

# Apéndice: circuitos integrados E/S

Estructura de Computadores

## Bibliografía:

Hojas de datos (datasheets) de los circuitos integrados (disponibles en SWAD)

# Apéndice: circuitos integrados E/S

- **Interfaz de periféricos programable 8255**
- **Controlador de interrupciones programable 8259**
  - Interrupciones en el PC en modo real
- **Controlador de DMA programable 8237**

# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ Interfaz de periféricos programable 8255 *(Programmable Peripheral Interface, PPI)*



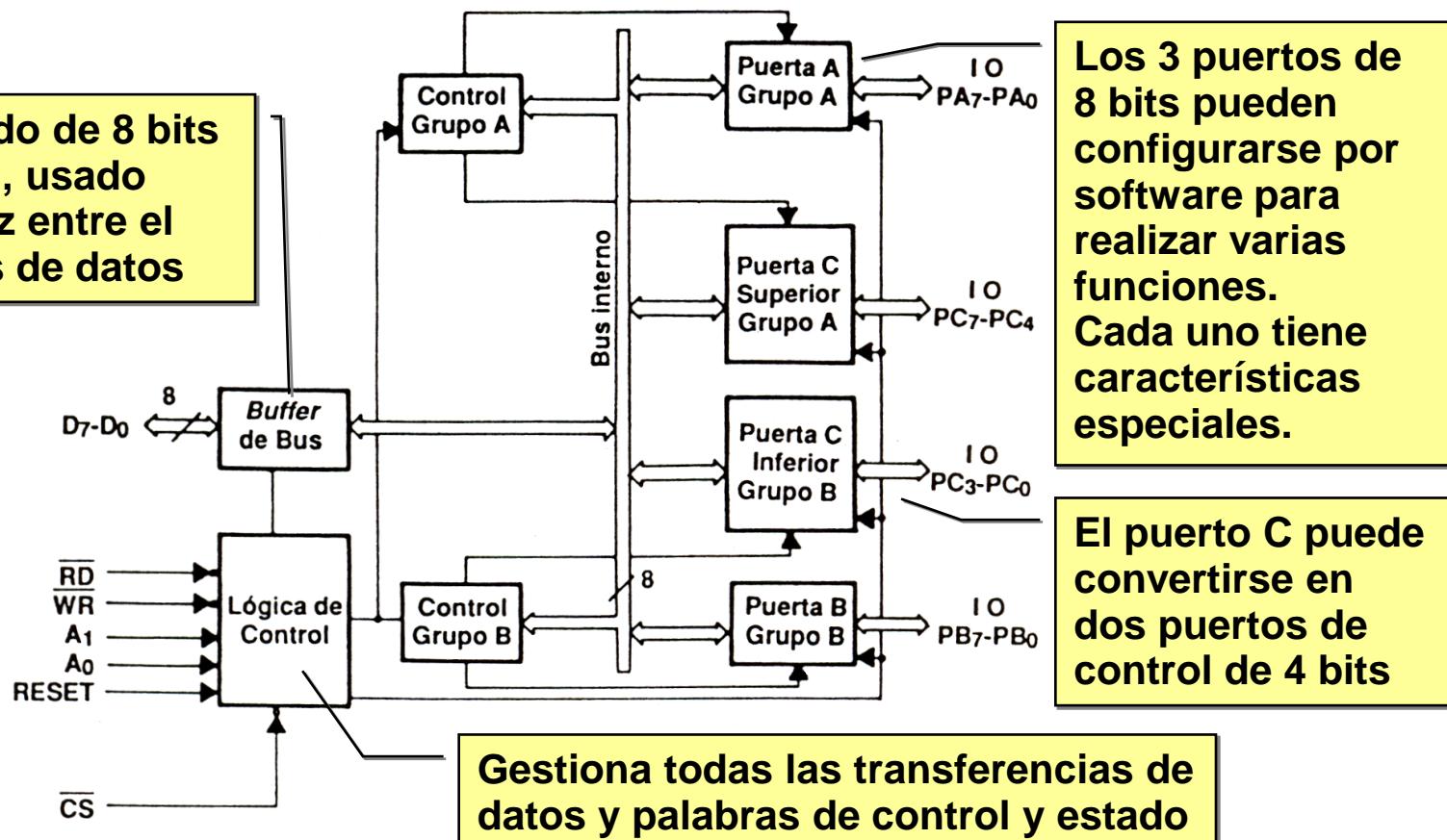
PA3	1	PA4	40
PA2	2	39	PA5
PA1	3	38	PA6
PA0	4	37	PA6
RD	5	36	WR
CS	6	35	RESET
GND	7	34	D0
A1	8	33	D1
A0	9	32	D2
PC7	10	8255A	31
PC6	11		D3
PC5	12		30
PC4	13		D4
PC0	14		29
PC1	15		D5
PC2	16		28
PC3	17		D6
PB0	18		27
PB1	19		D7
PB2	20		26
			Vcc
			25
			PB7
			24
			PB6
			23
			PB5
			22
			PB4
			21
			PB3

- Permite gestionar tres puertos de E/S de 8 bits.
- Se pueden seleccionar por software tres modos de funcionamiento:
  - E/S programada sin validación:
    - *Modo 0*: E/S básica (programada).
  - E/S programada o por interrupciones con validación (handshaking).
    - *Modo 1*: E/S con validación.
    - *Modo 2*: E/S con bus bidireccional, validada.

# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ Esquema general

- Permite gestionar 3 puertos de E/S de 8 bits.



# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ Asignación de patillas

**A0, A1:** Seleccionan el puerto del que se va a leer o escribir

A <sub>1</sub>	A <sub>0</sub>	RD	WR	CS	INPUT OPERATION (READ)
0	0	0	1	0	PORT A → DATA BUS
0	1	0	1	0	PORT B → DATA BUS
1	0	0	1	0	PORT C → DATA BUS
1	1	0	1	0	CONTROL WORD → DATA BUS
OUTPUT OPERATION (WRITE)					
0	0	1	0	0	DATA BUS → PORT A
0	1	1	0	0	DATA BUS → PORT B
1	0	1	0	0	DATA BUS → PORT C
1	1	1	0	0	DATA BUS → CONTROL
DISABLE FUNCTION					
x	x	x	x	1	DATA BUS → 3-STATE
x	x	1	1	0	DATA BUS → 3-STATE

**PC0, PC7:** Puerto C de Entrada/Salida

PA3	1	40	PA4
PA2	2	39	PA5
PA1	3	38	PA6
<b>PA0</b>	<b>4</b>	<b>37</b>	<b>PA7</b>
RD	5	36	WR
CS	6	35	RESET
gnd	7	34	D0
A1	8	33	D1
<b>A0</b>	<b>9</b>	<b>32</b>	<b>D2</b>
PC7	10	8255	31
PC6	11	PPI	30
PC5	12		D4
PC4	13		D5
PC0	14		D6
PC1	15		D7
PC2	16		Vcc
PC3	17		PB7
PB0	18	23	PB6
PB1	19	22	PB5
PB2	20	21	PB4
			PB3

**PA0, PA7:** Puerto A de Entrada/Salida

**PB0, PB7:** Puerto B de Entrada/Salida

# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ Asignación de patillas

**RD# (Read): =0** habilita al 8255 para que envíe la información de datos o de estado a la CPU a través del bus de datos

**CS# (Chip Select): =0** habilita la comunicación entre el 8255 y el procesador

PA3	1	40	PA4
PA2	2	39	PA5
PA1	3	38	PA6
<b>PA0</b>	4	37	<b>PA7</b>
RD	5	36	WR
CS	6	35	RESET
gnd	7	34	D0
A1	8	33	D1
A0	9	32	D2
PC7	10	8255	D3
PC6	11	PPI	D4
PC5	12		D5
PC4	13		D6
PC0	14		D7
PC1	15		Vcc
PC2	16		PB7
PC3	17		PB6
PB0	18		PB5
PB1	19		PB4
PB2	20		PB3

**WR# (Write): =0** permite que la CPU escriba datos o palabras de control en el 8255

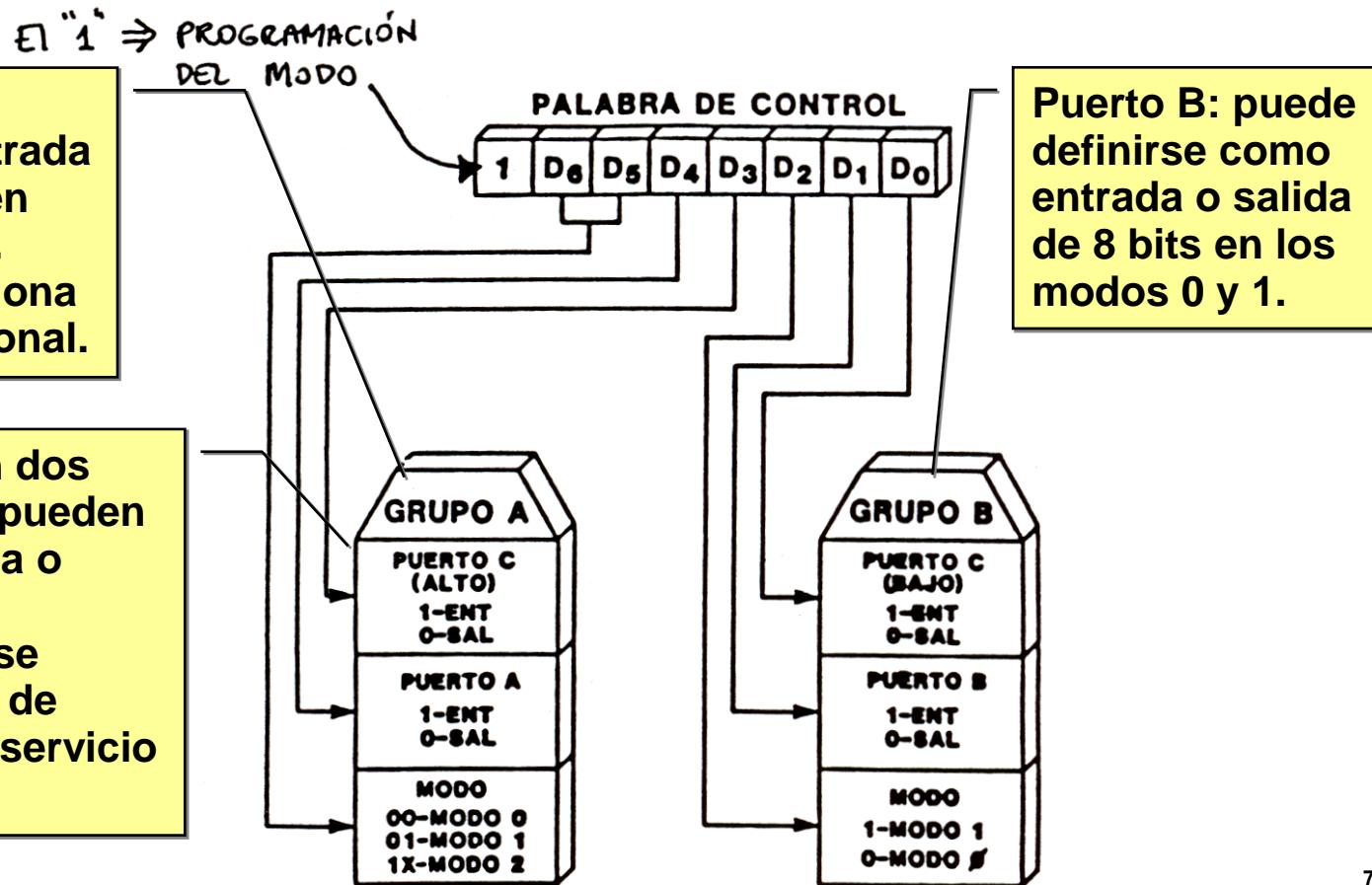
**RESET:** =1 borra el registro de control y pone los puertos en modo de entrada

**D0-D7:** conexión al bus de datos

# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ Palabra de control (bit 7 = 1)

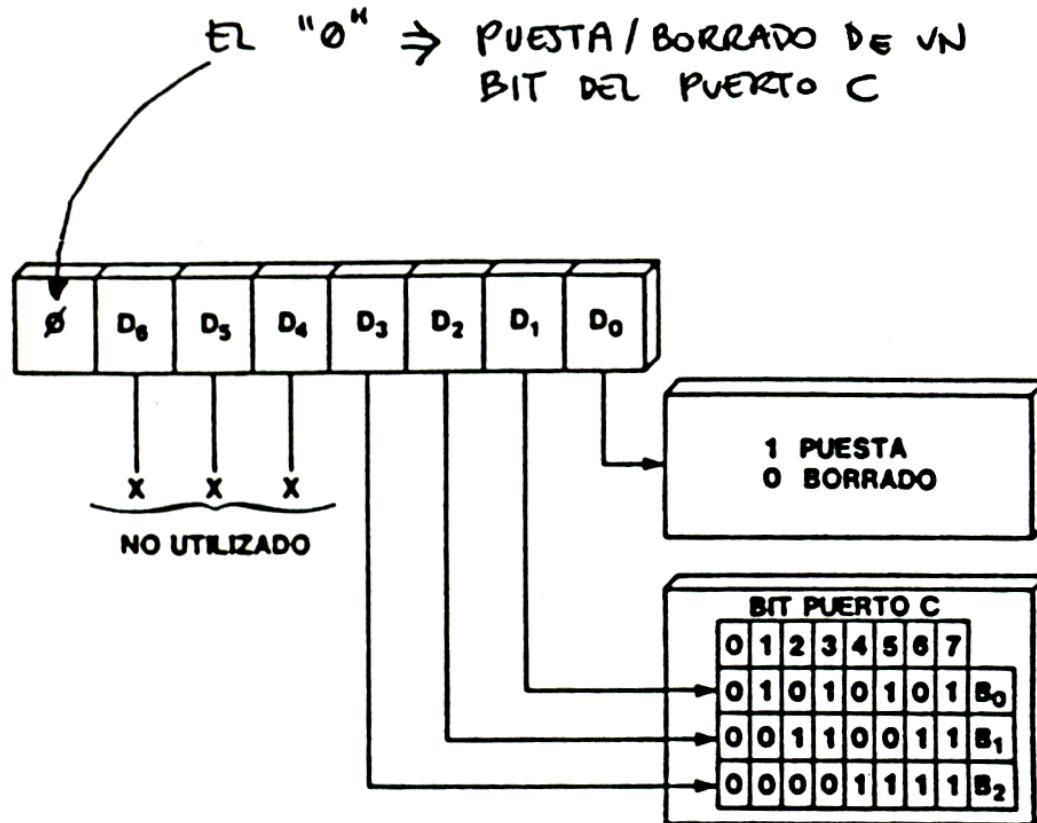
- Se puede programar la configuración funcional de cada puerto escribiendo una palabra de control en el 8255.



# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ Palabra de control (bit 7 = 0)

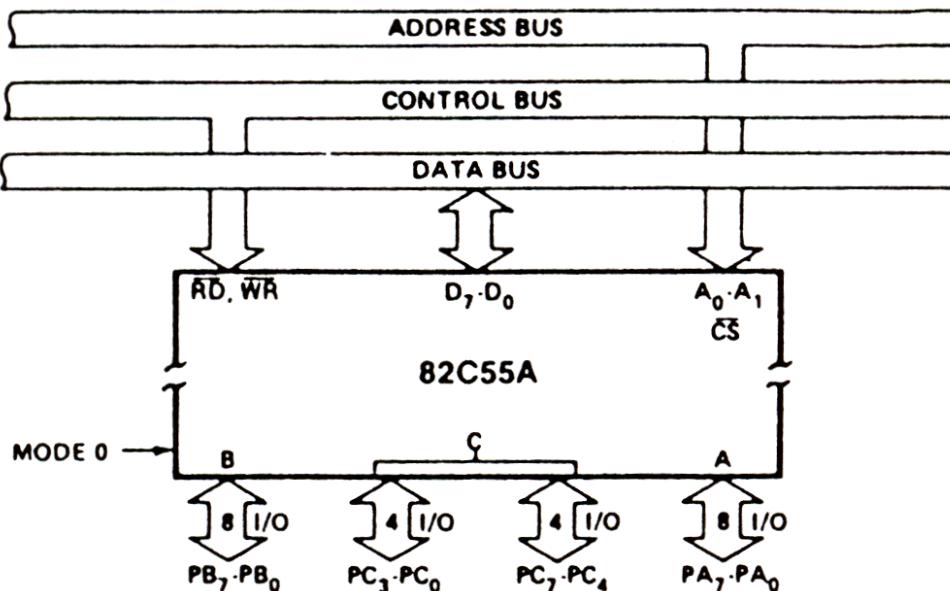
- Permite activar de forma individual cada bit del puerto C
- Es la única forma de escribir en C en los modos 1 y 2.



# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ MODO 0 (E/S básica)

- No requiere *handshaking*, sino que los datos son simplemente leídos o escritos de un puerto.
- Cada puerto puede programarse independientemente como de entrada o de salida (el puerto C se programa como dos puertos independientes de 4 bits).
- Cada bit puede utilizarse para datos o para señales de control o de estado.

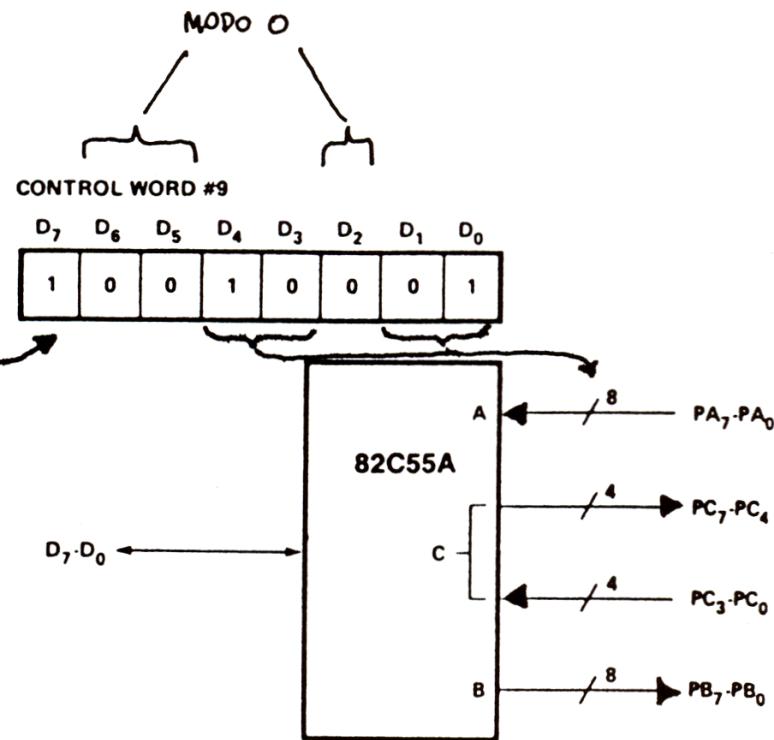


# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ MODO 0

- Según la aplicación deseada, cada puerto deberá declararse como entrada o como salida, escribiendo la oportuna palabra de control.

A		B		GROUP A			#	GROUP B	
D <sub>4</sub>	D <sub>3</sub>	D <sub>1</sub>	D <sub>0</sub>	PORT A	PORT C (UPPER)		PORT B	PORT C (LOWER)	
0	0	0	0	OUTPUT	OUTPUT	0	OUTPUT	OUTPUT	
0	0	0	1	OUTPUT	OUTPUT	1	OUTPUT	INPUT	
0	0	1	0	OUTPUT	OUTPUT	2	INPUT	OUTPUT	
0	0	1	1	OUTPUT	OUTPUT	3	INPUT	INPUT	
0	1	0	0	OUTPUT	INPUT	4	OUTPUT	OUTPUT	
0	1	0	1	OUTPUT	INPUT	5	OUTPUT	INPUT	
0	1	1	0	OUTPUT	INPUT	6	INPUT	OUTPUT	
0	1	1	1	OUTPUT	INPUT	7	INPUT	INPUT	
1	0	0	0	INPUT	OUTPUT	8	OUTPUT	OUTPUT	
1	0	0	1	INPUT	OUTPUT	9	OUTPUT	INPUT	
1	0	1	0	INPUT	OUTPUT	10	INPUT	OUTPUT	
1	0	1	1	INPUT	OUTPUT	11	INPUT	INPUT	
1	1	0	0	INPUT	INPUT	12	OUTPUT	OUTPUT	
1	1	0	1	INPUT	INPUT	13	OUTPUT	INPUT	
1	1	1	0	INPUT	INPUT	14	INPUT	OUTPUT	
1	1	1	1	INPUT	INPUT	15	INPUT	INPUT	

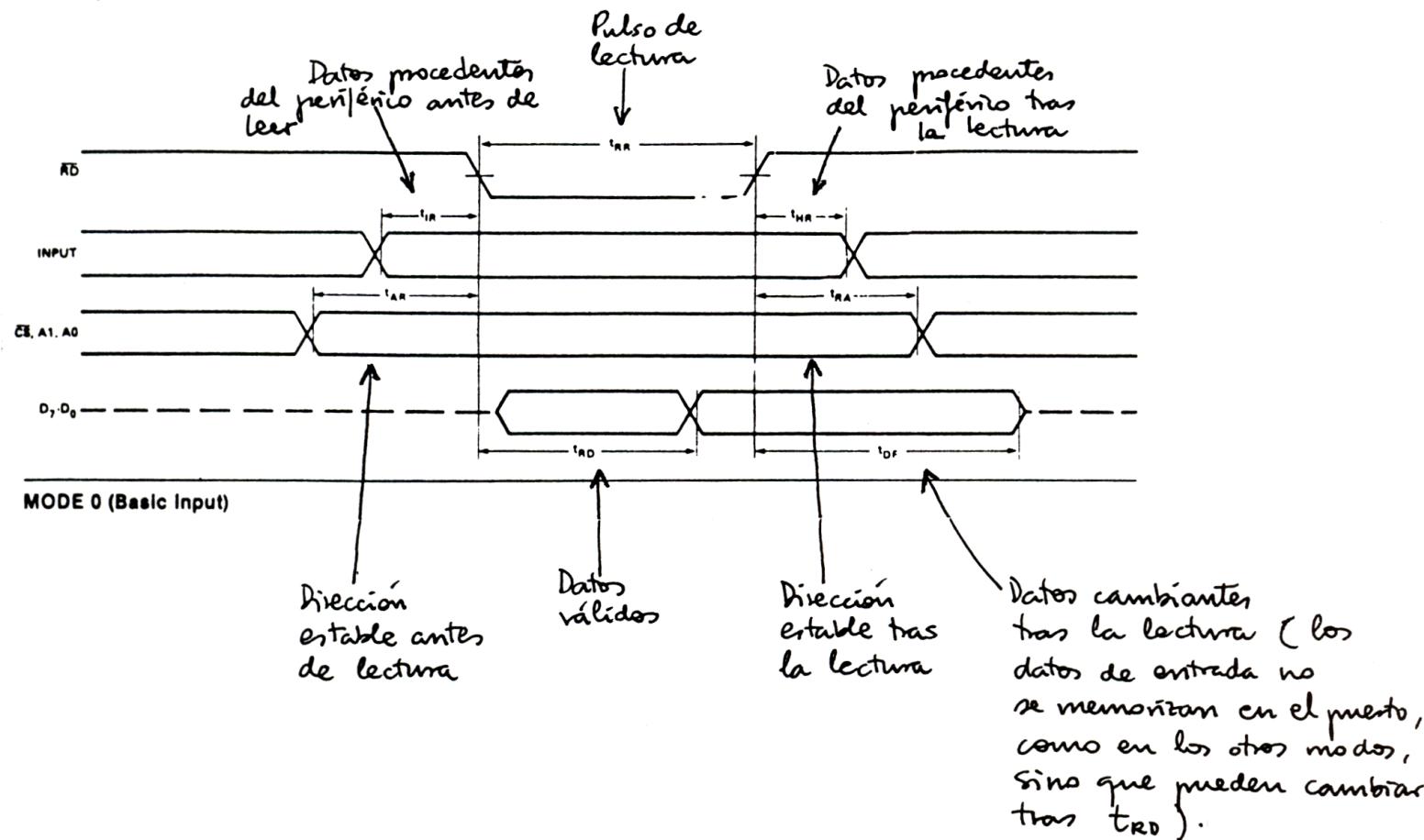


# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ MODO 0

- Temporización (entrada):

ENTRADA :

