2η Σειρά Ασκήσεων
Συστήματα Μικροϋπολογιστών

Βερέμης Αλέξανδρος 03115063

Σπυρίδων Κρητικός 03116133

Άσκηση 1

Το πρόγραμμα περιμένει να συμβεί διακοπή τύπου RST 6.5, και μόλις αυτή πραγματοποιηθεί ανάβει όλα τα LED εξόδου. Τα LED παραμένουν αναμμένα για 60 sec, ενώ τα δύο δεξιότερα 7-segments λειτουργούν ως χρονόμετρο που μετρά τα δευτερόλεπτα που τα LED παραμένουν αναμμένα. Αν πατηθεί και δεύτερη διακοπή πριν τη συμπλήρωση 60 sec, ο χρόνος στο χρονόμετρο ανανεώνεται και τα LED μένουν αναμμένα για άλλα 60sec.

Γίνεται απευθείας έλεγχος της οθόνης 7 τμημάτων μέσω της πόρτας σάρωσης και της πόρτας τμήματος, χωρίς δηλαδή τη χρήση των ρουτινών DCD και STDM. Παρακάτω φαίνεται ο κώδικας του προγράμματος:

```
EXERCISE_1:

START:
      IN 10H
      LXI H,0A02H ;αποθηκεύουμε τον αριθμό
      MVI M,10H   ;10H στις θέσεις μνήμης που
      INX H      ;αντιστοιχούν στα 4 πιο
      MVI M,10H   ;αριστερά ψηφία, ώστε να
      INX H      ;μην εμφανίζεται τίποτα στα
      MVI M,10H   ;συγκεκριμένα 7-segments
      INX H
      MVI M,10H
      LXI B,03E8H ;καθυστέρηση 1 sec=1000 msec με τη ρουτίνα DELB
      MVI A,0DH   ;στον A θέτουμε τον 00001101b για να
      SIM         ;επιτρέψουμε τη διακοπή RST 6.5
      EI         ;ενεργοποίηση διακοπών
WAIT:
      MVI A,FFH
      STA 3000H
      JMP WAIT    ;περιμένουμε μέχρι να συμβεί διακοπή

INTR_ROUTINE:          ;ρουτίνα εξυπηρέτησης της διακοπής RST 6.5
      MVI E,00H
      MOV A,E     ;το 0 διατηρείται στον E
      STA 3000H   ;άναμμα των LEDs μόλις γίνει διακοπή
      LXI H,0A00H ;θέτουμε και στα δύο ψηφία του δεκαδικού
      MVI M,C0H   ;χρονομέτρου τον αριθμό 0
      INX H
      MVI M,C0H
      CALL SHOW
      EI
```

```
COUNTER:
      INR E
      MOV A,E
      LXI H,0A00H
      CPI 01H     ;αν ο Α είναι 1, μεταβαίνουμε στην IF_1
      JZ IF_1     ;αλλιώς ελέγχουμε όλες τις πιθανές τιμές του
      CPI 02H     ;Α από 2 ως 10 και κάθε φορά μεταβαίνουμε στη
      JZ IF_2     ;συνάρτηση που ενημερώνει τα 7-segments του
      CPI 03H     ;χρονομέτρου
      JZ IF_3
      CPI 04H
      JZ IF_4
      CPI 05H
      JZ IF_5
      CPI 06H
      JZ IF_6
      CPI 07H
      JZ IF_7
      CPI 08H
      JZ IF_8
      CPI 09H
      JZ IF_9

CHNG_MSB:
      LXI H,0A00H
      MVI M,C0H
      MVI E,00H
      INX H
      MOV A,M     ;ο Α χρησιμοποιείται ως μετρητής για το LSB
      CPI C0H     ;του χρονομέτρου. Μόλις γίνει 10 ελέγχουμε το MSB
      JZ IF_1     ;που είναι αποθηκευμένο στην θέση 0A01H. Αν αυτό
      CPI F9H     ;είναι 5, δηλαδή αν έχουμε ξεπεράσει τα 59 sec,
      JZ IF_2     ;μεταβαίνουμε στην END_60_SEC, αλλιώς αυξάνουμε το
      CPI A4H    ;MSBτου χρονομέτρου, το αποθηκεύουμε στη θέση 0A01H
      JZ IF_3     ;και ενημερώνουμε το δεύτερο 7-segment από δεξιά
      CPI B0H
      JZ IF_4
      CPI 99H
      JZ IF_5

END_60_SEC:       ;μόλις φτάσουμε στο 60ο sec εμφανίζεται ο αριθμός
      MVI M,82H   ;60 και η ρουτίνα εξυπηρέτησης διακοπής επιστρέφει
      CALL SHOW   ;τον έλεγχο στο κυρίως πρόγραμμα
      RET

IF_1:
      MVI M,F9H   ;τοποθετεί στη θέση μνήμης που περιέχεται στους
      CALL SHOW   ;H-L τον κατάλληλο αριθμό για την εμφάνιση του
      JMP COUNTER ;1 στο αντίστοιχο 7-segment, και καλεί την SHOW
IF_2:
      MVI M,A4H   ;όμοια λειτουργούν και οι άλλες συναρτήσεις για
      CALL SHOW   ;τα υπόλοιπα ψηφία
      JMP COUNTER
IF_3:
      MVI M,B0H
      CALL SHOW
      JMP COUNTER
```

```
IF_4:
      MVI M,99H
      CALL SHOW
      JMP COUNTER
IF_5:
      MVI M,92H
      CALL SHOW
      JMP COUNTER
IF_6:
      MVI M,82H
      CALL SHOW
      JMP COUNTER
IF_7:
      MVI M,F8H
      CALL SHOW
      JMP COUNTER
IF_8:
      MVI M,80H
      CALL SHOW
      JMP COUNTER
IF_9:
      MVI M,90H
      CALL SHOW
      JMP COUNTER
SHOW:
      LXI H,0A01H ;στη διεύθυνση 0A01H
      MVI D,02H   ;περιέχεται ο αριθμός
      CALL DISP   ;για την απεικόνιση του
      DCX H       ;αριστερότερου ψηφίου, ενώ
      MVI D,01H   ;στη διεύθυνση 0A00H ο
      CALL DISP   ;αριθμός για την απεικόνιση
      CALL DELB   ;του δεξιότερου ψηφίου
      CALL CLEAN_DIGIT
      RET

DISP:
      MOV A,D
      STA 2800H
      MOV A,M
      STA 3800H
      RET

CLEAN_DIGIT:
      MVI A,FFH   ;σβήνονται όλα τα τμήματα
      STA 3800H   ;για να μην απεικονίζονται
      RET   ;προηγούμενα δεδομένα
      END
```

