
Συστήματα Μικροϋπολογιστών – 3^η Σειρά Ασκήσεων

Κυριακόπουλος Γιώργος – el18153

Τζελέπης Σεραφείμ – el18849

1η Άσκηση:

START:

```
IN 10H
LXI H, 0A00H      ; Φορτώνουμε τη διεύθυνση για το 7-seg display.

MVI M, 10H        ; Γεμίζουμε με κενά τις θέσεις που δεν θα
INX H              ; χρησιμοποιήσουμε.
MVI M, 10H
INX H
INX H
INX H
MVI M, 10H
INX H
MVI M, 10H

MVI A, 0DH        ; Μάσκα διακοπών 0DH = 00001101.
SIM
EI                ; Επίτρεψη διακοπών.
```

WAIT:

```
JMP WAIT          ; Αναμονή για διακοπή.
```

INTR_ROUTINE:

```
MVI E, 3CH        ; Χρονόμετρο 60 δευτερολέπτων.
LXI B, 0032H      ; 50 ms καθυστέρηση.
EI
```

INIT:

```
MVI D, 14H        ; Επανάληψη 20 καθυστερήσεων για 20x50ms = 1 sec.
CALL NEXT_SEC

ANI 00H           ; Θέτουμε τον καταχωρητή A ίσο με 0, ώστε να
STA 3000H         ; ανάψουμε όλα τα LEDs.
```

L1:

```
CALL DISP      ; Απεικόνιση του χρόνου που απομένει.  
CALL DELB      ; Καθυστέρηση 50 ms.  
DCR D  
JNZ L1         ; Επανάληψη 20 φορές για 1 sec καθυστέρηση.  
  
DCR E          ; Μείωση του χρονομέτρου δευτερολέπτων.  
JNZ INIT       ; Επανάληψη, όσο δεν έχει περάσει 1 λεπτό.  
  
CMA            ; Σβήσιμο των LEDs, εάν περάσει ο χρόνος.  
STA 3000H  
JMP WAIT
```

NEXT_SEC:

```
PUSH PSW  
PUSH B  
PUSH H  
  
MVI B,FFH      ; Αρχικοποίηση μετρητή δεκάδων σύμφωνα και με το  
MOV A,E         ; παράδειγμα του βιβλίου.
```

L3: INR B

```
SUI 0AH        ; Αφαιρούμε 10 από το αριθμό που χειριζόμαστε.  
JNC L3         ; Μέχρι να έχουμε αρνητικό υπόλοιπο.
```

```
ADI 0AH        ; Προσθέτουμε 10 για διόρθωση και έχουμε τα  
LXI H,0A02H    ; ζητούμενα, B δεκάδες και A μονάδες.  
MOV M,A        ; Αποθηκεύουμε μονάδες και δεκάδες στις θέσεις  
INX H          ; που θέλουμε.  
MOV M,B
```

```
POP H  
POP B  
POP PSW  
RET
```

DISP:

```
PUSH PSW  
PUSH D  
LXI D,0A00H    ; Μετακίνηση του block 0A00H – 0A05H στο σημείο  
               ; που διαβάζει η DCD.  
  
CALL STDM
```

```
CALL DCD  
POP D  
POP PSW  
RET
```

```
END
```

2η Άσκηση:

START:

```
IN 10H
MVI A,0DH      ; Μάσκα διακοπών 0DH = 00001101.
SIM
EI             ; Επίτρεψη διακοπών.

MVI B,06H      ; Μετρητής επανάληψης.
LXI H,0A00H    ; Φορτώνουμε τη διεύθυνση για το 7-seg display.
```

FILL:

```
MVI M,10H      ; Γεμίζουμε με κενά όλες τις θέσεις.
INX H
DCR B
JNZ FILL

MVI D,55H      ; Πρώτο κατώφλι (ορισμένο από το χρήστη).
MVI E,AAH      ; Δεύτερο κατώφλι (ορισμένο από το χρήστη).

PUSH D
LXI D,0A00H    ; Καλούμε την STDM και DCD με κενά σε όλες τις
CALL STDM      ; θέσεις, ώστε να μην εμφανιστεί κάτι όταν
POP D          ; θα πατήσουμε το INTRPT κουμπί.
CALL DCD
```

WAIT:

```
JMP WAIT      ; Αναμονή για διακοπή.
```

INTR_ROUTINE:

```
DI             ; Απενεργοποίηση των διακοπών.
CALL KIND      ; Διαβάζουμε το χαμηλότερης αξίας ψηφίο.
STA 0A04H
MOV B,A
CALL KIND      ; Διαβάζουμε το υψηλότερης αξίας ψηφίο.
STA 0A05H
RLC            ; 4 δεξιές περιστροφές ώστε το υψηλότερης αξίας
RLC            ; ψηφίο να πάει στις MSB θέσεις.
RLC
RLC
ADD B          ; Προσθέτουμε το χαμηλότερης αξίας ψηφίο και έτσι
               ; έχουμε έτοιμο τον αριθμό για τις συγκρίσεις.

CMP D
```

```
JC FIR          ; Εάν ο αριθμός μας είναι μικρότερης ή ίσος από  
JZ FIR          ; το κατώτερο κατώφλι τότε άναψε το 3ο από το  
JMP NOFIR       ; τέλος LED, αλλιώς συνέχισε στην άλλη σύγκριση.
```