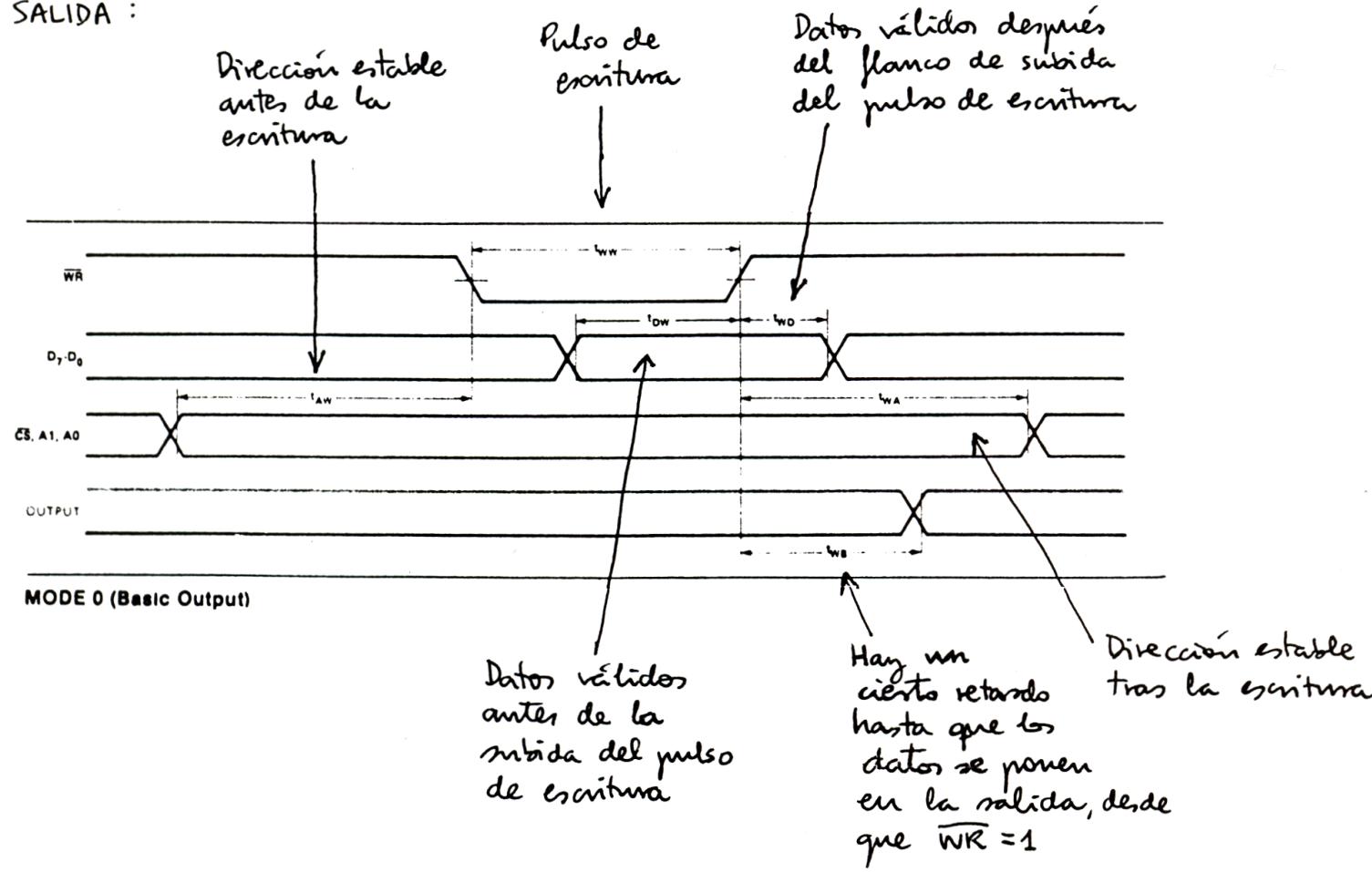


# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ MODO 0

- Temporización (salida):

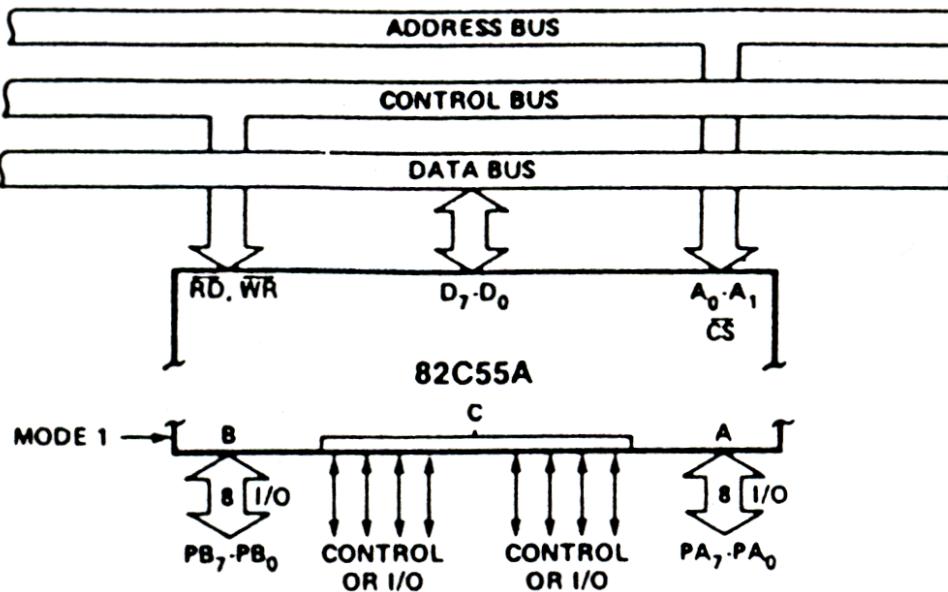
SALIDA :



## Ejemplo de circuito integrado de interfaz paralela: 8255

- **MODO 1 (E/S validada)**

- Puede utilizarse E/S en modo programado o por interrupciones.
- Se van a transferir datos usando señales de diálogo (*handshaking*) entre el 8255 y el periférico (¡yojo!, no entre el 8255 y la CPU).
- El puerto A y el puerto B se pueden programar como entrada o como salida, y el puerto C se usa tanto para control de A y B como para E/S en modo 0.



# Ejemplo de circuito integrado de interfaz paralela: 8255

## SEÑALES DE DIÁLOGO DEL 8255 EN MODO 1

	para una entrada de datos	para una salida de datos
«dato preparado»	$\overline{\text{STB}}$ { PC4 para el grupo A Strobe PC2 para el grupo B	$\overline{\text{OBF}}$ { PC7 para el grupo A Ouput Buffer Full PC1 para el grupo B
«dato recibido»	$\overline{\text{IBF}}$ { PC5 para el grupo A Input Buffer Full PC1 para el grupo B	$\overline{\text{ACK}}$ { PC6 para el grupo A Acknowledge PC2 para el grupo B
«petición de Intermup.»	$\overline{\text{INTR}}$ { PC3 para el grupo A Interrupt Request PC0 para el grupo B	

# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ MODO 1

- Es posible impedir o autorizar al 8255 interrumpir al procesador, poniendo a 0 ó 1 ciertas señales INTE (*Interrupt Enable*):

«habilitación de interrup.»	INTE { PC4 para grupo A PC2 para grupo B <i>Interrupt Enable</i>	INTE { PC6 para grupo A PC2 para grupo B
-----------------------------	---	--

- Para escribir en el puerto C se puede usar una orden de escritura en ese puerto en el modo 0, o una orden de puesta/borrado de un bit en los tres modos. Ésta última orden también se usará para modificar los biestables INTE.

## Ejemplo de circuito integrado de interfaz paralela: 8255

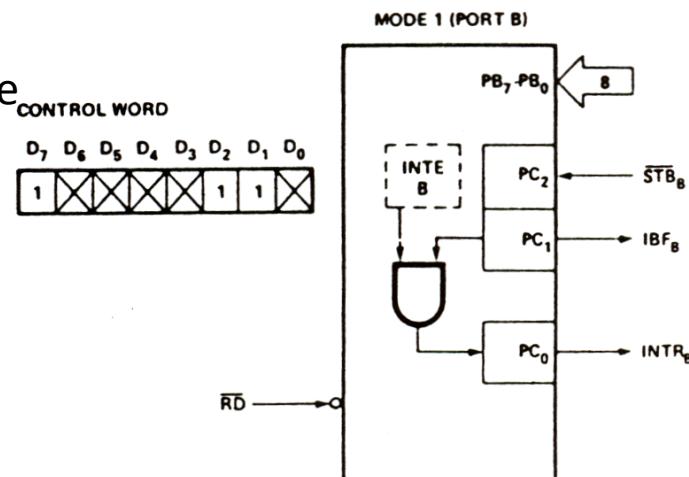
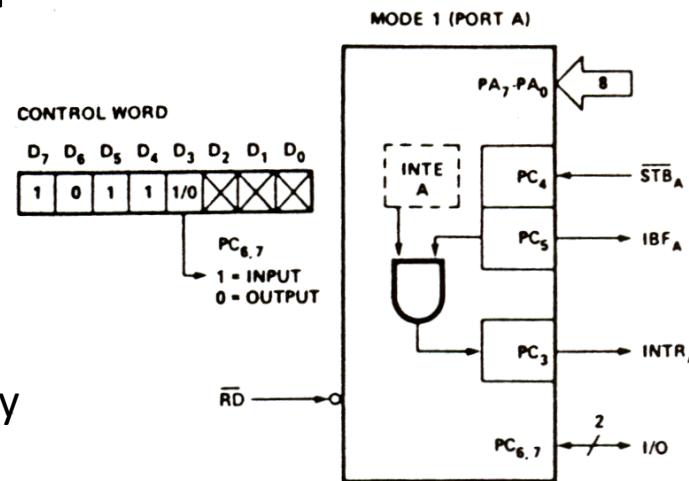
- En una lectura del puerto C en los modos 1 y 2, el estado de las líneas de este puerto pasa al bus de datos, excepto las líneas ACK# y STB#, que son reemplazadas por INTE.

Interrupt Enable Flag*	Position	Alternate Port C Pin Signal (Mode)
INTE B	PC2	$\overline{ACK_B}$ (Output Mode 1) or $\overline{STB_B}$ (Input Mode 1)
INTE A2	PC4	$\overline{STB_A}$ (Input Mode 1 or Mode 2)
INTE A1	PC6	$\overline{ACK_A}$ (Output Mode 1 or Mode 2)

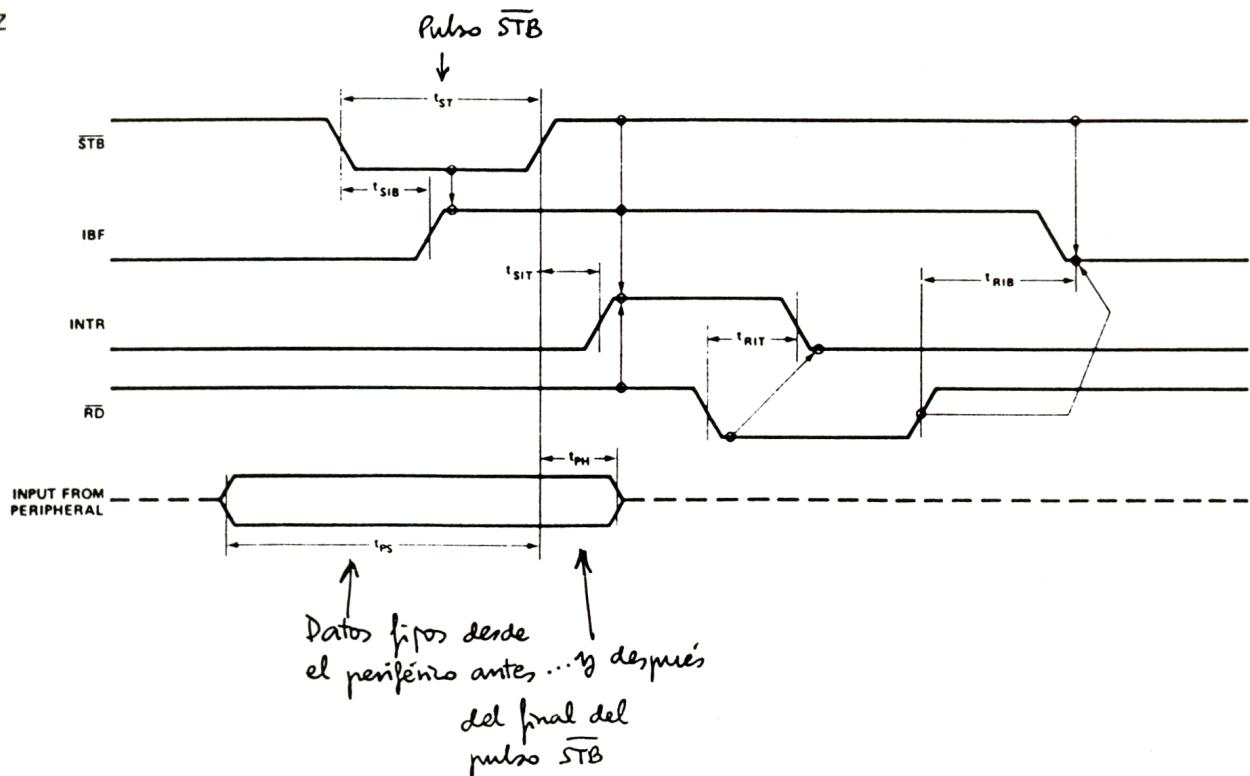
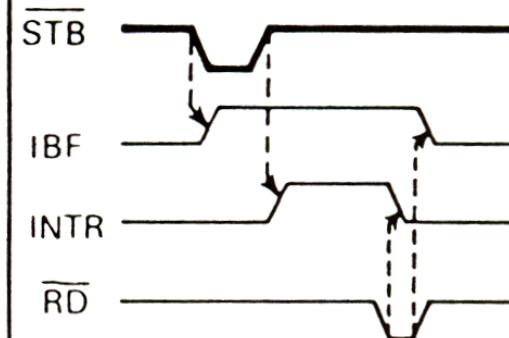
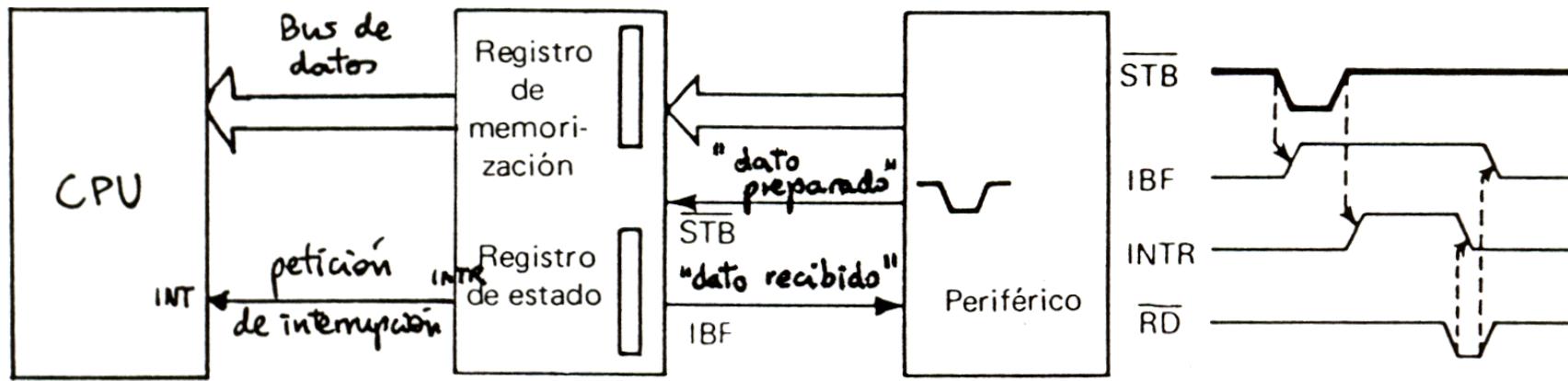
# Ejemplo de circuito integrado de interfaz paralela: 8255

## MODO 1 en entrada

1. Cuando el periférico tiene un dato para transmitir a la CPU, envía la señal STB#, que indica “dato preparado”, al 8255.
  2. El 8255 sube a 1 la señal IBF (“dato recibido”), y memoriza los datos de entrada (en modo 0 no se memorizan).
  3. El flanco de subida de STB# pone a 1 la señal de petición de interrupción INTR, si está habilitada. Puede tener lugar una interrupción si además la línea está conectada a la CPU.
  4. La señal de lectura generada por la CPU hace bajar las señales INTR e IBF.
- Se podría usar modo programado, no conectando INTR o desactivándola, y leyendo y comprobando los bits IBF hasta que estuvieran a 1, para después leer del puerto A o B.



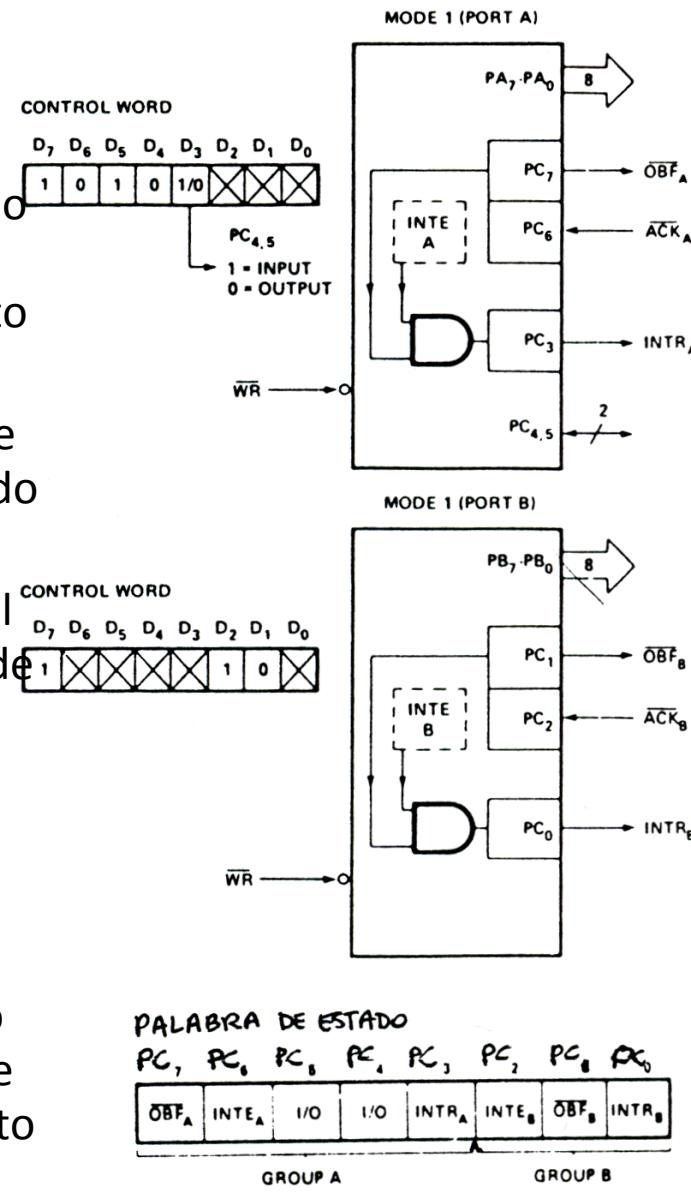
# Ejemplo de circuito integrado de interfaz paralela: 8255



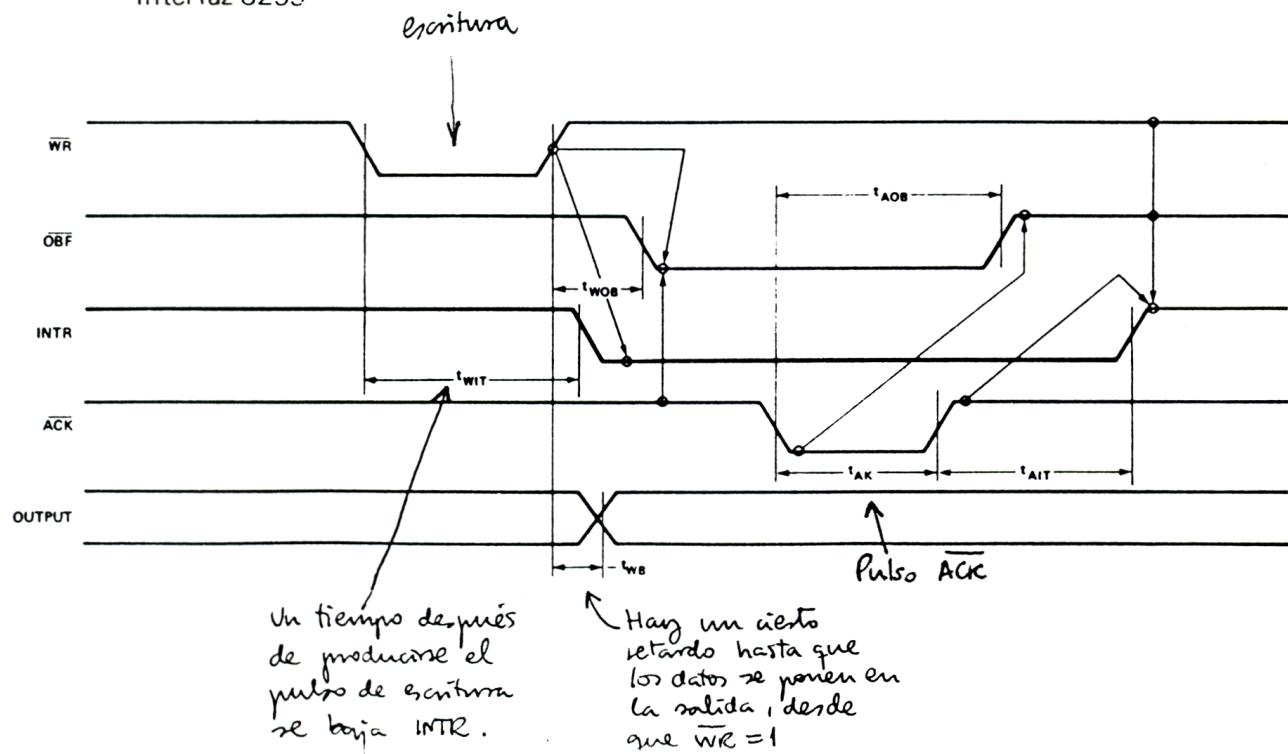
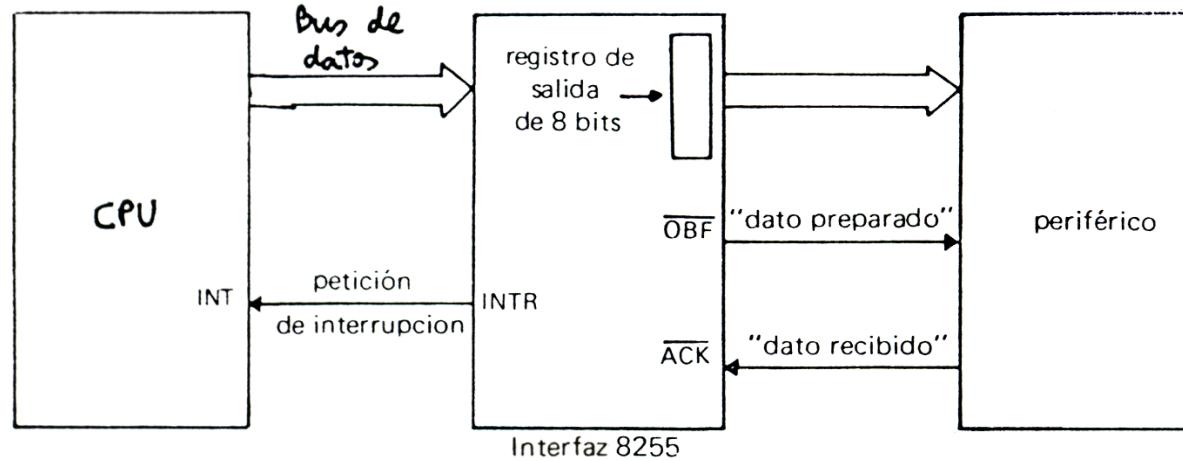
# Ejemplo de circuito integrado de interfaz paralela: 8255

## MODO 1 en salida

1. La escritura ( $WR\#=0$ ) de un dato en el puerto A o B hace que se active la señal  $OBF\#$  para comunicar al perif. que la CPU ha puesto un dato en el 8255 listo para ser enviado a dicho perif.
  2. Tras un tiempo más o menos largo que depende del periférico, éste activa la señal  $ACK\#$  indicando “dato aceptado”.
  3. Cuando el periférico pone de nuevo  $ACK\#$  a 1, el 8255 activa la línea  $INTR$  si está habilitada. Puede tener lugar una interrupción si además la línea está conectada a la CPU.
  4. La siguiente señal de escritura generada por la CPU hace bajar la señal  $INTR$ .
- Se podría usar modo programado para comprobar cuando el periférico ha leído el dato leyendo y comprobando los bits  $OBF\#$  hasta que estuvieran a 1, para después escribir en el puerto A o B.



