# Introducción a los sistemas de entrada/salida
## Profesores y tutorías

- Teoría:
  - Daniel Cascado Caballero

    Despacho: F070

    Horario de tutorías:

    Lunes: 17:30h a 19:30h
    Martes: 12:30h a 13:30h
    Jueves: 10:30h a 12:00h
    Viernes: 12:30h a 14:00h

  - Lourdes Miró Amarante

    Despacho: F061

    Horario de tutorías:

    Martes: 10:00h a 13:00h
    Jueves: 17:00h a 20:00h



# Introducción a los sistemas de entrada/salida
## OBJETIVOS

- Describir los ***mecanismos*** básicos de programación ***de entrada/salida***:
  - Entrada/salida programada o por encuesta (*polling*).
  - Por *interrupciones*.
  - Mediante DMA (*Direct Memory Access*).
- Describir varios ***dispositivos de entrada/salida básicos:***
  - Teclado y temporizador (i8254) → Práctica de E/S.
  - Puerto paralelo (interfaz *centronics*).
  - PPI (*Programmable Peripheral Interface*) i8255.
  - PIC (*Programmable Interrupt Controller*) i8259.

# Introducción a los sistemas de entrada/salida
## ÍNDICE





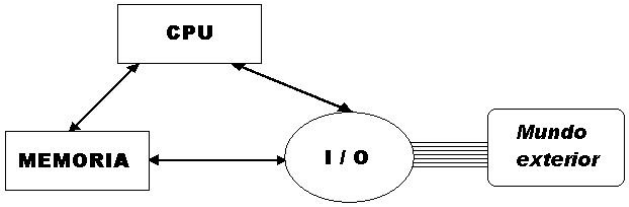
# Introducción a los sistemas de entrada/salida
## BIBLIOGRAFÍA


– *Estructura y Diseño de Computadores,* J.L. Hennessy y D. A. Patterson. Ed. Reverte, 2000.

– *Organización y Arquitectura de Computadores,* W. Stalling. Prentice-Hall, 2000.

# 1. INTRODUCCIÓN (I)

- Un computador no puede estar formado sólo por la CPU y la memoria.
- Para darle alguna utilidad debe de **comunicarse con el mundo exterior** a través del subsistema de entrada/salida (I/O *input/output*).






# 1. INTRODUCCIÓN (II)

La misión principal del subsistema de E/S es **adaptar los dispositivos externos** antes de conectarlos al bus del sistema.

**¿Por qué no se conectan directamente al bus del sistema?**

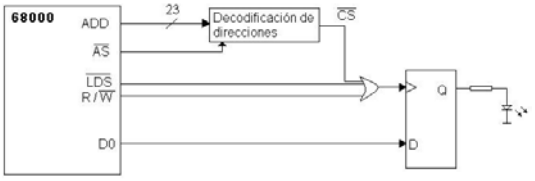
1. La velocidad de transmisión de datos de los periféricos es siempre menor que la de la memoria y la CPU.
2. Debido a la gran diversidad de periféricos no es posible incorporar toda la lógica necesaria en el computador para manejar cada uno de éstos.
3. Los formatos de datos de los periféricos son diferentes a los del resto del computador.

# 1. INTRODUCCIÓN (III)

- Hay dos **formas de implementar la E/S** en un computador:

  1. Construir a medida el subsistema utilizando elementos básicos
  2. o bien incorporar y programar dispositivos estándares más complejos y genéricos.

*Ejemplo primer caso: circuito para que al escribir un uno en una dirección de memoria se ilumine un LED.*






# 1. INTRODUCCIÓN (IV)

Funciones del sistema de E/S:

1. DIRECCIONAMIENTO: selección del dispositivo correspondiente de entre los dispositivos disponibles en el sistema.
2. SINCRONIZACIÓN: ha de posibilitar que la CPU y la memoria (alta velocidad transferencia de datos) se puedan comunicar con los dispositivos de E/S (baja velocidad) sincronizando los envíos de datos entre ambos.
3. TRANSFERENCIA: el sistema E/S debe de tener toda la circuitería y señales de comunicación apropiadas para llevar a cabo la comunicación con cada uno de los dispositivos del sistema.

