Συστήματα Μικροϋπολογιστών 3η Σειρά Ασκήσεων 2021-2022

Γιώργος Μυτιληναίος ΑΜ: 03119841

Κωνσταντίνος Ιωάννου ΑΜ:03119840

Άσκηση 1

```
IN 10H
MVI A,0DH           ; Κάνει μασκ το RST 6.5
       SIM
       EI                          ; Κάνει enable τα interrupts
LXI H,0A05H   ; Δίνουμε την τιμή 0 στα 2 αριστερότερα 7-seg-displays που θα χρησιμοποιήσ
       MVI M,00H
       DCX H
       MVI M,00H
       DCX H
       MVI C,04H
EMPTY_DISPLAY:              ; Αδειάζουμε τα 4 δεξιότερα 7-seg-displays
       MVI M,10H     ; (0A04) -> (0A00) = 10H κενά 7-seg-displays
       DCX H
       DCR C
       JNZ EMPTY_DISPLAY

RESET_DISPLAY:              ; Μηδενίζουμε τα αριστερότερα 7-seg-displays όταν τελειώσουν
       LXI H,0A05H
       MVI M,00H
       DCX H
       MVI M,00H
       MVI A,FFH
       STA 3000H
WAITTING_ROOM:              ; Περιμένει να πατηθεί το INTRPT κι δείχνει την 00 στο χρονόμ
       CALL DISPLAY         ; και θα γεμίσει τον A με άσσους σε περίπτωση που είχε κάποιο
       JMP WAITTING_ROOM ; κάνει enable τα interrupts ώστε να μπορεί να κάνει ανανέωση ά

INTR_ROUTINE:        ; Όταν πατηθεί το interrupt το PC θα πάει εδώ
       MVI C,2DH             ; θα δώσει στο C (=counter) την τιμή 45
       MVI A,FFH             ; και θα γεμίσει τον A με άσσους σε περίπτωση που είχε κάποιο
       EI                                  ; κάνει enable τα interrupts ώστε να μπορεί να

BLINK:
       CMA                         ; Αντιστροφή του A ώστε να αναβοσβήνουν τα LEDS
       STA 3000H
       CALL DEC_COUNTER            ; Διαμόρφωση του counter για να τυπωθεί η νέα τιμή

       MVI E,0AH                         ; Βαάζουμε στον καταχωρητή E τιμή 10 που θα είν
LOOP_DISPLAY1:                     ; την υποροutήνα DISPLAY για να αποφύγουμε να αναβωσβή
       CALL DISPLAY                ; Η DISPLAY κρατάει 25ms
       DCR E                                     ; την καλεί 10 φορές 10x25=250ms
       JNZ LOOP_DISPLAY1

       CMA                                       ; τα LEDS θα αλλάζουν τιμές κάθε 250ms ο
       STA 3000H                           ; το κάνουμε αυτό 4 φορές δηλαδή αναβοσβήνουν γ
```

```
       MVI E,0AH
LOOP_DISPLAY2:
       CALL DISPLAY
       DCR E
       JNZ LOOP_DISPLAY2

       CMA
       STA 3000H

       MVI E,0AH
LOOP_DISPLAY3:
       CALL DISPLAY
       DCR E
       JNZ LOOP_DISPLAY3

       CMA
       STA 3000H

       MVI E,0AH
LOOP_DISPLAY4:
       CALL DISPLAY
       DCR E
       JNZ LOOP_DISPLAY4

       DCR C                ; Μείωση του counter
       JNZ BLINK     ; Άμα δεν είναι 0 επανάλαβε
       JMP RESET_DISPLAY    ; αλλιώς κάνει RESET

DEC_COUNTER:
       PUSH PSW
       PUSH B
       PUSH H

       MVI B,FFH
       MOV A,C

MAKE_COUNTER:
       INR B                ; αυξάνουμε το B που αντιπροσωπεύει τις δεκάδες
       SUI 0AH              ;κι αφερούμε το A μεχρι να γίνει αρνητικό
       JNC MAKE_COUNTER

       ADI 0AH       ; του βάζουμε 10
```

```
        LXI H,0A04H
        MOV M,A             ; Αποθυκεύουμε την τιμή του A στο 7-seg-display
        INX H
        MOV M,B             ; Αποθυκεύουμε την τιμή του A στο 7-seg-display

        POP H
        POP B
        POP PSW
        RET

RET

DISPLAY:
        PUSH PSW
        PUSH D
        PUSH B
        LXI B,0019H
        LXI D,0A00H         ; Φέρνουμε τις διευθήνσεις που έχουμε αποθυκεύσει τις τιμές
        CALL STDM                   ; και τις βάζουμε την μνήμη που θα τις διαβάσει η DCD
        CALL DCD
        CALL DELB
        POP B
        POP D
        POP PSW
RET

END
```