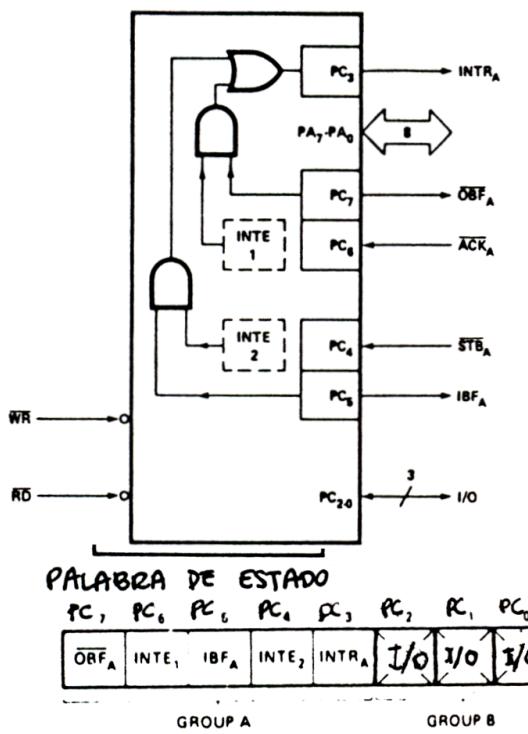
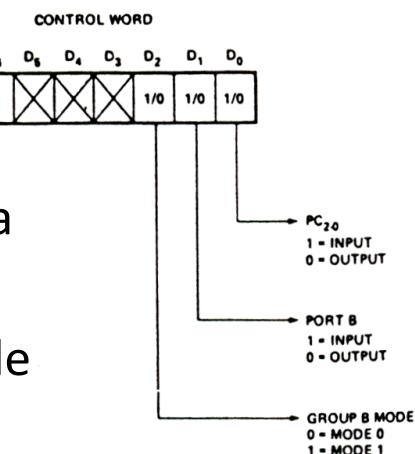
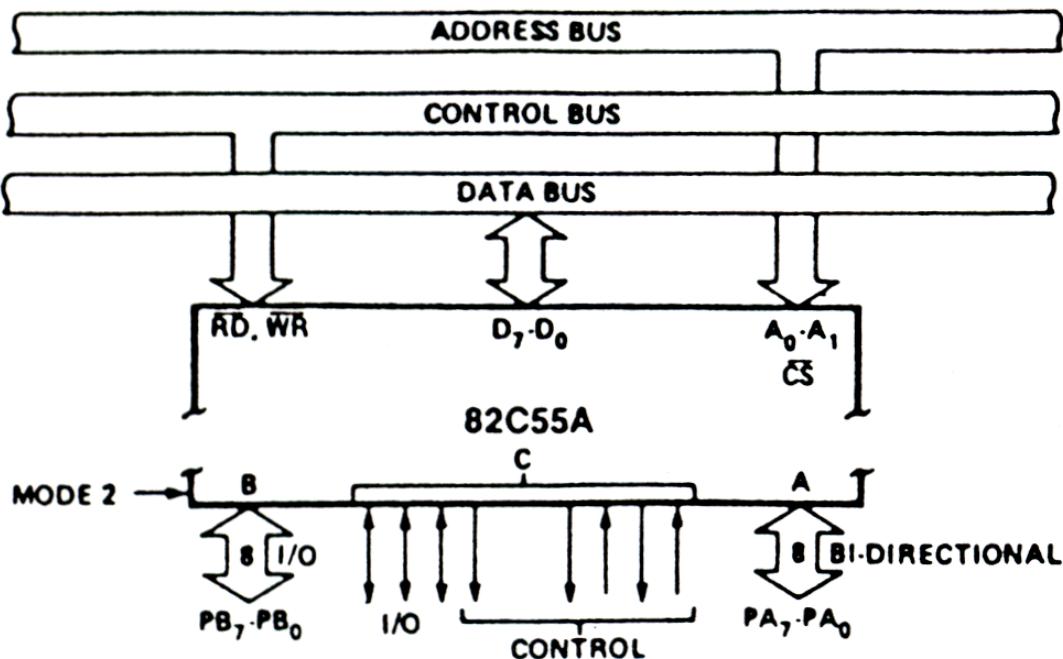
# Ejemplo de circuito integrado de interfaz paralela: 8255



# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ MODO 2 (E/S validada con bus bidireccional)

- Está previsto para un intercambio bidireccional entre la interfaz y el periférico (transmitir y recibir).
- Es similar al modo 1 en lo que concierne a las señales de diálogo.
- Sólo el puerto A puede funcionar en modo 2.

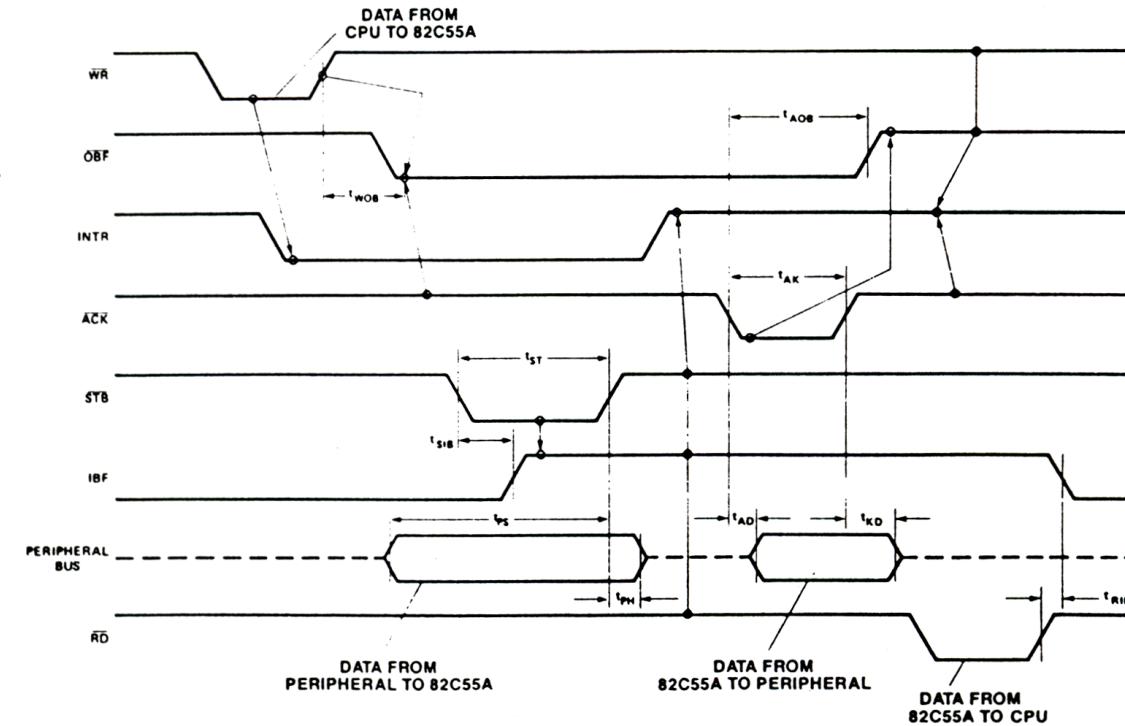


# Ejemplo de circuito integrado de interfaz paralela: 8255

## MODO 2

- En el modo 1 el 8255 escribía datos en el periférico en el flanco de subida de WR#. En el modo 2 el 8255 escribe datos en el periférico en el flanco de bajada de ACK#.
- El puerto A está en alta impedancia hacia el periférico cuando ACK# = 1.
- ACK# no indica “dato aceptado” sino perif. “listo para recibir datos”.

Cualquier secuencia en la que WR# ocurra antes que ACK# y STB# ocurra antes que RD# es permisible.



# Ejemplo de circuito integrado de interfaz paralela: 8255

Puerto A      Puerto B      Puerto C

	MODE 0	
	IN	OUT
PA <sub>0</sub>	IN	OUT
PA <sub>1</sub>	IN	OUT
PA <sub>2</sub>	IN	OUT
PA <sub>3</sub>	IN	OUT
PA <sub>4</sub>	IN	OUT
PA <sub>5</sub>	IN	OUT
PA <sub>6</sub>	IN	OUT
PA <sub>7</sub>	IN	OUT

	MODE 1	
	IN	OUT
PB <sub>0</sub>	IN	OUT
PB <sub>1</sub>	IN	OUT
PB <sub>2</sub>	IN	OUT
PB <sub>3</sub>	IN	OUT
PB <sub>4</sub>	IN	OUT
PB <sub>5</sub>	IN	OUT
PB <sub>6</sub>	IN	OUT
PB <sub>7</sub>	IN	OUT

	MODE 2	
	GROUP A ONLY	
PC <sub>0</sub>		
PC <sub>1</sub>		
PC <sub>2</sub>		
PC <sub>3</sub>		
PC <sub>4</sub>		
PC <sub>5</sub>		
PC <sub>6</sub>		
PC <sub>7</sub>		

	MODE 1	
	IN	OUT
INTR <sub>B</sub>	INTR <sub>B</sub>	I/O
IBF <sub>B</sub>	<u>OBF<sub>B</sub></u>	I/O
STB <sub>B</sub>	ACK <sub>B</sub>	I/O
INTRA	INTRA	I/O
STB <sub>A</sub>	I/O	I/O
IBFA	I/O	I/O
I/O	ACKA	I/O
I/O	<u>OBFA</u>	I/O

	MODE 2	
	GROUP A ONLY	
PC <sub>0</sub>		
PC <sub>1</sub>		
PC <sub>2</sub>		
PC <sub>3</sub>		
PC <sub>4</sub>		
PC <sub>5</sub>		
PC <sub>6</sub>		
PC <sub>7</sub>		

Tabla comparativa de los modos 0, 1 y 2.

MODE 0  
OR MODE 1  
ONLY

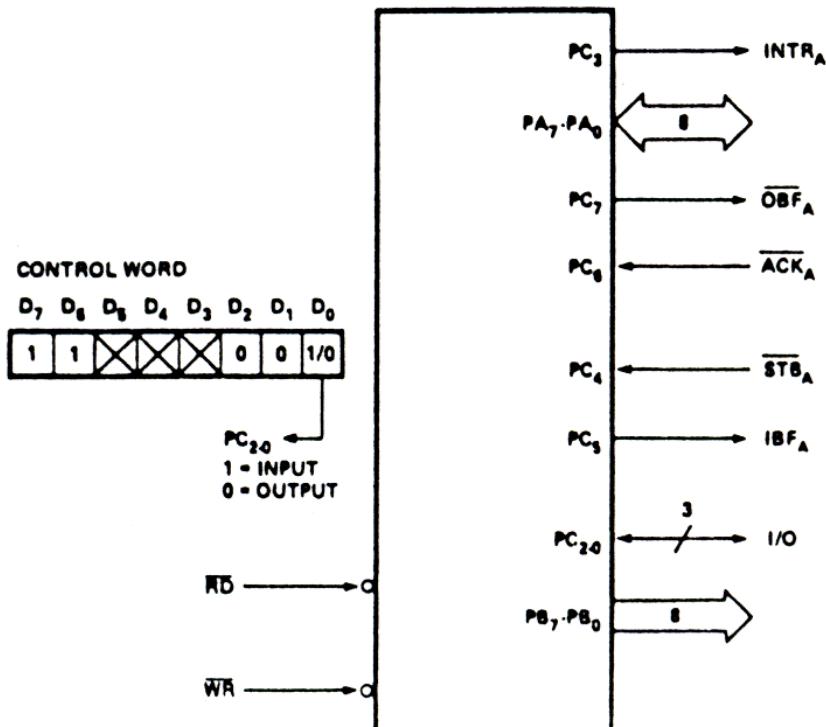
Si el puerto B está en modo 0

# Ejemplo de circuito integrado de interfaz paralela: 8255

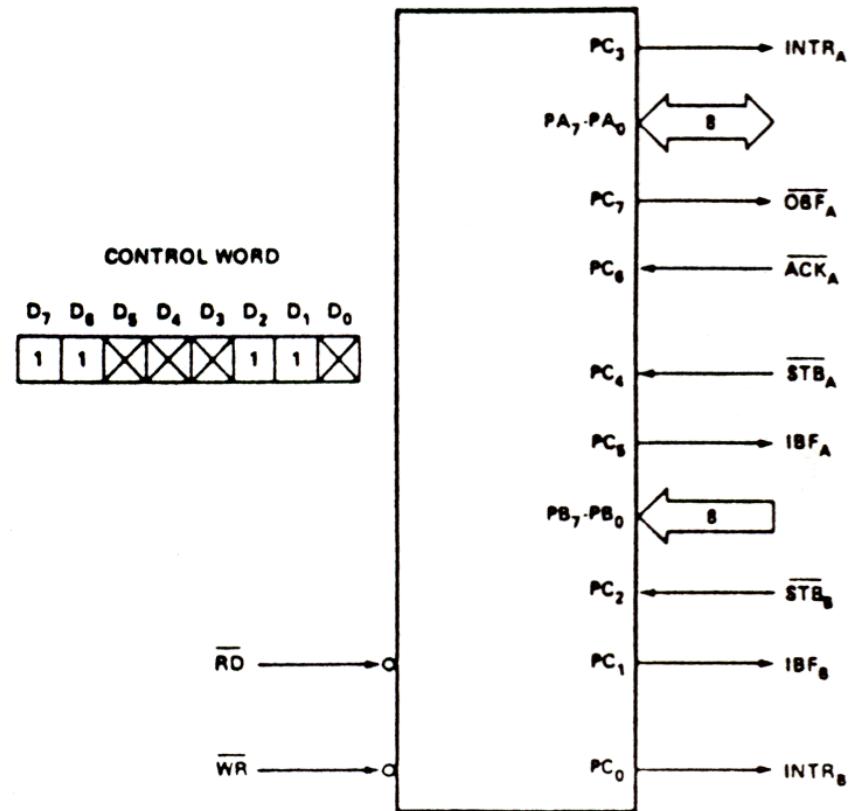
## ■ Combinación de modos

- El 8255 puede tener un puerto programado en cada modo.
- Ejemplos:

MODE 2 AND MODE 0 (OUTPUT)

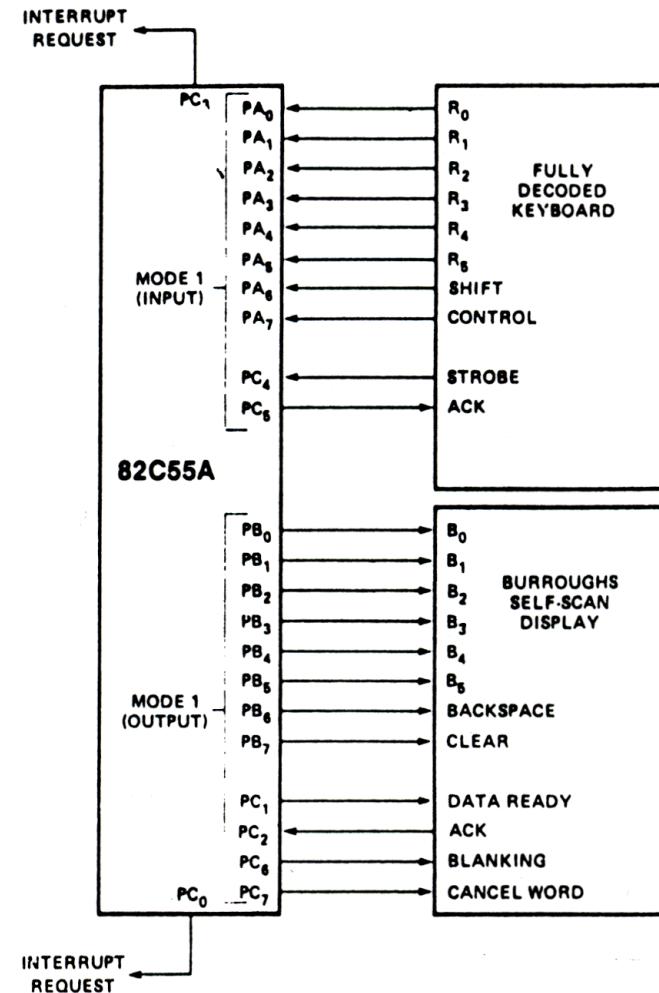
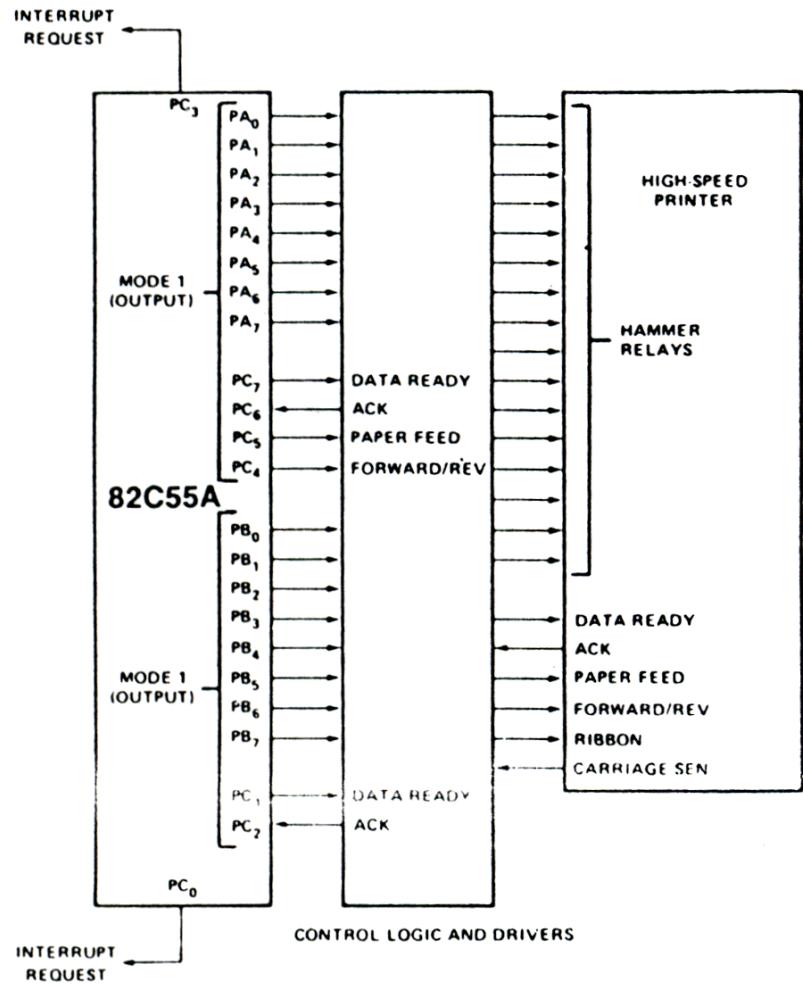


MODE 2 AND MODE 1 (INPUT)



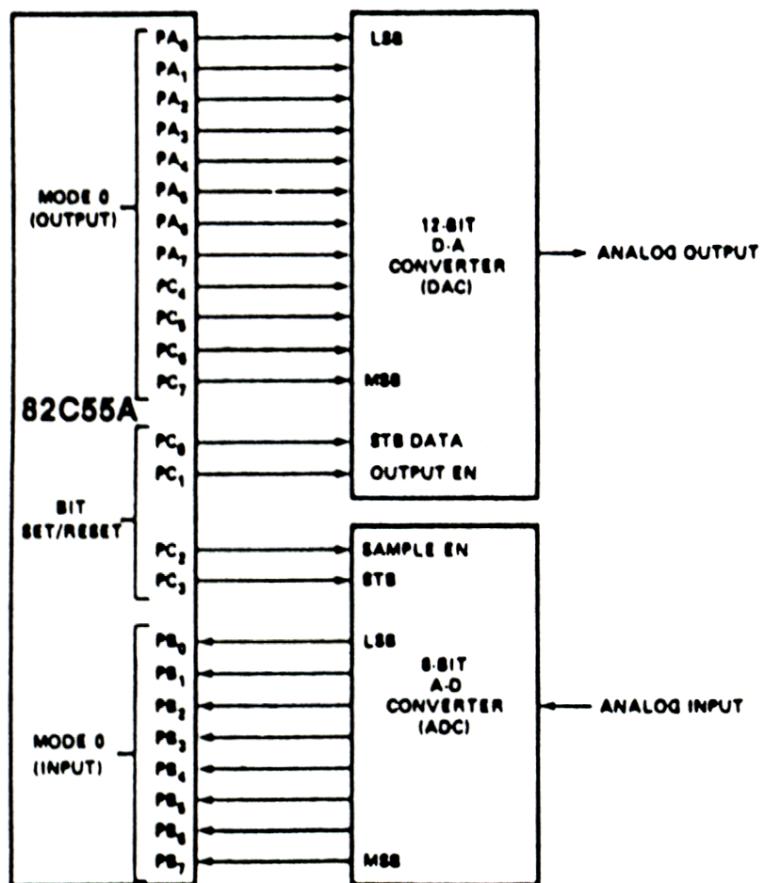
# Ejemplo de circuito integrado de interfaz paralela: 8255

- Interfaz de impresora:
- Interfaz de teclado y display:

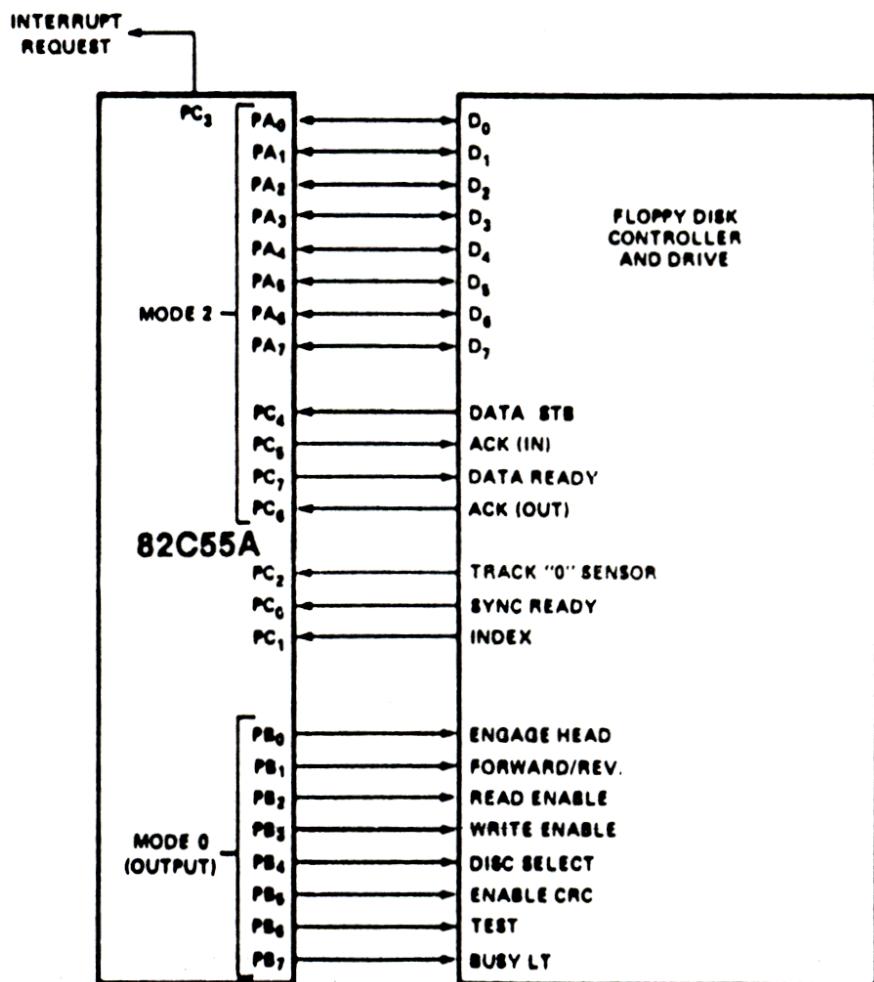


# Ejemplo de circuito integrado de interfaz paralela: 8255

- Interfaz con conversores D/A y A/D:

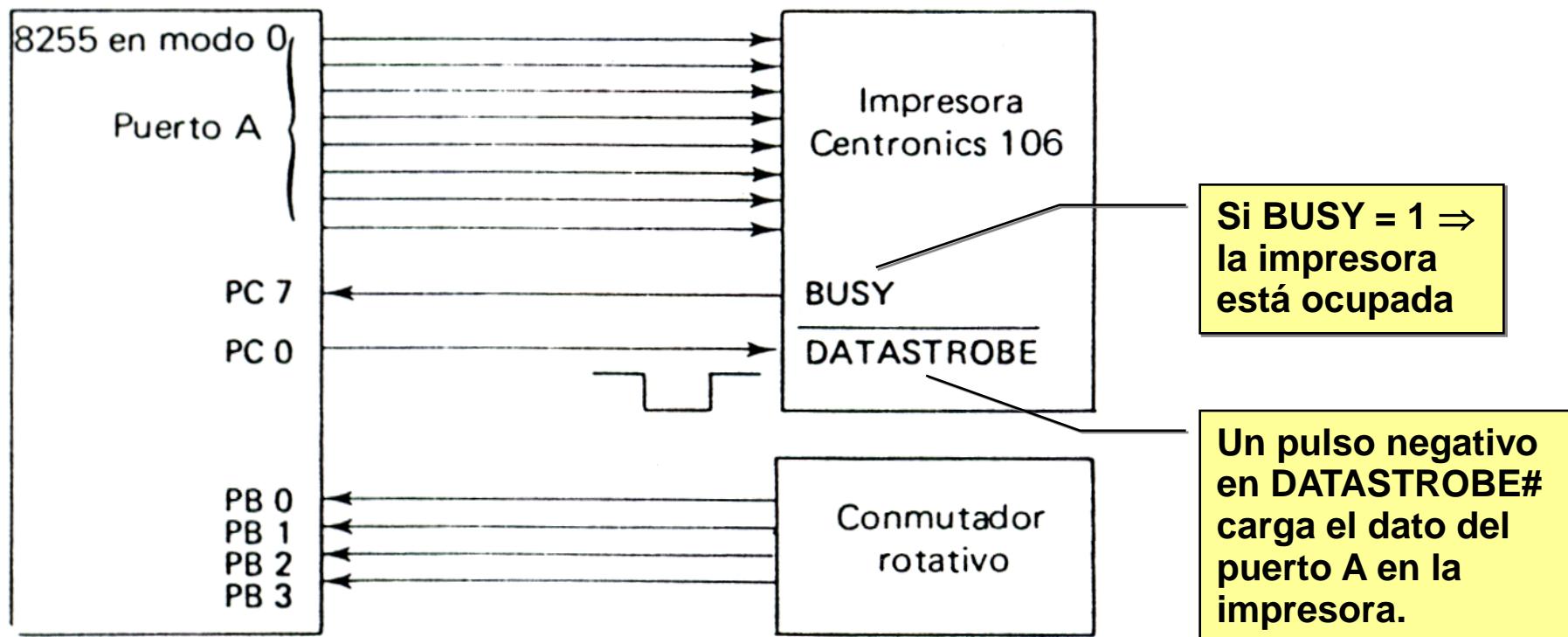


- Interfaz de disquetera:



## Ejemplo de circuito integrado de interfaz paralela: 8255

- Ej. de operación de E/S programada usando el 8255
  - Control de una impresora (salida), y de un conmutador rotativo (entrada) que permite seleccionar entre 16 valores.



# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ Ej. de operación de E/S programada usando el 8255

- Programación del 8255:

- Palabra de control:

**MVI A, 8A**

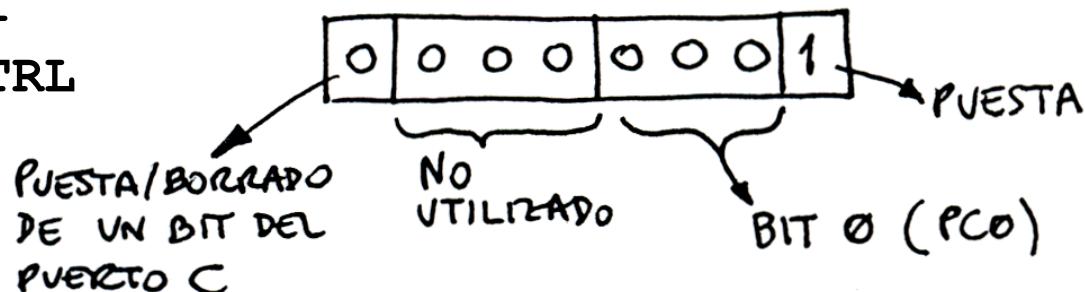
**OUT PPICTRL**



- Hay que inicializar PC0 a 1:

**MVI A, 01**

**OUT PPICTRL**

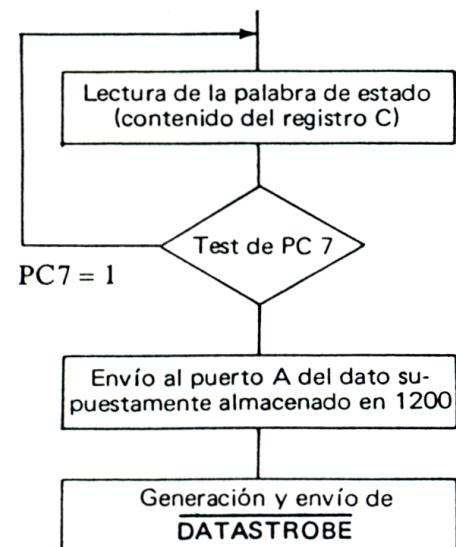


# Ejemplo de circuito integrado de interfaz paralela: 8255

## ■ Ej. de operación de E/S programada usando el 8255

- Impresión de un carácter (almacenado por ej, en M(1200)):

**BUCLE:**



IN PPIC	; A ← puerto C del 8255
ANI 80	; Test de PC7 por enmascaramiento
JNZ BUCLE	; Salto a BUCLE si BUSY = 1
LDA 1200	; A ← dato almacenado en la...
	; ...dirección 1200 de memoria
OUT PPIA	; puerto A del 8255 ← A
MVI A, 00	; Puesta a 0...
OUT PPICTRL	; ...de PC0
MVI A, 01	; Puesta a 1...
OUT PPICTRL	; ...de PC0

- Lectura del conmutador rotativo:

IN PPIB	; A ← puerto B del 8255
STA 1201	; almacenamiento en la...
	; ...dirección 1201 de memoria

# Apéndice: circuitos integrados E/S

- Interfaz de periféricos programable 8255
- Controlador de interrupciones programable 8259
  - Interrupciones en el PC en modo real
- Controlador de DMA programable 8237

# Ej. de circuito controlador de interrupciones: 8259

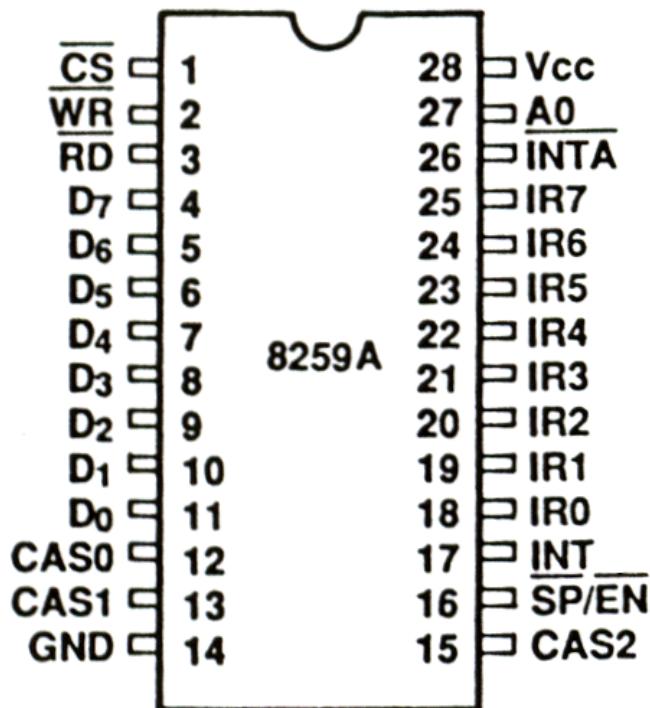
**La mayoría de las familias de microprocesadores contienen circuitos especiales denominados “controladores de interrupciones”, que**

- simplifican las diversas tareas de diseño (prioridades, enmascaramiento,...) asociadas con el control de las interrupciones externas, y
- permiten aumentar el número de líneas de petición de interrupción disponibles.

## Ej. de circuito controlador de interrupciones: 8259

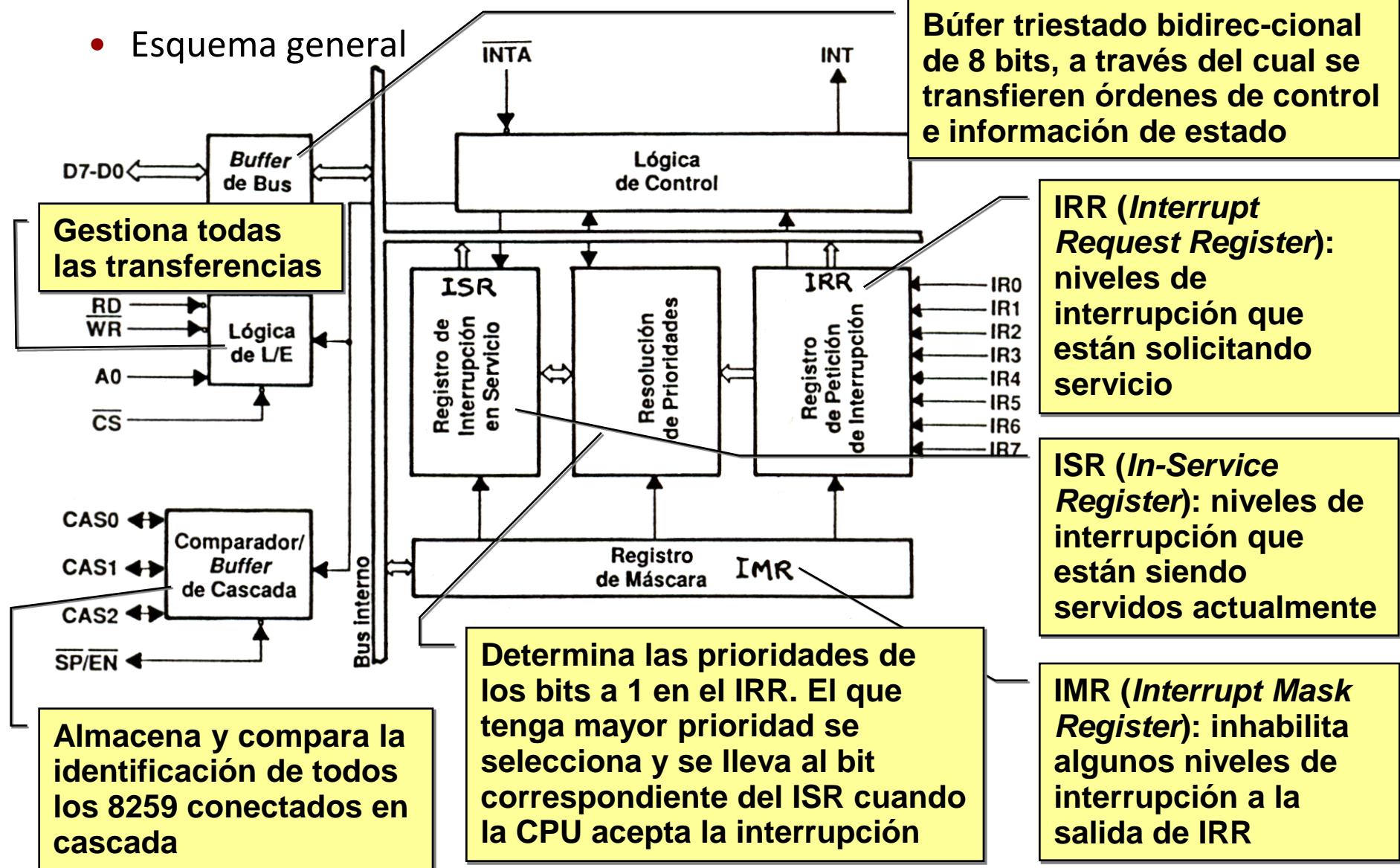
### ■ Programmable Interrupt Controller, PIC

- Permite manejar 8 líneas de interrupciones vectorizadas con prioridad.
- Compatible con las familias 8085 y 8086.
- Prioridades alterables por software.
- Permite enmascaramiento individual de cada línea.
- Es posible conectar 9 controladores para manejar hasta 64 niveles de interrupción.



# Ej. de circuito controlador de interrupciones 8259

- Esquema general



Almacena y compara la identificación de todos los 8259 conectados en cascada

# Ej. de circuito controlador de interrupciones 8259

- Asignación de patillas

**CS#:** a 0 habilita la comunicación entre 8259 y CPU

**WR#:** a 0 habilita la escritura de órdenes de control desde la CPU

**RD#:** a 0 habilita la lectura por parte de la CPU del estado de los registros IRR, ISR o IMR, o el nivel de interrupción (en modo “polled”)

**D0-D7:** conexión al bus de datos

**CAS0 - CAS2:** Salidas cuando el 8259 es maestro (envían la identificación del 8259 al que le es aceptada una petición de interrupción). Entradas cuando es esclavo

CS	1	28	Vcc
WR	2	27	A0
RD	3	26	INTA
D7	4	25	IR7
D6	5	24	IR6
D5	6	23	IR5
D4	7	8259	IR4
D3	8	PIC	IR3
D2	9	20	IR2
D1	10	19	IR1
D0	11	18	IR0
CAS0	12	17	INT
CAS1	13	16	SP/EN
gnd	14	15	CAS2

**A0:** junto con RD# y WR# selecciona el registro a leer/escribir

**INTA# (Interrupt Acknowledge):** De la CPU al 8259. En forma de pulsos causa que el 8259 ponga información de vectorización en el bus de datos

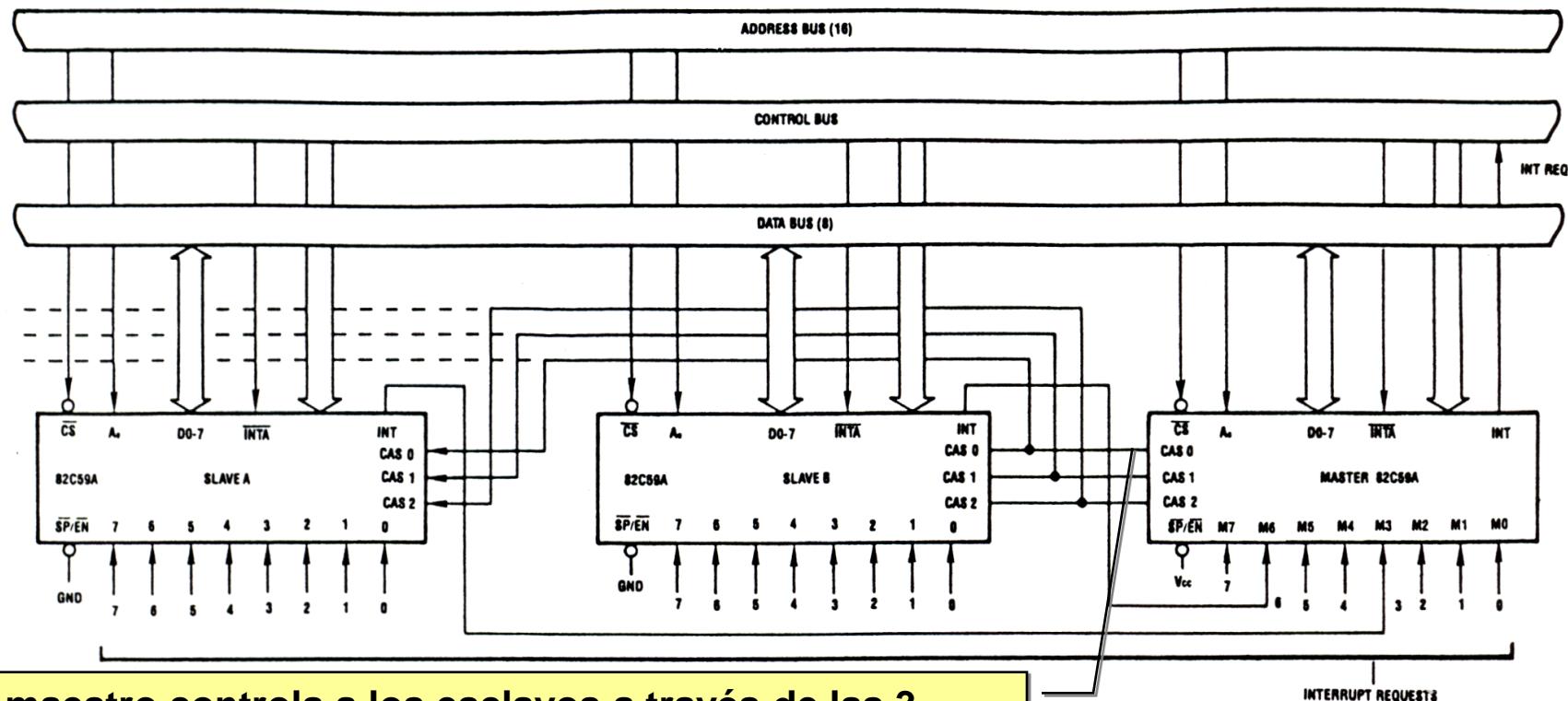
**IR0, IR7:** Peticiones de interrupción desde los periféricos

**INT:** Del 8259 a la CPU para realizar una petición de interrupción

**SP#/EN# (Slave Progress / Enable Buffer)**

# Ej. de circuito controlador de interrupciones 8259

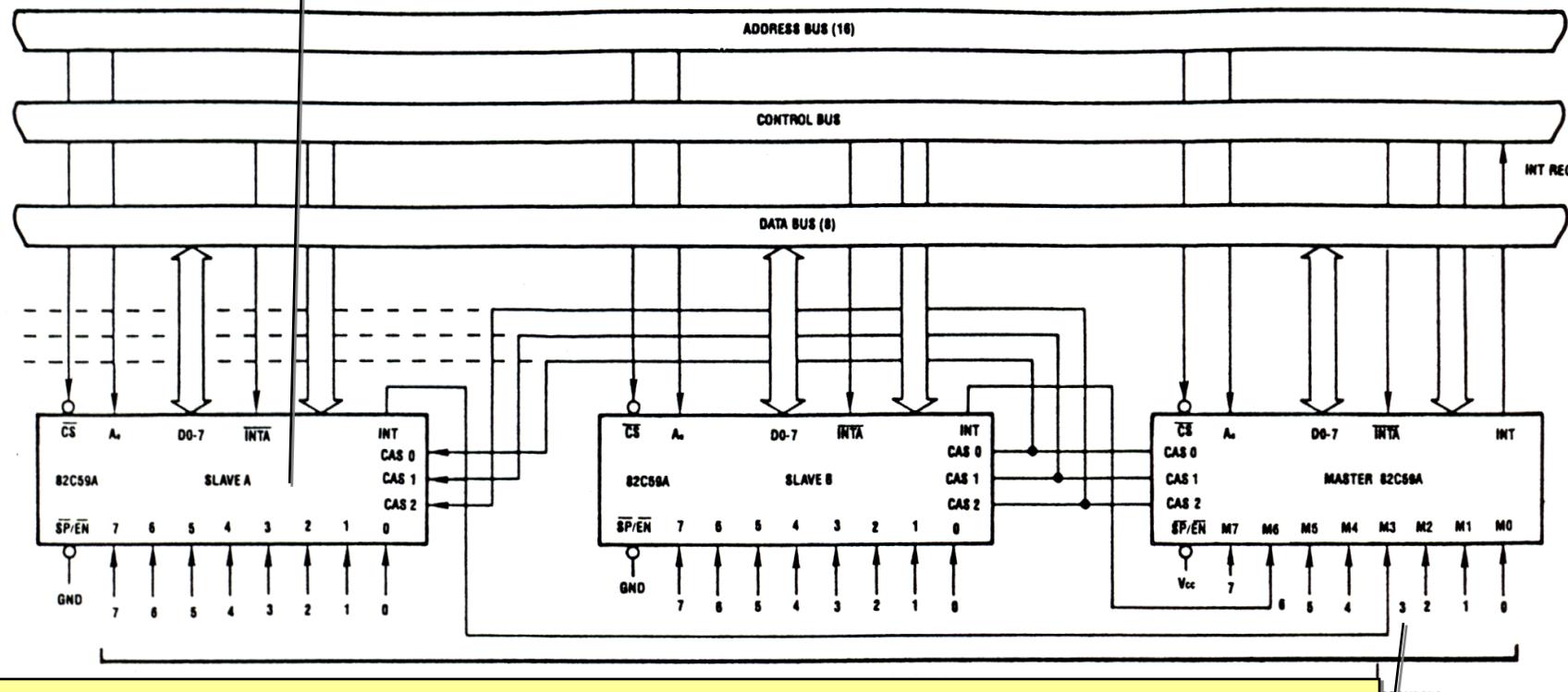
- Conexión en cascada de varios 8259
  - El 8259 puede ser conectado en un sistema de un maestro con hasta 8 esclavos, para manejar hasta 64 niveles de prioridad.



El maestro controla a los esclavos a través de las 3 líneas de cascada, que actúan como *Chip Selects* para los esclavos durante los pulsos INTA# (excepto el 1º)

# Ej. de circuito controlador de interrupciones 8259

Cada 8259 debe seguir una secuencia de inicialización y puede programarse para trabajar en un modo diferente



Las salidas INT de los esclavos se conectan a las entradas IR del maestro. Cuando una línea IR<sub>i</sub> se activa y después es reconocida, el maestro permite que el esclavo ponga en el bus de datos el vector de interrupción

# Ej. de circuito controlador de interrupciones 8259

## ■ Secuencia de petición de interrupción para la CPU 8085

1. Una o más líneas IR0-IR7 se ponen a 1, activándose los correspondientes bits de IRR.
2. El 8259 evalúa estas peticiones y si es apropiado envía una petición de interrupción al **8085** por la patilla INT.
3. El **8085** reconoce la interrupción respondiendo con pulso INTA#.
4. Al recibir el INTA#, el 8259 pone a 1 el bit de ISR de mayor prioridad de entre los correspondientes a bits de IRR a 1. El correspondiente bit de IRR lo pone a 0.
  - El 8259 **envía el codop CALL (11001101)** por el bus de datos.
  - Si la patilla IR atendida no sigue a 1 (petición demasiado breve) el 8259 envía una petición de nivel 7.
5. El **8085** envía **otros dos pulsos** INTA#.
6. El 8259 envía **la dirección de la ISR** a través del bus de datos (**en el primer pulso envía el LSB y en el segundo el MSB**).
  - Si el modo es AEOI, el bit de mayor prioridad a 1 del ISR se pone a 0 al final del tercer pulso INTA#.
  - Si no, el bit ISR permanece a 1 hasta que se envíe la orden EOI al 8259, normalmente al final de la rutina de servicio.

