Estructura de Computadores 2 [08/09]

Tema 6: Organización de la Entrada/Salida

1. Introducción
2. Medidas de rendimiento
3. Modelo de periférico
4. Interfaz CPU - E/S
5. Gestión de la E/S: E/S programada
6. Gestión de la E/S: E/S mediante interrupciones
7. Gestión de la E/S: Acceso directo a memoria (DMA)
8. Canales y procesadores de E/S

22 de abril de 2009

Bibliografía

- *Computer Architecture: A Quantitative Approach (3rd or 4th ed.)*. John L. Hennessy y David A. Patterson. Morgan Kaufmann Publishers, Inc.
- *Organización y arquitectura de computadores (7th ed.)*. William Stallings. Prentice Hall.
- *Organización de Computadores*. C. Hamacher, Z. Vranesic y S. Zaky. Mc Graw Hill, 2003.
- *Computer Organization and Design: The hardware/software interface (3rd ed.)*. David A. Patterson and John L. Hennessy. Morgan Kaufmann Publishers, Inc.

GAC: Grupo de Arquitectura de Computadores. Dpt. Electrónica e Sistemas. Universidade da Coruña.

Introducción

- Tres subsistemas en un computador: procesador, memoria y **sistema de entrada/salida**
- Sistema de E/S (*I/O system*): formado por los componentes que permiten el movimiento de datos entre los dispositivos externos y el tándem CPU - Memoria. Incluye:
  - Dispositivos de E/S: constituyen la interfaz del ordenador con el exterior
    - Periféricos que interaccionan con el usuario para proporcionar entradas al sistema: teclado, ratón, etc.
    - Dispositivos que interaccionan con otros dispositivos: redes de comunicación, etc.
    - Dispositivos de almacenamiento ...
  - Interconexión entre procesador, memoria y los diferentes dispositivos periféricos
    - conexión física entre componentes
    - mecanismos básicos de transmisión de información
    - interfaz con los disposivos de E/S
    - organización de las operaciones de E/S...
- Gran parte de las características de los sistemas de E/S vienen determinadas por la tecnología existente en cada momento, además de verse condicionadas por el resto de componentes del sistema
- Características E/S:
  - Amplia variedad de dispositivos con formas diferentes de funcionamiento
  - Velocidad de transferencia de los periféricos mucho menor que la de CPU o memoria
  - Diferentes formatos y tamaños de datos (palabra)
- Parámetros de diseño de sistemas de E/S: rendimiento, escalabilidad o expansibilidad y tolerancia a fallos

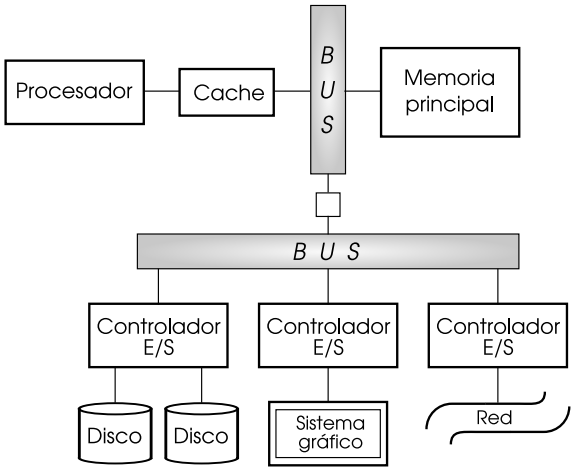
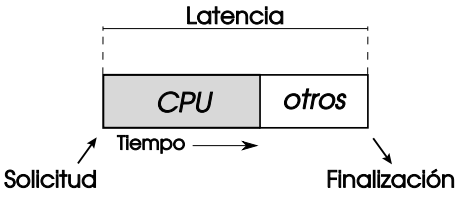


Figura 1: Componentes básicos de un computador: CPU, memoria, dispositivos de E/S y sistema de interconexión

Medidas de rendimiento

- **Latencia**
  (*aka* tiempo de respuesta o tiempo de ejecución)
  (en inglés: *latency/response time/execution time/elapsed time/wall time*)

  → **Tiempo total** transcurrido desde el comienzo hasta la finalización de una tarea
  (medido en unidades de tiempo o en ciclos de reloj)



T. CPU = T. usuario + T. sistema
Otros: acceso memoria, esperas E/S, esperas ejecución otros programas...

  Mayor rendimiento = Menor latencia

- **Ancho de banda**
  (*aka* potencia o productividad)
  (en inglés: *bandwidth/throughput*)

  → **Cantidad** de trabajo realizado **en un tiempo** determinado
  (medido en cantidad por unidad de tiempo)

  En función del contexto, las dos maneras más habituales de medir el ancho de banda son:

  – Cantidad de datos que pueden moverse a través del sistema en un determinado tiempo (ancho de banda de datos o *data rate*)
  – Número de operaciones de E/S que se pueden realizar por unidad de tiempo (ancho de banda de operaciones o *I/O rate*)

  Mayor rendimiento = Mayor ancho de banda

- El ancho de banda es la tasa a la que un sistema es capaz de atender peticiones, y no siempre es el inverso de la latencia: la concurrencia en el manejo de peticiones permite incrementar el ancho de banda por encima de $\frac{1}{latencia}$

- Ancho de banda, latencia y coste están íntimamente relacionados. En líneas generales:

  – El ancho de banda se puede incrementar aumentando el coste del sistema
  – La latencia puede ser mucho más difícil de mejorar sin cambiar la tecnología de implementación.

- Otros índices de rendimiento de E/S:

  Interferencia de la E/S con el procesador. Porcentaje de ciclos de reloj que emplea el procesador en tareas de E/S

  Diversidad. Variedad de dispositivos de E/S que pueden conectarse al sistema

  Capacidad/escalabilidad/expansibilidad. Número de dispositivos de E/S que pueden conectarse al sistema

  Capacidad de almacenamiento, en el caso dispositivos de almacenamiento.

