

Tema 8: El Sistema de E/S

8.1- Introducción

8.2- Estructura de una interfaz

8.3- Direccionamiento de las interfaces (mapeo en el E.D.)

8.4- Periféricos e interfaces del computador elemental

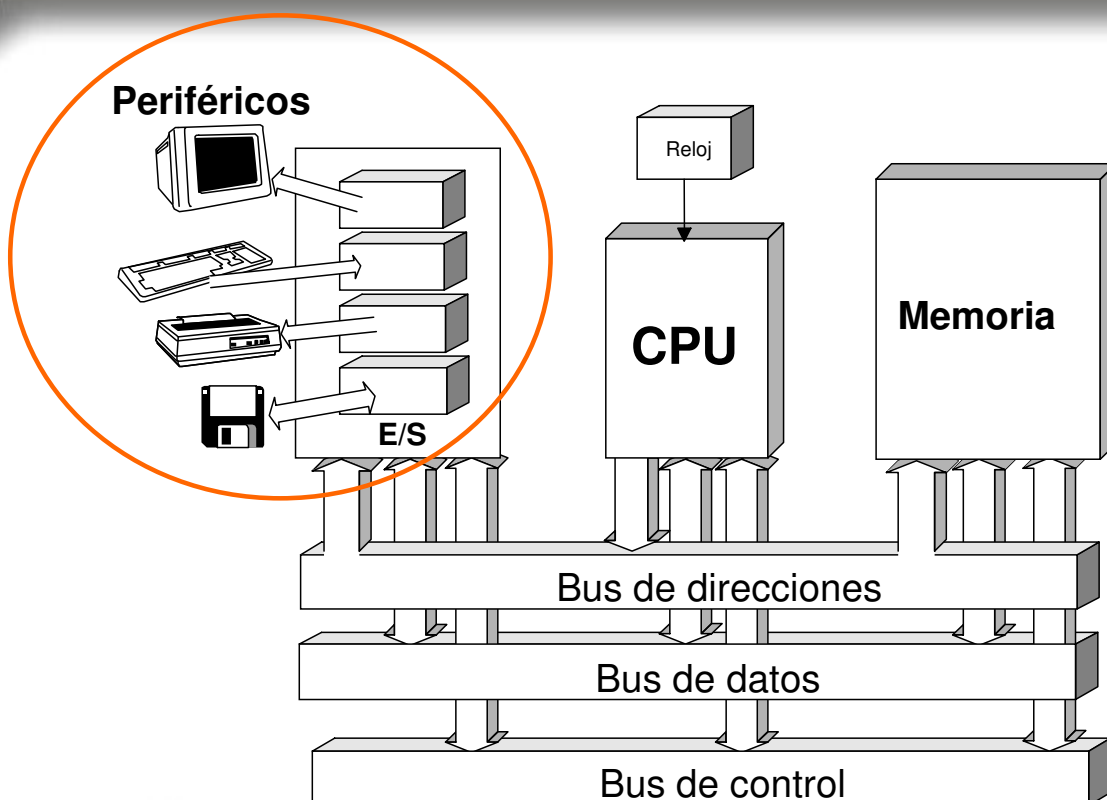
8.4.1- Pantalla

8.4.2- Teclado

8.5- Entrada/Salida programada

8.6- Entrada/Salida mediante interrupciones

Introducción



Introducción

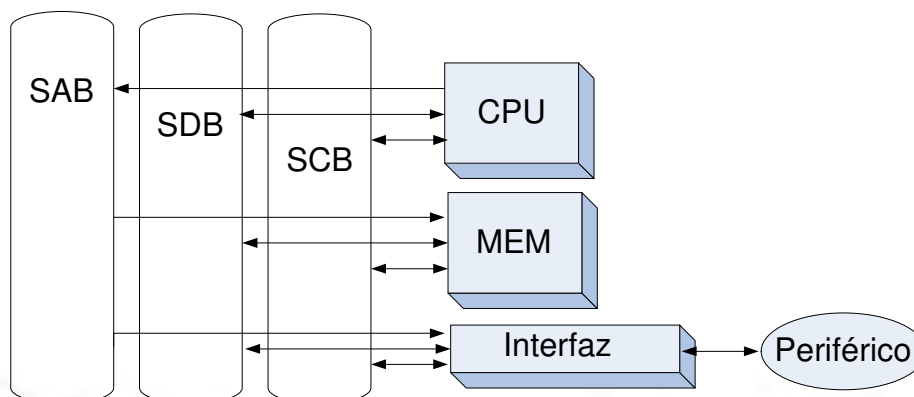
- Objetivo
 - Permitir la comunicación entre el sistema computador y los sistemas periféricos asociados a éste
- Diversos tipos de dispositivos de entrada/salida:
 - Teclado: velocidad de transmisión de 10 bytes/s
 - Modem: velocidad de transmisión de 8 KB/s
 - Pantalla gráfica: velocidad de transmisión de 60 MB/s
- Es necesario idear un mecanismo para conectar dispositivos con características completamente diferentes al computador.

Se utilizarán interfaces entre el computador y el dispositivo



Introducción

- Concepto de interfaz
 - Mecanismo que permite conectar un periférico a un computador
- La interfaz debe ajustarse a las características del periférico.
- Funciones de la interfaz:
 - Dialogar con el computador a través del sistema de buses
 - Dialogar con el periférico (el mecanismo de diálogo depende totalmente del periférico).



Ejemplos de interfaz



Identifica el tipo de interfaz



Área de Arquitectura y Tecnología de Computadores
Departamento de Informática de la Universidad de Oviedo

Fundamentos de Computadores
Tema 8: El Sistema de E/S

5

Ejemplos de interfaz

- Interfaces en la placa base:



Identifica dónde se conecta la interfaz anterior



Área de Arquitectura y Tecnología de Computadores
Departamento de Informática de la Universidad de Oviedo

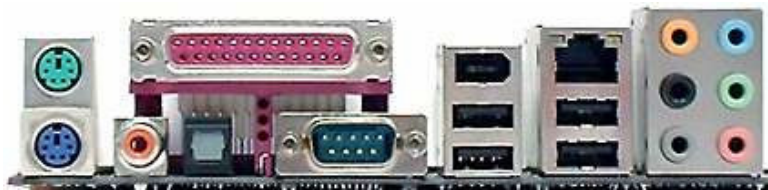
Fundamentos de Computadores
Tema 8: El Sistema de E/S

6

Ejemplos de interfaz

- Otras interfaces en la placa base:
 1. 2x PS2 (teclado y ratón)
 2. 1x Puerto serie
 3. 1x Puerto paralelo
 4. 1x Conexión de red
 5. 4x USB 2.0
 6. 1x IEEE 1394 FireWire
 7. 1x Digital Audio Out (óptico)
 8. 1x Digital Audio Out (coaxial)
 9. 1x 7.1 Channel Audio, Mic-In, Line-In (6 jacks)

Identifica las interfaces

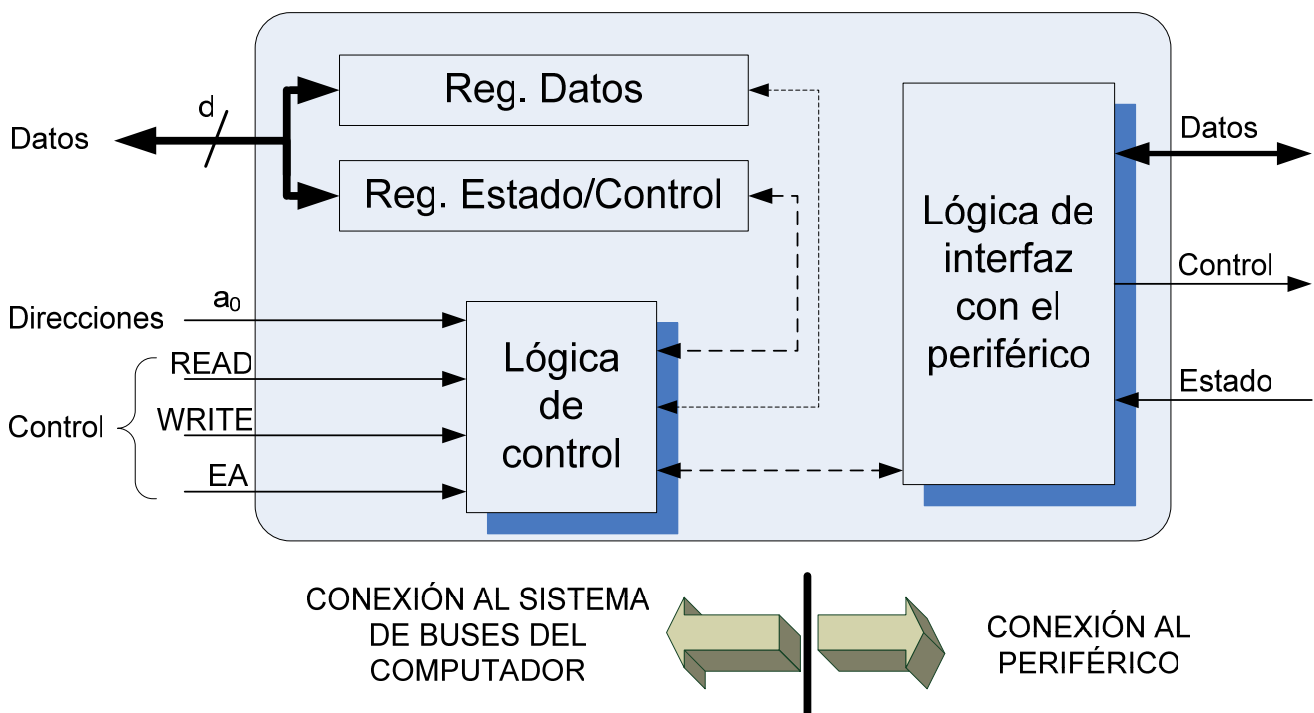


Área de Arquitectura y Tecnología de Computadores
Departamento de Informática de la Universidad de Oviedo

Fundamentos de Computadores
Tema 8: El Sistema de E/S

7

Estructura de una interfaz simple



Área de Arquitectura y Tecnología de Computadores
Departamento de Informática de la Universidad de Oviedo

Fundamentos de Computadores
Tema 8: El Sistema de E/S

8

Estructura de una interfaz simple

- Registro de datos:
 - Se utiliza para almacenar temporalmente los datos transferidos entre el computador y el periférico.
- Registro de Estado/Control
 - Se utiliza para indicar el estado en el que se encuentra el periférico (por ejemplo, si está listo para enviar o recibir un nuevo dato), o bien para que la CPU envíe órdenes al periférico.
- Lógica de Interfaz con el Periférico:
 - Controla la transferencia de datos con el periférico.
- Lógica de Control:
 - Controla a todos los elementos de la interfaz.

