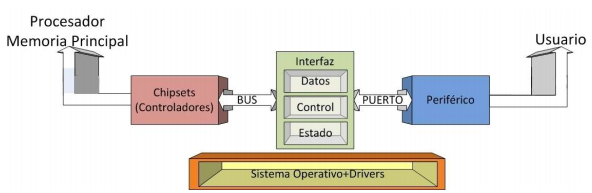
[Resumen clase 3](https://drive.google.com/file/d/1Rtr5HCTEjKobz53ojrqh49b7P0PE7CXr/view?usp=sharing)

# Problemas de E/S

* Gran variedad de periféricos con varios métodos de operación
  + Transmisión de diferentes cantidades de datos
  + A diferentes velocidades
  + Usan diferentes formatos de dato y tamaño de palabra
* Todos más lentos que la CPU y la RAM
* Necesidad de módulos de E/S (con alguna “inteligencia”)

# Módulo de E/S

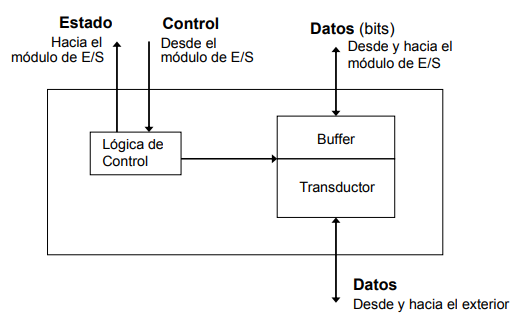
* Realiza la interfaz entre el procesador y la memoria (bus) y los periféricos
* Pueden manejar uno o más periféricos



# Dispositivos externos

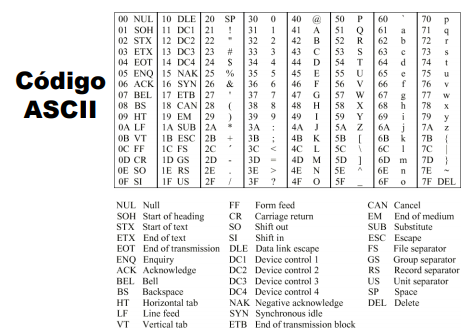
* E/S básicos (monitor, mouse, teclado)
* Almacenamiento (disco duro, cd, dvd)
* Impresión (impresora, escáner)
* Comunicación con dispositivos remotos (módem, acceso/interfaz de red)
* Multimedia (micrófono, parlantes)
* Automatización y control (sensores, alarmas, adquisición de datos)

Dispositivo externo tipo:



# Características de un puerto

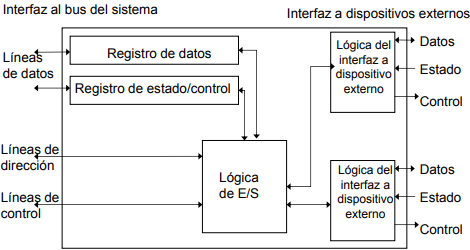
* Interfase entre el periférico y el módulo de E/S
* Señales de control, estado y datos
  + Señal de control: función a realizar (Ej: read, write)
  + Señal de estado: ready/not ready
  + Control lógico: manejo de direccionamiento
  + Transductor: conversión del datos
  + Buffer: adaptación (1, 8 o 16 bits)



# Funciones de un módulo de E/S

* Control y temporización de uno o más dispositivos externos
* Interpretar las órdenes que recibe de CPU y transmitirlas al periférico
* Comunicación con la CPU (registros) y memoria
* Controlar las transferencias de datos entre CPU y el periférico (convertir formatos, adaptar velocidades)
* Comunicación con los dispositivos (periféricos)
* Informar a la CPU del estado del periférico
* Almacenamiento temporal (buffering) de datos
* Detección de errores

Diagrama de bloques de un módulo de E/S



# Capacidades de un módulo de E/S

* Ocultar las propiedades del dispositivo a la CPU (ej: temporizados, formatos, electromecanismos, etc)
* Ocuparse de uno o varios dispositivos
* Controlar o no las funciones del dispositivio
  + Canales de E/S o procesador de E/S (manejo de parte importante de la carga del procesamiento). Presentes en Mainframes.
  + Controlador de E/S o controlador de dispositivo (manejo primitivo). Presentes en microcomputadoras.

