Interruzioni (1)

• Un ***interrupt*** è un evento inatteso che "interrompe" il normale flusso di esecuzione del programma e causa il trasferimento del controllo ad una apposita procedura di servizio in grado di gestire opportunamente l'evento che ha originato l'interruzione. Terminata la procedura di servizio, l'esecuzione riprende secondo il flusso normale.

• Classi di interruzioni:

•***Interrupt Hardware:*** interrupt generati da dispositivi *esterni* ed inoltrati alla CPU mediante un apposito pin (INT). Il caso più importante è quello delle richieste di servizio generate dalle interfacce di I/O. Questi interrupt sono *mascherabili*: tramite un flag (DLX: IEN, IA: IF) è possibile rendere la CPU insensibile alla loro ricezione.
Tipicamente, le CPU hanno anche un pin (NMI) tramite cui viene inoltrato un Interrupt Hardware non mascherabile (tipicamente inviato alla CPU in situazioni di emergenza, ad esempio una caduta di tensione).

•***Exceptions (Eccezioni):*** sono interrupt generati da circuiti situati all'*interno* del processore in seguito al verificarsi di eventi legati all'esecuzione del programma. Esempi: divisione per zero, overflow in un'operazione aritmetica, fetch di un opcode ignoto, esecuzione "single-step", break-point, accesso in memoria non allineato (se la CPU richiede l'allineamento), page-fault (in CPU con supporto per la memoria virtuale), protection-fault (violazione della protezione, in CPU che supportano la protezione)….

Interruzioni (2)

• ***Interrupt Software:*** il programma, tramite un apposita istruzione, richiede esplicitamente il trasferimento del controllo ad una procedura di servizio. Ad esempio, nell'IA esiste l'istruzione "*INT n*" che trasferisce il controllo ad una procedura di servizio specificata mediante l'immediato *n*. Anche nel DLX esiste un'istruzione analoga (*TRAP*).

L'interrupt software è molto simile ad una chiamata a procedura; inoltre, si osservi che rispetto alla definizione di interrupt data nella slide precedente, nel caso di interrupt software viene a mancare il requisito di evento "inatteso". Tuttavia, l'interrupt software viene incluso nella classe degli interrupt perché la *modalità di trasferimento del controllo* è quella degli interrupt HW e delle Exceptions e non quella delle normali chiamate a procedura.

L'invocazione di una procedura mediante interrupt software (al posto della normale chiamata a procedura) viene usata tipicamente per invocare procedure del *Sistema Operativo* (dette *System Call*).

Questo perché la modalità di trasferimento del controllo tipicamente impiegata nel caso di interrupt (*trasferimento di controllo vettorizzato*) non richiede che il programma sia linkato alle procedure del Sistema Operativo. Si tratta infatti di una chiamata "*anonima*", effettuata senza specificare l'indirizzo della procedura chiamata. Inoltre, il trasferimento del controllo impiegato nel caso di interrupt implica generalmente una maggiore separazione degli ambienti di esecuzione ("*contesto*") di chiamante e chiamato.

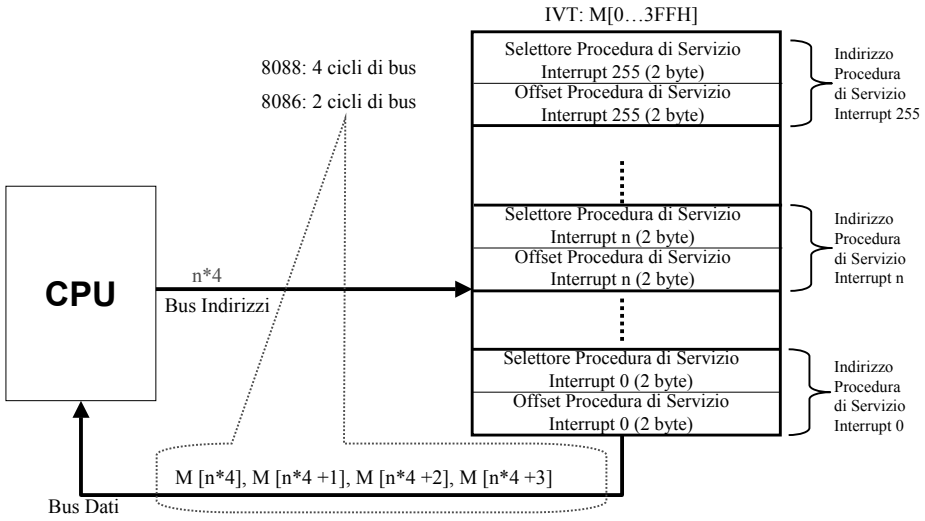
µP 8088 - Classi di interruzione

Per il µP 8088 esistono 4 modi possibili per richiedere il servizio di un interrupt:

- 1) Interrupt di origine software generati internamente alla CPU
- 2) Interrupt di origine software generati da istruzioni di un programma
- 3) Interrupt di origine hardware non mascherabili
- 4) Interrupt di origine hardware mascherabili

Trasferimento del controllo vettorizzato nelle CPU iA16 (1)

L'interrupt type (*n*) è un numero naturale ad 8 bit: esistono allora 256 interrupt distinti (0…255) e la IVT è una tabella contenente 256 elementi.
Gli elementi hanno dimensione 4 byte, la IVT inizia all'indirizzo fisico 0.



Periferiche I/O 2° 4

Trasferimento del controllo vettorizzato nelle CPU iA16 (2)

• Sequenza di operazioni eseguite dalla CPU quando deve trasferire il controllo alla procedura di servizio di un'interrupt:

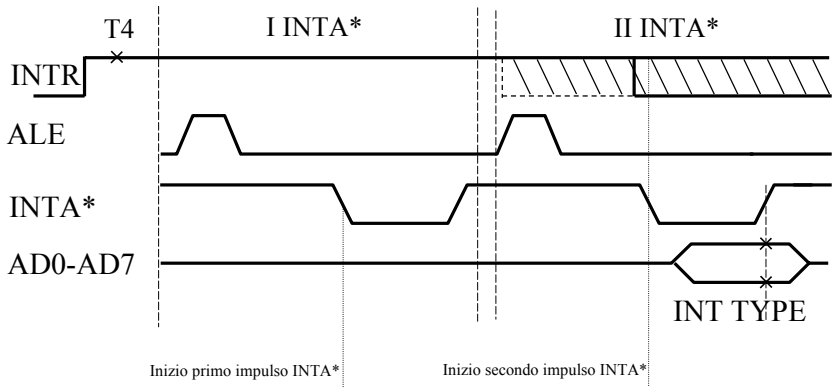
1. Salvataggio sullo stack del registro dei FLAGS (PUSH FLAGS)
2. Azzeramento del bit IF del registro dei FLAGS (CLI)
3. Salvataggio sullo stack di CS (PUSH CS)
4. Salvataggio sullo stack di IP (PUSH IP)
5. Lettura dell'offset della procedura di servizio da IVT e trasferimento del valore letto in IP (IP ← M[n*4])
6. Lettura del selettore della procedura di servizio da IVT e trasferimento del valore letto in CS (CS ← M[n*4 +2])

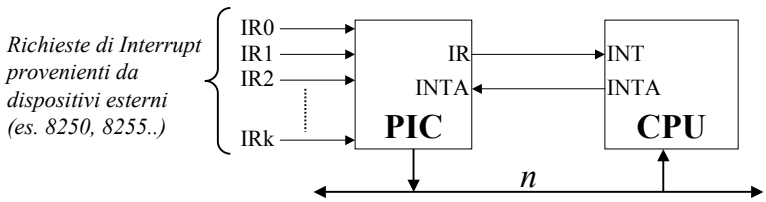
*7. Fetch della prossima istruzione all'indirizzo CS*16 + IP (cioè fetch della prima istruzione della procedura di servizio dell'interrupt).*

• L'azzeramento di IF disabilita il riconoscimento degli *interrupt hardware* all'interno della procedura di servizio (*interrupt hardware mascherati*). Tuttavia, nella procedura di servizio è possibile riabilitare il riconoscimento degli *interrupt hardware* tramite l'istruzione STI ("*Set Interrupt*").

• La procedura di servizio deve terminare con le istruzioni STI+IRET ("Interrupt Return") che riabilitano l'ingresso INT e ripristinano dallo stack IP, CS ed il registro dei FLAGS (POP IP, POP CS, POP FLAGS).