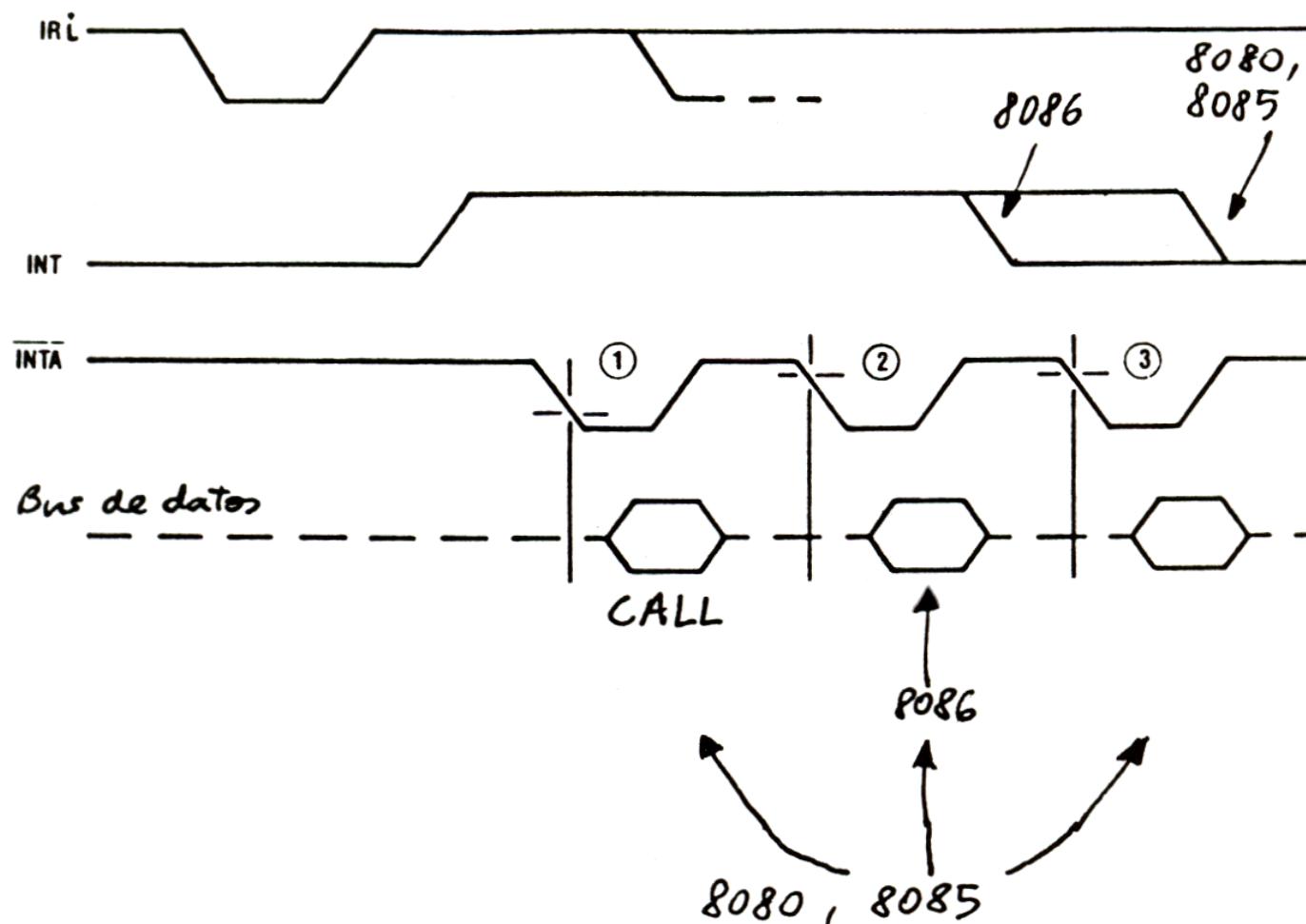
# Ej. de circuito controlador de interrupciones 8259

## ■ Secuencia de petición de interrupción para la CPU 8086

1. Una o más líneas IR0-IR7 se ponen a 1, activándose los correspondientes bits de IRR.
2. El 8259 evalúa estas peticiones y si es apropiado envía una petición de interrupción al **8086** por la patilla INT.
3. El **8086** reconoce la interrupción respondiendo con pulso INTA#.
4. Al recibir el INTA#, el 8259 pone a 1 el bit de ISR de mayor prioridad de entre los correspondientes a bits de IRR a 1. El correspondiente bit de IRR lo pone a 0.
  - El 8259 **no envía nada en este ciclo** por el bus de datos.
  - Si la patilla IR atendida no sigue a 1 (petición demasiado breve) el 8259 envía una petición de nivel 7.
5. El **8086** envía **otro pulso** INTA#.
6. El 8259 envía **un puntero de 8 bits** a través del bus de datos (**vector de interrupción**).
  - Si el modo es AEOI, el bit de mayor prioridad a 1 del ISR se pone a 0 al final del tercer pulso INTA#.
  - Si no, el bit ISR permanece a 1 hasta que se envíe la orden EOI al 8259, normalmente al final de la rutina de servicio.

## Ej. de circuito controlador de interrupciones 8259

- Vectores enviados a la CPU (I)



# Ej. de circuito controlador de interrupciones 8259

- Vectores enviados a la CPU (II)

1º INTA ; CONTENIDO DEL 1º BYTE DEL VECTOR :

	D7	D6	D5	D4	D3	D2	D1	D0
CALL CODE	1	1	0	0	1	1	0	1

2º INTA ; CONTENIDO DEL 2º BYTE DEL VECTOR :

IR                    Intervalo entre rutinas = 4 bytes

	D7	D6	D5	D4	D3	D2	D1	D0
7	A7	A6	A5	1	1	1	0	0
6	A7	A6	A5	1	1	0	0	0
5	A7	A6	A5	1	0	1	0	0
4	A7	A6	A5	1	0	0	0	0
3	A7	A6	A5	0	1	1	0	0
2	A7	A6	A5	0	1	0	0	0
1	A7	A6	A5	0	0	1	0	0
0	A7	A6	A5	0	0	0	0	0

IR                    Intervalo entre rutinas = 8 bytes

	D7	D6	D5	D4	D3	D2	D1	D0
7	A7	A6	1	1	1	0	0	0
6	A7	A6	1	1	0	0	0	0
5	A7	A6	1	0	1	0	0	0
4	A7	A6	1	0	0	0	0	0
3	A7	A6	0	1	1	0	0	0
2	A7	A6	0	1	0	0	0	0
1	A7	A6	0	0	1	0	0	0
0	A7	A6	0	0	0	0	0	0

3º INTA ; CONTENIDO DEL 3º BYTE DEL VECTOR :

	D7	D6	D5	D4	D3	D2	D1	D0
A15	A14	A13	A12	A11	A10	A9	A8	

A0-A15 ≡ Dirección de comienzo de la rutina de servicio de interrupción

8080, 8085

8086

2º INTA ; ÚNICO BYTE DEL VECTOR :

	D7	D6	D5	D4	D3	D2	D1	D0
IR7	T7	T6	T5	T4	T3	1	1	1
IR6	T7	T6	T5	T4	T3	1	1	0
IR5	T7	T6	T5	T4	T3	1	0	1
IR4	T7	T6	T5	T4	T3	1	0	0
IR3	T7	T6	T5	T4	T3	0	1	1
IR2	T7	T6	T5	T4	T3	0	1	0
IR1	T7	T6	T5	T4	T3	0	0	1
IR0	T7	T6	T5	T4	T3	0	0	0

T0-T7 ≡ Posición o índice de la tabla de vectores de interrupción

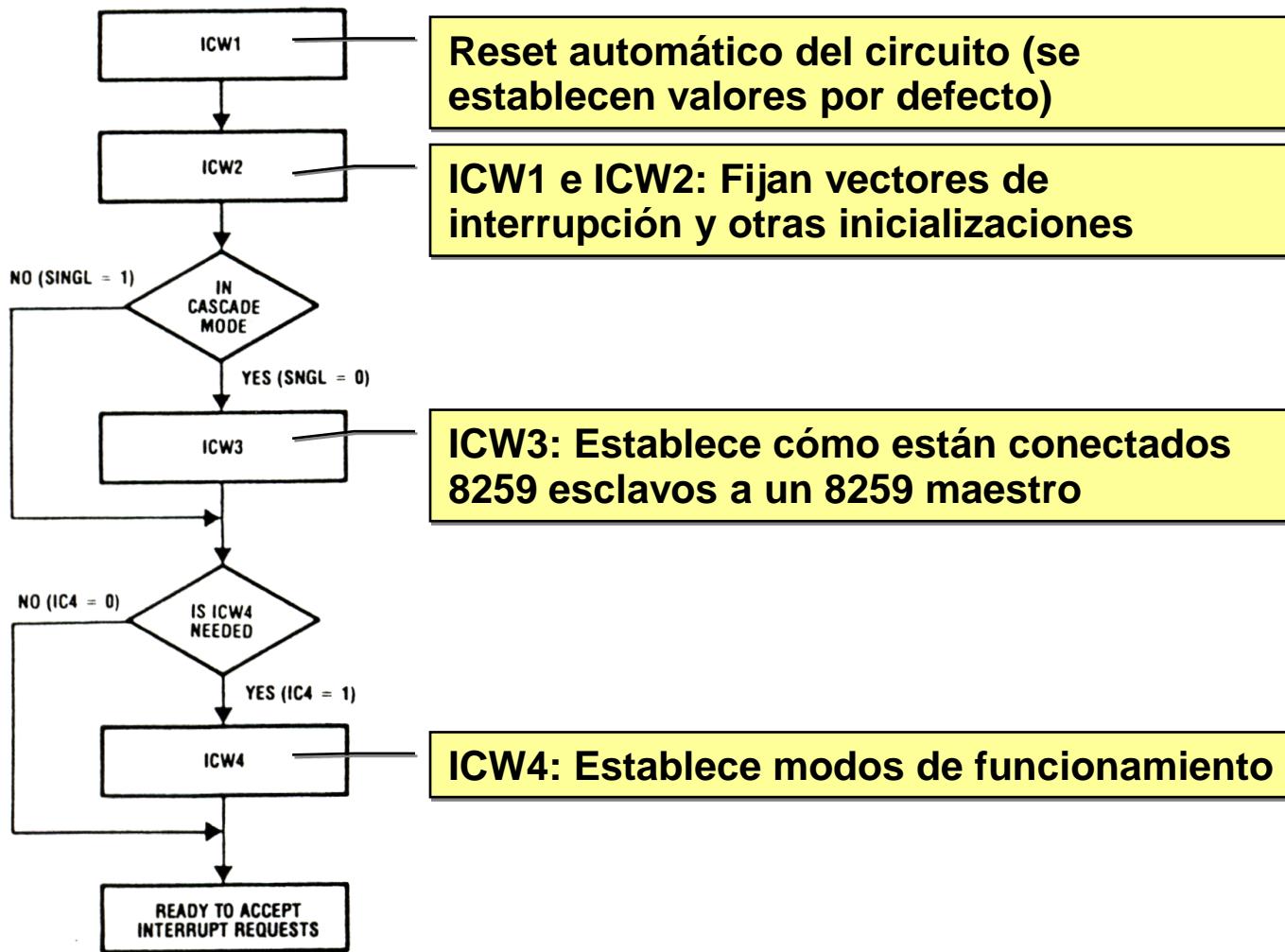
# Ej. de circuito controlador de interrupciones 8259

## ■ Programación del 8259

- El 8259 acepta dos tipos de órdenes desde la CPU:
  - De  **inicialización (*Initialization Command Words, ICWs*)**:
    - Deben ser enviadas a cada 8259 antes de que comience el funcionamiento normal.
  - De **funcionamiento (*Operation Command Words, OCWs*)**:
    - Hacen que el 8259 funcione en varios modos:
      - » Completamente anidado
      - » Con rotación de prioridad
      - » Modo de máscara especial
      - » “Polled” (por sondeo)
    - ...
    - Las OCWs pueden escribirse en cualquier momento tras la inicialización.

# Ej. de circuito controlador de interrupciones 8259

## ■ Secuencia de inicialización del 8259



## Ej. de circuito controlador de interrupciones 8259

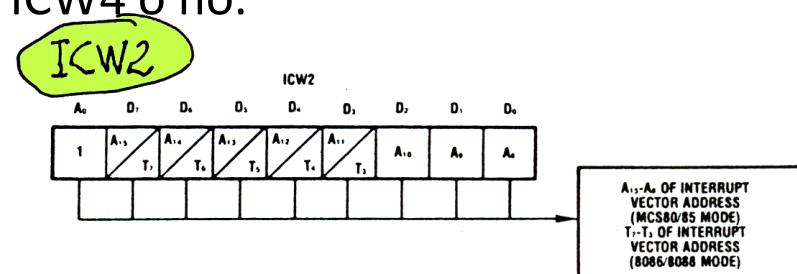
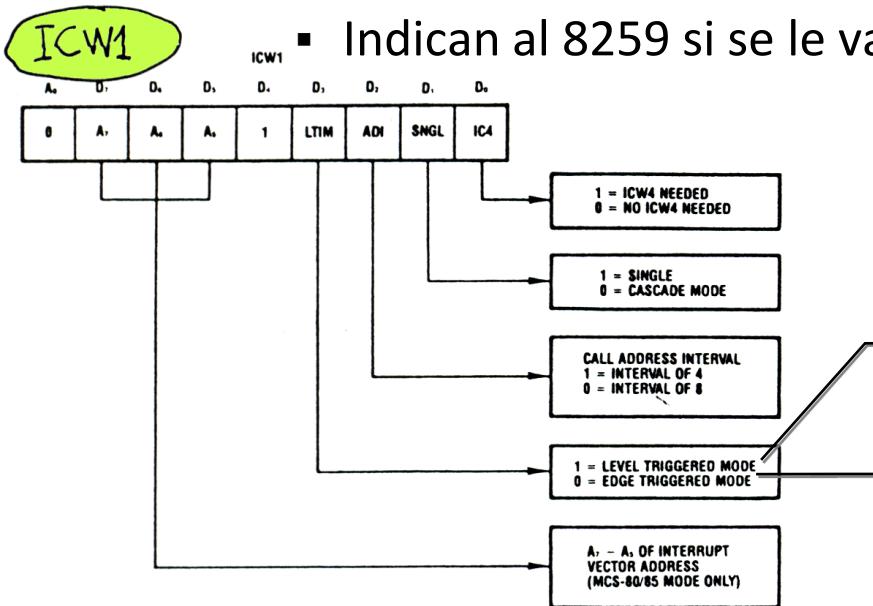
- Despues de enviar las órdenes ICW, el 8259 queda programado por defecto en el modo completamente anidado.
- **Modo completamente anidado** (*Fully Nested Mode*)
  - Las peticiones de interrupción están **ordenadas desde 0** (IR0, la más prioritaria), **hasta 7** (IR7, la menos prioritaria).
  - Las prioridades **no son alterables automáticamente** (de ser necesario, podrían ser cambiadas por software).
  - Mientras que un bit ISR esté a 1, **sólo las líneas con mayor prioridad pueden interrumpir**.
    - Sólo serán reconocidas esas interrupciones si el biestable de habilitación de interrupciones del microprocesador ha sido rehabilitado por software (se haya ejecutado un EI / STI en la ISR).

# Ej. de circuito controlador de interrupciones 8259

## ■ Órdenes de inicialización (ICW)

### ■ ICW1 e ICW2

- Establecen las direcciones de comienzo de las ISR (8080/8085) o los índices de la tabla de vectores de interrupción (8086).
- Informan al 8259 de si va a trabajar solo o en cascada.
- Establecen separación entre subrutinas (8080/8085) de 4/8 B.
- Establecen si las IRi se detectarán por flanco o por nivel.
- Indican al 8259 si se le va a enviar ICW4 o no.



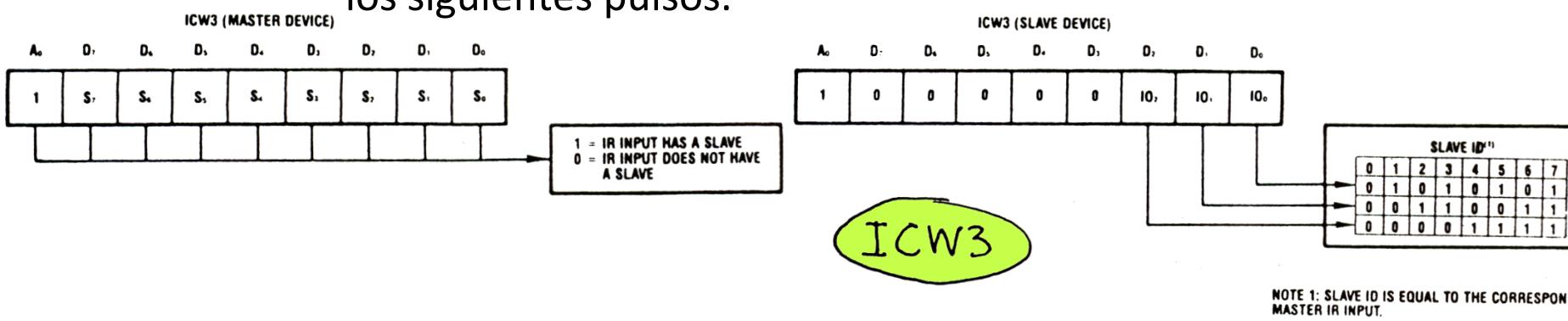
IRi debe quitarse antes de EOI para evitar que ocurra una segunda interrupción

IRi puede seguir en alto sin generar nuevas interrupciones

# Ej. de circuito controlador de interrupciones 8259

## ■ ICW3

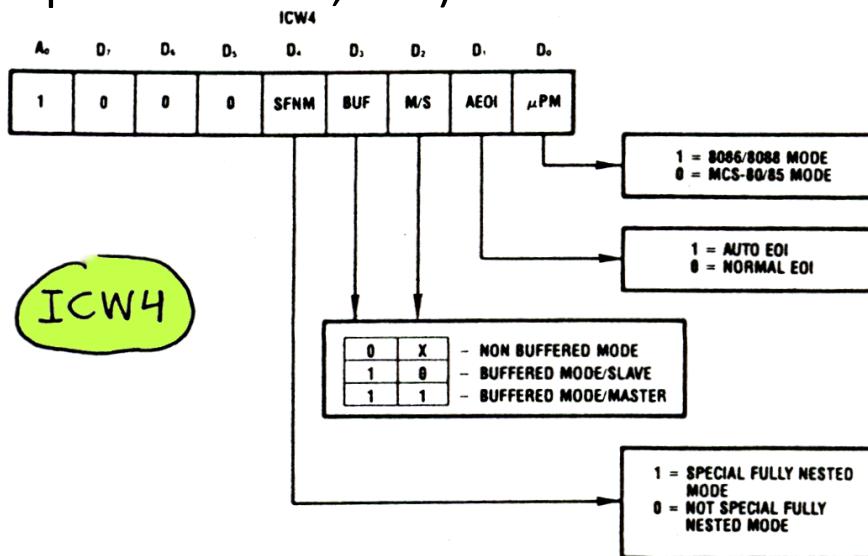
- Usada sólo cuando hay más de un 8259 (cascada)
- El formato de ICW3 es distinto para maestro y esclavo:
- **MAESTRO** (SP#/EN# = 1, o BUF=1&M/S=1 en ICW4)
  - Un 1 en cada bit i tal que IRi esté conectado a un esclavo.
- **ESCLAVO** (SP#/EN# = 0, o BUF=1&M/S=0 en ICW4)
  - Los bits 2-0 identificarán en adelante a ese esclavo.
  - Cada vez que se produzca el primer INTA#, el esclavo los comparará con C2-C0 y si son iguales enviará el vector en los siguientes pulsos.



# Ej. de circuito controlador de interrupciones 8259

## ■ ICW4

- Establece si se van a usar los modos:
  - Fin de interrupción automático
  - Con buffers
  - Completamente anidado especial
- Establece si se va a usar con un 8080/8085 o con un 8086.
- Si no se envía ICW4, todos los bits estarán a 0 (8080/8085, fin de interrupción normal, etc.)



# Ej. de circuito controlador de interrupciones 8259

- **Tipos de fin de interrupción:**
  - El bit  $ISR_i = 1$  correspondiente a la instrucción que se está atendiendo puede ponerse a 0 de dos formas.
    - » AEOI: automáticamente tras el flanco de subida del último pulso INTA#.
    - » EOI normal: mediante una orden de fin de interrupción (EOI en OCW2).
- **Modo con buffers:**
  - Hay sistemas con buffers en los buses, que deben ser habilitados para mandar información a través de ellos.
  - En el modo con buffers, siempre que el 8259 saca datos al bus de datos, activa la señal SP#/EN#.
  - Como la señal SP#/EN# ya no se puede usar para determinar si un 8259 es maestro o esclavo, esto se hará por software mediante el bit 2 de ICW4.

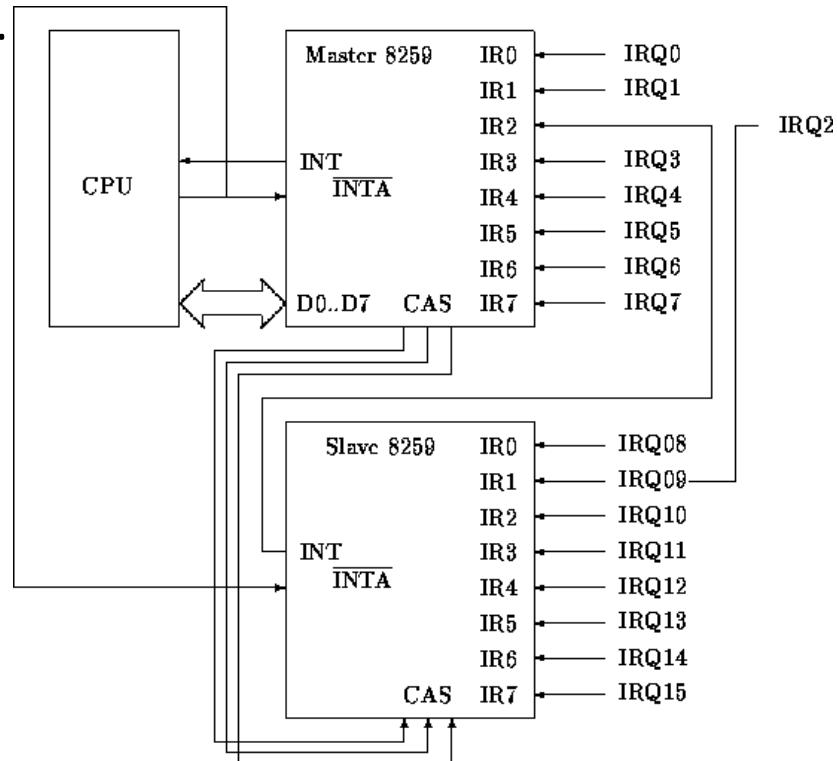
## Ej. de circuito controlador de interrupciones 8259

- **Modo completamente anidado especial:**
  - Puede usarse en sistemas que trabajen en cascada, si se quiere conservar las prioridades dentro de cada esclavo.
  - Es similar al modo completamente anidado normal, excepto:
    - » En el modo completamente anidado normal, un esclavo queda enmascarado cuando su petición de interrupción está en servicio (no se pueden servir peticiones del mismo esclavo ni siquiera si son de más alta prioridad).
    - » En el modo completamente anidado especial, el esclavo no queda bloqueado (las peticiones de mayor prioridad dentro del esclavo serán reconocidas por el maestro y generarán peticiones en el procesador).

...

## Ej. de circuito controlador de interrupciones 8259

- La ISR tiene que comprobar si la interrupción servida por ella fue la única desde ese esclavo. Para ello:
  1. Envía una orden EOI al esclavo (OCW2).
  2. Lee su registro ISR viendo si está a 0. Si es 0 tiene que mandar otro EOI al maestro. Si no es 0 no manda EOI al maestro.

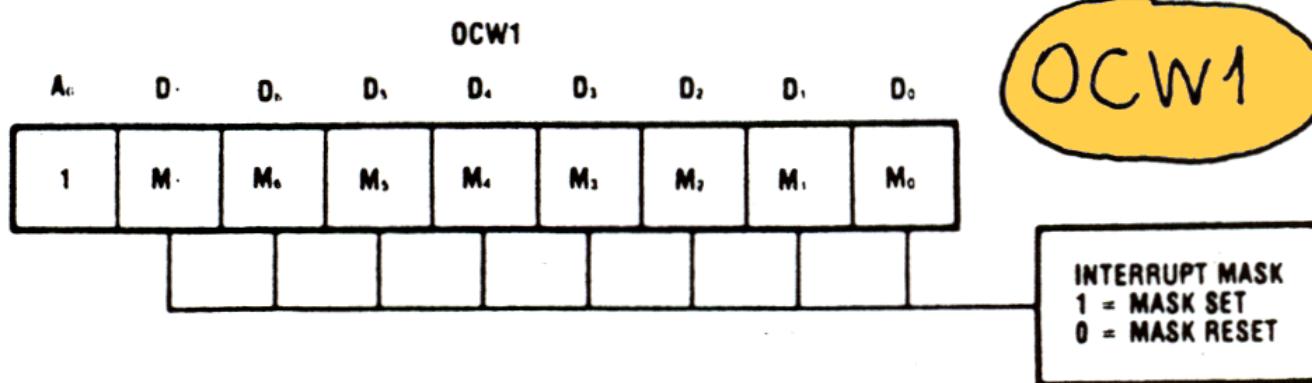


# Ej. de circuito controlador de interrupciones 8259

## ■ Órdenes de funcionamiento (OCW)

### ■ OCW1

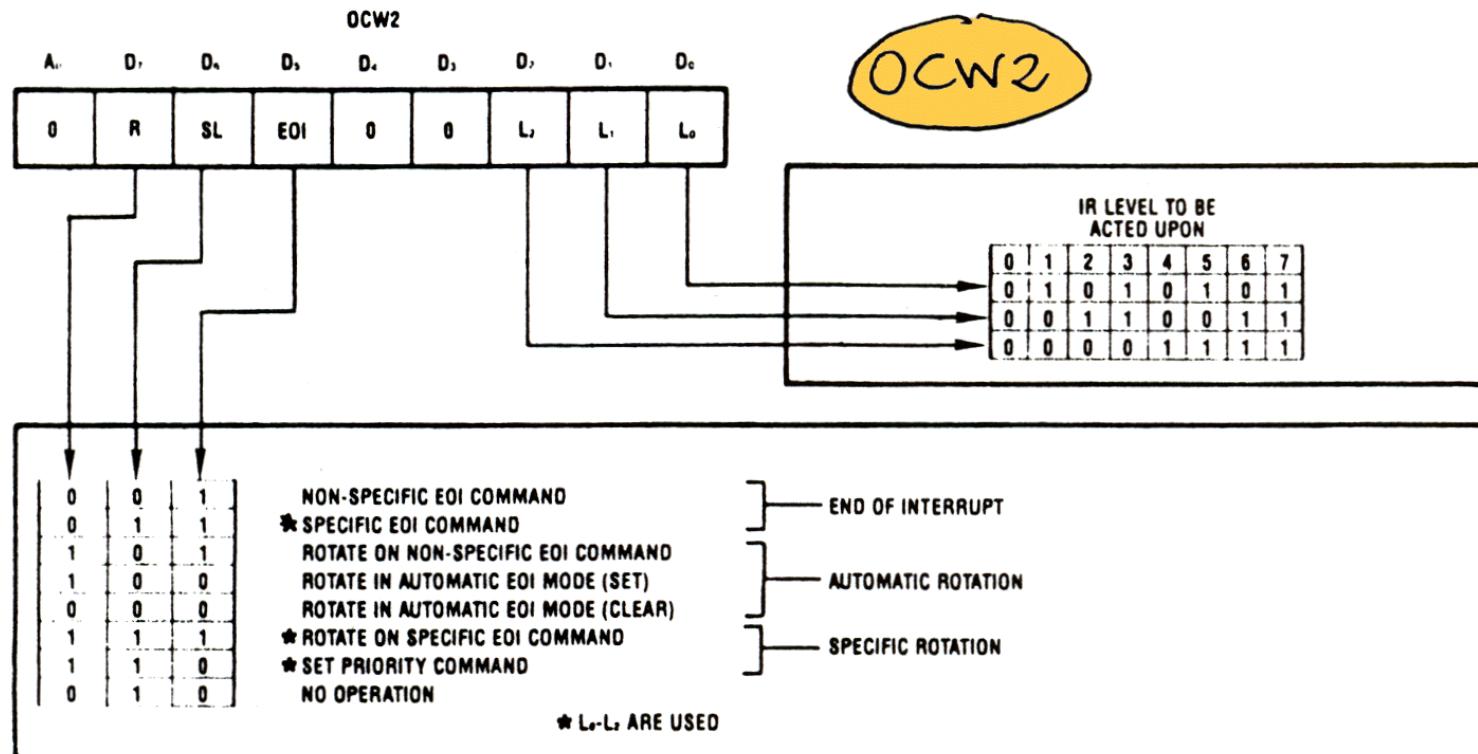
- Pone a 1 ó a 0 cada bit del registro IMR.
  - $M_i = 1 \Rightarrow \text{IR}_i$  enmascarada (inhabilitada)
  - $M_i = 0 \Rightarrow \text{IR}_i$  habilitada



# Ej. de circuito controlador de interrupciones 8259

## ■ OCW2

- Los bits R, SL y EOI controlan varios modos de rotación de prioridad y de fin de interrupción.
  - Los bits  $L_2, L_1, L_0$  determinan el nivel de interrupción sobre el que se actúa cuando  $SL = 1$ .