Medidas de rendimiento: E/S vs. CPU-Mem

- Rendimiento en un computador
  - a menudo se tiene en cuenta únicamente el rendimiento de la CPU, obviando el resto del sistema.
    De esta forma estamos considerando únicamente:

$$\frac{tiempo}{programa} = \frac{tiempo}{ciclo} \times \frac{ciclos}{instrucción} \times \frac{instrucciones}{programa}$$

  - Optimizar ese tiempo de CPU no es la única forma de incrementar el rendimiento del sistema: memoria y E/S influyen también en gran medida en ese rendimiento. La E/S suele ser la gran olvidada.
  - Opciones de mejora:
    - Optimizar CPU: maximizar velocidad y eficiencia de las operaciones ejecutadas por la CPU
    - Optimizar memoria: maximizar eficiencia en el acceso a memoria
    - Optimizar E/S: maximizar eficiencia de las operaciones de E/S
  - En función de qué factor está gravando más el rendimiento del sistema distinguimos entre **sistemas**
    - **limitados por CPU** (*CPU bound*)
    - **limitados por memoria** (*Memory bound*)
    - **limitados por E/S** (*I/O bound*)

- Medida de rendimiento tras incorporar una mejora a un sistema: **aceleración** (*speedup*)

$$\text{Aceleración} = \frac{\text{Rendimiento}_{\text{ tras mejora}}}{\text{Rendimiento}_{\text{ antes mejora}}} = \frac{\text{T. Ejecución}_{\text{ antes mejora}}}{\text{T. Ejecución}_{\text{ tras mejora}}}$$

- Es importante acotar la aceleración potencial a obtener con una posible mejora (que puede ser muy costosa) antes de realizarla:

  Ley de Amdahl Establece que la aceleración completa de un sistema depende tanto de la aceleración concreta de uno de sus componentes como de cuánto es usado ese componente por el sistema

Medidas de rendimiento: E/S vs. CPU-Mem (II)

- La **ley de Amdahl** nos dice cómo calcular la aceleración que se obtendrá en un sistema con la incorporación de una determinada mejora, en función de dos factores:

  1. La fracción del tiempo de ejecución en el sistema original que va a aprovecharse de la mejora ($Fracción_{\,mejora}$)
  2. La aceleración que se lograría si el sistema completo pudiera aprovecharse de la mejora ($Aceleración_{\,mejora} > 1$)

  El tiempo de ejecución del sistema tras la incorporación de la mejora será la suma del tiempo de ejecución de la parte del sistema que no se ve afectada por la mejora, más el tiempo de ejecución de la parte mejorada:

$$\begin{aligned} T.\,Ejecución_{\,nuevo} &= T.\,Ejecución_{\,original} \times (1 - Fracción_{\,mejora}) \\ &\quad + \frac{T.\,Ejecución_{\,original}}{Aceleración_{\,mejora}} \times Fracción_{\,mejora} \end{aligned}$$

$$T.\,Ejecución_{\,nuevo} = T.\,Ejecución_{\,original} \times \left((1 - Fracción_{\,mejora}) + \frac{Fracción_{\,mejora}}{Aceleración_{\,mejora}} \right)$$

  Por tanto:

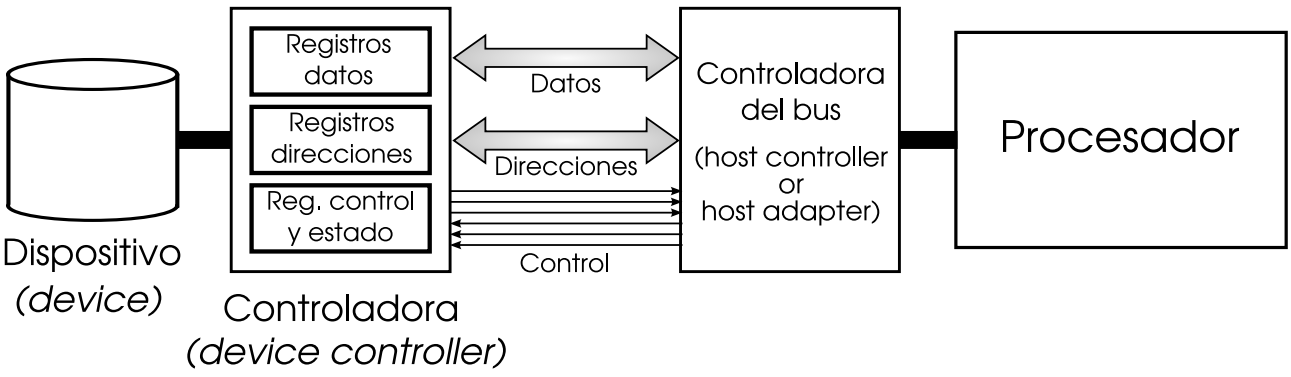
$$Aceleración_{\,conjuntaglobal} = \frac{T.\,Ejecución_{\,original}}{T.\,Ejecución_{\,nuevo}} = \frac{1}{(1 - Fracción_{\,mejora}) + \frac{Fracción_{\,mejora}}{Aceleración_{\,mejora}}}$$

- Por supuesto, la mayor mejora en el rendimiento de un sistema se producirá cuando aceleremos el componente que más lo está limitando.