# 2. MÓDULOS DE E/S:
## Comunicación CPU – Periférico

La E/S se implementa mediante periféricos.

- **PERIFÉRICO**:
  - Elemento que permiten la transferencia de información entre la CPU y el mundo exterior.
  - Interfaz que traduce la información asíncrona y analógica del mundo exterior a la información síncrona y codificada del computador.
  - Dos partes: **módulo de E/S y dispositivo (externo).**

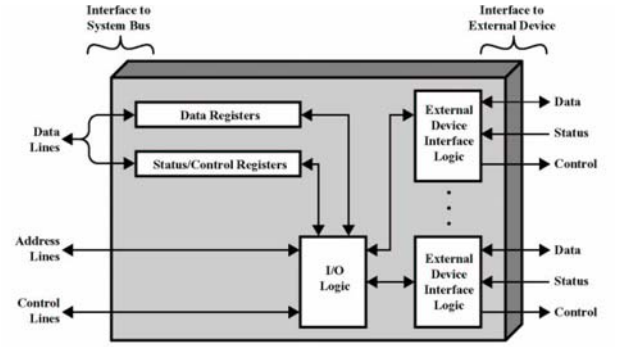


# Módulo de E/S (I)

- **MÓDULO DE E/S**:
  - Coordina el correcto flujo de información entre uno o varios dispositivos externos (impresora, monitor, ...) e internos (memoria, procesador).
  - FUNCIONES:
    1. Reconocer la dirección de la CPU que identifica al dispositivo externo.
    2. Transferencia de datos entre la CPU y el dispositivo externo.
    3. Recepción de mandatos (comandos) desde la CPU.
    4. Mantener información del estado del periférico y mantener el protocolo de comunicaciones con el periférico.
  - Un módulo de E/S puede controlar varios dispositivos externos.

# Módulo de E/S (II)

- El módulo de E/S también almacena datos temporalmente debido a las diferencias de velocidades entre los periféricos y la CPU o la memoria. Dispone de un mecanismo de detección de errores tales como el uso del *bit* de paridad.
- Los términos "controlador", "procesador de E/S" y "módulo de E/S" son equivalentes. La diferencia radica en su complejidad.






# Dispositivo Externo

- **DISPOSITIVO EXTERNO**:
  La forma de comunicación con el módulo de E/S se hace mediante:
  - SEÑALES DE CONTROL (mandatos/comandos enviados por la CPU),
  - ESTADO (información sobre el funcionamiento del dispositivo)
  - y DATOS.

  Internamente suele haber una serie de *buffers* que son capaces de almacenar temporalmente datos o información adicional de contexto.

  Un componente importante suele ser el transductor que transforma la información analógica en digital.


Fig. Periférico genérico

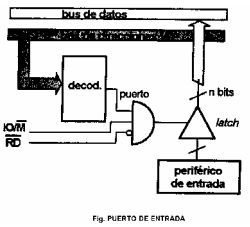



Fig. PUERTO DE ENTRADA

# 3. MAPA DE E/S:
## E/S común y separada

Según el modo de seleccionar el periférico y el acceso a sus registros de control, datos y estado:

1. **E/S COMÚN O ASIGNADA/*MAPEADA* EN MEMORIA**:
   - El acceso a estos módulos se realiza de igual modo a como se accede a un dato de memoria principal.
   - Los periféricos se integran en el computador como si fueran parte de la memoria → Comunicarse con un módulo de E/S es leer y escribir en memoria. Ej: 68000.
   - VENTAJA: Se aprovecha la potencia del juego de instrucciones.
   - INCONVENIENTE: Se desperdicia parte del espacio de direcciones.

2. **E/S AISLADA O SEPARADA**:
   - El acceso a la E/S está contemplado en la arquitectura.
   - Existen dos mapas de memoria separados: uno para memoria y otro para E/S → Existen señales e instrucciones específicas. Ej: intel 80x86.
   - (Las ventajas y desventajas son contrarias a las de la E/S común.)