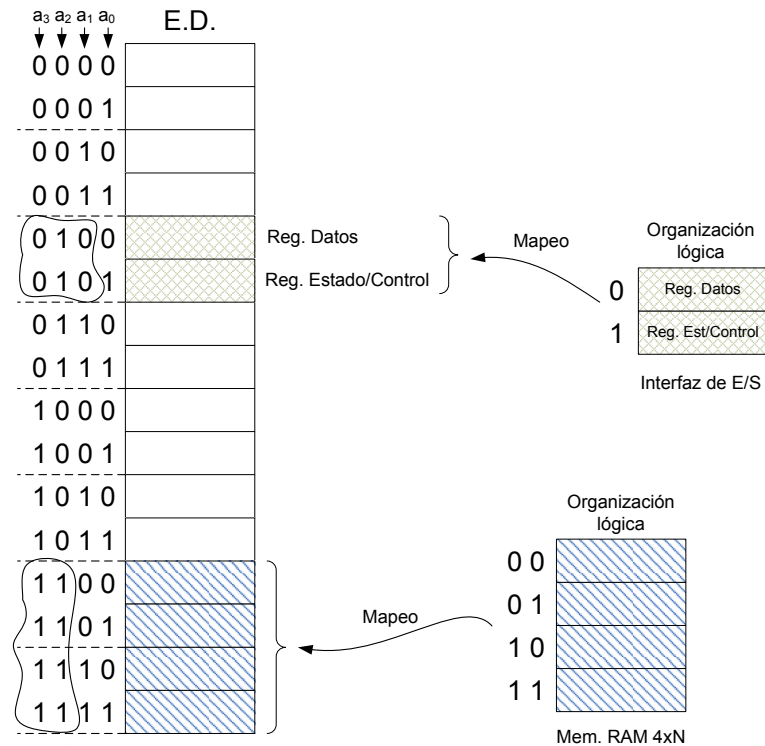


Direccionamiento de las interfaces (mapeo en el E.D.)

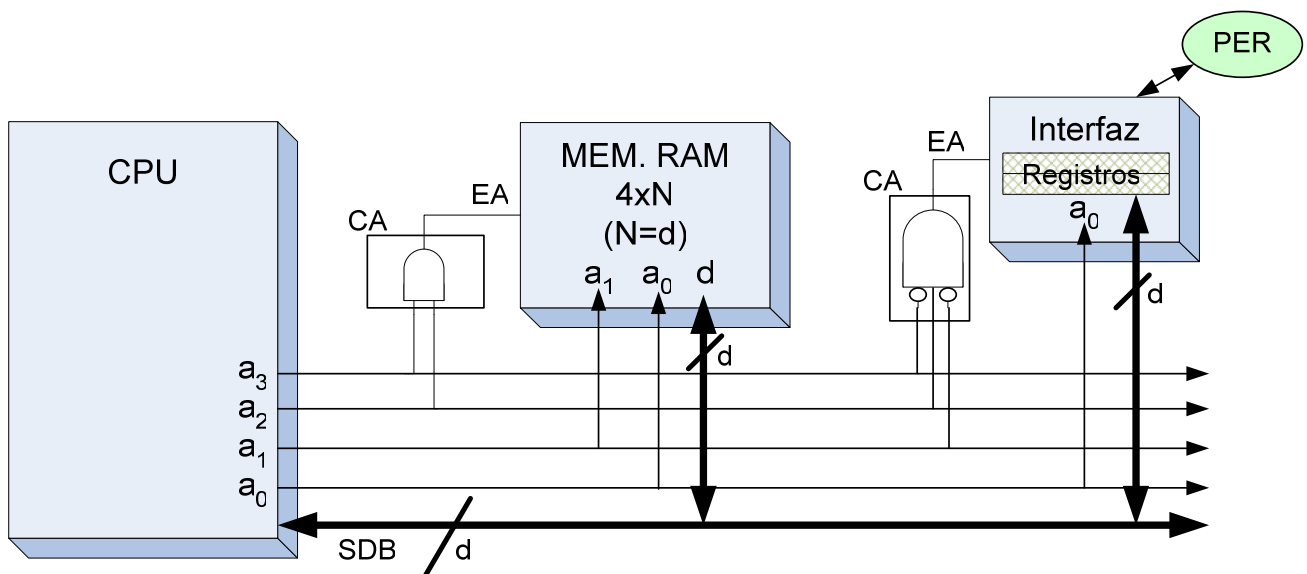
- Las interfaces tienen unos registros que tienen que ser accesibles por la CPU.
 - La solución es mapearlas en el espacio de direcciones (ED).
- Una vez que un registro está mapeado en una dirección del ED de la CPU, trabajar con ese registro equivale a trabajar sobre esa dirección.
- Ejemplo de ubicación:
 - SAB del computador: $a = 4$
 - SDB del computador: d
 - Dispositivo de memoria RAM: $4 \times N$ ($N = d$)
 - Interfaz de E/S: dos registros (estado/control y datos) de N bits ($N = d$)



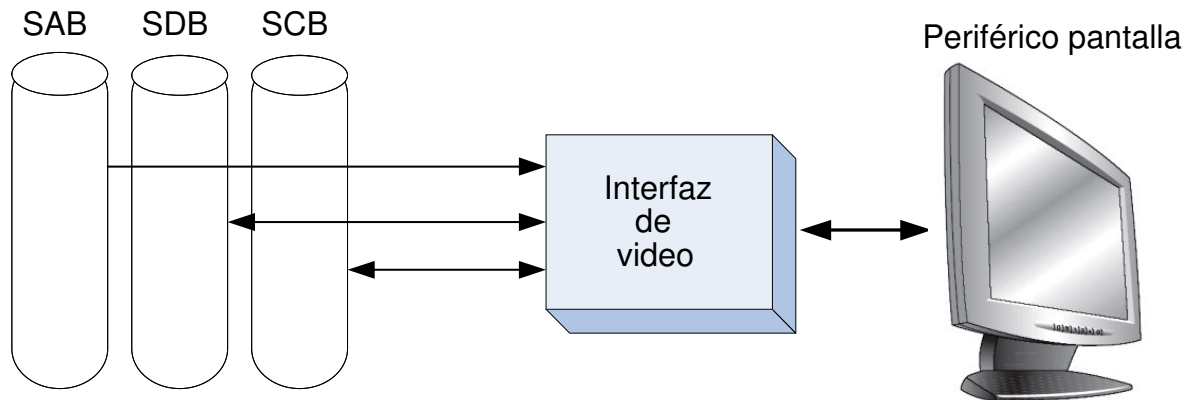
Direcccionamiento de los interfaces (mapeo en el E.D.)



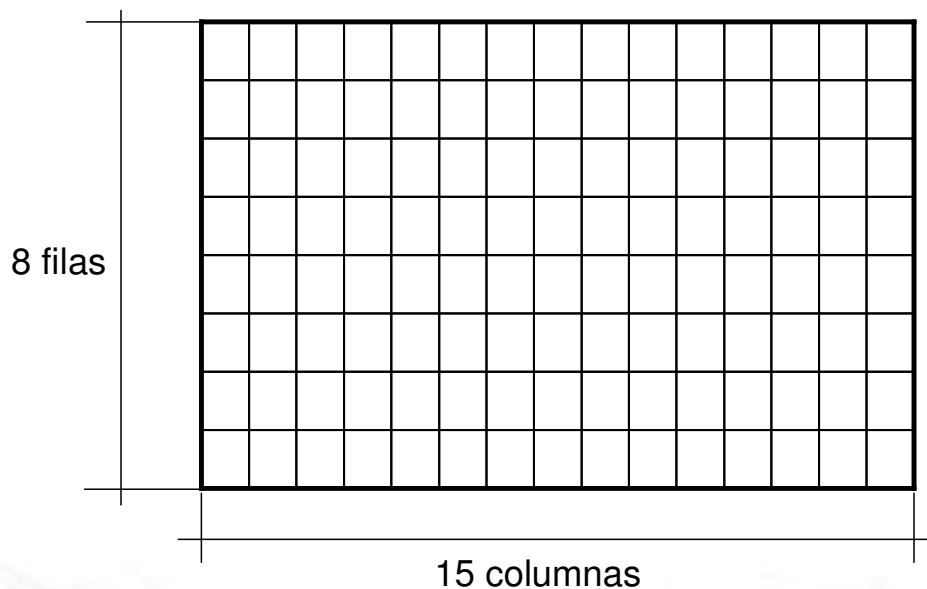
Direcccionamiento de las interfaces (mapeo en el E.D.)