# Operación de E/S

Requiere:

* Direccionamiento
  + E/S mapeada en memoria
  + E/S aislada
* Transferencia de información (lectura o escritura)
* Gestión de la transferencia
  + Mecanismos de sincronización y control de la transferencia de datos

## Direccionamiento de E/S

* E/S asignada en memoria (memory-mapped)
  + Dispositivos de E/S y memoria comparten un único espacio de direcciones
  + E/S se parece a la memoria de lectura/escritura
  + No hay ordenes específicas para E/S
    - Variedad de órdenes de acceso a memoria (programación eficiente)
* E/S aislada
  + Espacios de direcciones separados
  + Necesidad de líneas especiales de E/S y de memoria
  + Órdenes específicas para E/S
    - Conjunto limitado de instrucciones

## Técnicas de gestión de E/S

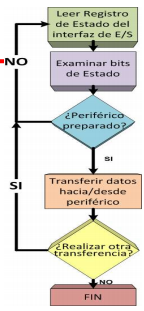
* E/S Programada con espera de respuesta
* E/S con interrupciones
* E/S con acceso directo a memoria (DMA)

### E/S programada

* Intercambio de datos entre la CPU y el módulo
* La CPU tiene control directo sobre la operación de E/S
  + Comprobación del estado del dispositivo
  + Envío de comandos de lectura/escritura
  + Transferencia de datos
* La CPU espera que el modulo de E/S termine la operación
* Por lo tanto la CPU permanece ociosa durante un período de tiempo (no deseable)

Detalles de la E/S programada:

* La CPU solicita la operación de E/S al módulo
* El módulo E/S realiza la operación
* El módulo E/S activa los bits de estado del dispositivo direccionado y espera
* La CPU comprueba periódicamente el estado de esos bita, hasta que detecta que la operación fue completada
* En caso contrario, la CPU espera y vuelve a comprobarlo más tarde



#### Órdenes (comandos) de E/S

* La CPU emite una dirección: Específica el módulo (y el dispositivo si hay más de uno por módulo)
* La CPU da una orden:
  + Control: indica al módulo qué hacer (ej: rebobinar una cinta magnética)
  + Test: comprueba el estado del módulo y sus periféricos (ej: está conectado? Hubo algun error?)
  + Lectura/escritura: transfiere datos desde o hacia el dispositivo por el bus de datos

### E/S con interrupciones

* La CPU no tiene que esperar la finalización de la tarea de E/S, puede seguir procesando
* No se repite la comprobación de los estados de los módulos
* El modulo envía un pedido de interrupción a la CPU cuando está listo nuevamente

Detalle de la CPU:

* La CPU envía una orden de lectura (READ): El módulo E/S obtiene los datos del periférico mientras que la CPU realiza otro trabajo
* La CPU chequea si hay pedidos de interrupciones pendientes al final de cada ciclo de instrucción: El módulo E/S emite un pedido de interrupción a la CPU
* La CPU detecta el pedido, guarda el contexto, interrumpe el proceso y realiza la gestión de la interrupción.
* La CPU solicita los datos: El módulo E/S transfiere los datos

Cuestiones de diseño:

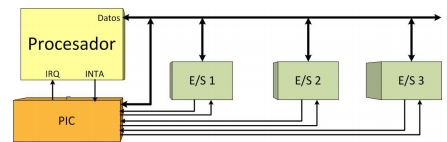
* ¿Cómo saber qué dispositivo ha provocado la interrupción?
* Con múltiples interrupciones, ¿cómo elegir la interrupción que se debe atender? Establecemos prioridades?
  + Una rutina de interrupción que a su vez es interrumpida

