Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή: Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Συστήματα Μικροϋπολογιστών (6ο εξάμηνο)
5η Ομάδα Ασκήσεων

Δημήτριος Καλαθάς - el18016
Δημήτριος Καλέμης - el18152

Ασκήσεις Προσομοίωσης emu8086

Άσκηση 1

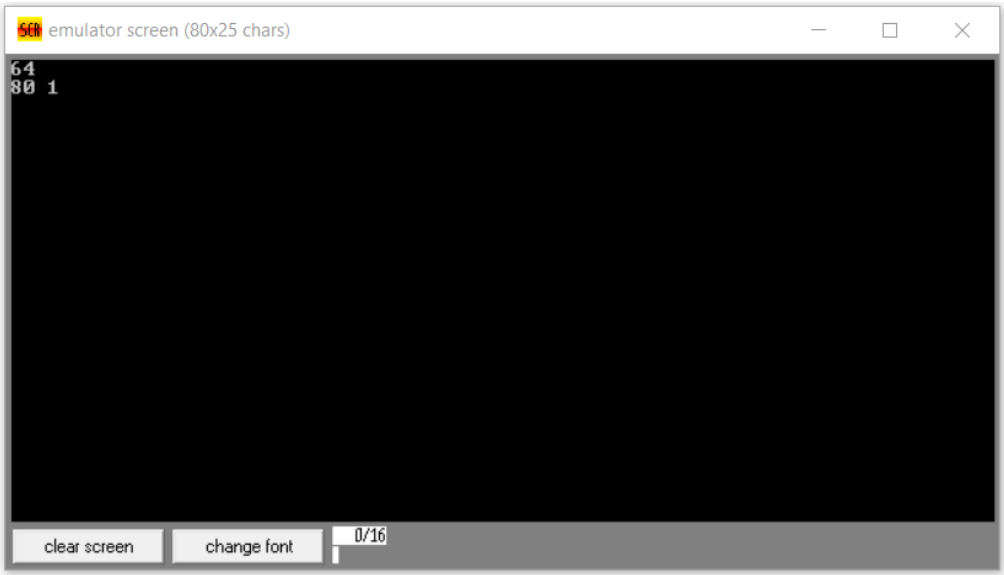
Να δοθεί πρόγραμμα που να αποθηκεύει τους αριθμούς 128, 127, 126, …, 2, 1 με τη σειρά αυτή, σε διαδοχικές θέσεις της μνήμης (τύπου byte) αρχίζοντας από την θέση TABLE. Στη συνέχεια το πρόγραμμα να συμπληρωθεί με τους εξής δύο (2) υπολογισμούς και να τυπώνει τα αποτελέσματα σε 2 γραμμές στην οθόνη:

α. Το ακέραιο μέρος (στρογγυλευμένο στα 16 bit) του μέσου όρου των περιττών αριθμών (64) από τα 128 δεδομένα σε δεκαδική μορφή.

β. Το μέγιστο και τον ελάχιστο σε μέγεθος από το παραπάνω σύνολο δεδομένων. Τα δύο αυτά αποτελέσματα να τυπωθούν με ένα κενό μεταξύ τους σε δεκαεξαδική μορφή. Ο αλγόριθμος να αναζητάει ταυτόχρονα και τους δύο αριθμούς σε ένα βρόχο αναζήτησης.

Λύση

Με τη χρήση του προσομοιωτή παίρνουμε το εξής αποτέλεσμα:



Δηλαδή έχουμε το μέσο όρο των αριθμών σε δεκαδική μορφή = 64 καθώς και το μέγιστο και τον ελάχιστο αριθμό του πίνακα σε δεκαεξαδική μορφή.