- Tradicionalmente se ha dejado a la E/S de lado, en favor del tándem CPU-Memoria. Las cosas han cambiado.
  Supongamos, por ejemplo, una tarea con:

	CPU acelerada	→ (Amdahl)	Latencia acelerada
Latencia: L	$x10$		$x5$
Tiempo de CPU: $0{,}9 \times L$	$x100$		$x10$

Modelo de dispositivo de E/S



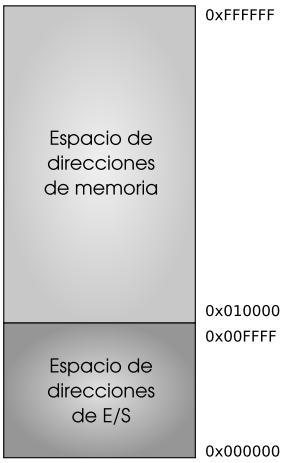
Dispositivo de E/S. Distinguimos 2 componentes fundamentales:

- Dispositivo físico: constituido normalmente por la mayor parte del dispositivo. En determinados periféricos, como dispositivos de almacenamiento, se trata de una parte esencialmente mecánica. Es la parte del dispositivo que realmente realiza las tareas de las que se encarga el periférico.
- Controladora de dispositivo: parte electrónica que sirve de interfaz entre el dispositivo (o los dispositivos) y el resto del sistema Funciones principales:
  - control y temporización
  - almacenamiento temporal de datos (*buffering*)
  - detección de errores
- La CPU se comunica con los diferentes dispositivos esencialmente utilizando los registros de E/S de los mismos.

Interfaz (lógica) CPU - E/S

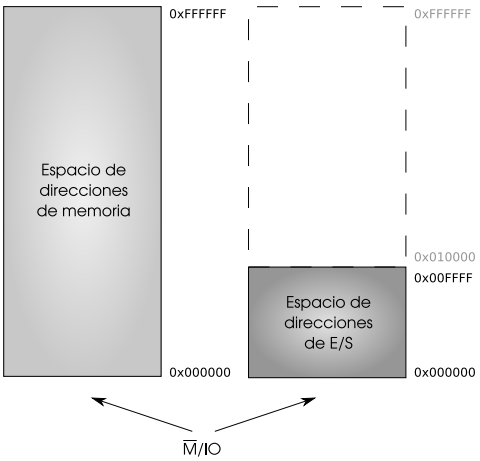
Para poder ser accedido, un dispositivo de E/S tiene que ser direccionable por el procesador. Dos aproximaciones:

E/S asignada al espacio de memoria (*Memory-mapped I/O*, *MMIO*). Con este esquema, una parte del mapa de memoria del sistema es reservado para E/S en lugar de para memoria, asignado las direcciones en ese rango a los distintos dispositivos de E/S. De esta forma, existe un único espacio de direcciones en el sistema, que se emplea tanto para la memoria principal como para la E/S.



- La CPU trata los registros de los disp. E/S como posiciones de memoria, utilizando las mismas instrucciones para acceso a memoria y a disp. E/S
- Permite un diseño de la CPU más sencillo, y por tanto una CPU más rápida y más barata
- Esta es la alternativa para el tratamiento de la E/S más popular hoy en día, usada sobre todo en arquitecturas RISC y sistemas empotrados
- Ejemplos: PDP-11, VAX, ARM, familia Motorola 68000 ...

E/S aislada (*Isolated I/O*, *Instruction-based I/O*, *Port I/O* o *Port-mapped I/O*, *PMIO*). En este caso se mantienen espacios de direcciones diferentes para memoria y E/S, disponiendo el procesador de instrucciones especiales para el acceso a direcciones de E/S, distintas a las de acceso a memoria

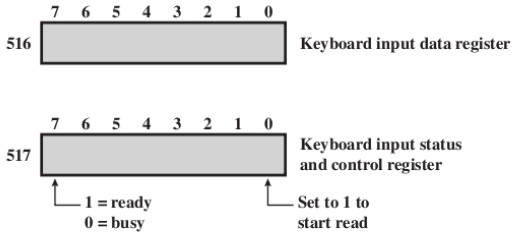


- Al tener los disp. E/S un mapa de memoria separado de la memoria principal, la CPU utiliza una señal de control ($\bar{M}/IO$) para indicar si la dirección que está en el bus de memoria es de E/S, o bien se utiliza un bus de E/S propio
- Popular en su momento en máquinas con una capacidad limitada de direccionamiento. Se sigue utilizando, pero menos
- Ejemplos: la arquitectura intel x86, IBM 370 ...

Interfaz CPU - E/S: MMIO vs. PMIO

- La E/S aislada tiene la ventaja de no restringir el espacio de direcciones a utilizar, además de facilitar la protección de la E/S. El código ensamblador se hace también más limpio de cara al usuario, al ser distinguibles las operaciones especiales de E/S.
- La E/S asignada en memoria se beneficia, por su parte, del uso de un conjunto mayor y más flexible de instrucciones (las de acceso a memoria), lo que además facilita, simplifica y abarata el diseño de la CPU. Además, con el paso a arquitecturas de 32 y 64 bits el espacio de direccionamiento ha dejado de ser un problema.

Ejemplos de código ensamblador para entrada de datos por teclado:



ADDRESS	INSTRUCTION	OPERAND	COMMENT
200	Load AC	"1"	Load accumulator
	Store AC	517	Initiate keyboard read
202	Load AC	517	Get status byte
	Branch if Sign = 0	202	Loop until ready
	Load AC	516	Load data byte

(a) Memory-mapped I/O

ADDRESS	INSTRUCTION	OPERAND	COMMENT
200	Load I/O	5	Initiate keyboard read
201	Test I/O	5	Check for completion
	Branch Not Ready	201	Loop until complete
	In	5	Load data byte

(b) Isolated I/O

Gestión de la E/S

Normalmente un evento de E/S implica más que una simple operación. Tres son las técnicas más utilizadas para gestionar las operaciones de E/S, en lo que respecta al control de la comunicación entre el módulo de E/S y la CPU:
1. E/S programada 2. E/S con interrupciones 3. E/S mediante acceso directo a memoria

E/S programada

- Es la técnica de gestión de E/S más simple. Los datos son intercambiados directamente entre el procesador y el dispositivo de E/S
- La CPU ejecuta un programa que le da un control completo y directo sobre la operación de E/S, incluyendo la comprobación del estado del dispositivo, el envío de comandos de lectura y escritura, y la transferencia de datos.
- Cada vez que la CPU le envía un comando al disp. tiene que esperar a que la operación de E/S se complete, interrogando periódicamente al disp. sobre su estado, lo que se conoce como **encuesta** (*polling*).
- El dispositivo de E/S nunca interrumpe al procesador tras acabar una operación. Simplemente indica el hecho con sus bits de estado.
- Al ser el procesador mucho más rápido que el dispositivo, se están desperdiciando muchos ciclos de reloj en encuestar al dispositivo, degradando el rendimiento del sistema.
- La E/S programada es muy útil en sistemas de propósito específico como cajeros automáticos o sistemas empotrados para el control o monitorización de eventos.