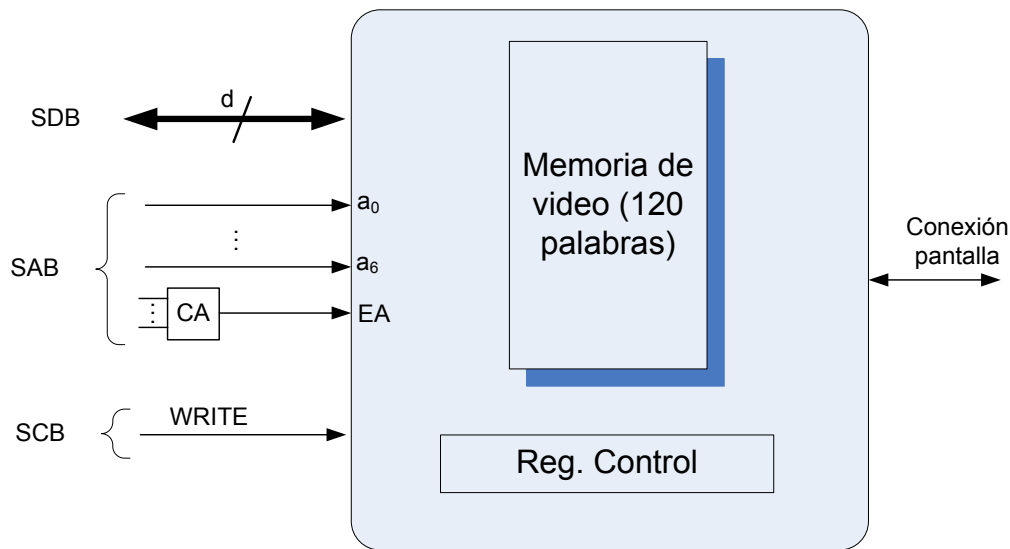
- Pantalla:



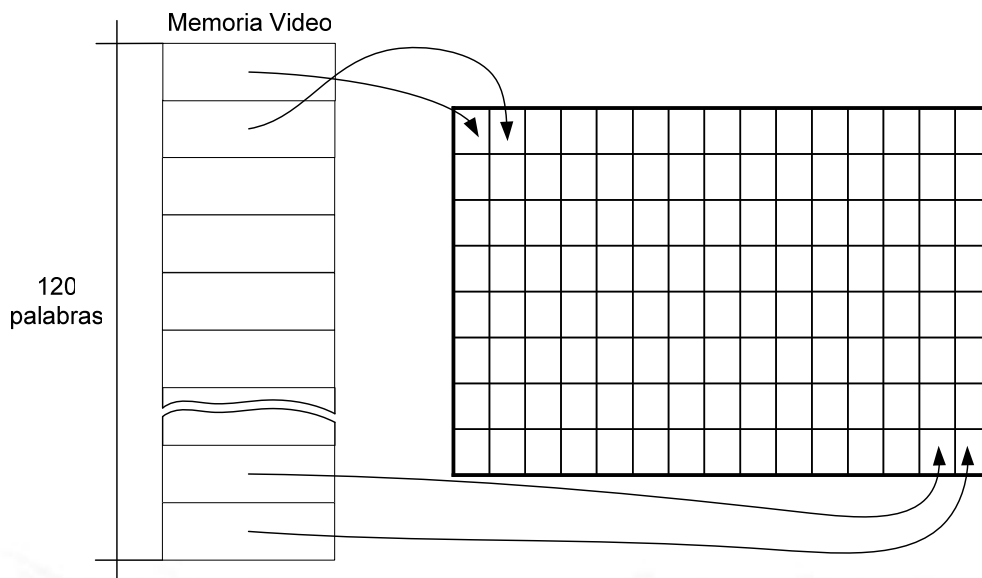
- Periférico pantalla:
 - Representa caracteres ASCII organizados en filas y columnas.
 - 8 filas x 15 columnas = 120 caracteres



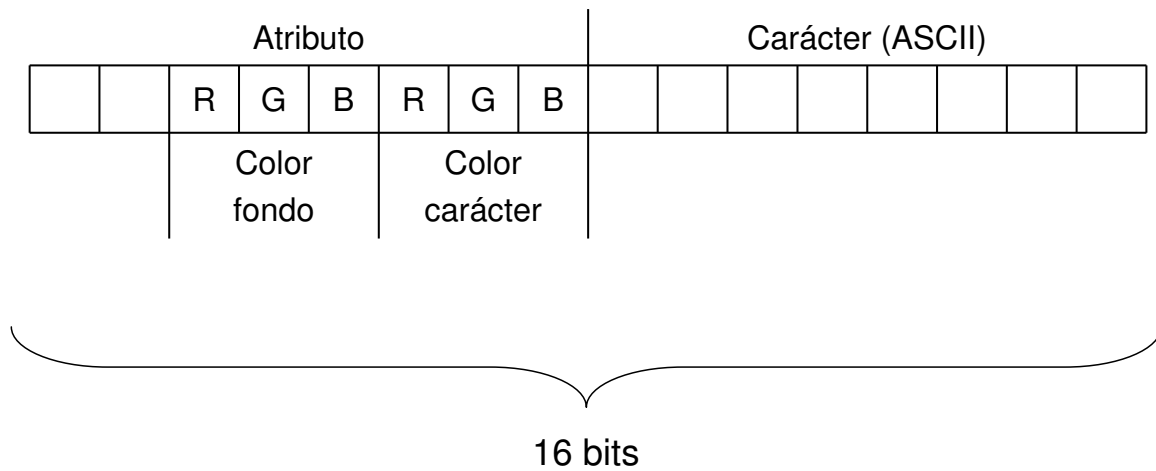
- Interfaz de vídeo:



- Memoria de vídeo:
 - Cada palabra de la memoria de video se corresponde con una determinada posición de la pantalla.



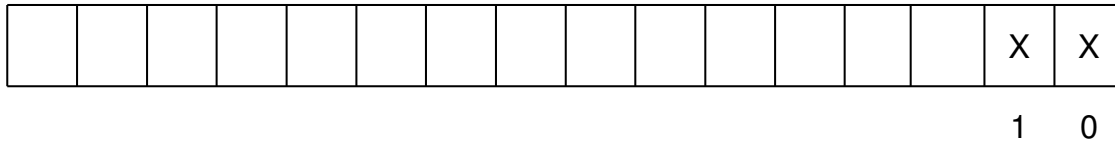
- Significado de cada palabra de la memoria de vídeo:



- Codificación RGB en las palabras de la memoria de vídeo:

Código RGB	Color asociado
000	Negro
001	Azul
010	Verde
011	Cian
100	Rojo
101	Magenta
110	Amarillo
111	Blanco

- Registro de control:
 - Bit 0: borra el contenido de la pantalla.
 - Bit 1: apaga/enciende la pantalla.



- Mapeo de la interfaz de vídeo:
 - Elementos mapeables (en este orden):
 - Memoria de vídeo
 - Registro de control
 - 121 elementos \rightarrow 128 posiciones (2^7)
- Ejercicio:
 - Diseñar el circuito de activación necesario para mapear la interfaz de vídeo a partir de la dirección F100h

El número de posiciones ocupadas es potencia de 2 para simplificar el circuito de activación

- Acceso de la CPU a la interfaz de vídeo:
 - Ejercicio:
 - Interfaz de video mapeada a partir de la dirección C100h
 - Escribir un programa que escriba la cadena de caracteres “Hola, mundo” en la segunda línea de la pantalla. Los caracteres deben imprimirse en color rojo sobre fondo amarillo.

```
ORIGEN 0100h
.PILA 100h
.DATOS
        cadena VALOR "Hola, mundo", 0
.CODIGO
```



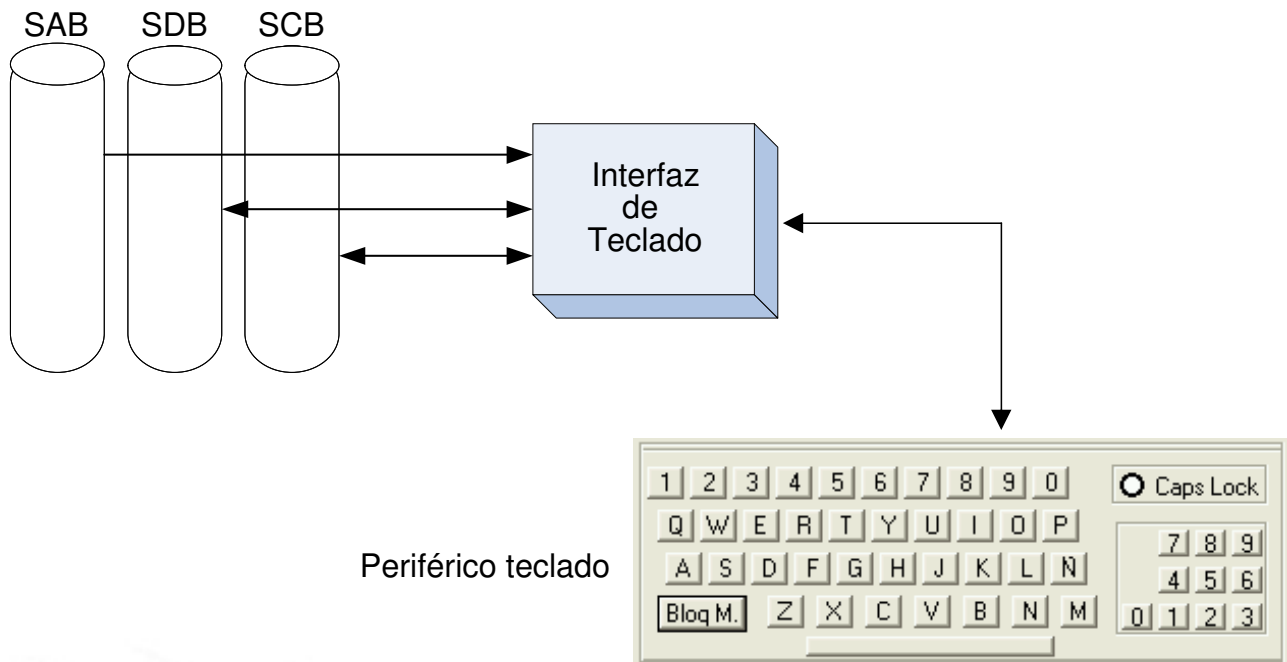
FIN

Atributo								Carácter (ASCII)							
0	0	1	1	0	1	0	0								
				Color fondo				Color carácter							



