

Corso di Calcolatori Elettronici I

Alcuni programmi assembly MC68000

Prof. Roberto Canonico



Università degli Studi di Napoli Federico II
Dipartimento di Ingegneria Elettrica
e delle Tecnologie dell'Informazione

Esempio #1 – Somma i primi 17 interi

* Somma i primi 17 numeri interi: $17+16+15+\dots+3+2+1$

```

        ORG      $8000
START    CLR.W    D0
        MOVE.W   N,D1
LOOP     ADD.W    D1,D0
        ADD.W    #-1,D1
        BNE      LOOP
        MOVE.W   D0,SUM
LAST     JMP      LAST
        ORG      $8200
N        DC.W     17
SUM      DS.W     1
        END      START
```

Esempio #2 - Moltiplicazione di due interi

* Programma per moltiplicare MCND e MPY

* attraverso somme successive

	ORG	\$8000	
MULT	CLR.W	D0	D0 accumula il risultato
	MOVE.W	MPY,D1	D1 e' il contatore di ciclo
	BEQ	DONE	Se il contatore e' zero e' finito
LOOP	ADD.W	MCND,D0	Aggiunge MCND al prodotto parziale
	ADD.W	#-1,D1	Decrementa il contatore
	BNE	LOOP	e ripete il giro
DONE	MOVE.W	D0,PROD	Salva il risultato
LAST	JMP	LAST	Ciclo infinito per terminare
	ORG	\$8100	
PROD	DS.W	1	Riserva spazio di memoria per PROD
MPY	DC.W	3	Definisce il valore di MPY
MCND	DC.W	4	Definisce il valore di MCND
	END	MULT	

Esempio #3: prodotto scalare

- Scrivere un programma che esegua il prodotto scalare tra due vettori di interi word A e B
 - A e B allocati staticamente ed inizializzati con DC
 - La dimensione N è nota a priori ed è costante
-

DBcc: Test condition, decrement, and branch

Operazione: IF (cc false) THEN
 $[Dn] \leftarrow [Dn] - 1$
 IF $[Dn] = -1$ THEN $[PC] \leftarrow [PC] + 2$
 ELSE $[PC] \leftarrow [PC] + d$
 ELSE $[PC] \leftarrow [PC] + 2$

Sintassi: DBcc Dn,<label>

Attributi: Size = word

Descrizione:

Fintantoché la condizione *cc* rimane falsa, decrementa il registro *Dn*, e se questo non era zero prima del decremento (ovvero se non vale -1) salta all'istruzione a distanza *d*. Negli altri casi, passa all'istruzione seguente.

Fornisce un modo sintetico per gestire i cicli, sostituendo con un'unica istruzione il decremento di un registro di conteggio e la verifica di una condizione normalmente fatti con istruzioni separate.

Supporta tutti i cc usati in Bcc. Inoltre, ammette anche le forme DBF e DBT (F = false, e T = true) per ignorare la condizione ed usare solo il registro di conteggio.

Esempio #3: prodotto scalare

```
ORG      $8000
START    MOVEA.L  #A,A0      oppure -->  LEA   A,A0
        MOVEA.L  #B,A1      oppure -->  LEA   B,A1
        MOVE.L   #N-1,D0
        CLR      D2
LOOP     MOVE     (A0)+,D1    equivale a:  MOVE.W  (A0)+,D1
        MULS     (A1)+,D1    equivale a:  MULS.W  (A0)+,D1
        ADD      D1,D2       equivale a:  ADD.W   D1,D2
        DBRA     D0,LOOP
        MOVE     D2,C        risultato 10=$000A in C
LAST     JMP      LAST
N        EQU     $000A
        ORG      $80B0
A        DC.W     1,1,1,1,1,1,1,1,1,1
        ORG      $80D0
B        DC.W     1,1,1,1,1,1,1,1,1,1
C        DS.W     1
        END      START
```

Esempio #4 - Ricerca di un token in una stringa di caratteri

- Scrivere un programma che:
 - Riconosca un token (un carattere speciale noto) in una stringa di caratteri
 - La lunghezza della stringa non sia nota
 - La fine della stringa segnalata dal byte zero (come in C/C++)
 - Memorizzi l'indirizzo della prima istanza del token in una locazione di memoria TOKENA
 - Assemblare ed eseguire il programma sul simulatore
-

Esempio #4 - Ricerca di un token in una stringa di caratteri

```

                ORG      $8000
START  MOVEA.L  #STRING,A0 ;equivalente a LEA STRING,A0
        MOVE.B  #TOKEN,D0
LOOP   TST.B    (A0)
        BEQ     DONE
        CMP.B   (A0)+,D0
        BNE     LOOP
FOUND  SUBQ.L   #1,A0
DONE   MOVE.L   A0,TOKENA
LAST   JMP      LAST
        ORG     $8100
STRING DC.B    'QUI QUO:QUA',0
TOKEN  EQU     ':'
TOKENA DS.L    1
        END     START
```

Es. #5: conta non-spazi in stringa (1)