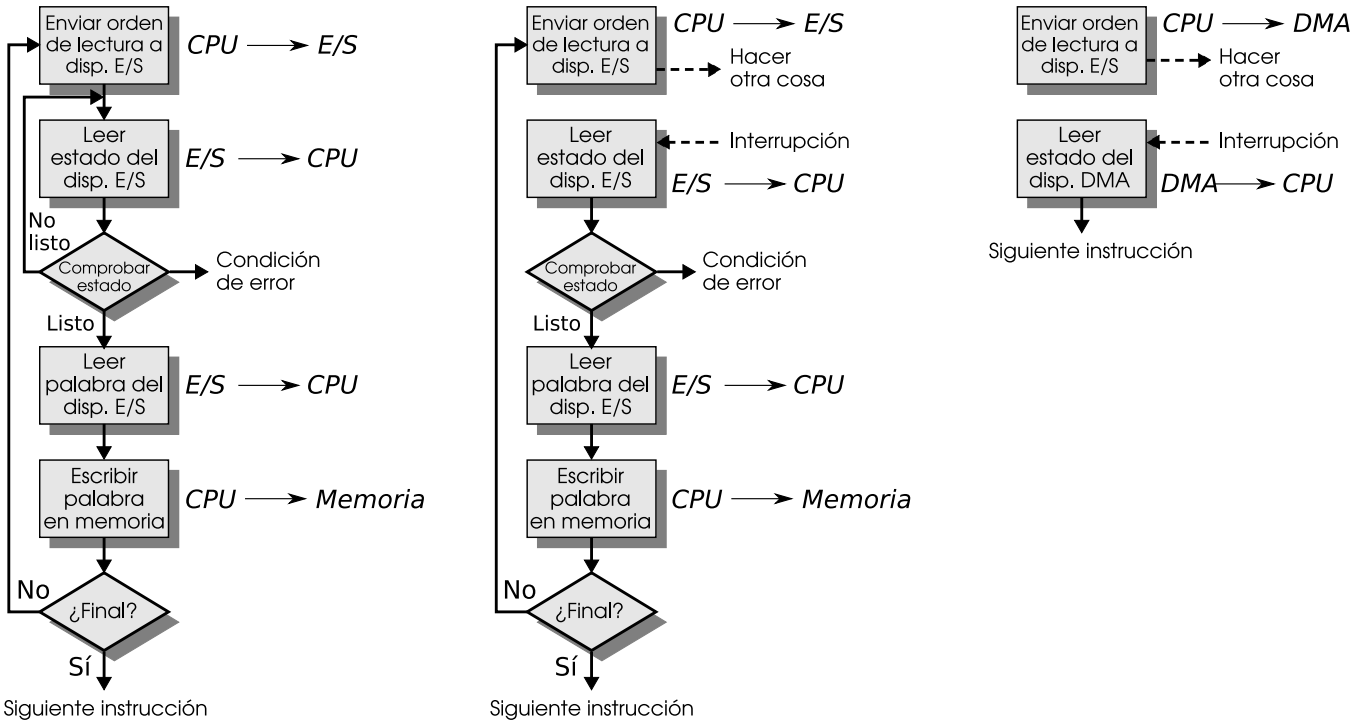


Figura: Tres maneras de gestionar la lectura de un dato

E/S con interrupciones

- La E/S con interrupciones permite al procesador trabajar en otro proceso mientras se espera por una operación de E/S.
- El disp. E/S interrumpirá al procesador para solicitar sus servicios cuando esté listo para completar la operación: línea de petición de interrupción ($\overline{\text{INT}}$)
- El procesador, interrumpido, realiza la transferencia de datos (mediante la ejecución de la rutina de manejo de la interrupción) y a continuación retoma la tarea que estaba llevando a cabo antes de ser interrumpido.
- El mecanismo de interrupciones es una de las llaves de la multitarea, pero en determinados sistemas (pe. sistemas de tiempo real con mucha E/S) la sobrecarga introducida en el SO para el manejo de cada evento de E/S puede ser excesiva.

