Tema 6

CONEXIÓN ENTRE SUBSISTEMAS:
BUSES

ÍNDICE

- Definiciones, estructura y jerarquía de buses
- Características de diseño
  - Anchura
  - Tipos
  - Arbitraje
  - Temporización
  - Tipos de transferencia
- Ejemplos de buses comerciales

Buses

Estructura de interconexión

Un computador está constituido por un conjunto de unidades o módulos de tres tipos (procesador, memoria y E/S) que deben comunicarse entre sí → se hace necesaria una estructura de interconexión. Las más comunes son las estructuras de bus y de buses múltiples.

Interconexión con buses

Se trata de un camino de comunicación entre dos o más dispositivos. Al ser un medio compartido se hace necesario un mecanismo de control (arbitraje) para que varias señales no se solapen y distorsionen. Constituido usualmente por varias líneas (anchura del bus), cada una de las cuales puede transmitir una señal binaria (un byte puede transmitirse mediante 8 líneas del bus).

Buses

Los computadores poseen diferentes tipos de buses para comunicar sus componentes a distintos niveles dentro de la jerarquía del sistema.

Estructura del bus

Bus del sistema: 50-100 líneas divididas en tres grupos funcionales

- Líneas de datos → bus de datos (anchura 8, 16, 32 o 64; factor clave a la hora de determinar las prestaciones del conjunto del sistema)
- Líneas de direcciones → bus de direcciones (anchura 8, 16, 32 o 64; determina la máxima capacidad de memoria en el sistema). También para direccionar puertos de E/S.
- Líneas de control → para controlar (mediante órdenes e información de temporización) el acceso y uso de las líneas de datos y de direcciones, ya que son compartidas por todos los componentes
- (puede haber también alguna de alimentación)

Buses

Líneas de control:

- Temporización: indican la validez del dato y las direcciones
- Órdenes: especifican la operación a realizar
  - Memory Write
  - Memory Read
  - I/O Write
  - I/O Read
  - Transfer ACK
  - Bus Request
  - Bus Grant
  - Interrupt Request
  - Interrupt ACK
  - Clock
  - Reset

Buses: jerarquía de buses

Jerarquía de buses

Las prestaciones pueden disminuir si se conectan muchos dispositivos al bus, principalmente debido a:

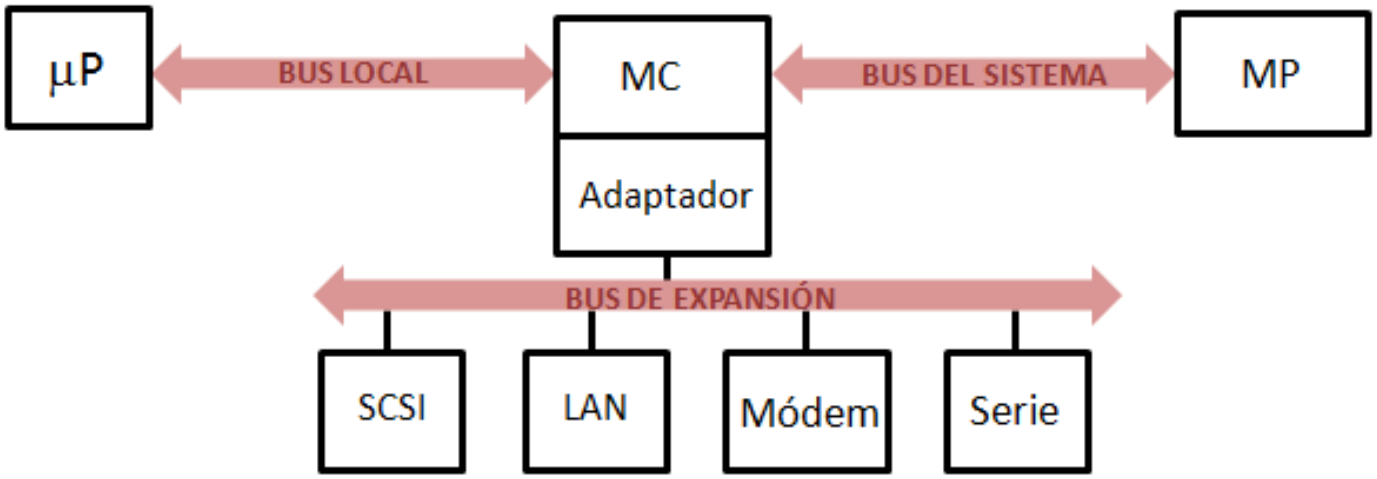
- Mayor tiempo de propagación → mayor tiempo para coordinarse
- Cuello de botella si las peticiones acumuladas se aproximan a su capacidad. Podría aumentarse la velocidad del bus (incrementando la anchura); pero la velocidad de transferencia que necesitan los dispositivos (controladores gráficos, interfaces de red,…) está incrementándose rápidamente…

⇒ Solución: varios buses organizados jerárquicamente

Buses: jerarquía de buses

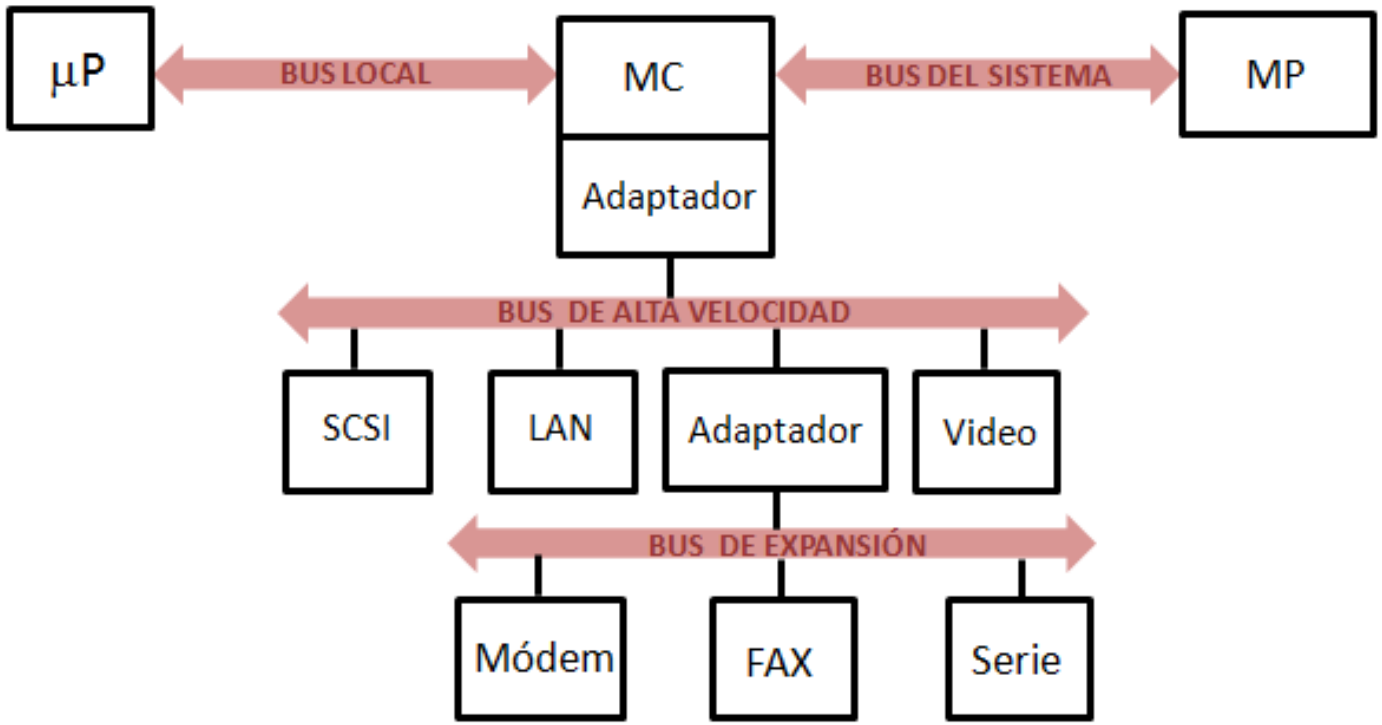
- Controladores de E/S conectados a uno o varios buses de expansión.
- Una interfaz regula la transferencia de datos entre el bus del sistema y estos controladores
- Permite conectar al sistema una amplia gama de dispositivos de E/S
- Aísla el tráfico de información entre la memoria y el procesador del tráfico correspondiente a la E/S

Buses: jerarquía de buses



Arquitectura tradicional de bus

Buses: jerarquía de buses



Arquitectura de bus de altas prestaciones

Buses: elementos de diseño

- Anchura del bus
- Tipos de buses
  - Dedicado
  - Multiplexado
- Arbitraje
  - Método
  - Estrategias
- Temporización
  - Síncrono
  - Asíncrono
- Tipo de transferencia de datos (operaciones)
  - Lectura
  - Escritura
  - Lectura-modificación-escritura
  - Lectura después de escritura
  - Bloque

Buses: características de diseño

- Anchura del bus

  Nº de señales eléctricas que transportan algún tipo de información (pistas de cobre). Cuanto más denso sea un bus, más cara la placa.

  - Bus de direcciones:
    - n bits => 2^n direcciones direccionables (puede necesitarse alguna línea para indicar dirección disponible)
  - Bus de datos:
    - Entre 8 y 64 bits
  - Bus de control:
    - No presentan limitaciones

Buses: características de diseño

- Tipos de buses
  - Dedicado
    - Bus de direcciones, bus de datos y bus de control separados.
  - Multiplexado
    - Un mismo conjunto de líneas eléctricas para direcciones y datos (multiplexadas en el tiempo). Una línea de control para discernir.

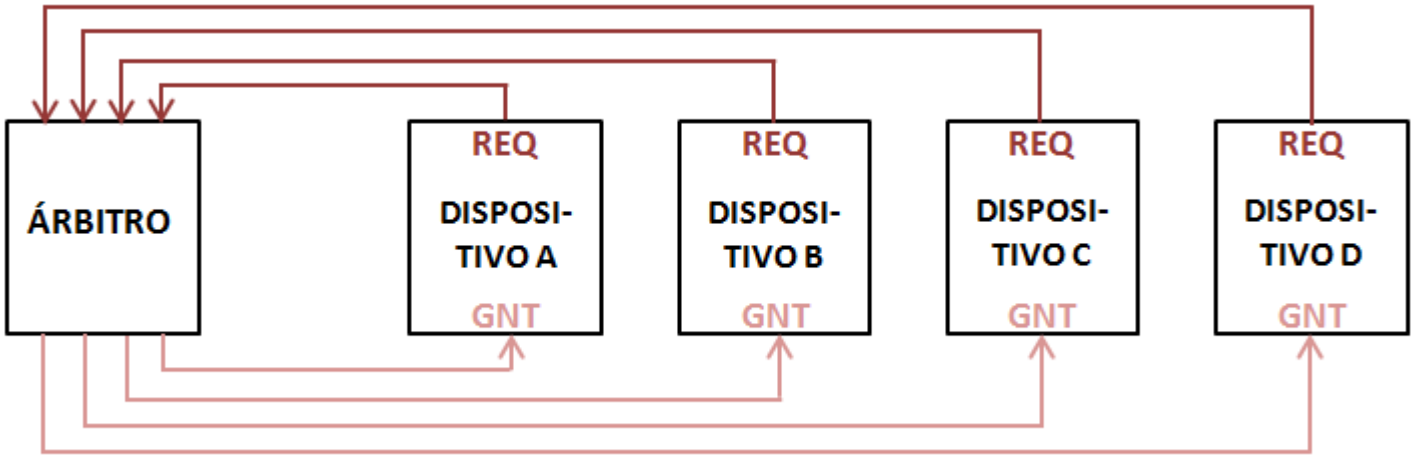
Buses: características de diseño

Más de un módulo puede necesitar el control del bus, para evitar colisiones → árbitro de bus

- Arbitraje: método
  - Centralizado: un único dispositivo hw (*controlador del bus* o *árbitro*) es responsable de asignar tiempos en el bus (CPU u otro dispositivo). *Ejemplo: bus PCI.*
  - Distribuido: cada módulo dispone de lógica para controlar el acceso y los propios dispositivos acuerdan quién lo usa en cada instante. *Ejemplo: Multibus.*