Άσκηση 2

```
MAIN:   IN 10H
        MVI A,0DH       ; Κάνει μασκ το RST 6.5
        SIM
        EI                          ; Κάνει enable τα interrupts

        MVI B,06H
        LXI H,0A00H
FILL:   MVI M,10H       ; Αδειάζουμε τα 7-seg-displays
        INX H
        DCR B
        JNZ FILL

        MVI C,05H       ; K1
        MVI D,64H       ; K2
        MVI E,C8H       ; K3

        PUSH D
        LXI D,0A00H         ; Φέρνουμε τις διευθήνσεις που έχουμε αποθυκεύσει τις τιμές
        CALL STDM           ; και τις βάζουμε την μνήμη που θα τις διαβάσει η DCD
        POP D
        CALL DCD

WAIT1: JMP WAIT1        ; περιμένει να πατηθεί το INTRPT

INTR_ROUTINE:
        CALL KIND                   ; Διαβάζει την δεκάδα
        STA 0A03H                   ; την αποθυκεύει στην αριστερα απο το κέντρο 7-seg-dis
        RLC                                 ; τα πάμε στα τέσσερα MSB  και τον κάνουμε stor
        RLC
        RLC
        RLC
        MOV B,A
        CALL KIND                   ; Διαβάζει την μονάδα
        STA 0A02H
        ADD B           ;προσθέτει την με τον B για να πάρουμε όλο τον αριθμό για να τον συ
        CMP C           ; Αμα C=>A jump
        JC C_won
        JZ C_won
        JMP D_COMP      ; Αλλιώς jump D_COMP
C_won:
        MVI A,08H       ; Βάζει άσσο στην θέση που δηλώνει σε ποιά περιοχή βρήσκεται
        JMP END
```

```
D_COMP:             ; Άμα D=>A jump
       CMP D
       JC D_won
       JZ D_won
       JMP E_COMP    ; Αλλιώς jump E_COMP
D_won:
       MVI A,04H
       JMP END

E_COMP:             ; Άμα E=>A jump
       CMP E
       JC E_won
       JZ E_won
       JMP A_won           ; Αλλιώς jump A_won
E_won:
       MVI A,02H
       JMP END

A_won: MVI A,01H

END:
       CMA    ;αντιστρέφει την A για να ανάψει το σωστό LED
       STA 3000H
       PUSH D
       LXI D,0A00H
       CALL STDM
       POP D
       EI            ; Κάνει enable τα interrupts

DISPLAY:
       CALL DCD
       JMP DISPLAY

       END
```

Άσκηση 3

```
FILL MACRO ADDR,K
       PUSH PSW
       PUSH H
       MOV A,K
       LXI H,ADDR
LOOP0:
       MOV M,A
       INX H
       DCR A
       JNZ LOOP0
       POP H
       POP PSW
ENDM
```

```
INR16 MACRO ADDR
       PUSH PSW
       PUSH H
       LXI H,ADDR
       INR M
       INX H
       INR M
       POP H
       POP PSW
ENDM
```

```
RHLL MACRO Q,R
       PUSH PSW
       MOV A,Q
       RAL
       MOV Q,A
       MOV A,R
       RAL
       MOV R,A
       POP PSW
ENDM
```

Άσκηση 4

Δ| Στον μΕ εκτελείται η εντολή **JMP 2200H** , με **PC = 2000H** (μετρητής προγράμματος) και (δείκτης σωρού) **SP=3000H**.Στο μέσον της εκτέλεσης συμβαίνει διακοπή **RST 6.5** .

Ζ|Δώστε τις νέες τιμές των PC,SP,Περιεχόμενου σωρού μετρά την διακοπή καθώς και τις λειτουργίες που συμβαίνουν .

Λύση

Αρχικά ο μετρητής προγράμματος είναι PC = 2000H και ο δείκτης σωρού SP = 3000H. Καρά την εκτέλεση της εντολή **JMP** 2200H προκαλείται hard-ware διακοπή RST 6.5 , οπότε θα πάρει προτεραιότητα η εξυπηρέτηση της σε σχέση με το υπόλοιπο πρόγραμμα , οπότε οι λειτουργίες που θα ακολουθήσουν είναι οι εξής:

(POP is 2 bytes in 8085)

1) Ολοκληρώνεται η εκτέλεση της εντολής JMP 2200 H και προφανώς PC =2200H.

