ENTRADA-SALIDA (I/O) Y COMUNICACIÓN

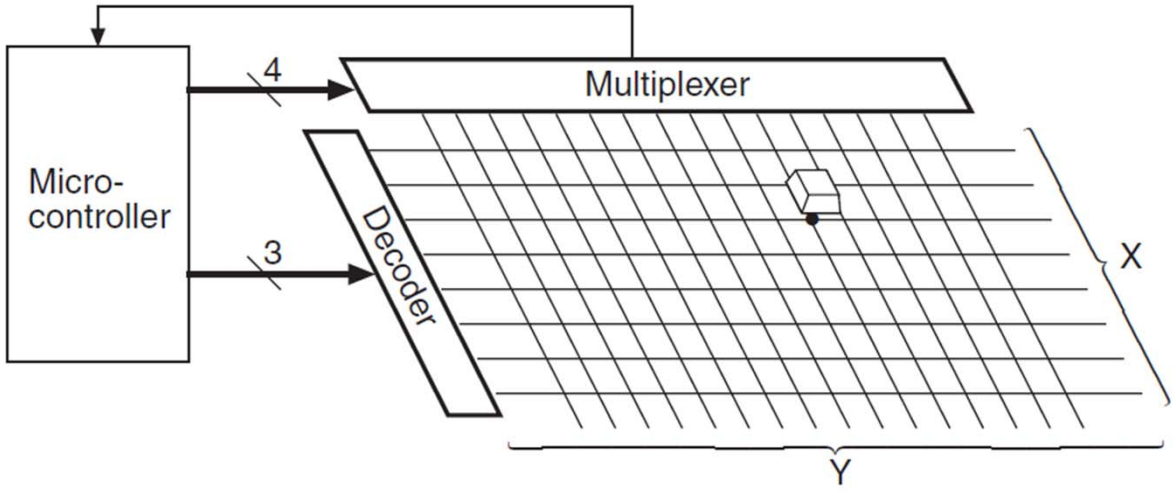
- Son necesarios instrumentos que permitan la entrada en memoria de programas y/o datos y que almacenen y/o muestren los resultados
- Dispositivos de I/O más comunes: teclado, ratón, monitor, impresora, disco duro, CD-ROM, tarjeta de red.
- Otros dispositivos: scanner, altavoz, web-cam, micrófono, lector tarjetas flash, convertidores Analógico/digitales y Digital/analógicos, tarjeta de adquisición de datos, componentes de control etc….
- Cada dispositivo presenta una problemática diferente debido a la naturaleza de los datos que maneja, modo de transmisión y a la velocidad con que se comunica con la CPU.

Periféricos – son dispositivos conectados en línea (on line) es decir dispositivos directamente controlados por la CPU o que transfieren información con memoria a través de comandos de la CPU

Ejemplos periféricos:

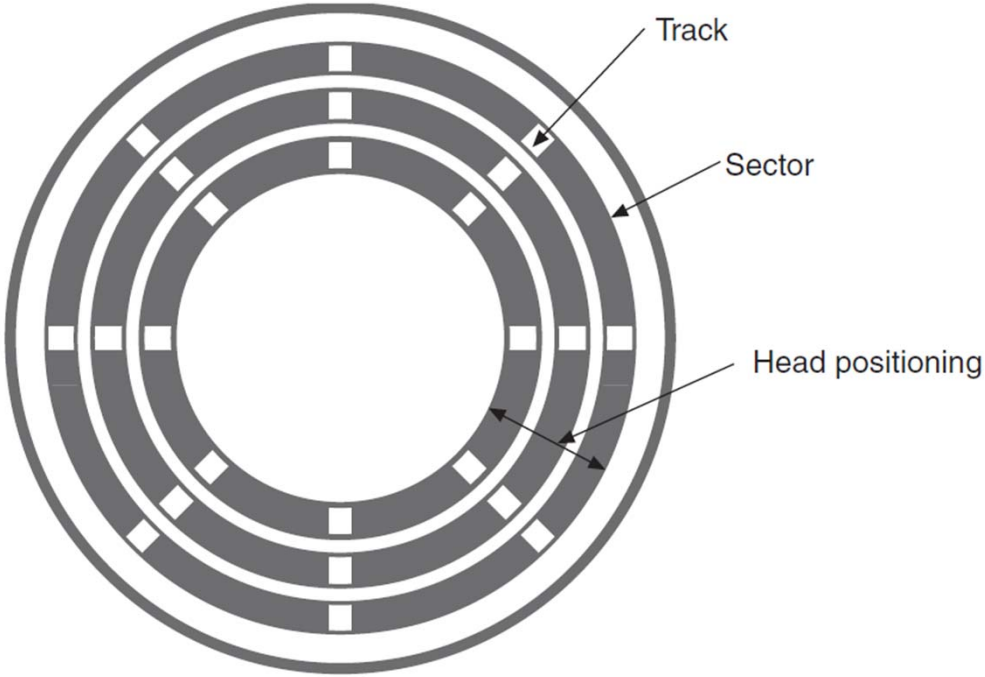
- Teclado – Dispositivo de entrada – baja velocidad
- Disco duro – Dispositivo de entrada/salida (write/read) – velocidad media
- Pantalla gráfica – Dispositivo de salida – velocidad alta

EJEMPLO DE PERIFÉRICOS - TECLADO



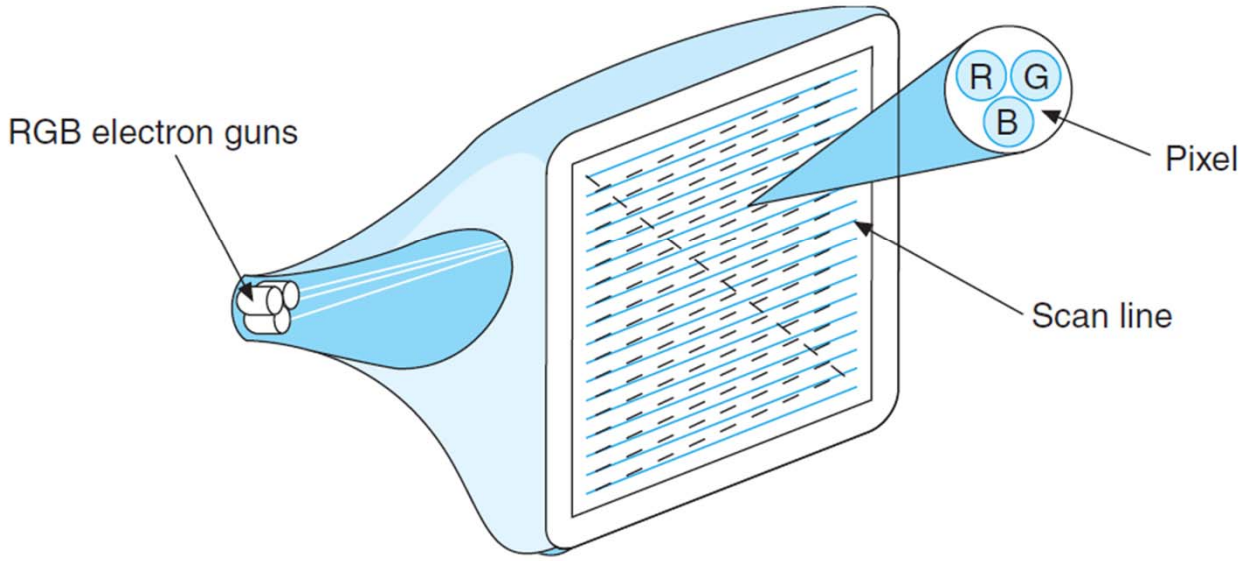
- Matriz: 8x16=128 intersecciones (teclas)
- Microcontrolador: con RAM, ROM, temporizador y una interfaz I/O
- K-scan code
- Velocidad de transmisión: 10 bytes/s

EJEMPLO DE PERIFÉRICOS – DISCO DURO (HDD)



- Almacenamiento de datos no volátil. Transmisión serie. Emplea un sistema de grabación magnética
- Compuesto de 2-4 platos o discos rígidos unidos por un mismo eje
- Dentro de cada plato dos caras => dos cabezales de lectura/escritura
- Pista/cilindro –sector 512 bytes – 4 KB
- Velocidad de transferencia – typ 20 MB/s (hasta 600 MB/s (SATA)) – velocidad de rotación 7200 rpm (129Km/s periferia 3,5’’)

EJEMPLO DE PERIFÉRICOS – PANTALLA GRÁFICA



- Tecnologías LCD (liquid crystal display) CRT (cathode-ray tube), OLED (organic light emitter diode)
- Pixel – 3 colores (RGB) – 1 byte/color => 2^{24} = 16777216 colores
- Barrido horizontal y vertical => 60 Hz
- Resolución – 1920x1200 pixeles
- Velocidad de transferencia de información – 518400000 bytes/s = 494.4 MB/s

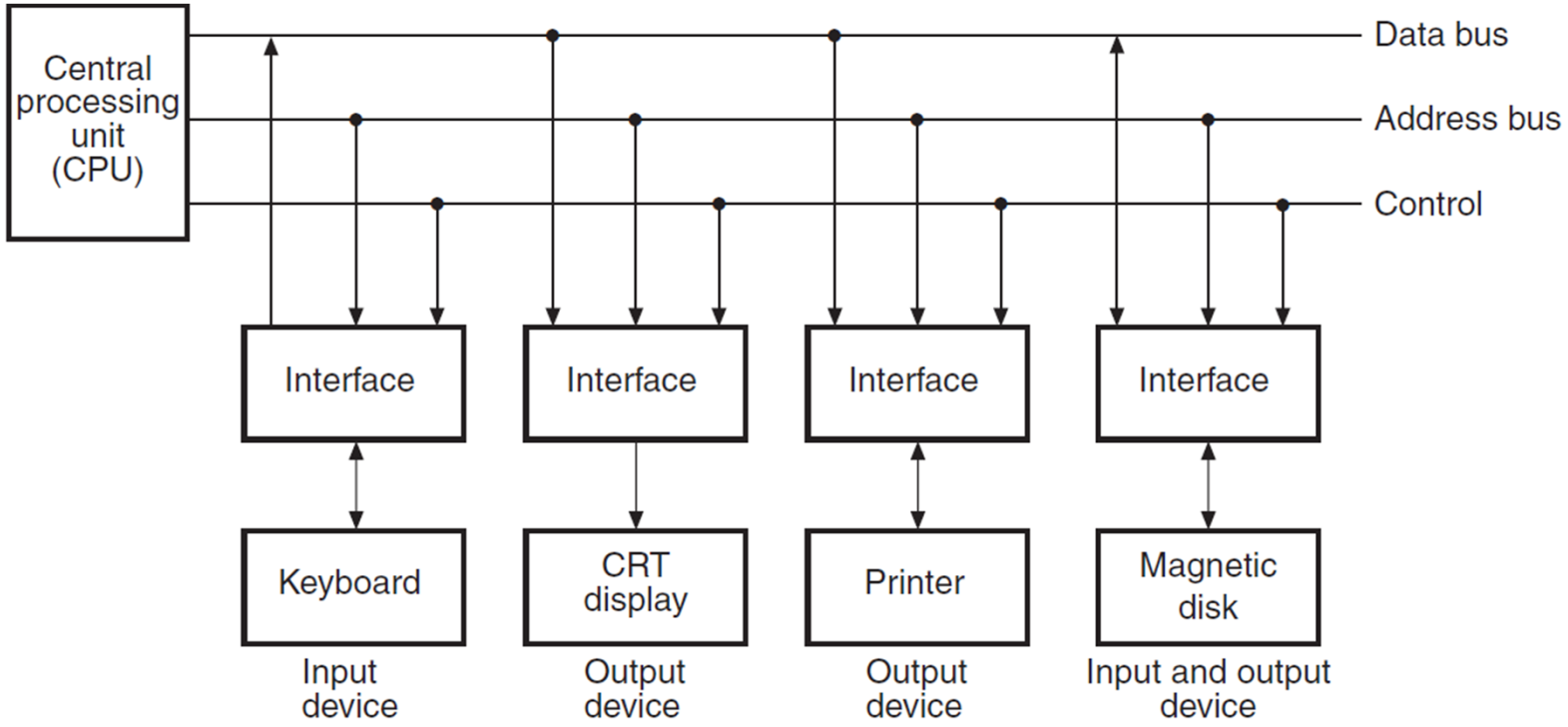
INTERFACES DE ENTRADA/SALIDA (I/O)

