

## IO.asm – Analisi simulazione

a cura di Daniele Sana

Questo programma permette di trasformare una lettera da maiuscola a minuscola.

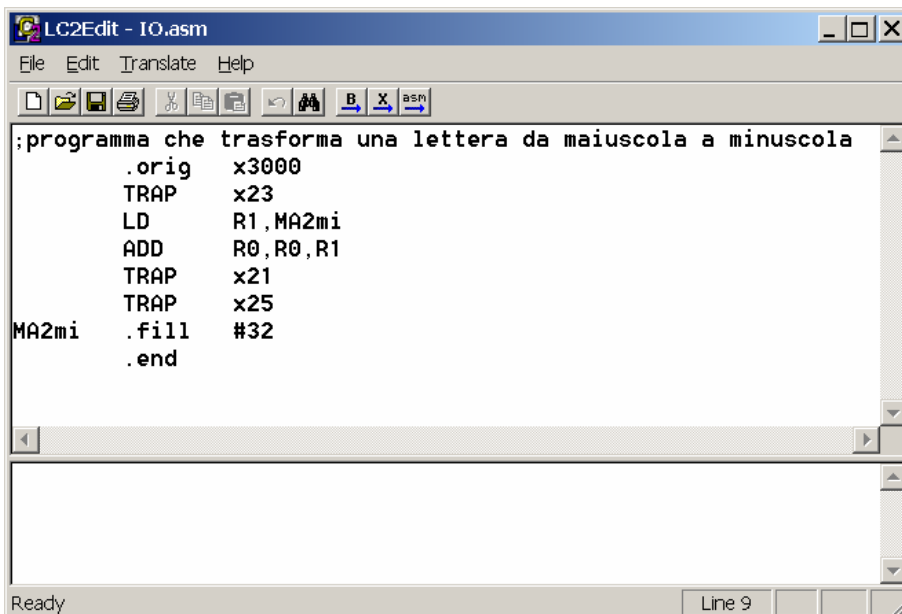
Il risultato viene poi visualizzato a video mediante un'istruzione di **TRAP x21** (o **TRAP OUT**).

Ricordiamo che in ASCII le lettere dell'alfabeto vengono così codificate:

- le lettere maiuscole da 65 a 90 ('A'=65, 'B'=66, ... , 'Z' = 90)
- le lettere minuscole da 97 a 122 ('a'=97, 'b'=98, ... , 'z'=122)

Dalla codifica ASCII è pertanto evidente che, data una lettera maiuscola, la corrispondente minuscola si ottiene semplicemente sommando 32 alla codifica della lettera maiuscola.

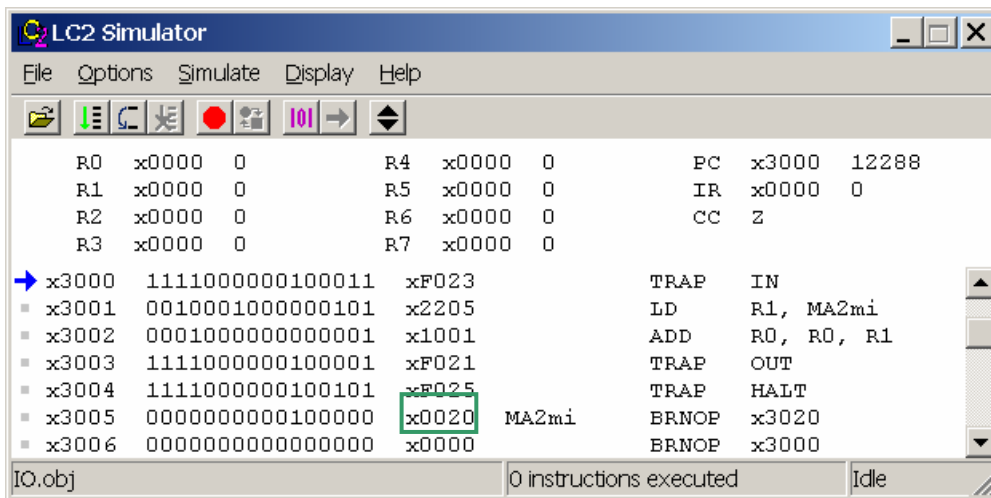
Il testo del programma, così come appare nell'*editor*, è il seguente:



```
;programma che trasforma una lettera da maiuscola a minuscola
      .orig      x3000
      TRAP      x23
      LD        R1,MA2mi
      ADD       R0,R0,R1
      TRAP      x21
      TRAP      x25
MA2mi .fill     #32
      .end
```

A questo punto, dopo aver compilato il programma, passiamo allo strumento *Simulate* ed effettuiamo il debugging, ossia eseguiamo il programma un'istruzione alla volta.