#### Identificación del módulo que interrumpe

* Diferentes líneas para cada módulo
  + PC
  + Limita el número de dispositivos
* Consulta software (Poll o encuesta)
  + Ocurrido un pedido de interrupción la CPU consulta a cada módulo para determinar quién fue el demandante
  + Resulta lento
* Conexión en cadena (daisy chain) “hard poll”
  + La línea de reconocimiento de interrupción se conecta encadenando los módulos, la línea de pedido es compartida
  + Una vez enviada la confirmación de parte de la CPU el módulo responderá colocando un vector (palabra), en el bus, que lo identifica
  + La CPU emplea el vector como puntero para acceder a la rutina de servicio

#### Interrupciones múltiples

* Todas las líneas de interrupción tienen un orden de prioridad
* Las líneas con mas prioridad pueden interrumpir a las líneas con menor prioridad
* Si existe un maestro del bus, solo él puede interrumpir

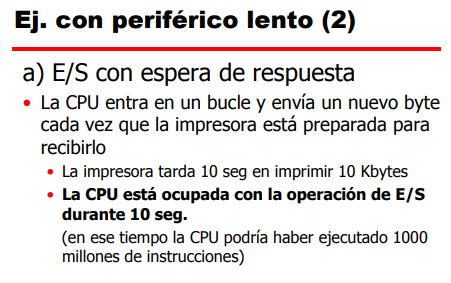
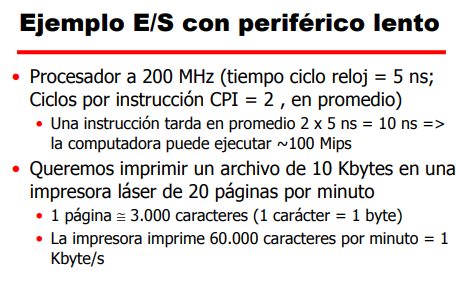


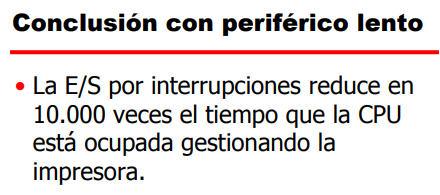
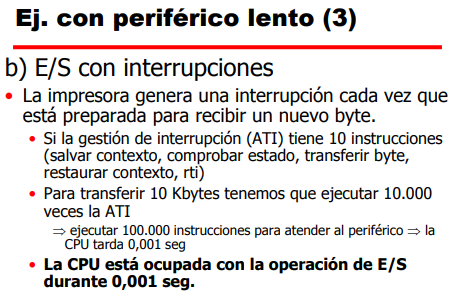
#### Estructuras de interrupciones

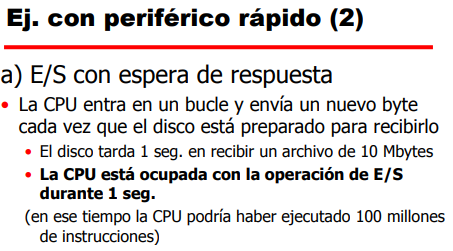
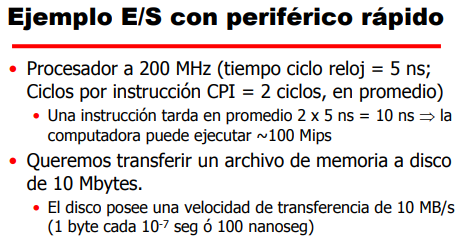
* En Intel 8086 tiene solo una línea de petición de interrupción (INTReq) y por lo tanto una sola de confirmación (INTAck)
* Se debería utilizar un árbitro o gestor de interrupciones externo, el 8259A (PIC)
* Este chip tiene 8 líneas de interrupción, por lo tanto podrá manejar 8 módulos de E/S
  + Usando conexión en casacada se puede gestionar hasta 64 módulos
* Puede ser útil tener una Interfase de Perifericos Programable (PIO)
* El Intel 8255A es un chip para usar también en entorno 8086
* Es un módulo de E/S de propósito general
* Posee 24 líneas de E/S programable vía los registros de control
* Usado para una variedad de periféricos (Ej: teclado/pantalla)

