

#### Problema 4 Compito 29/5/2002

Sia dato il seguente programma SEC. Nell'istruzione 0002 sia  $K=m_6+1$  ove  $m_6$  rappresenta la 6<sup>a</sup> cifra del numero di matricola. Descrivere sinteticamente scopo e funzionamento del programma e riportare il contenuto delle celle di memoria da 1000 a 1004 al termine dell'esecuzione. Nella seguente tabella la prima colonna indica la locazione di memoria, la seconda colonna la codifica ottale e la terza la codifica mnemonica di un'istruzione.

|      |         |              |
|------|---------|--------------|
| 0000 | 1000001 | ENT, 0, 1    |
| 0001 | 0202000 | STA, 0, 2000 |
| 0002 | 100kkkk | ENT, 0, K    |
| 0003 | 0201000 | STA, 0, 1000 |
| 0004 | 1000005 | ENT, 0, 5    |
| 0005 | 0402000 | SUB, 0, 2000 |
| 0006 | 2200200 | JPA, 0, 0200 |
| 0007 | 0000000 | HLT          |
| 0200 | 0101000 | LDA, 0, 1000 |
| 0201 | 0301000 | ADD, 0, 1000 |
| 0202 | 0201001 | STA, 0, 1001 |
| 0203 | 1000001 | ENT, 0, 1    |
| 0204 | 0302000 | ADD, 0, 0200 |
| 0205 | 0200200 | STA, 0, 0200 |
| 0206 | 1000001 | ENT, 0, 1    |
| 0207 | 0300201 | ADD, 0, 0201 |
| 0210 | 0200201 | STA, 0, 0201 |
| 0211 | 1000001 | ENT, 0, 1    |
| 0212 | 0300202 | ADD, 0, 0202 |
| 0213 | 0200202 | STA, 0, 0202 |
| 0214 | 1000001 | ENT, 0, 1    |
| 0215 | 0302000 | ADD, 0, 2000 |
| 0216 | 0202000 | STA, 0, 2000 |
| 0217 | 2000004 | JMP, 0, 0004 |

#### Soluzione

Per evidenziare la logica il programma viene riscritto con istruzioni che usano simboli anche per gli indirizzi (omettendo la cifra intermedia ottale 0). Si fanno le seguenti assegnazioni di simboli:

**count** = 2000<sub>8</sub>

**tab** = 1000<sub>8</sub>

|       |                                                           |                                            |
|-------|-----------------------------------------------------------|--------------------------------------------|
|       | <b>ENT</b> 1                                              | /Inizializza variabile contatote           |
|       | <b>STA</b> count                                          |                                            |
|       | <b>ENT</b> K                                              | /Memorizza il valore K                     |
|       | <b>STA</b> tab                                            |                                            |
| loop: | <b>ENT</b> 5                                              |                                            |
|       | <b>SUB</b> count                                          | /Salta se count<5                          |
|       | <b>JPA</b> t1                                             |                                            |
|       | <b>HLT</b>                                                | /FINE                                      |
| t1:   | <b>LDA</b> tab +i                                         | /M[tab+1+i]=2*M[tab+i]                     |
| t2:   | <b>ADD</b> tab +i                                         | /con i = 0, 1, 2, 3                        |
| t3:   | <b>STA</b> tab+1 +i                                       |                                            |
|       | /aggiorna i puntatori istruzioni di indirizzi t1, t2, t3. |                                            |
|       | <b>ENT</b> 1                                              |                                            |
|       | <b>ADD</b> t1                                             | /incrementa di uno operando di LDA tab+i   |
|       | <b>STA</b> t1                                             |                                            |
|       | <b>ENT</b> 1                                              |                                            |
|       | <b>ADD</b> t2                                             | /incrementa di uno operando di ADD tab+i   |
|       | <b>STA</b> t2                                             |                                            |
|       | <b>ENT</b> 1                                              |                                            |
|       | <b>ADD</b> t3                                             | /incrementa di uno operando di STA tab+1+i |
|       | <b>STA</b> t3                                             |                                            |
|       | <b>ENT</b> 1                                              |                                            |
|       | <b>ADD</b> count                                          | /count = count + 1                         |
|       | <b>STA</b> count                                          |                                            |
|       | <b>JMP</b> loop                                           |                                            |

Nell'ipotesi che la sesta cifra del numero di matricola sia  $m_6=4$ : si ricava  $K=m_6+1=5$ . Questo valore viene memorizzato nella locazione di indirizzo **tab** (si ricorda che il simbolo **tab** rappresenta l'indirizzo  $1000_8$ ) dalla terza e quarta istruzione. La parte restante del programma, a iniziare dalla istruzione di indirizzo **loop**, è un ciclo, che viene ripetuto tante volte quante sono le unità della variabile **count** (cinque volte in questo caso). Ad ogni iterazione i puntatori **tab** e **tab+1** delle istruzioni di indirizzo **t1**, **t2**, **t3** vengono incrementati di una unità (dalle nove istruzioni poste nel doppio riquadro). Queste tre istruzioni, parte centrale del programma, memorizzano all'indirizzo **tab+1+i** il doppio del valore memorizzato all'indirizzo precedente **tab+i**.

Lo scopo del programma è quindi quello di memorizzare, nella tabella di memoria di indirizzo iniziale **tab**, cinque valori (pari al valore memorizzato nella variabile **count**) in modo che il dato in ogni locazione sia il doppio di quello contenuto nella locazione precedente.

|        |                      |                                    |
|--------|----------------------|------------------------------------|
| tab:   | 0000005 <sub>8</sub> |                                    |
| tab+1: | 0000012 <sub>8</sub> | /primo passo $2 \cdot 0000005_8$   |
| tab+2: | 0000024 <sub>8</sub> | /secondo passo $2 \cdot 0000012_8$ |
| tab+3: | 0000050 <sub>8</sub> | /terzo passo $2 \cdot 0000024_8$   |
| tab+4: | 0000120 <sub>8</sub> | /quarto passo $2 \cdot 0000050_8$  |
|        |                      |                                    |

La logica del programma può essere descritta dal seguente diagramma di flusso

