

#01 Testo del problema

Date due word residenti in memoria, depositare in una terza locazione la loro somma.

#02 Analisi del problema

La procedura che conduce al calcolo della somma di due word è del tutto analoga a quella utilizzata per sommare due byte. Si tratterà quindi di prelevare il primo addendo, sommarci il secondo e depositare in un registro di supporto l'eventuale Carry.

Anche in questo caso, in realtà, ci sarebbero sufficienti 16 bit + 1 per l'eventuale Carry, ma per questioni di coerenza stabiliamo di depositare il risultato di due word.

Questo anche per il fatto che, se dovessimo sommare più di due word, sarebbe sempre comodo ragionare in questo modo, piuttosto che aggiungere solamente un byte ogni volta che risulta necessario.

Inoltre, anche solo per un aspetto di coerenza, è più corretto che la somma di due word sia depositata in due word, piuttosto che in una word e un byte.

Essendo il processore 8086 dotato di un data bus a 16 bit, i trasferimenti dei dati e la loro definizione possono essere immediati.

#03 Progettazione delle routine

Nessuna routine necessaria.

#04 Registri utilizzati

Registri a 8 bit	
	Nessun registro a 8 bit utilizzato
Registri a 16 bit	
AX	▪ Primo addendo ▪ Parte bassa della somma
BX	▪ Parte alta della somma

#05 Codice del programma

p02.asm

```
-----  
;  
; Direttive all'assemblatore  
-----  
DOSSEG                ; Segmentazione DOS  
.MODEL                small          ; Modello di memoria  
.STACK                100h          ; Dimensione dello stack  
  
-----  
; Data Segment  
-----  
.DATA  
X    dw    124Ah        ; Primo addendo  
Y    dw    4BD0h        ; Secondo addendo  
S1   dw    ?            ; Parte bassa somma  
Sh   dw    ?            ; Parte alta somma  
  
-----  
; Code segment  
-----  
.CODE  
  
;;;;;;;;;;;;;;;;;;;;;;;;;  
; Programma principale  
;;;;;;;;;;;;;;;;;;;;;;;;;  
Begin:                StartupCode;      ; Inizio del programma  
  
; Inizializzazione variabili  
    MOV    AX , [X]      ; Prelevo il primo operando  
    XOR    BX , BX      ; Azzero il registro per il Carry  
  
; Algoritmo  
    ADD    AX , [Y]      ; Aggiungo il secondo operando...  
    ADC    BX , 0        ; Aggiungo l'eventuale Carry  
  
; Salvataggio risultati  
    MOV    [S1] , AX     ; Salvo la parte bassa...  
    MOV    [Sh] , BX     ; ...e la parte alta della somma  
  
; Operazioni finali  
ExitCode 0            ; Esecuzione corretta del programma  
END Begin            ; Fine del programma
```

#06 Prove effettuate

Prova 01			
Significato della prova	▪ Prova con dati generici pseudo – casuali		
Valori in input	X	=>	124Ah
	Y	=>	4BD0h
Valori attesi in output	S	=>	00005E1Ah
Valori ottenuti in output	S	=>	00005E1Ah

Prova 02			
Significato della prova	▪ Prova al limite inferiore		
Valori in input	X	=>	0000
	Y	=>	0000
Valori attesi in output	S	=>	00000000
Valori ottenuti in output	S	=>	00000000

Prova 03			
Significato della prova	▪ Prova al limite superiore		
Valori in input	X	=>	FFFFh
	Y	=>	FFFFh
Valori attesi in output	S	=>	1FFFEh
Valori ottenuti in output	S	=>	1FFFEh