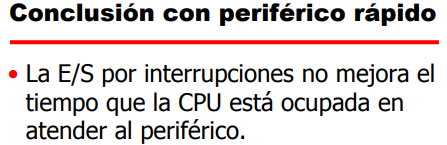
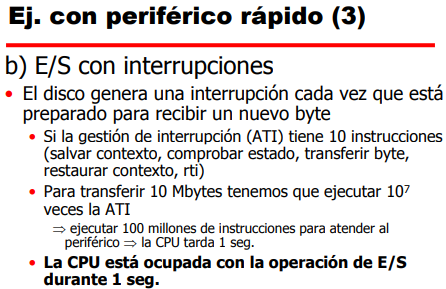
#### Análisis

* Las operaciones de E/S mediante interrupciones son más efectivas que las programadas
* Pero ambas necesitan la intervención directa de la CPU
  + La velocidad de transferencia es limitada
  + La CPU permanece ocupada mucho tiempo durante la operación
* Qué sucede si el volumen a transferir es grande?



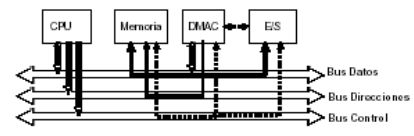






### Acceso directo a memoria (DMA)

* El controlador de DMA es un dispositivo capaz de controlar una transferencia de datos entre un periférico y memoria sin intervención de la CPU

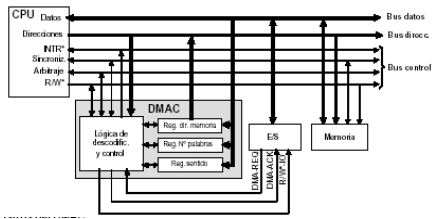


#### Controlador de DMA (DMAC)

El controlador de DMA (DMAC) debe actuar como maestro del bus durante la transferencia DMA y debe ser capaz de:

* Solicitar el uso del bus mediante las señales y la lógica de arbitraje necesarias
* Especificar la dirección de memoria sobre la que se realiza la transferencia
* Generar las señales de control del bus
  + Tipo de operación (lectura/escritura)
  + Señales de sincronización de la transferencia

Estructura de un DMAC:



#### Etapas de una transferencia DM

**Inicialización de la transferencia**: La CPU debe enviar al interfaz del periférico y al DMAC los parámetros de la transferencia

* Inicialización del interfaz (Bus master: CPU-Bus slave: Interfaz)
  + N° de bytes a transferir
  + Tipo de transferencia (lectura/escritura)
  + Otra información de control (pista, sector, etc)
* Inicialización controlador DMA (Bus master: CPU-Bus slave: DMAC)
  + N° de bytes o palabras a transferir
  + Tipo de transferencia (lectura/escritura)
  + Dirección de memoria inicial para la transferencia
  + N° de canal (para DMAs con varios canales)

Después de la inicialización la CPU retoma a sus tareas y ya no se preocupa más de la evolución de la transferencia

**Realización de la transferencia**:

* Cuando el periférico esta listo para realizar la transferencia se lo indica al DMAC
* El DMAC pide el control del bus y se realiza la transferencia entre el periférico y la memoria
  + Bus master: DMAC + periférico - Bus Slave: memoria
  + Después de la transferencia de cada palabra se actualizan los registros del DMAC
    - N° de bytes o palabras a transferir
    - Dirección de memoria

**Finalización de la transferencia**

* El DMAC libera el bus y devuelve el control a la CPU
* El DMAC suele activar una señal de interrupción para indicar a la CPU la finalización de la operación de E/S solicitada