- Conversión de formato de señal (periféricos electromecánicos – CPU electrónico)
- Velocidad de transferencia de los periféricos muy diferente a la del reloj de la CPU. Mecanismo de sincronización
- Los códigos de datos y formatos en los periféricos son diferentes a los de la CPU
- Modos de operación de los periféricos muy diferentes. Deben ser controlados de manera independiente

SOLUCIÓN:

- Interfaces de entrada/salida – supervisan y sincronizan todas las señales de I/O entre la CPU y cada uno de los periféricos.
- Controladores para cada periférico para supervisar las operaciones particulares de los mecanismos del periféricos

INTERFACES DE ENTRADA/SALIDA (I/O)



- Cada interfaz posee un decodificador a la entrada del bus de direcciones
- La CPU coloca una dirección en el bus de direcciones y un código de función en las líneas de control
- La interfaz correspondiente habilita la conexión entre el bus de datos y el dispositivo que controla. El resto de las interfaces se desconectan
- La interfaz seleccionada responde a la CPU y ejecuta la función – si se trata de una transferencia de datos sincroniza CPU y controlador

INTERFACES DE ENTRADA/SALIDA

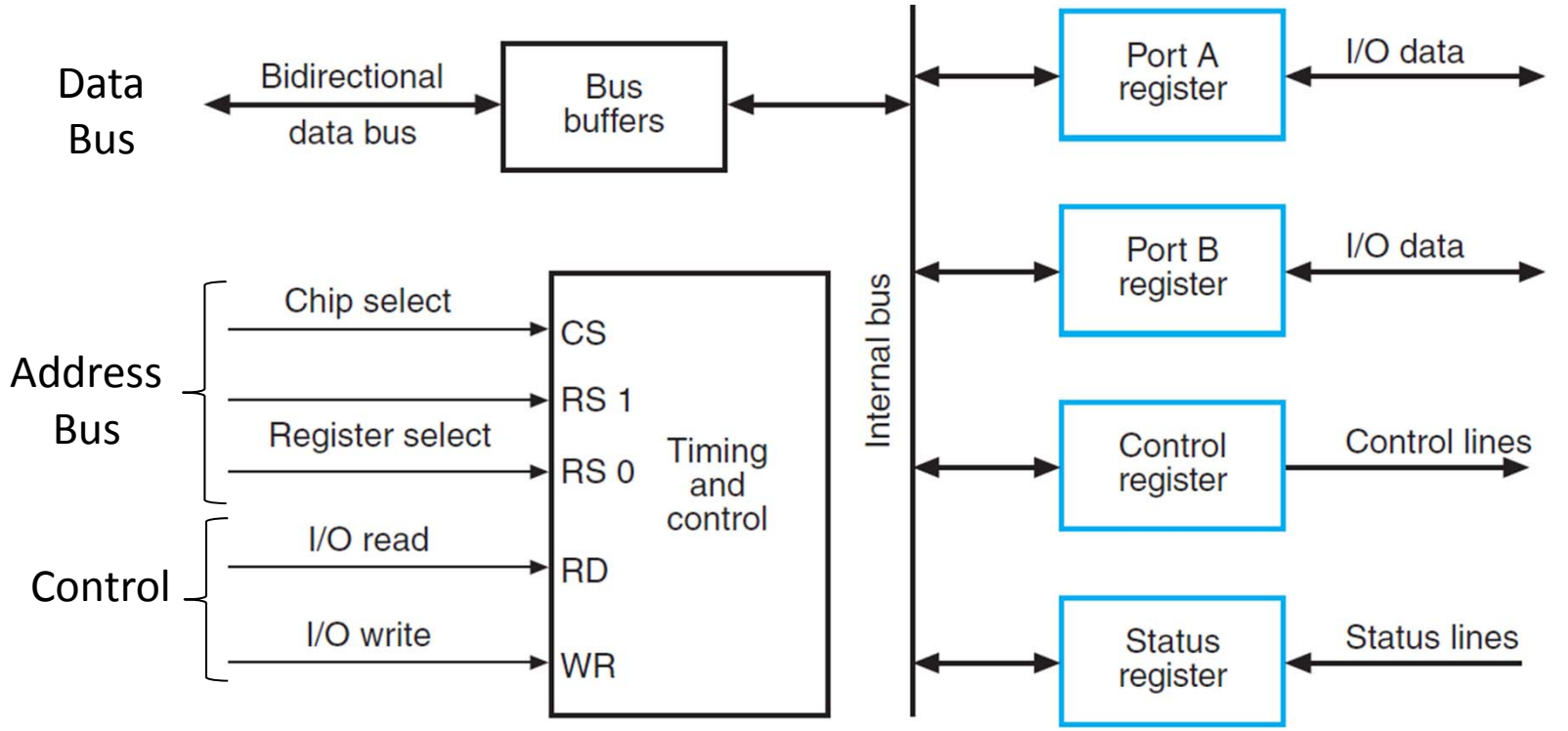
Conexión CPU – Memoria – Sistema I/O:

- Entrada/salida mapeada en memoria *(memory-mapped I/O)* – Mismos buses de dirección, datos y control => direcciones diferentes para memoria y para dispositivos I/O. Instrucciones WRITE y READ para operaciones de entrada y salida.

- Configuración de Entrada/salida aislada *(isolated I/O configuration)* – Mismos buses de dirección y datos pero buses diferentes de control.

- Procesadores de I/O *(data channel)* – Sistemas con un procesador de I/O independiente de la CPU. La memoria se comunica con CPU y procesador I/O a través de un bus común de memoria. El procesador I/O se comunica con los dispositivos I/O a través de buses de dirección, datos y control separados.

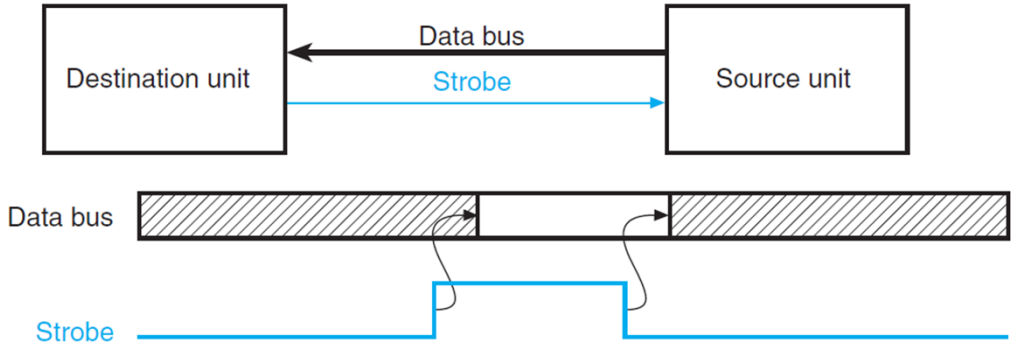
EJEMPLO DE INTERFAZ DE ENTRADA/SALIDA (I/O)



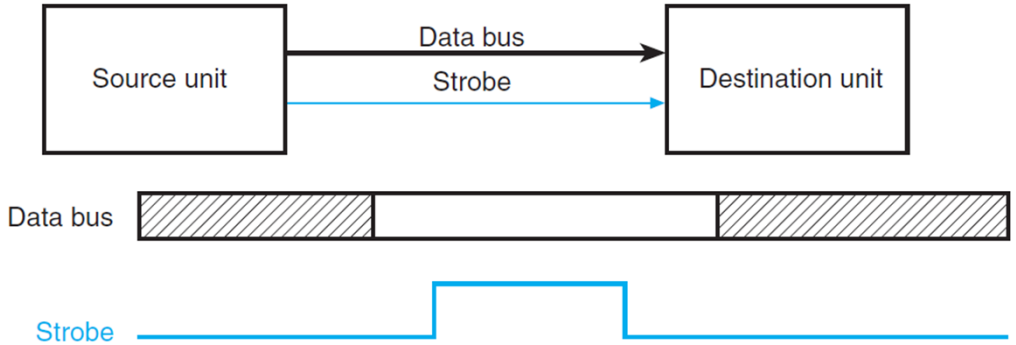
CS	RS1	RS0	Register selected
0	x	x	None: data bus in high-impedance state
1	0	0	Port A register
1	0	1	Port B register
1	1	0	Control register
1	1	1	Status register

INTERFACES DE ENTRADA/SALIDA (I/O)

Transferencia asíncrona con *Strobing*:



(a) Destination-initiated transfer



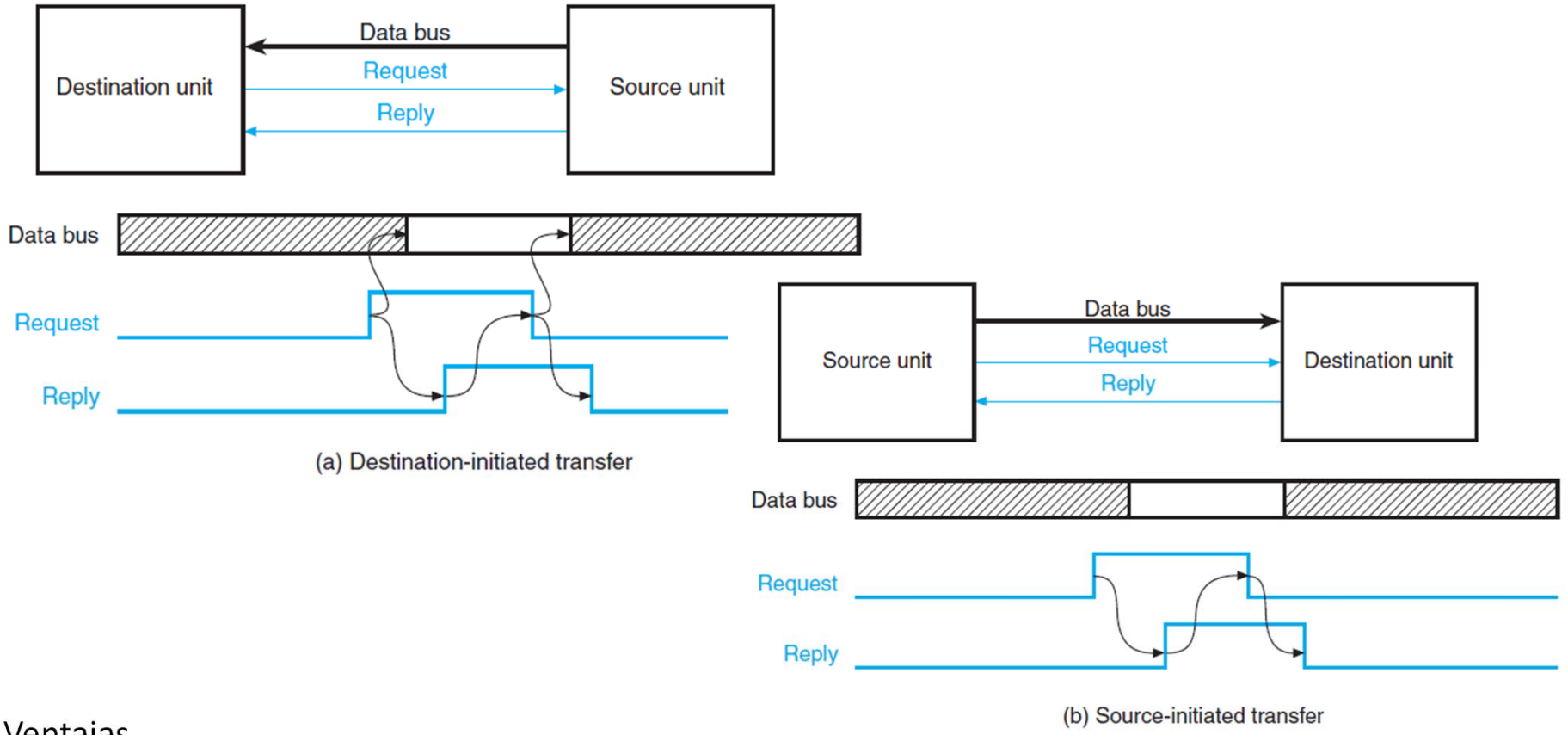
(b) Source-initiated transfer

Inconvenientes

- En (a) no hay indicación de que la fuente haya colocado los datos en el bus
- En (b) no hay indicación de que el destino haya capturado los datos
- Si hay varias unidades, el tiempo para cada transferencia está determinado por la más lenta

INTERFACES DE ENTRADA/SALIDA (I/O)

Transferencia asíncrona con *Handshaking*:



Ventajas

- Gran flexibilidad y fiabilidad ya que el éxito de la transferencia recae en la participación activa de ambas unidades
- Se detecta fácilmente un error por medio de un mecanismo de time-out
- La transferencia tiene lugar en un tiempo que depende de la velocidad de transferencia de las unidades
- Transmisión CPU/interface – Bus de direcciones con la dirección de la interface antes, durante y después de la transmisión

COMUNICACIÓN SERIE

Transmisión en paralelo – rápida, requiere muchas líneas. Distancias cortas

Transmisión en serie – lenta, requiere un único conductor. Distancias largas

Tres modos de transmisión serie:

- Simplex – Una única dirección. No suele utilizarse para transmisión de datos. 1 cable
- Half-duplex – En ambas direcciones pero sólo en una dirección al tiempo. 2 cables
- Full-duplex - En ambas direcciones simultáneamente. 2 cables + 1 línea de tierra

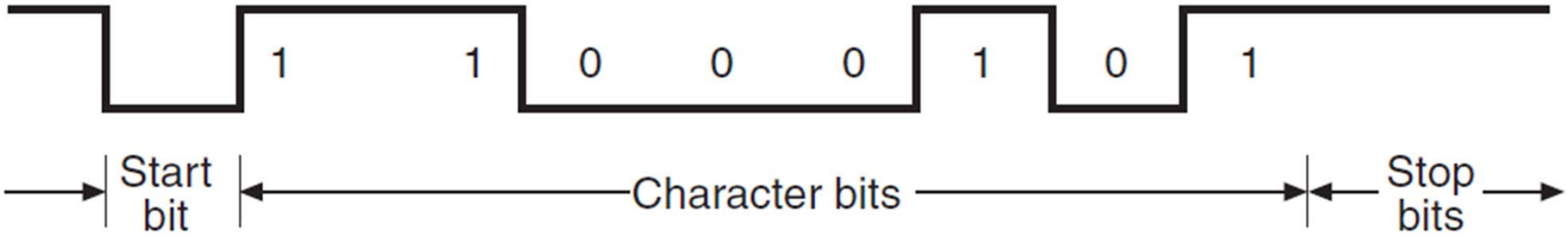
Todas en único cable si hay dos canales de recepción y transmisión con bandas de frecuencia que no se solapan

Transmisión serie:

- Síncrona – Las dos unidades trabajan a una misma frecuencia de reloj, la misma a la que se transmite la información. A larga distancia relojes diferentes a la misma frecuencia. Se trasmiten señales de sincronización periódicamente. SE transmite información ininterrumpidamente para mantener sincronización.
- Asíncrona – Sólo se transmite información cuando está disponible. Si no, linea en IDLE.

COMUNICACIÓN SERIE ASÍNCRONA

- **Cada carácter (unidad de información) contiene:** bit de comienzo (START), bits de datos (DATA) y bit o bits (1 o 2) de parada (STOP)
- El receptor debe conocer la frecuencia de transmisión del emisor así como el número de bits del carácter
- **Ejemplo: Formato de transmisión serie asíncrona - modem**



Transmisión serie:

- Baudio – máximo número de cambios de la señal que se transmite por segundo – a menudo coincide con la velocidad de transferencia de datos en bits por segundo.
  Ejemplo – 10 caracteres por segundo en un formato de 11-bits => velocidad de transferencia de 110 baudios. (comprobar)

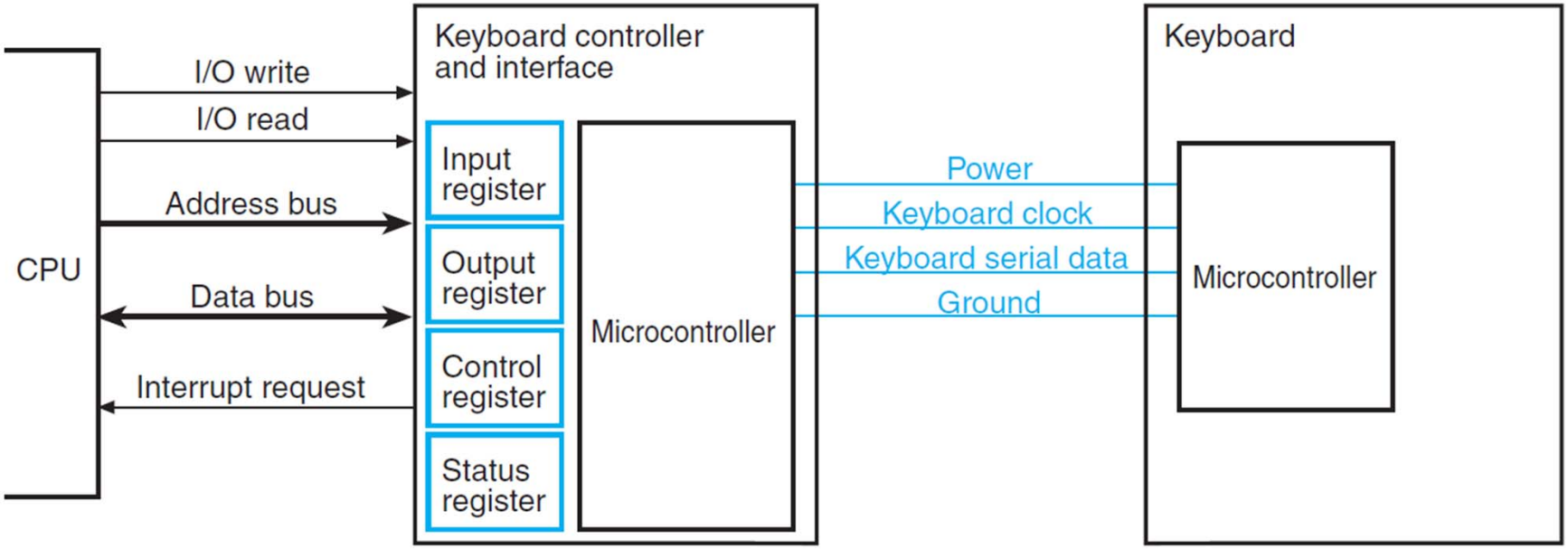
COMUNICACIÓN SERIE SÍNCRONA

- No existen bits de START ni STOP. Se transfiere un conjunto de bits que forman un bloque de datos. Este bloque se transmite con bits de control especiales al comienzo y al final para mantener el sincronismo entre emisor y receptor.
- Receptor y transmisor deben utilizar la misma frecuencia de reloj y además sus relojes deben permanecer sincronizados.
- En el caso de un modem, el receptor deduce la frecuencia de transmisión a partir de los datos que le van llegando. Cualquier cambio en las frecuencias de emisor y receptor se ajusta continuamente gracias a que el reloj del receptor se ajusta continuamente a la frecuencia de los datos que llegan.

COMUNICACIÓN SERIE – EJEMPLO: TECLADO

3 procesadores implicados: microcontrolador en el teclado, interface, CPU

- Microcontrolador en el teclado – transmite en serie un dato en código K-scan en sincronismo con la señal *Keyboard clock*. Las mismas líneas también se utilizan para transmitir información en sentido contrario (comandos de control)..
- La interface convierte el dato a un formato más estándar y lo coloca en el registro de entrada (Input register). Además envía una señal de interrupción a la CPU indicando que una tecla ha sido presionada.
- La CPU ejecuta la rutina de atención a la interrupción que coloca el dato en un lugar reservado de memoria que posteriormente es manipulada por un programa almacenado en la BIOS que lo convierte en un carácter ASCII disponible para las aplicaciones que lo requieran.