Άσκηση 2

Μόλις γίνει διακοπή RST 6.5 το πρόγραμμα διαβάζει από το πληκτρολόγιο τα δύο ψηφία ενός δεκαεξαδικού αριθμού και τα εμφανίζει στα δύο δεξιότερα 7-segments. Παράλληλα, συγκρίνει τον αριθμό αυτόν με δύο αριθμούς που περιέχονται στους καταχωρητές B και C, έστω K_1 και K_2 αντίστοιχα με $K_1<K_2$. Αν ο αριθμός είναι μικρότερος του K_1, ανάβει το δεξιότερο LED, αν ο αριθμός βρίσκεται ανάμεσα στα K_1 και K_2 ανάβει το δεύτερο δεξιότερο LED, ενώ αν ο αριθμός είναι μεγαλύτερος του K_2 ανάβει το τρίτο LEDαπό δεξιά. Χρησιμοποιήθηκε η ρουτίνα KIND για ανάγνωση από το πληκτρολόγιο, καθώς και οι ρουτίνες DCD και STDM για έλεγχο των 7-segments:

```
EXERCISE_2:

START:
      IN 10H
      LXI H,0A02H ;οπως στην άσκηση 1
      MVI M,10H
      INX H
      MVI M,10H
      INX H
      MVI M,10H
      INX H
      MVI M,10H
      MVI A,0DH
      SIM
      EI
WAIT:
      JMP WAIT

INTR_ROUTINE:
      EI
      CALL KIND  ;περιμένει το πάτημα ενός αριθμού
      ANI 0FH    ;παίρνει τα 4 LSB και τα τοποθετεί
      LXI H,0A00H      ;στη θέση 0A00H, που αντιστοιχεί
      MOV M,A    ;στο δεξιότερο 7-segment
      INX H
      CALL KIND  ;περιμένει το πάτημα ενός δεύτερου αριθμού
      ANI 0FH    ;παίρνει τα 4 LSB και τα τοποθετεί
      MOV M,A    ;στη θέση 0A01H, που αντιστοιχεί
      LXI D,0A00H      ;στο δεύτερο 7-segment από δεξιά
      CALL STDM  ;εμφάνιση του διψήφιου δεκαεξαδικού
      CALL DCD   ;αριθμού
      LXI H,0A00H
      MOV D,M    ;τα δύο ψηφία που βρίσκονται στις
      INX H      ;θέσεις 0A00H και 0A01H ενώνονται σε
      MOV A,M    ;έναν αριθμό που αποθηκεύεται στον A
      RRC        ;με την εντολή ORA D
      RRC
      RRC
      RRC
      ORA D
      MVI B,55H  ;επιλέγουμε ως κατώφλια τους αριθμούς
```

```
      CMP B       ;K1=55H=85D και K2=AAH=170D
      JC LESS_B   ;αν ο αριθμός που είναι αποθηκευμένος
      JZ EQUAL_B ;στον A είναι <= 55H, ανάβει το LSB,
      MVI C,AAH   ;αν είναι > 55H και <= AAH, ανάβει το
      CMP C       ;δεύτερο LSB ενώ αν είναι > AAH ανάβει
      JC LESS_C   ;το τρίτο LSB
      JZ EQUAL_C
      MVI A,04H
HERE:
      CMA
      STA 3000H
      JMP FIN
LESS_B:
      MVI A,01H
      JMP HERE
EQUAL_B:
      MVI A,01H
      JMP HERE
LESS_C:
      MVI A,02H
      JMP HERE
EQUAL_C:
      MVI A,02H
      JMP HERE
FIN:
      EI
      RET
      END
```

