Εθνικο Μετσοβιο Πολυτεχνειο

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧ. ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΫΠΟΛΟΓΙΣΤΩΝ ΚΑΙ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΣΥΣΤΗΜΑΤΑ ΜΙΚΡΟΫΠΟΛΟΓΙΣΤΩΝ

3η ΑΣΚΗΣΗ

Αχλάτης Στέφανος-Σταμάτης (03116149)

<el16149@central.ntua.gr>

Καπερώνη Φρειδερίκη (03116685)

<el16685@central.ntua.gr>

Ιούνιος 2020

Άσκηση 1:
Το πρόγραμμα μαζί με τα απαραίτητα σχόλιο παρατίθενται ακολούθως όπως ακριβώς δόθηκαν στον προσομοιωτή:

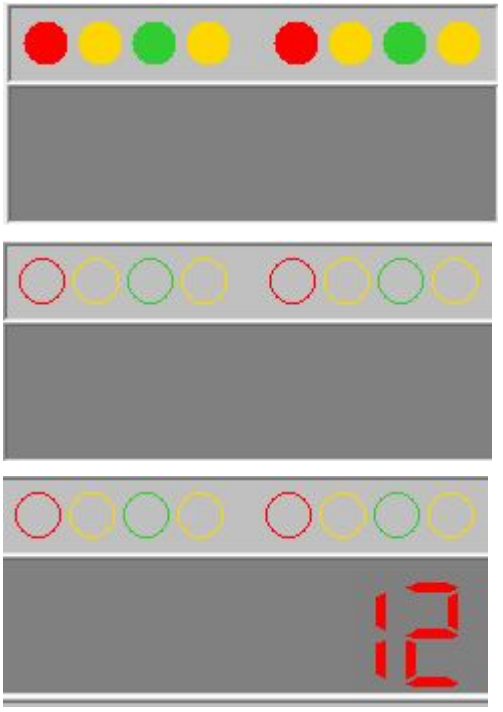
```
START:
       IN 10H ; Άρση της προστασίας μνήμης
       MVI A,0DH ; Αρχικοποίηση διακοπής 6.5
       SIM
       MVI A,FFH ; Σβήσιμο των leds εξόδου
       STA 3000H
       MVI A,00H ; Αρχικοποίηση σημαίας διακοπής, την οποία
       STA 0900H ; διατηρούμε στην ελεύθερη θέση μνήμης 0900H
       LXI H,0BF0H ; Αρχικοποίηση του τμήματος μνήμης που
       MVI A,10H ; χρησιμοποιείται από τη ρουτίνα DCD με τιμές
       MOV M,A ; τις οποίες απεικονίζει σε σβηστά 7 segment
       INX H ; displays
       MOV M,A
       INX H
       MOV M,A
       INX H
       MOV M,A
       INX H
       MOV M,A
       INX H
       MOV M,A
       EI ; Ενεργοποίηση των διακοπών

WAIT:           ; Αναμονή για διακοπή
       LDA 0900H ; Ελέγχουμε αν έχει γίνει 1 η σημαία, που
       CPI 00H ; σημαίνει ότι έγινε διακοπή
       JZ WAIT
       MVI A,00H ; Άναμμα των leds και αλλαγή της σημαίας της
       STA 3000H ; διακοπής
       STA 0900H
       MVI E,3CH ; Ο E == counter για τις 60 επαναλήψεις

MINUTE_LOOP:
       MOV A,E ; Το τμήμα αυτό του προγράμματος μετατρέπει τον
       MVI D,FFH ; counter της επανάληψης (E), ο οποίος είναι
```