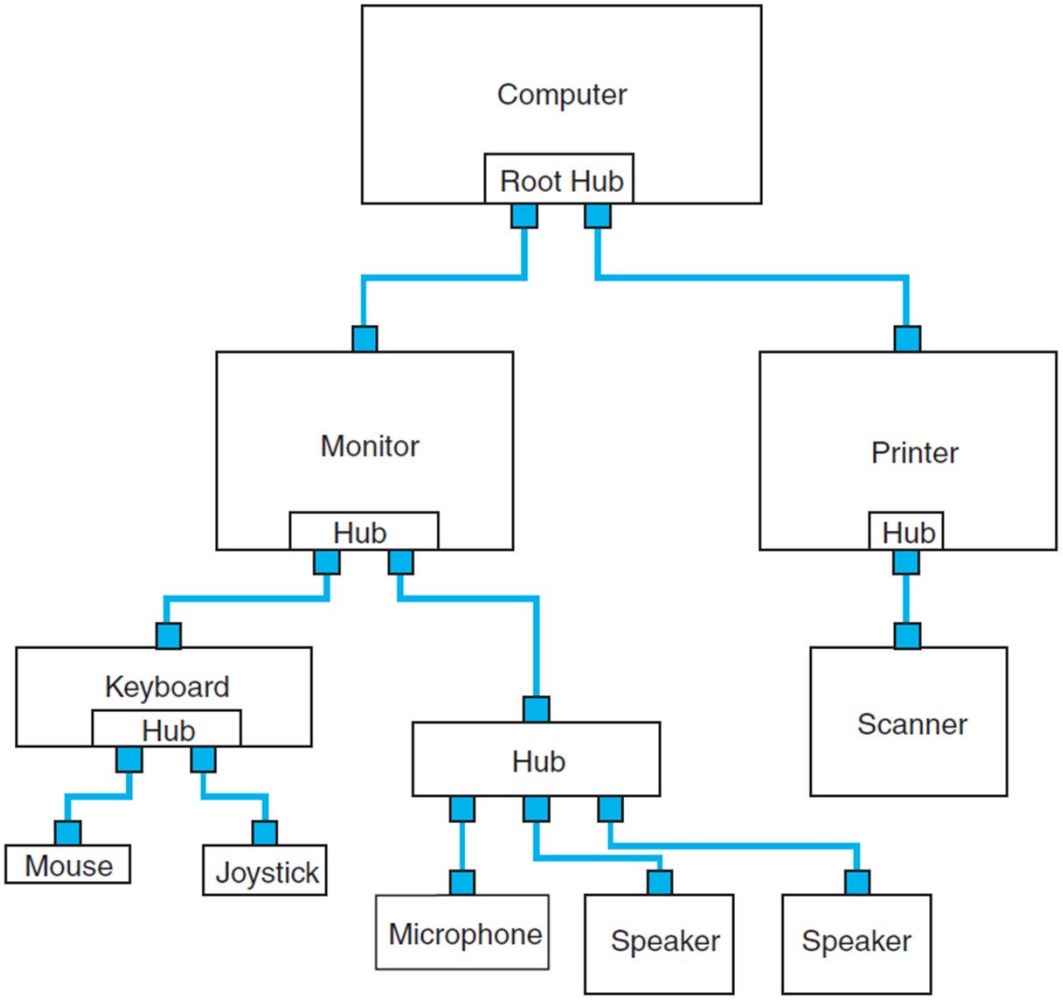
Registro Salida *(Output register),* Registro de Control *(Control register),* Registro de estado *(Status register)*

COMUNICACIÓN SERIE – EJEMPLO: USB

USB: Universal Serial Bus – Packet-based Serial I/O Bus

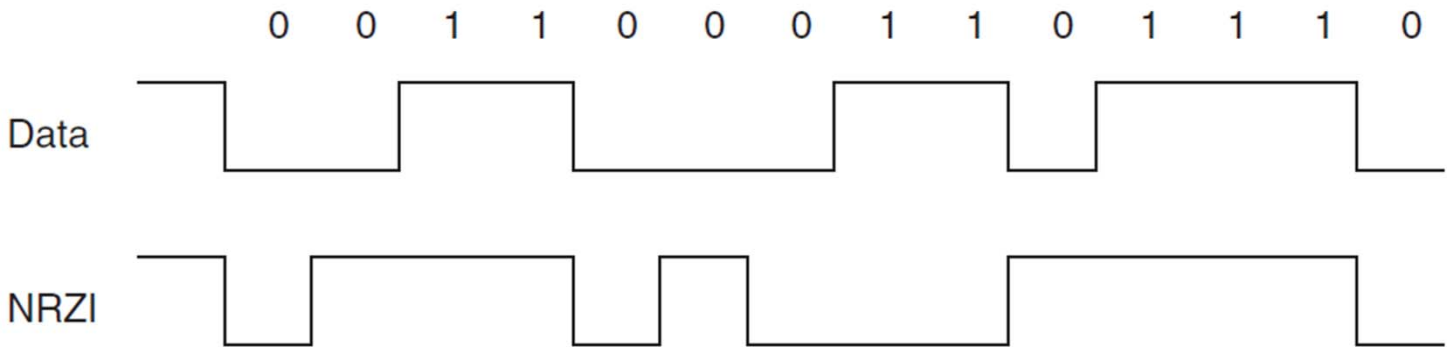
- Permite la conexión de diferentes dispositivos de I/O a una única estructura de comunicación compartida que está conectada al ordenador a través de sólo uno o dos conectores

- Los elementos del sistema de comunicación se clasifican como hubs, dispositivos y dispositivos complejos (contienen un hub). Un hub proporciona puntos de conexión para dispositivos USB y otros hubs. Los hubs contienen una interfaz USB para el procesamiento del control y el estado y un repetidor para transferir información a través del hub. El ordenador contiene un controlador USB y el hub raíz (principal)



COMUNICACIÓN SERIE – EJEMPLO: USB

- 4 cables: – tierra, alimentación y dos líneas de datos: D+ y D- (señal diferencial – mayor inmunidad al ruido). Diferencia mayor de ±200 mV niveles H y L.
- Código **NRZI** (non-Return to Zero Inverter). 0 transición de 0 a 1 ó de 1 a 0. 1 valor fijo en 0 ó en 1. NRZI proporciona flancos para la sincronización. Para evitar desincronizaciones, se intercala un 0 cada 6 datos.



COMUNICACIÓN SERIE – EJEMPLO: USB

Información se transmite en paquetes de datos compuestos de varios campos

SYNC	PID	Packet Specific Data	CRC	EOP

(a) General packet format

SYNC 8 bits	Type 4 bits 1001	Check 4 bits 0110	Device Address 7 bits	Endpoint Address 4 bits	CRC	EOP

(b) Output packet

SYNC 8 bits	Type 4 bits 1100	Check 4 bits 0011	Data (Up to 1024 bytes)	CRC	EOP

(c) Data packet (Data0 type)

SYNC 8 bits	Type 4 bits 0100	Check 4 bits 1011	EOP

(d) Handshake packet (Acknowledge type)

Ejemplo: formatos utilizados en una operación de salida

- SYNC – patrón de sincronización 00000001. Precedido de IDLE
- PID – identificador de paquete. 8 bits – 4 primeros identifican tipo de paquete, 4 últimos son los complementos de los 4 primeros.
- Información del paquete
- CRC (opcional) - de 5 a 16 bits
- EOP - Final del paquete – D+, D- en L durante 2 ciclos seguidos de IDLE durante 1 ciclo.

En todos los campos se transmite el bit menos significativo primero

COMUNICACIÓN SERIE – EJEMPLO: USB

Información se transmite en paquetes de datos compuestos de varios campos

SYNC	PID	Packet Specific Data	CRC	EOP

(a) General packet format

SYNC 8 bits	Type 4 bits 1001	Check 4 bits 0110	Device Address 7 bits	Endpoint Address 4 bits	CRC	EOP

(b) Output packet

SYNC 8 bits	Type 4 bits 1100	Check 4 bits 0011	Data (Up to 1024 bytes)	CRC	EOP

(c) Data packet (Data0 type)

SYNC 8 bits	Type 4 bits 0100	Check 4 bits 1011	EOP

(d) Handshake packet (Acknowledge type)

Campo información del paquete

- (b) – Device Address – dispositivo receptor de datos.
  Endpoint Address –puerto receptor de datos
- (c) – CRC consta de 16 bits ya que la longitud de Data puede ser de hasta 1024 bytes
- (d) – Sin campo de datos. En este caso el PID nos da la información a transmitir.
  - 01001011 Acknowledge ACK
  - No ACK package – error
  - 01011010 No ACK – temporary unable to complete transfer
  - 0111000 STALL –unable to complete transfer – software intervention is requiered

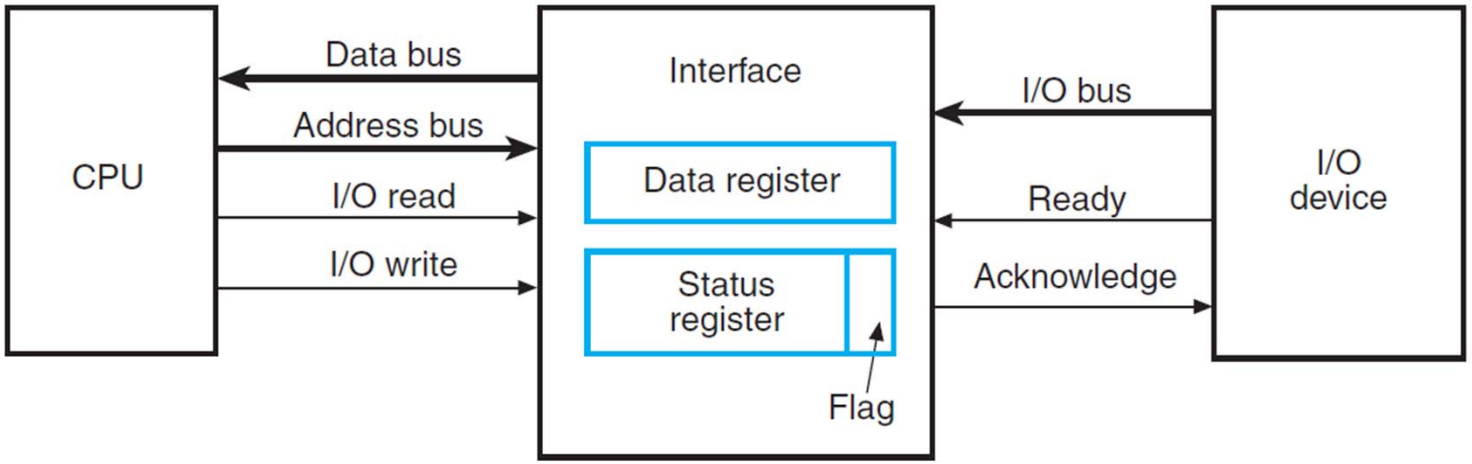
MODOS DE TRANSFERENCIA

- La transferencia de información entre el dispositivo I/O y la memoria
- La CPU ejecuta las instrucciones de I/O y puede aceptar temporalmente los datos
- La transferencia de información desde I/O ó memoria hacia memoria ó I/O se ejecuta en 1 d los 4 modos siguientes:

1. Bajo control de programa – se inicia a través de instrucciones contenidas dentro de un programa. Normalmente entre un I/O y un registro.
2. Iniciada por interrupción
3. Acceso directo a memoria (DMA) – CPU inicializa la transferencia proporcionando a la interface la dirección inicial y el número de palabras a transferir
4. A través de un procesador de I/O (IOP) – Ordenador dividido en 3 módulos: Memoria, CPU e IOP (Input-Output processor)

1.- MODOS DE TRANSFERENCIA – bajo control programa

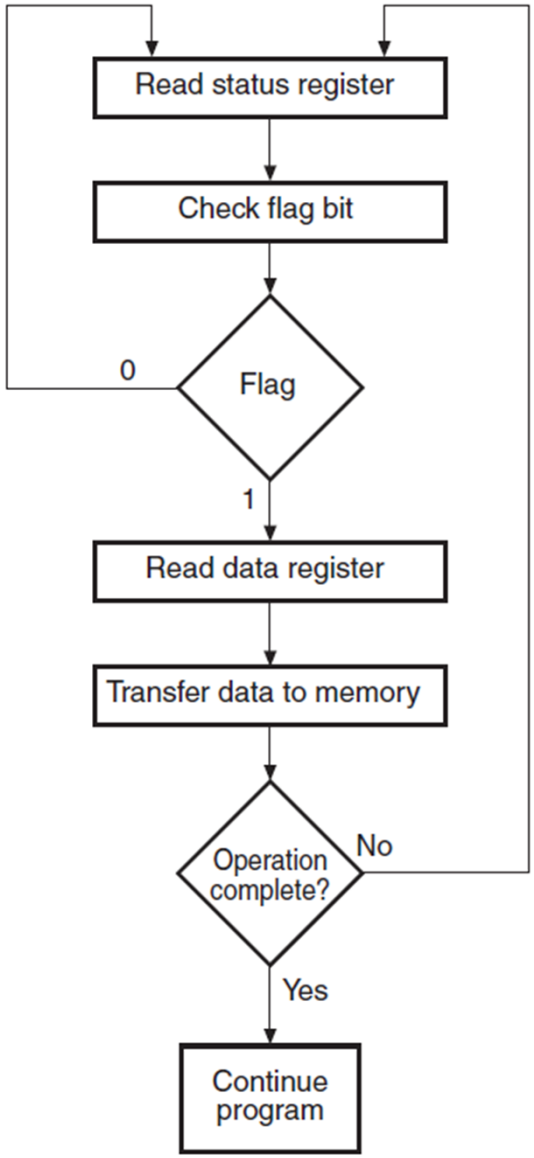
Ejemplo: transferencia desde dispositivo I/O hacia CPU a través de Interface



1. I/O coloca dato en I/O bus y activa Ready
2. La interface graba el dato en el Data Register, coloca Flag=1 en el registro de estado y activa Acknowledge (Handshaking)
3. I/O desactiva Ready
4. CPU comprueba el Flag. Si está a 1, se transfiere el dato a la CPU y la CPU ó la Interface ponen el flag a 0. A continuación Interface desactiva Acknowledge (fin Handsh.). I/O ya puede transmitir otro dato.

1.- MODOS DE TRANSFERENCIA – bajo control programa

Ejemplo: transferencia desde dispositivo I/O hacia CPU a través de Interface



Flag=0 busy wait loop

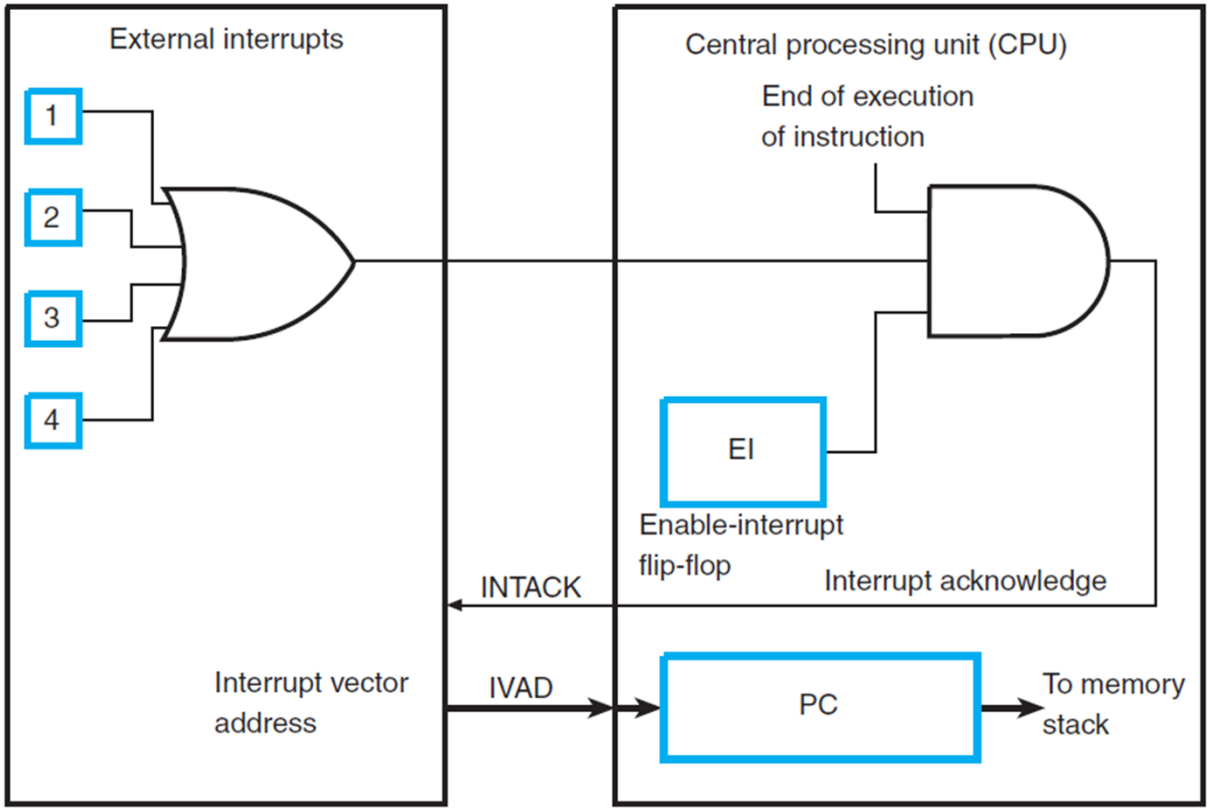
- Velocidades de transferencia de información de CPU e I/O muy diferentes => proceso muy ineficiente. La CPU normalmente dentro de un bucle a la espera de recibir datos desde el I/O sin hacer nada.

  Ejemplo: CPU revisa Flag cada 100 ns.

  Velocidad I/O=100bytes/s => CPU comprueba el flag 100000 veces por cada transferencia
- Sólo se utiliza en sistemas dedicados a monitorizar constantemente un dispositivo.

2.- MODOS DE TRANSFERENCIA – iniciada por interrupción

Alternativa: en vez de CPU compruebe el valor del flag, es la propia interface la que informa al ordenador cuando está lista para la transferencia a través de una interrupción.



- Cuando se activa la petición de servicio, la CPU deja momentáneamente de ejecutar el programa, guarda la dirección del PC (retorno) y pasa a ejecutar la rutina de servicio a la interrupción.

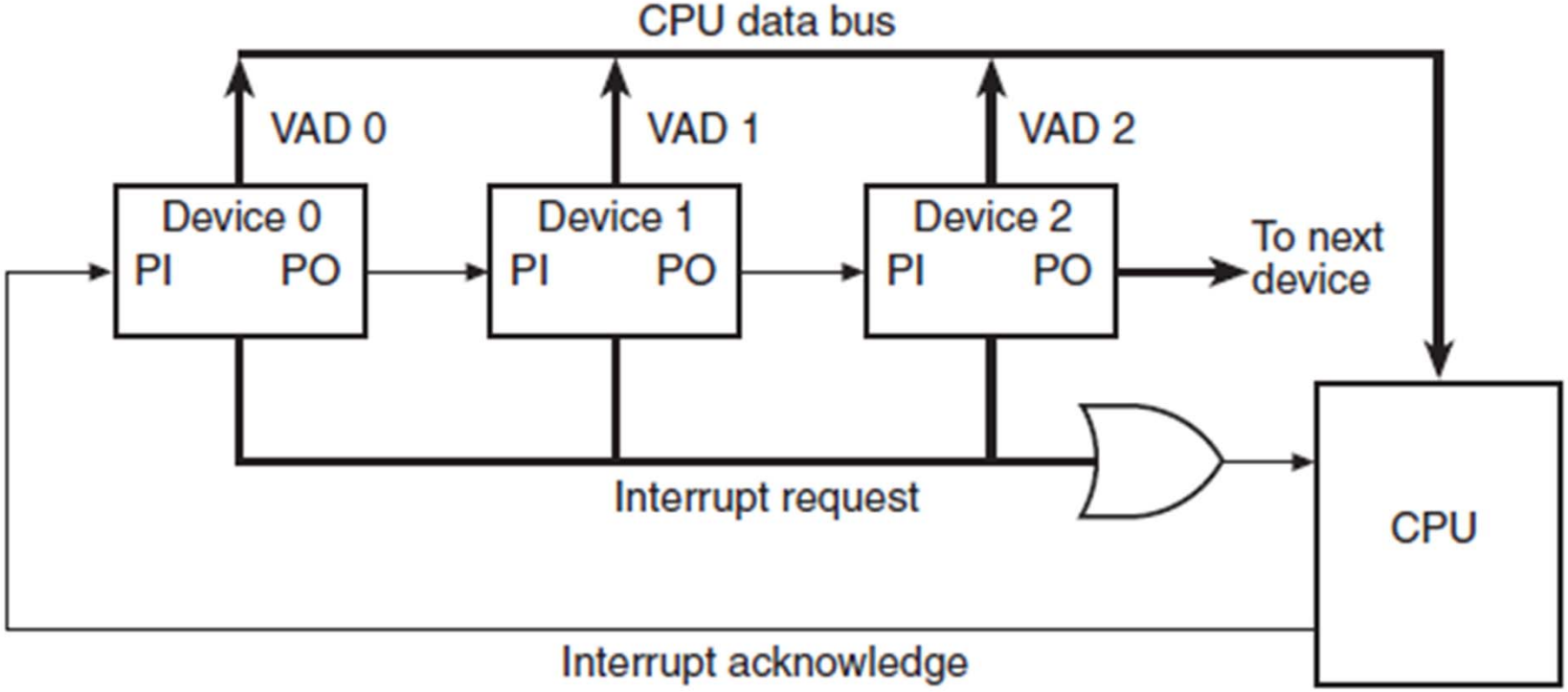
- Interrupciones vectoriales (IVAD) y no vectoriales (dirección fija)
- IVAD puede ser la 1º instrucción de la rutina de servicio o un puntero que apunta a la dirección de memoria donde comienza la rutina de servicio