Άσκηση 3

α) Μας ζητείται να γράψουμε μια μακροεντολή SWAP Q, R που να εναλλάσει τα περιεχόμενα οποιωνδήποτε καταχωρητών γενικού σκοπού B, C, D, E, H και L. Η εκτέλεση της μακροεντολής δεν πρέπει να επηρεάζει τα περιεχόμενα των καταχωρητών που δεν μετέχουν στην εναλλαγή. Ο κώδικας της μακροεντολής αυτής δίνεται παρακάτω:

```
SWAP MACRO Q,R

      PUSH Q

      PUSH R

      POP Q

      POP R

ENDM
```

β)

```
FILL MACRO ADDR, LEN, K

      PUSH H

      PUSH L

      PUSH B

      LXI H,ADDR

      MVI B,LEN

LP:

      MVI M,K

      INX H

      DCR B

      JNZ LP

      POP B

      POP L

      POP H
ENDM
```

γ) Μας ζητείται να γράψουμε μια μακροεντολή RHLL n που να περιστρέφει τα περιεχόμενα του κρατουμένου CY των καταχωρητών H και L κατά n ψηφία αριστερά. Θεωρούμε ότι έχουμε 17- bit καταχωρητή με CY στο MSB.
Ο κώδικας της μακροεντολής αυτής δίνεται παρακάτω:

```
RHLL MACRO N
      PUSH C
ROTATE_LOOP:
      MOV A,L
      RAL
      MOV L,A
      MOV A,H
      RAL
      MOV H,A
      DCR C
      JNZ ROTATE_LOOP
FILL_FINISHED:
      POP C
ENDM
```

Άσκηση 4

Η εντολή **CALL 3000H** βρίσκεται στη διεύθυνση 2000H, η οποία περιέχεται στον PC. Με την εκτέλεση της **CALL,** η διεύθυνση της επόμενης εντολής του προγράμματος, δηλαδή η 2003H αποθηκεύεται στη στοίβα, ώστε να επιστρέψει σε αυτήν ο έλεγχος μετά το RET της ρουτίνας. Επίσης, ο PC παίρνει τη διεύθυνση 3000H. Άρα, αφού SP=4000H, έχουμε:

CALL 3000H

(3FFFH) ← 20H
(3FFEH) ← 03H
(SP) ← 3FFEH
(PC) ← 3000H

Η διεύθυνση μνήμης που αντιστοιχεί στην διακοπή RST 5.5 είναι η 002CH. Μόλις αναγνωριστεί η RST 5.5 εκτελείται κύκλος μηχανής για τη διακοπή αυτή, ο οποίος είναι ο άεργος κύκλος.Στη διάρκεια αυτού του κύκλου προετοιμάζεται για εκτέλεση η εσωτερική RST, η οποία έχει τα εξής αποτελέσματα:

(3FFDH) ← 30H

(3FFCH) ← 00H

(SP) ← 3FFCH

(PC) ← 002CH, δηλαδή μόλις αναγνωριστεί η διακοπή, η τιμή του PC αποθηκεύεται στη στοίβα, ο δείκτης στοίβας SP μειώνεται κατά δύο ώστε να δείχνει τη θέση της στοίβας όπου αποθηκεύτηκε το τελευταίο στοιχείο, ενώ στον PC καταχωρείται η διεύθυνση της διακοπής. Έτσι ο έλεγχος του προγράμματος μεταφέρεται στην διεύθυνση διακοπής και από εκεί με μια εντολή άλματος στη ρουτίνα εξυπηρέτησης διακοπής.