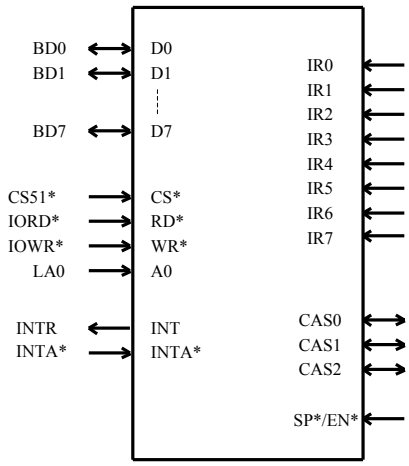
µP 8088 - Ciclo di Interrupt Acknowledge





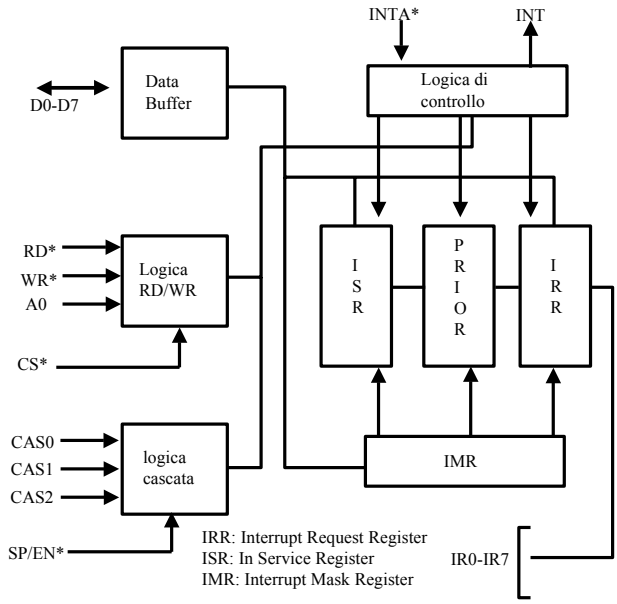
Periferiche I/O 2° 6

Controllore di Interrupt programmabile (PIC) 8259



Periferiche I/O 2° 7

8259 - Schema interno



8259 - Sequenza di Intr

1) Si attivano uno o piu' ingressi IR (i corrispondenti bit nel registro IRR vengono posti a 1)

2) L'8259 invia un segnale di INTR alla CPU

3) La CPU riconosce l'INTR e invia un primo impulso INTA* in corrispondenza del quale viene portato a 1 il bit del registro ISR relativo alla richiesta di interruzione di priorità più alta e portato a 0 il corrispondente bit del registro IRR

4) La CPU invia un secondo impulso INTA* durante il quale l'8259 rilascia sul data bus il codice di interruzione (INTERRUPT TYPE VECTOR)

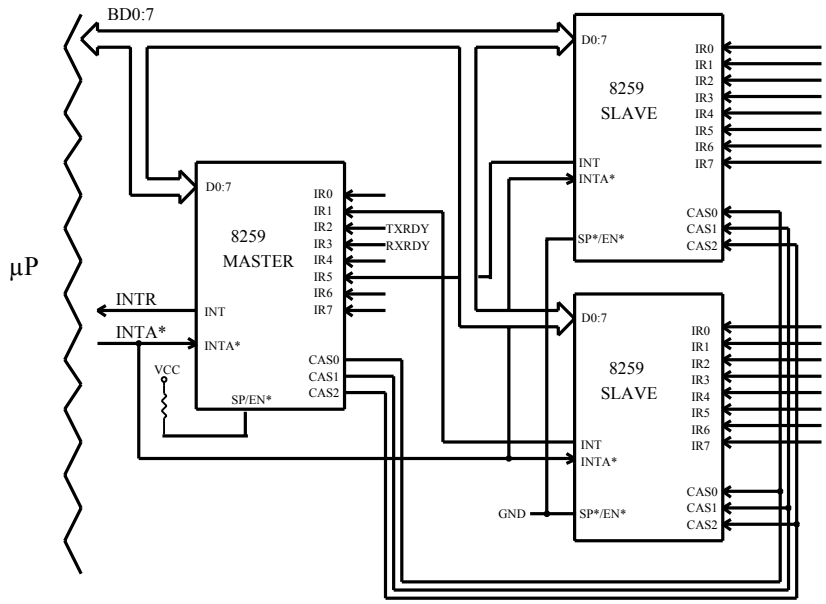
5) Al termine viene portato a 0 il bit nel registro ISR