Periféricos e interfaces del computador elemental: teclado

- Teclado:



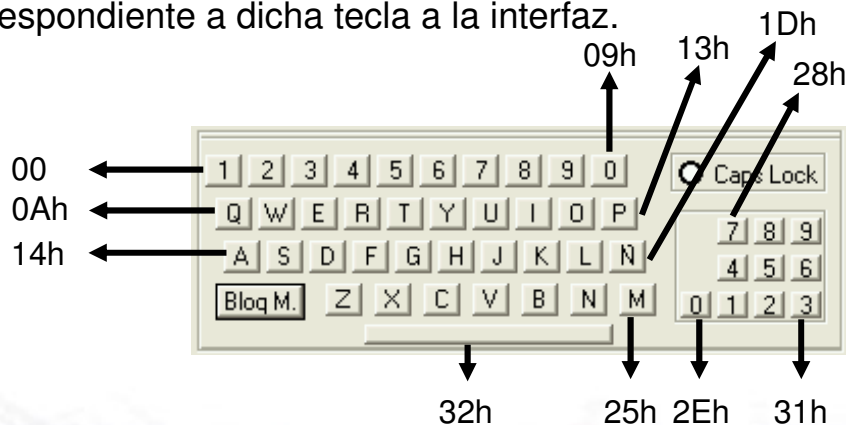
Periféricos e interfaces del computador elemental: teclado

- Periférico teclado:

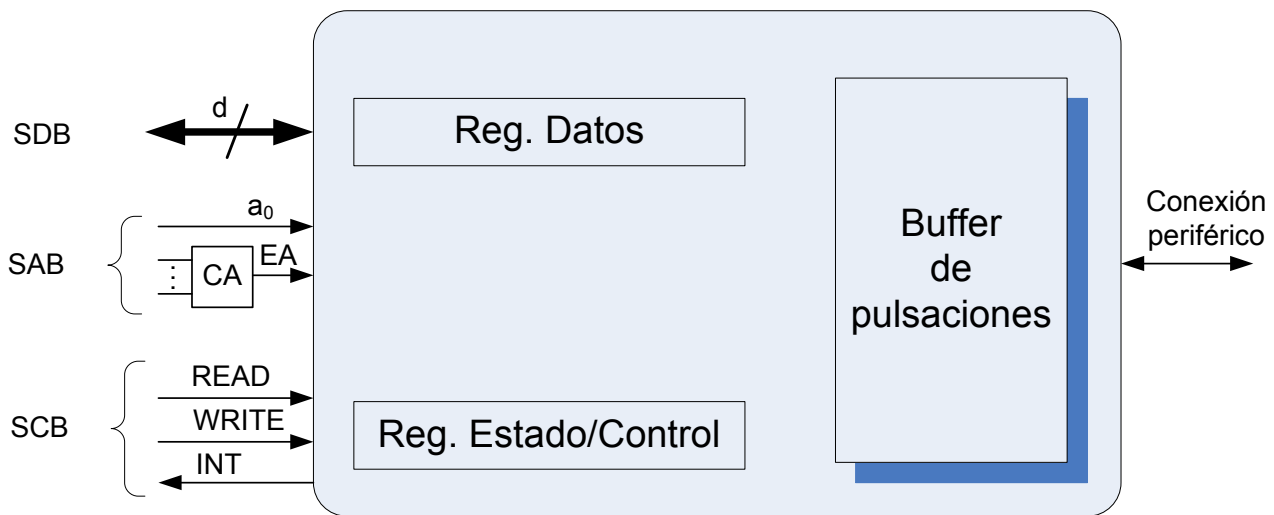
- Contiene las teclas básicas de cualquier teclado real. Se eliminan algunas teclas; por ejemplo, las teclas de función.

- Funcionamiento:

- Cada tecla tiene asociado un código de 8 bits, denominado código de SCAN.
- Cuando se pulsa una tecla, el teclado envía el código de SCAN correspondiente a dicha tecla a la interfaz.



- Interfaz del teclado:

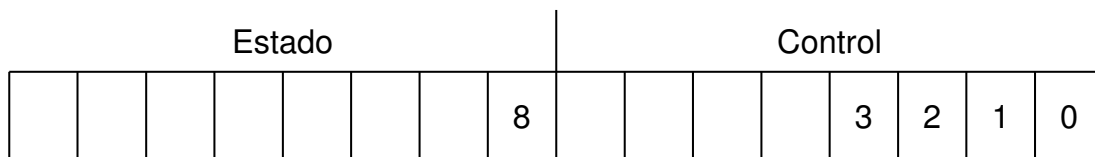


- Buffer de pulsaciones:**
 - Almacena los códigos de hasta 16 pulsaciones enviadas por el periférico.
 - Para cada pulsación almacena el código de SCAN y el código ASCII correspondiente a la tecla pulsada. El código ASCII es calculado automáticamente por la propia interfaz.
 - Cuando el buffer está lleno los códigos enviados desde el teclado son rechazados.
- Registro de datos:**
 - El registro de datos es un registro de 16 bits.
 - Se usa para acceder a la primera pulsación almacenada en el buffer.
 - La pulsación es eliminada después de realizar el acceso.

Periféricos e interfaces del computador elemental: teclado

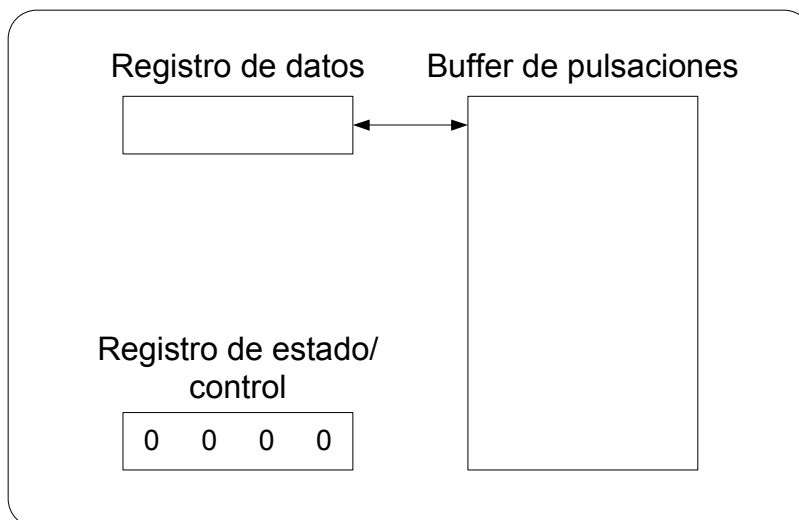
- Registro de estado/control:

- Bit 8: Está activado mientras existan datos en el buffer
 - Bit 0: Borrar primer carácter del buffer
 - Bit 1: Borrar último carácter del buffer
 - Bit 2: Borrar todos los caracteres del buffer
 - Bit 3: Activar/Desactivar generación de interrupciones
- Estado
- Control



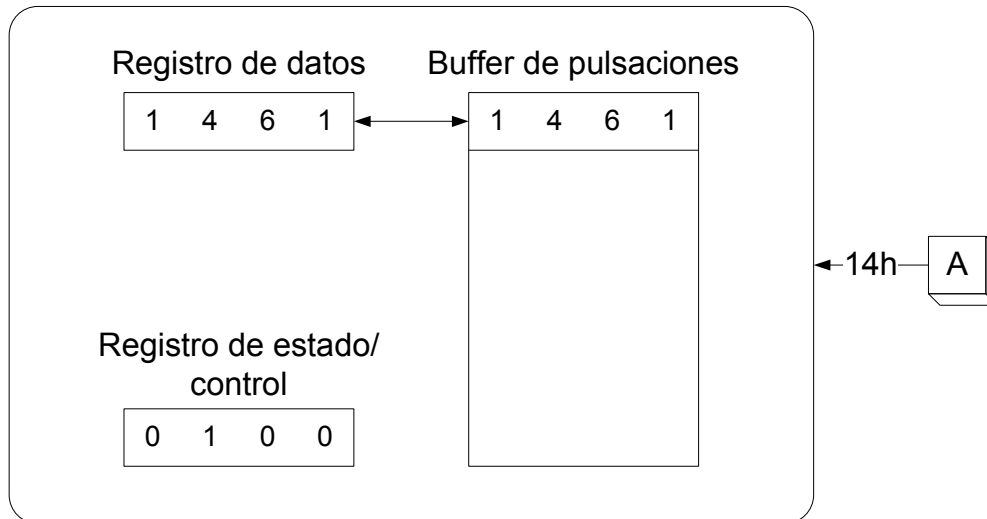
Periféricos e interfaces del computador elemental: teclado