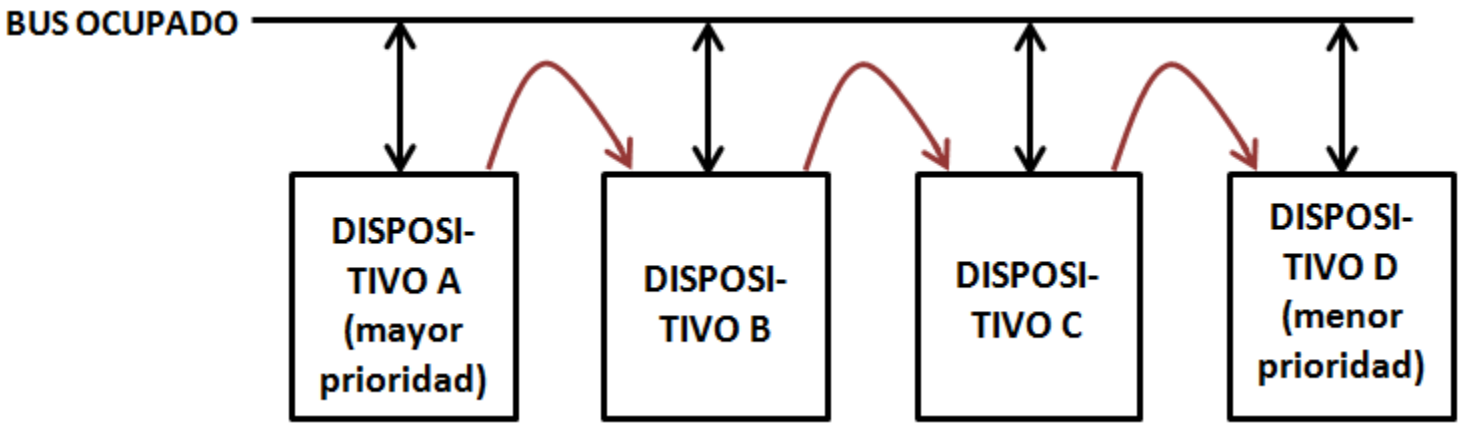
Buses: características de diseño

Ejemplo de arbitraje centralizado



Buses: características de diseño

Ejemplo de arbitraje distribuido

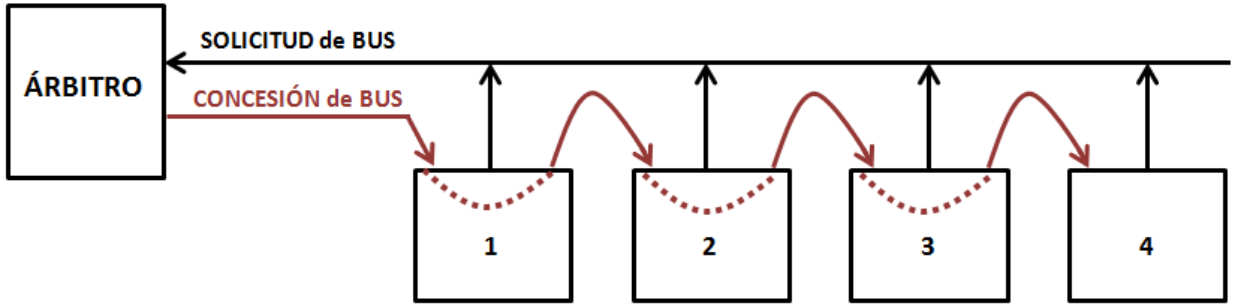


Buses: características de diseño

- Arbitraje: estrategia
  - Daisy-Chain (serie)
    - Centralizado
    - Distribuido
  - Encuesta
    - Centralizada
    - Distribuida
  - Petición independiente (paralelo)
    - Centralizada
    - Distribuida

Buses: características de diseño

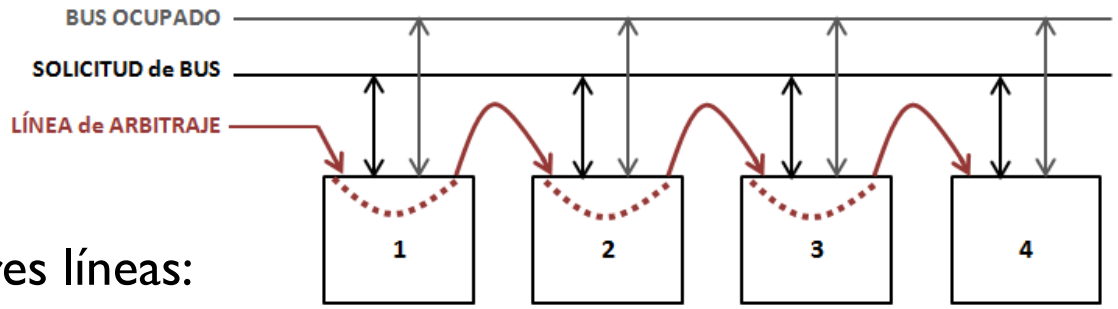
- Daisy-Chain distribuido



  - Ante una solicitud el árbitro sondea uno a uno. El más cercano al árbitro es el de mayor prioridad.
  - Ventajas:
    - Simplicidad
    - Facilidad para añadir dispositivos
  - Desventajas
    - Propenso a fallos
    - Prioridades fijas
    - Lentitud de funcionamiento

Buses: características de diseño

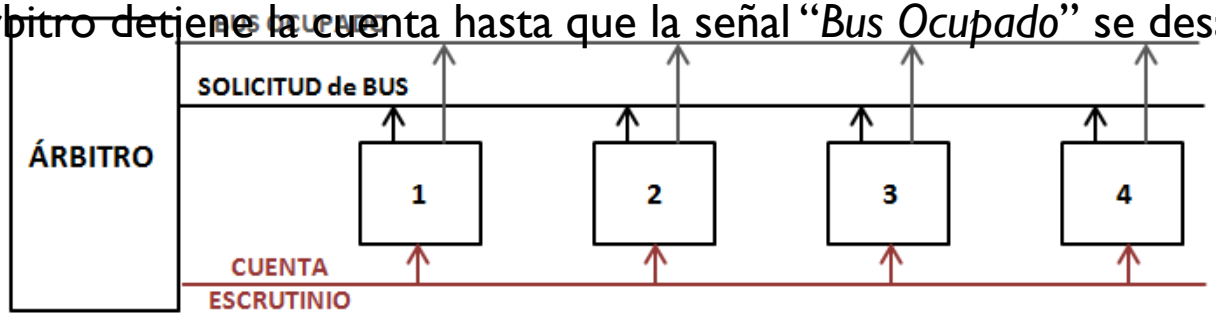
- Daisy-Chain distribuido



  - Tres líneas:
    - Solicitud de bus (OR de las todas las peticiones)
    - BUSY (activada por el dispositivo que tiene concedido el bus)
    - Línea de arbitraje (se conecta en serie a todos los dispositivos)
  - Ventajas
    - Simplicidad
  - Desventajas
    - Propenso a fallos (muy sensible la ruido eléctrico)
    - Prioridades fijas
    - Lentitud de funcionamiento

Buses: características de diseño

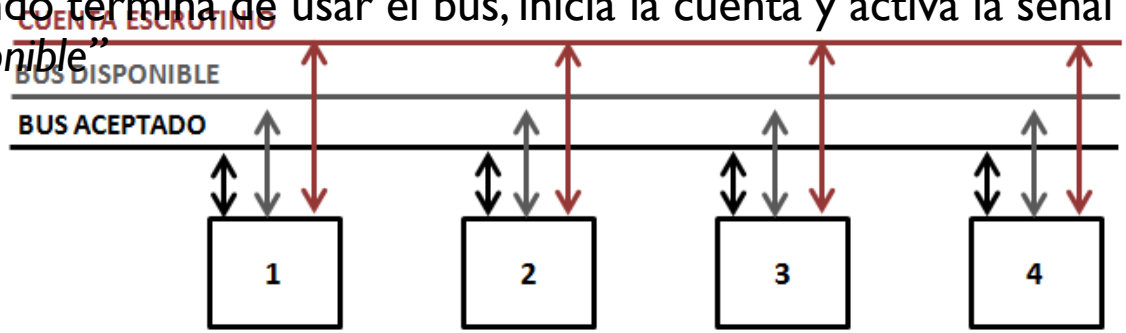
- Encuesta centralizado
  - Cada dispositivo posee un número de identificación
  - Cuando se activa la línea "*Solicitud Bus*" el árbitro inicia una cuenta
  - El dispositivo que desea usar el bus activa "*Bus Ocupado*" al detectar su identificador
  - El árbitro detiene la cuenta hasta que la señal "*Bus Ocupado*" se desactiva



  - Ventajas
    - Prioridad fácilmente alterable
    - Asignación de bus relativamente rápida
  - Desventajas
    - Complejidad

Buses: características de diseño

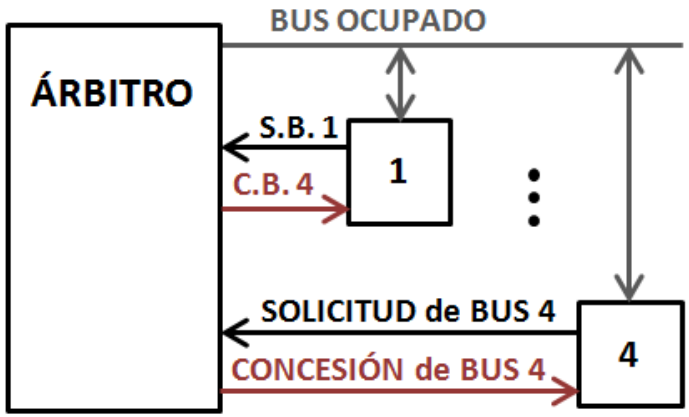
- Encuesta distribuido
  - El dispositivo cuyo código aparece en la cuenta puede optar al bus activando *"Bus Aceptado"* → se le cederá tanto el control como el arbitraje del bus
  - Cuando termina de usar el bus, inicia la cuenta y activa la señal *"Bus Disponible"*



  - Ventajas:
    - Prioridad no fija
    - Asignación del bus relativamente rápida
  - Desventajas:
    - Hay que asignar el bus a un dispositivo al iniciar el sistema
    - Complejidad

Buses: características de diseño

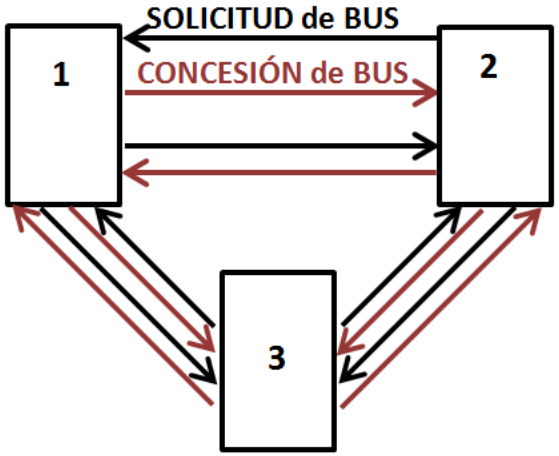
- Petición independiente centralizado
  - Cada dispositivo posee una línea "*Solicitud Bus*" que va al árbitro y una línea "*Concesión de Bus*" procedente del mismo.
  - El árbitro otorga el uso del bus al dispositivo que considere oportuno en cada instante (sabe siempre quiénes lo quieren)



  - Ventajas:
    - Prioridades totalmente configurables
    - Asignación inmediata
  - Desventajas
    - Enorme complejidad
    - Dificultad para añadir dispositivos al sistema

Buses: características de diseño

- Petición independiente descentralizado
  - Cada dispositivo cuenta con tantas líneas "*Solicitud Bus*" y "*Concesión de Bus*" como dispositivos hay en el sistema
  - Cada dispositivo puede recibir "*Solicitud Bus*" de cada uno de los restantes y puede enviar "*Concesión de Bus*" al que desee ceder el control del bus



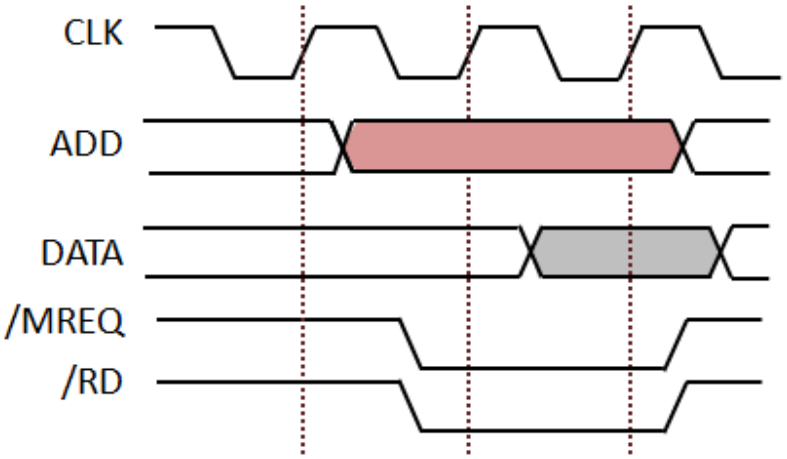
  - Ventajas
    - Prioridades totalmente configurables
    - Asignación inmediata
  - Desventajas
    - Enorme complejidad
    - Dificultad extrema para añadir dispositivos al sistema

Buses: características de diseño

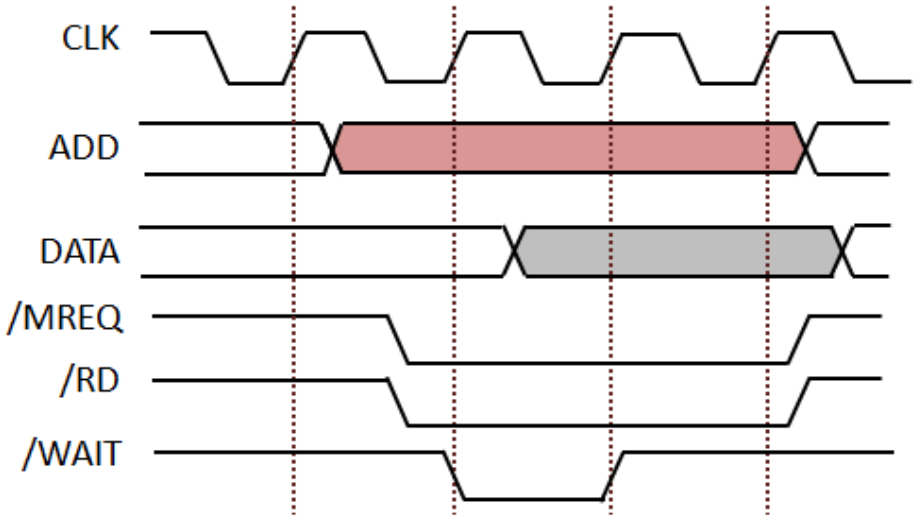
- Temporización:
  - Bus síncrono:
    - Un reloj maestro genera una onda cuadrada que marca las transiciones
    - La presencia de un evento en el bus está determinada por los flancos de reloj
    - Ventaja: fácil de gestionar
    - Desventaja: la transferencia debe completarse en ciclos enteros de reloj (aunque pueda realizarse en menos tiempo, ha de esperar a que termine el ciclo de reloj)
  - Además del bus de datos y direcciones, requiere señales de control:
    - /MREQ : petición de memoria
    - /RD : señal de lectura

Buses: características de diseño

- Temporización



BUS SÍNCRONO



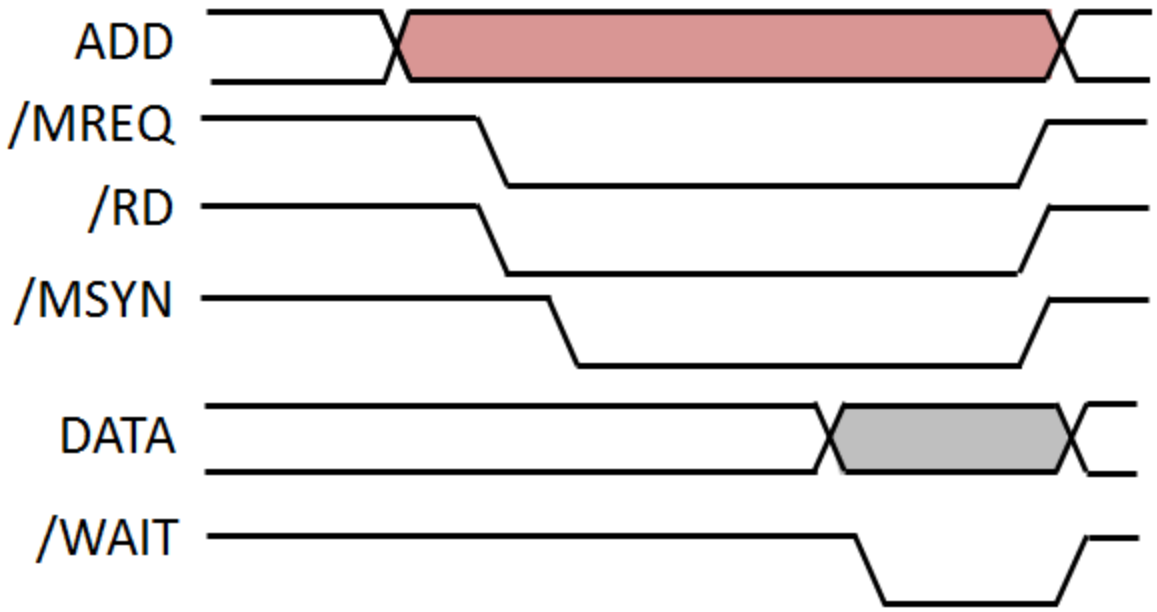
BUS SEMISÍNCRONO

Buses: características de diseño

- Temporización:
  - Bus asíncrono:
    - Los dispositivos irán a la velocidad que puedan.
    - Además de las señales de los buses síncronos, necesitarán:
      - /MSYN: Master SYNchronitation
      - /SSYN: Slave SYNchronitation
  - Bus semisíncrono:
    - Funciona de manera síncrona con la señal de reloj pero permite la introducción de ciclos de espera para atender a dispositivos más lentos.
    - /WAIT : señal de Espera (ciclo de espera) o READY

Buses: características de diseño

- Temporización

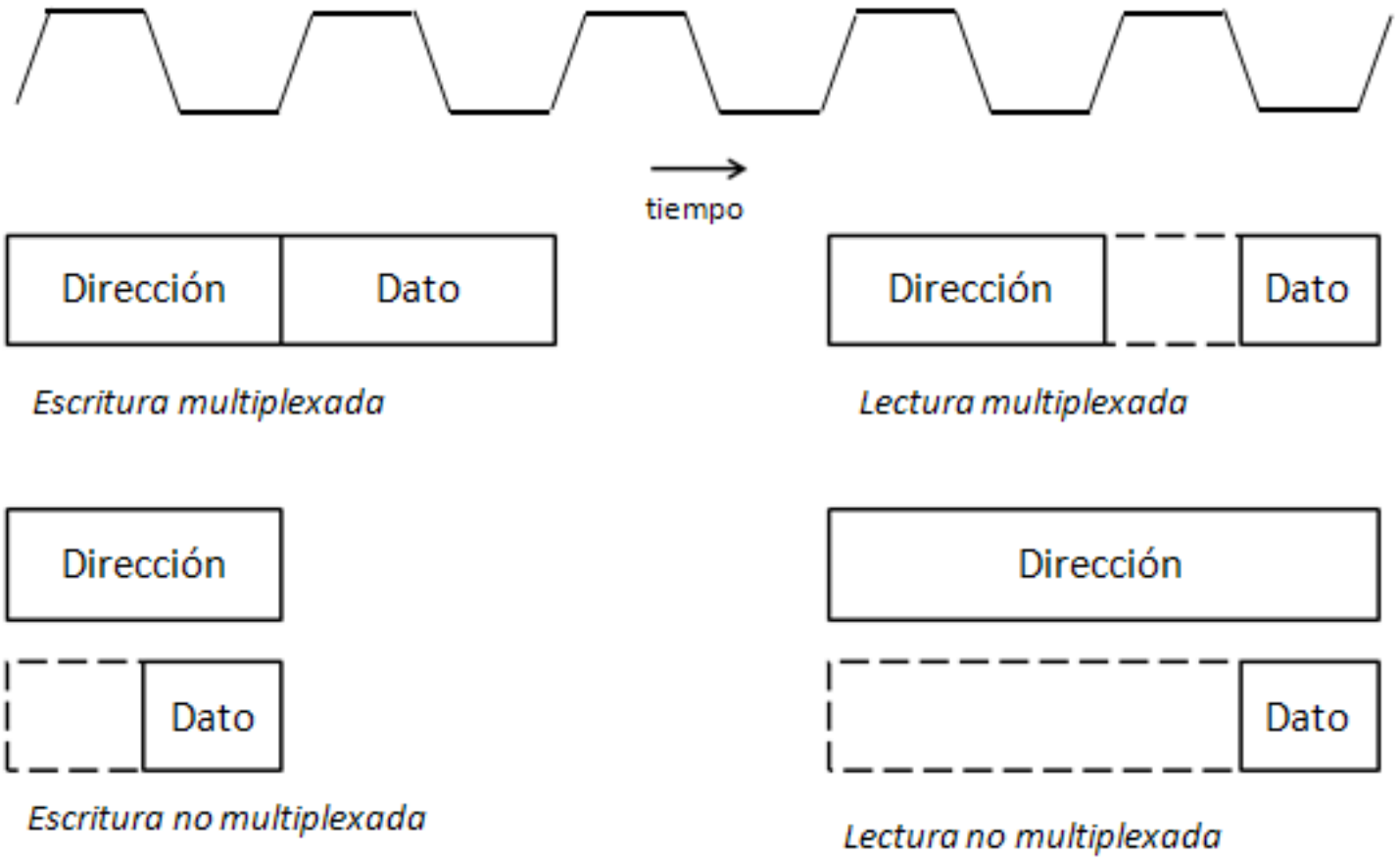


BUS ASÍNCRONO

Buses: características de diseño

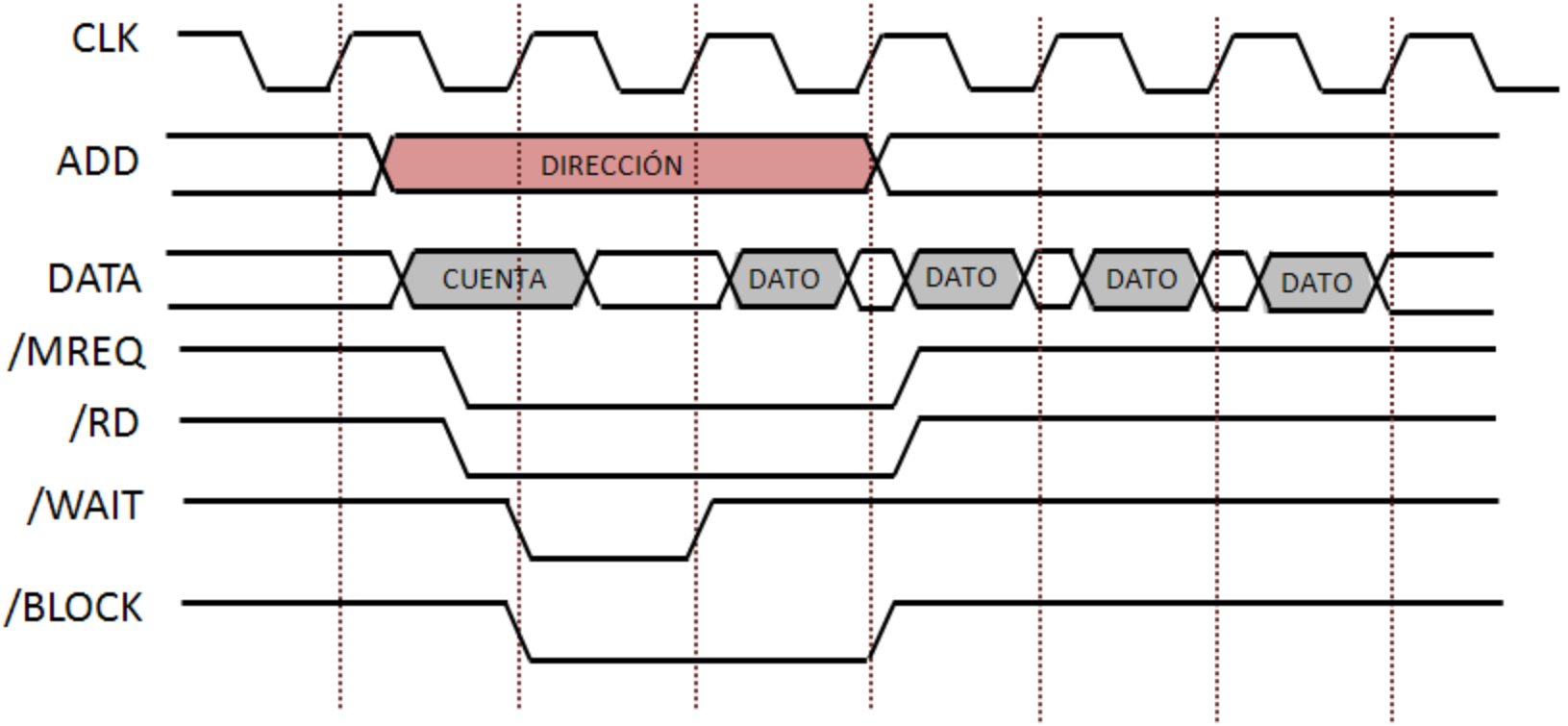
- Tipo de transferencia de datos
  - Lectura
  - Escritura
  - Lectura-modificación-escritura
  - Lectura después de escritura (escritura-lectura)
  - Transferencia de bloques

Buses: características de diseño



Buses: características de diseño

- Transferencia por bloques:
  - Para las arquitecturas que requieren transferencia de bloques



Buses

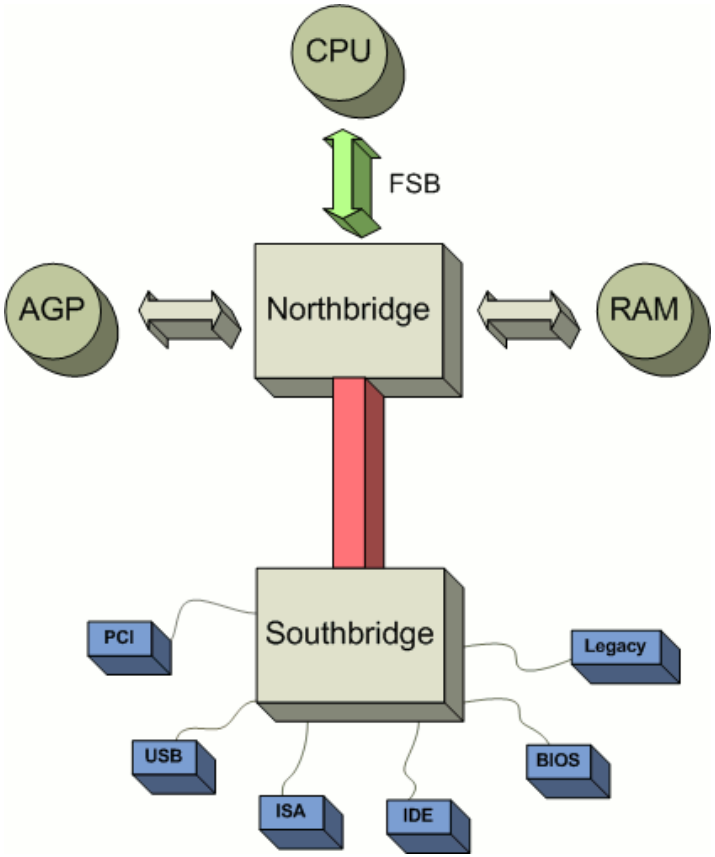
- Vimos en los dispositivos de E/S que tenían diferentes tasas de transferencia y que eran asíncronos. Para sincronizar analizábamos dos métodos: *strobing* y *handshaking*
- Podemos sincronizar con un reloj (diferente al del micro, mucho más rápido) → tendremos así una transferencia síncrona (recordad el ejemplo del controlador del teclado…)
- Al igual que en el caso asíncrono también aquí hay dos opciones:
  - Completar en un ciclo → síncrono
  - En los ciclos que haga falta (se añade una señal de tipo READY, WAIT o BUSY) → semisíncrono

Buses

- También vimos el controlador de E/S y las funciones de la interfaz (entre otras la conversión paralelo-serie y viceversa, en el caso de que se trate de un puerto serie)
- Se hacen necesarias normalizaciones (*standards*) para las señales de interfaz y protocolo.

