

Sistemas Operativos. Tema 1

Arquitectura Básica de los Computadores

<http://www.ditec.um.es/so>

Departamento de Ingeniería y Tecnología de Computadores
Universidad de Murcia

Arquitectura Básica de los Computadores

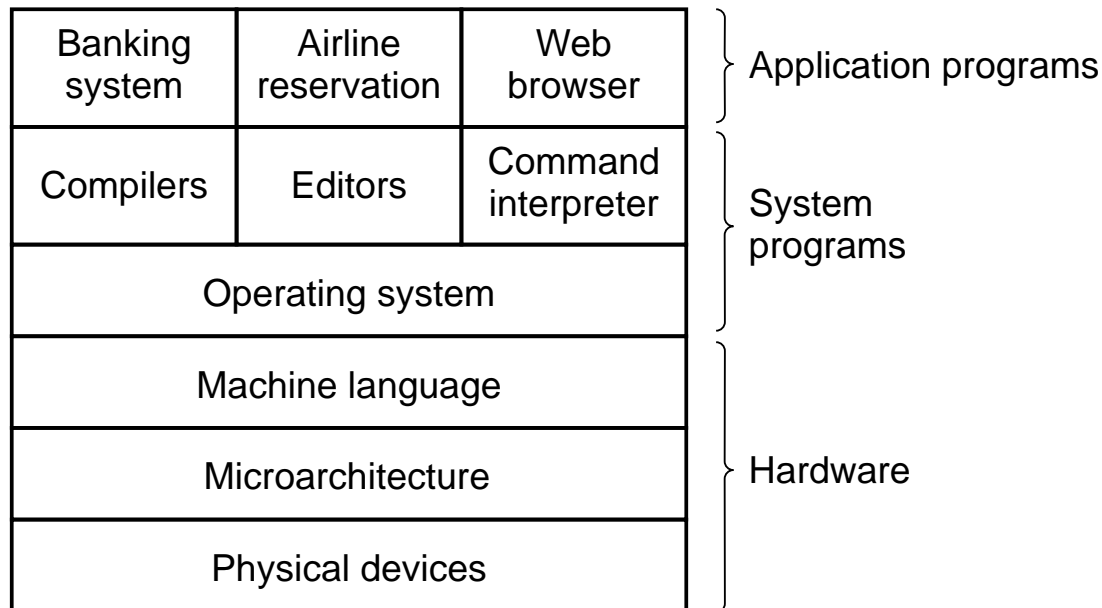
- Estructura y Funcionamiento General
- Procesador
- Memoria
- Entrada/Salida
- Interrupciones
- Protección

Bibliografía básica: Tanenbaum[P1-3, C1.4]

Bibliografía complementaria: Silberschatz[C2], Carretero[C1], Stallings[C1]