Τα περιεχόμενα της στοίβας φαίνονται παρακάτω:

3FFCH	00H	← SP
3FFDH	30H	
3FFEH	03H	
3FFFH	20H	

Προφανώς αφού εκτελεστεί η ρουτίνα εξυπηρέτησης πρέπει ο μΕ να επανέλθει στην προηγούμενη κατάστασή του. Αυτό γίνεται με την εξαγωγή του αριθμού 3000H από τη στοίβα και την αποθήκευση του στον PC. Επίσης, ενημερώνεται και ο δείκτης στοίβας, δηλαδή (SP)←3FFEH.

Άσκηση 5

Στο παρακάτω πρόγραμμα, η ρουτίνα εξυπηρέτησης της διακοπής διαβάζει πρώτα τα 4 LSB του δεδομένου, ύστερα διαβάζει τα 4 MSB, καταχωρεί το δεδομένο στον A και το προσθέτει στον αθροιστή, ενώ το κύριο πρόγραμμα, όταν ολοκληρωθεί η ανάγνωση των 32 δεδομένων υπολογίζει τον μέσο όρο τους. Ως συσσωρευτής χρησιμοποιείται το ζεύγος καταχωρητών H-L. Ο καταχωρητής C χρησιμοποιείται ως μετρητής για τα 64 βήματα που χρειάζονται για την αποστολή των 32 δεδομένων. Κάθε δεδομένο που διαβάζεται από τη ρουτίνα εξυπηρέτησης τοποθετείται ως αριθμός των 2 byte, με το πιο σημαντικό byte μηδενισμένο, στο ζεύγος καταχωρητών D-E και στη συνέχεια προστίθεται στο ζεύγος H-L.

Για να υπολογίσουμε τον μέσο όρο, διαιρούμε τον διπλό καταχωρητή H-L με το $32=2^5$. Επειδή ο διαιρέτης είναι δύναμη του 2, το πηλίκο μπορεί να υπολογιστεί με δεξιά ολίσθηση κατά 5 θέσεις. Το ίδιο μπορεί να επιτευχθεί και με αριστερή ολίσθηση κατά 3 θέσεις, με τη διαφορά ότι ο μέσος όρος αντί να βρεθεί στον καταχωρητή L μεταφέρεται στον καταχωρητή H. Μετά την ολίσθηση, ο H περιέχει το ακέραιο μέρος του αριθμού και ο L το κλασματικό μέρος, εκφρασμένο ως πολλαπλάσιο του 1/256:

```
EXERCISE_5:

      MVI A,0EH  ;αρχικοποίηση της μάσκας διακοπών
      SIM        ;και του μετρητή δεδομένων C στο 64
      LXI H,0000H     ;μηδενισμός συσσωρευτή
      MVI C,40H
      MVI E,00H
      EI         ;ενεργοποίηση διακοπών
ADDR:
      MOV A,C    ;βρόχος αναμονής μέχρι να διαβαστούν
      CPI 00H    ;όλα τα δεδομένα, οπότε ο μετρητής
      JNZ ADDR   ;C θα μηδενιστεί
      DI         ;αφού διαβαστούν και τα 32 δεδομένα,
                 ;δηλαδή μόλις γίνουν και οι 64
                 ;διακοπές, οι διακοπές απενεργοποιούνται

      DAD H      ;υπολογίζουμε τον μέσο όρο με ολίσθηση
      DAD H      ;τριών θέσεων προς τα αριστερά του
      DAD H      ;ζεύγους H-L. Αυτό επιτυγχάνεται με την
      HLT        ;πρόσθεση του H-L δύο φορές στον εαυτό του


RST 5.5:
      PUSH PSW
      IN PORT_IN ;τα 4 bits μεταφέρονται από την πόρτα εισόδου
      MOV B,A    ;στον A
      MVI A,00H
      CMP E      ;αν ο E είναι 0, στον A,B περιέχονται τα 4
      JNZ MSBITS ;LSB του δεδομένου. Τότε μηδενίζεται ο
      MVI D,00H  ;D και τα 4 LSB μεταφέρονται στον E
      MOV E,B
      JMP FIN
```

```
MSBITS:
      MOV A ,B    ;αν ο Ε δεν είναι 0, στον Α περιέχονται
      RLC         ;τα 4 MSB του δεδομένου. Με 4 αριστερές
      RLC         ;ολισθήσεις τα μεταφέρουμε στις θέσεις
      RLC         ;των 4 MSB του Α και μέσω της ORA
      RLC         ;μεταβιβάζονται και αυτά στον Ε.
      ORA E
      DAD D       ;πρόσθεση δεδομένων στον H-L
                  ;Τώρα στο ζεύγος D-E περιέχεται το δεδομένο
      MVI E,00H   ;στα 8 LSB, ενώ τα 8 MSB είναι 0
FIN:
      DCR C       ;ελάττωση του μετρητή
      POP PSW
      EI
      RET
```