#### Bus del sistema y DMA

#### 

#### Problema que puede haber

* Se puede degradar el rendimiento de la CPU si el DMAC hace uso intensivo del bus
  + Si el bus está ocupado en una transferencia DMA, la CPU no puede acceder a memoria para leer instrucc./datos
* El problema se reduce con el uso de memoria cache
  + La mayor parte del tiempo, la CPU lee instrucciones de la cache, por lo que no necesita usar el bus de memoria
  + El DMAC puede aprovechar estos intervalos en los que la CPU está leyendo instrucciones de la caché (y por lo tanto no usa el bus de memoria) para realizar transferencias
* En caso de computadoras sin cache
  + El procesador no utiliza el bus en todas las fases de la ejecución de una instrucción
  + El DMAC puede aprovechar las fases de ejecución de una instrucción en las que la CPU no utiliza el bus para realizar sus transferencias

### Tipos de transferencias

* Si el DMAC sólo toma el control del bus durante los intervalos de tiempo en los que la CPU no hace uso del mismo **el rendimiento del sistema no sufrirá degradación alguna**
* Se distinguen dos tipos de transferencias:
  + Por ráfagas (burst)
  + Por robo de ciclo (cycle-stealing)

#### DMA modo ráfaga

* El DMAC solicita el control del bus a la CPU
* Cuando la CPU concede el bus, el DMAC no lo libera hasta haber finalizado la transferencia de todo el bloque de datos completo
* Ventajas: la transferencia se realiza de forma rápida
* Desventajas: Durante el tiempo que dura la transferencia la CPU no puede usar el bus con memoria, lo que puede degradar el rendimiento del sistema

#### DMA modo robo de ciclo

* El DMAC solicita el control del bus a la CPU
* Cuando la CPU concede el bus al DMAC, se realiza la transferencia de una única palabra y después el DMAC libera el bus
* El DMAC solicita el control del bus tantas veces como sea necesario hasta finalizar la transferencia del bloque completo
* Ventajas: No se degrada el rendimiento del sistema
* Desventajas: La transferencia tarda más en llevarse a cabo
* Para la CPU no es una interrupción: el procesador NO debe guardar el contexto
* Si bien el trabajo de la CPU es lento, no será tanto como si ella realizará la transferencia
* Por lo tanto, para transferencias de E/S de múltiples palabras, es la técnica más eficiente

### Canales de E/S

Los dispositivos de E/S son cada vez más sofisticados (Ej: tarjetas gráficas 3D)

Evolución:

1. La CPU controla directamente los periféricos
2. Se agrega un módulo de E/S o controlador
3. Idem 2 más llamado de interrupción
4. El módulo de E/S provee el acceso directo a memoria (DMA)
5. El módulo de E/S tiene su propio procesador con su pequeño conjunto de instrucciones
6. El módulo además tiene su memoria local o sea se convierte en una computadora en sí mismo

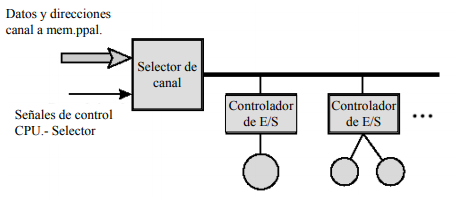
#### Características de canales de E/S

* Representan una extensión al concepto de DMA: tienen la habilidad de ejecutar instrucciones de E/S
* Completo control de la transferencia de datos: por lo tanto la CPU no ejecuta instrucciones de E/S
* Programa almacenado en memoria principal
* La CPU inicia la transferencia de E/S
  + Ordena ejecutar el programa que está en memoria
  + El programa especifica dispositivos, áreas de memoria a usar, prioridades y acciones ante errores

#### Tipos de canales de E/S

##### Selector

* Controla varios dispositivos de alta velocidad y uno por vez, por lo tanto el canal se dedica para la transferencia de datos de ese dispositivo
* El canal selecciona un dispositivo y efectúa la transferencia
* Los dispositivos son manejados por un controlador o módulo de E/S
* Por lo tanto el canal de E/S ocupa el lugar de la CPU en el control de esos controladores



##### Multiplexor

* Puede manejar E/S con varios dispositivos a la vez
* Multiplexor de bytes: Acepta y transmite caracteres
* Multiplexor de bloques: Intercala bloques de datos desde distintos dispositivos

##### 