# 3. MAPA DE E/S:
## E/S común y separada

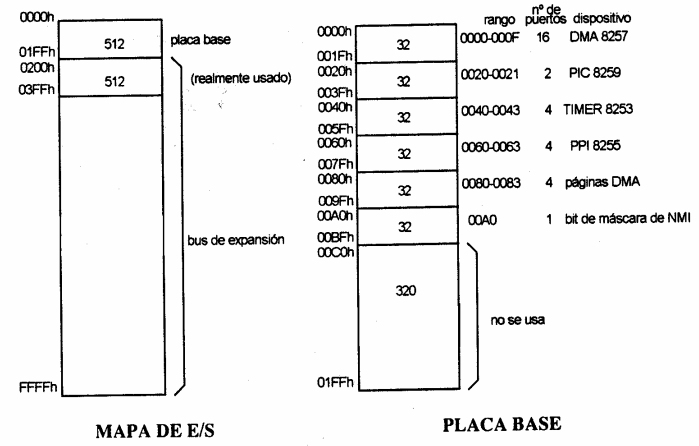



Fig. Esquema direcciones E/S en un PC

# 4. Introducción a los métodos de programación de E/S.

4.1. E/S Programada (Encuesta o *pooling*)
4.2. E/S por interrupciones.
4.3. Acceso directo a memoria (DMA).



## 4.1. E/S Programada (Encuesta o *pooling*)

- La **CPU** tiene el **control absoluto** de la operación de E/S: inicia y lleva a cabo la transferencia.
- La CPU está dedicándose por completo a realizar la operación de E/S: realiza tanto la comprobación de estado como la transferencia y la inicialización: poco eficiente.
- Hardware mínimo.

```
Inicio
        ...
        Inicializar Periferico()
        Mientras existan datos que transferir hacer
                Recoger estado del periférico()
                Si el estado del periferico es PREPARADO entonces
                        Transferir informacion()
                Fin Si
        Fin Mientras
        ...
Fin
```

Bucle de espera activa

# 4.2. E/S por interrupciones (I)

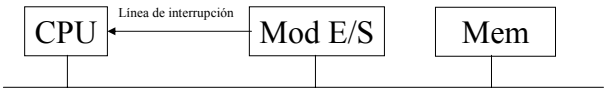
- La E/S le indica a la CPU cuando está preparada para transferir datos (genera una interrupción a la CPU), activando una línea especial conectada a la CPU (**línea de interrupción**).
- Funcionamiento:

  1) El procesador ejecuta instrucciones de un programa. Al finalizar cada instrucción comprueba si se ha producido una interrupción.

  2) En caso afirmativo se salva el estado actual del programa (contador del programa y registros) y se salta a ejecutar la **rutina de servicio** correspondiente.

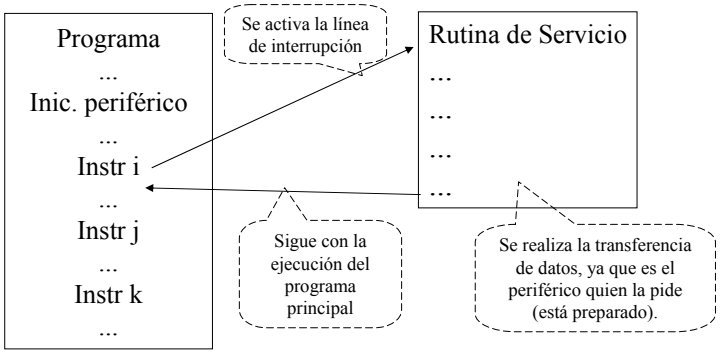
  3) La rutina de servicio efectúa las operaciones apropiadas en la E/S para realizar la transferencia de datos solicitada.

  4) Al finalizar la rutina de servicio se recupera el estado de la CPU y se continúa ejecutando el programa que se estaba ejecutando antes de la interrupción.