Κώδικας σε ASSEMBLY

```
INCLUDE macros.asm


DATA SEGMENT
    TABLE DB 128 DUP(?)          ;sunolo dedomenon
    TWO DB DUP(2)                ;elegxos isotimias
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA
    MAIN PROC FAR
            MOV AX,DATA
            MOV DS,AX            ;save numbers in memory
            MOV DI,0             ;pointer of number table
            MOV CX,128           ;all the numbers
        STORE:
            MOV TABLE[DI],CL
            INC DI
            LOOP STORE           ;sum and count
            MOV DH,0             ;sum AX+DL
            MOV AX,0             ;sum odd
            MOV BX,0             ;oloi oi odd
            MOV DI,0
            MOV CX,128
        FINDADDODD:
            PUSH AX
            MOV AH,0             ;AX/2
            MOV AL,TABLE[DI]     ;elegxos isotimias
            DIV TWO
            CMP AH,0
            POP AX
            JE SKIPEVEN          ;AX div 2 = 0 ?
            MOV DL,TABLE[DI]     ;save
            ADD AX,DX            ;sum
            INC BX               ;odd
        SKIPEVEN:                ;even
            INC DI
            LOOP FINDADDODD      ;count  m.o
            MOV DX,0             ;AX/BX
            DIV BX               ;sum/total
                                 ;prin m.o
            MOV BL,10
            MOV CL,AL
            DIV BL
            ADD AL,30H
            PRINTCH AL
            SUB AL,30H
            MUL BL
            SUB CL,AL
            ADD CL,30H
            PRINTCH CL

            PRINTLN
                                 ;check max,min
            MOV AL,TABLE[0]      ;first max
            MOV BL,TABLE[127]    ;first min
            MOV DI,0
            MOV CX,128
        MAXMIN:
            CMP AL,TABLE[DI]     ;check max
            JC NEWMAX
            JMP TOMIN
        NEWMAX:                  ;new max
            MOV AL,TABLE[DI]
            JMP NEXTNUM
        TOMIN:                   ;check min
            CMP TABLE[DI],BL
            JC NEWMIN
            JMP NEXTNUM
        NEWMIN:                  ;new min
            MOV BL,TABLE[DI]
        NEXTNUM:
            INC DI
            LOOP MAXMIN          ;print max,min
            CALL PRINT_NUM8_HEX      ;print max
            PRINTCH ' '
            MOV AL,BL
            CALL PRINT_NUM8_HEX      ;pirnt min
            EXIT
    MAIN ENDP
                                 ;print hex  AL
```

```
82
83      PRINT_NUM8_HEX PROC NEAR        ;80x86_programs.pdf selida 17
84              MOV DL,AL
85              AND DL,0F0H             ;1o digit hex
86              MOV CL,4
87              ROR DL,CL
88              CMP DL,0
89              JE SKIPZERO
90              CALL PRINT_HEX
91          SKIPZERO:
92              MOV DL,AL
93              AND DL,0FH              ;2o digit hex
94              CALL PRINT_HEX
95              RET
96      PRINT_NUM8_HEX ENDP
97
98                                      ;print hex DL
99
100     PRINT_HEX PROC NEAR             ;80x86_programs.pdf selida 18
101             CMP DL,9                ;0...9
102             JG LETTER
103             ADD DL,48
104             JMP SHOW
105         LETTER:
106             ADD DL,55               ;A...F
107         SHOW:
108             PRINTCH DL
109             RET
110     PRINT_HEX ENDP
111
112
113 CODE ENDS
114 END MAIN
```

Άσκηση 2

Σε ένα προσωπικό υπολογιστή, που βασίζεται στον μΕ 80x86, να γραφεί πρόγραμμα Assembly με τις παρακάτω προδιαγραφές:

1. Να δέχεται δυο (2) διψήφιους δεκαδικούς αριθμούς: Z και W από το πληκτρολόγιο (0-9), τους οποίους να τυπώνει στην οθόνη, όπως φαίνεται παρακάτω:
Z=28 W=39

2. Στη συνέχεια, μόλις συμπληρωθούν οι δύο αριθμοί (4 έγκυρα δεκαδικά ψηφία) να υπολογίζει το άθροισμα και τη διαφορά τους και να τυπώνει τα αποτελέσματα στην επόμενη γραμμή της οθόνης σε δεκαεξαδική μορφή, όπως φαίνεται παρακάτω:
Z+W=43 Z-W=-B

Το πρόγραμμα να είναι συνεχούς λειτουργίας

Λύση

Με τη χρήση του προσομοιωτή παίρνουμε το εξής αποτέλεσμα για διάφορες εισόδους:

Κώδικας σε ASSEMBLY

```
INCLUDE macros.asm

DATA SEGMENT

    MSGZ DB "Z=$"
    MSGW DB "W=$"
    MSGSUM DB "Z+W=$"
    MSGSUB DB "Z-W=$"
    MSGMINUS DB "Z-W=-$"
    Z DB 0
    W DB 0
    TEN DB DUP(10)              ;gia tis dekades
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA

MAIN PROC FAR

        MOV AX,DATA
        MOV DS,AX
    START:
        PRINTSTR MSGZ           ;kataskebei,emfanisi,apothikeusi tou z
        CALL READ_DEC_DIGIT     ;1o digit dekades
        MUL TEN
        LEA DI,Z                ;save 1o digit
        MOV [DI],AL
        CALL READ_DEC_DIGIT     ;2o digit monades
        ADD [DI],AL             ;save 2o digit
        PRINTCH ' '
        PRINTSTR MSGW           ;kataskebei,emfanisi,apothikeusi tou w
        CALL READ_DEC_DIGIT     ;1o digit dekades
        MUL TEN
        LEA DI,W                ;save 1o digit
        MOV [DI],AL
        CALL READ_DEC_DIGIT     ;2o digit monades
        ADD [DI],AL             ;save 2o digit
        PRINTLN                 ;sum
        MOV AL,[DI]             ;w
        LEA DI,Z                ;z
        ADD AL,[DI]             ;sum
        PRINTSTR MSGSUM
        CALL PRINT_NUM8_HEX     ;print sum
        PRINTCH ' '
        MOV AL,[DI]             ;difference
        LEA DI,W                ;z
        MOV BL,[DI]             ;w
        CMP AL,BL               ;z>w or w>z
        JB MINUS                ;difference for z>w
        SUB AL,BL
        PRINTSTR MSGSUB
        JMP SHOWSUB
    MINUS:
        SUB BL,AL              ;difference for z<w
        MOV AL,BL
        PRINTSTR MSGMINUS
    SHOWSUB:
        CALL PRINT_NUM8_HEX    ;print difference
        PRINTLN
        PRINTLN
        JMP START
MAIN ENDP


READ_DEC_DIGIT PROC NEAR
    READ:
        READCH
        CMP AL,48               ;if<0
        JB READ
        CMP AL,57               ;if>9
        JA READ
        PRINTCH AL
        SUB AL,48               ;ascii
        RET
READ_DEC_DIGIT ENDP             ;print 8-bit to hex  AL


PRINT_NUM8_HEX PROC NEAR        ;selida 17 pdf 80x86_programming
        MOV DL,AL
        AND DL,0F0H             ;1o hex digit
        MOV CL,4
```

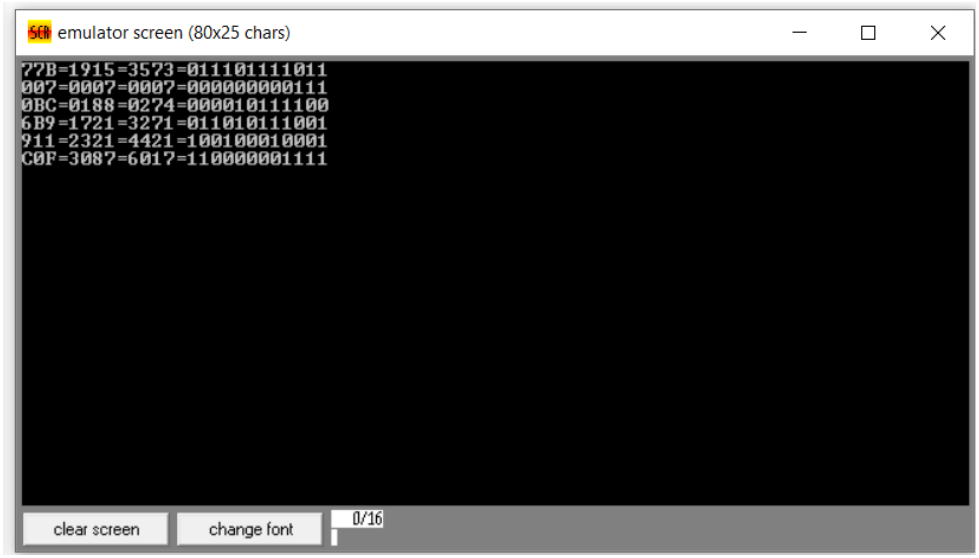
```
82          ROR DL,CL
83          CMP DL,0                ;exept zero
84          JE SKIPZERO
85          CALL PRINT_HEX
86      SKIPZERO:
87          MOV DL,AL
88          AND DL,0FH              ;2o hex digit
89          CALL PRINT_HEX
90          RET
91  PRINT_NUM8_HEX ENDP             ;print 8-bit to hex DL
92
93  PRINT_HEX PROC NEAR             ;selida 18 pdf 80x86_programming
94          CMP DL,9
95          JG LETTER
96          ADD DL,48
97          JMP SHOW
98      LETTER:                     ;A..F
99          ADD DL,55
100         SHOW:
101         PRINTCH DL
102         RET
103 PRINT_HEX ENDP
104
105 CODE ENDS
106 END MAIN
```

