Esercizi Assembly 3

M. Sonza Reorda – M. Grosso

Politecnico di Torino
Dipartimento di Automatica e Informatica

Esercizio 1

- Si scriva un programma in linguaggio Assembly 8086 che scriva in un vettore definito di 20 elementi di tipo *word* i primi 20 valori della serie di Fibonacci.

- Serie di Fibonacci
  - vet[i] = vet[i-1] + vet[i-2] => vet = 1, 1, 2, 3, 5, 8, …

Esercizio 2

- Si scriva un programma in Assembly 8086 che, presi due vettori di 4 *word* ciascuno come matrici riga e colonna, ne calcoli il prodotto.
- Si ricorda che

  Se $x = (x_1, x_2, \ldots, x_n)$ e $y = (y_1, y_2, \ldots, y_n)$ sono due vettori a n componenti, il prodotto fra il vettore colonna x e il vettore riga y coincide con la matrice di ordine $n \cdot n$ in cui l'elemento di indice ij è dato dal prodotto tra la i-esima componente di x e la j-esima componente di y. In formule:

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} (y_1 \quad y_2 \quad \cdots \quad y_n) = \begin{pmatrix} x_1y_1 & x_1y_2 & \cdots & x_1y_n \\ x_2y_1 & x_2y_2 & \cdots & x_2y_n \\ \vdots & \vdots & \ddots & \vdots \\ x_ny_1 & x_ny_2 & \cdots & x_ny_n \end{pmatrix}$$

Implementazione

- Per il prodotto serve una matrice: utilizziamo il *base indexed addressing*
  - Esempi:

spiazzamento[BX][DI]	*spiazzamento[BX][SI]*
spiazzamento[BP][DI]	*spiazzamento[BP][SI]*
[BX][DI]	*[BX][SI]*
[BP][DI]	*[BP][SI]*

Implementazione [cont.]

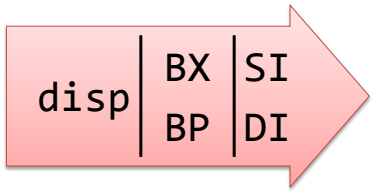
- Memorizziamo la matrice per righe, su WORD
- matrice[x][y] = matrice[BX][SI], con
  - x ∈ [1, NUM_RIG] y ∈ [1, NUM_COL]
  - BX = (x-1)*2*NUM_COL SI = (y-1)*2

Per passare da una riga alla successiva: sommare NUM_BYTE*NUM_COL (NUM_BYTE = 2 se si lavora con word)

Per passare da una colonna alla successiva: sommare NUM_BYTE

Indirizzamento 8086

- La seguente rappresentazione riassume tutti i 17 possibili modi di indirizzamento dell'8086:



- Esempi:
  [BX][SI], [DI], disp, disp[DI], disp[BX][DI]…
- N.B.: quando viene utilizzato [BP], il processore fa accesso allo *stack segment* (*data segment* in tutti gli altri casi).

Esercizio 3

- Si scriva un programma in grado di generare una tavola pitagorica (10x10) e memorizzarla.

Esercizio 4

- Sia data la seguente tabella di *word*:

154	123	109	86	4	?
412	-23	-231	9	50	?
123	-24	12	55	-45	?
?	?	?	?	?	?

- Implementare in Assembly 8086 il programma che scriva la somma di ciascuna riga e colonna rispettivamente nell'ultima colonna e riga.

Esercizio 5

- Un modo per calcolare la radice quadrata approssimata di un numero intero consiste nel contare la quantità di numeri dispari che possono essere sottratti dal numero di partenza. La soluzione proposta in linguaggio C è la seguente:

```
main()
{
 int num, sqr=0, disp=1;
  ...
  num--;
  while (num >= 0)
        { sqr++;
          disp += 2;
          num -= disp; }
  ...
}
```

- Realizzare un programma in Assembly 8086 che calcoli la radice quadrata approssimata di un intero positivo (16 bit).