- Funcionamiento de la interfaz del teclado:
 - Estado inicial



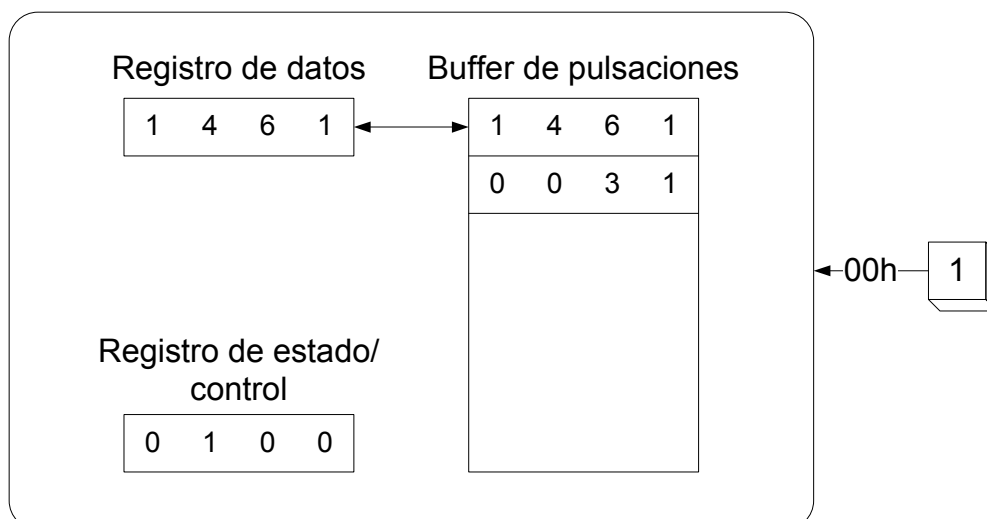
Periféricos e interfaces del computador elemental: teclado

- Funcionamiento de la interfaz del teclado:
 - Se pulsa la tecla 'A' (Código de scan 14h, ASCII 61h)



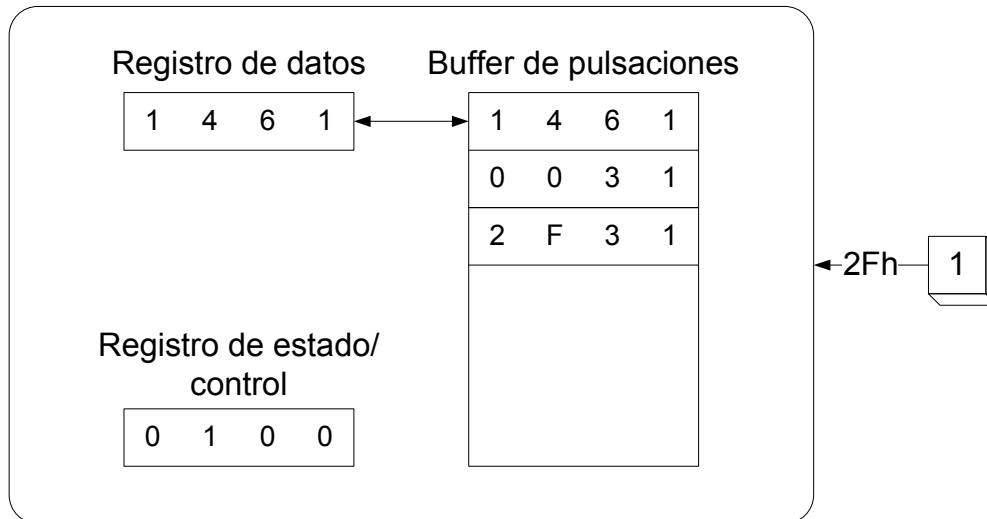
Periféricos e interfaces del computador elemental: teclado

- Funcionamiento de la interfaz del teclado:
 - Se pulsa la tecla '1' teclado alfanumérico (Código de scan 00h, ASCII 31h)



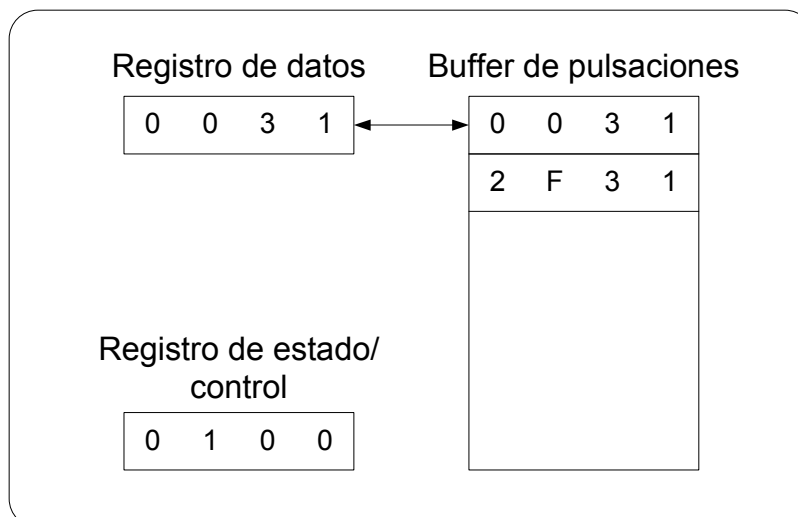
Periféricos e interfaces del computador elemental: teclado

- Funcionamiento de la interfaz del teclado:
 - Se pulsa la tecla '1' teclado numérico (Código de scan 2Fh, ASCII 31h)



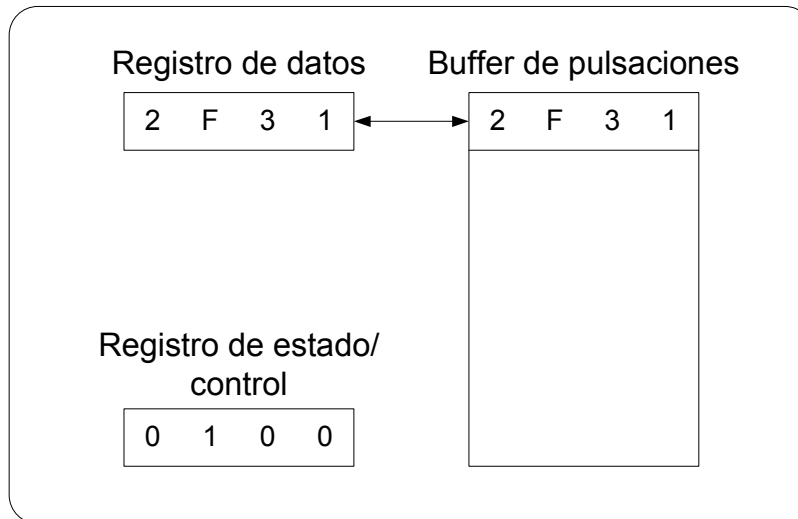
Periféricos e interfaces del computador elemental: teclado

- Funcionamiento de la interfaz del teclado:
 - Se realiza una lectura del registro de datos



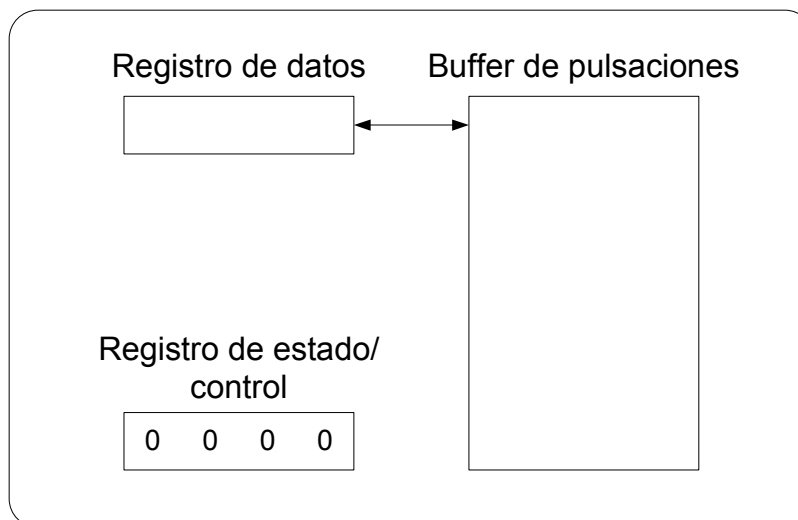
Periféricos e interfaces del computador elemental: teclado

- Funcionamiento de la interfaz del teclado:
 - Se realiza una lectura del registro de datos



Periféricos e interfaces del computador elemental: teclado

- Funcionamiento de la interfaz del teclado:
 - Se realiza una lectura del registro de datos



- Mapeo de la interfaz de teclado:
 - Dos elementos mapeables (en este orden):
 - Registro de datos
 - Registro de estado/control
- Ejercicio:
 - Diseñar el circuito de activación necesario para mapear la interfaz de teclado a partir de la dirección F000h



Entrada/Salida programada

- Ejercicio:
 - Realizar un programa que lea una pulsación del teclado suponiendo que su interfaz está mapeada a partir de la dirección F000h:

```
ORIGEN 0100h
.CODIGO
    MOVL R0, 00h
    MOVH R0, 0F0h
    MOV R5, [R0] }
FIN
```

—————▶ Transferencias de datos

- Problema:
 - ¿Qué ocurre si no se ha pulsado ninguna tecla? Falta algo más

Es necesario realizar una sincronización



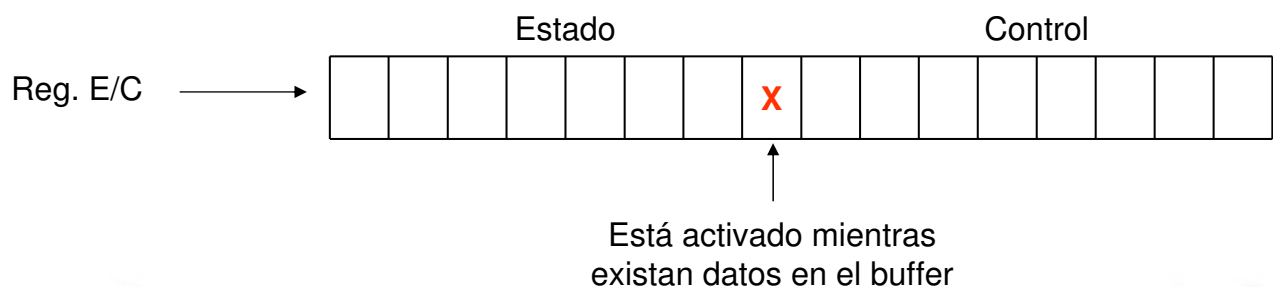
Entrada/Salida programada

- Necesidad de la sincronización:
 - La sincronización es necesaria debido a la diferencia de velocidad existente entre la CPU y los periféricos.
- Sincronización en operaciones de salida:
 - Consiste en que la CPU sólo envíe datos al periférico cuando éste esté listo para recibirlos.
- Sincronización en operaciones de entrada:
 - Consiste en que la CPU sólo lea datos del periférico cuando éste tenga datos disponibles.