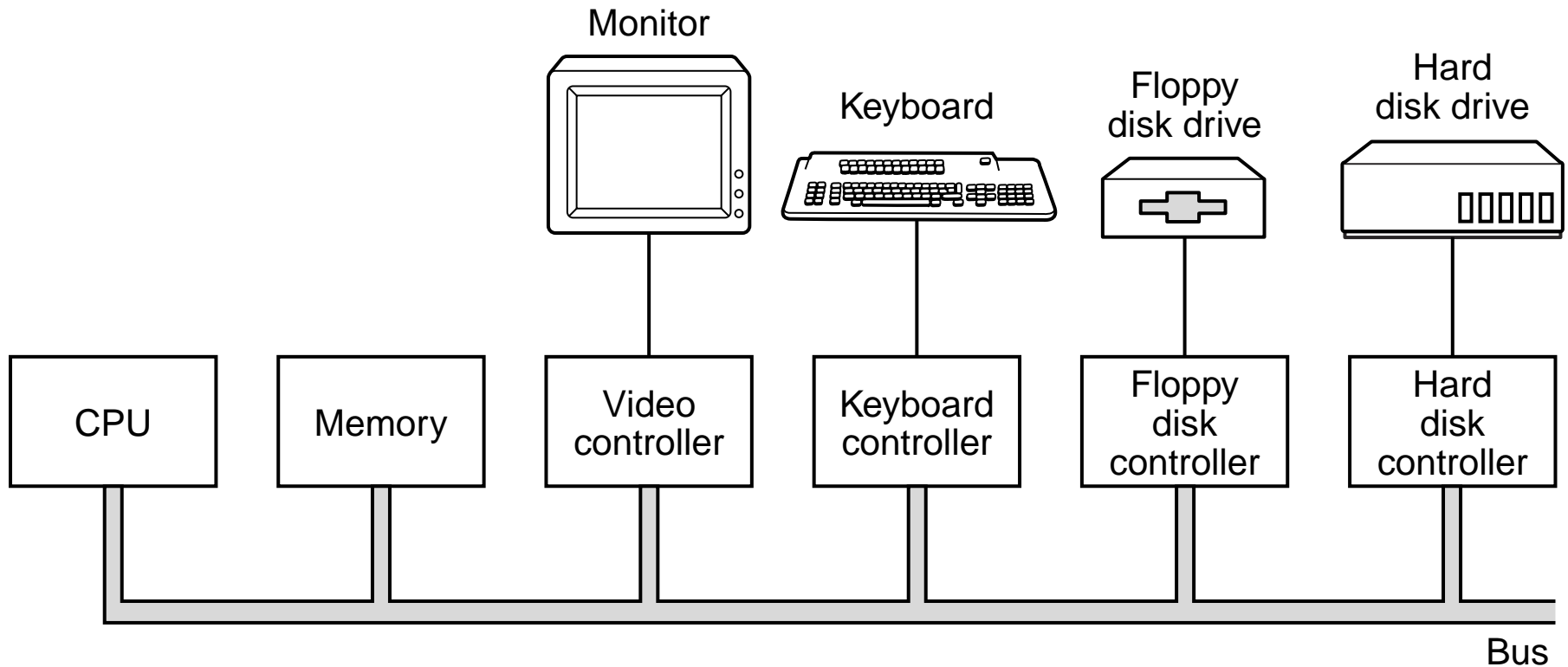
Estructura y Funcionamiento General

- Un sistema de cómputo moderno es un sistema complejo
- Para administrar todos estos dispositivos y proporcionar una interfaz sencilla del hardware \Rightarrow capa software: **sistema operativo**



Ubicación del sistema operativo

Estructura y Funcionamiento General



Visión simplificada de un ordenador

Procesador

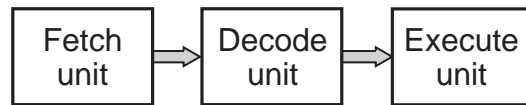
- La CPU es el «cerebro» del ordenador
- Ciclo básico de funcionamiento:
 1. Leer instrucción de memoria
 2. Decodificarla para determinar su tipo y operandos
 3. Ejecutarla
 4. Calcular la posición de la siguiente instrucción y volver al paso 1
- Cada CPU ejecuta un conjunto de instrucciones específico

Procesador

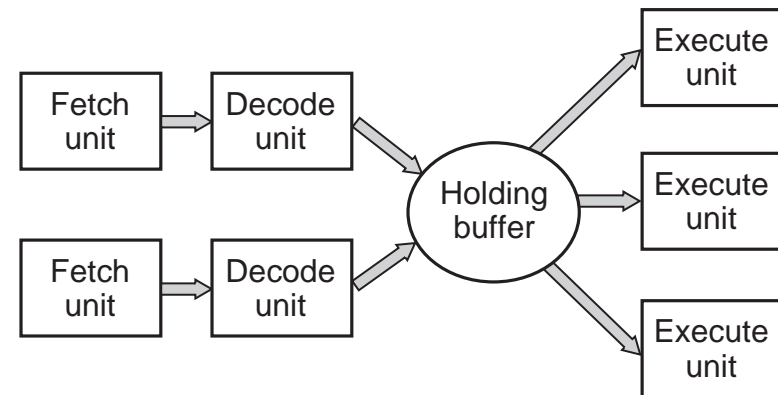
- Conjunto de registros: memoria en la propia CPU
- Registros generales de datos
- Registros especiales:
 - contador de programa
 - apuntador de pila
 - palabra de estado del programa
- El contenido de los registros determina el **contexto de ejecución de un programa** en un instante dado.