FIR:

```
MVI A,04H      ; 04H = 0100 (3ο LED από το τέλος).  
JMP FINAL
```

NOFIR:

```
CMP E  
JC SEC          ; Εάν ο αριθμός μας είναι μικρότερης ή ίσος από  
JZ SEC          ; το δεύτερο κατώφλι τότε άναψε το 2ο από το  
JMP LAST       ; τέλος LED, αλλιώς άναψε το LSB LED.
```

SEC:

```
MVI A,02H      ; 02H = 0010 (2ο LED από το τέλος).  
JMP FINAL
```

LAST:

```
MVI A,01H      ; 01H = 0001 (LSB LED).
```

FINAL:

```
CMA             ; Αντιστροφή λόγω αρνητικής λογικής των LEDs.  
STA 3000H
```

```
PUSH D  
LXI D,0A00H    ; Ετοιμάζουμε την STDM και το σημείο της μνήμης  
CALL STDM      ; από το οποίο θα διαβάσει η DCD.  
POP D  
EI             ; Εκ νέου ενεργοποίηση των διακοπών.
```

DISP:

```
CALL DCD       ; Κλήση της DCD για την απεικόνιση και ατέρμων  
JMP DISP       ; βρόγχος, μέχρι να έρθει η επόμενη διακοπή.
```

END

3η Άσκηση:

α)

```
SWAP Nibble MACRO Q
    PUSH PSW
    MOV A,Q      ; Μεταφορά του καταχωρητή στον A.
    RLC          ; 4 αριστερές περιστροφές, με αποτέλεσμα τα 4 MSB
    RLC          ; bits να αλλάξουν θέση με τα 4 LSB bits.
    RLC
    RLC
    MOV Q,A      ; Γυρίζουμε το αποτέλεσμα στον αρχικό καταχωρητή.
    MOV A,M      ; Μεταφορά της σημείου της μνήμης στο οποίο δείχνει
    RLC          ; το H-L ζευγάρι στον A.
    RLC          ; 4 αριστερές περιστροφές, με παρόμοια λογική.
    RLC
    RLC
    MOV M,A      ; Γυρίζουμε το αποτέλεσμα στο σημείο που ήταν.
    POP PSW      ; Επαναφορά των καταχωρητών και των flags που
ENDM             ; πειράξαμε.
```

β)

```
FILL MACRO RP, X, K
    PUSH PSW      ; Αποθήκευση στη στοίβα όσους καταχωρητές
    PUSH H        ; χρησιμοποιούμε συν τα flags.
    PUSH L
    MOV H,R        ; Μετακινούμε τον R στον H και τον P στον L,
    MOV L,P        ; ώστε να διαχειριζόμαστε τη μνήμη μέσω αυτών.
    MOV A,X        ; Αποθηκεύουμε το μήκος στον A.
    FOR:
        MOV M,K    ; Βάλε K στο σημείο μνήμης που δείχνει το H-L.
        INX H      ; Αύξηση του H-L κατά 1 για το επόμενο σημείο.
        DCR A      ; Μείωσε το μήκος (A).
        JNZ FOR    ; Επανάληψη μέχρι να μήκος (A) να γίνει 0.

        POP L      ; Επαναφορά των καταχωρητών και των flags που
        POP H      ; πειράξαμε.
        POP PSW
ENDM
```

γ)

```
RHLR MACRO N
    PUSH PSW      ; Αποθήκευση στη στοίβα όσους καταχωρητές
    PUSH B        ; χρησιμοποιούμε συν τα flags.
    MVI A, N      ; Αποθηκεύουμε τον αριθμό περιστροφών στον A.
    CPI 00H       ; Έλεγχος εάν ο αριθμός περιστροφών είναι 0.
    JZ END
    MVI B, N      ; Αποθηκεύουμε τον αριθμό των περιστροφών και στον B.
FOR:
    MOV A, H      ; Το CY πάει MSB του H και το LSB του H πάει CY.
    RAR
    MOV H, A
    MOV A, L
    RAR           ; Το CY πάει MSB του L και το LSB του L πάει CY.
    MOV L, A
    DCR B         ; Μείωση του αριθμού περιστροφών.
    JNZ FOR       ; Επανάληψη μέχρι ο αριθμός περιστροφών να γίνει 0.
END:
    POP B         ; Επαναφορά των καταχωρητών και των flags που
    POP PSW       ; πειράξαμε.
ENDM
```