2) Έπειτα προκαλείται hardware διακοπή RST 6.5 και ο PC παίρνει ως τιμή την διεύθυνση 6.5*8 = 52 = **37H** . Ενώ στη στοίβα αποθηκεύεται η διεύθυνση 2200 H στις θέσεις μνήμης SP-1 = 2FFFH και SP -2 = 2FFCH.Συγκεκριμένα (2FFFH) = 22H και (2FFCH) = 00H

3) Αφου ολοκληρωθεί η ρουτίνα διακοπής RST 6.5 , γίνεται POP από την στοίβα , Δηλαδή αυξάνουμε το μέγεθος της στοίβας κατά 2 , (SP - 2 = 2FFEH).Στον PC θα αποθηκευτεί η διεύθυνση της διακοπής που αναγνωρίστηκε δηλαδή PC = 0037H

4) Όταν τελειώσει η εκτέλεση της ρουτίνας της διακοπής θα επαναφερθεί η παλιά τιμή του PC από την στοίβα και θα μειωθεί το μέγεθος της στοίβας κατά 2 (SP = SP +2) .Τελικά θα έχουμε PC = 2200H και SP =

2FFCH			SP-4|	
2FFDH			SP-3|	
2FFEH		SP-2|00H	SP-2|00H	SP-2|00H
2FFFH		SP-1|22H	SP-1|22H	SP-1|22H
SP	PC|2200H	PC|2200H	PC|0037H	PC|2200H
Στοίβα	1	2	3	4

Άσκηση 5

Α)

```
MVI A,0DH  ;Μάσκα διακοπών
SIM
LXI  H,0000H   ; Εδώ θα βάζουμε τα δεδομένα
MVI C,20H   ; counter = 32 (demical)
EI      ; Ενεργοποίηση δεδομένων

ADDR:
 MOV A,C
 CPI 00H
 JNZ ADDR  ; Έλεγχος εισόδου δεδομένων
 DI   ;Απενργοποιήση διακοπών
 MOV A,L
 ANI 80H
 CPI 00H
 JNZ CUTBACK ; περικοπη

;Για την περικοπή φτιάχνουμε ρουτίνα εδω
L1:
HLT

CUTBACK:
ANI 00H
JP L1

002C:
JMP RST5.5

RST5.5:

PUSH PSW
MOV A,C
ANI 01H  ; Κρατάει δυαδικό LSB 00000001 binary
JPO ALL_MSB   ; MSB  from data
IN 20H
ANI 0FH ; 00001111 δυαδικό για τα 4  LSB
MOV B,A   ; αποθηκέουμε προσωρνά τα LSB μέχρι να λάβουμε MSB
JMP ALL_LSB ;  LSB from data

ALL_MSB:
IN 20H
ANI 0FH
RLC
RLC
RLC
RLC
ORA B ; συολικό αποτέλεσμα μαζί με LSB
MVI D,00H
MOV E,A
DAD D

ALL_LSB:
   POP PSW
   DCR C ; counter--
   EI
   RET

END
```

B)

```
LXI  H,0000H
MVI C,20H  ; C =32 demical

MAIN:
IN 20H
ANI 00H ; wait until  x0 = 0
JNZ MAIN
MOV A,C
ANI 00H     ;00000001 binary
JPO ALL_MSB
IN 20H
ANI 0FH  ; 00001111 BINARY
MOV B,A
JMP ALL_LSB


ALL_MSB:
IN 20H
ANI 0FH
RLC
RLC
RLC
RLC
ORA B ; συολικό αποτέλεσμα μαζί με LSB
MVI D,00H
MOV E,A
DAD D

ALL_LSB:
 DCR C ;counter --
 JZ ADDR

CHECK:
 IN 20H
ANI 00H
JNZ CHECK  ; WAIT UNTIL X0 = 0
JMP MAIN
```

```
ADDR:
 MOV A,C
 CPI 00H
 JNZ ADDR  ; Έλεγχος εισόδου δεδομένων
 DI   ;Απενργοποιήση διακοπών
 MOV A,L
 ANI 80H
 CPI 00H
 JNZ CUTBACK ; περικοπη


CUTBACK:
ANI 00H
JP L1


;Για την περικοπή φτιάχνουμε ρουτίνα εδω
L1:
HLT


END
```