Procesador

- Para mejorar el desempeño de las CPUs \Rightarrow ejecutar varias instrucciones al mismo tiempo
- Varios mecanismos:



(a)



(b)

- Complican la construcción de compiladores y sistemas operativos

Procesadores

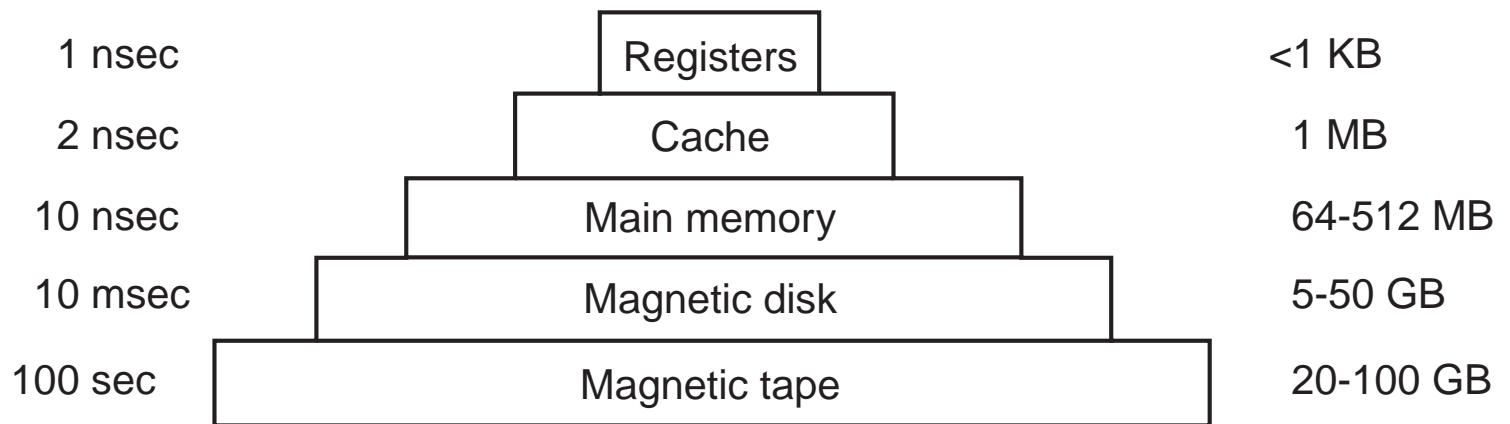
- Dos modos de funcionamiento: modos núcleo y usuario
- **Modo núcleo:**
 - Permite ejecutar todas las instrucciones posibles de la CPU y acceder a todo el hardware
 - En el que se ejecuta el **sistema operativo**
- **Modo usuario:**
 - Permite ejecutar un subconjunto de las instrucciones y proporciona acceso limitado al hardware
 - Instrucciones prohibidas: E/S, protección de memoria, etc
 - En el que se ejecutan los programas de usuario
 - Servicios del SO: mediante **llamadas al sistema**
- Paso de un modo a otro: **interrupciones** software (trap, int, ...) o hardware (división por cero, dispositivos de E/S)

Memoria

- Estructura jerárquica:

Typical access time

Typical capacity



- Cada nivel es un subconjunto del nivel inferior \Rightarrow Hay información que no se encuentra en un nivel \Rightarrow **Aciertos/fallos y algoritmos de reemplazo**
- Modificación en un nivel \Rightarrow **problemas de coherencia** \Rightarrow propagación de modificación a niveles inferiores