CODICE DI INTERRUZIONE

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

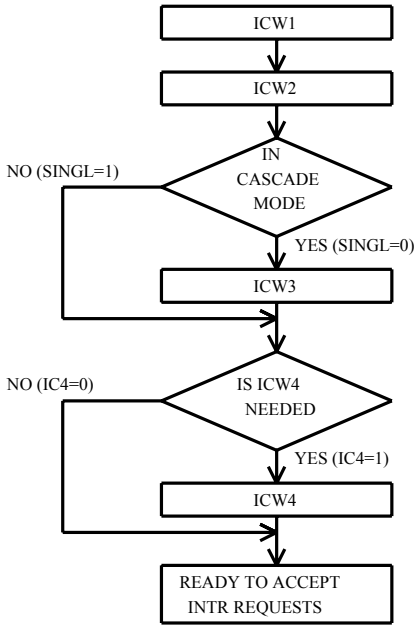
Periferiche I/O 2° 9

8259 - Disposizione in cascata



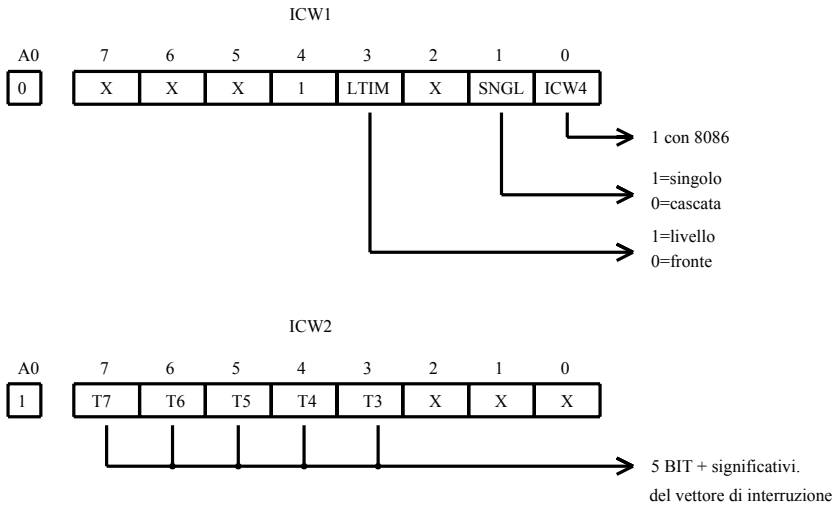
Periferiche I/O 2° 10

8259 - Programmazione



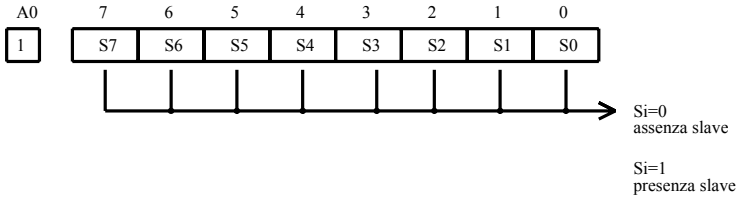
Periferiche I/O 2° 11

8259 - ICW1 e ICW2

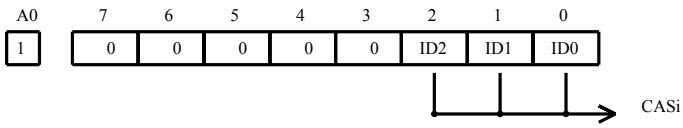


8259 - ICW3 master e slave

ICW3 MASTER



ICW3 SLAVE



Periferiche I/O 2° 13

8259 - OCW1

OCW1

A0	7	6	5	4	3	2	1	0