# 4.2. E/S por interrupciones (II)




- Las interrupciones pueden ser:
  - **ENMASCARABLES** (se pueden dejar de atender por software)
  - **o NO ENMASCARABLES** (siempre atendidas).
- Dos formas de conocer la dirección/posición (vector) donde se encuentra la rutina de servicio de la interrupción:
  - vector de interrupciones siempre **FIJO**
  - ó el **periférico suministra** el vector de interrupción.

# 4.2. E/S por interrupciones (III)
## (Conexión de varios periféricos y gestión de interrupciones)

- Generalmente existen **VARIOS PERIFÉRICOS** (y no uno sólo) conectados que pueden realizar interrupciones,
  → obliga a **ESTABLECER PRIORIDADES** y decidir cómo se conectan a la CPU.

- También hay que determinar para cada periférico su vector de interrupciones.

SOLUCIONES más extendidas:

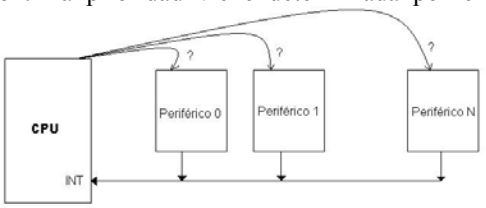
A. Una sola línea de interrupción
B. Varias líneas de interrupción
C. Líneas de interrupción y aceptación



# 4.2. E/S por interrupciones (IV)
## (Conexión de varios periféricos y gestión de interrupciones)

**A. Una sola línea de interrupción**

- Todos los periféricos interrumpen por la misma línea.
- El vector de interrupción es fijo y común a todos los periféricos.
- Mediante encuesta (*polling*) la CPU identifica el periférico y desactiva la interrupción. La prioridad viene determinada por el orden de la encuesta.




**B. Varias líneas de interrupcion**

- Cada periférico tiene su línea de interrupción.
- Cada línea tiene su propio vector de interrupción asociado
- y la CPU determina la prioridad.

# 4.2. E/S por interrupciones (V)
## (Conexión de varios periféricos y gestión de interrupciones)

**C. <u>Líneas de interrupción y aceptación</u>:**

- Una línea de entrada para aceptar interrupciones y otra para dar el reconocimiento de la interrupción al periférico. Ej: procesador i8086.

Hay varias variantes:

**C.1. Daisy-chain o encadenamiento.**
**C.2. Interrupciones vectorizadas.**
**C.3. Gestión centralizada por un controlador de interrupciones (PIC, *programmable interrupt controller*).**



# 4.2. E/S por interrupciones (VI)
## (Conexión de varios periféricos y gestión de interrupciones)

**C.1. Daisy-chain o encadenamiento**:

- Una única línea INT para todos los periféricos.
- Una vez aceptada la interrupción la CPU envía la señal INTA y el periférico de mayor prioridad desactiva la petición de interrupción INT. La señal INTA se propaga de un periférico a otro.
- La prioridad viene determinada por la posición del periférico en la cadena: en cuanto un periférico que generó la señal INT recibe la señal de INTA, activa un biestable especial.
- El vector de interrupciones es fijo y es la CPU mediante *polling* quien identifica al periférico con el biestable activado.

**C.2. Interrupciones vectorizadas**: (similar al anterior)

- El vector de interrupciones no es fijo.
- Cuando un periférico que generó la señal INT recibe la señal INTA, deja el vector de su interrupción en el bus y no propaga más la señal INTA.

# 4.2. E/S por interrupciones (VII)
## (Conexión de varios periféricos y gestión de interrupciones)

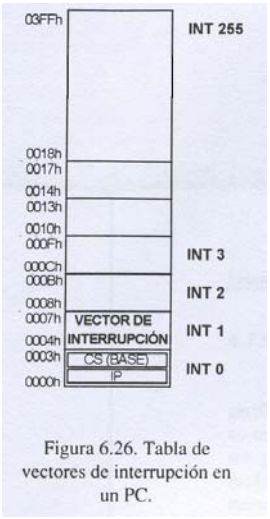
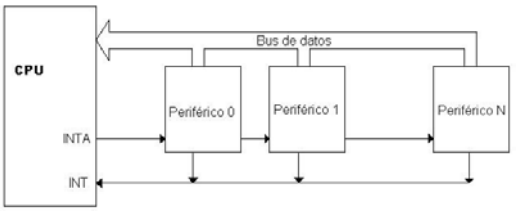



Figura 6.26. Tabla de vectores de interrupción en un PC.