(Συνεχίζει στην επόμενη σελίδα)

```
CALC_SEC: ; δυαδικός αριθμός (μεταξύ 0 και 60), σε
       INR D ; σε δεκαδικό. Τα δύο ψηφία του αποθηκεύονται
       SUI 0AH ; στις θέσεις μνήμης 0BF0H και 0BF1H,
       JNC CALC_SEC ; απεικονίζονται στα δύο δεξιότερα 7 segment
       ADI 0AH ; displays
       STA 0BF0H
       MOV A,D
       STA 0BF1H

OUT:
       CALL DCD
       MVI A,FFH
       LXI B,0860H
       STA 3000H
       CALL DELB
       CMA
       STA 3000H
       CALL DELB
       CMA
       STA 3000H
       CALL DELB
       CMA
       STA 3000H
       CALL DELB
       CMA
       STA 3000H ; Με αυτόν τον τρόπο πετυχαίνουμε να έχουμε περίοδο 1/2sec
       CALL DELB ; στον pc που τρεξαμε τις προσομοιώσεις
       LDA 0900H ; Έλεγχος για το αν ξαναέγινε διακοπή
       CPI 00H
       JZ REFRESH_END_IF ; Αν έγινε, τότε επαναφέρω τον counter της
       MVI E,3DH ; επανάληψής μου στην αρχική του τιμή και
       MVI A,00H ; μηδενίζω τη σημαία διακοπής.
       STA 0900H ; Παρατήρηση: επαναφέρουμε τον counter (E) στην
                 ; τιμή 61, διότι αμέσως μετά ακολουθεί εντολή DCR,
                 ; ενώ εμείς θέλουμε η νέα εκτέλεση της επανάληψης
                 ; να ξεκινήσει με τον counter στο 60

REFRESH_END_IF:
       DCR E
       JNZ MINUTE_LOOP
       LXI H,0BF0H ; Απεικόνιση στα 7 segment displays του αριθμού 00
       MVI M,00H
       MVI M,00H
       CALL DCD
       MVI A,FFH ; Σβήσιμο των leds
       STA 3000H
       JMP WAIT

INTR_ROUTINE: ; Η ρουτίνα εξυπηρέτησης της διακοπής το μόνο που κάνει
              ; είναι να "σηκώνει" τη σημαία που μας δείχνει
              ; ότι έχει γίνει διακοπή, την οποία έχουμε
              ; αποθηκευμένη στη θέση μνήμης 0900H

       PUSH PSW ; Αποθήκευση της κατάστασης του μΕ, αφού πρόκειται
                ; να αλλάξουμε τα περιεχόμενα του καταχωρητή A

       MVI A,01H
       STA 0900H ; "Σήκωμα" του flag
       POP PSW
       EI ; Επανενεργοποίηση των διακοπών
       RET
END
```

Ας σημειωθεί ότι οι χρόνοι υπολογίστηκαν πειραματικά για το δεδομένο υπολογιστικό σύστημα έτσι ώστε να πετυχαίνουμε τις ακριβείς προδιαγραφές που ζητάει η άσκηση.

Ας δούμε ορισμένα αποτελέσματα της προσομοίωσης:



Όπως φαίνεται και από τα αποτελέσματα τα leds αναβοσβήνουν 2 φορές στην διάρκεια του ενός δευτερολέπτου και το ποσό του χρόνου που απομένει φαίνεται αμέσως μετά στα 2 lsb του 7 segment display