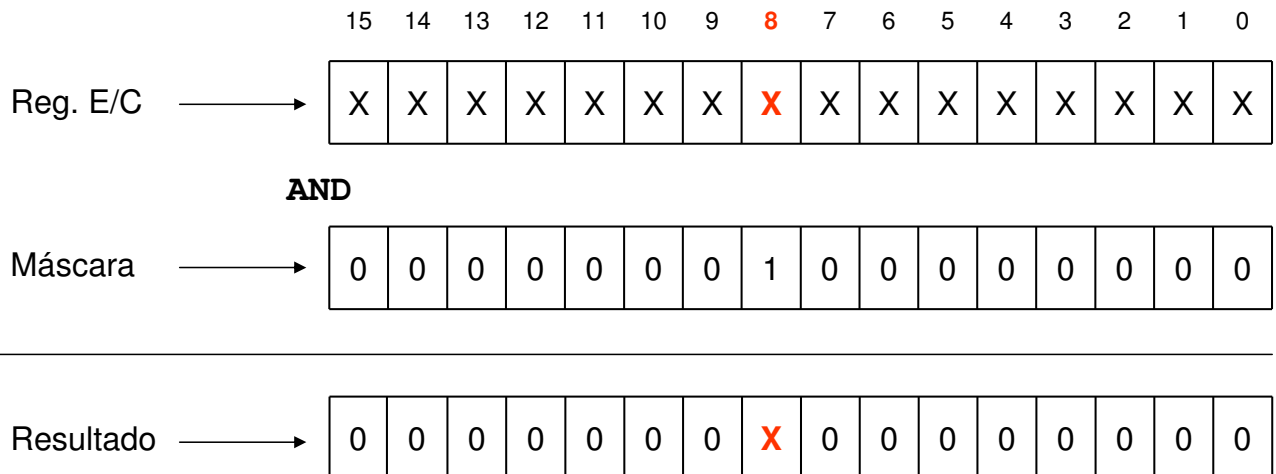
Entrada/Salida programada

- Ejercicio:
 - Realizar un programa que lea una pulsación del teclado suponiendo que su interfaz está mapeada a partir de la dirección F000h.
- Sincronización:
 - La CPU debe esperar a que se pulse una tecla antes de leer la pulsación del registro de datos de la interfaz de teclado.
 - En el registro de estado/control de la interfaz de teclado hay un bit que sirve para determinar si se ha pulsado una tecla.



Entrada/Salida programada

- Determinar si el valor de un bit es 1:



Si X = 0 → Resultado = 0 → ZF = 1
Si X = 1 → Resultado ≠ 0 → ZF = 0



Entrada/Salida programada

- Solución del ejercicio con sincronización:

```
ORIGEN 0100h
.CODIGO
    MOVL R0, 00h
    MOVH R0, 0F0h
    MOVL R1, 01h
    MOVH R1, 0F0h
    MOVL R2, 00h
    MOVH R2, 01h

bucle_de_espera:
    MOV R3, [R1]
    AND R3, R3, R2
    BRZ bucle_de_espera

    MOV R5, [R0]

FIN
```

Registro de datos

Registro de estado/control

Máscara

Bucle de sincronización

Transferencias de datos



Entrada/Salida programada

- Entrada/Salida programada:
 - Técnica de E/S en la que la CPU controla todo el proceso de E/S mediante la ejecución de un programa.
- Sincronización: muestreo periódico
 - Mecanismo de sincronización usado en la E/S programada (como el del ejemplo anterior).
 - Se basa en leer constantemente el registro de E/C de una interfaz para determinar cuando se dispone de un nuevo dato.
 - Cuando se detecta un nuevo dato se abandona el bucle de sincronización y se lleva a cabo la lectura del dato.

Problemas del muestreo periódico: mantiene a la CPU ocupada



Entrada/Salida mediante interrupciones

- Nosotros, al igual que los computadores, usamos periféricos para comunicarnos con el exterior, por ejemplo el teléfono.
- Forma de determinar si alguien se quiere comunicar con nosotros a través del teléfono:
 1. Descolgar cada 5 segundos para ver si hay alguien al otro lado.
 - Nos mantenemos ocupados a la espera de que alguien se quiera comunicar con nosotros y no podemos hacer otra cosa.
 2. Hacer que el teléfono nos avise de cuando alguien está llamando.
 - Cuando alguien se quiera comunicar con nosotros, el teléfono nos interrumpe.
 - Tenemos la opción de atender a la interrupción o no.
 - Podemos hacer otras cosas mientras esperamos la posible llamada.

**Entrada/Salida programada.
Sincronización mediante muestreo periódico**

**Entrada/Salida programada.
Sincronización mediante interrupciones**

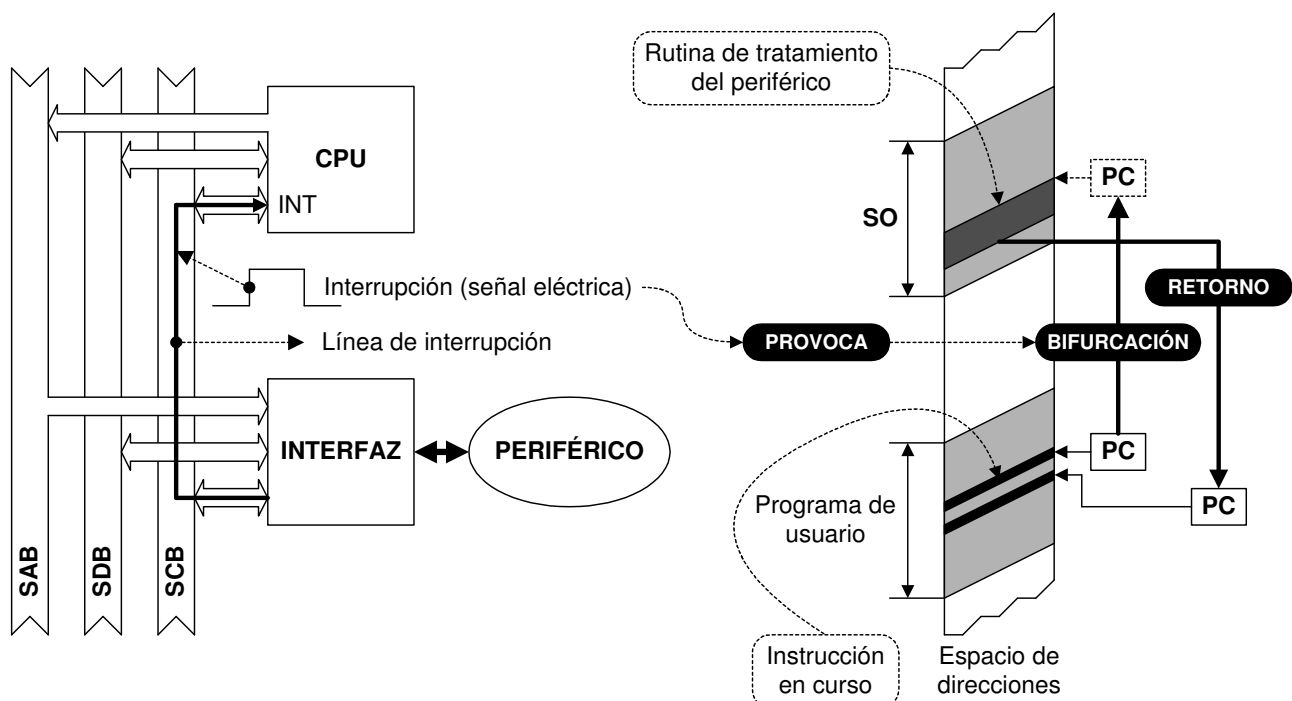


Entrada/Salida mediante interrupciones

- Desde el punto de vista de la E/S en el computador:
 - Los periféricos solicitan la atención de la CPU mientras ésta ejecuta los programa de usuario.
 - Atender a un periférico consiste en abandonar la ejecución del programa en curso y pasar a ejecutar una rutina que atiende al periférico. Tras la ejecución de la rutina se retorna al punto del programa en el que se produjo la interrupción.
- Concepto de interrupción:
 - Una interrupción es una señal eléctrica enviada desde una interfaz hasta la CPU, a través de una línea conocida normalmente como INT.
 - Se genera un interrupción cuando un periférico quiere solicitar la atención de la CPU.



Entrada/Salida mediante interrupciones



Entrada/Salida mediante interrupciones

- Aspectos de la gestión de las interrupciones:

1. Identificación de los dispositivos:

- Cuando la CPU detecta una interrupción, la identificación implica determinar cuál ha sido el dispositivo periférico que la generó.

Vectorización

2. Prioridad de atención:

- Si varios dispositivos generan una interrupción simultáneamente, ¿cuál de ellos debe ser atendido primero?