# Ej. de circuito controlador de interrupciones 8259

- ***End Of Interrupt*, fin de interrupción:**

- En el modo normal de fin de interrupción (AEOI = 0 en ICW4), hay que enviar una orden EOI al 8259 antes de regresar de la ISR (o dos EOI, uno para el esclavo y otro para el maestro si se usa modo cascada).
- Hay dos tipos de órdenes EOI:
  - **EOI no específico** (el caso normal). No hace falta decir qué interrupción ha terminado, pues será la más prioritaria de las que están siendo atendidas. El 8259 borra de su registro ISR el bit con mayor prioridad que esté puesto a 1.
  - **EOI específico** (sólo se usa si se ha alterado la prioridad de interrupciones totalmente anidada). El 8259 no será capaz de determinar el último nivel de interrupción reconocido, por tanto hace falta mandar en  $L_2$ ,  $L_1$ ,  $L_0$  el nivel de interrupción que se desea borrar del registro ISR.

# Ej. de circuito controlador de interrupciones 8259

- ***Automatic Rotation***, rotación automática (si se desea que los dispositivos tengan igual prioridad):
  - Tras ser servido, el dispositivo recibe la prioridad más baja.
  - Cualquier dispositivo que está pidiendo interrupción espera como mucho el tiempo que tardan otros 7 dispositivos en ser atendidos una vez.
  - Hay dos formas de llevar a cabo la rotación automática:
    - Rotación al enviar una orden de EOI no específico.
    - Rotación en el modo AEOI (se puede activar / desactivar).
- Ejemplo:

**Antes de rotar, IR4 tiene la prioridad más alta de las líneas en servicio**

"IS" STATUS	IS7	IS6	IS5	IS4	IS3	IS2	IS1	ISO
PRIORITY STATUS	7	6	5	4	3	2	1	0
lowest								highest
"IS" STATUS	IS7	IS6	IS5	IS4	IS3	IS2	IS1	ISO
PRIORITY STATUS	0	1	0	0	0	0	0	0

**Después de rotar (IR4 ha sido servida)**

"IS" STATUS	IS7	IS6	IS5	IS4	IS3	IS2	IS1	ISO
PRIORITY STATUS	2	1	0	7	6	5	4	3
highest								lowest
"IS" STATUS	IS7	IS6	IS5	IS4	IS3	IS2	IS1	ISO
PRIORITY STATUS	0	1	0	0	0	0	0	0

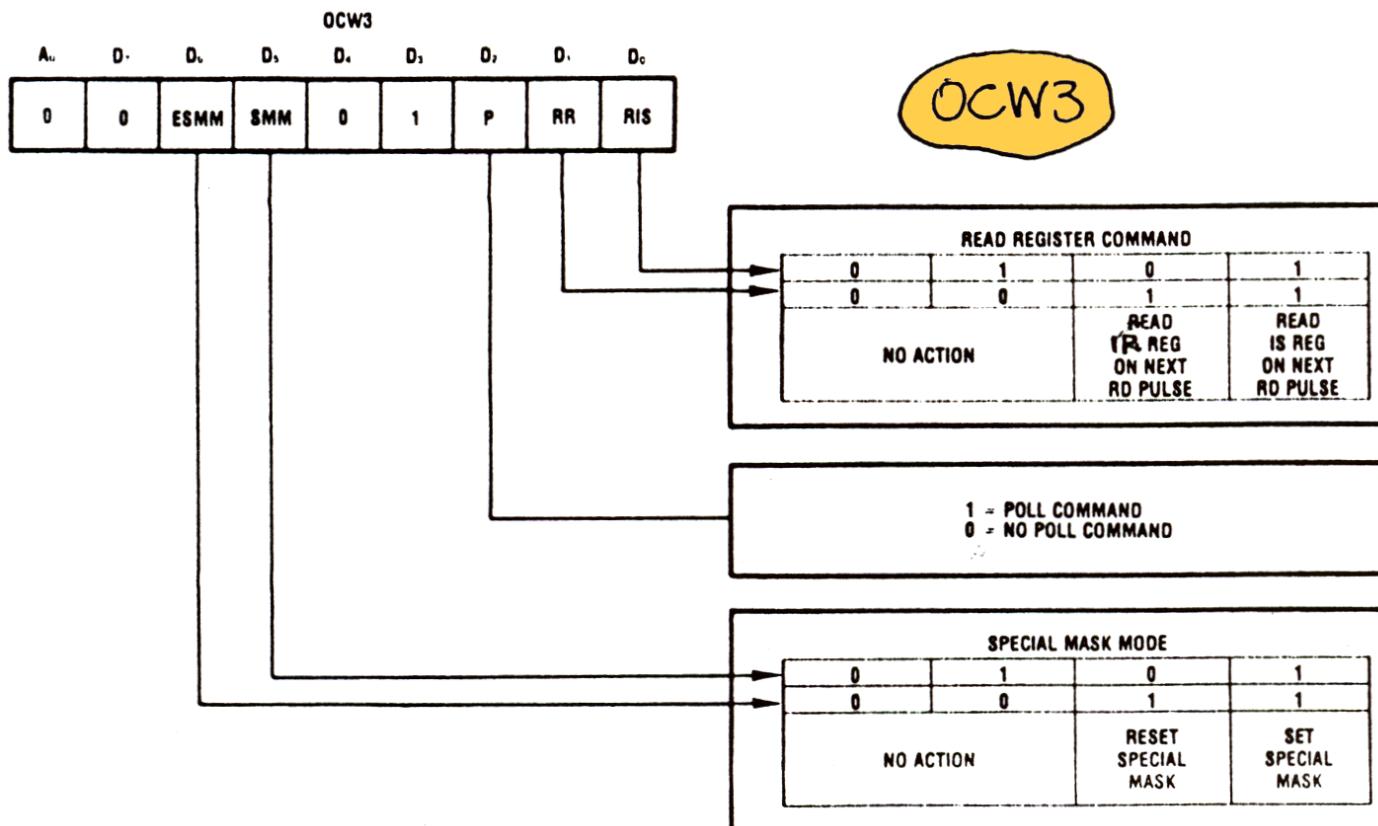
## Ej. de circuito controlador de interrupciones 8259

- ***Specific Rotation*, rotación específica:**
  - El programador puede cambiar específicamente las prioridades decidiendo la línea con la prioridad más baja (el resto se fijará automáticamente).
    - Ej.: Si IR5 se programa como la línea de menor prioridad, IR6 tendrá la mayor prioridad, luego IR7, luego IR0,...
  - Hay dos formas de llevar a cabo la rotación específica:
    - Rotación al enviar una orden de EOI específico.
    - Orden “set priority” (fijar prioridad).

# Ej. de circuito controlador de interrupciones 8259

## ■ OCW3

- El bit ESMM permite activar/desactivar el modo máscara especial.
  - El bit P permite escribir una orden “poll” (sondeo).
  - El bit RR = 1 permite que los registros IRR e ISR sean leídos.



## Ej. de circuito controlador de interrupciones 8259

- ***Special Mask Mode***, modo de máscara especial:
  - Establece que todas las interrupciones puedan ser tratadas en el momento en que se produzcan **sin tener en cuenta las prioridades**, pero teniendo en cuenta los bits de máscara.
  - Interrupciones de niveles más bajos que la que está siendo atendida pueden interrumpir a la CPU si no están enmascaradas.

## Ej. de circuito controlador de interrupciones 8259

- **Poll command, orden de sondeo:**

- El 8259 pasa a no generar interrupciones a la CPU.
- El 8259 tratará el siguiente pulso RD# como una aceptación de interrupción, fijará el bit ISR apropiado y escribirá en el bus de datos lo siguiente:

D7	D6	D5	D4	D3	D2	D1	D0
I	—	—	—	—	W2	W1	W0

↓  
Igual a "1"  
si hay una  
interrupción

Código binario del  
nível de prioridad  
más alto que pide  
servicio

## Ej. de circuito controlador de interrupciones 8259

- *Read Register command*, lectura de IRR e ISR:

- Se envía una orden OCW3 con RR = 1.
- El siguiente pulso RD# transfiere a la CPU por el bus de datos el contenido de IRR (si RIS = 0) o el de ISR (si RIS = 1) (siempre con A0 = 0).

**(Para leer el IMR no se necesita una orden OCW3, basta con leer con A0 = 1)**

# Apéndice: circuitos integrados E/S

- Interfaz de periféricos programable 8255
- Controlador de interrupciones programable 8259
  - Interrupciones en el PC en modo real
- Controlador de DMA programable 8237

# Interrupciones en el PC (modo real)

## ■ Interrupciones vectorizadas:

- Existen 256 interrupciones posibles (0 a 255 = 0xFF), vectorizadas.
- Tabla de vectores de interrupción:
  - primeros 1024 bytes de memoria (0 a 0x3FF).
- Cada vector de interrupción:
  - doble palabra (32 bits, 4 bytes)
  - dirección de la ISR asociada a esa interrupción

Segmento (CS)
Desplazamiento (IP)

# Interrupciones en el PC (modo real)

## ■ Interrupciones vectorizadas:

- Sólo unas pocas se generan por hardware (interna o externamente).
  - si IF (*Interrupt Flag*) = 1, detienen temporalmente la ejecución del programa
  - si IF = 0, el procesador ignora int. que llegan por la patilla INTR
  - IF se borra con `CLI/disable()` y activa con `STI/enable()`
  - Hay una interrupción no enmascarable especial que se reconoce siempre, independientemente de IF, cuando se activa la patilla NMI (*Non Maskable Interrupt*)
- Todas se pueden generar por software
  - A través de las instrucciones INT e INTO
  - En este caso no interrumpen realmente nada
  - Son rutinas del SO que ejecutan tareas de E/S, gestión de ficheros, gestión de memoria, etc.

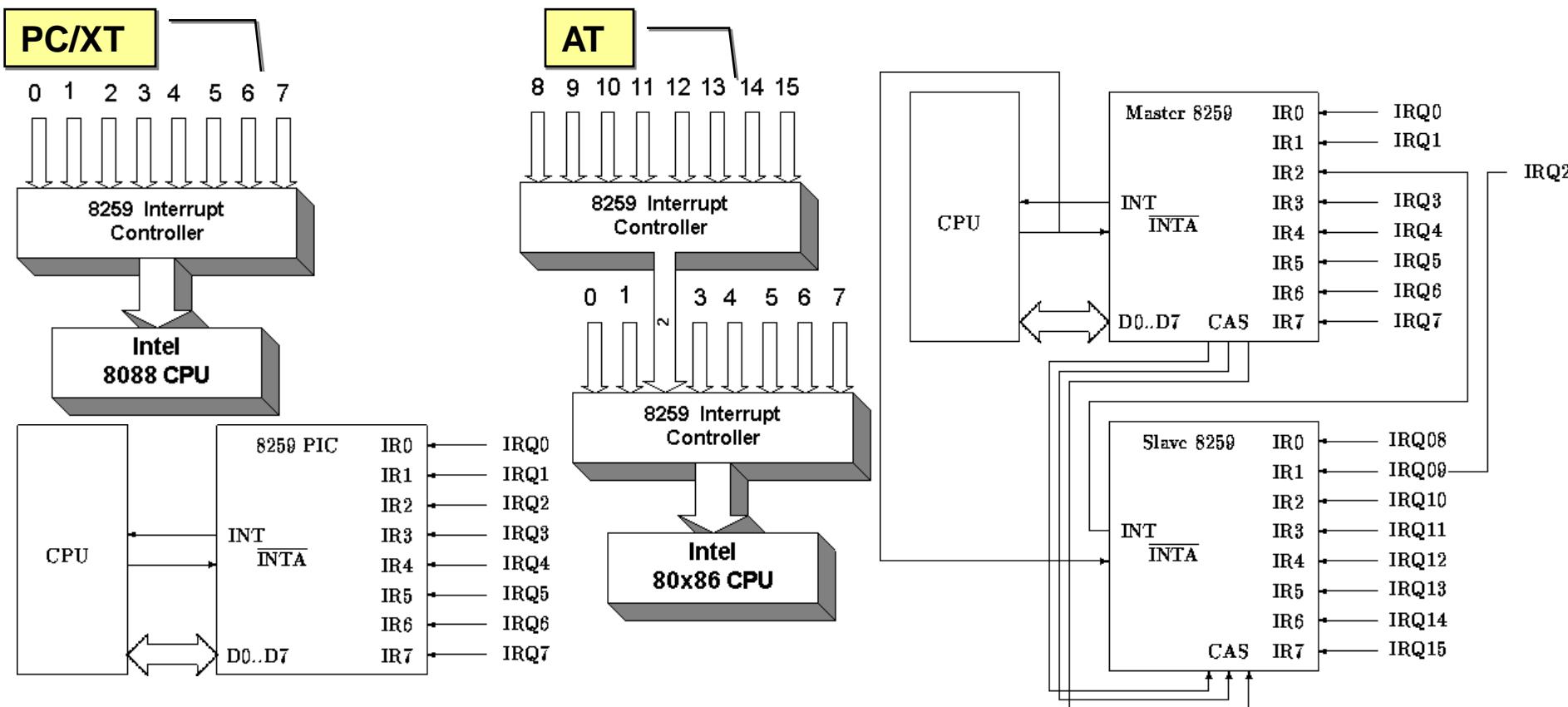
# Interrupciones en el PC (modo real)

## ■ Interrupciones vectorizadas:

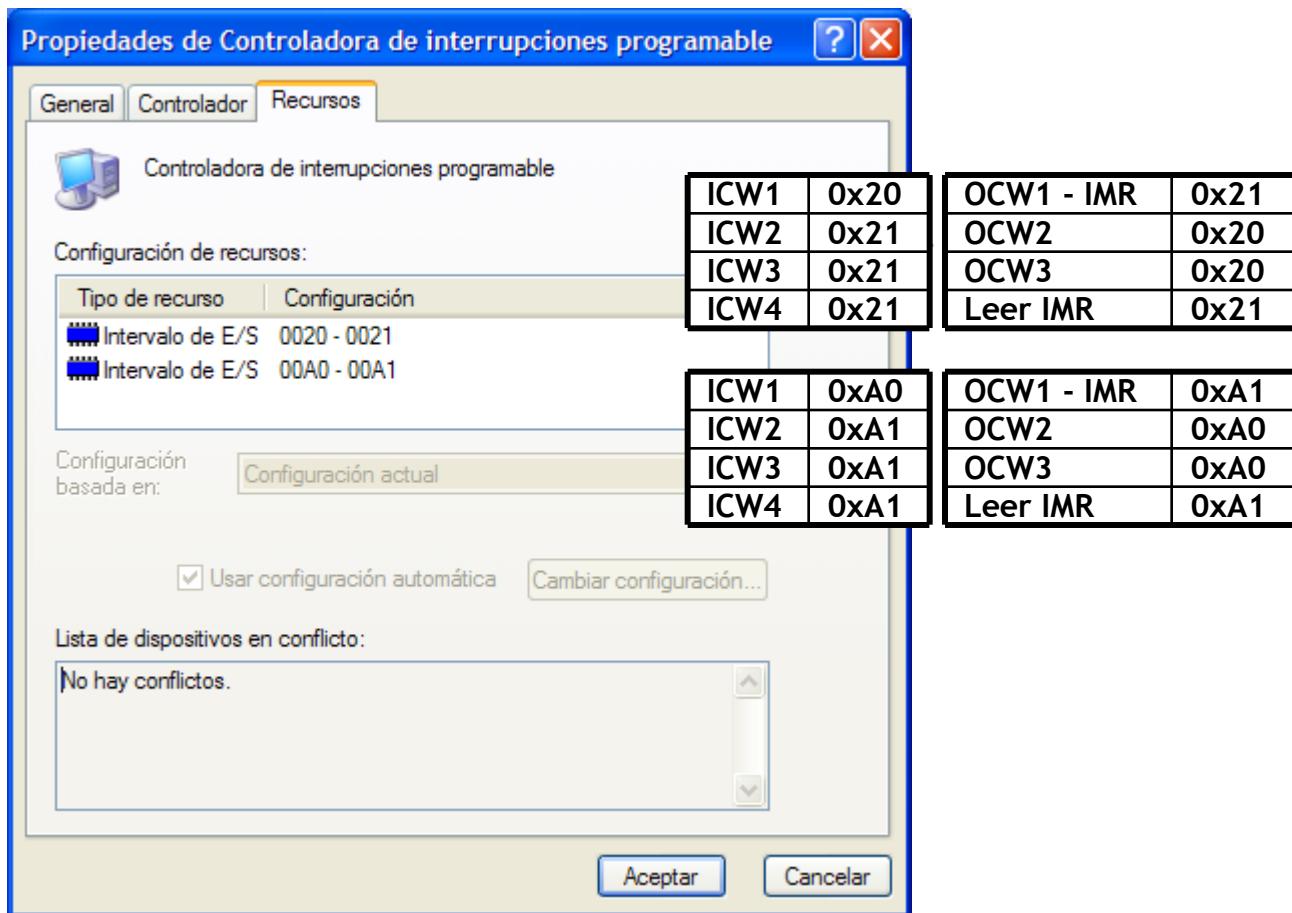
- Cuando se produce una interrupción, el procesador:
  - almacena el estado actual de la máquina introduciendo en la pila el registro de *flags*, el registro CS y el registro IP
  - pone IF a 0 para evitar nuevas interrupciones
  - multiplica por 4 el número de interrupción (que le llega por el bus de datos) para obtener la dirección de un vector de interrupción
  - asigna a IP y CS el vector de interrupción
    - se salta a la rutina que haya en esa dirección, que finalizará al ejecutar IRET

# Interrupciones en el PC (modo real)

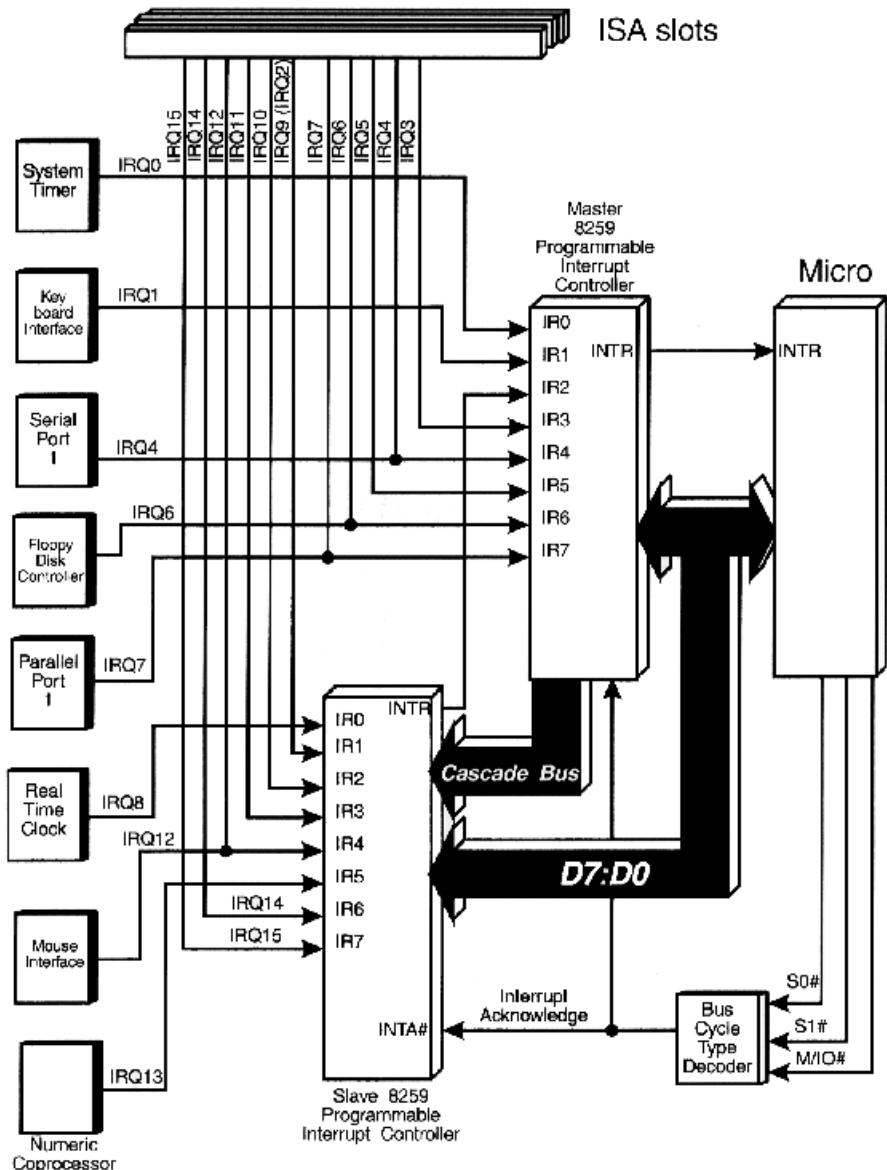
- Para gestionar las interrupciones hardware que provienen de la patilla INTR se dispone de dos 8259 (sólo uno en el PC/XT), con varios niveles de prioridad



# Interrupciones en el PC (modo real)



# Interrupciones en el PC (modo real)



8259 maestro	8259 esclavo	PC	AT y post.	Vector
IRQ0		temporizador	0x08	
IRQ1		teclado	0x09	
IRQ2		Bus PC	-	0x0A
	IRQ8	-	reloj tiempo real	0x70
	IRQ9	-	Bus ISA/PCI (tarjeta sonido)	0x71
	IRQ10	-	Bus ISA/PCI	0x72
	IRQ11	-	Bus ISA/PCI (ej. tarjeta video, NIC)	0x73
	IRQ12	-	Bus ISA/PCI Ratón PS/2	0x74
	IRQ13	-	Procesador matemático	0x75
	IRQ14	-	IDE1	0x76
	IRQ15	-	IDE2	0x77
IRQ3		COM1	COM2	0x0B
IRQ4		COM2	COM1	0x0C
IRQ5		Disco duro	Normalmente LPT2 o tarjeta de sonido ISA	0x0D
IRQ6		Disquetera		0x0E
IRQ7		LPT1		0x0F