4η Άσκηση:

Στο μΕ 8085 εκτελείται η εντολή CALL 0880H. Την στιγμή αυτή ο μετρητής προγράμματος PC βρίσκεται στην θέση 0800H και ο δείκτης σωρού SP = 3000H. Στο μέσο της εντολής αυτής συμβαίνει η hardware διακοπή(υποθέτουμε ότι υπάρχουν η κατάλληλη μάσκα και η εντολή ενεργοποίησης διακοπών EI) άρα αρχικά θα ολοκληρωθεί η εκτέλεση της τρέχουσας εντολής και ο μετρητής προγράμματος θα πάει στην θέση 0880H και η στοίβα θα ανέβει δυο θέσεις αποθηκεύοντας την διεύθυνση 0800H από την οποία καλέστηκε η εντολή CALL 0880H.

Έτσι έχουμε αρχικά:

PC: 0800H

SP: 0300H

→

ADDRESS	CONTENT
0800H	CALL 0880H
...	<DATA>
...	<DATA>

→

ADDRESS	STACK
0300H	<DATA>

Αφού ολοκληρωθεί η εντολή CALL 0880H και σύμφωνα με ότι ειπώθηκε παραπάνω θα έχουμε:

PC: 0880H

SP: 0302H

→

ADDRESS	CONTENT
0880H	<DATA>
...	<DATA>
...	<DATA>


→

ADDRESS	STACK
0302H	00
0301H	08
0300H	<DATA>


Στην συνέχεια αφού ολοκληρώθηκε η εντολή CALL 0880H σώζεται στη στοίβα ο PC άρα ανεβαίνει δυο θέσεις και θα εκτελεστεί η ρουτίνα εξυπηρέτησης της διακοπής RST 7.5 άρα ο μετρητής προγράμματος PC λαμβάνει την τιμή $8 \times 7.5 = 60(\text{DEC}) = 003C(\text{HEX})$ οπότε έχουμε :

PC: 003CH

SP: 0304H



ADDRESS	CONTENT
003CH	<DATA>
...	<DATA>
...	<DATA>




ADDRESS	STACK
0304H	80
0303H	08
0302H	00
0301H	08
0300H	<DATA>


Αφού εκτελεστεί η ρουτίνα εξυπηρέτησης της διακοπής τότε το PC επιστρέφει στην διεύθυνση που βρισκόταν στην κορυφή της στοίβας οπότε η στοίβα κατεβαίνει δύο θέσεις. Άρα έχουμε:

PC: 0880H

SP: 0302H



ADDRESS	CONTENT
0880H	<DATA>
...	<DATA>
...	<DATA>



ADDRESS	STACK
0302H	00
0301H	08
0300H	<DATA>

Τέλος αφού ολοκληρωθεί η ρουτίνα που έχουμε καλέσει και βρίσκεται στην διεύθυνση 0880H ο μετρητής ο προγράμματος επαναφέρεται στην διεύθυνση που βρίσκεται στην κορυφή της στοίβας και η στοίβα κατεβαίνει δυο θέσεις άρα έχουμε:

PC: 0800H



ADDRESS	CONTENT
0800H	CALL 0880H
...	<DATA>
...	<DATA>

SP: 0300H



ADDRESS	STACK
0300H	<DATA>

5η Άσκηση:

α)

MAIN:

```
MVI A,0DH      ; Μάσκα διακοπών 0DH = 00001101.
SIM
LXI H,0000H    ; Αρχικοποίηση συσσωρευτή.
MVI C,40H      ; Μετρητής 64 αναγνώσεων.
EI             ; Ενεργοποίηση διακοπών.
```

WAIT:

```
MOV A,C        ; Μέχρι ο μετρητής να γίνει 0 διάβαζε τα
CPI 00H        ; δεδομένα από την θύρα εισόδου.
JNZ WAIT

DI             ; Απενεργοποίηση διακοπών.
DAD H          ; Για να βρούμε το μέσο όρο προσθέτουμε 3 φορές
DAD H          ; το H-L στον εαυτό του, ώστε να γίνει αριστερή
DAD H          ; ολίσθηση 3 θέσεις, ισοδύναμη με δεξιά ολίσθηση
HLT           ; 5 θέσεων που είναι η διαίρεση με το 32 = 2^5.
```

0034:

```
JMP RST6.5    ; Πήγαινε στην διαδικασία εξυπηρέτησης διακοπής.
```

RST6.5:

```
PUSH PSW

MOV A,C        ; Ελέγχουμε το μετρητή, εάν είναι άρτιος διάβασε
ANI 01H        ; τα LSBs από την παρακάτω διαδικασία, αλλιώς
CPI 00H        ; διάβασε τα MSBs μέσω του άλματος.
JNZ GET_MSBS
```