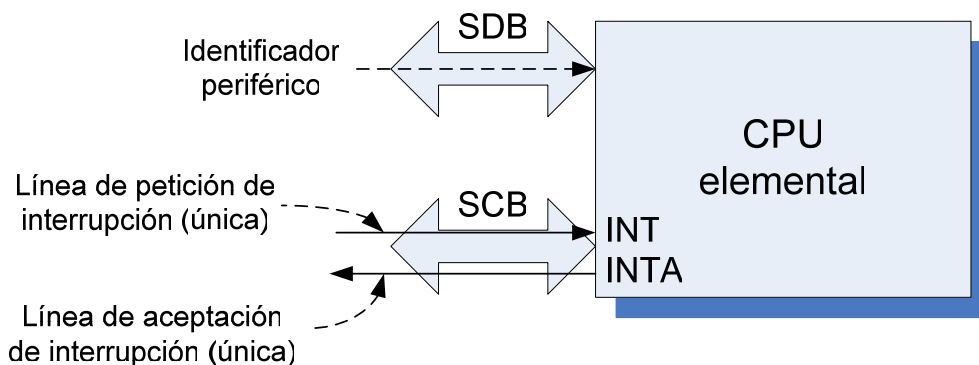
Encadenamiento de la línea INTA



Entrada/Salida mediante interrupciones

- Secuencia básica de operaciones en la gestión de interrupciones:

1. Un periférico genera una petición de interrupción a través de INT.
2. Si la CPU acepta la petición de interrupción, genera una señal de aceptación a través de la línea INTA.
3. El periférico se identifica mediante un identificador (número) a través del SDB.



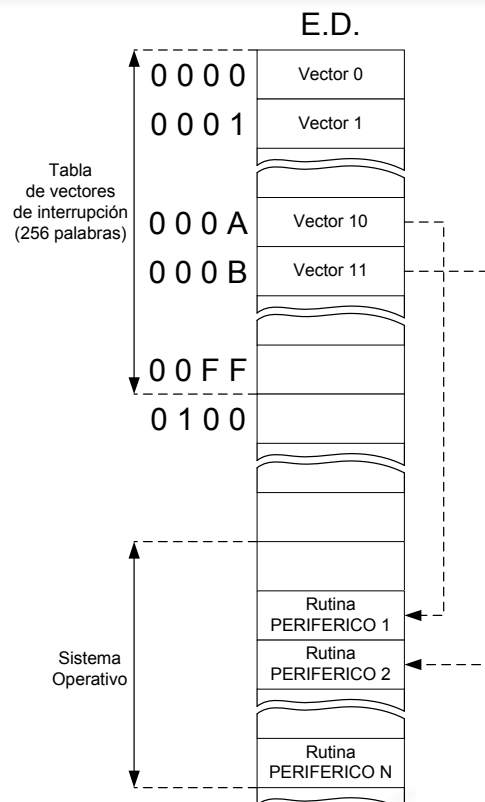
Entrada/Salida mediante interrupciones

- Identificación de los periféricos: vectorización
 - La atención por parte de la CPU a una interrupción consiste en la ejecución de un código que sabe atender a ese periférico. A este código se le llama **rutina de interrupción**.
 - Para saber qué rutina de interrupción se debe ejecutar, es necesario saber qué periférico genera la interrupción, es decir, el periférico debe identificarse. Al identificador de un periférico se le denomina **número de vector de interrupción**.
 - Hay una estructura de datos, la **tabla de vectores de interrupción** (TVI), que relaciona cada número de vector con una dirección de una rutina de interrupción para el periférico identificado con ese número de vector.
 - En el computador elemental se reservan las 256 primeras palabras del E.D. para contener la TVI.



Entrada/Salida mediante interrupciones

- Vectorización:



Entrada/Salida mediante interrupciones

- Pasos de atención de una interrupción
 - Fase de desencadenamiento de la interrupción:
 1. Un periférico hace una petición de interrupción a través de la línea INT.
 2. En el último paso de ejecución de la instrucción en curso se comprueba el estado de la línea INT. Si INT está activa:
 - Si IF = 0 → la interrupción es rechazada y se continúa ejecutando la instrucción siguiente.
 - Si IF = 1 → la interrupción es aceptada y se continúa con el paso siguiente de atención de la interrupción.
 3. Se salva en la pila del programa el SR y el PC. En este momento el PC contiene la dirección de retorno de la interrupción.
 4. La CPU notifica al periférico que acepta la interrupción activando la línea INTA.
 5. El periférico responde colocando en el SDB su número de identificación (nº de vector que apunta a su rutina de tratamiento).



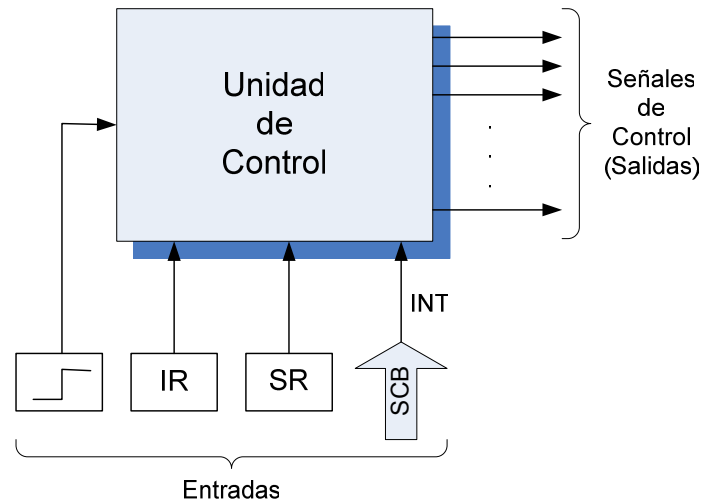
Entrada/Salida mediante interrupciones

- Pasos de atención de una interrupción (continuación)
 6. La CPU accede a la TVI y obtiene la dirección de la rutina de tratamiento del periférico.
 7. Recarga el PC con la dirección obtenida en el paso anterior, quedando así preparada para ejecutar la rutina de tratamiento del periférico.
 8. Pone a '0' el bit IF del registro de estado, inhibiendo así las interrupciones.
 - Fase de ejecución de la rutina de tratamiento y retorno:
 9. Se ejecuta la rutina de tratamiento del periférico, que debe terminar con una instrucción IRET.
 10. Se ejecuta la instrucción IRET, la cual desapila PC y SR retornándose así al programa interrumpido.



Entrada/Salida mediante interrupciones

- Unidad de control:
 - Recibe como entrada la línea de petición de interrupciones.

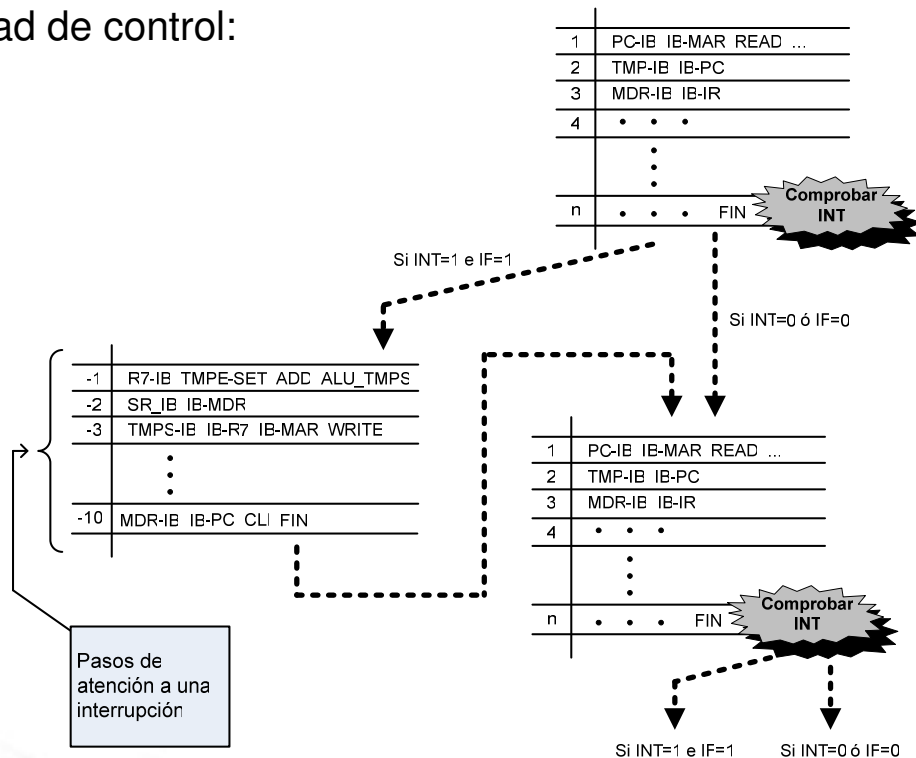


Entrada/Salida mediante interrupciones

- Unidad de control:
 - La unidad de control comprueba el estado de la línea INT en el último paso de ejecución de todas las instrucciones.
 - Si $INT=0$ ó $IF=0 \rightarrow$ comienza la fase de búsqueda e incremento del PC de la instrucción siguiente.
 - Si $INT=1$ e $IF=1 \rightarrow$ se ejecutan un conjunto de pasos especiales para atender a la interrupción.
 - Las interrupciones pueden ocurrir durante cualquier paso de ejecución de la instrucción en curso; sin embargo sólo son detectadas en el último paso de la ejecución de la instrucción.

