

Problema SEC Compito 27/6/2002

Sia dato il seguente programma SEC. Nell'istruzione 0000 sia **K** la rappresentazione ottale del numero ottenuto dalla moltiplicazione $2 \cdot m_1 \cdot m_2$, ove m_2 ed m_1 sono le due cifre meno significative del numero di matricola. Alcune istruzioni fanno parte di una ipotetica estensione del set di istruzioni standard del SEC. In particolare:

SHR,0,Y

carica il contenuto della cella di locazione Y, eseguendone shift verso destra, e scrive il risultato nell'accumulatore. (Esempio di shift su 4 bit: $1011 \rightarrow 0101$).

XOR,0,Y

esegue lo XOR, bit a bit, del contenuto dell'accumulatore con il contenuto della cella di locazione Y e scrive il risultato in accumulatore:

Descrivere sinteticamente scopo e funzionamento del programma e riportare (in rappresentazione binaria, ottale e decimale) il contenuto delle celle di memoria da 1000 a 1001 al termine dell'esecuzione.

Nel programma che segue, la prima colonna indica la locazione di memoria, la seconda colonna la codifica ottale e la terza la codifica mnemonica di un'istruzione.

0000	100kkkk	ENT, 0, K
0001	0201000	STA, 0, 1000
0002	0203000	STA, 0, 3000
0003	3003000	SHR, 0, 3000
0004	4003000	XOR, 0, 3000
0005	0201001	STA, 0, 1001
0006	0000000	HLT

Soluzione

Il programma utilizza tre variabili in memoria. La prima di indirizzo 1000_8 contenente il parametro di ingresso del programma, verrà indicata con il simbolo **input**, La seconda di indirizzo 3000_8 contenente una copia dello stesso dato ed utilizzata nelle elaborazioni, verrà indicata con il simbolo **temp**. La terza di indirizzo 1001_8 contenente il risultato, verrà indicata con il simbolo **result**. Si fanno quindi le seguenti assegnazioni di simboli:

input = 1000_8

temp = 3000_8

result = 1001_8

Si fa inoltre l'ipotesi che la parte del numero di matricola che interessa sia: **xxxx94**, per cui

$$K = 2m_2m_1 = 2 \cdot 9 \cdot 4 = 72_{10} = 110_8 = 001001000_2$$

Con queste premesse il programma può essere riscritto:

```

start:  ENT 0110      /Memorizza valore di K = 0...0b11b10...b1b0
        STA input

        STA temp      /Copia del valore K
        SHR temp      /In acc. Shift right K = 00...0b11b10... b2b1
        XOR temp      /In acc. b0= b0⊕b1, b1= b1⊕b2, .....

        STA result    /Memorizza codice Gray di K

        HLT           /FINE

```

Nella scrittura del programma con l'uso dei simboli si è usata l'etichetta **start** (= 0000₈) per indicare l'inizio del programma. Non è stato necessario usare altri simboli in quanto non vi sono istruzioni di salto che fanno riferimento ad indirizzi di istruzioni.

Dopo le prime due istruzioni che inizializzano la locazione **input** con il valore di K, le successive tre istruzioni elaborano la copia di K nella locazione **temp** con la seguente logica:

temp	0 0 0 0 0 0 0 0 0 0										b ₁₁	b ₁₀	b ₉	b ₈	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	b ₀	
accum.	0 0 0 0 0 0 0 0 0 0										0	b ₁₁	b ₁₀	b ₉	b ₈	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	SHR
accum	0 0 0 0 0 0 0 0 0 0										g ₁₁	g ₁₀	g ₉	g ₈	g ₇	g ₆	g ₅	g ₄	g ₃	g ₂	g ₁	g ₀	XOR

Appare evidente che in accumulatore, dopo le due operazioni di SHR e XOR, vi è il codice Gray del valore binario K, caricato inizialmente in accumulatore ($g_i = b_i \oplus b_{i+1}$).

La successiva istruzione memorizza questo valore in **result**.

Lo scopo del programma è quindi quello di memorizzare il valore di K, in forma binaria, nella locazione **input** (1000₈) ed in codice Gray, nella locazione **result** (1001₈).

Nel caso esemplificato $M[\text{input}] = 72_{10} = 0000110_8 = 0000000000000001001000_2$ si ha:

temp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	
accum.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	SHR
accum	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	XOR

$$M[\text{result}] = 0000000000000001101100_2 = 0000154_8 = 108_{10}$$