# Interrupciones en el PC (modo real)

## ■ Ejemplo de ISR:

```
;rutina de interrupción
ISR proc far
    sti          ; Si queremos que entren las...
                  ; ...interrupciones prioritarias
    push ax      ; guardar AX
    push bx      ; guardar otros regs. cambiados
    .
    .
    .
    pop bx      ; recuperar otros regs. cambiados
    mov al,20h   ; EOI
    out 20h,al   ; OCW2 —————
    pop ax      ; recuperar AX
    iret
ISR endp
```

Se envía al 8259 maestro la orden de fin de interrupción antes del IRET  
(sólo para interrupciones provenientes del 8259)

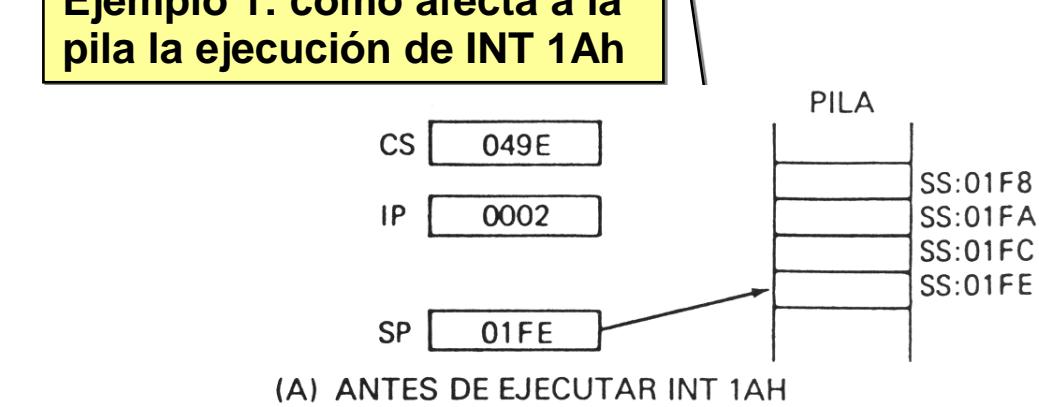
# Interrupciones en el PC (modo real)

## ■ Pasos en la ejecución de una interrupción:

```

SP ← SP - 2
pila ← FLAGS
IF, TF ← 0
SP ← SP - 2
pila ← CS
CS ← M[nº int. * 4 + 2]
SP ← SP - 2
pila ← IP
IP ← M[nº int. * 4]
(Ejecución de ISR)
IP ← pila
SP ← SP + 2
CS ← pila
SP ← SP + 2
FLAGS ← pila
SP ← SP + 2
    
```

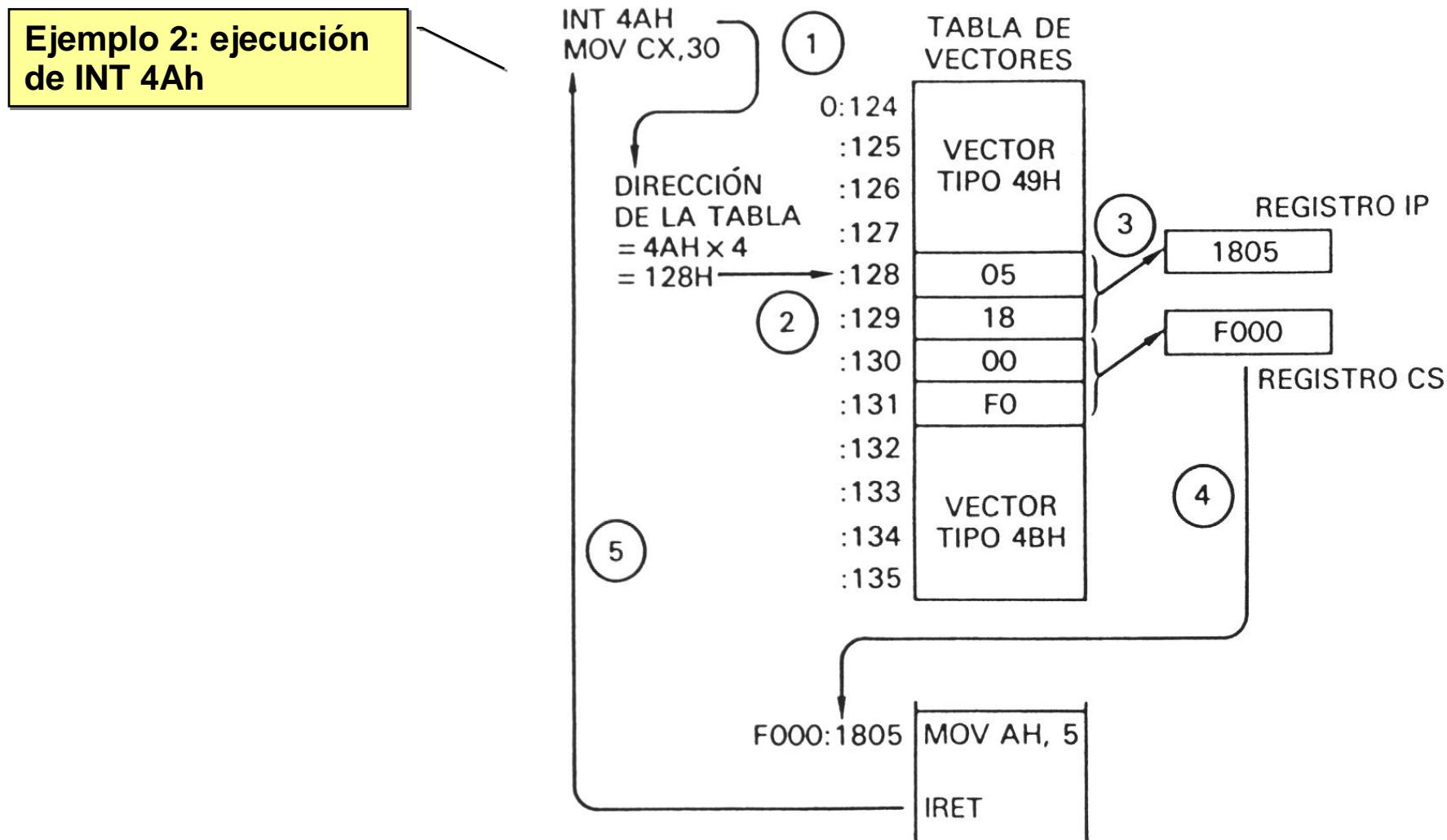
### Ejemplo 1: cómo afecta a la pila la ejecución de INT 1Ah



(B) DESPUÉS DE EJECUTAR INT 1AH

# Interrupciones en el PC (modo real)

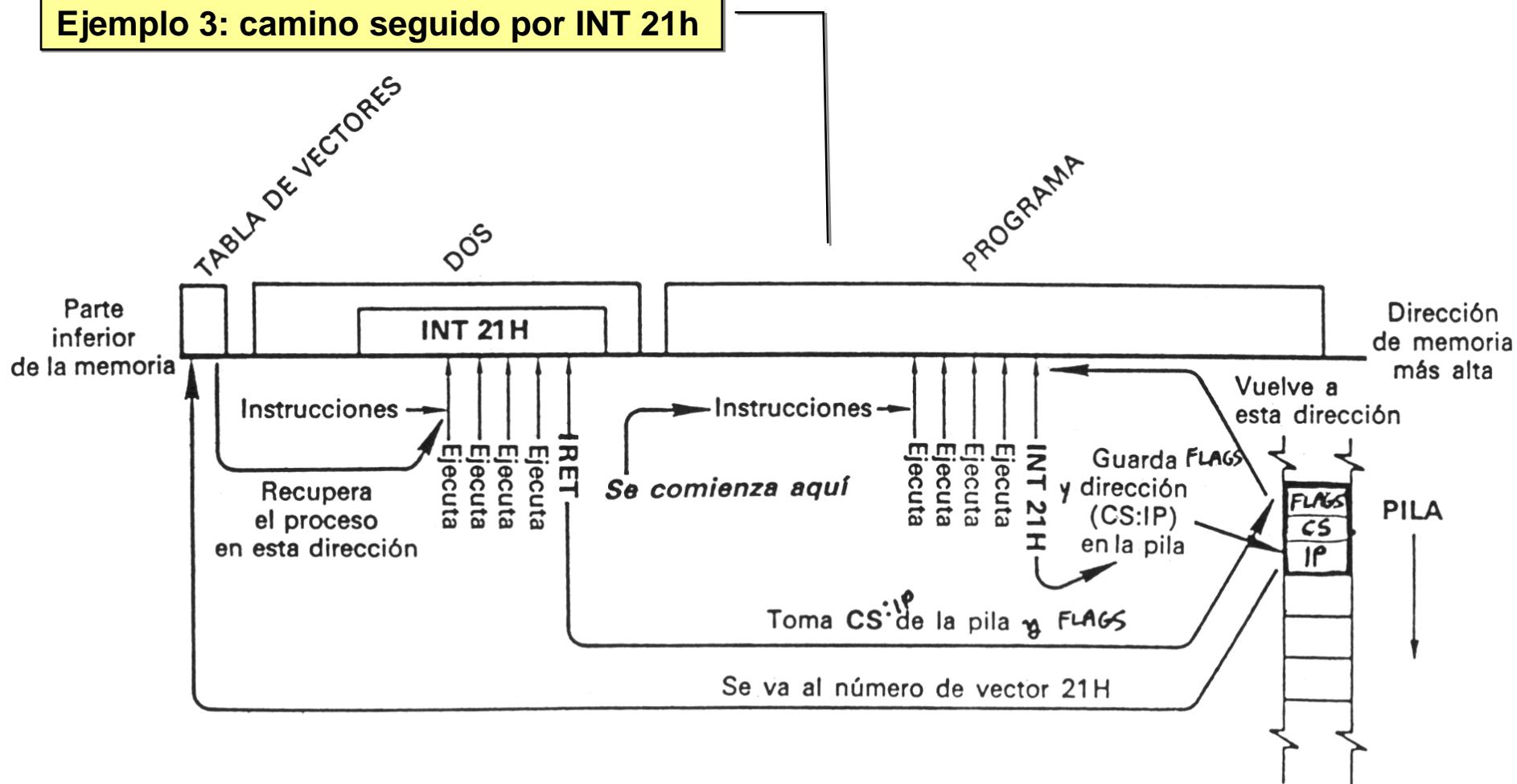
## ■ Pasos en la ejecución de una interrupción:



# Interrupciones en el PC (modo real)

## ■ Pasos en la ejecución de una interrupción:

### Ejemplo 3: camino seguido por INT 21h

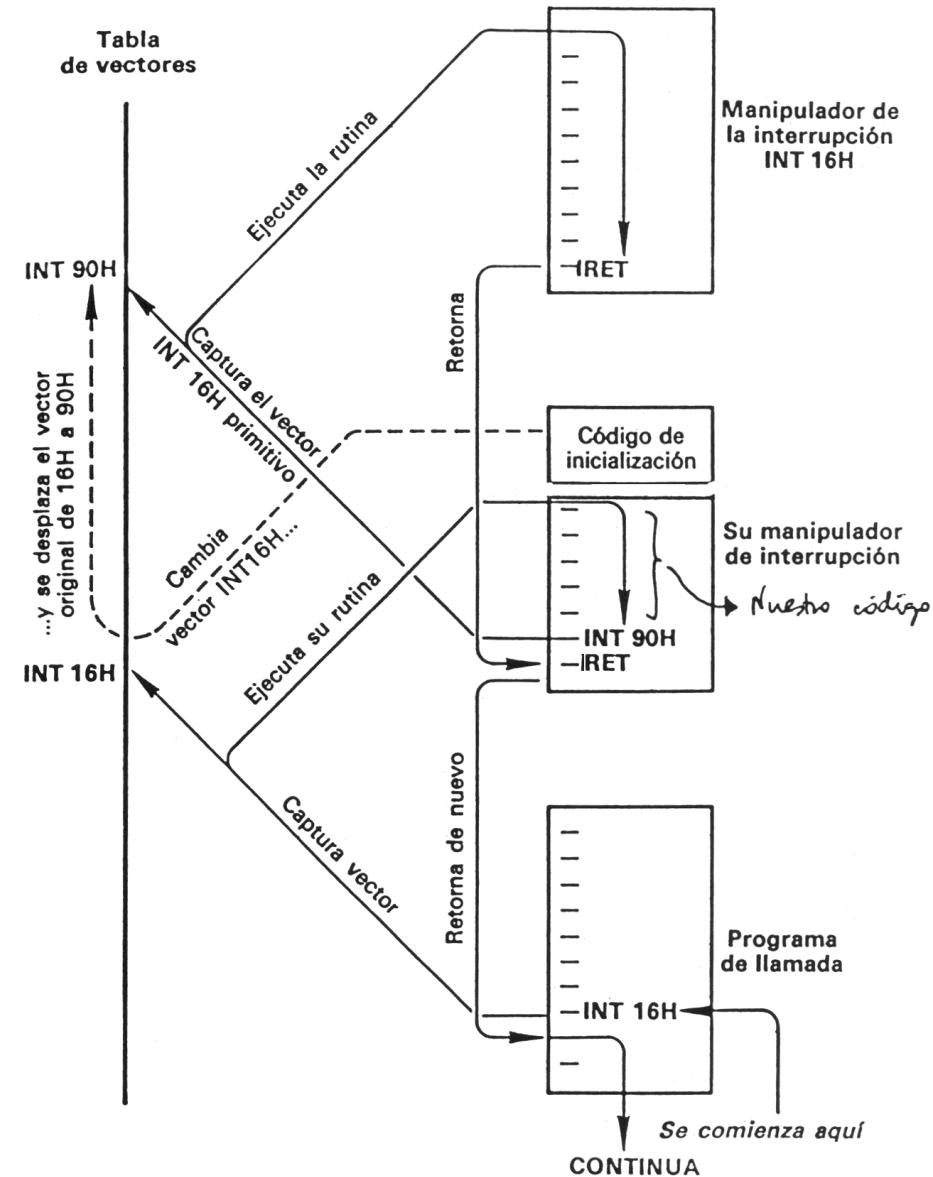


# Interrupciones en el PC (modo real)

## ■ Modificación de una interrupción:

- Función 35h de INT 21h (`getvect()`) devuelve el valor actual de un vector
- Función 25h de INT 21h (`setvect()`) cambia el valor de un vector

**Ejemplo: modificación de la int. 16h para que se ejecute después de nuestro código**



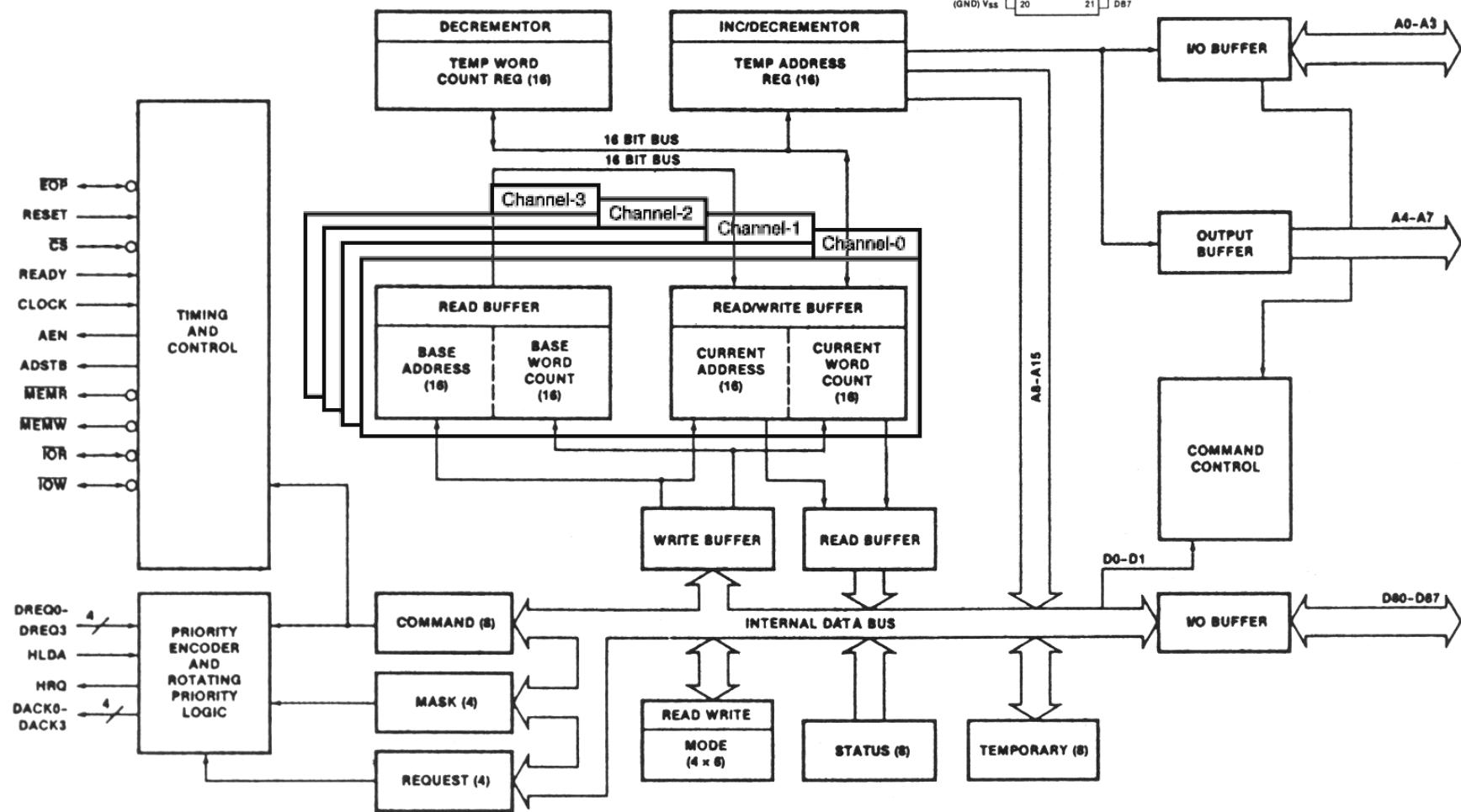
# Apéndice: circuitos integrados E/S

- **Interfaz de periféricos programable 8255**
- **Controlador de interrupciones programable 8259**
  - Interrupciones en el PC en modo real
- **Controlador de DMA programable 8237**

# Ejemplo de DMAC: 8237

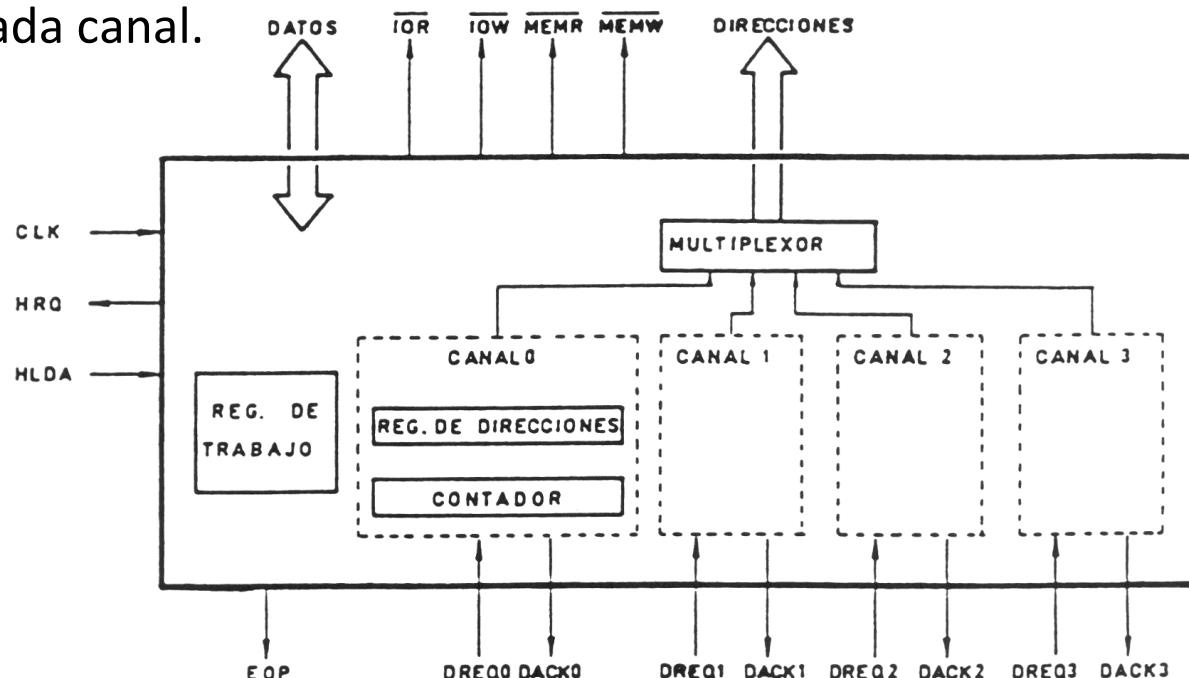
## ■ Programmable DMA Controller

- 4 canales de DMA independientes:



# Ejemplo de DMAC: 8237

- Cada canal dispone de:
  - un registro de direcciones de 16 bits:
    - dir. a partir de la cual se comenzará a realizar la transfer.
  - un contador de 16 bits:
    - número de bytes a transmitir (65536 como máximo).
- Además existen otros registros para establecer las condiciones de trabajo de cada canal.



# Ejemplo de DMAC: 8237

## ■ Funcionamiento del 8237:

- Ciclo inactivo:
  - Cuando ningún canal está realizando una transferencia.
  - El procesador puede escribir los registros de direcciones, contador de bytes, modos de trabajo y tipos de transferencia.
- Ciclo activo:
  - Un periférico solicita una transferencia activando DREQi.
  - El 8237 activa HRQ (pide los buses al procesador).
  - El procesador pasa los buses a alta impedancia y activa HLDA.
  - El 8237 activa DACKi permitiendo la transferencia al periférico.
  - El 8237 genera IOR#, IOW#, MEMR# y MEMW#.
  - El registro de direcciones se incrementa o decrementa para apuntar a la siguiente posición de memoria.
  - El contador de bytes se decrementa. Si pasa de 0000 a FFFF se genera la señal EOP (*End Of Process*).
  - El 8237 retira la señal HRQ.

# Ejemplo de DMAC: 8237

## ■ Tipos de transferencia:

- Lectura:
  - De memoria a periférico.
  - Se activan MEMR# e IOW#.
- Escritura:
  - De periférico a memoria.
  - Se activan IOR# y MEMW#.
- Memoria a memoria:
  - Se copian bytes desde un área de memoria, indicada por el registro de direcciones del canal 0, a otra área de memoria apuntada por el registro de direcciones del canal 1.

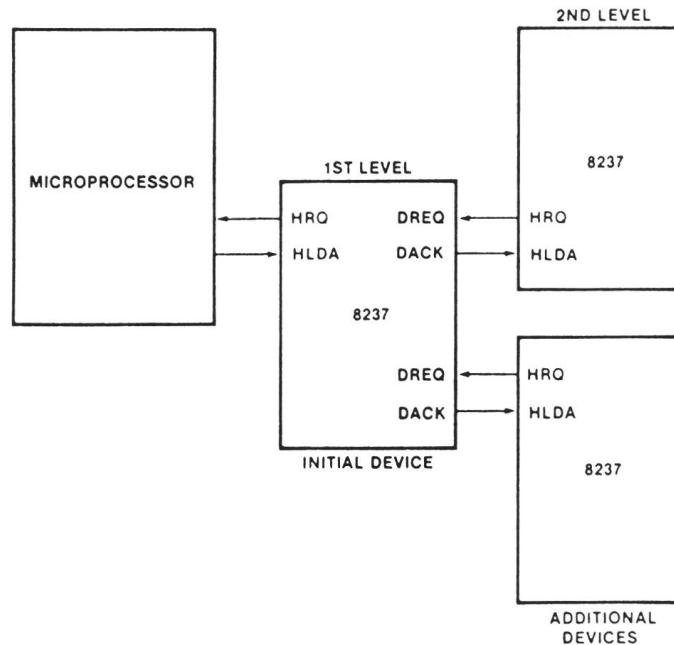
# Ejemplo de DMAC: 8237

## ■ Modos de transferencia:

- Transferencia simple:
  - Se transfiere un byte y se devuelve el control a la CPU.
  - Contador--; si pasa de 0000 a FFFF se activa EOP.
  - La línea DREQi debe mantenerse activa hasta que el 8237 responda con DACKi.
- Transferencia en bloques:
  - Se transfieren tantos bytes, de forma consecutiva, como indique el contador.
  - Cuando el contador pasa de 0000 a FFFF se genera EOP.
  - La línea DREQi debe mantenerse activa hasta que el 8237 responda con DACKi.
- Transferencia por demanda:
  - No se tiene en cuenta el contador.
  - Transferencia continua hasta que desaparezca la activación de DREQi, o hasta que se introduzca un nivel bajo por EOP.

# Ejemplo de DMAC: 8237

- Modo cascada:
  - El canal correspondiente no actúa como un canal de transferencia propiamente dicho, sino que da paso a un segundo 8237, que es el que la realiza.
  - Cada canal puede activar otro 8237. un 8237 puede activar a 4.
  - Los otros 8237 podrían controlar a otros 8237 en cascada.



# Ejemplo de DMAC: 8237

## ■ Otras opciones:

- Autoincremento / autodecremento:
  - La transferencia puede seguir un curso ascendente (desde la dirección de memoria inicial a otras superiores) o descendente.
- Autoinicio:
  - Cuando finaliza la transferencia de un bloque, se reponen los valores originales de dirección y contador, y se comienza de nuevo automáticamente.
- Prioridad fija:
  - El canal 0 es el de mayor privilegio, el 3 el menos prioritario.
- Prioridad rotatoria:
  - Cuando un canal termina una transferencia, pasa a tener el mínimo nivel de privilegio, y el siguiente canal, el máximo nivel.
- Nivel de activación (a 0 ó a 1):
  - Es posible fijar cuáles son los niveles activos (0 ó 1) de las líneas DREQ y DACK.

# Ejemplo de DMAC: 8237

## ■ Registros del 8237:

Name	Size	Number
Base Address Registers	16 bits	4
Base Word Count Registers	16 bits	4
Current Address Registers	16 bits	4
Current Word Count Registers	16 bits	4
Temporary Address Register	16 bits	1
Temporary Word Count Register	16 bits	1
Status Register	8 bits	1
Command Register	8 bits	1
Temporary Register	8 bits	1
Mode Registers	6 bits	4
Mask Register	4 bits	1
Request Register	4 bits	1

# Ejemplo de DMAC: 8237

## Direcciones de registros y órdenes:

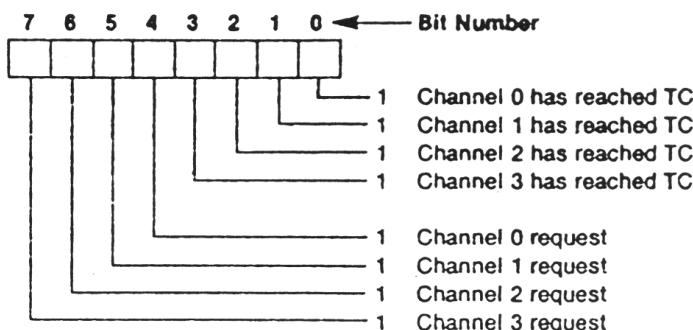
Channel	Register	Operation	Signals						Internal Flip-Flop	Data Bus DB0-DB7	
			CS	IOR	IOW	A3	A2	A1	A0		
0	Base and Current Address	Write	0	1	0	0	0	0	0	0	A0-A7 A8-A15
	Current Address	Read	0	0	1	0	0	0	0	0	A0-A7 A8-A15
	Base and Current Word Count	Write	0	1	0	0	0	0	1	0	W0-W7 W8-W15
	Current Word Count	Read	0	0	1	0	0	0	1	0	W0-W7 W8-W15
1	Base and Current Address	Write	0	1	0	0	0	1	0	0	A0-A7 A8-A15
	Current Address	Read	0	0	1	0	0	1	0	0	A0-A7 A8-A15
	Base and Current Word Count	Write	0	1	0	0	0	1	1	0	W0-W7 W8-W15
	Current Word Count	Read	0	0	1	0	0	1	1	0	W0-W7 W8-W15
2	Base and Current Address	Write	0	1	0	0	1	0	0	0	A0-A7 A8-A15
	Current Address	Read	0	1	0	0	1	0	0	1	A0-A7 A8-A15
	Base and Current Word Count	Write	0	1	0	0	1	0	1	0	W0-W7 W8-W15
	Current Word Count	Read	0	0	1	0	1	0	1	0	W0-W7 W8-W15
3	Base and Current Address	Write	0	1	0	0	1	1	0	0	A0-A7 A8-A15
	Current Address	Read	0	0	1	0	0	1	1	0	A0-A7 A8-A15
	Base and Current Word Count	Write	0	1	0	0	1	1	1	0	W0-W7 W8-W15
	Current Word Count	Read	0	0	1	0	1	1	1	0	W0-W7 W8-W15

Register	Operation	Signals							
		CS	IOR	IOW	A3	A2	A1	A0	
Command Mode Request Mask Mask Temporary Status	Write	0	1	0	1	0	0	0	
	Write	0	1	0	1	0	1	1	
	Write	0	1	0	1	0	0	1	
	Set/Reset	0	1	0	1	0	1	0	
	Write	0	1	0	1	1	1	1	
	Read	0	0	1	1	1	0	1	
	Read	0	0	1	1	0	0	0	

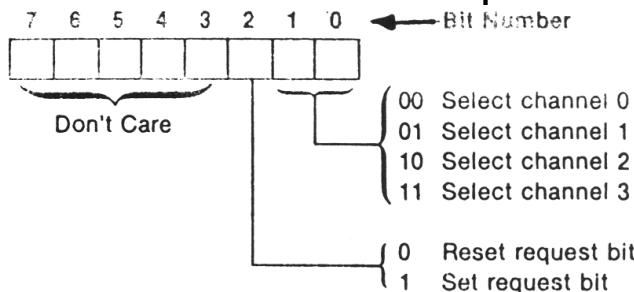
Signals						Operation	
A3	A2	A1	A0	IOR	IOW		
1	0	0	0	0	1	Read Status Register	
1	0	0	0	1	0	Write Command Register	
1	0	0	1	0	1	Illegal	
1	0	0	1	1	0	Write Request Register	
1	0	1	0	0	1	Illegal	
1	0	1	0	1	0	Write Single Mask Register Bit	
1	0	1	1	0	1	Illegal	
1	0	1	1	1	0	Write Mode Register	
1	1	0	0	0	1	Illegal	
1	1	0	0	1	0	Clear Byte Pointer Flip / Flop	
1	1	0	1	0	1	Read Temporary Register	
1	1	0	1	1	0	Master Clear	
1	1	1	0	0	1	Illegal	
1	1	1	0	1	0	Illegal	
1	1	1	1	0	1	Illegal	
1	1	1	1	1	0	Write All Mask Register Bits	

# Ejemplo de DMAC: 8237

- Registros dirección (*Address*) y contador (*Word Count*):
  - Base: valores iniciales
  - Current: valores actuales
- Registro de estado (*Status Register*):
  - Información sobre el estado de los canales.

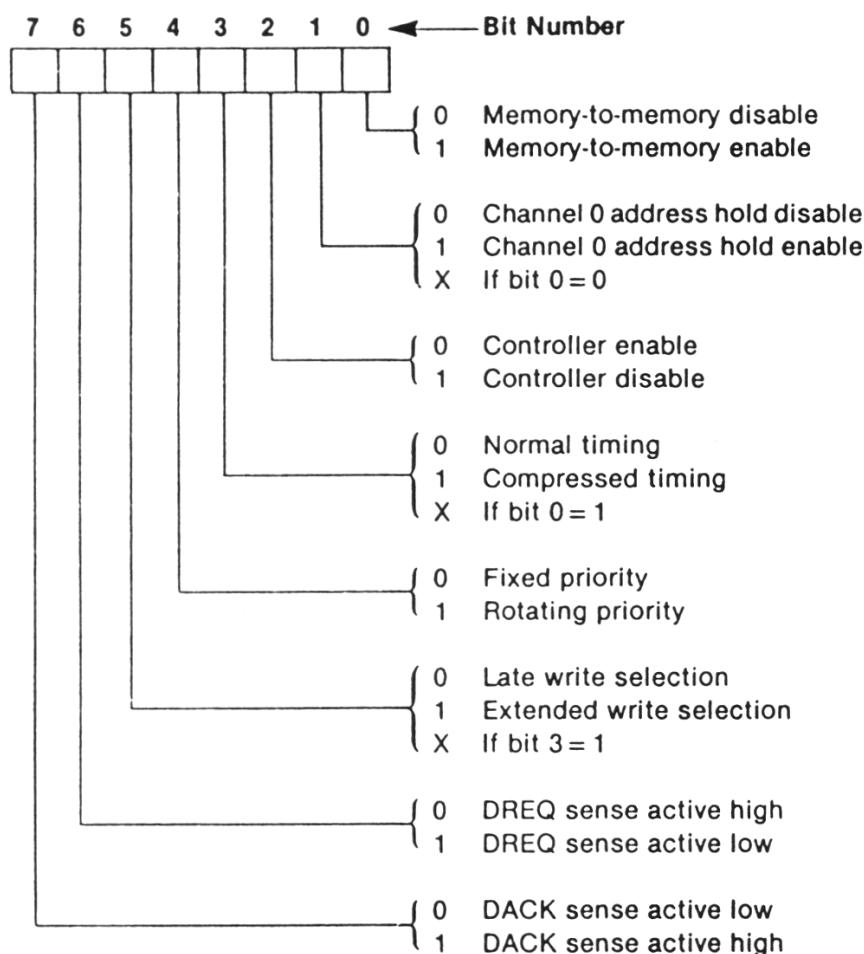


- Registro de peticiones (*Request Register*):
  - Permite comenzar operaciones de DMA por software.



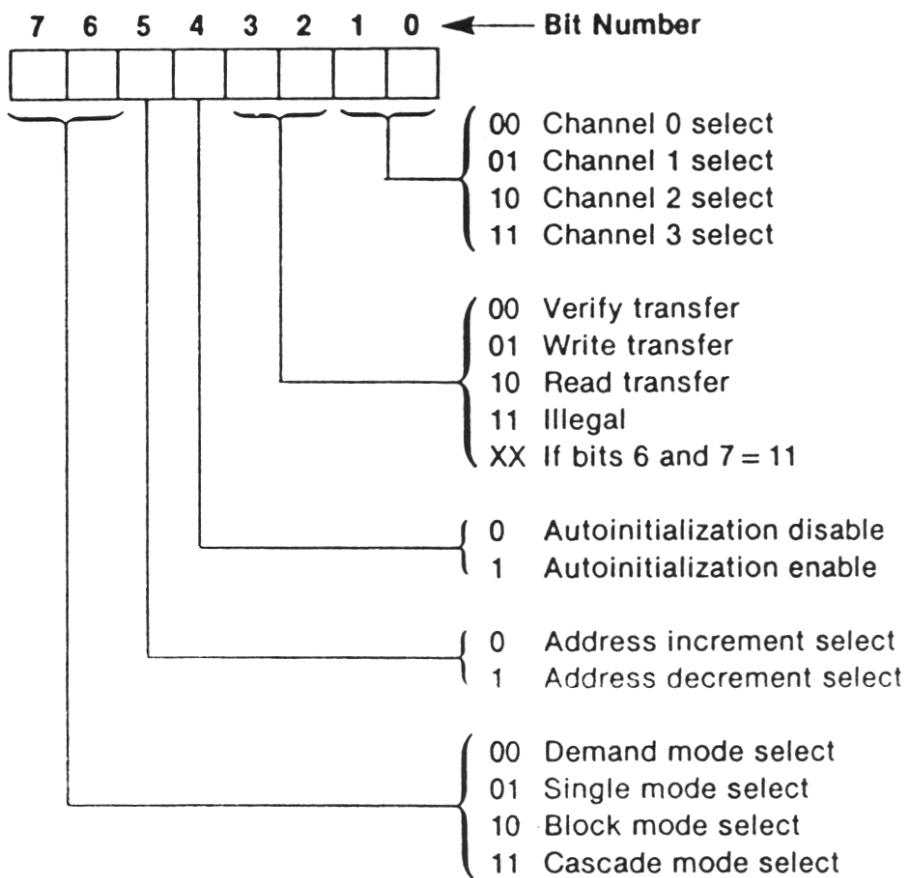
# Ejemplo de DMAC: 8237

- Registro de órdenes (*Command Register*):
  - Controla el funcionamiento del 8237.



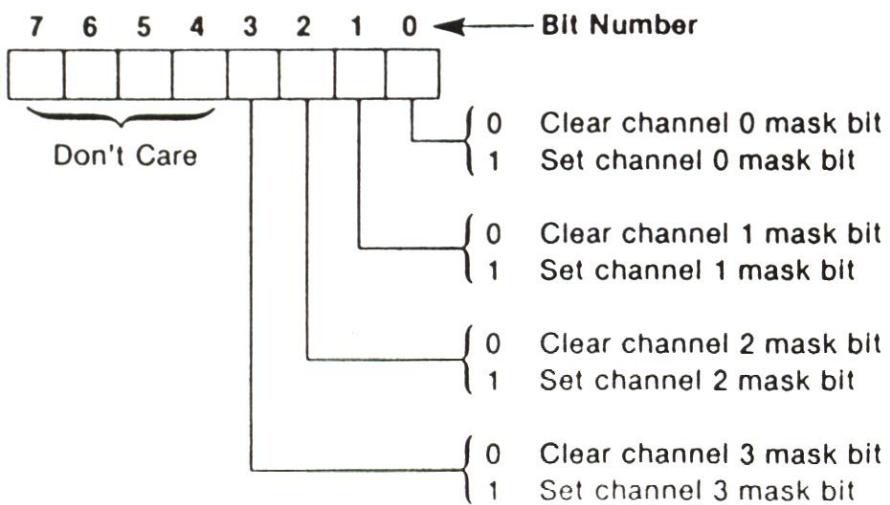
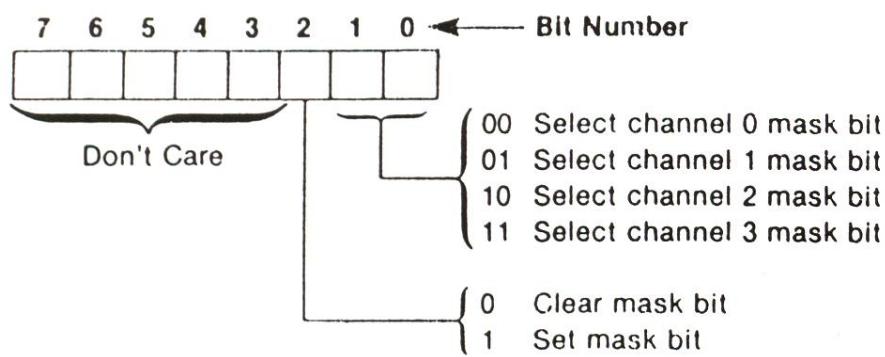
# Ejemplo de DMAC: 8237

- Registros de modo (*Mode Registers*):
  - Modo de transferencia de cada canal.



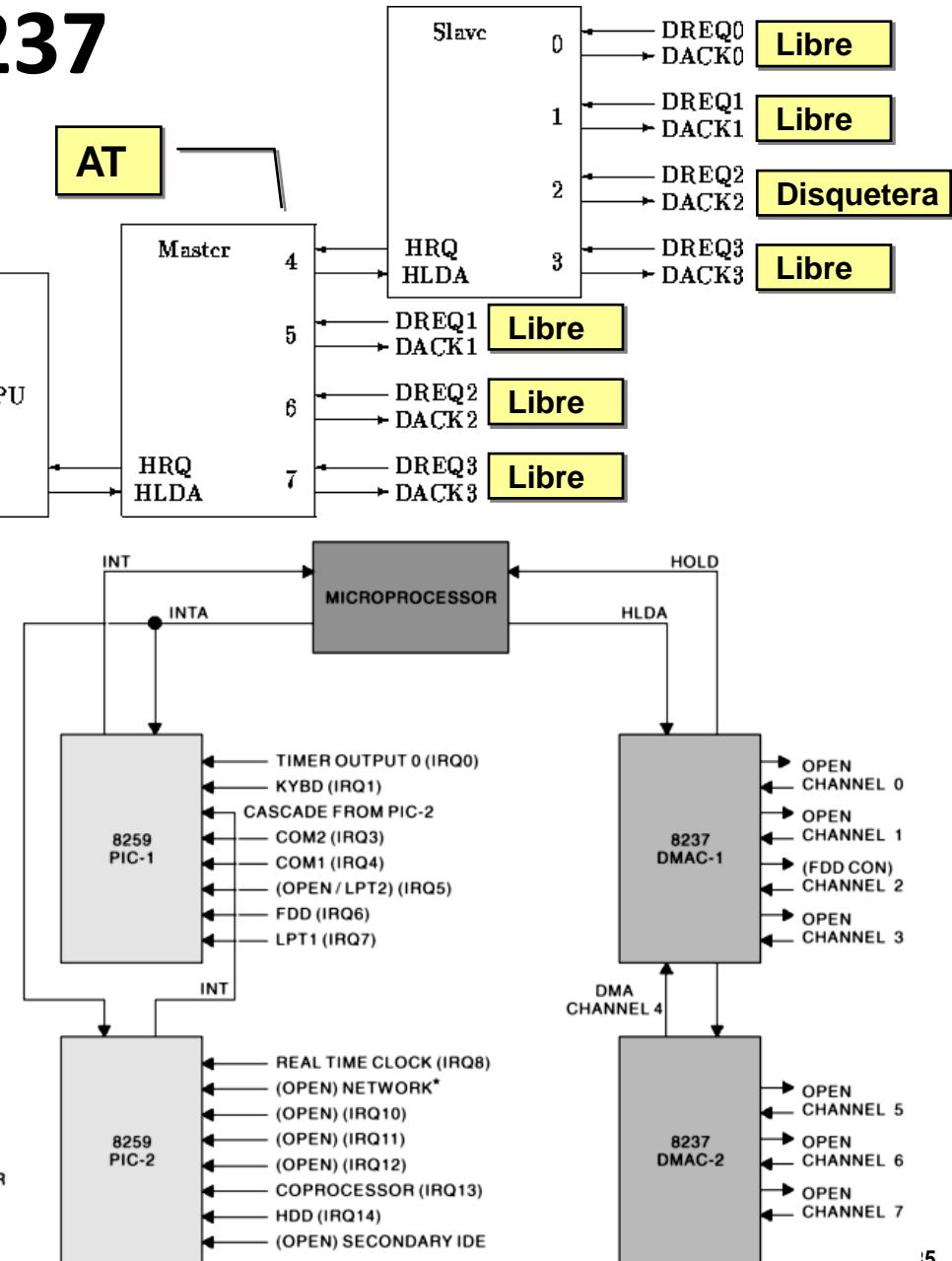
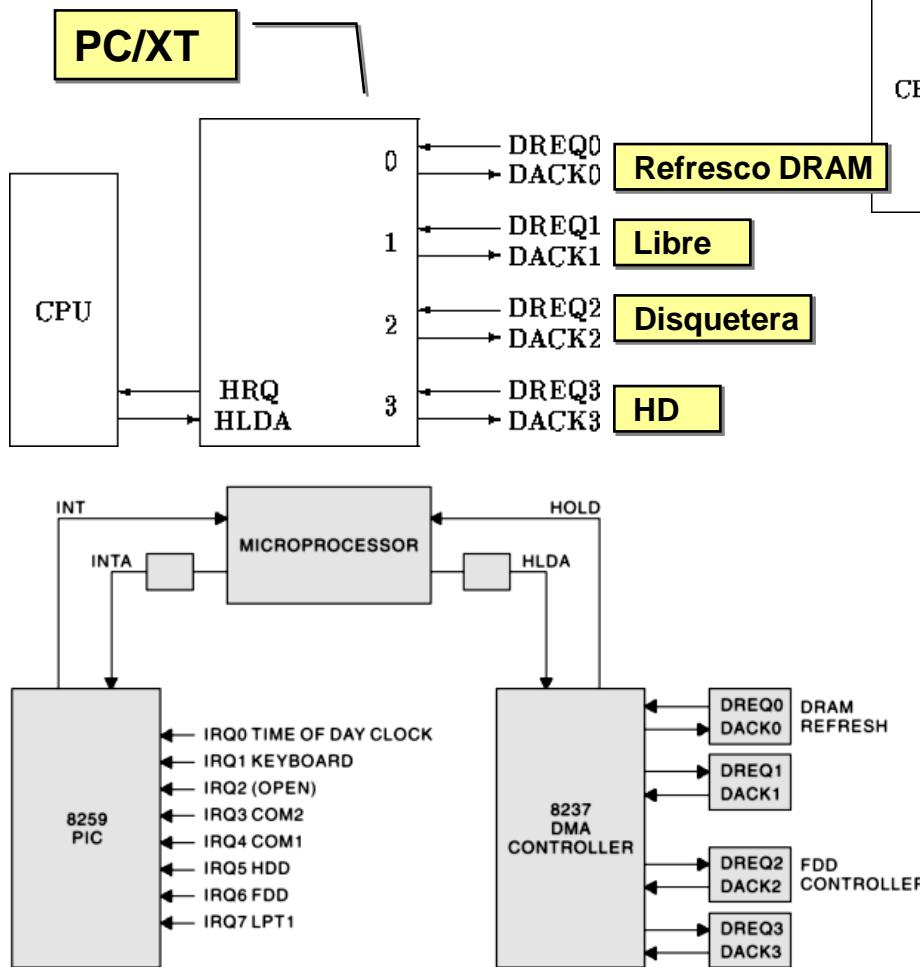
# Ejemplo de DMAC: 8237

- Registro de máscara (*Mask Register*):
  - Permite deshabilitar (bit = 1) algunas líneas DREQi.



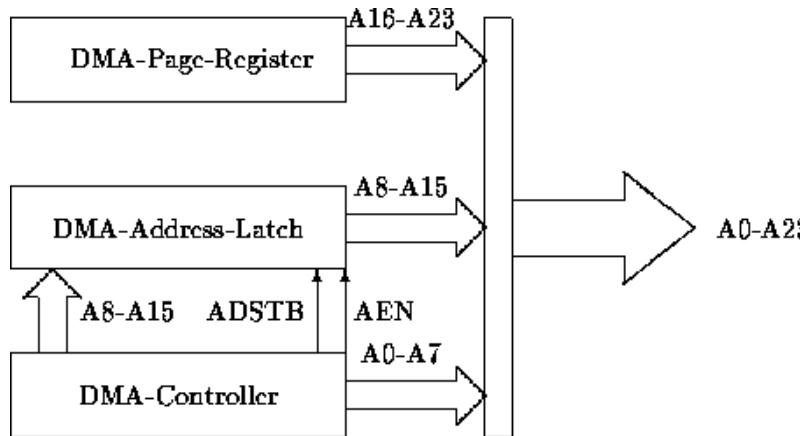
# Ejemplo de DMAC: 8237

## ■ El 8237 en PC/XT y AT:



# Ejemplo de DMAC: 8237

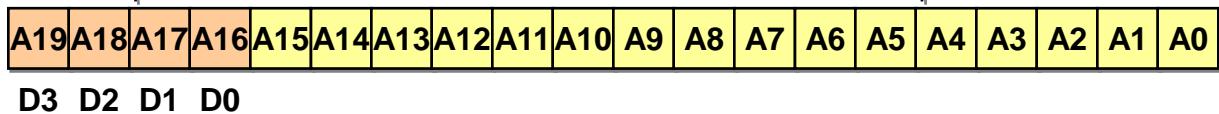
- Generación de direcciones de DMA en PC/XT y AT:



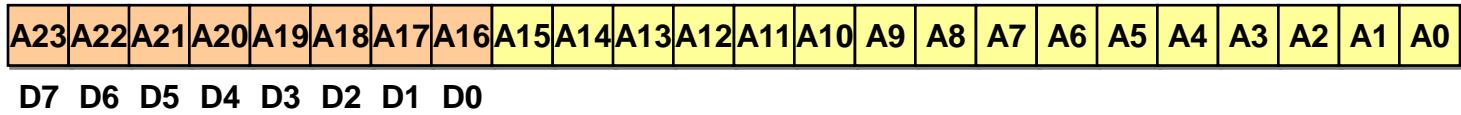
**Registro de página**

**Registro de dirección del 8237**

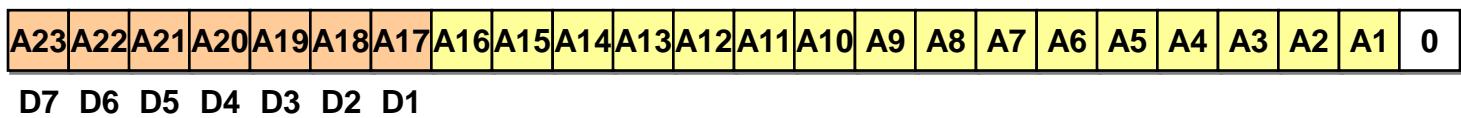
**PC/XT**  
(16 blq. de 64 KB)



**AT 8 bits**  
(256 blq. de 64 KB)



**AT 16 bits**  
(128 blq. de 128 KB)  
Transf. de palabras  
en direcc. pares



# Ejemplo de DMAC: 8237

## ■ Direcciones de puertos importantes:

