

Esercizi Assembly 3

M. Sonza Reorda – M. Grosso – M. Monetti

Politecnico di Torino
Dipartimento di Automatica e Informatica

Esercizio 1

- 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 $x = (x_1, x_2, \dots, x_n)$ e $y = (y_1, y_2, \dots, y_n)$ 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:

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} (y_1 \quad y_2 \quad \cdots \quad y_n) = \begin{pmatrix} x_1 y_1 & x_1 y_2 & \cdots & x_1 y_n \\ x_2 y_1 & x_2 y_2 & \cdots & x_2 y_n \\ \vdots & \vdots & \ddots & \vdots \\ x_n y_1 & x_n y_2 & \cdots & x_n y_n \end{pmatrix}$$

Implementazione

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

– Esempi:

spiazzamento[BX][DI]

spiazzamento[BP][DI]

[BX][DI]

[BP][DI]

spiazzamento[BX][SI]

spiazzamento[BP][SI]

[BX][SI]

[BP][SI]

Implementazione [cont.]

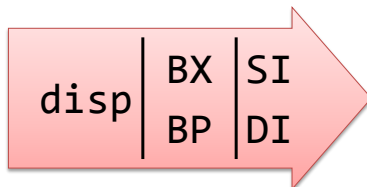
- Memorizziamo la matrice per righe, su WORD
- $\text{matrice}[x][y] = \text{matrice}[\text{BX}][\text{SI}]$, con
 - $x \in [1, \text{NUM_RIG}]$ $y \in [1, \text{NUM_COL}]$
 - $\text{BX} = (x-1) * 2 * \text{NUM_COL}$ $\text{SI} = (y-1) * 2$

Per passare da una riga alla successiva: sommare $\text{NUM_BYTE} * \text{NUM_COL}$ ($\text{NUM_BYTE} = 2$ 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 2

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

Esercizio 3

- Sia data la seguente tabella di *word*:

154	123	109	86	4	?	476
412	-23	-231	9	50	?	217
123	-24	12	55	-45	?	121
?	?	?	?	?	?	
689	76	110	150	9	814	

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

Esercizio 4

- Scrivere un programma in Assembly che sommi i seguenti numeri rappresentati in un vettore di *byte*: -5, -45, -96, -128
- Sommare ancora al risultato il valore di addendo, variabile di tipo *doubleword* con valore 69000
- La somma deve essere salvata nella variabile **risultato** di tipo *doubleword*
 - In fase di debug, porre particolare attenzione al modo in cui sono memorizzate le *doubleword*.

Implementazione

- Le variabili di vettore sono *byte* in CA2
 - Per effettuare correttamente la somma è necessario estendere la rappresentazione su word: CBW (NB: solo per CA2)
- Il risultato parziale su *word* deve essere esteso a *doubleword*: CWD (NB: solo per CA2)
- Attenzione alla somma del *carry* quando necessario: istruzione ADC
- Le variabili di tipo *doubleword* sono memorizzate a partire dal byte meno significativo.