Esercizi Assembly 9

M. Rebaudengo – R. Ferrero

Politecnico di Torino
Dipartimento di Automatica e Informatica

Emulazione di periferiche

- L'emulatore **emu8086** consente di emulare periferiche utilizzando appositi programmi che sono eseguiti contemporaneamente all'emulatore 8086, e che scambiano con esso dati e segnali controllo mediante alcuni file di sistema.

Emulazione di periferiche [cont.]

- Con questa esercitazione è fornito l'archivio `periferiche.zip` , che contiene i file per l'emulazione dei dispositivi Intel 8253, 8255 e 8259
  - In laboratorio **NON** è necessario installare alcun file
  - Sul proprio PC, in ambiente Windows occorre seguire le indicazioni contenute nel file `readme.txt`
  - È richiesta la Java Virtual Machine
  - Stiamo lavorando per estendere il supporto in ambiente Linux e Mac/OS X.

Emulazione del dispositivo 8255

- L'interazione con le periferiche è effettuata utilizzando le istruzioni Assembly `IN` e `OUT`
- Per consentire l'avvio dell'emulatore 8255 è necessario anteporre al codice la seguente direttiva per l'emulatore:

  ```
  #START=8255.exe#
  ```

- Il dispositivo Intel 8255 è accessibile a partire dall'indirizzo I/O `0x80`.

Emulazione del dispositivo 8255

[cont.]

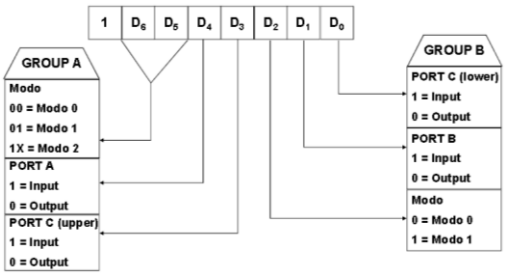
- L'emulatore 8255 può lavorare con o senza la presenza di un interrupt controller (Intel 8259)
- Per la presente esercitazione, si lavorerà con il solo 8255:
  - A ogni esecuzione, un messaggio segnalerà l'assenza del modulo 8259
  - Selezionare dal menù *File**Use with PIC 8259**No*
  - Ricordarsi di chiudere la finestra **emu8255** al termine di ogni esecuzione
  - NB: In modalità *Run* impostare *delay* ≥ 100 ms.

Esercizio 1

- Scrivere un programma in grado di scrivere i caratteri 'O' e 'K', rispettivamente sulla porta A e sulla porta B del dispositivo Intel 8255 (indirizzo 0x80); successivamente acquisire un valore binario dalla porta C nella variabile *byte* `lettura`
- N.B: Eseguire il programma in modalità step-by-step.

Implementazione

- Prima di tutto è necessario configurare i gruppi A e B in modo 0, le porte A e B in direzione *output* e la porta C in direzione *input*
- Si procede quindi con le operazioni di *input*/*output*
- Registro di configurazione:



Codice

```
PORTA    EQU 80h
PORTB    EQU PORTA+1
PORTC    EQU PORTA+2
CONTROL  EQU PORTA+3

         #start=8255.exe#
         .model small
         .stack
         .data
lettura  db ?
         .code
         .startup
         MOV DX, CONTROL
         MOV AL, 10001001b
         OUT DX, AL
         MOV DX, PORTA
         MOV AL, 'O'
         OUT DX, AL
         MOV DX, PORTB
         MOV AL, 'K'
         OUT DX, AL
         MOV DX, PORTC
         IN AL, DX      ; impostare valore della porta C PRIMA dell'esecuzione di questa istruzione
         MOV lettura, AL
         .exit
         end
```

Esercizio 2

- Scrivere un programma in grado di scrivere alternativamente sulle porte A e B dell'Intel 8255 i valori decrescenti da 255 a 0

  A ← 255

  B ← 254

  A ← 253

  B ← 252

  […]

  B ← 0

Codice

```
PORTA    EQU 80h
PORTB    EQU PORTA+1
PORTC    EQU PORTA+2
CONTROL  EQU PORTA+3
         #start=8255.exe#
         .model small
         .stack
         .data
         .code
         .startup
         MOV DX, CONTROL
         MOV AL, 10000000b
         OUT DX, AL
         MOV CX, 255
ciclo:   MOV AL, CL
         TEST CL, 1
         JZ salto
         MOV DX, PORTA
         JMP salto2
salto:   MOV DX, PORTB
salto2:  OUT DX, AL
         DEC CX
         JNS ciclo
         .exit
         end
```