Άσκηση 2:
Το πρόγραμμα μαζί με τα απαραίτητα σχόλιο παρατίθενται ακολούθως όπως ακριβώς δόθηκαν στον προσομοιωτή:

```
START:
       IN 10H ; Άρση της προστασίας μνήμης
       MVI A,0DH ; Αρχικοποίηση της διακοπής RST6.5
       SIM
       MVI B,00H ; Αρχικοποίηση σημαίας που μας δείχνει αν η
                 ; διακοπή προήλθε από πάτημα ή από άφημα του
                 ; πλήκτρου INTRPT.
                 ; E=0 -> πάτημα
                 ; E=1 -> άφημα
                 ; Η ζητούμενη λειτουργία της άσκησης υλοποιείται
                 ; μετά το άφημα του πλήκτρου
       MVI C,10H ; Ορισμός κατωφλιού K1
       MVI D,80H ; Ορισμός κατωφλιού K2
       MVI E,F0H ; Ορισμός κατωφλίου K3
       INR C ; Αύξηση των κατωφλιών κατά ένα, ώστε να μπορούμε
       INR D ; να κάνουμε τις απαραίτητες εντολές διακλάδωσης
       INR E ; με μία μόνο εντολή JMP.
       LXI H,0900H ; Αρχικοποίηση του τμήματος μνήμης που θα
       MVI A,10H ; χρησιμοποιηθεί για την απεικόνιση των εξόδων
       MOV M,A ; των 7 segment displays με τιμές τις οποίες η
       INX H ; ρουτίνα DCD απεικονίζει σε σβηστά 7 segment
       MOV M,A ; displays (διευθύνσεις 0900H-0905H).
       INX H
       MOV M,A
       INX H
       MOV M,A
       INX H
       MOV M,A
       INX H
       MOV M,A
       EI ; Ενεργοποίηση των διακοπών

WAIT:     ; Βρόχος αναμονής διακοπής
       DI ; Εντολές για συνεχή απεικονιση της εξόδου στα
          ; 7 segment displays.

       LXI D,0900H
       PUSH PSW ; Αποθήκευση όλων των καταχωρητών
       PUSH B
       PUSH D
       PUSH H
```

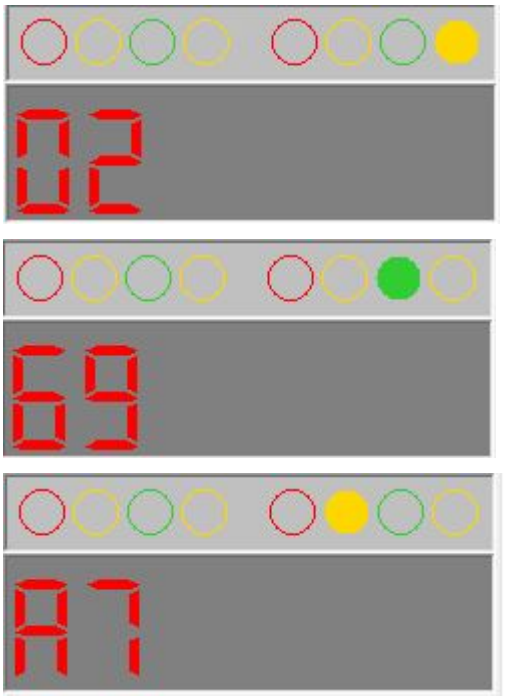
(Συνεχίζει στην επόμενη σελίδα)

```
        CALL STDM
        POP H
        POP D
        POP B
        POP PSW
        MVI D,80H ; Επαναφορά των περιεχομένων των καταχωρητών D και E
        MVI E,F0H
        INR D
        INR E
        CALL DCD
        EI ; Επανενεργοποίηση των διακοπών
        JMP WAIT

INTR_ROUTINE:  ; Ρουτίνα εξυπηρέτησης διακοπής
        MOV A,B ; Τμήμα ελέγχου της σημαίας. Αν είναι 0, τότε η
        CPI 01H ; διακοπή προέρχεται από πάτημα του διακόπτη,
        JZ MAIN_INTR ; οπότε την αγνοούμε και απλώς επιστρέφουμε.
        XRI 01H ; Αλλιώς, προχωράμε στο κύριο μέρος της
        MOV B,A ; ρουτίνας.
        EI
        RET

MAIN_INTR:      ; Κύριο μέρος της ρουτίνας
        CALL KIND ; Διάβασμα των δύο αριθμών που θα πατηθούν στο
        STA 0904H ; πληκτρολόγιο και αποθήκευσή τους στις θέσεις
        MOV D,A ; μνήμης 0904H και 0905H, που είναι αυτέ που θα
        CALL KIND ; απεικονιστούν στα δύο αριστερότερα 7 segment
        STA 0905H ; displays.
        RRC ; Τα περιεχόμενα των καταχωρητών A και D μετά την
        RRC ; είσοδο βρίσκονται στη μορφή 0000XXXX (δυαδικός), με
        RRC ; τον A να περιέχει τα 4 MSB και τον D τα 4 LSB
        RRC ; του 8-bit αριθμού που θέλουμε να πάρουμε ως
            ; είσοδο. Έτσι, περιστρέφουμε τον A 4 θέσεις και
        ORA D ; με μία OR, τους συγχωνεύουμε.
        MVI D,80H ; Επαναφέρουμε τα περιεχόμενα του καταχωρητή D
        INR D
        CMP E ; Αυτό το τμήμα του προγράμματος καθορίζει σε
        JC LESS_THAN_K3 ; ποια από τις 4 περιοχές βρίσκεται ο αριθμός
        MVI A,08H
        JMP END1 ; αποθηκευει στον A την τιμή που πρέπει να δοθεί
                 ; ως έξοδος στα leds.
```