Jerarquía de Memoria

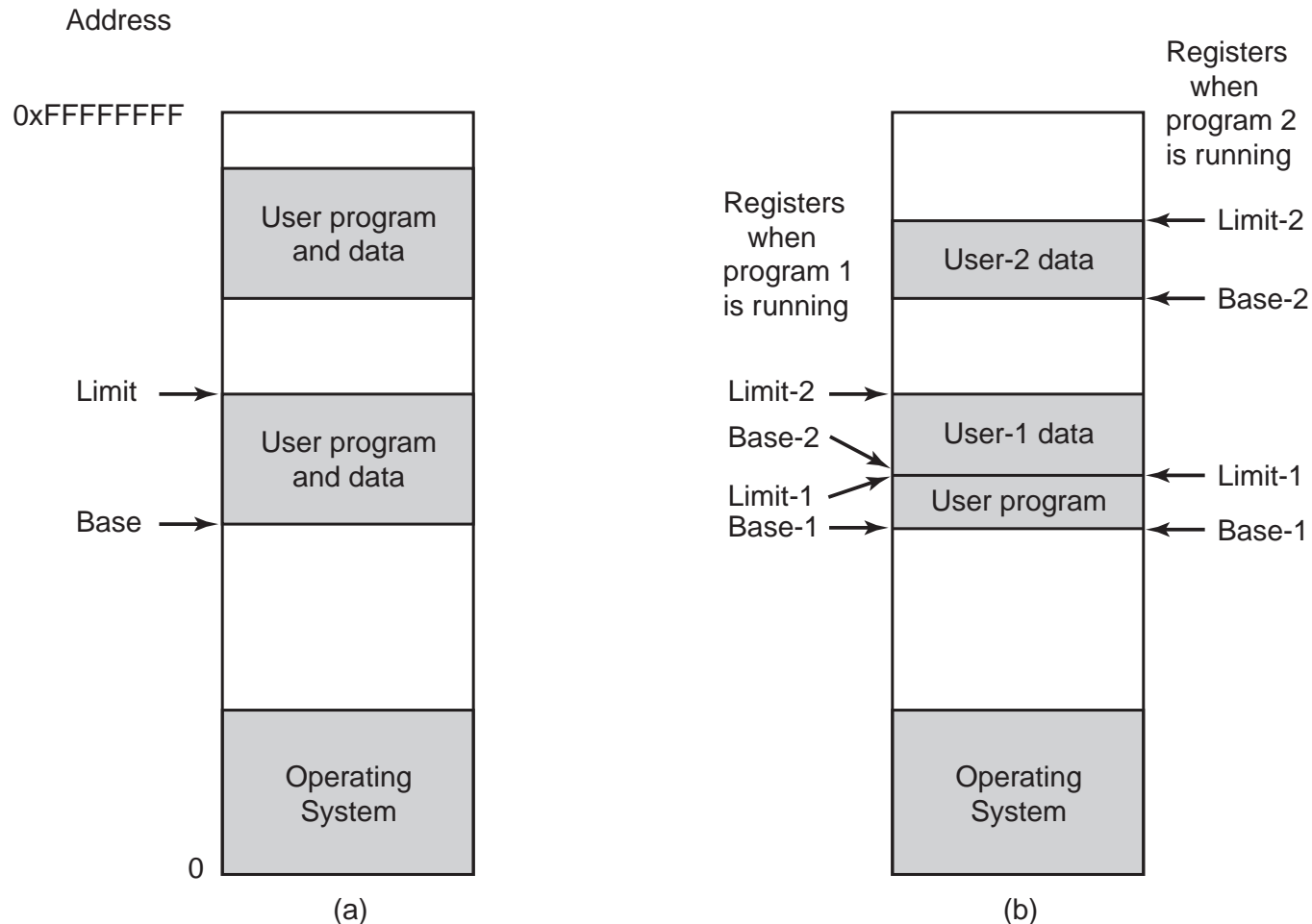
- La jerarquía descrita es típica, pero hay sistemas con más capas y otros con menos
- Otros tipos de memoria:
 - ROM: memoria lenta no volátil. Utilizada para almacenar código de arranque, código de control de dispositivos, etc.
 - EEPROM y *flash*: memorias lentas no volátiles pero actualizables
 - CMOS: memoria volátil alimentada por batería. Para mantener fecha y hora, y parámetros de configuración