Buses

- Los dispositivos que requieran una alta velocidad de conexión con el procesador (como la MP) se conectan directamente a este bus, pero solo pueden ser unos pocos (por razones eléctricas) → el resto a otro bus proporcionado por la placa base. Ambos buses interconectados por un circuito (bridge).
- Los dispositivos conectados al bus de expansión se muestran al micro como si estuvieran conectados a su bus (pero hay un cierto retardo)



Chipset schematic (Wikipedia, CC BY-SA 4.0)

Buses

- No es posible definir un estándar uniforme par el bus del procesador ya que su estructura depende de la arquitectura del procesador y de las características eléctricas del chip.
- El bus de expansión no está sujeto a estas limitaciones → puede utilizar un esquema estandarizado para sus señales

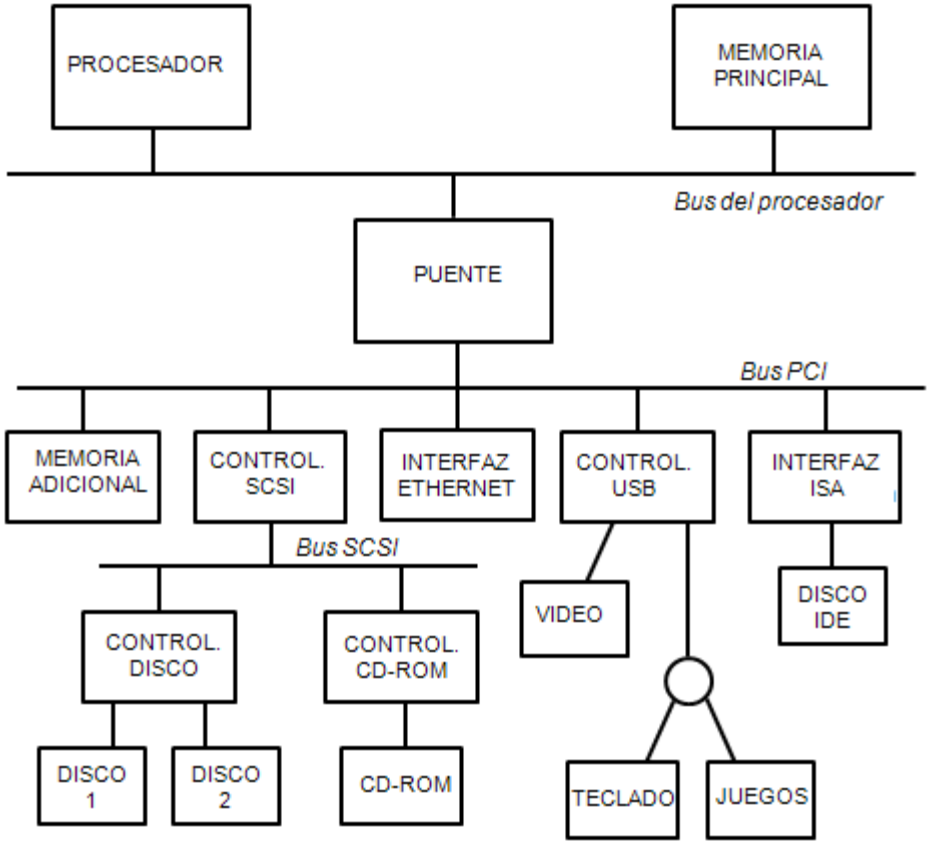
Buses

- Se han desarrollado diversos estándares
  - Algunos de forma natural → ISA (*Industry Stdandard Architecture*) (IBM)
  - Esfuerzos industriales cooperativos → USB (*Universal Serial Bus*)
  - Organizaciones como IEEE (*Institute of Electrical & Electronics Engineers*), ANSI (*American National Stdandards Organization*) o ISO (*International Standards Organization*) han favorecido esos estándares (de facto o cooperativos) dándoles una homologación oficial.

Buses

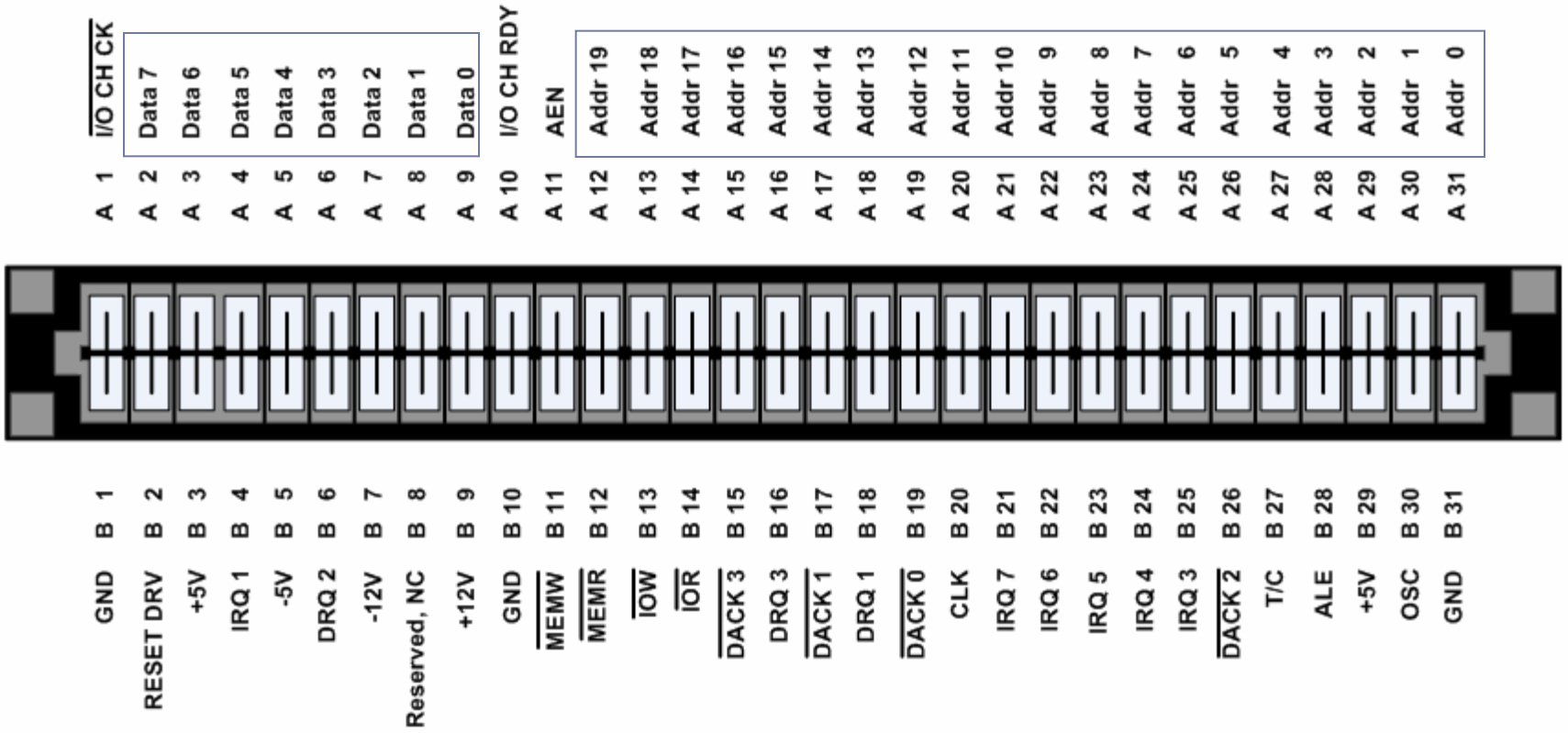
- Analizaremos 4 ampliamente utilizados:
  - ISA (*Industry Stdandard Architecture*)
  - PCI (*Peripheral Component Interconnect*)
  - SCSI (*Small Computer System Interface*)
  - USB (*Universal Serial Bus*)

Buses



Ejemplo de computador con diferentes estándares de interfaces

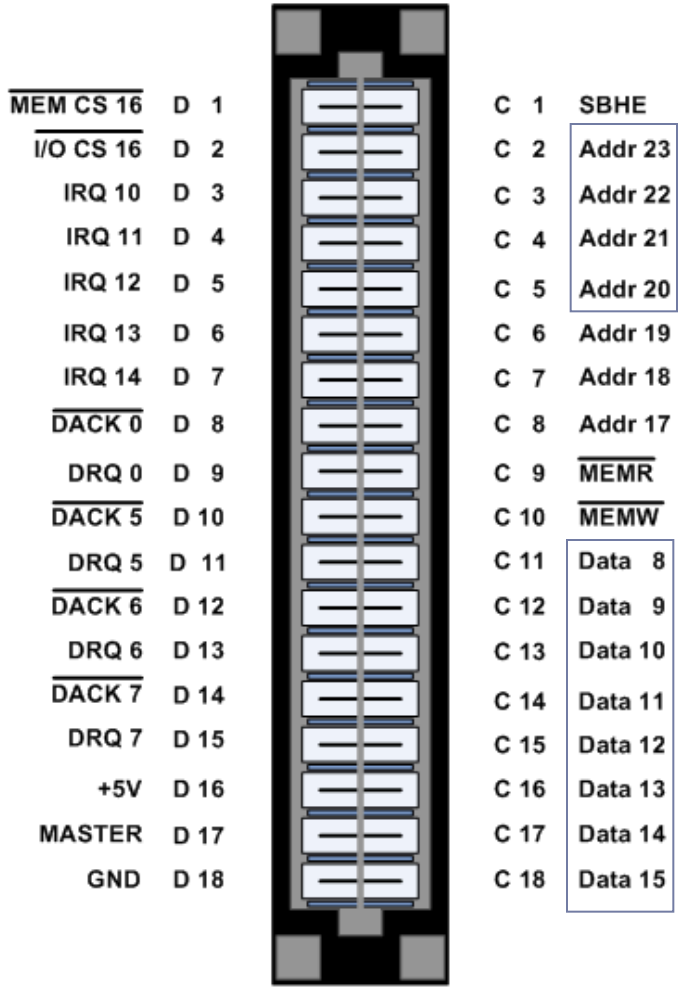
Buses: ISA



ISA Bus pins (Wikipedia, GFDL)

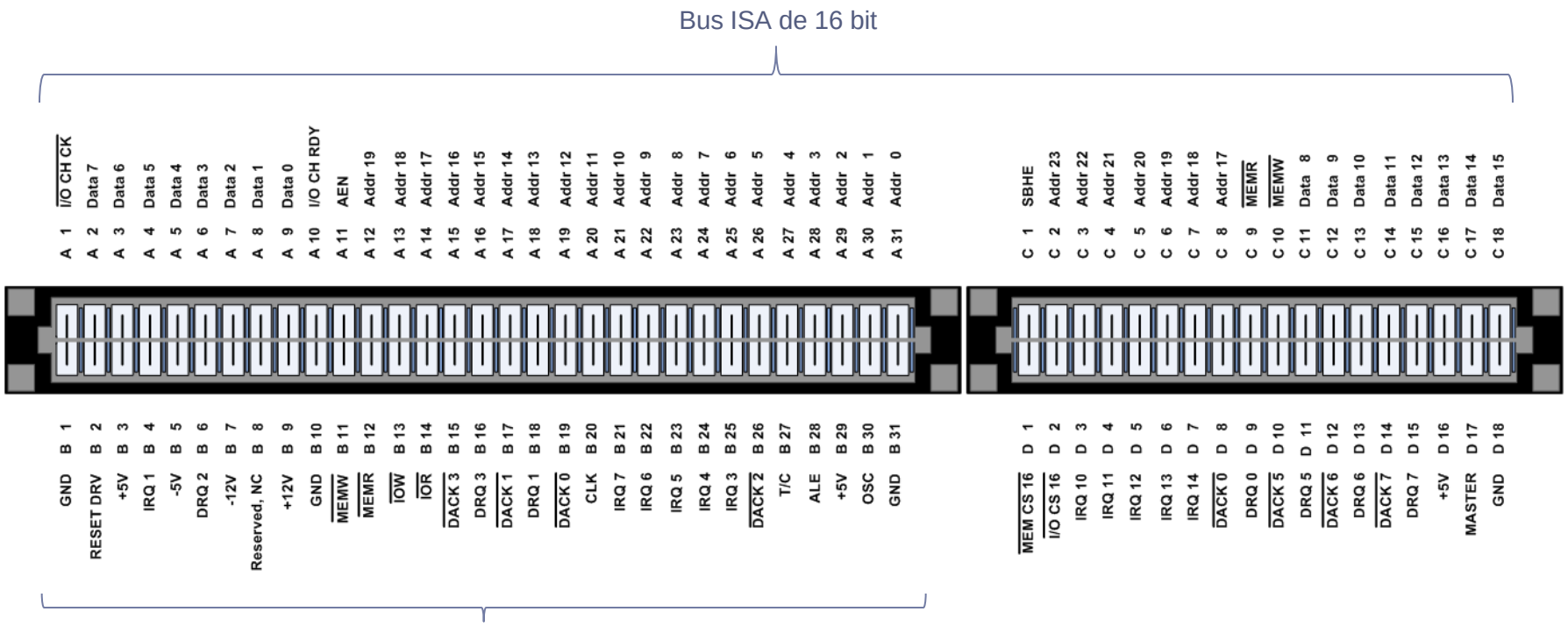
Buses: ISA

- Estas señales irán a través de la placa base.
- La introducción del 80286 en el mercado presenta el siguiente problema:
  - Tiene 16 bits de datos.
  - Si se crea un nuevo tipo de bus, las tarjetas existentes no servirían.
  - Se decide ampliar el bus, manteniendo las conexiones anteriores, incorporando un nuevo conector (1984).