(Συνεχίζει στην επόμενη σελίδα)

```
LESS_THAN_K3:
       CMP D
       JC LESS_THAN_K2
       MVI A,04H
       JMP END1
       LESS_THAN_K2:
       CMP C
       JC LESS_THAN_K1
       MVI A,02H
       JMP END1
       LESS_THAN_K1:
       MVI A,01H

END1:  ; Έξοδος της κατάλληλης τιμής στα leds
       CMA
       STA 3000H
       MOV A,B ; Απεικόνιση του αριθμού, που είναι αποθηκευμένος
       LXI D,0900H ; στον A, στα 7 segment displays
       PUSH PSW
       PUSH B
       PUSH D
       PUSH H
       CALL STDM
       POP H
       POP D
       POP B
       POP PSW
       CALL DCD
       MVI D,80H ; Επαναφορά των περιεχομένων των καταχωρητών D και E
       MVI E,F0H
       INR D
       INR E
       XRI 01H ; Ενημέρωση της σημαίας και επαναφορά της στον B
       MOV B,A
       EI
       RET
END
```

Ας δούμε ορισμένα αποτελέσματα της προσομοίωσης:



Θεωρητικές Ασκήσεις

3η ΑΣΚΗΣΗ: ***α)*** Δώστε τη μακροεντολή INR16 ADDR που να αυξάνει έναν αριθμό X των 16 bit αποθηκευμένο σε 2 διαδοχικές θέσεις στη μνήμη ως εξής: X_{LOW}=(ADDR) , X_{HIGH}=(ADDR+1). Το αποτέλεσμα να επιστρέφει στις ίδιες θέσεις. Η εκτέλεση της μακροεντολής δεν πρέπει να επηρεάζει τα περιεχόμενα των υπολοίπων καταχωρητών γενικού σκοπού.

```
INR16 MACRO ADDR
        PUSH PSW    ; επειδη χειριζόμαστε τον A
        PUSH H      ; επειδή χειριζόμαστε τους H-L

        LXI H,ADDR
        MOV A,M     ; φορτώνουμε το low από την μνήμη
        CPI FFH     ; ελεγχουμε αν είναι ίσο με το 11111111
        JNZ CASE_2  ; αν δεν είναι πήγαινε στην CASE_2
        ANI 00H     ; αν είναι τότε το low θα πρέπει να γίνει ίσο με 0
        MOV M,A     ; και να το τοποθετήσουμε στην θέση που ήταν στην μνήμη
        INX         ; και μετά να πάρουμε από την μνήμη
        MOV A,M     ; το high
        ADI A, 01H  ; και να το αυξήσουμε κατά ένα
        MOV M,A     ; και μετά να το βάλουμε στην θέση που ήταν στην μνήμη
        JMP TELOS   ; και μετά πηγαινε στο TELOS και κάνε τις τελικές ρυθμίσεις

CASE_2:
        ADI A, 01H  ; αυξησε το περιεχόμενο του A κατά ένα
        MOV M,A     ; και φόρτωσε το στην μνήμη
        JMP TELOS   ; και μετά πηγαινε στο TELOS και κάνε τις τελικές ρυθμίσεις

TELOS:
        POP H       ; επανέφερε παλιό H-L
        POP PSW     ; επανέφερε παλιό A και flags
ENDM
```