Esercizio 3

- Si configuri l'Intel 8255 in modo 0 per i gruppi A e B, con le porte A e C in modalità di *input*, e la porta B in *output*
- Si scriva una procedura in grado di leggere un *byte* dalla porta A, e se corrispondente a un carattere minuscolo lo converta in maiuscolo e lo scriva sulla porta B
- Si scriva quindi un programma che interroghi periodicamente la porta C e, al riconoscere di una transizione 0→1 sul bit meno significativo di essa, lanci la procedura precedentemente realizzata.

Codice

```
PORTA   EQU 80h
PORTB   EQU PORTA+1
PORTC   EQU PORTA+2
CONTROL EQU PORTA+3
        #start=8255.exe#
        .model small
        .stack
        .data
        .code
        .startup
        MOV DX, CONTROL
        MOV AL, 10011001b
        OUT DX, AL
        XOR AH, AH  ;vecchio valore di PORT C
                    ; (suppongo inizialmente 0)
        MOV DX, PORTC
ciclo:  IN AL, DX
        TEST AH, 1
        JNZ next
        TEST AL, 1
        JZ next
        CALL converti
next:   MOV AH, AL
        JMP ciclo  ; ciclo infinito
        .exit
```

```
converti proc
         PUSH AX
         PUSH DX
         MOV DX, PORTA
         IN AL, DX
         CMP AL, 'a'
         JB ritorno
         CMP AL, 'z'
         JA ritorno
         MOV DX, PORTB
         ADD AL, 'A'-'a'
         OUT DX, AL
ritorno: POP DX
         POP AX
         RET
converti endp
end
```

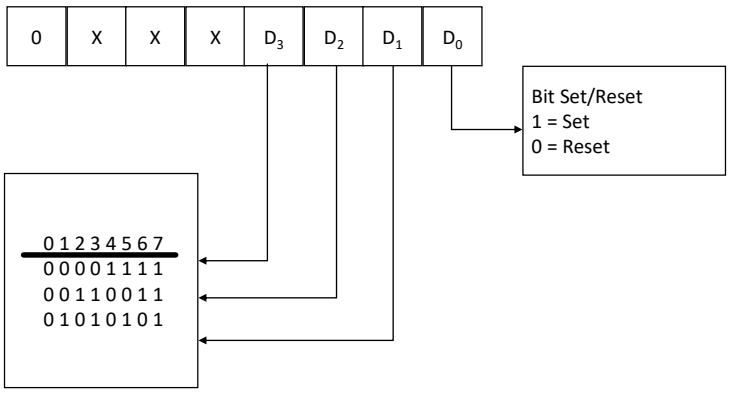
Esercizio 4

- Si scriva un programma in grado di leggere due *byte* a e b rispettivamente dalle porte A e B. Successivamente, deve essere eseguita la seguente operazione logica:

  `NOT (a XOR b)`

- Il risultato deve essere salvato nella variabile byte `ris` e quindi scritto sulla porta C dell'8255 utilizzando la modalità *Single Bit Set/Reset*, a partire dal bit più significativo.

Single Bit Set/Reset



Attraverso un'operazione di scrittura sul Registro di Controllo si può forzare il valore di un singolo bit della porta C.

Codice

```
PORTA    EQU 80h
PORTB    EQU PORTA+1
PORTC    EQU PORTA+2
CONTROL  EQU PORTA+3
         #start=8255.exe#
         .model small
         .stack
         .data
a        db ?
b        db ?
ris      db ?
         .code
         .startup
         MOV DX, CONTROL
         MOV AL, 10010010b
         OUT DX, AL
         MOV DX, PORTA
         IN AL, DX
         MOV a, AL
         MOV DX, PORTB
         IN AL, DX
         MOV b, AL
         XOR AL, a
         NOT AL
         MOV AH, AL
         MOV ris, AL

         MOV CX, 7
         MOV DX, CONTROL
ciclo:   MOV AL, CL
         SHL AL, 1
         SHL AH, 1
         ADC AL, 0
         OUT DX, AL
         DEC CX
         JNS ciclo

         .exit
         end
```