Άσκηση 3

Να γραφούν τρεις ρουτίνες PRINT_DEC, PRINT_OCT και PRINT_BIN που να δέχονται μέσω του BX έναν 12-bit αριθμό και να τον τυπώνουν στην οθόνη ενός προσωπικού υπολογιστή σε δεκαδική, οκταδική και δυαδική μορφή αντίστοιχα. Στη συνέχεια να γραφεί πρόγραμμα που να διαβάζει από το πληκτρολόγιο έναν τριψήφιο αριθμό σε δεκαεξαδική μορφή χρησιμοποιώντας την ρουτίνα HEX_KEYB (που διαβάζει κάθε φορά ένα δεκαεξαδικό ψηφίο) και μόλις συμπληρωθούν 3 έγκυρα ψηφία να τον τυπώνει σε δεκαεξαδική, δεκαδική, οκταδική και δυαδική μορφή με ένα χαρακτήρα '=' μεταξύ τους, κάνοντας χρήση των παραπάνω ρουτινών. Στη συνέχεια να αναμένει νέο διψήφιο δεκαεξαδικό αριθμό κλπ. Το πρόγραμμα να είναι συνεχούς λειτουργίας και να αγνοεί όλους τους υπόλοιπους χαρακτήρες πλην των δεκαεξαδικών και του χαρακτήρα 'Τ' με τον οποίον να τερματίζεται η λειτουργία.

Λύση

Με τη χρήση του προσομοιωτή παίρνουμε το εξής αποτέλεσμα για διάφορες εισόδους:



Κώδικας σε ASSEMBLY

```
INCLUDE macros.asm

CODE SEGMENT
    ASSUME CS:CODE

    MAIN PROC FAR
        START:
            CALL HEX_KEYB       ;insert first digit
            CMP AL,'T'          ;check if input = T
            JE FINISH           ;and if so terminate
            MOV BH,AL           ;store first element in 4 LSBs of BH
            CALL HEX_KEYB
            CMP AL,'T'
            JE FINISH
            MOV BL,AL           ;store second elemets
            ROL BL,4            ;and move it in the 4 MSBs of BL
            CALL HEX_KEYB
            CMP AL,'T'
            JE FINISH
            ADD BL,AL           ;store third element in 4 LSBs of BL


            PRINTCH '='
            CALL PRINT_DEC      ;print bin
            PRINTCH '='
            CALL PRINT_OCT      ;print oct
            PRINTCH '='
            CALL PRINT_BIN      ;print hex

            PRINTLN
            JMP START

        FINISH:
            EXIT
    MAIN ENDP


    HEX_KEYB PROC NEAR

            PUSH DX             ;store DX in stack
        IGNORE:
            READCH              ;read character from keyboard
            CMP AL,'T'          ;check if character is 'T'
            JE ADDR2
            CMP AL,30H          ;check if character is digit
            JL IGNORE           ;if not ignore it and read next
            CMP AL,39H
            JG ADDR1
            PUSH AX
            PRINTCH AL          ;print digit
            POP AX
            SUB AL,30H          ;get actual number from ascii code
            JMP ADDR2
        ADDR1:
            CMP AL,'A'
            JL IGNORE
            CMP AL,'F'
            JG IGNORE
            PUSH AX
            PRINTCH AL
            POP AX
            SUB AL,37H          ;conver into actual number from ascii code
        ADDR2:
            POP DX
            RET
    HEX_KEYB ENDP


    PRINT_DEC PROC NEAR
            PUSH AX             ;
            PUSH BX             ;store AX and BX in stack
            MOV AL,0
        LOOP1:
            CMP BX,1000         ;if BX < 1000
            JL LABEL1           ;go to LABEL1
            SUB BX,1000         ;else decrease BX by 1000
            INC AL              ;and increase AL (thousands) by 1
            JMP LOOP1           ;repeat until BX <1000
        LABEL1:
            ADD AL,30H          ;AL has thousands so by adding 30H we get
            PRINTCH AL          ;the ascii code for thousands
            MOV AL,0
        LOOP2:
            CMP BX,100          ;repeat the same for hundreds
```

```
            JL LABEL2
            SUB BX,100
            INC AL
            JMP LOOP2
        LABEL2:
            ADD AL,30H
            PRINTCH AL
            MOV AL,0
        LOOP3:
            CMP BX,10           ;repeat the same for tens
            JL LABEL3
            SUB BX,10
            INC AL
            jMP LOOP3
        LABEL3:
            ADD AL,30H          ;get tens in AL and convert to ascii
            PRINTCH AL          ;
            ADD BL,30H          ;get ones in BL and convert to ascii
            PRINTCH BL
            POP BX              ;
            POP AX              ;restore AX and BX
            RET
    PRINT_DEC ENDP

    PRINT_OCT PROC NEAR
            PUSH AX
            PUSH BX             ;store AX and BX in stack
            MOV AL,0
        LOOPA1:
            CMP BX,512          ;if BX < 512
            JL LABELA1          ;go to LABELA1
            SUB BX,512          ;decrease BX by 512
            INC AL              ;and increase by 1 register BX which counts the times of (8^3)
            JMP LOOPA1          ;repeat until BX < 512
        LABELA1:
            ADD AL,30H          ;get ascii number of number of (8^3)s
            PRINTCH AL
            MOV AL,0
        LOOPA2:
            CMP BX,64           ;repeat and count number of (8^2)s
            JL LABELA2
            SUB BX,64
            INC AL
            JMP LOOPA2
        LABELA2:
            ADD AL,30H
            PRINTCH AL
            MOV AL,0
        LOOPA3:
            CMP BX,8            ;repeat the same for (8^1)s
            JL LABELA3
            SUB BX,8
            INC AL
            jMP LOOPA3
        LABELA3:
            ADD AL,30H          ;get times (8^1) in AL and convert to ascii
            PRINTCH AL
            ADD BL,30H          ;get times of ones in BL and convert to ascii
            PRINTCH BL
            POP BX              ;restore BX and AX
            POP AX
            RET
    PRINT_OCT ENDP

    PRINT_BIN PROC NEAR
            PUSH AX             ;store AX and BX
            PUSH BX
            MOV AL,0
            ROL BX,4            ;first bit of our number is in first place of BX

            MOV CX,12           ;counter so we repeat for 12 times
        LOOPB1:
            SHL BX,1            ;get first digit of our number in carry
            MOV AL,0
            ADC AL,30H          ;add 30 to get the asccii code
            PRINTCH AL
            LOOP LOOPB1         ;repeat until CX = 0
            POP BX              ;restore AX and BX
            POP AX
            RET
    PRINT_BIN ENDP
CODE ENDS
END MAIN
```

Άσκηση 4

Σε ένα προσωπικό υπολογιστή, που βασίζεται στον μΕ 80x86, να γραφεί πρόγραμμα Assembly με τις παρακάτω προδιαγραφές:

1. Να αναμένει την πληκτρολόγηση 20 χαρακτήρων που να αποτελούνται από πεζούς αγγλικούς χαρακτήρες (a-z) και τους αριθμούς 0-9, τους οποίους να τυπώνει στην οθόνη, αγνοώντας κάθε άλλο χαρακτήρα.
2. Στη συνέχεια, με τη συμπλήρωση 20 έγκυρων χαρακτήρων ή πριν αν δοθεί ο χαρακτήρα ENTER, το πρόγραμμα να τυπώνει στην επόμενη γραμμή το ίδιο κείμενο με κεφαλαίους χαρακτήρες (A-Z), και τους αριθμούς 0-9 στο τέλος μετά από μια παύλα '-' διατηρώντας όμως τη σειρά με την οποίαν δόθηκαν. Το πρόγραμμα να είναι συνεχούς λειτουργίας και να τερματίζεται σε οποιοδήποτε σημείο με τον χαρακτήρα '='.