β) Δώστε τη μακροεντολή FILL ADDR, *K*, η οποία γεμίζει ένα τμήμα μνήμης με αρχική διεύθυνση ADDR και μήκος K με τους αριθμούς *K, K*-1, ..2, 1. Το μέγεθος του τμήματος K μπορεί να είναι από 0 έως 255. Θεωρείστε ότι για K=0 το μέγεθος του τμήματος να είναι ίσο με 256 και οι αριθμοί που θα αποθηκευθούν να είναι 0, 255, 254, …1.

```
FILL MACRO ADDR, K
        PUSH B          ; επειδη χειριζόμαστε τον A
        PUSH H          ; επειδη χειριζόμαστε τους H-L
        LXI H, ADDR
        MVI B, K
        JZ              ; αν το K είναι ίσο με το μηδεν πηγαινε στο LOOP1
        JMP LOOP2       ; αν το K δεν είναι ίσο με το μηδεν πήγαινε στο LOOP2

LOOP1:
        MVI B, 255d     ; τοποθετούμε στο B τον αριθμό 255
        MVI M, 00H      ; τοποθετούμε στην πρώτη θέση της μνήμης(ADDR) το μηδέν
        INX             ; αυξάνουμε την διευθυνση της μνήμης κατά ένα
LOOPA:
        MVI M, B        ; βάζουμε το περιεχόμενο του B στην μνήμη, την πρώτη φορα τον
255
        INX H           ; αυξάνουμε την διευθυνση της μνήμης κατά ένα
        DCR B           ; μειώνουμε το περιεχόμενο του B κατά ένα
        JNZ LOOPA       ; όσο το περιεχόμενο του B είναι μεγαλύτερο του μηδενος επανέλαβε
        JMP TELOS       ; όταν γίνει ίσο με μηδέν πήγαινε στο TELOS

LOOP2:
        MVI B, K        ; τοποθετούμε στο B τον αριθμό K
LOOPB:
        MVI M,B         ; βάζουμε το περιεχόμενο του B στην μνήμη, την πρώτη φορα τον K
        INX             ; αυξάνουμε την διευθυνση της μνήμης κατά ένα
        DCR B           ; μειώνουμε το περιεχόμενο του B κατά ένα
        JNZ LOOPB       ; όσο το περιεχόμενο του B είναι μεγαλύτερο του μηδενος επανέλαβε
        JMP TELOS       ; όταν γίνει ίσο με μηδέν πήγαινε στο TELOS

TELOS:
        POP H           ; επανέφερε παλιό H-L
        POP B           ; επανέφερε παλιό B-C (το C δεν το πειράξαμε έτσι και αλλιώς)
ENDM
```

γ) Δώστε τη μακροεντολή RHLR Q, R που περιστρέφει τα περιεχόμενα του κρατουμένου CY, των καταχωρητών Q και R κατά μια θέση αριστερά. Οι καταχωρητές Q και R μπορεί να είναι ένας συνδυασμός εκ των B, C, D, E, H και L (φυσικά Q≠R). Η μακροεντολή συμπεριφέρεται στα CY, Q και R σαν να είναι ένας 17-bit καταχωρητής: CY(17ο bit): Q(16ο -9ο bit): R(8ο -1ο bit). Μπορείτε να κάνετε χρήση της στοίβας για την αποθήκευση και επαναφορά τιμής καταχωρητών.

```
RHLR MACRO Q,R
        PUSH PSW    ; επειδη χειριζόμαστε τον A
        MOV A, R
        RAL
        MOV R,A     ; τώρα ο καταχωρητής R έχει την μορφή: R6|R5|R4|R3|R2|R1|R0|CY
                    ; και το CY έχει την τιμή R7

        MOV A,Q
        RAL
        MOV Q,A     ; τώρα ο καταχωρητής Q έχει την μορφή: Q6|Q5|Q4|Q3|Q2|Q1|Q0|CY
                    ; αλλα το (CY)=R7 άρα Q έχει την μορφή: Q6|Q5|Q4|Q3|Q2|Q1|Q0|R7
                    ; και το CY έχει την τιμή Q7
        POP PSW     ; επανέφερε παλιό A και flags
ENDM
```

4η ΑΣΚΗΣΗ: Στο μΕ 8085 εκτελείται η εντολή **JMP 0900H**. Ο μετρητής προγράμματος είναι (PC)=0800H και ο δείκτης σωρού (SP)=1FF0H. Στο μέσον της εκτέλεσης της εντολής συμβαίνει διακοπή RST 6.5. Δώστε τις νέες τιμές των PC, SP, το περιεχόμενο του σωρού καθώς και τις λειτουργίες που συμβαίνουν.

