**RELAZIONE DI LABORATORIO DI SISTEMI**

Titolo dell’esercitazione

Prodotto di 2 numeri binari

**Obiettivo:** Calcolo del prodotto di 2 numeri binari a 8 bit utilizzando il simulatore Z80

**Studio del Sistema:** Per poter effettuare il calcolo del prodotto di 2 numeri binari a 8 bit dobbiamo porre, il moltiplicando e il moltiplicatore in 2 registri separati e il risultato in un unico registro.

Ci viene in aiuto il programma delle somme successive, per l’idea dello sviluppo del nostro ma non soddisfa i nostri requisiti in fatto di qualità perché se utilizzassimo il massimo numero binario, 255, dovrebbe svolgere (facendo 255\*255) 255 moltiplicazioni e questo per noi sarebbe molto svantaggioso.

L’algoritmo del nostro programma è il seguente schematizzato:

**Simulazione del sistema:** Il programma che ci fa eseguire il prodotto fra 2 numeri binari a 8 bit è, quindi, il seguente:

|  |  |
| --- | --- |
| **Istruzioni** | **Clock** |
|  LD A,LOC. | 13 |
|  LD C,A | 9 |
|  LD A,LOC | 13 |
|  LD E,A | 9 |
|  LD B,08H | 7 |
|  LD D,00H | 7 |
|  LD HL,0000H | 10 |
| LOOP3 SRL C | 8 |
|  JR C,LOOP1 | 7/12 |
|  SLA E | 8 |
|  RL D | 4 |
| LOOP1 JR NC,LOOP2 | 7/12 |
|  ADD HL,DE | 11 |
|  SLA E | 8 |
|  RL D | 4 |
| LOOP2 DEC B | 4 |
|  JR NZ,LOOP3 | 7/12 |
|  LD HL,LOC. | 20 |
|  HALT | 4 |

Carica A 🡨 Locazione

Carica A 🡨 Locazione

Carica E 🡨 A

Carica C 🡨 A

Carica B 🡨 Mol.re

Carica D 🡨 Mol.do

Carica HL 🡨 Risultato

Shift destro di C

Carry

=1?

Ruota a sinistra D

Shift sinistro di E

Il programma elencato qui sopra svolge le seguenti funzioni:

* Carica nel registro A la locazione di memoria;
* Carica nel registro C il moltiplicatore;
* Carica nel registro A l’altra locazione;
* Carica in E il moltiplicando;
* Carica in B gli 8 bit per decrementare;
* Carica in D 0 per azzerare i registri;
* Carica in HL il risultato;
* Esegue lo shift sinistro di C;
* Salta se carry al loop1;
* Esegue lo shift sinistro di E;
* Ruota a sinistra D;

* Salta se non carry al loop2;
* Aggiunge DE ad HL;
* Esegue lo shift a sinistra di E;
* Ruota a sinistra D;
* Decrementa B;
* Salta se non zero al loop 3;
* Carica in HL la locazione di memoria.

Adesso verifichiamo se il nostro sistema funziona:

* Per fare la moltiplicazione tra 0 e 0: inseriamo 00H nella locazione 0110 e 00H in 0112. Il risultato ottenuto è 00H anche in qualsiasi locazione.
* Per fare la moltiplicazione tra 1 e 255: inseriamo 01H in 0110 e in 0112. Il risultato in 0123 è FFH;
* Per fare la moltiplicazione tra 255 e 255: inseriamo FFH in 0110 e in 0112. Il risultato in 0123 è 01H e in 0124 è FEH, che in esadecimale;

Il nostro programma funziona correttamente e anche al contrario cioè 255\*1 ecc…

Occupazione di memoria: 37 byte

Numero di clock Min: 68+39\*8+16\*7+11=503

Numero di clock Max: 68+20\*8+46\*7+41=591

Carry C

=0?

ADD HL 🡨 DE

Ruota a sinistra D

Shift sinistro di E

DEC B

Flag 0

=0?

Carica HL 🡨 Locazione

**Conclusioni:** Possiamo dire di aver concluso questa nostra esperienza perché abbiamo dimostrato che il programma funziona correttamente, grazie alla istruzione del salto relativo, in qualunque locazione di memoria che ad esso assegniamo.