Λύση

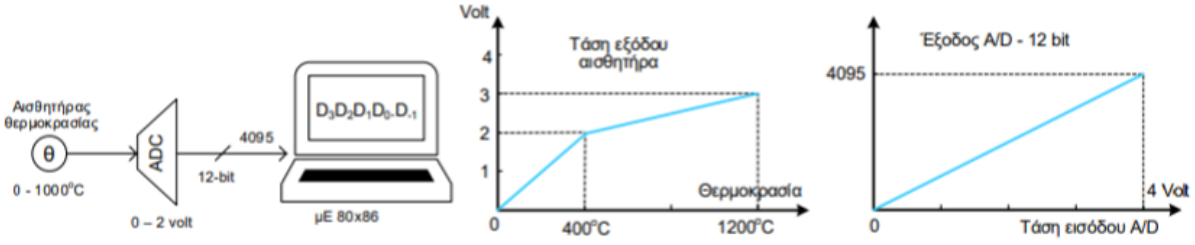
Με τη χρήση του προσομοιωτή παίρνουμε το εξής αποτέλεσμα για διάφορες εισόδους:

Κώδικας σε ASSEMBLY

```
INCLUDE macros.asm

DATA SEGMENT
    CHARS DB 20 DUP(?) ;save the char or number t
    DATA ENDS
CODE SEGMENT

    ASSUME CS:CODE, DS:DATA

MAIN PROC FAR
        MOV AX,DATA
        MOV DS,AX
        MOV CL,0      ;counter
    START:
        MOV DI,0      ;pointer table
    NEXTCHAR:
        READCH
        CMP AL,61    ;check for '='
        JE FINISH
        CMP AL,13    ;check for 'enter'
        JE CAPSLINE
        CMP AL,48    ; if <0;
        JB NEXTCHAR
        CMP AL,122    ;if >z
        JA NEXTCHAR
        CMP AL,57     ; if <=9
        JBE SAVECHAR
        CMP AL,97
        JB NEXTCHAR
    SAVECHAR:
        PRINTCH AL    ;print all
        MOV CHARS[DI],AL
        INC DI
        INC CL
        CMP CL,20     ;if they are 20
        JB NEXTCHAR
    CAPSLINE:
        PRINTLN
        MOV C1,20      ;check for empy table
        CMP CL,0
        JE  NEXTCHAR
        MOV CX,20
        MOV DI,0
        PRINT_LETTERS:          ;if its letter then print them first
        MOV AL,CHARS[DI]
        CMP AL,'a'              ;if <a then not letter
        JB  NOT_LETTER
        CMP AL,'z'              ;if >z then not letter
        JA NOT_LETTER
        SUB AL,32               ;low to capital
        PRINTCH AL              ;print letter first
        MOV CHARS[DI],' '       ;fill table with empty slots
    NOT_LETTER:
        INC DI
        LOOP PRINT_LETTERS       ;if not letter loop until chek all table
        PRINTCH '-'             ;print -
        MOV CX,20
        MOV DI,0
    PRINT_NUMS:                 ;cher for numbers
        MOV AL,CHARS[DI]
        CMP AL,30H              ; we check if it is an ASCII coded digit
        JL NOT_A_NUMBER
        CMP AL,39H
        JG NOT_A_NUMBER
        PRINTCH AL              ; print only the digits
        MOV CHARS[DI],' '       ;fill table with empty slots
    NOT_A_NUMBER:
        INC DI
        LOOP PRINT_NUMS          ;print them second
        PRINTLN
        PRINTLN
        JMP START
    FINISH:
    EXIT
MAIN ENDP

CODE ENDS
END MAIN
```

Άσκηση 5

Σε ένα προσωπικό υπολογιστή, που βασίζεται στον μΕ 80x86 και περιλαμβάνει σύστημα λήψης δεδομένων να γραφεί πρόγραμμα Assembly με τις παρακάτω προδιαγραφές:
Να παρακολουθεί και να απεικονίζει θερμοκρασίες από 0°C ως 1200,0°C στην οθόνη του PC, σε δεκαδική μορφή (το πολύ 4ων ακέραιων ψηφίων) και με ακρίβεια ενός κλασματικού δεκαδικού ψηφίου (για την ζητούμενη ακρίβεια επιλέξτε περικοπή όπως στο παράδειγμα που δόθηκε). