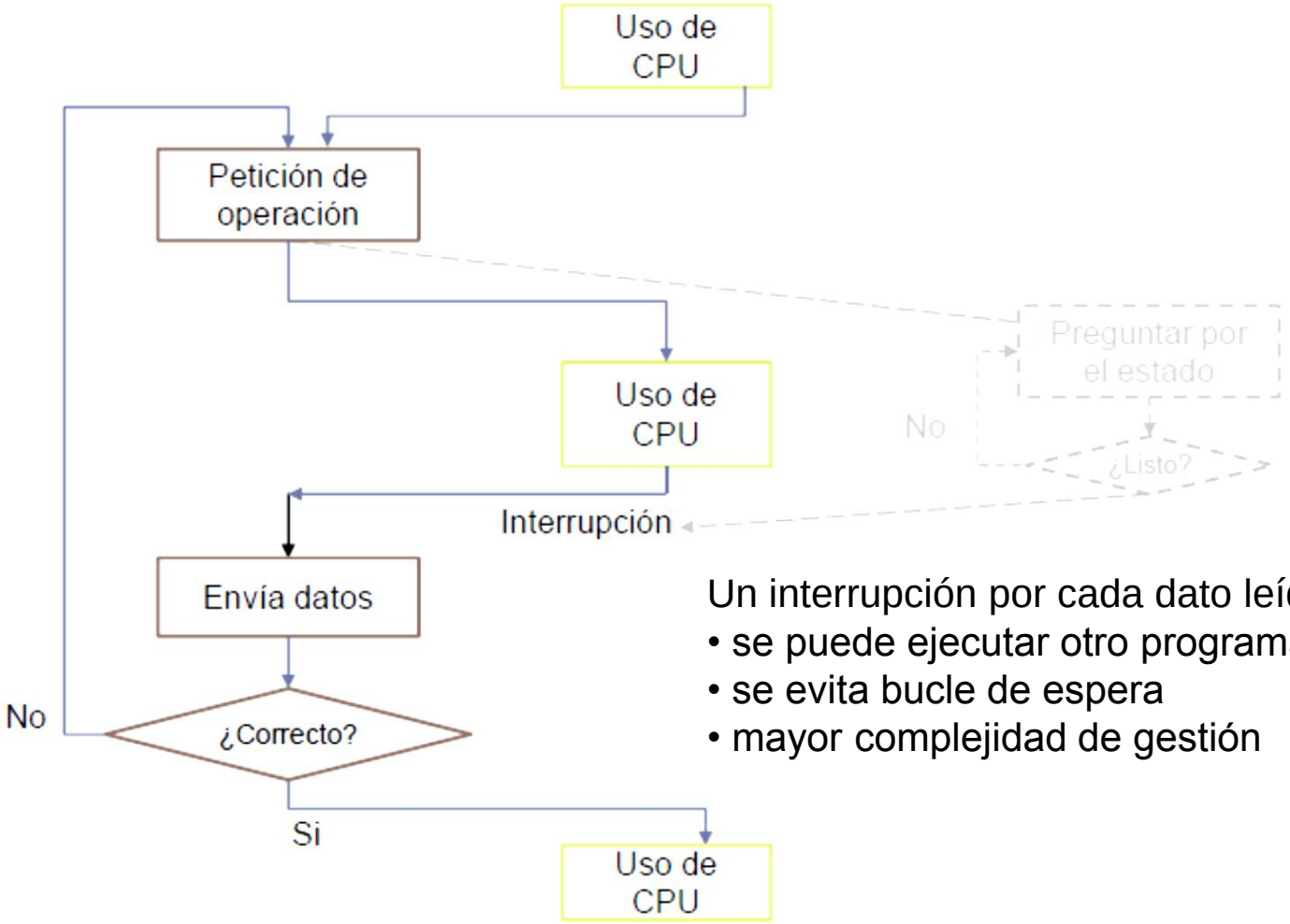
Cómo se produce: Señal externa llega a la unidad de control y tras completar la instrucción en curso, salta a la rutina de gestión.

Terminales:

INTREQ . Petición de interrupción

INTAK: Aceptación de interrupción

Interrupciones



Un interrupción por cada dato leído
- se puede ejecutar otro programa
- se evita bucle de espera
- mayor complejidad de gestión

Interrupciones. Gestión

Gestión de interrupciones: Se añade una nueva fase a la secuencia de ejecución de instrucciones.

1. Lectura de la Instrucción desde memoria
2. Decodificación:
3. Ejecución
4. **Ciclo de reconocimiento de interrupción**: Comprueba se hay activada alguna señal de interrupción. (¿Varias? → Prioridad)
    - Si está activada:
        - Salva el contador de programa y el registro de estado
        - Pasa de modo usuario a modo núcleo
        - Obtiene la dirección de la rutina de gestión de la interrupción
        - Almacena en el contador de programa la dirección obtenida (de esta forma la siguiente instrucción será la de la rutina de tratamiento)
    - Si no está activada. Se reinicia le ciclo de instrucción con normalidad

Interrupciones. Gestión

Rutinas de gestión de interrupciones: Encargada de dar respuesta a la petición de interrupción de la secuencia normal del programa.

Forman parte del código del sistema operativo. ¿Qué deben de realizar? ¿Cómo deben programarse?

1. Salvar el resto de registros del procesador
2. Atender la interrupción ejecutándose la rutina propiamente dicha. Tras terminar de ejecutarse,
3. Restaurar los registros del procesador utilizados por el programa interrumpido
4. Ejecutar una instrucción máquina especial, RETI, que
   - Restaura el registro de estado del programa interrumpido (fijando de nuevo el modo del procesador a modo núcleo)
   - Restaura el contador de programa (de forma que la siguiente instrucción es la del programa interrumpido).

Interrupciones. Clasificación

1. **Hardware:** Generadas por periféricos
2. **Trampas (Traps):** Síncronas con la ejecución del programa: división por cero, Modo paso a paso, etc
3. **Excepciones:** Ejecución de instrucciones inexistentes o no permitidas

---

1. **Enmascarables.** Si se pueden activar o desactivar
2. **No enmascarables**: Si no se pueden desactivar

Interrupciones. Clasificación

¿Cómo se gestiona la interrupción?

- **Software:** La rutina de gestión tiene que determinar quien ha solicitado la interrupción y la prioridad en el caso de coincidir varias
- **Hardware:** Se dispone de un sistema hard que determina la prioridad y la rutina de gestión

---

Modos de determinar la rutina de gestión:

- Dirección fija
- Tabla de direcciones programable
- Vectoriales (el periférico indica la rutina y la comunica por el bus de datos)

Interrupciones. Clasificación

Gestión de la prioridad

- **Software:** La rutina de gestión tiene que determinar quien ha solicitado la interrupción y la prioridad en el caso de coincidir varias
  - **Polling o encuesta:** La cpu pregunta a cada periférico hasta conocer quien ha realizado la petición
  - **Daisy Chain:** Los propios perifericos determinan la prioridad enviándose la petición unos a otros en una cadena serie
- **Hardware:** Se dispone de un sistema hard que determina la prioridad y la rutina de gestión. Procesadores de interrupciones.

Interrupciones. Clasificación

Modo de determinar la rutina de gestión:

- Dirección fija: Siempre se salta a una dirección fija donde la rutina determina todos las condiciones de ejecución de la rutina de gestión (Ej: Polling)

- Tabla de direcciones programable. Hay un nº limitado de terminales de interrupción y se asigna una rutina a cada uno de ellos.