2.- MODOS DE TRANSFERENCIA – iniciada por interrupción

Prioridades: asignadas por software o hardware

asignadas por Software – Una única rutina decide qué interrupción atender. Ventaja: programable. Desventaja: Lento

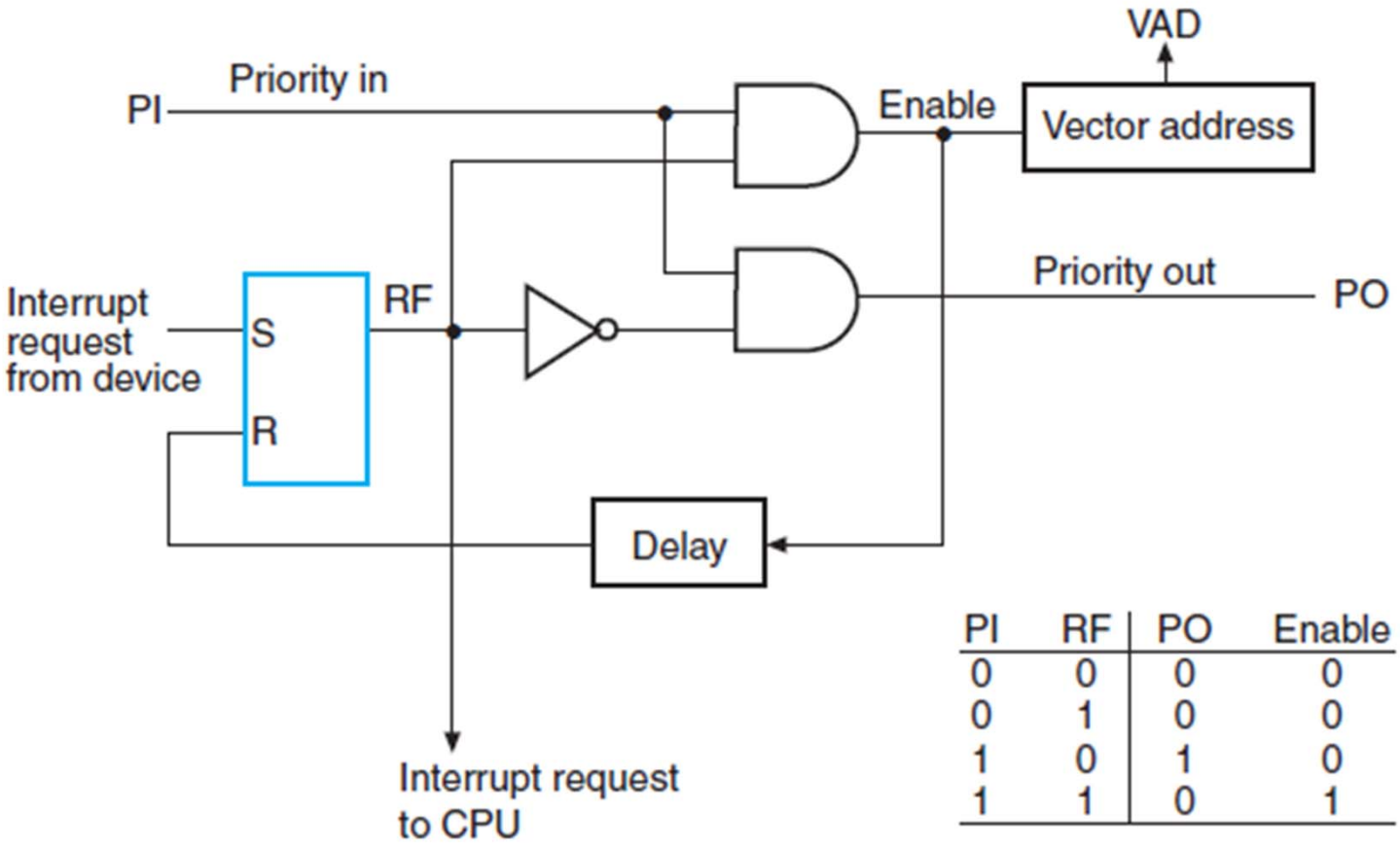
asignadas por Hardware – Cada interrupción tiene su propio vector de interrupción para acceder directamente a su rutina de petición de servicio.



Daisy Chain Priority – conexión serie

2.- MODOS DE TRANSFERENCIA – iniciada por interrupción

Asignadas por Hardware – Daisy Chain Priority



PI	RF	PO	Enable
0	0	0	0
0	1	0	0
1	0	1	0
1	1	0	1

2.- MODOS DE TRANSFERENCIA – iniciada por interrupción

Asignadas por Hardware – Parallel Priority Interrupt

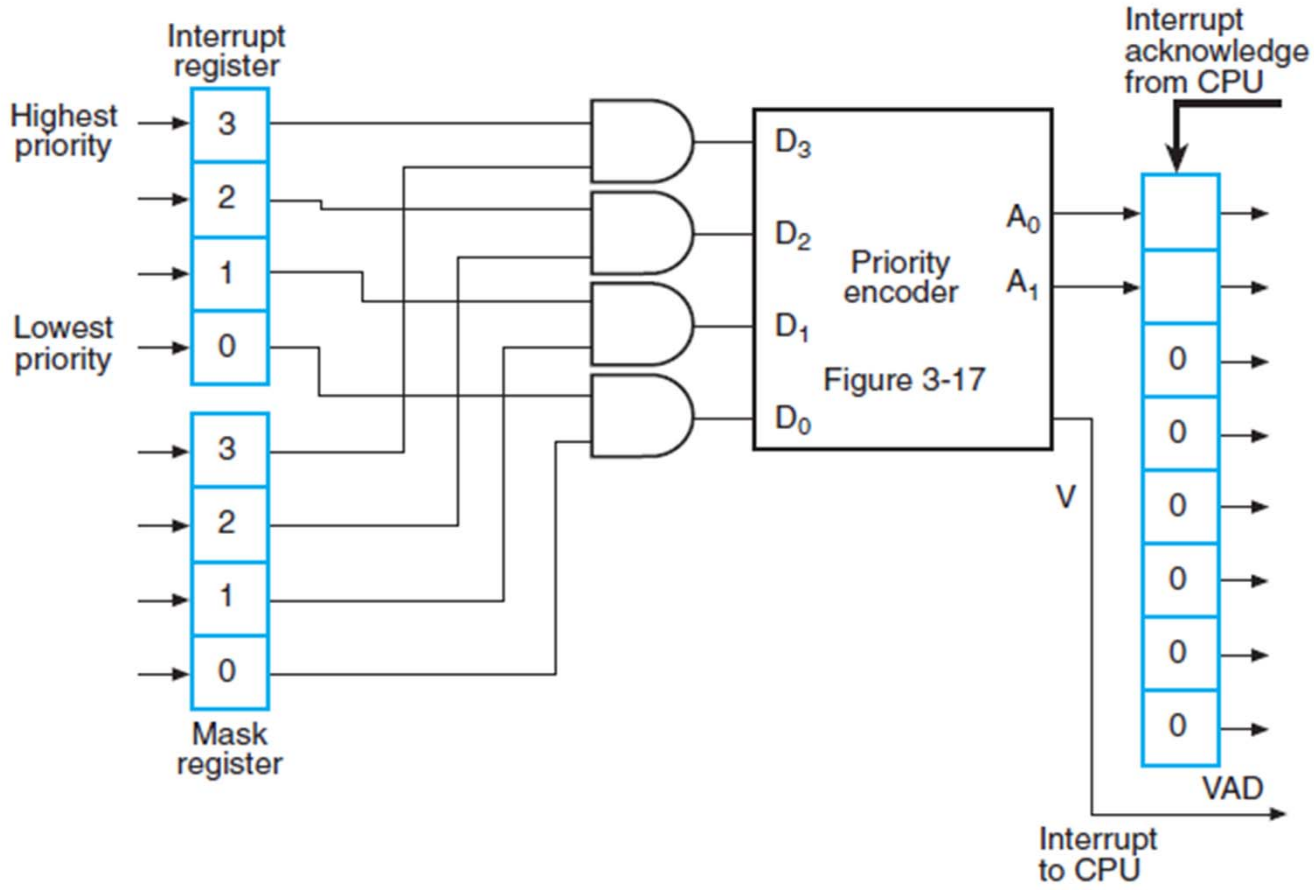


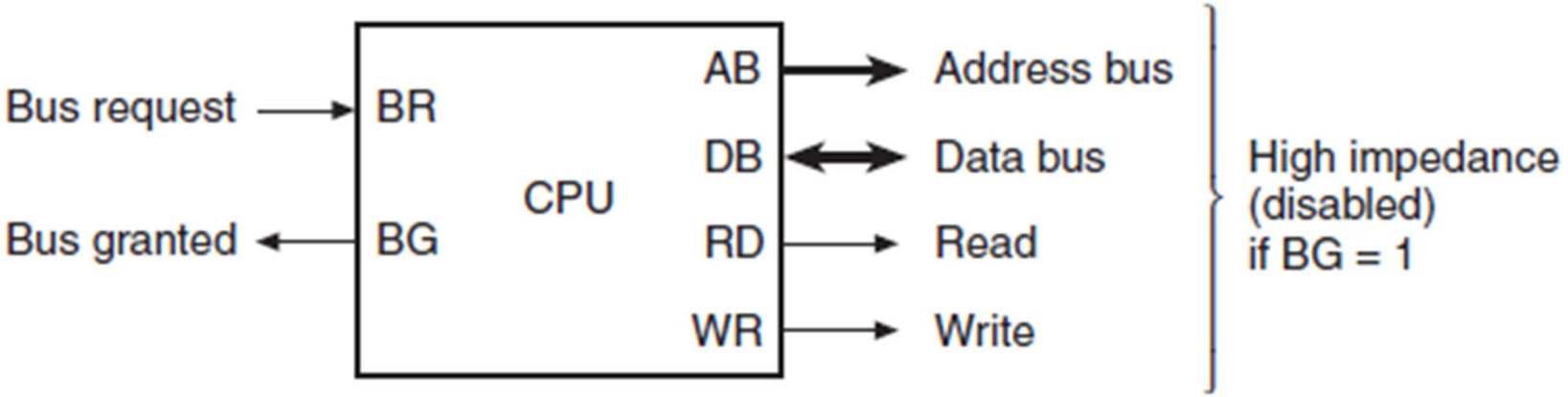
Figure 3-17

- Registro de máscaras – mientras se atiende una interrupción, inhibe las de menor prioridad. También puede permitir interrupciones de mayor prioridad de la que se está atendiendo (anidadas)