In Simulate la situazione è la seguente:



Vediamo innanzitutto quali sono stati gli effetti delle pseudo-istruzioni *Assembly* presenti nel codice:

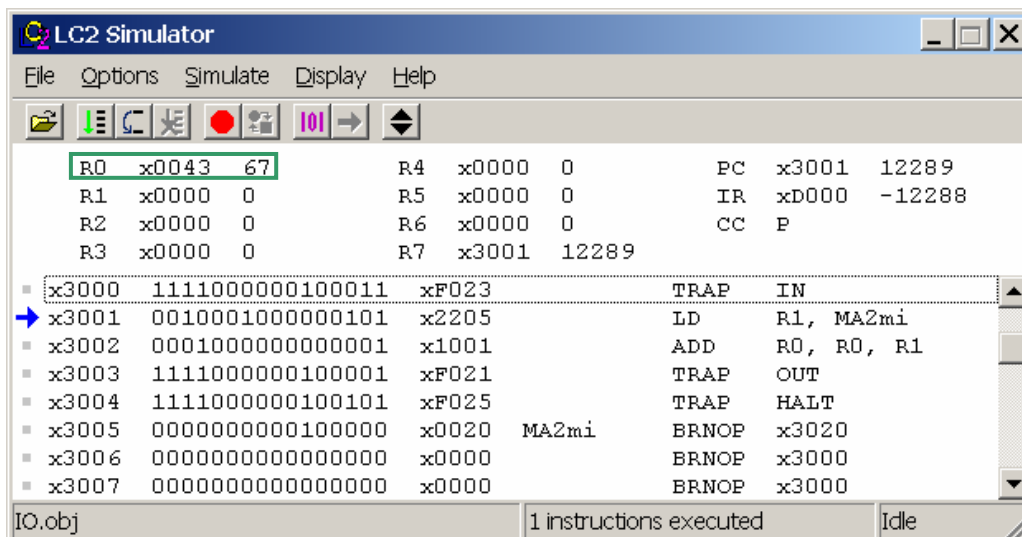
- **.orig x3000** ha segnalato all'*Assembler* che il programma deve essere caricato all'indirizzo x3000;
- **.fill #32** inizializza la cella di indirizzo x3000c al valore costante 32 (rettangolo verde);

Inoltre, la tabella dei simboli generata dall'*Assembler* è la seguente:

```
// Symbol table
// Scope level 0:
//      Symbol Name      Page Address
//      -----
//      MA2mi             3005
```

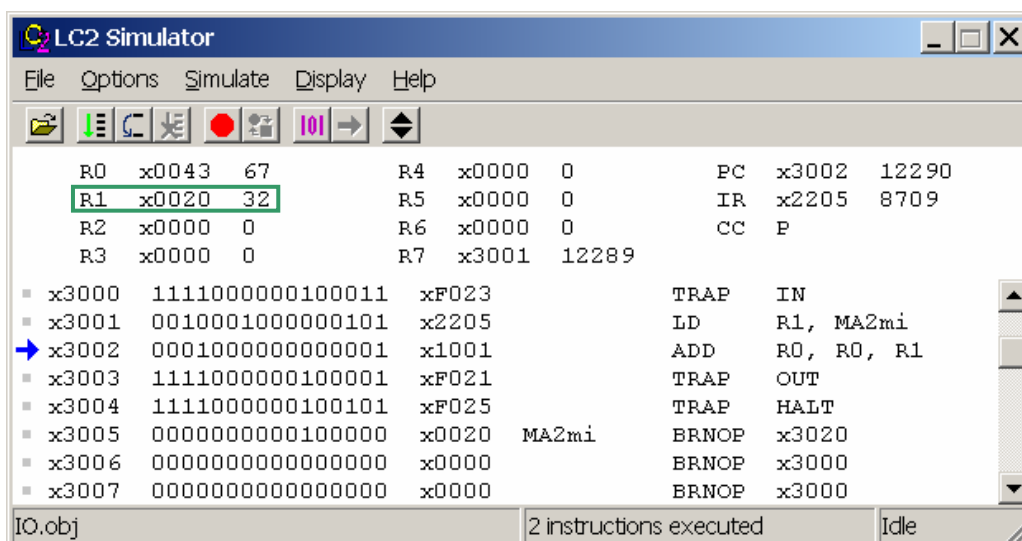
Facciamo ora partire il programma con il comando *Simulate* → *Step (F8)* e analizziamo ogni singola istruzione.

Eseguendo l'istruzione **TRAP IN** si può notare che la *Console* è cambiata, dal momento che ora chiede l'immissione di un carattere da tastiera. Pertanto il programma non potrà proseguire fino a che tale valore verrà letto da tastiera (e salvato in R0, secondo quanto previsto da un'istruzione TRAP). Inseriamo, a titolo di esempio, il carattere maiuscolo C e premiamo Invio.

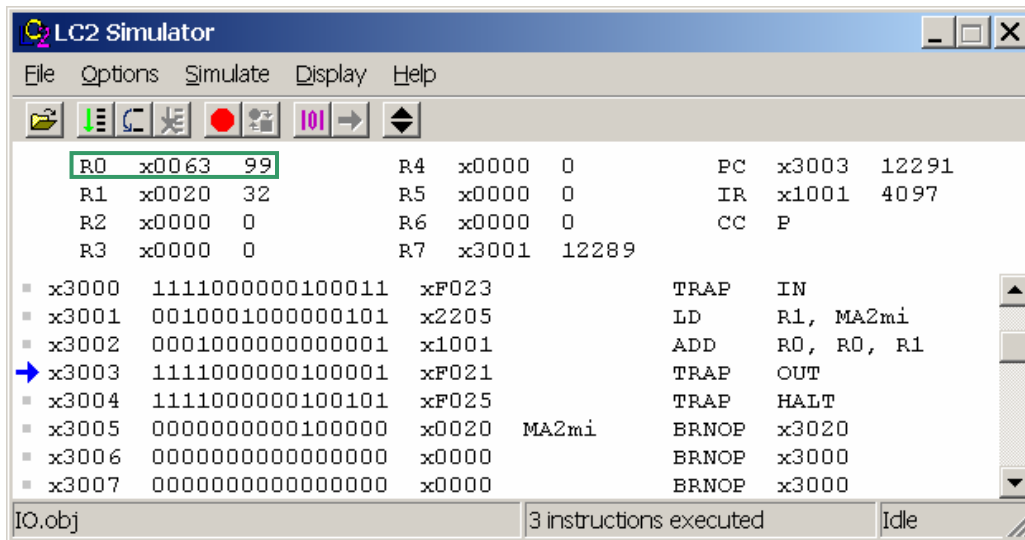


Come si può facilmente notare il registro R0 contiene ora la codifica ASCII del carattere 'C' (rettangolo verde).

A questo punto viene eseguita l'istruzione **LD R1, MA2mi**, che ha l'effetto di caricare nel registro R1 la costante numerica #32 posizionata all'indirizzo identificato dalla label **MA2mi** (x3005).



L'istruzione ADD R0, R0, R1 effettua la conversione da maiuscolo a minuscolo, come si può notare dal risultato contenuto nel registro R0.



Infine:

- **TRAP OUT** visualizza a video sulla *Console* il risultato della conversione, prelevato dal registro R0;
- **TRAP HALT** arresta l'esecuzione del programma.

Si noti che il programma qui presentato non effettua nessun controllo sul carattere ricevuto da tastiera, presupponendo che si tratti di una lettera maiuscola. Se dunque l'operatore preme un altro tasto, l'effetto della conversione è privo di significato.

Una possibile evoluzione – lasciata allo studente – consiste appunto nel controllare che il carattere ricevuto sia effettivamente una lettera maiuscola, e in caso contrario segnalare l'errore.

Oppure il programma potrebbe convertire le lettere maiuscole in minuscole, le minuscole in maiuscole, e lasciare inalterati gli altri caratteri.