- * Si scriva un programma assembly che conti i caratteri
 - * di una stringa diversi dallo spazio.
 - * Il codice ASCII dello spazio e' 32 in decimale
 - *
 - * Il conteggio deve essere effettuato attraverso un
 - * sottoprogramma CONTA, che riceve l'indirizzo della
 - * Stringa come parametro sullo stack e restituisce
 - * come parametro di ritorno il numero di caratteri
 - * della stringa.
 - * La stringa e' terminata da '\0' come in C/C++
 - *
 - *
 - * ... Continua ...
-

Es. #5: conta non-spazi in stringa (2)

*Inizio area dati del MAIN

ORG \$8000

STR DC.B 'Qui, Quo e Qua',0

SUM DS.W 1

* Inizio area codice del MAIN

ORG \$8400

MAIN ADDA #-2,SP

PEA STR

JSR CONTA

MOVE.W 4(SP),SUM

ADDA #6,SP

LAST JMP LAST

* ... Continua ...

Es. #5: conta non-spazi in stringa (3)

* Subroutine CONTA

```

        ORG      $8800
CONTA    LINK      A6, #0
        MOVEM.L   A0/D0-D2, - (SP)
        CLR.L     D2
        MOVEA.L   8 (A6), A0
CICLO    MOVE.B    (A0)+, D1
        BEQ       FINE
        CMP       #32, D1           oppure: CMP    #' ', D1
        BEQ       CICLO
        ADD       #1, D2
        BRA       CICLO
FINE     MOVE      D2, 12 (A6)
```

* ... Continua ...

Es. #5: conta non-spazi in stringa (4)

```
* ... Continua ...
* Chiusura subroutine CONTA
FINE      MOVE      D2,12(A6)
          MOVEM.L   (SP)+,A0/D0-D2
          UNLK      A6
          RTS
*
* Fine del programma
*
          END       MAIN
```

Es. #6 – Conversione di una stringa in tutte lettere maiuscole (1)

* TOUPPER

* Programma strutturato con una subroutine (TOUP)

```
        ORG      $8000      Inizio area dati del main
STR1 DC.B      'Giorno: 11; Mese: 06; anno: 2014',0
        ORG      $8200      Inizio area codice
MAIN PEA      STR1          Push indirizzo iniziale dell'array
        BSR      TOUP
        ADDA.L   #4,SP      Disalloca da stack parametri ingresso
LAST JMP      LAST
```

* ... Continua ...

Es. #6 – Conversione di una stringa in tutte lettere maiuscole (2)

* ... Continua ...

ORG \$8300

TOUP LINK A6, #0

MOVEM.L A0/D0, - (SP)

MOVEA.L 8(A6), A0

LOOP MOVE.B (A0)+, D0

BEQ DONE

CMP.B #'a', D0

BLT NEXT

CMP.B #'z', D0

BGT NEXT

ADD.B #'A'-'a', D0 equivale a: ADD.B #-32, D0

* ... Continua ...

Es. #6 – Conversione di una stringa in tutte lettere maiuscole (3)

```
* ... Continua ...  
        MOVE.B    D0,-1(A0)  
NEXT    BRA        LOOP  
DONE    MOVEM.L    (SP)+,A0/D0  
        UNLK      A6  
        RTS  
  
*  
*   Fine del programma  
*  
        END        MAIN
```

Es. #7: elabora vettore di int (1)

- * Dato un vettore A di N=10 numeri interi di tipo word,
 - * memorizzato a partire dall'indirizzo \$8000,
 - * scrivere un programma che calcoli,
 - * attraverso un unico ciclo, i seguenti valori:
 - * - la somma X dei soli elementi positivi
 - * - il minimo elemento negativo Y
 - * - il numero di elementi nulli Z
 - *
 - * Ad es. se A = 3,8,-2,0,10,-1,0,-5,1,0
 - * e' il vettore di partenza, al termine della esecuzione,
 - * dovranno risultare:
 - * X = 22, Y = -5, Z = 3
 - * ... Continua ...
-

Es. #7: elabora vettore di int (2)

```
      ORG    $8000
A      DC.W  3,8,-2,0,10,-1,0,-5,1,0
N      EQU   10
X      DS.W  1
Y      DS.W  1
Z      DS.W  1

      ORG    $8100
MAIN   CLR    D0
        CLR    D1
        CLR    D2
        LEA    A,A0
        MOVE   #N-1,D3
* ... Continua ...
```

Es. #7: elabora vettore di int (3)

```
* ... Continua ...  
LOOP  MOVE    (A0)+,D4  
      BLT     NEGAT  
      BEQ     ZERO  
POSIT  ADD     D4,D0  
      BRA     NEXT  
ZERO   ADDQ    #1,D2  
      BRA     NEXT  
NEGAT  CMP     D1,D4  
      BGE     NEXT  
      MOVE    D4,D1  
NEXT   DBRA    D3,LOOP  
* ... Continua ...
```

Es. #7: elabora vettore di int (4)

```
* ... Continua ...  
    MOVE    D0,X  
    MOVE    D1,Y  
    MOVE    D2,Z  
LAST  JMP    LAST  
  
*  
*   Fine del programma  
*  
  
    END     MAIN
```