GET_LSBs:

```
IN 20H        ; Διάβασε την είσοδο και απομόνωσε τα 4 LSBs
ANI 0FH       ; X0 έως και X3.
MVI D,00H     ; Βάλε στον καταχωρητή D το 0.
MOV E,A       ; Βάλε στον καταχωρητή E τα 4 LSBs.
DCR C        ; Μείωσε τον μετρητή κατά 1.
JMP EXIT
```

GET_MSBS:

```
IN 20H        ; Διάβασε την είσοδο και απομόνωσε τα 4 LSBs
ANI 0FH       ; X0 έως και X3.
```

```

RLC          ; 4 αριστερές περιστροφές, ώστε τα ψηφία που
RLC          ; που διάβασες να πάνε στη θέση των MSBs.
RLC
RLC
MVI D,00H   ; Βάλε στον καταχωρητή D το 0.
ORA E       ; Κάνε OR τον A που έχει τα MSBs στη σωστή θέση
MOV E,A     ; με τον E που έχει τα LSBs και αποθήκευσε τον.
DAD D       ; Πρόσθεσε το αποτέλεσμα στον συσσωρευτή.
DCR C       ; Μείωσε τον μετρητή κατά 1.

```

EXIT:

```

POP PSW
EI          ; Ενεργοποίηση διακοπών.
RET

```

END

β)

MAIN:

```

LXI H,0000H ; Αρχικοποίηση συσσωρευτή.
MVI C,40H   ; Μετρητής 64 αναγνώσεων.

```

READY:

```

MOV A,C     ; Μέχρι ο μετρητής να γίνει 0 διάβαζε τα
CPI 00H     ; δεδομένα από την θύρα εισόδου.
JZ AVERAGE ; Όταν γίνει 0 υπολόγισε το μέσο όρο.

IN 20H      ; Διάβασε την είσοδο και αποθήκευσε την,
MOV B,A     ; ελέγχοντας εάν το X7 είναι 1, ώστε να επιλέξει
RAL         ; ανάμεσα σε ανάγνωση η επανάληψη μέχρι X7 = 1.
JC SELECT
JMP READY

```

SELECT:

```

MOV A,C     ; Ελέγχουμε το μετρητή, εάν είναι άρτιος διάβασε
ANI 01H     ; τα LSBs από την παρακάτω διαδικασία, αλλιώς
CPI 00H     ; διάβασε τα MSBs μέσω του άλματος.
JNZ GET_MSBS

```

GET_LSBs:

```

MOV A,B     ; Ανάκτησε την είσοδο και απομόνωσε τα 4 LSBs
ANI 0FH     ; X0 έως και X3.

```

```

MVI D,00H      ; Βάλε στον καταχωρητή D το 0.
MOV E,A        ; Βάλε στον καταχωρητή E τα 4 LSBs.
DCR C          ; Μείωσε τον μετρητή κατά 1.
JMP EXIT

```

GET_MSBS:

```

MOV A,B        ; Ανάκτησε την είσοδο και απομόνωσε τα 4 LSBs
ANI 0FH        ; X0 έως και X3.
RLC            ; 4 αριστερές περιστροφές, ώστε τα ψηφία που
RLC            ; που διάβασες να πάνε στη θέση των MSBs.
RLC
RLC
MVI D,00H      ; Βάλε στον καταχωρητή D το 0.
ORA E          ; Κάνε OR τον A που έχει τα MSBs στη σωστή θέση
MOV E,A        ; με τον E που έχει τα LSBs και αποθήκευσε τον.
DAD D          ; Πρόσθεσε το αποτέλεσμα στον συσσωρευτή.
DCR C          ; Μείωσε τον μετρητή κατά 1.

```

EXIT:

```

IN 20H         ; Διάβασε την είσοδο και έλεγξε εάν το X7 = 0.
RAL            ; Εάν όχι, τότε επανάλαβε μέχρι να γίνει 0.
JC EXIT
JMP READY

```

AVERAGE:

```

DAD H          ; Για να βρούμε το μέσο όρο προσθέτουμε 3 φορές
DAD H          ; το H-L στον εαυτό του, ώστε να γίνει αριστερή
DAD H          ; ολίσθηση 3 θέσεις, ισοδύναμη με δεξιά ολίσθηση
END            ; 5 θέσεων που είναι η διαίρεση με το 32 = 2^5.

```

Σημείωση:

Θεωρήσαμε πως μετά από κάθε ανάγνωση δεδομένων πρέπει το σήμα Data Ready (X_7) να γυρνάει στην κατάσταση 0. Δηλαδή, γίνεται 1, διαβάζουμε τα LSBs, γίνεται 0, γίνεται 1, διαβάζουμε τα MSBs, γίνεται 0, γίνεται πάλι 1 κοκ.