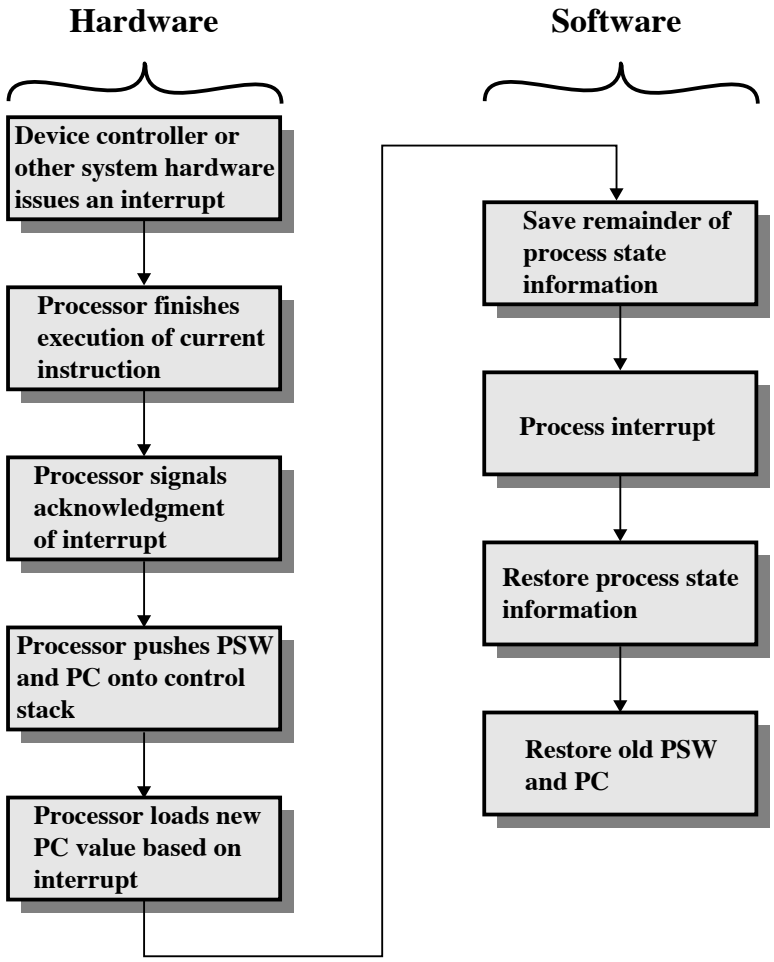


Figure 7.6 Simple Interrupt Processing

E/S con interrupciones: ciclo de ejecución de una instrucción sin interrupciones

E/S con interrupciones: ciclo de ejecución de una instrucción con interrupciones

E/S con interrupciones: ejemplo estado de la memoria

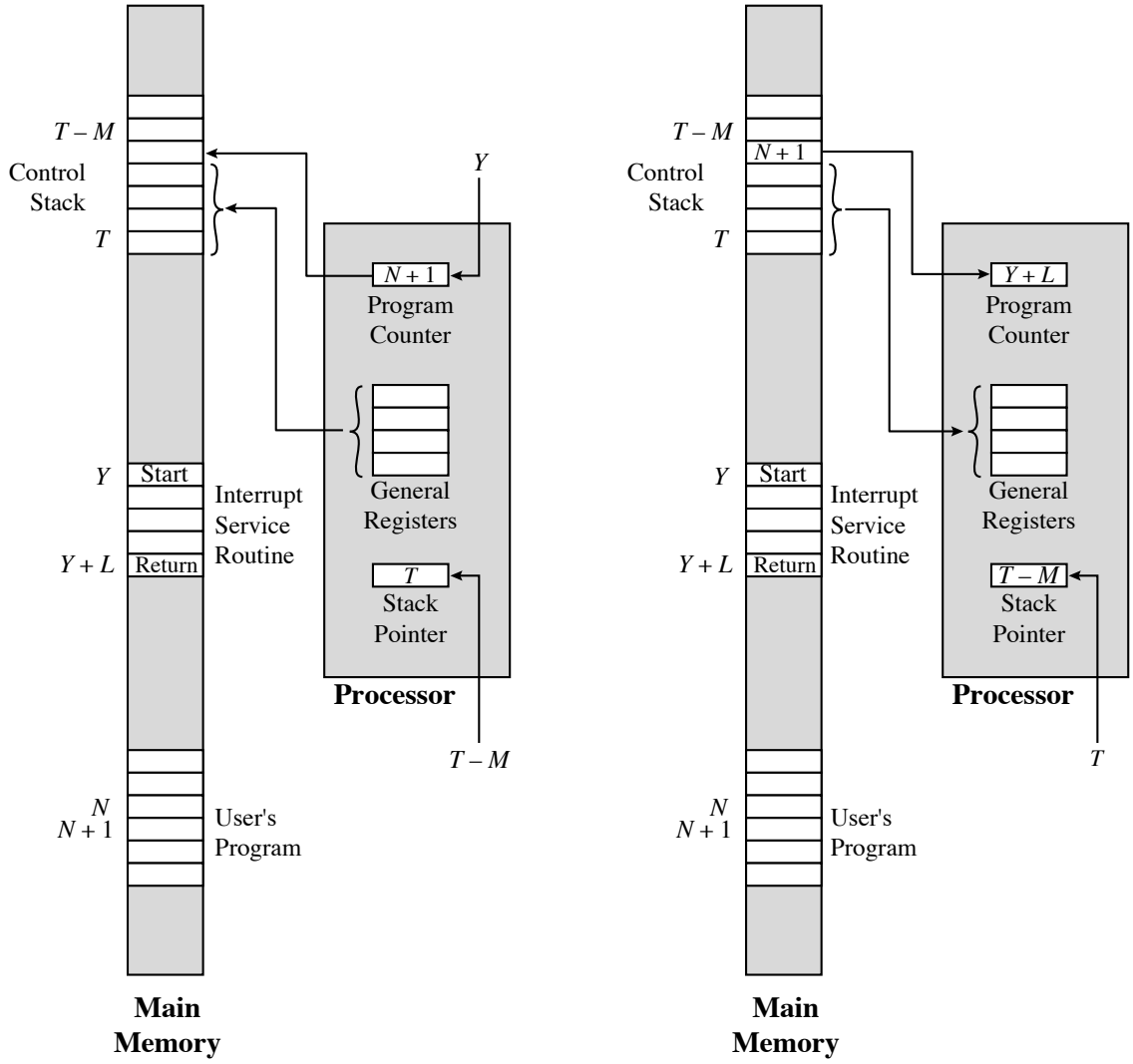


Figure 7.7 Changes in Memory and Registers for an Interrupt

E/S con interrupciones: identificación del dispositivo y asignación de prioridades

