

Esercizi Assembly 10

M. Rebaudengo – R. Ferrero

Politecnico di Torino
Dipartimento di Automatica e Informatica

Esercizio 1

- Si scriva un programma in grado di leggere periodicamente le porte A e B del modulo 8255 e fornisca sulla porta C la somma dei due valori (da considerare espressi in complemento a 2). Eventuali condizioni di *overflow* devono essere segnalate con il valore 0FFh.
- La lettura deve essere fatta a intervalli regolari di 5 ms: si realizzi un opportuno ciclo di ritardo, considerando una frequenza di *clock* di 10 MHz e una media di 10 cicli di clock per istruzione.

Esercizio 2

- Si scriva una procedura in grado di leggere un intero (compreso tra 0 e 7) dalla porta A del modulo 8255, e inverta il valore logico del bit della porta C che ha come indice tale intero.
- Per la lettura del contenuto della porta C programmata in modo 0 – output si veda Q16 all'indirizzo

<https://web.archive.org/web/20131006011609/http://www.intel.com/design/archives/periphrl/docs/7190.HTM>

Esercizio 3

- Sia dato un sistema basato su processore 8086, con il gruppo A del modulo 8255 configurato in modo 0 e la porta A in input
- Si scriva un programma che, periodicamente, lanci una procedura in grado di
 - Leggere dalla porta A dell'8255 un byte proveniente da una periferica esterna
 - Accorpare i byte ricevuti in due chiamate a funzione successive in una *word* corrispondente a un valore intero (è ricevuto prima il *byte* più significativo);
 - Inserire le *word* acquisite in due vettori, *vet_pari* e *vet_disp*, a seconda che ciascuna di esse sia, rispettivamente, pari o dispari.

Esercizio 3 [cont.]

- Si assuma di avere definito e inizializzato le seguenti strutture:

```
vet_pari dw DIM DUP (?)
vet_disp dw DIM DUP (?)
ind_pari db 0
ind_disp db 0
```

dove DIM è una costante di valore massimo 255, mentre `ind_pari` e `ind_disp` contengono l'indice dove scrivere il valore successivo per ciascuno dei due vettori. Qualora uno dei vettori risultasse pieno, la memorizzazione deve ripartire dalla prima posizione (come in un *buffer* circolare).

- Esempio:

Sequenza di byte ricevuti: 01, 0A, 31, 28, 33, 45
vet_pari: 010A, 3128
vet_dispari: 3345

Esercizio 4

- Siano dati:
 - un vettore di *byte* `vet1` contenente DIM elementi (DIM dichiarato come costante)
 - un vettore di *word* `vet2`, della stessa dimensione, non inizializzato.
- Si scriva una procedura **extract** in linguaggio Assembly 8086 in grado di ottenere, a partire dai dati espressi come *byte* in `vet1`, una sequenza di valori su 12 bit componendo *nibble* (insieme di 4 bit) di dati consecutivi, come esemplificato di seguito:

Esercizio 4 [cont.]

dati	risultati
0010 1101	
0100 0010	0000 0010 1101 0100
0100 1011	0000 0010 0100 1011
1000 0001	
0110 0011	0000 1000 0001 0110
1100 0000	0000 0011 1100 0000
1111 1111	
0000 1011	0000 1111 1111 0000

- La procedura deve memorizzare i risultati ottenuti in vet2 (azzerando il *nibble* più significativo), procedendo fino a quando sono disponibili dati su vet1 sufficienti a comporre un risultato; deve inoltre restituire al programma chiamante il numero di risultati memorizzati attraverso il registro DI.
- Si supponga che il programma chiamante lanci la procedura una volta sola. Si utilizzino variabili globali per l'indirizzamento dei vettori.