ISA Bus pins (Wikipedia, GFDL)

Buses: ISA

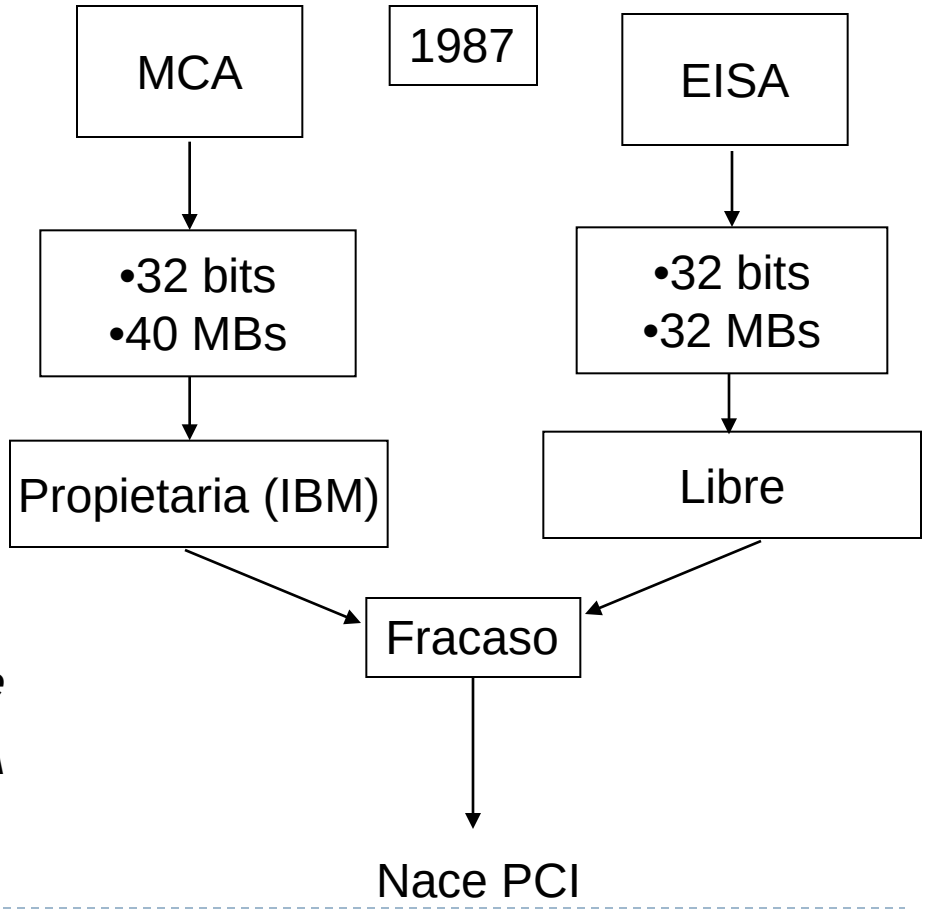


ISA Bus pins (Wikipedia, GFDL)

Buses: ISA

Inconvenientes:

- Velocidad insuficiente para aplicaciones modernas:
  - ISA 8 bits (a 4 MHz) → 4 MBs
  - ISA 16 bits (a 8 MHz)→ 16MBs
- No es plug & play



MCA: Micro Channel Architecture

EISA: Extended ISA

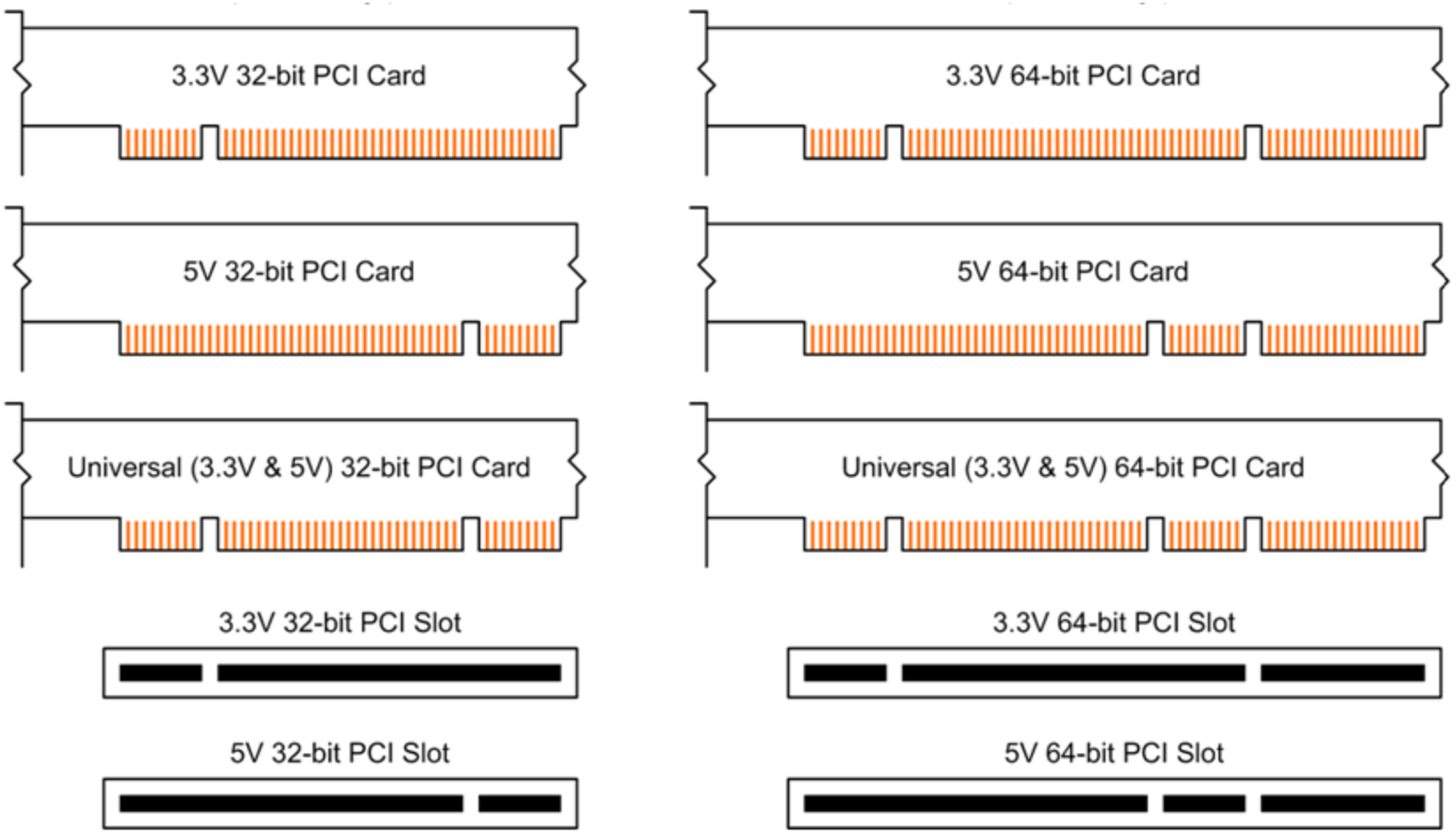
Buses: PCI

- Surge a raíz de mayores necesidades:
  - Pantallas en continuo movimiento: 67.5 MB/s
  - Vídeo: 135 MB/s
- 1992 primera aparición del bus PCI (Intel):
  - Primer puerto de expansión físicamente incompatible con ISA que logró sustituirlo
  - Característica pionera: ***plug & play*** para conectar dispositivos de E/S (ISA configuración mediante switches)
  - La estructura limita a 21 el número de dispositivos de E/S
  - La configuración sw asigna direcciones y prioridades
  - Es propietario pero de dominio público (un consorcio controla y desarrolla equipos basados en el bus PCI)

Buses: PCI

- Limitaciones:
  - No es lo suficientemente bueno para ser bus de memoria.
  - No es compatible con las viejas tarjetas para ISA que todavía existen
- Como ya vimos la solución es:
  - Usar tres o más buses
  - Elementos clave: puentes (los fábrica Intel…)
- Características:
  - Voltaje: 3.3V o 5V.
  - Capacidad: 32 bits o 64 bits.
  - Temporización: 33MHz o 66MHz
  - PCI 3.0: última especificación (estándar final oficial, sin soporte de 5v)

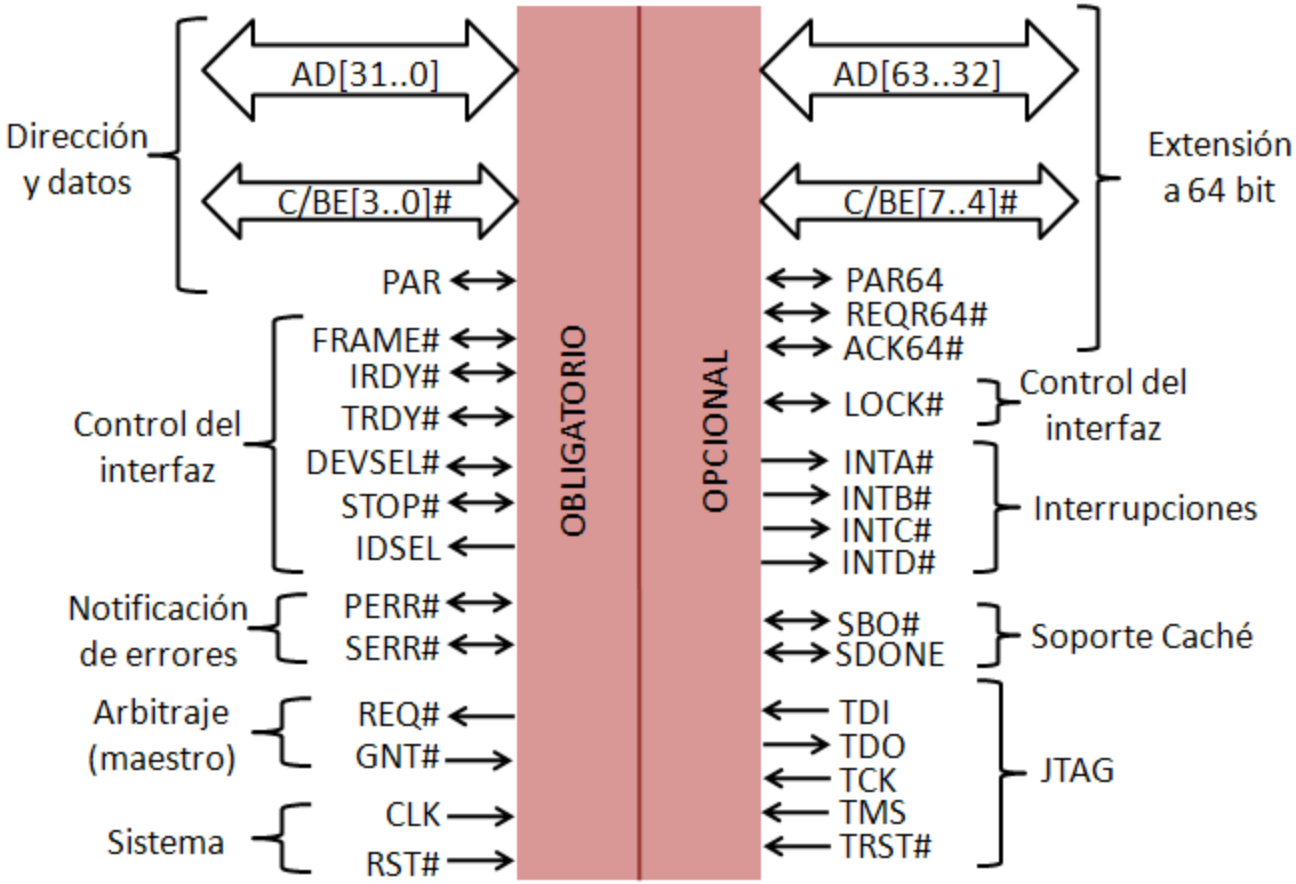
Buses: PCI



PCI Keying (Wikipedia, GFDL)

Buses: PCI

- Esquema del bus PCI



Buses: PCI

- Principales grupos de señales:
  - Bus multiplexado direcciones/datos: AD[31..0]
    - Líneas dirección AD[31..0] dirección de los accesos a memoria o a entrada/salida.
    - Líneas dedicadas a datos AD[31..0]: durante fase de datos
  - Bus comando/*byte enable*: C/BE[3..0]#
    - Comando durante dirección
    - Habilitación de *bytes* durante datos
  - Línea de control de paridad
    - Paridad par de AD[31..0] y C/BE[3..0]#
  - Señales de control
    - FRAME#: Duración de la transacción (*Initiator*)
    - IRDY#: Dispositivo *Initiator Ready*
    - TRDY#: Dispositivo *Target Ready*
    - DEVSEL#: Dispositivo seleccionado (respuesta de *Target*)
    - STOP#: Interrupción de la transacción (señal del *Target*)
    - IDSEL: Selección de dispositivo para inicialización (configuración)