Identificación del dispositivo que realiza la petición de IRQ

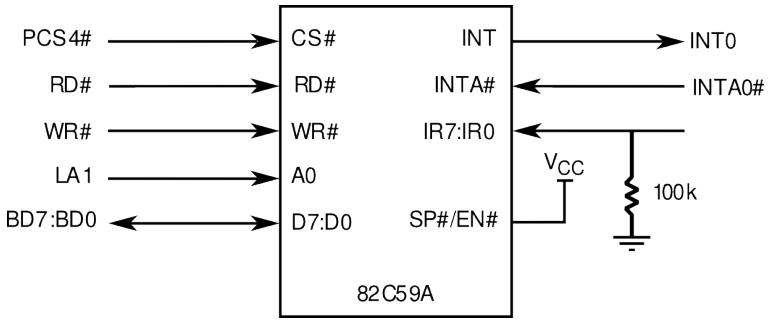
- Múltiples líneas de interrupción. El número de líneas de bus y de conexiones de la CPU que se pueden dedicar a IRQs es limitado, por lo que esta técnica por si misma suele ser siempre insuficiente y es acompañada de alguna de las otras.
- Consulta software (*Software poll*). Una rutina genérica de manejo de interrupciones se encarga de encuestar a los diferentes dispositivos -comprobando el valor del bit IRQ en el registro de estado- cuando se produce una IRQ. Una vez identificado el dispositivo se ejecuta su rutina de servicio asociada.
- Interrupciones vectorizadas (*Vectored interrupt*). Se trata de lograr la autoidentificación del dispositivo para ejecutar directamente su rutina de servicio, a la que se accede mediante lo que se conoce como *vector de interrupción*, que proporciona **en el bus de datos** el dispositivo peticionario.
  El vector suele consistir en un código con el que la CPU es capaz de componer la dirección de comienzo de la rutina.
  Se necesita, además, una línea adicional de aceptación de interrupción (INTA).
  Múltiples alternativas de implementación, normalmente relacionadas con cómo se gestiona la prioridad en las peticiones.

Manejo de múltiples IRQS

- Deben existir mecanismos para habilitar/deshabilitar total o parcialmente -de forma selectiva- las interrupciones
- Manejo de prioridades ante múltiples IRQS:
  - Con múltiples líneas de IRQ: se asigna una prioridad a cada línea y se utiliza un pequeño dispositivo que haga de árbitro de prioridades.
  - Con consulta software: prioridad determinada por el orden en la consulta
  - En el caso de interrupciones vectorizadas hay múltiples alternativas:
    - Conexión en cadena (*Daisy chain*): dispositivos conectados en cadena a la línea INTA. El orden en la conexión determina la prioridad.
    - Arbitraje de bus: solo un disp. -el maestro del bus- puede interrumpir.
- La asignación de prioridades a los dispositivos también es útil para permitir el anidamiento de interrupciones: una interrupción que da servicio a un dispositivo poco prioritario puede ser interrumpido por otro que lo es más.

E/S con interrupciones: Interrupciones en el Intel 80386

- Una única línea de petición (INTR), una única línea de aceptación (INTA)



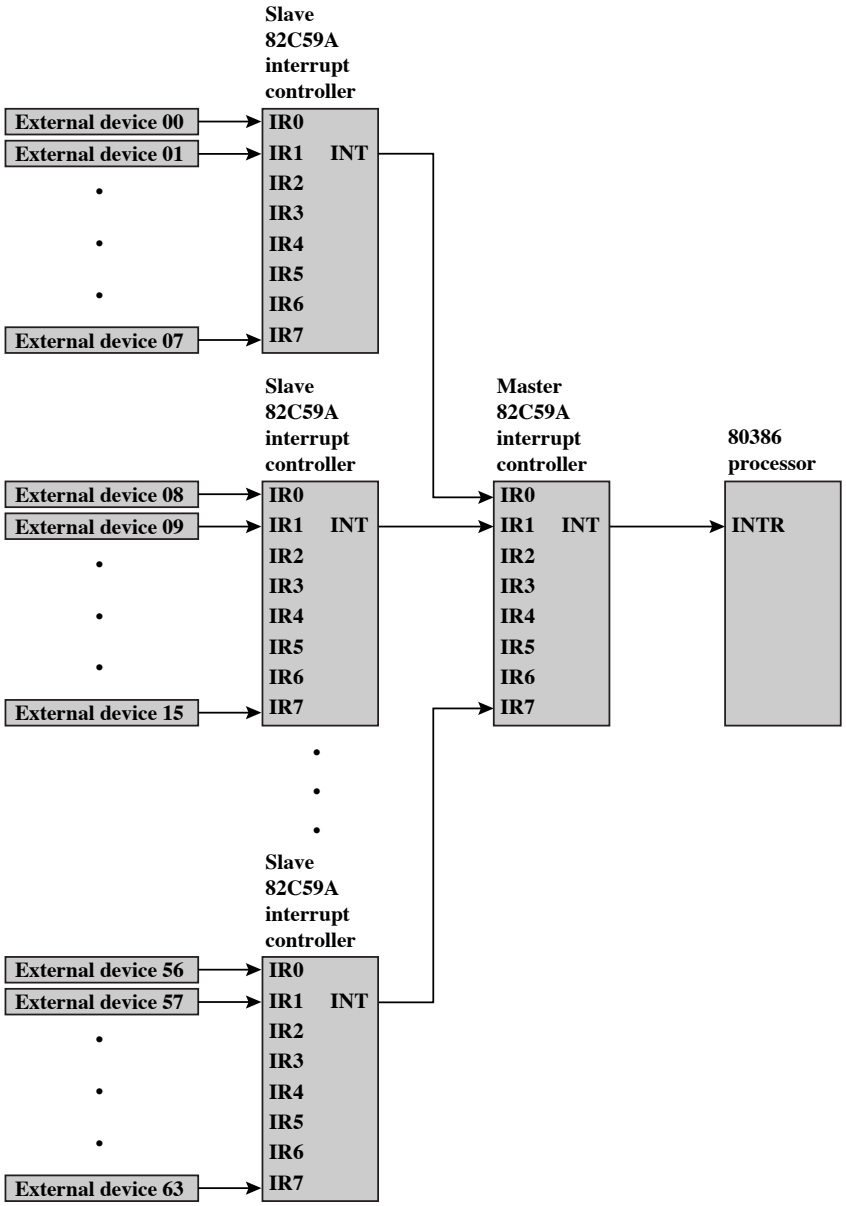


Figure 7.8 Use of the 82C59A Interrupt Controller

E/S con acceso directo a memoria (DMA)