3.- MODOS DE TRANSFERENCIA – DMA

- Se utiliza para la transferencia de bloques de información entre dispositivos de almacenamiento rápido (i.e. disco duro) y memoria
- La CPU cede el control del Bus al controlador DMA que dirige la transferencia directa entre el dispositivo I/O y la memoria. De este modo, la CPU se puede dedicar a otras tareas mientras se realiza la transferencia

Señales de control del Bus y de transferencia en la CPU

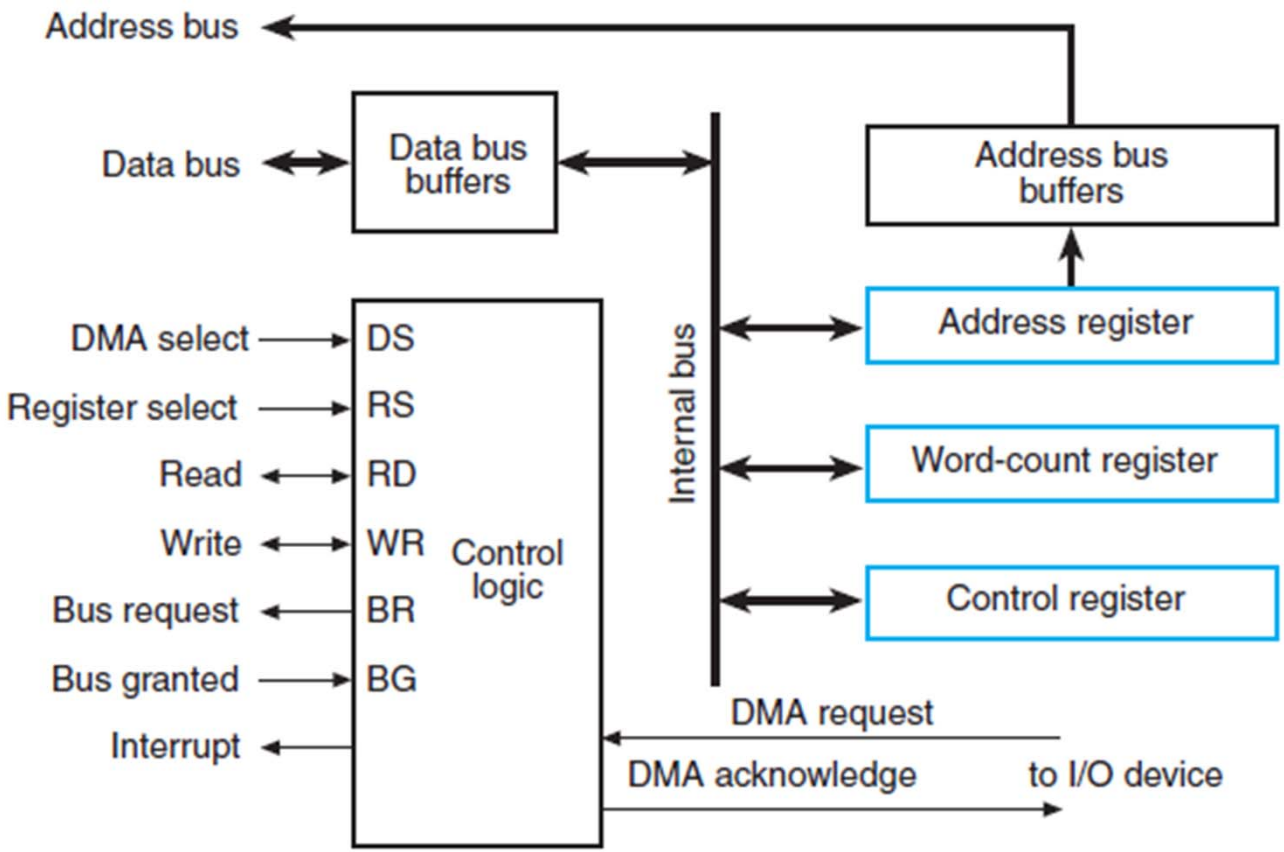


Modos de transferencia bajo el controlador DMA (BG=1):

- Modo ráfaga (Burst transfer)
- Modo de ciclo robado (single-cycle transfer, cycle stealing)

3.- MODOS DE TRANSFERENCIA – DMA

Diagrama de Bloques de un controlador DMA



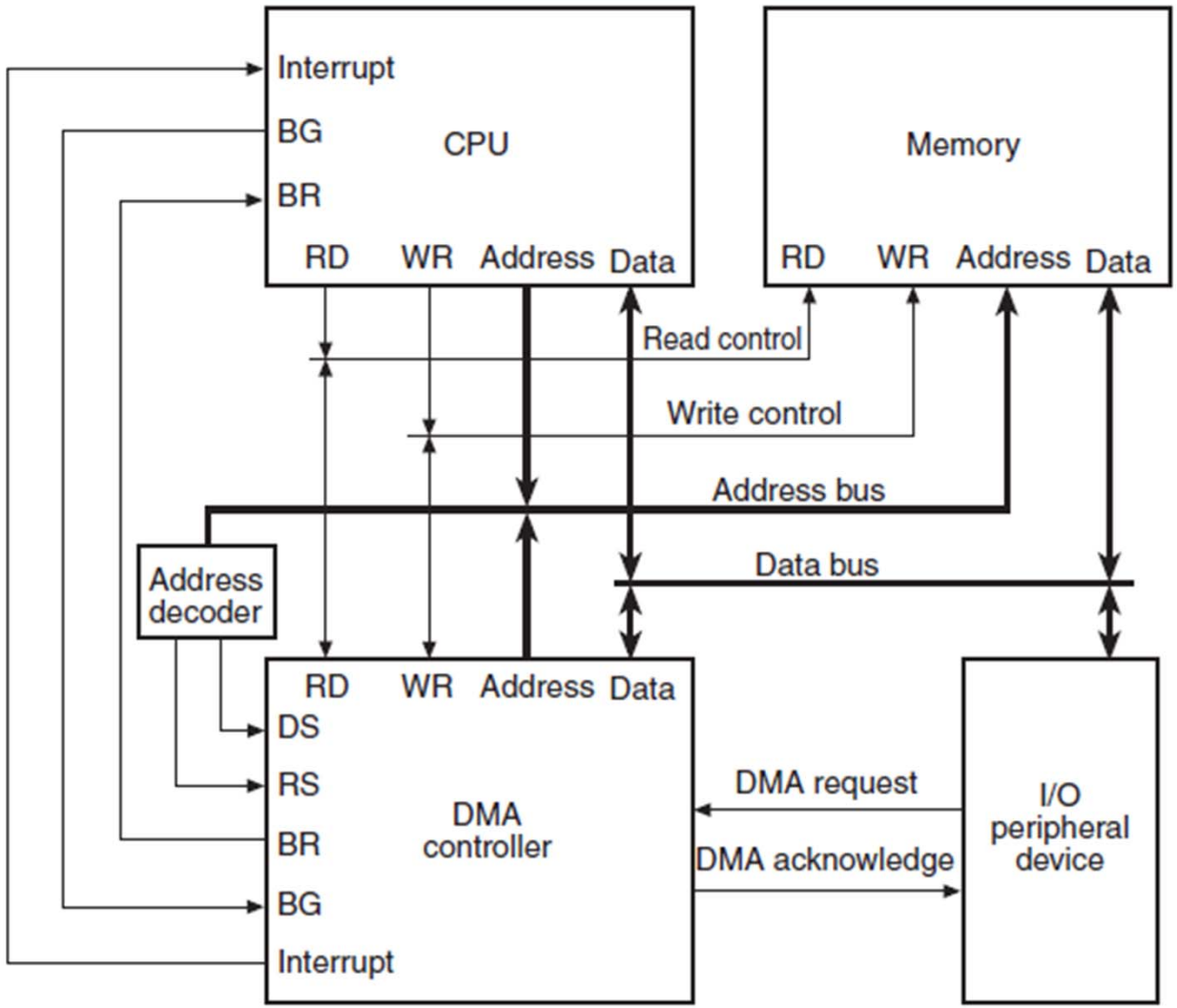
CPU escribe/lee los registros del controlador DMA a través del bus de datos cuando BG=0

- Address register: dirección de memoria donde realizar la operación
- Word-count register: almacena el número de palabras a transmitir. Se decrementa cada vez que se transfiere una palabra y se comprueba si es o no 0
- Registro de control: almacena el modo de transferencia.

Inicialización del controlador DMA por la CPU: 1 – dirección inicial del bloque de memoria. 2 - número de palabras a transmitir. 3 – modo de transferencia. 4 – bit de comienzo de transferencia.

3.- MODOS DE TRANSFERENCIA – DMA

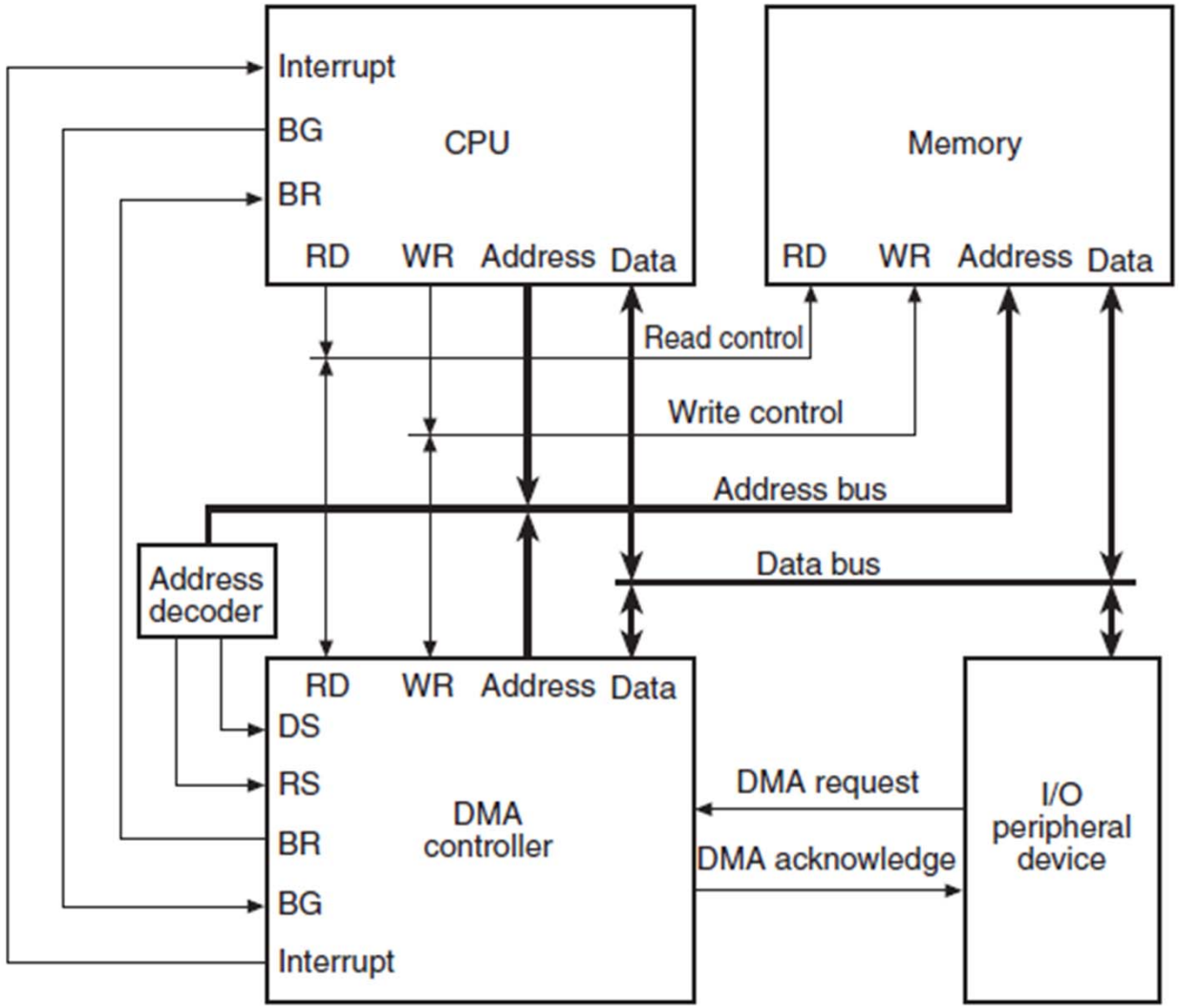
Sistema de transferencia completo DMA



- CPU inicializa controlador DMA a través del bus de datos y de direcciones indicando los contenidos de los registros del DMA y activando el bit de comienzo
- Cuando el dispositivo I/O envía a la DMA una «DMA request», ésta activa BR solicitando a la CPU el control de los buses
- CPU responde activando BG
- DMA coloca el valor de su registro de dirección en el bus de direcciones, activa el modo lectura o escritura y envia «DMA acknowledge» al disp I/O
- El disp I/O coloca/lee el dato en el bus de datos. Mientras, el acceso de la CPU al bus de datos está deshabilitado

3.- MODOS DE TRANSFERENCIA – DMA

Sistema de transferencia completo DMA



- Por cada palabra transferida el cont. DMA incrementa su registro de dirección y decrementa el registro contador de palabras. Mientras éste sea distinto a 0, el DMA espera otro «DMA request» desde el I/O (I/O de alta velocidad) y se repite la transferencia o bien deshabilita BR para devolver el control a la CPU (I/O de baja velocidad) esperando a una nueva DMA request.
- Cuando el registro contador de palabras llega a 0, DMA deshabilita BR y activa una interrupción.
- La CPU comprueba que efectivamente el registro contador de palabras es 0.

Controlador DMA con más de un canal: Cada canal tiene dos líneas BR y BG y su registro de dirección y contador de palabras de modo que se establecen prioridades entre ellos.

4.- MODOS DE TRANSFERENCIA – IOP

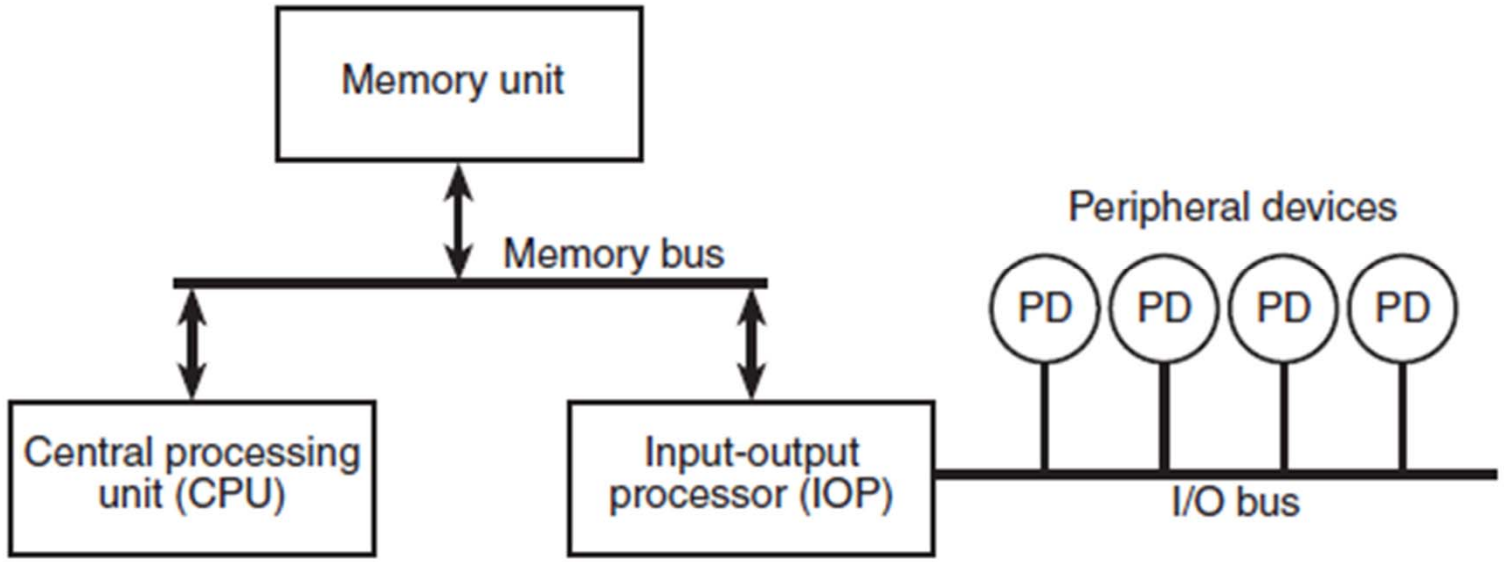
- IOP – es un procesador con acceso directo a memoria que se comunica con los dispositivos I/O. Se encarga de las tareas de I/O dejando para la CPU solo la asignación de las transferencias.

- Es similar a una CPU, excepto que se diseña para realizar tareas de I/O. Al contrario que el controlador DMA, el IOP busca y ejecuta sus propias instrucciones (comandos).

- Los comandos están diseñados específicamente para facilitar las transferencias I/O y además algunas tareas de procesamiento (aritméticas, logicas, de bifurcación…)

4.- MODOS DE TRANSFERENCIA – IOP

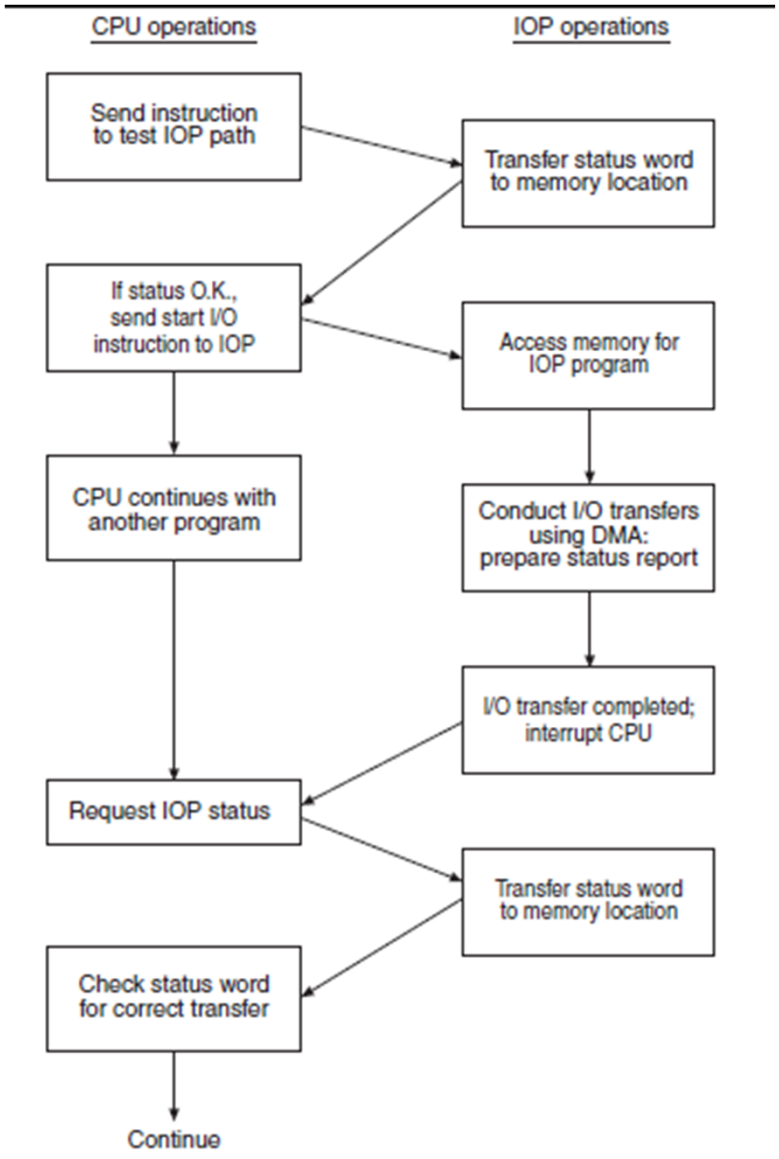
Diagrama de bloques de un ordenador con procesador I/O



- Comunicación entre el IOP y los dispositivos es similar a la realizada bajo control de programa.
- Comunicación entre el IOP y la memoria es similar a DMA
- La comunicación entre la CPU y el IOP depende del sistema. En los grandes cada procesador trabaja por separado y cualquiera puede inicializar una operación. En la mayoría la CPU es el maestro y el IOP el esclavo. La CPU inicializa las operaciones pero las instrucciones las ejecuta la IOP.

MODOS DE TRANSFERENCIA – IOP

Esquema del protocolo de comunicación entre CPU e IOP



- En la mayoría de los casos, la memoria actúa como un centro de mensajes, donde cada procesador deja información para el otro.
- CPU e IOP compiten por el uso de la memoria. La mayoría de los dispositivos de I/O son lentos y no supone problema pero para los casos de unidades rápidaas como pueden ser discos duros o tarjetas gráficas la velocidad de la CPU puede verse afectadas debido a que la CPU a menudo debe esperar a que el IOP acabe la transferencia