Memoria Principal

- Elemento más importante de la jerarquía de memoria que debe administrar el SO
- Los SSOO modernos suelen cargar varios programas en memoria \Rightarrow Hay que **proteger** a unos programas de otros y al SO de éstos
- Además, un programa puede colocarse en cualquier posición de memoria \Rightarrow **Problema de relocalización**
- Varias soluciones para ambos problemas:
 - Registro base y límite
 - Memoria virtual, . . .

Registros Base y Límite

- **Registro límite** → tamaño máximo del programa y los datos
- **Registro base** → posición de inicio del programa en memoria



Dirección virtual/física

- ¿Dirección de memoria $<$ registro límite?
 - SI \rightarrow sumarle el registro base \rightarrow acceder a memoria
 - NO \rightarrow la dirección no es válida \rightarrow trap al S.O.
- **Dirección virtual** \rightarrow generada por el programa
- **Dirección física** \rightarrow accedida en memoria
- **MMU** \rightarrow Unidad de administración de memoria (*memory management unit*):
 - Verifica las direcciones generadas por el programa
 - **Convierte las direcciones virtuales en físicas**
- El manejo de la MMU es función del S.O.
- Cambio de contexto \rightarrow modificar la configuración de la MMU

Dispositivos de E/S

- Dispositivo de E/S → controladora + dispositivo
- **Controladora del dispositivo:**
 - Dispositivo electrónico que controla físicamente al dispositivo
 - Acepta comandos del S.O. y los ejecuta
 - Presenta al S.O. una interfaz más sencilla del dispositivo
 - Tiene una serie de registros para comunicarse con el S.O.
- **Manejador de dispositivo** → SW que se comunica con la controladora: da órdenes y procesa respuestas
 - Se ejecuta en modo kernel
 - Como parte del kernel
 - En tiempo de arranque lo carga el S.O.
 - El S.O. lo carga cuando lo necesita (sin reiniciar)

Dispositivos de E/S

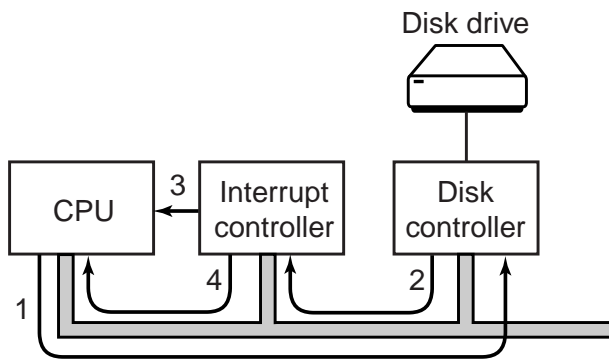
- Manejador recibe una petición del S.O.:
 - escribe la petición en los registros de la controladora
 - lee de los registros el resultado de la operación
- ¿Cómo se accede a esos registros?
 - Se corresponden con el **espacio de direcciones** de la memoria principal:
 - Se leen/escriben como si fueran palabras de memoria
 - No se necesitan instrucciones especiales de E/S
 - Fácil protección: protección memoria → protección HW
 - Se colocan en un **espacio de puertos de E/S especial**:
 - Cada registro tiene una dirección de puerto.
 - Los registros se leen/escriben mediante instrucciones IN y OUT especiales ejecutables en modo kernel
 - No se ocupa parte del espacio de direcciones de la memoria