1º) El periférico X interrumpe por INT.

2º) La CPU devuelve señal INTA.

3º) El periférico X devuelve al bus de datos su vector de interrupción.

4º) Con el vector de interrupción se indexa una tabla de vectores en memoria principal para obtener la dirección de la rutina de atención a la interrupción a la que hay que saltar.



# 4.2. E/S por interrupciones (VIII)
## (Conexión de varios periféricos y gestión de interrupciones)

**C.3. Gestión centralizada por un controlador de interrupciones (PIC, *programmable interrupt controller*)**:

- Permite conectar varios periféricos a una misma línea de interrupciones permitiendo enmascarar interrupciones por software, definir vectores de interrupciones asociados a cada interrupción y gestionar prioridades. Ej: el i8259.

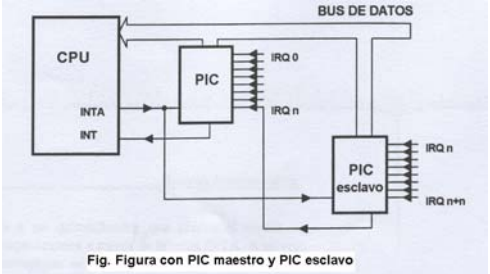


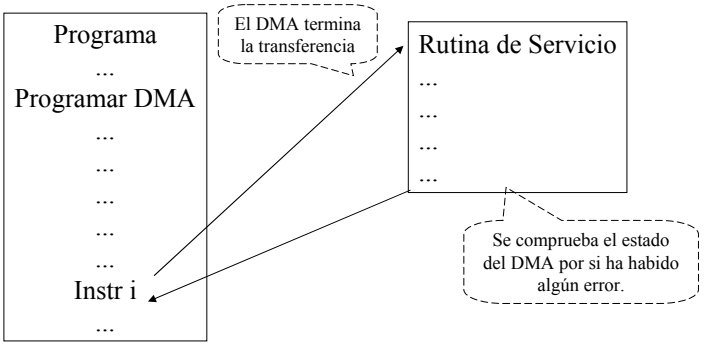

Fig. Figura con PIC maestro y PIC esclavo

# 4.3. Acceso Directo a Memoria (I)

- El DMA (*Direct Memory Access*) es un procesador/controlador especializado en transferencias "muy grandes" desde periféricos a memoria y viceversa.
- Es programable. La CPU no realiza ninguna tarea (salvo programar el DMA) ya que la inicialización y transferencia son gobernadas por el periférico.
- Para programar el DMA hay que enviarle al menos los siguientes datos:
  - Dirección/puerto periférico E/S.
  - Posición/dirección en memoria principal.
  - Tamaño (número de *bytes* a transferir).
  - Tipo transferencia: lectura o escritura.
- Al finalizar el DMA avisa mediante una interrupción. Esta interrupción al igual que el resto de interrupciones son normalmente atendidas al final de cada instrucción. La rutina de servicio asociada comprobará el estado del DMA para ver si se han producido errores al ejecutar la transferencia que se le ha encomendado.



# 4.3. Acceso Directo a Memoria. (II)




- A diferencia del mecanismo por interrupción convencional, una orden DMA puede transferir muchísimos datos de una sola vez. Por lo tanto, el número de interrupciones por byte transferido es mucho menor que con las interrupciones convencionales: se gana en rapidez.
- El procesador no se encarga de la transferencia de datos.

# 4.3. Acceso Directo a Memoria (III)

Existen dos formas básicas de realizar el acceso directo a memoria:

1. **MEMORIA MULTIPUERTA**:
   - La memoria permite realizar transferencias simultáneas por parte de la CPU y otros dispositivos. La memoria tiene varias "puertas" que permiten acceso concurrente a un mismo bloque de memoria por lo que hay que establecer un sistema de arbitraje.
2. **ACCESO A MEMORIA POR ROBO DE CICLO**:
   - Solución mas económica. El DMA cada vez que quiere tomar el control del bus del sistema para realizar la transferencia de un dato, lo solicita a la CPU mediante la señal HOLD. La CPU concede el control del bus mediante la señal HLDA. Después de realizar la transferencia el controlador de DMA lo comunica a la CPU mediante las señales de control correspondientes.