- Vectoriales: el periférico indica la rutina y la comunica por el bus de datos

DMA. Definición

Método de acceso directo a memoria desde el exterior de la CPU. DMA= Acceso Directo a Memoria

Lo realiza un controlador (procesador) especializado, que permite conectar el periférico que lo solicita a la memoria

Terminales:

BUSREQ . Petición de acceso a bus

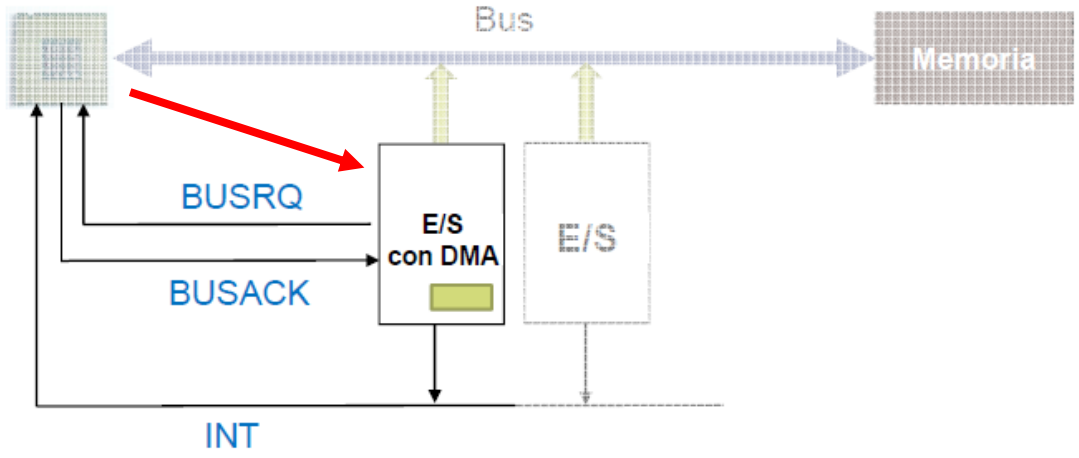
BUSAK: Aceptación de acceso a bus

DMA. Definición

Antes de iniciar la transferencia hay que conocer:

- Dirección de memoria inicial de envío de datos
- Nº de datos a transferir
- Modo de transferencia

La CPU debe programar el controlador con estos datos

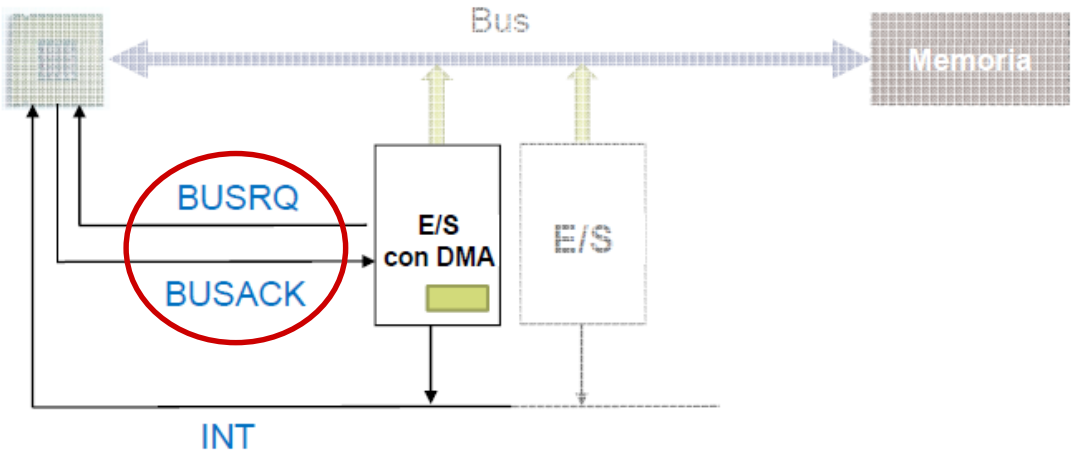


DMA. Definición

A partir de ese momento el controlador se encarga de todo el proceso.

Cada dato transferido a memoria supone:

- Pedir permiso para acceder a memoria (BUSRQ)
- Esperar el permiso (BUSACK) (Buses de la CPU en triestado)
- Transfiere a memoria
- Desactiva petición de permiso (BUSRQ).
- Al siguiente ciclo de reloj, la CPU vuelve a la normalidad



DMA. Definición

Una vez transferidos todos los datos:

- Generar una interrupción (INT) para avisar a la CPU