Espera Activa

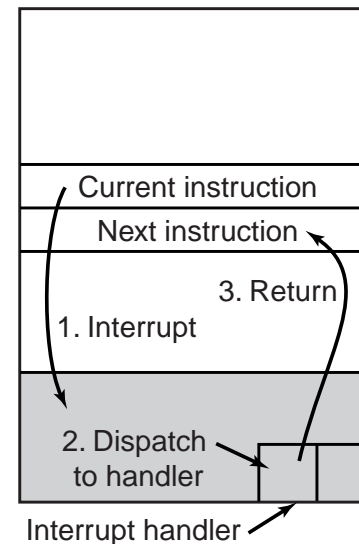
- Llamada al sistema → llamada al manejador del dispositivo
- Manejador pide a la controladora la operación de E/S
- Manejador a la controladora: ¿ha terminado mi petición?
 - NO → continúa esperando
 - Sí:
 - Manejador de dispositivo coloca los datos donde se necesitan
 - Regresa
 - S.O. devuelve el control al proceso invocador
- Esto se conoce como **espera activa**
- CPU ocupada: controlando cuándo termina el dispositivo
- Desperdicio de CPU (tanto de ciclos como de energía)

Interrupciones

- **Manejador de dispositivo:**
 - pide a la controladora la operación de E/S
 - le pide también generar una **INTERRUPCIÓN** al terminar
 - regresa
- **S.O.:**
 - bloquea al proceso invocador (no le pasa la CPU)
 - hace otras cosas
- Al finalizar la transferencia → **interrupción** generada por la controladora



(a)



(b)

Interrupciones (i)

- CPU acepta la interrupción → pasa a modo kernel y salta al manejador de interrupciones del dispositivo
- N° de dispositivo → índice de una zona de memoria (**vector de interrupciones**) que contiene las direcciones de los manejadores de interrupciones
- El manejador de interrupciones:
 - Pregunta al dispositivo su estado
 - Cuando termina devuelve el control al programa de usuario que se estaba ejecutando (que **no tiene por qué ser el que solicitó la operación de E/S**)

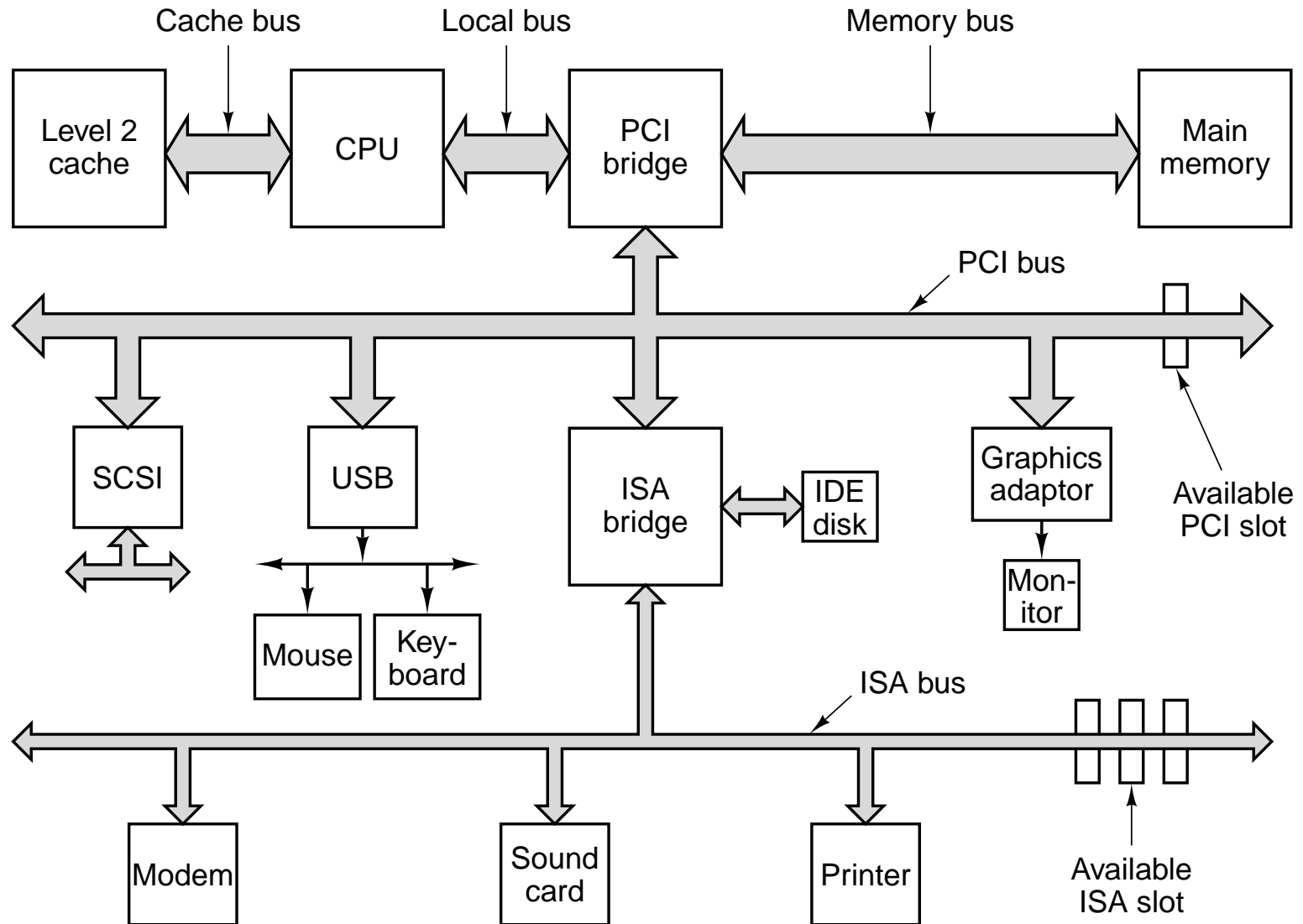
Interrupciones (ii)