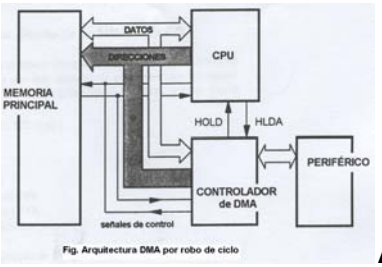


Fig. Arquitectura DMA por robo de ciclo

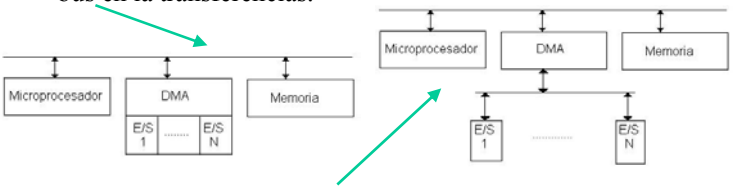


# 4.3. Acceso Directo a Memoria (IV)

Distintas formas de conectar el DMA al sistema:

1. **Bus único, DMA independiente**: Actúa como una CPU de E/S. Necesita un ciclo para acceder al módulo de E/S o periférico y otro para acceder a memoria.


2. **Bus único y E/S integradas**: alternativa para reducir un ciclo de bus en la transferencias.




3. **Bus de E/S**: es una variación de la anterior que permite hacer la arquitectura más escalable.

# 4.3. Acceso Directo a Memoria (V)

Problemas de cohesión con la jerarquía de memoria:

Puede ocurrir que se tenga dos copias de un dato y el DMA sólo sobrescriba sobre una de ellas.

Hay tres SOLUCIONES:

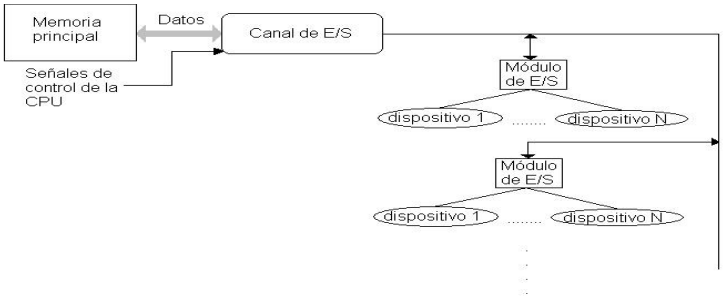
1. Volcar toda la E/S a caché: sólo sirve con E/S asignada a memoria. Costoso.
2. Volcar toda la E/S a memoria: se vacían todos los datos de la caché (bit de validez a cero) que tengan que ver con la transferencia del DMA.
3. Usar técnicas/protocolos de coherencia: invalidar datos de la caché después de que el DMA haya escrito sobre esos datos. Ej. MESI.

Ejemplo de DMA: **i8237**. Posee 4 "canales" (procesadores de DMA) programables con tres modos diferentes y además se puede poner en cascada con otros i8237.



# 5. Canales y procesadores de E/S. (I)

- Siguiente evolución en los sistemas de E/S: tener un procesador capaz de interpretar secuencias de operaciones y de esa forma tener bajo su control un mayor número de operaciones y módulos de E/S, cada vez más complejas.
- El **canal de E/S** es un "pequeño" procesador especializado en operaciones de E/S. Si además tiene memoria propia, entonces se lo llama **procesador de E/S**.



# 5. Canales y procesadores de E/S. (II)

- Para realizar una transferencia de E/S, la CPU primero ha de indicar qué canal de E/S ejecuta un determinado programa.
- La CPU también debe definir el área de almacenamiento temporal, establecer una prioridad y establecer las correspondientes acciones en caso de error. El programa a ejecutar está cargado en memoria principal y puede contener instrucciones propias sólo procesables por el canal de E/S.
- Después de terminar la operación de E/S, el canal de E/S deja el resultado en un área de memoria y a continuación genera una interrupción para indicar que ha acabado.



