RELAZIONE DI SISTEMI

**TITOLO DELL’ESERCITAZIONE**

CONVERSIONE DA BINARIO A 8 BIT A BCD 8421 IN ASSEMBLY

**Obiettivo:**

Creazione e simulazione di un programma in grado di convertire un numero binario a 8 bit a BCD 8421.

LD A 🡨 n

LD C 🡨 0

INC C

LD E 🡨 0

SUB 100

DEC C

LD L 🡨 0

Carry=0?

ADD A🡨100

SUB 10

INC E

Carry=0?

DEC E

ADD A🡨10

LD L🡨A

HALT

LOOP 1

LOOP 2

NO

NO

SI

SI

**Studio del Sistema:**

Lo scopo dell’esercitazione è quello convertire un numero binario a 8 bit in BCD 8421.

Partiamo innanzitutto dallo studio dell’algoritmo attraverso il diagramma di flusso a destra:

**Spiegazione delle istruzioni:**

Nella 1° istruzione si carica nel registro A un numero binario a 8 bit che può andare da 0 a 255.

Nella 2°, 3° e 4° istruzione si azzeriamo i registri C, E e L.

Nella 5° istruzione si sottrae 100 e nella 6° si incrementa il registro C.

Nella 7° istruzione (un Loop) si verifica se il Carry è 0 se non lo è ritorna ad eseguire la 5° istruzione altrimenti esegue la prossima.

Nella 8° istruzione si decrementa il registro C.

Nella 9° istruzione si somma 100 al registro A.

Nella 10° istruzione si sottrae 10 e nella 11° si incrementa il registro E.

Nella 12° istruzione (un Loop) si verifica se il Carry è 0 se non lo è ritorna ad eseguire la 10° istruzione altrimenti esegue la prossima.

Nella 13° istruzione si decrementa il registro E.

Nella 14° istruzione si somma 10 al registro A.

Nella 15° istruzione si carica in L il contenuto del registro A.

Nella 16° istruzione si termina il programma.

**Reale simulazione del sistema:**

|  |  |  |
| --- | --- | --- |
| N° | Istruzione | ClockIn questo programma assembly si carica un numero binario nell’accumulatore e abbiamo poi utilizzato i registri: C per le centinaia, E per le decine ed L per le unità.Utilizziamo le istruzioni di salto (JP NC) per sostituire la IF, in quanto in questo linguaggio di basso livello (Assembly) non esiste questo tipo di istruzione.Il funzionamento è semplice: in questo caso se il carry è minore del numero va avanti decrementando le centinaia, altrimenti se è minore ritorna all’istruzione della SUB 100.Per il calcolo dei clock minimi basti sommare il numero di clock di ogni istruzione sapendo che n è 0.La stessa cosa vale per i clock massimi basti impostare n a 255 e sommare quante volte viene ripetuta ogni singola istruzione.**Occupazione in memoria:** 28byte.**Clock MIN:** 97.**Clock MAX:** 223. |
| 1 | LD A🡨n | 7 |
| 2 | LD C🡨0 | 7 |
| 3 | LD E🡨0 | 7 |
| 4 | LD L🡨0 | 7 |
| 5 | SUB 100 (LOOP1) | 4 |
| 6 | INC C | 4 |
| 7 | JP NC,LOOP1 | 10 |
| 8 | DEC C | 4 |
| 9 | ADD A,100 | 7 |
| 10 | SUB 10 (LOOP2) | 4 |
| 11 | INC E | 4 |
| 12 | JP NC,LOOP2 | 10 |
| 13 | DEC E | 4 |
| 14 | ADD A,10 | 7 |
| 15 | LD L,A | 7 |
| 16 | HALT | 4 |



**Interpretazione e commento dei risultati ottenuti e del loro significato:**

Possiamo concludere con l’esperienza riuscita in quanto il nostro programma è completato avendo verificato il suo funzionamento raggiungendo l’obiettivo prefissato.