1	M7	M6	M5	M4	M3	M2	M1	M0

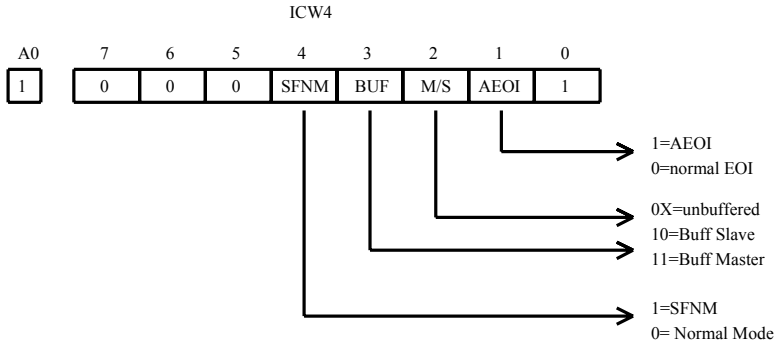
INTERRUPT MASK

0=RESET

1=SET

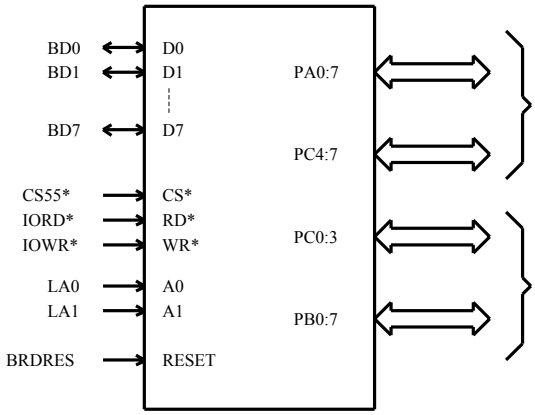
Periferiche I/O 2° 14

8259 - ICW4



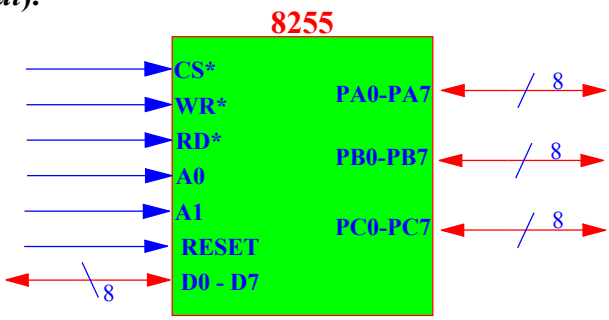
Periferiche I/O 2° 15

Interfaccia parallela 8255



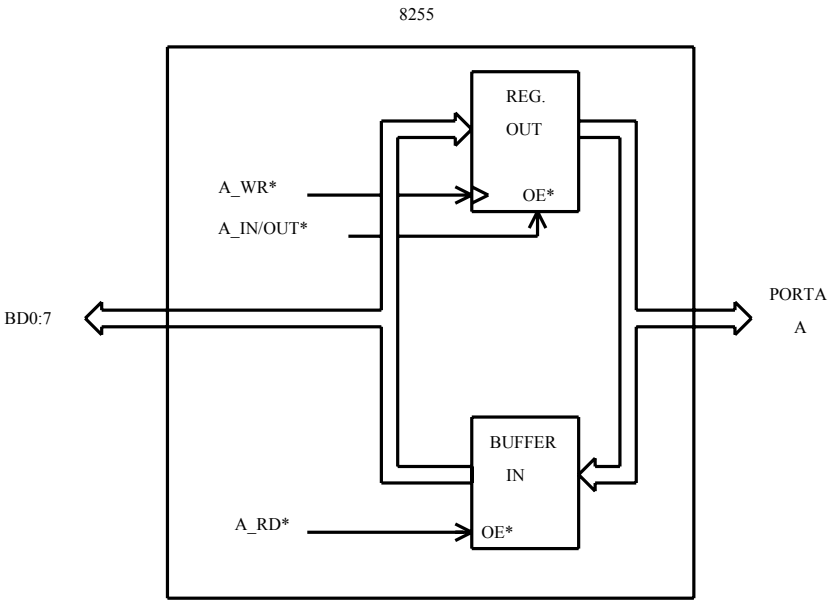
Interfaccia parallela 8255

• **Interfaccia che supporta la gestione differenziata e programmabile (3 diversi *modi di funzionamento*) di tre porte parallele bidirezionali (*input* o *output*).**