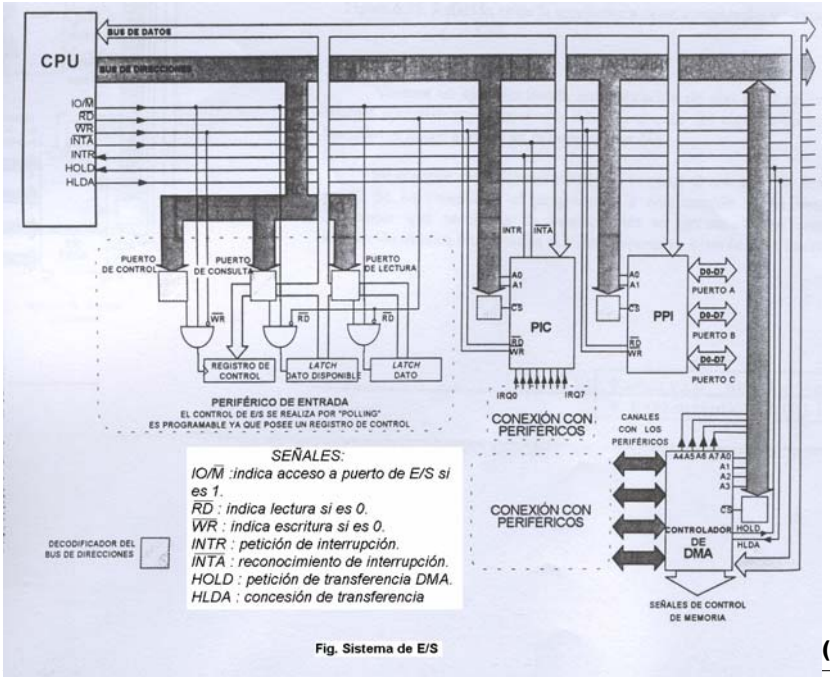



Fig. Sistema de E/S

# 6. Ejemplo de dispositivo de E/S: puerto paralelo (interfaz *centronics*).

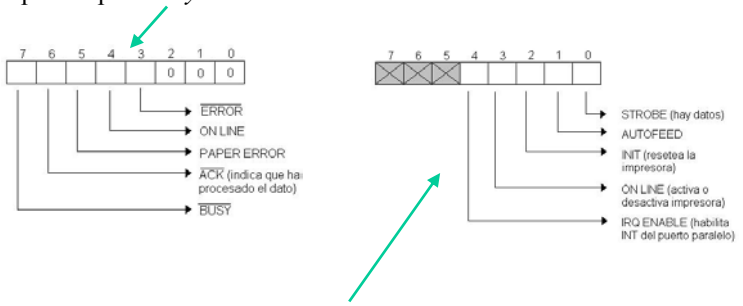
- Es un controlador de E/S que está pensado, en principio, para ser usado solamente por impresoras.
- Para ello habría que utilizar un cable adaptado al bus de puerto paralelo diferente al *Centronics*, que es el que se utiliza habitualmente.
- Un PC tiene dos puertos paralelos llamados LPT1 y LPT2 a partir de las direcciones o puertos de E/S 378 (LPT1) y 278 (LPT2).

  NOTA: ¡¡ no confundir puerto de E/S (dirección) con puerto hardware (puerto serie, puerto paralelo, ...) !!



# 6. Ejemplo de dispositivo de E/S: puerto paralelo (interfaz *centronics*).

- Registro de datos: es de sólo escritura. Direcciones 378 (LPT1) y 278 (LPT2). Compuesto por un *byte*.
- Registro de estado: es de sólo lectura. Direcciones 379 (LPT1) y 279 (LPT2). Compuesto por un *byte*.




- Registro de control: es de lectura/escritura. Direcciones 37A (LPT1) y 27A (LPT2). Compuesto por un *byte*.