Entrada/Salida mediante interrupciones

- Unidad de control:



Entrada/Salida mediante interrupciones

- Unidad de control: pasos de atención a una interrupción

Paso	Señales de control	
I-1	R7-IB, TMPE-SET, ADD, ALU-TMPS	Apilar SR
I-2	SR-IB, IB-MDR	
I-3	TMPS-IB, IB-R7, IB-MAR, WRITE	
I-4	R7-IB, TMPE-SET, ADD, ALU-TMPS	Apilar PC
I-5	PC-IB, IB-MDR	
I-6	TMPS-IB, IB-R7, IB-MAR, WRITE, INTA	
I-7	Ciclo de Espera	Cargar la dirección de comienzo de la rutina
I-8	MDR-IB, IB-MAR, READ	
I-9	Ciclo de Espera	
I-10	MDR-IB, IB-PC, CLI, FIN	

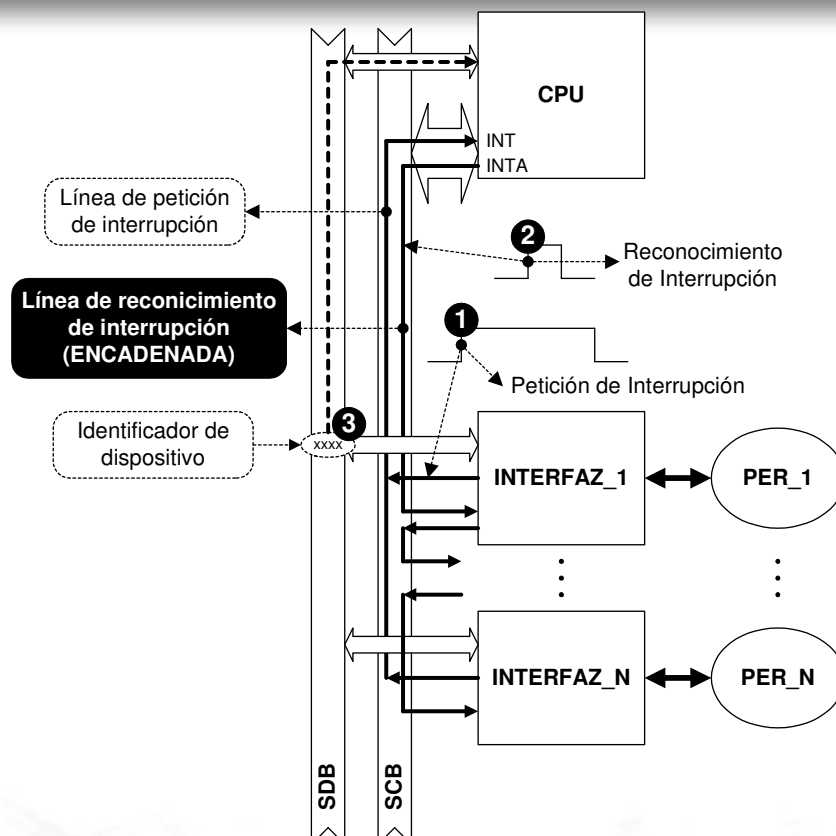


Entrada/Salida mediante interrupciones

- Gestión de prioridad de atención: encadenamiento
 - El encadenamiento se realiza sobre la línea de reconocimiento de interrupciones INTA. Esta línea se conecta encadenadamente a todas las interfaces que forman el sistema de E/S del computador.
 - Cuando la CPU genera una señal de reconocimiento de interrupción, ésta se propaga mediante la línea INTA a través de la cadena de interfaces, hasta que alcanza a una que haya solicitado interrupción.
 - El orden de colocación de las interfaces en la cadena INTA determina la prioridad de sus correspondiente periféricos.

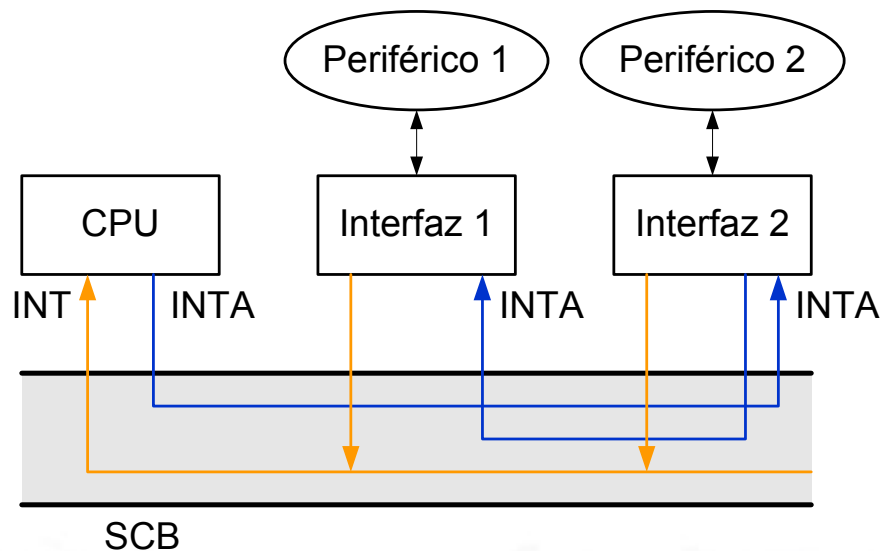


Entrada/Salida mediante interrupciones



Entrada/Salida mediante interrupciones

- Ejercicio:
 - La interfaz1, cuyo identificador es 1, y la interfaz2, cuyo identificador es 2, generan una petición de interrupción simultáneamente. ¿Qué valor aparecerá en el bus de datos tras la activación de la línea INTA?



Entrada/Salida mediante interrupciones

- Implementación de una rutina de interrupción:
 - Igual que un procedimiento normal pero terminado en **IRET**.

```
ORIGEN 0100h
.CODIGO
...

PROCEDIMIENTO rutina
...

IRET
FINP

FIN
```

Entrada/Salida mediante interrupciones

- Instalación de una rutina de interrupción:
 - Proceso mediante el cual se asocia un vector de interrupción a la rutina y se carga dicho vector con la dirección de memoria donde comienza la rutina.

ORIGEN 0100h

.CODIGO

MOVL R0, 3
MOVH R0, 0

} Número de vector

MOVL R1, BYTEBAJO DIRECCION rutina

MOVH R1, BYTEALTO DIRECCION rutina

} Dirección
de comienzo
de la rutina

MOV [R0], R1

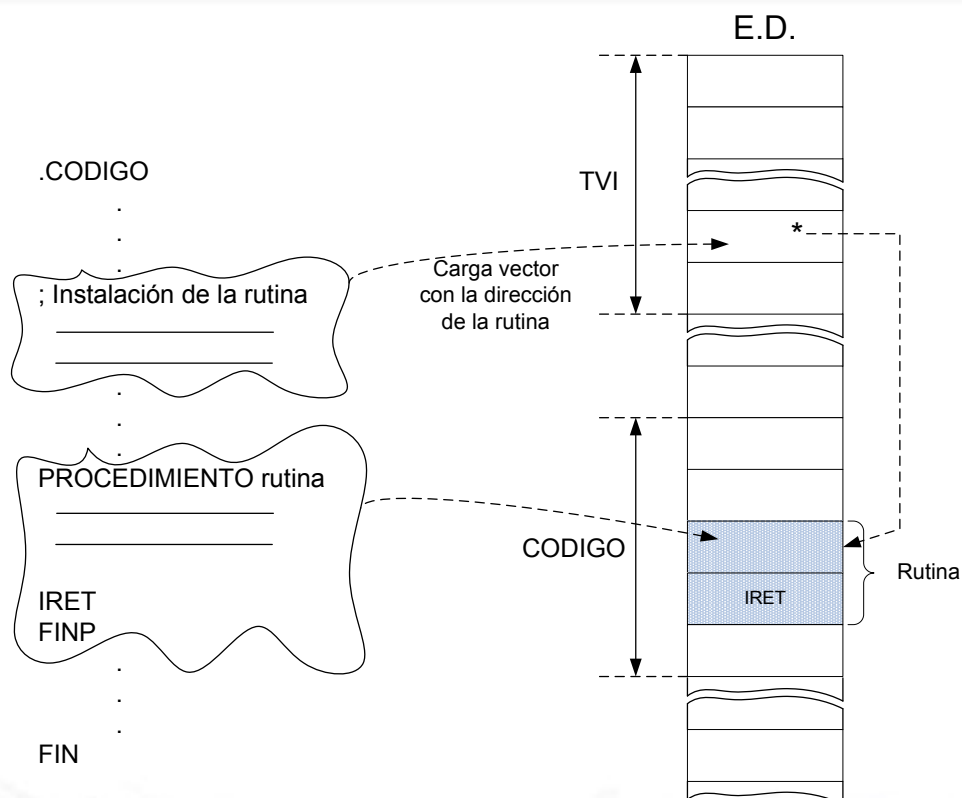
STI } Habilita las interrupciones

...

FIN



Entrada/Salida mediante interrupciones



Entrada/Salida mediante interrupciones

- Ejercicio:
 - Escribir un programa estructurado en dos partes: un programa principal y una rutina de interrupción.
 - Programa principal:
 1. Instalar la rutina de interrupción que debe estar asociada al vector 3.
 2. Ejecutar infinitamente un bucle en el que se impriman en la esquina superior izquierda de la pantalla los números del 9 al 0 en orden decreciente.
 - Rutina de interrupción:
 - Borra los caracteres impresos en la esquina superior izquierda de la pantalla y retorna.
 - La interfaz de vídeo está mapeada a partir de la dirección F100h.



Entrada/Salida mediante interrupciones

```
ORIGEN 300h
.PILA 20h
.CODIGO

    MOVL R0, 0      ; Puntero a la memoria
    MOVH R0, 0F1h   ; de video

; Instalar rutina
    MOVL R1, 3
    MOVH R1, 0
    MOVL R2, BYTEBAJO DIRECCION RutinaInt
    MOVH R2, BYTEALTO DIRECCION RutinaInt
    MOV [R1], R2
    STI
```

```
    ; Bucle infinito
Reinicia:
    MOVL R1, '9'
    MOVH R1, 07h   ; blanco
                                ; sobre
                                ; negro

    MOVL R2, 10    ; Contador
    MOVH R2, 0

Sigue:
    MOV [R0], R1
    DEC R1
    DEC R2
    BRNZ Sigue
    JMP Reinicia
```



PROCEDIMIENTO RutinaInt

```
PUSH R0
PUSH R1
MOVL R0, 0      ; Puntero a la memoria de vídeo
MOVH R0, 0F1h
MOVL R1, ' '    ; Carácter blanco
MOVH R1, 07h    ; Atributo blanco sobre negro
MOV [R0], R1    ; Borrar esquina izquierda
POP R1
POP R0
IRET
```

FINP

FIN