Θα σκιαγραφήσουμε τη συμπεριφορά του μετρητή προγράμματος PC και του δείκτη στοίβας SP του μΕ 8085, κατά τη διάρκεια εκτέλεσης ενός προγράμματος, που περιέχει την εντολή JMP 0900H, σε κάποιο σημείο του, και εμφανίζεται διακοπή τύπου RST 6.5. Υποθέτουμε ότι ο μηχανισμός αναγνώρισης διακοπών του μΕ 8085 είναι ενεργοποιημένος (εντολή EI) και ότι η διακοπή RST 6.5 δεν έχει παρεμποδιστεί από κάποια μάσκα.

Αρχικά η εντολη JMP 0900H βρίσκεται στην διεύθυνση 0800H, η οποία περιέχεται στο PC. Μόλις την εκτελέσουμε, η διεύθυνση της επόμενης εντολής του προγράμματος, η 0803H, αποθηκεύεται στην στοίβα και ο PC παίρνει τη διεύθυνση 0900H:

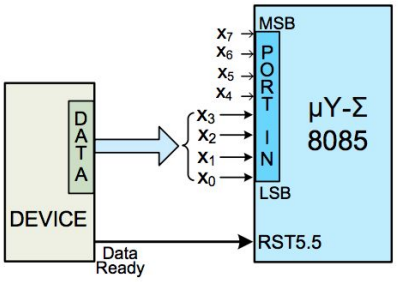
JMP 0900H
(1FEF) <- 08H
(1FEE) <- 03H
(SP) <- 1FEE
(SP) <- 0900H

Κατά την διάρκεια εκτέλεσης της JMP γίνεται διακοπή RST 6.5. Αφού ολοκληρωθεί η τρέχουσα εντολή αναγνωρίζεται η διακοπή, αφού η διακοπή εξυπηρείται αφ ότου ολοκληρωθεί η εκτέλεση της εντολής. Η πρωταρχική του κίνηση της διακοπής είναι να ενεργοποιήσει όλες τις διακοπές (εκτός φυσικά της TRAP, που δεν επιδέχεται απενεργοποίησης) και μετά η διακοπή επιφέρει τα εξείς αποτελέσματα:

RST 6.5
(1FED) <- 09H
(3FEC) <- 00H
(SP) <- 3FEC
(PC) <- 0034H (διεύθυνση διακοπής)

Έπειτα, εκτελείται η ρουτίνα εξυπηρέτησης της διακοπής RST 6.5, με πρωταρχικό βήμα την πλήρη αποθήκευση της κατάστασης του μΕ και συγκεκριμένα όλων των καταχωρητών, των οποίων οι τιμές τροποποιούνται από αυτή, στη στοίβα (με τη χρήση της εντολής PUSH). Μόλις πριν την ολοκλήρωση της ρουτίνας, χρειάζεται να προηγηθεί η ανάκτηση από τη στοίβα (με τη χρήση της εντολής POP) των καταχωρητών που αποθηκεύτηκαν στην αρχή (με αντίστροφη σειρά από αυτή που αποθηκεύτηκαν). Ακόμη, ενεργοποιείται ο μηχανισμός ελέγχου διακοπών (με τη χρήση της εντολής EI) και ανακτάται η κατάσταση του μΕ με την επιστροφή του ελέγχου στην προηγούμενη κατάσταση.
Δηλαδή ο PC παίρνει ξανά την τιμή 0900H που εξάγεται από την στοίβα.
Τέλος, ενημερώνεται ο δείκτης στοίβας και γίνεται: (SP) <- 1FEE