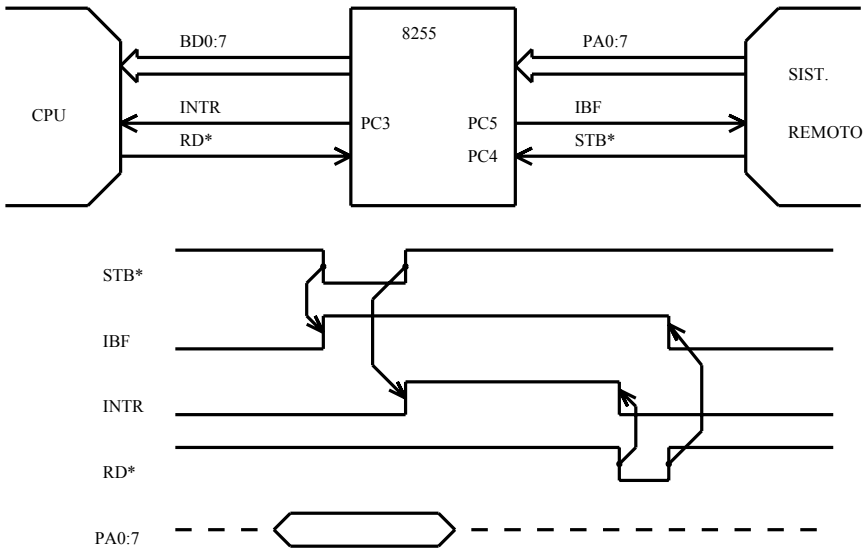
• **L'interfaccia dispone di 4 registi interni che consentono lo scambio di dati, informazioni di controllo (programmazione) ed informazioni di stato con la CPU.**

8255 - Modo 0 (schema interno)