- El uso de IRQS mejora la eficiencia de la E/S, pero todavía todos los datos transferidos en una operación tienen que pasar por la CPU, que además tiene que intervenir durante toda la transferencia.
- Inconvenientes de la E/S programada y con interrupciones:
  - La velocidad de transferencia de E/S está limitada por la velocidad con la cual el procesador puede comprobar y dar servicio a un dispositivo
  - El procesador está obligado a manejar directamente las operaciones de E/S.
- Se necesita una técnica más eficiente para atender a los dispositivos de E/S que demandan un mayor ancho de banda: acceso directo a memoria.
- La función de DMA es realizada por un dispositivo especializado que se va a encargar de transferir los datos entre el dispositivo de E/S y la memoria, sin necesidad de la intervención de la CPU. El diagrama de bloques de un módulo DMA típico es:

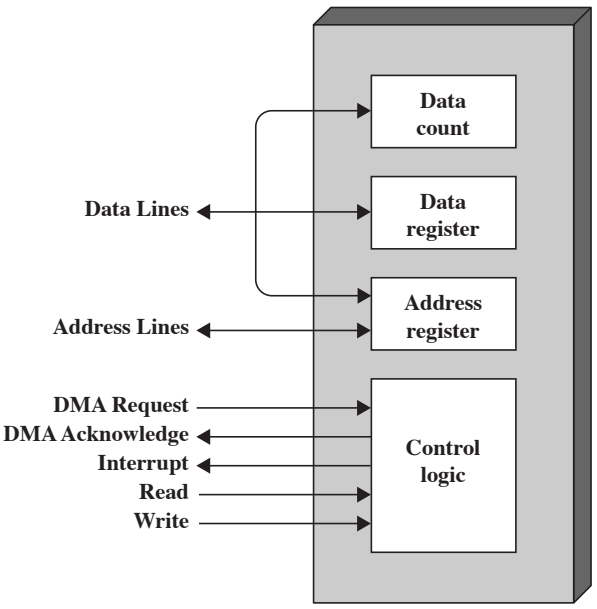


Figure 7.11 Typical DMA Block Diagram

- El controlador DMA puede actuar como maestro del bus, tarea en principio encomendada al procesador, y dirigir las lecturas y escrituras entre él mismo y la memoria.
- Opciones para el uso del bus por parte del controlador DMA:
  - utilizar el bus cuando el procesador no lo necesita
  - uso de memorias multipuerta: una puerta para CPU, resto E/S
  - es capaz de forzar al procesador a suspender temporalmente su operación (técnica más habitual y conocida como *robo de bus*)

- Pasos en una transferencia DMA:

  1. El procesador desea realizar una lectura o escritura en un dispositivo. La información proporcionada al DMA es la siguiente:
     - Activación de la línea de control que indica si se trata de lectura o escritura
     - Dirección del dispositivo de E/S involucrado en las líneas de datos
     - La dirección de comienzo en memoria principal en la que se va a iniciar la operación (lectura o escritura) en las líneas de dirección. La DMA guarda esta dirección en su propio registro de direcciones.
     - El número de palabras a leer o escribir, de nuevo en las líneas de datos. La DMA guarda esta valor en su registro de cuenta de datos.
  2. El procesador continua trabajando en otro proceso, delegando la operación de E/S al módulo DMA, que se encarga de realizar la transferencia directamente desde o a memoria, palabra por palabra.
  3. Cuando la operación ha sido completada, la DMA lanza una petición de interrupción al procesador. De esta forma, la CPU solo se ha visto involucrada en las etapas inicial y final de la transferencia.

- Posibles puntos de robo de ciclo durante la ejecución de una instrucción

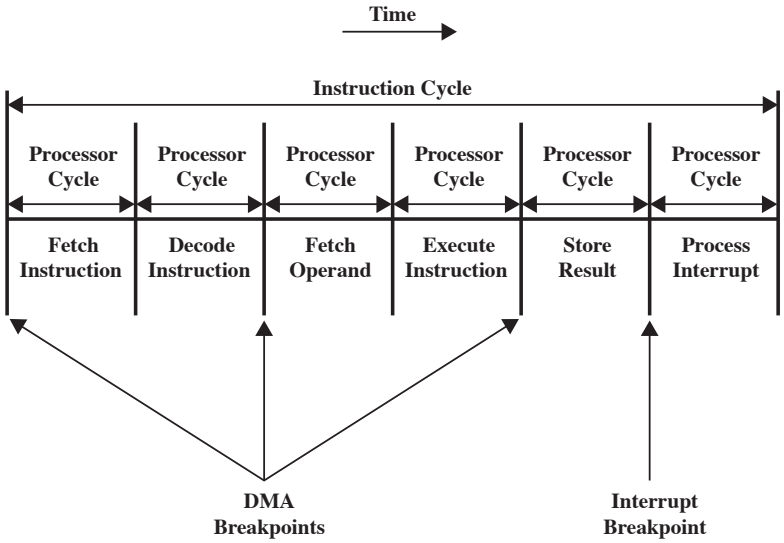


Figure 7.12 DMA and Interrupt Breakpoints During an Instruction Cycle

- Algunas configuraciones posibles de DMA:

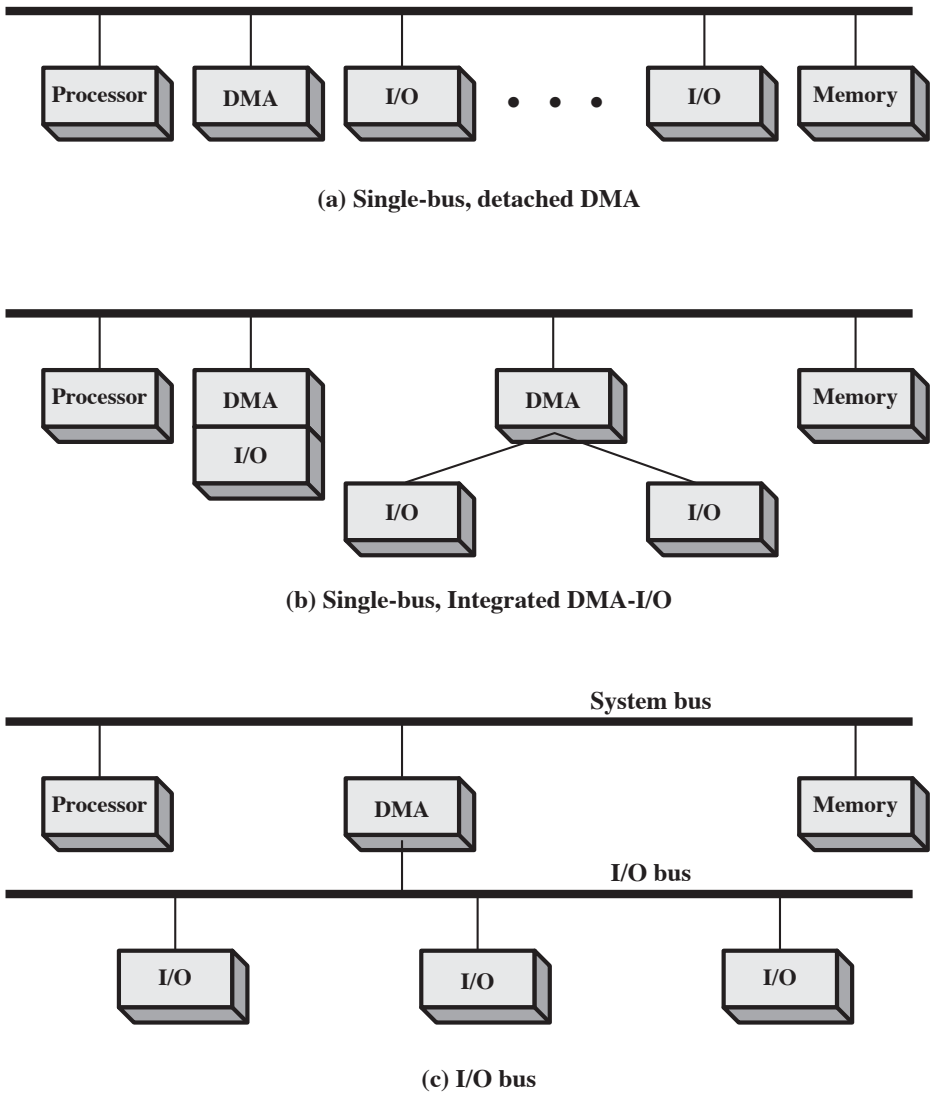


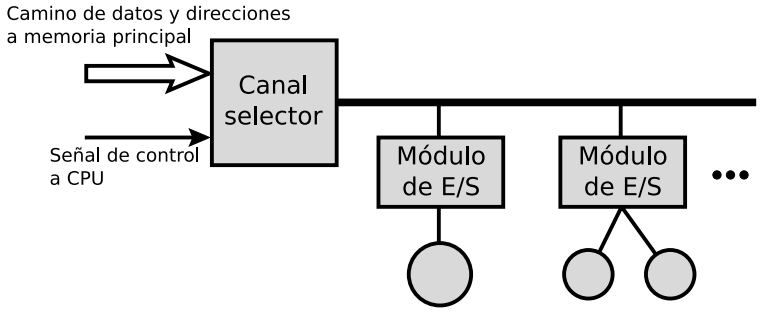
Figure 7.13 Alternative DMA Configurations

- Problemática de la incorporación del DMA
  - Memoria virtual: direcciones físicas y virtuales de las páginas de memoria. Soluciones:
    - que la DMA trabaje con direcciones virtuales
    - que el SO descomponga cada operación de DMA en transferencias más pequeñas
  - Coherencia cache: problema de datos obsoletos o de coherencia. Soluciones:
    - realizar las operaciones de E/S a través de la cache
    - que el SO invalide de forma selectiva el contenido de la cache antes de una operación de E/S de lectura (lectura de disp. E/S y escritura en memoria), y que fuerce la actualización de la mem. ppal. con el contenido de la cache (vaciado de cache o *cache flushing*) antes de una operación de E/S de escritura (escritura en disp. E/S y lectura de memoria).

Procesadores de E/S

- Procesador de E/S o canal: ampliación del concepto de DMA para reducir todavía más la interferencia de la E/S con la CPU.
- Tradicionalmente usado en grandes sistemas *mainframe*, su uso se ha extendido a otro tipo de sistemas como servidores de ficheros.
- Estos pequeños procesadores especializados ejecutan una serie de operaciones de E/S denominada programa de E/S, que normalmente coloca en la memoria principal del sistema el procesador central.
- Tipos de canales de E/S

1. Canal selector. Controla varios dispositivos de velocidad elevada. Solo puede transferir datos a/desde uno de los dispositivos conectados al canal en un determinado momento.



2. Canal multiplexor. Puede manejar la E/S de varios dispositivos lentos al mismo tiempo. Para dispositivos de velocidad muy reducida, un multiplexor de byte acepta o transmite caracteres tan rápido como es posible a varios dispositivos. Para dispositivos de velocidad algo más elevada, un multiplexor de bloque entrelaza bloques de datos de los distintos dispositivos.