5η ΑΣΚΗΣΗ: Να γραφεί πρόγραμμα Assembly (και η ρουτίνα εξυπηρέτησης) σε μΥ-Σ 8085 που να λαμβάνει 16 δεδομένα των 8 bit από μια συσκευή. Το καθένα μεταφέρεται σε 2 βήματα (πρώτα τα 4 MSB και μετά τα 4 LSB – συνολικά θα χρειαστούν 32 βήματα) μέσω των (X_0-X_3) της θύρας PORT_IN (20^H) ενώ τα υπόλοιπα MSbit της θύρας (X_4-X_7) δεν χρησιμοποιούνται. Η συσκευή για κάθε 4 bit που αποστέλλει, προκαλεί πριν διακοπή RST5.5. Να υπολογιστεί ο μέσος όρος των 16 δεδομένων με ακρίβεια 8 bit.



Το πρόγραμμα μαζί με τα απαραίτητα σχόλια παρατίθενται ακολούθως:

```
START:
        MVI A,0EH       ; Ενεργοποίηση διακοπής RST5.5
        SIM
        MVI C,20H       ; counter για τις 32 διακοπές που πρόκειται να γίνουν
        MVI B,01H       ; flag για το αν έχουμε είσοδο των LSB ή των MSB
        LXI H,0000H     ; accumulator του αθροίσματος των αριθμών
        EI              ; Ενεργοποίηση διακοπών

WAIT:                   ; Βρόχος αναμονής
        MOV A,C         ; Εάν δεν έχουν γίνει 32 διακοπές, δηλαδή δεν έχει
        CPI 00H         ; ολοκληρωθεί η λήψη των 16 δεδομένων από τη συσκευή
        JNZ WAIT        ; περίμενε και άλλη διακοπή
        DI              ; Αλλιώς, απενεργοποίησε το μηχανισμό ελέγχου διακοπών
        DAD H           ; Ολίσθηση αριστερά κατά 4 θέσεις που ισούται με ακέραια
        DAD H           ; διαίρεση με το 2^4=16 και αποθήκευση αποτελέσματος
        DAD H           ; στον καταχωρητή H. Το κλασματικό μέρος βρίσκεται στον L
        DAD H
        HLT

RST 5.5:                ; Έχουμε αποστολή 4 bits από τη συσκευή
        PUSH PSW        ; Αποθήκευση στη στοίβα της κατάστασης του μΕ,
        IN PORT_IN      ; Λήψη 8bitou αριθμού από τον μΕ
        ANI F0H         ; Μάσκα έτσι ώστε να γίνουν μηδενιστούν τα bits 0-3
        MOV E,A
        MOV A,B         ; Ενημέρωση flag για το
        XRA 01H         ; εαν έχουμε είσοδο των
        MOV B,A         ; LSB ή MSB
        CPI 01H         ;Εάν έχουμε τα 4 MSB
        MOV A,E
        JZ LSB_PART ; κάνε άλμα
        MOV D,A         ; Αλλιώς, αποθήκευσε τα 4 LSB του A στον D
        JMP END
```

```
LSB_PART:
        RRC         ; Κάνε δεξιές ολισθήσεις ώστε
        RRC         ; τα 4 LSB να γίνουν
        RRC         ; 4 MSB
        RRC
        ORA D       ; 0000XXXX OR XXXX0000 και αποθήκευσε στον A τον 8bit αριθμό
        MVI D,00H   ; που αποθηκεύεται στα 8 LSB του ζεύγους D-E
        MOV E,A     ; ενώ τα 8 MSB παίρνουν την τιμή 0
        DAD D       ; πρόσθεσε το ζεύγος D-E στο ζεύγος H-L

END:
        DCR C       ; Ενημέρωσε τον μετρητή ότι έγινε μία διακοπή
        POP PSW     ; Απέκτησε από τη στοίβα την κατάσταση του μE
        EI          ; Ενεργοποίησε τις διακοπές
        RET
```