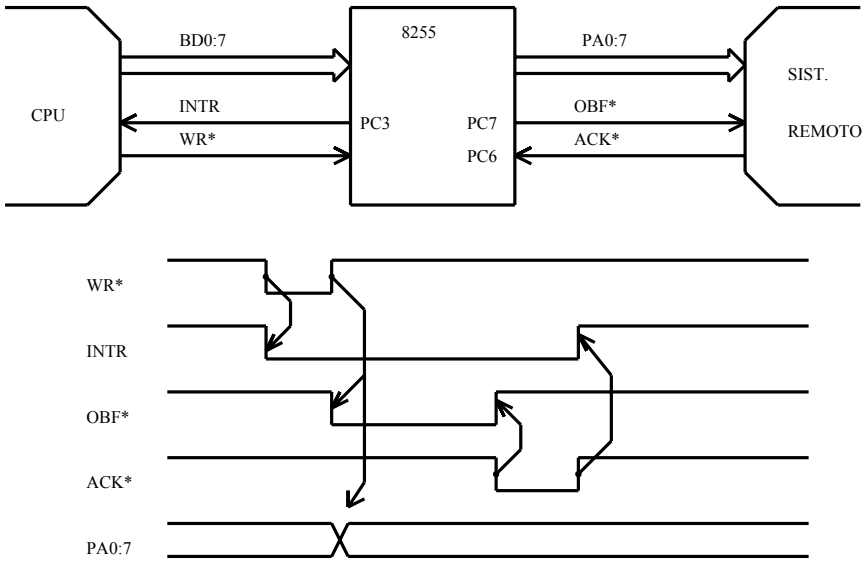
Periferiche I/O 2° 18

8255 - Modo 1 Input



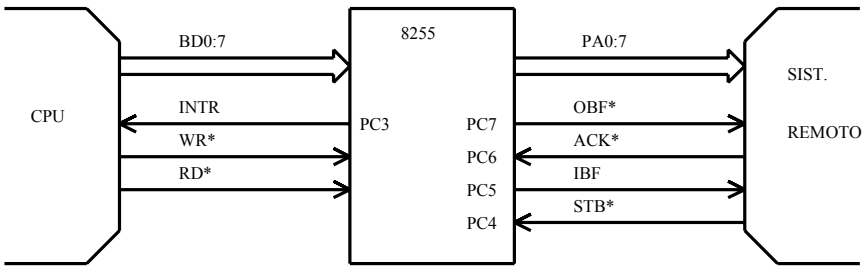
Periferiche I/O 2° 19

8255 - Modo 1 Output



Periferiche I/O 2° 20

8255 - Modo 2

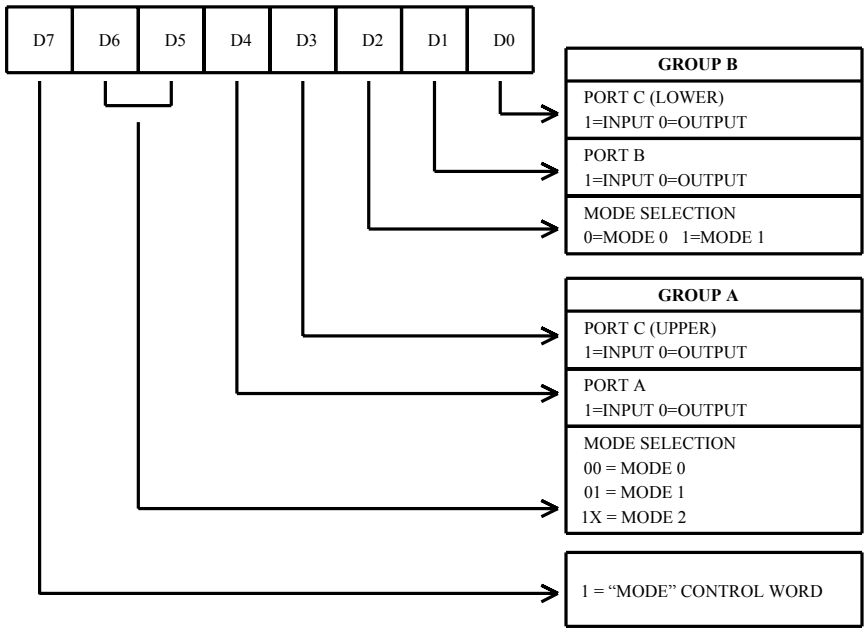


Periferiche I/O 2° 21

8255 - Programmazione

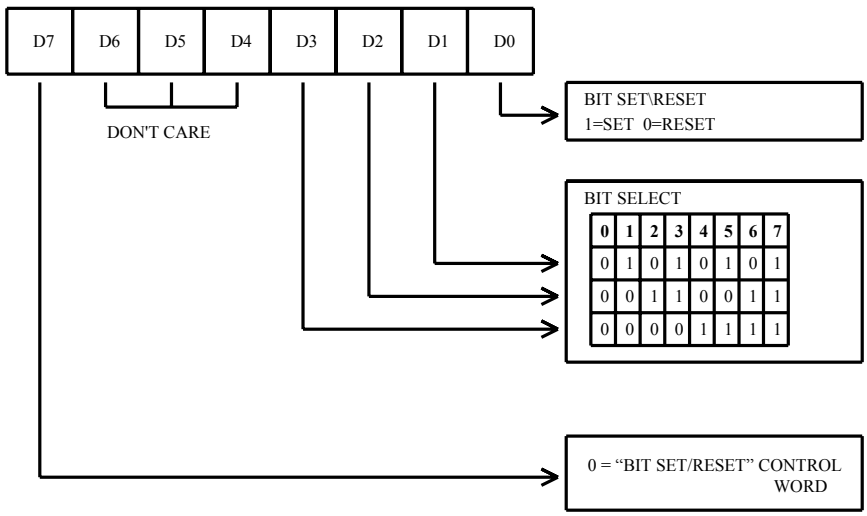
	A1	A0	READ	WRITE
IND BASE + 0	0	0	PORTA A	PORTA A
IND BASE + 1	0	1	PORTA B	PORTA B
IND BASE + 2	1	0	PORTA C	PORTA C
IND BASE + 3	1	1		CONTROL WORD

8255 - Mode Control Word



Periferiche I/O 2° 23

8255 - Bit set/reset Control Word



Periferiche I/O 2° 24

8255 – Routine di inizializzazione

```
CWMODE EQU B0h (porta A in modo1-input, don’t care
su porte B e C)


INI8255 PROC FAR
        PUSH AX
        MOV AL, CWMODE
        OUT INDB55+3, AL ; INDB55 indir base 8255
        POP AX
        RET
INI8255 ENDP
```

8255 – Gestione a Polling e a Interrupt

```
poll_read_8255A   PROC  FAR
; la procedura legge un carattere dalla porta A
; in modo1-input dell'8255 interrogando il registro
; finchè non trova un carattere ricevuto
; il bit 3 della porta C contiene la copia SW del pin
; "INTRA" – La porta C si trova all'INDB55+2
; Attesa che il buffer di ricezione sia pieno...
wait_pieno: IN    AL, INDB55+2
            TEST  AL, 04H
            JZ    wait_pieno
; lettura del carattere ricevuto
            IN    AL, INDB55
            RET
poll_read_8255A ENDP
```

```
int_read_8255A   PROC  FAR
; la procedura legge un carattere dalla porta A
; in modo1-input dell'8255 ed è eseguita su richiesta
; del PIC8259 a sua volta triggerato da "INTRA"
; dell'8255 stesso
; La procedura ritorna il carattere letto al chiamante
; tramite AL
            IN AL, INDB55
            IRET
int_read_8255A ENDP
```