 # indica que son activas a nivel bajo

Buses: PCI

- Señalización de errores:
  - PERR#: Indica errores de paridad
  - SERR#: Indica errores graves del sistema
- Arbitración (sólo dispositivos *master*)
  - REQ#: Solicitud del bus
  - GNT#: Concesión del bus
- Sistema:
  - CLK: Reloj (de 33 o 66 MHz)
  - RST#: Señal de *reset* (inicialización)
- Interrupción (hasta 4 interrupciones Hw por parte del dispositivo PCI):
  - INTA# .. INTD#

Buses: PCI

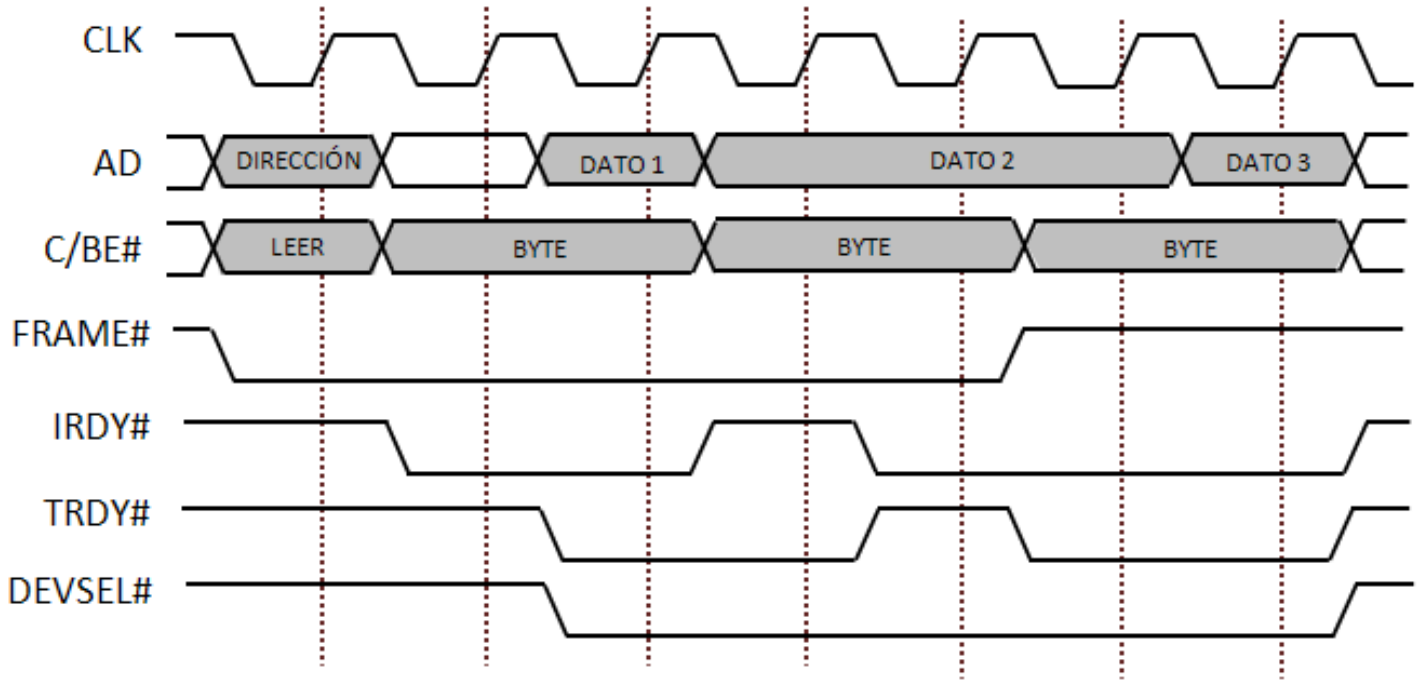
- Comandos

C/BE#	Comando
0000	Reconocimiento de interrupción
0001	Ciclo especial
0010	Lectura E/S
0011	Escritura E/S
0100	Reservado
0101	Reservado
0110	Lectura memoria
0111	Escritura memoria
1000	Reservado
1001	Reservado
1010	Configuración lectura
1011	Configuración escritura
1100	Lectura memoria múltiple
1101	Ciclo de dirección dual
1110	Línea lectura memoria
1111	Escritura de memoria e invalidación

Buses: PCI

- Inicialmente diseñado para transferencia por ráfagas (la E/S de una única palabra es una ráfaga de longitud igual a uno)
- "Transacción": operación de transferencia completa (una dirección y una ráfaga de datos)
- "Fases": transferencias individuales de las palabras de una transacción

Buses: PCI

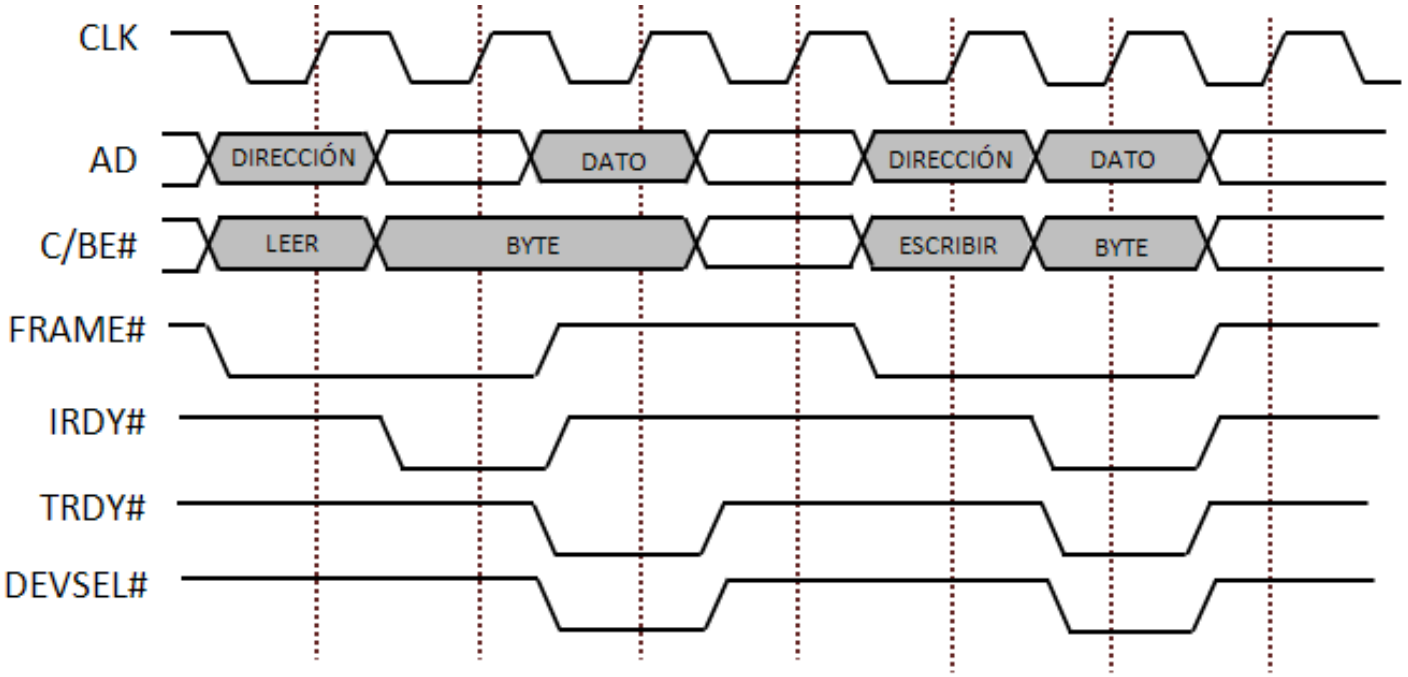


Ciclo de lectura de tres datos con dos ciclos de espera.

Buses: PCI

- El iniciador solicita unos datos (lectura)
- Indica su disponibilidad (IRDY#)
- El objetivo reconoce que ha sido seleccionado (DEVSEL#)
- Pone el dato e indica que está preparado (TRDY#)
- Los datos se leen en los flancos de reloj (señalados)
- Si alguno de los participantes en la comunicación deja de estar disponible (no puede poner o recoger el dato del bus), desactiva su señal (IRDY# o TRDY#), y la lectura del dato se posterga.
- La señal FRAME# se desactiva una vez leído el penúltimo dato (cuando sólo falta uno).

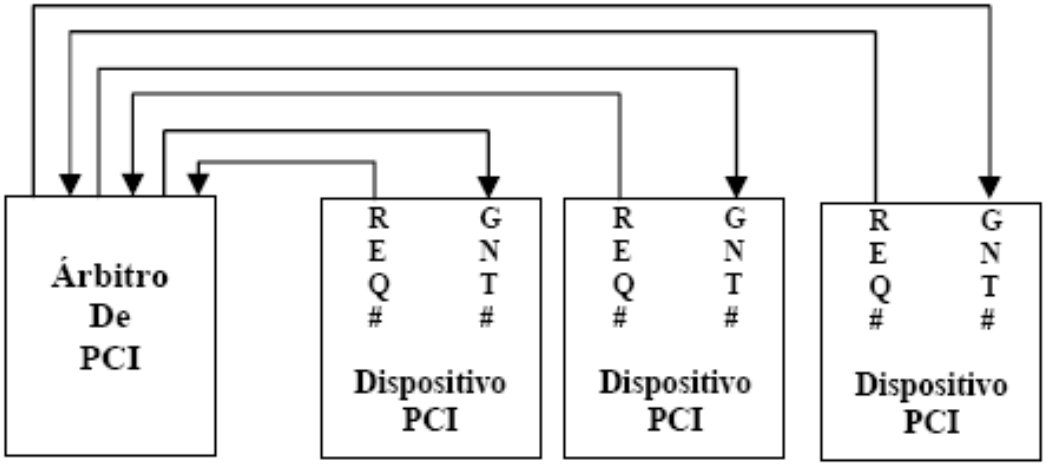
Buses: PCI



Ciclo de lectura seguido de ciclo de escritura (sin ciclos de espera).

Buses: PCI

- Arbitraje del bus (Bus Mastering)
  - Para solicitar el bus:
    - El dispositivo activa #REQ.
    - Espera hasta que el árbitro habilita su #GNT.
    - El dispositivo puede utilizar el bus en el siguiente ciclo.

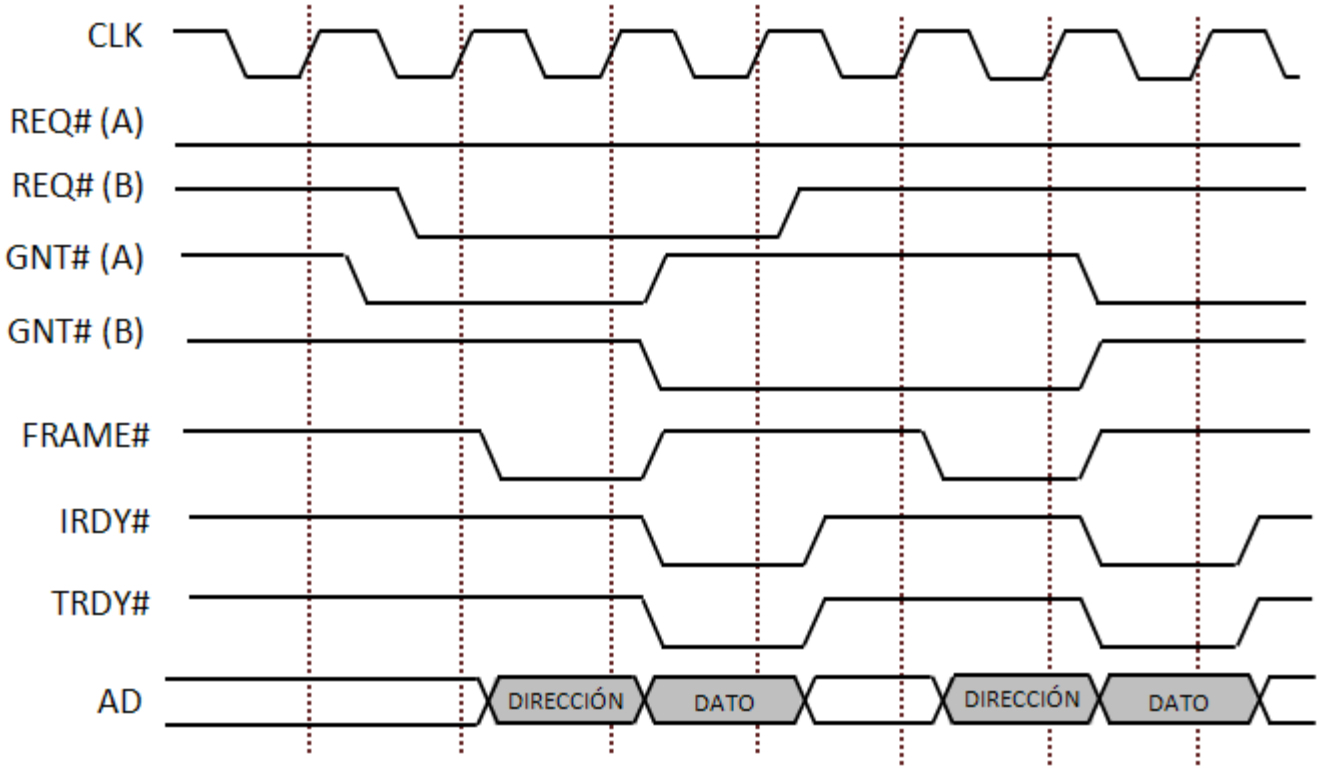


Buses: PCI

- Una concesión de bus es válida para una transacción.
- Si un dispositivo quiere ejecutar una segunda transacción y ningún otro dispositivo está solicitando el bus, puede continuar dejando un ciclo ocioso entre transacciones.
- En circunstancias especiales, sin competencia por el bus, un dispositivo puede efectuar transacciones una tras otra sin tener que insertar ningún ciclo ocioso.
  - Para parar a este dispositivo, el árbitro deshabilita #GNT.

Buses: PCI

Arbitraje bus PCI (centralizado)



Buses: PCI

- El dispositivo A solicita el bus.
- Se le concede (para una transacción). Como no ha terminado no desactiva la solicitud.
- En el siguiente ciclo de reloj tanto A como B están solicitando el bus → se le concede a B (para una transacción). B no tiene más datos → desactiva la solicitud.
- En el siguiente ciclo de reloj de nuevo solo A solicita el bus → le es concedido (para una transacción, si no termina y nadie más lo solicita se le seguirá concediendo intercalando ciclos ociosos).

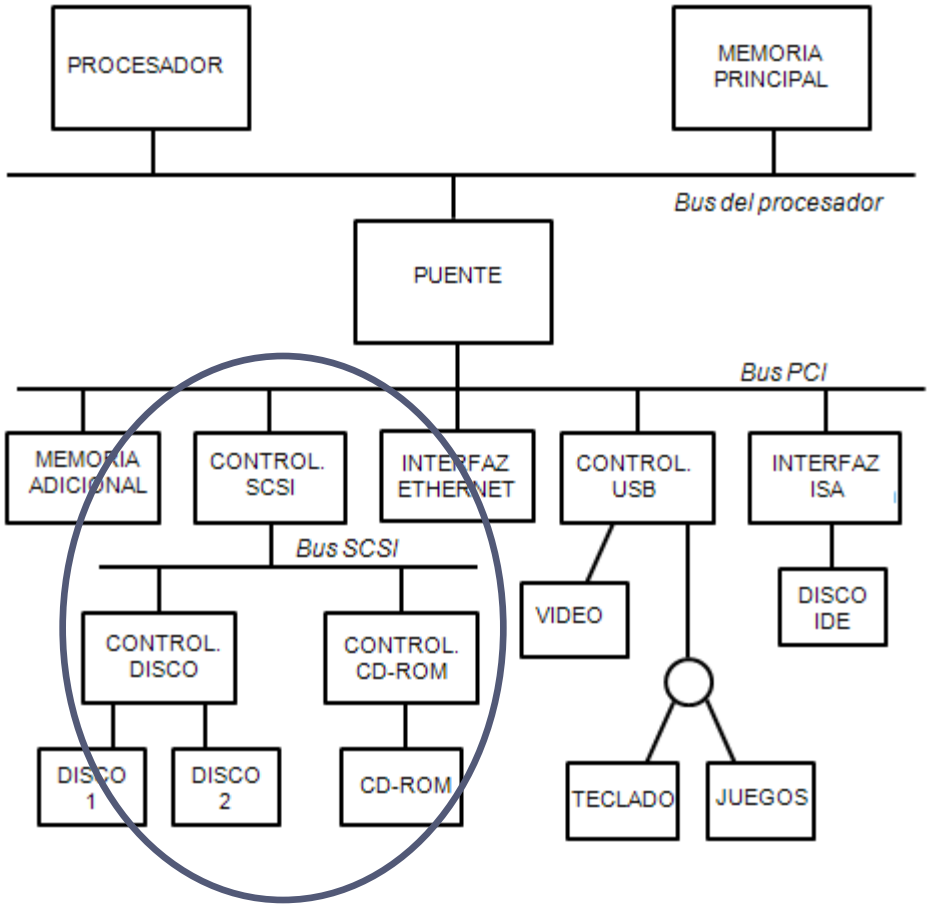
Buses: SCSI

- Características iniciales
  - Std definido por ANSI bajo la denominación X3.131
  - 50 hilos
  - Hasta 50 metros
  - Velocidad de 5MB/s
  - Habilidad para solapar peticiones → altas prestaciones

Buses: SCSI

- Muchas revisiones (su capacidad de transferencia de datos ha crecido rápidamente) por lo que podemos encontrar:
  - Bus de datos:
    - Estrecho: 8 bits
    - Ancho: 16 bits
  - Esquema de señales:
    - Single Ended: cada señal un cable + tierra común
    - Diferenciación de señales: un cable de retorno para cada señal (en este caso son posibles dos niveles de tensión 5v o 3,3v)
  - Conectores: 50, 68, 80 pines
  - Velocidad de transferencia: es función de la longitud y del número de dispositivos conectados (5MB/s – 640 MB/s)

Buses: SCSI



- Se conecta mediante un controlador SCSI que utiliza DMA para transferir paquetes (datos, órdenes o estado) entre la MP y los dispositivos
- Los dispositivos conectados al SCSI no son parte del espacio de direcciones del procesador

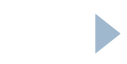
Buses: SCSI

- Señales del bus SCSI

Señal	
DB[7..0] DB(P)	Líneas de datos (información durante la fase de transferencia, identificación del byte durante la fase de arbitraje, selección y reselección). Bit de paridad para los datos.
BSY SEL	Ocupado. Indica que el bus está siendo usado. Selección. Indica que un dispositivo está intentando seleccionar (ini. al obj.) o reseleccionar (obj. al ini.) otro dispositivo.
C/D MSG	Control/Dato. Indica si los datos en el bus son de control (1) o de info.(0). Mensaje. Indica que un dispositivo SCSI tiene un mensaje para otro.
REQ ACK	El objetivo la activa para solicitar una negociación para transferencia de info. El indicador la activa para confirmar una negociación para transferencia.
I/O	Activa indica que la operación es de entrada (al iniciador).
ATN RST	Atención: la activa el iniciador cuando desea enviar un mensaje al objetivo. Reset: hace que todos los dispositivos se desconecten volviendo a su situación inicial.

Buses: SCSI

- Fases principales del funcionamiento del SCSI:
  - Arbitraje
    - Arbitraje distribuido de prioridad fija (el controlador 7 es el de mayor prioridad)
  - Selección
    - El controlador que haya ganado el arbitraje del bus (iniciador) activa /SEL y la línea del dispositivo seleccionado
  - Transferencia de información
    - Se transfieren órdenes, respuestas o datos; se utilizan señales de diálogo de conformidad para controlar la transferencia (/REQ, /ACK)
  - Reselección
    - Cuando, tras la suspensión de una conexión lógica, el objetivo está preparado para restaurarla (debe primero ganar el control del bus). Este mecanismo permite establecer una conexión local que puede ser suspendida y restablecida y que les permite intercambiar paquetes.



Buses: USB

- En busca de un mecanismo simple y de bajo coste para conectar una amplia variedad de dispositivos, un consorcio de empresas (de comunicación y computadores) liderado por Intel (Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, Nortel Networks, Philips,…) desarrollan el estándar industrial **USB**

USB Icon (Wikipedia, dominio público)

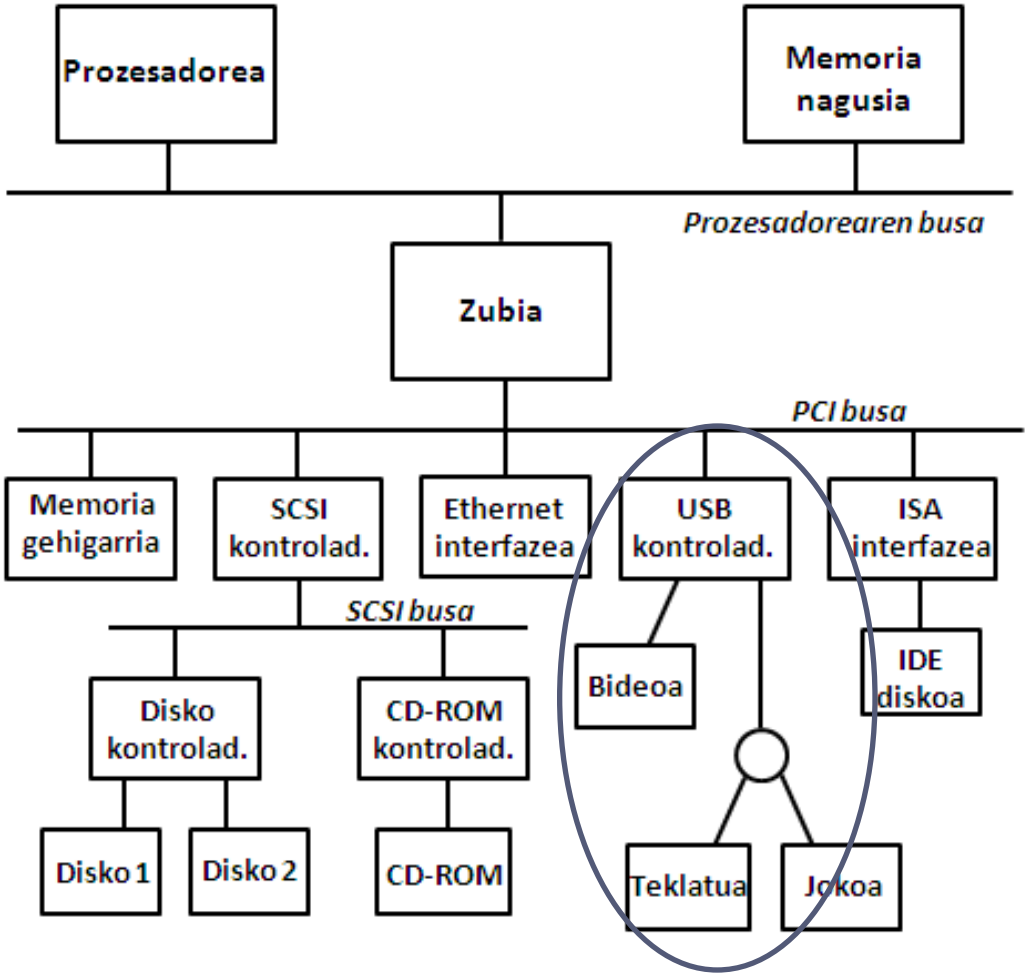
Buses: USB

- USB
  - Ideas iniciales (simple y de bajo coste):
    - El Bus y sus dispositivos de E/S deberán ser de bajo coste
    - El usuario no tendrá que actuar sobre microinterruptores en el dispositivo (fácil de utilizar)
    - El usuario no tendrá que abrir el computador para incorporar nuevos dispositivos (fácil de utilizar)
    - Solo existirá un único tipo de cable de interconexión
    - Los dispositivos de E/S serán alimentados desde el cable del bus (podrán conectarse hasta 127 dispositivos por computador)
    - Deberá ajustarse a un amplio rango de dispositivos de E/S, incluyendo dispositivos que trabajen en tiempo real
    - Los dispositivos podrán conectarse estando en funcionamiento el computador, sin necesidad de reiniciarlo (plug & play)

Buses: USB

- Características
  - Es un bus serie (paralelo caro y limita la distancia (*skew*))
  - Velocidad (tráfico dividido)
    - USB 1.0, 1.1: 1,5 Mb/s (low-speed)
    - USB 1.0, 1.1 : 12 Mb/s (full-speed)
    - USB 2.0: 480 Mb/s (high-speed)
    - USB 3.0: 5Gb/s (super-speed)
  - Inicialización:
    - Cable principal donde se conectan los dispositivos
    - Al conectar se interrumpe al S.O.
    - En un primer ciclo se detecta el tipo de dispositivo y el ancho de banda permitido
    - Si el ancho de banda es soportado por el S.O. le asigna una dirección de 7 bits (1-127, inicialmente 0)
    - El reloj se transmite implícito en los datos

Buses: USB

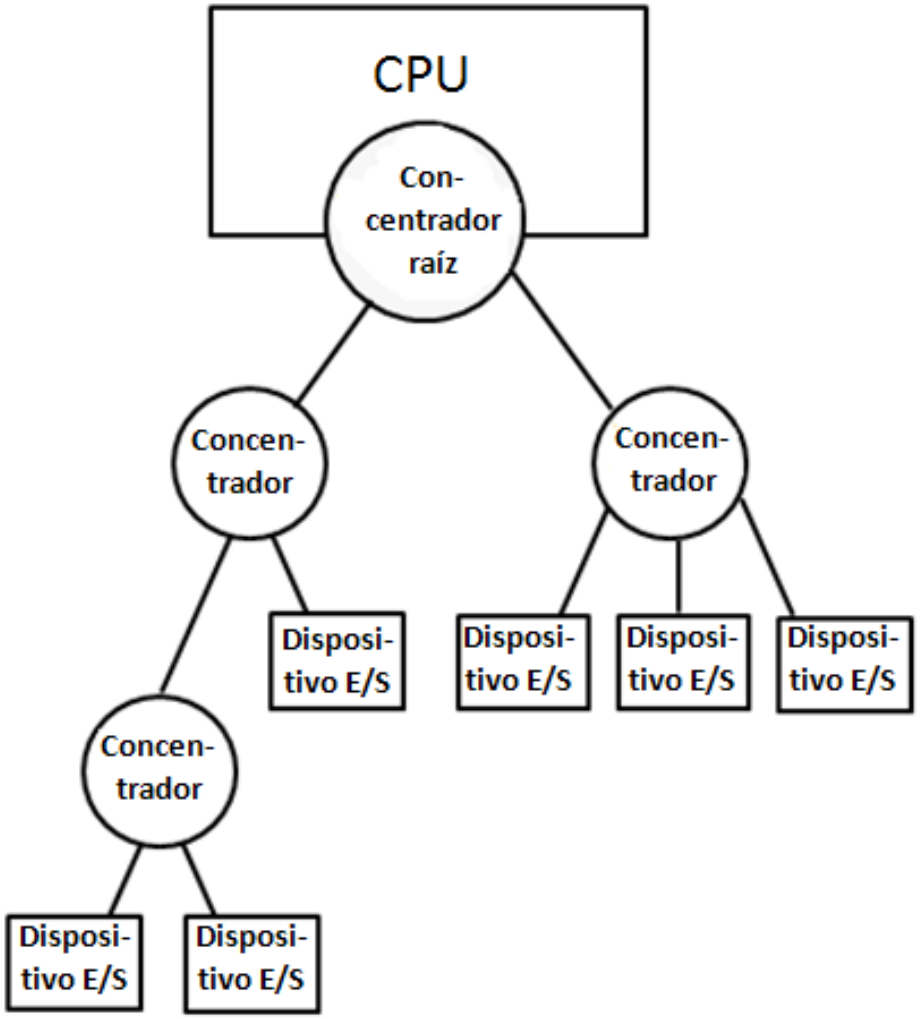

Prozesadorea
Memoria nagusia
Prozesadorearen busa
Zubia
PCI busa
Memoria gehigarria
SCSI kontrolad.
Ethernet interfazea
USB kontrolad.
ISA interfazea
SCSI busa
Bideoa
IDE diskoa
Disko kontrolad.
CD-ROM kontrolad.
Disko 1
Disko 2
CD-ROM
Teklatua
Jokoa

Buses: USB

- Topología
  - Una estructura en forma de árbol permite añadir o quitar dispositivos (ha de mantener una imagen de la topología en cada momento)
  - Cada nodo tiene un concentrador (*hub*) que actúa como punto intermedio entre el computador y los dispositivos de E/S
  - Cada concentrador tiene un número de puertos en los que se pueden conectar los dispositivos (también otros concentradores)
  - Los dispositivos de E/S (funciones en terminología USB) serían las hojas
  - Un concentrador raíz conecta el árbol entero al computador

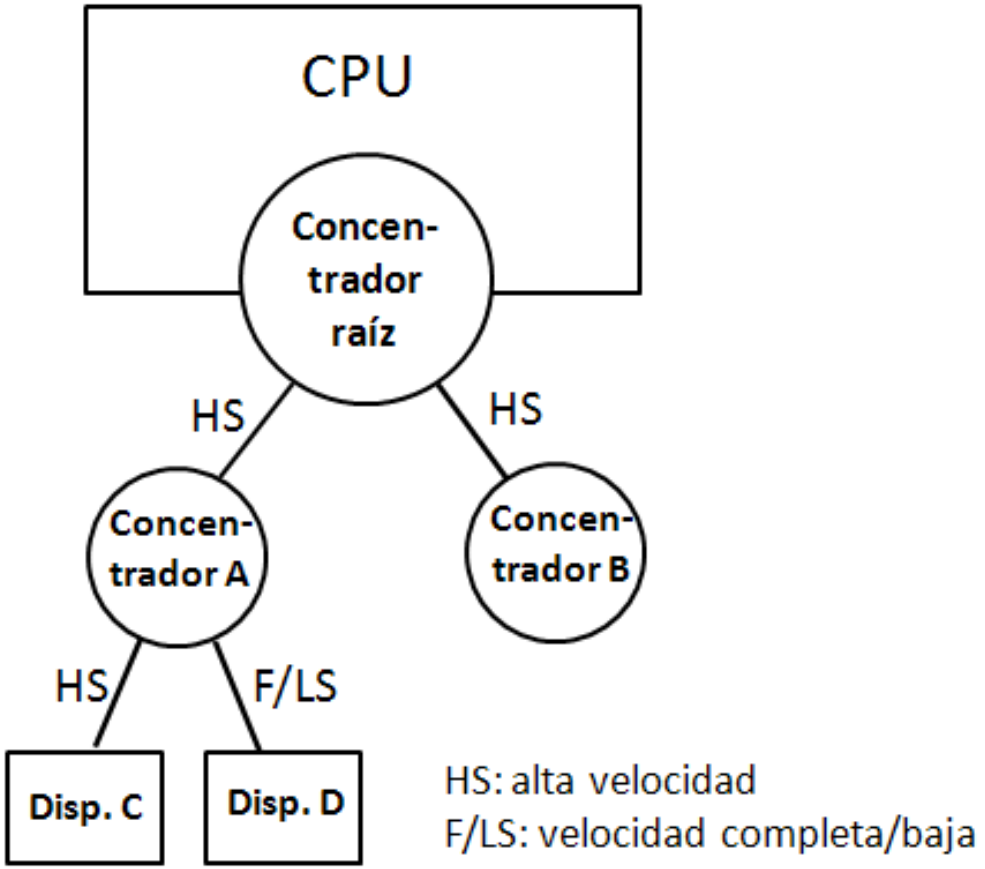
Buses: USB

Topología USB



Buses: USB

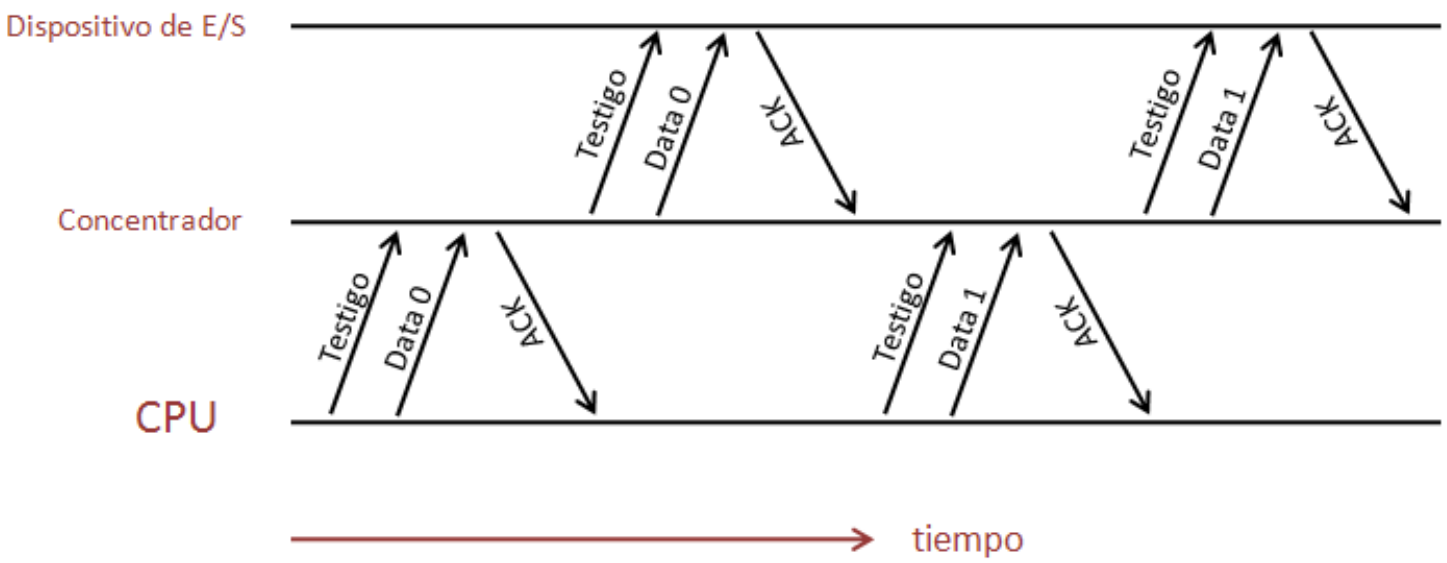
Tráfico compartido



Buses: USB

- La estructura de árbol permite conectar gran número de dispositivos a través de pocos puertos
- Cada dispositivo se conecta mediante una conexión serie punto a punto (facilita plug & play)
- El concentrador copia los mensajes que recibe del computador y los transmite a todos los dispositivos de E/S; sólo el direccionado responde
- Un mensaje de un dispositivo de E/S se envía sólo hacia la raíz, no lo ven los demás → no pueden comunicarse entre ellos
- El USB opera mediante sondeo
  - No hay conflictos entre dispositivos
  - Sencillez y bajo coste en el concentrador

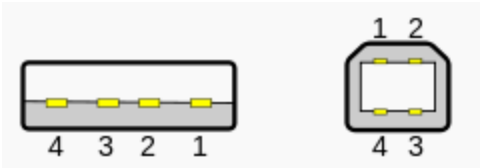
Buses: USB


Dispositivo de E/S
Concentrador
CPU
Testigo
Data 0
ACK
Testigo
Data 1
ACK
Testigo
Data 0
ACK
Testigo
Data 1
ACK
tiempo

Buses: USB

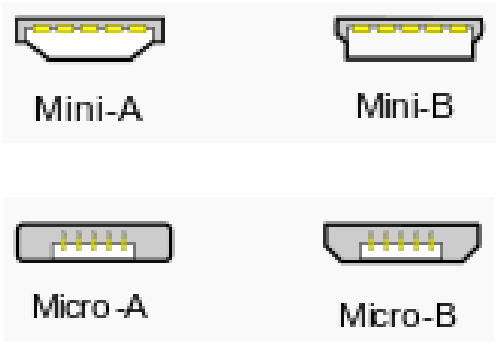
- Configuración Hardware:
  - 4 hilos

	Nombre	Color	Descripción
1	Vcc	Rojo	+ 5 V
2	D-	Blanco	Data -
3	D+	Verde	Data +
4	GND	Negro	Masa



  - Tipos mini y micro: 5 hilos

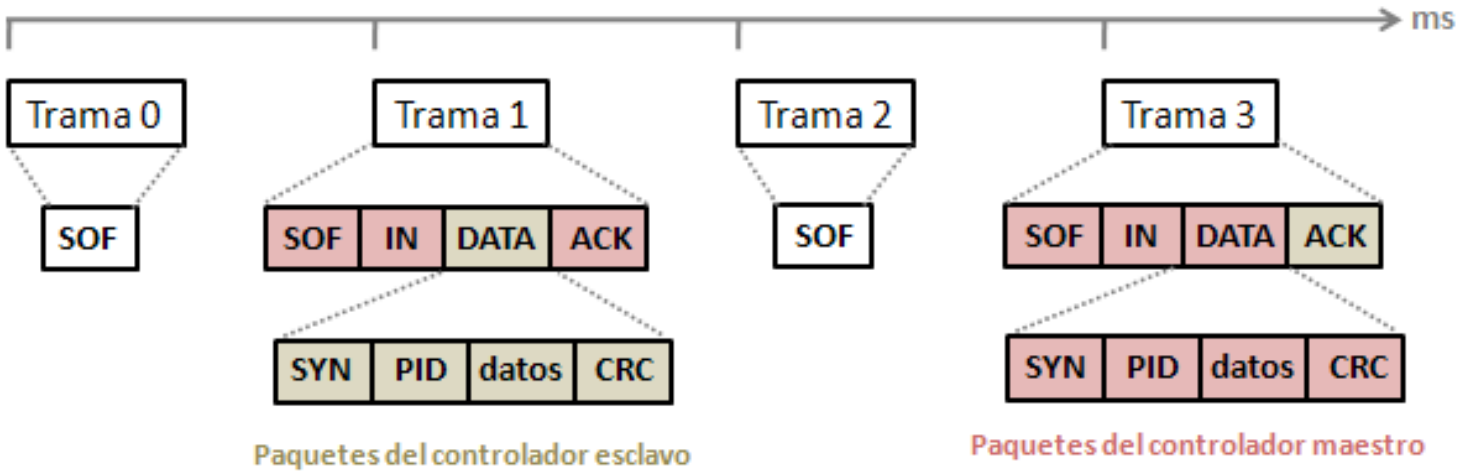
	Nombre	Color	Descripción
1	Vcc	Rojo	+ 5 V
2	D-	Blanco	Data -
3	D+	Verde	Data +
4	ID	---	Distingue entre micro A y B
5	GND	Negro	Masa



Types-usb new (Wikipedia, CC BY-SA 3.0)

Buses: USB

- Funcionamiento:
  - Cada milisegundo se transmite una trama de sincronización de dispositivos:
    - Las transacciones se denominan tramas.
    - Las tramas están formadas por paquetes.
    - Las tramas comienzan por SOF: Start Of Frame



Buses: USB

- SYN (sincronización), PID (identificador del paquete), CRC (código de redundancia cíclica).

- Los paquetes se dividen en cuatro tipos:
  - Testigo: son de control y viajan del controlador al dispositivo. Estos son SOF, IN (pide datos), OUT (indica que se envían datos) y SETUP (configuración).
  - Datos: paquetes DATA que transmiten hasta 64 bytes de información en ambos sentidos.
  - Saludo: ACK (paquete anterior recibido correctamente), NAK (error en paquete – CRC), STALL (ocupado) y NYET.
  - Especiales: PRE, ERR, SPLIT, PING, RESERVED.

Buses: USB

- Cuatro tipos de transferencias
  - Interrupciones: la emplean los dispositivos más lentos, que envían información con poca frecuencia (ratones, teclados…)
  - Por bloques: se utiliza con dispositivos que mueven grandes paquetes de información en cada transferencia (impresoras…)
  - Isócrona: flujo de datos constante y en tiempo real, sin aplicar detección ni corrección de errores (altavoces USB…)
  - Control: configurar dispositivos, emisión de órdenes y revisión del estado