- ¿Qué pasa si llega una interrupción mientras se trata otra? ¿Paramos la interrupción actual? ¿Perdemos la nueva?
- Mecanismo en la CPU para inhabilitar interrupciones:
 - Empieza a tratar una interrupción → inhabilita las interrupciones
 - Si llega una nueva → dispositivo seguirá aplicando la interrupción
 - CPU no se interrumpe mientras estén inhabilitadas
 - Termina tratamiento → habilita de nuevo las interrupciones
 - CPU se puede volver a interrumpir
- ¿Qué pasa si varios dispositivos esperan a la CPU para su interrupción?
 - Controladora de interrupciones decide a cuál atender primero
 - Prioridades estáticas asignadas a los dispositivos

DMA: Acceso Directo a Memoria

- Chip que controla el flujo de bits entre la memoria principal y alguna controladora
- CPU no interviene en las transferencias *memoria ↔ controladora*
- También basado en interrupciones
- Funcionamiento:
 - El S.O. localiza un *buffer de memoria*
 - Programa la controladora DMA y la controladora específica
 - El DMA realiza la transferencia *buffer de la controladora ↔ buffer de memoria*
 - Controladora de DMA avisa a la CPU mediante una interrupción
 - CPU trata esa interrupción
- Dispositivos muy rápidos (discos duros, tarjetas de sonido, ...)

Buses



Protección

Varios procesos en ejecución \Rightarrow Protección

- Protección del HW \Rightarrow Operación en modo dual:
 - Modo núcleo: instrucciones de E/S, configuración MMU, ...
 - Modo usuario: instrucciones de acceso al HW ilegales
- Protección de memoria \Rightarrow Registros base y límite, ...:
 - Protege a unos programas de otros y al S.O.
 - Protege el vector de interrupciones
- Protección de la CPU \Rightarrow Interrupciones periódicas (cronómetros o relojes, ...):
 - Interrupciones \rightarrow ejecución del sistema operativo
 - Impiden que un proceso se apropie de la CPU
- Protección total del S.O. y el HW \Rightarrow **uso simultáneo de los tres mecanismos**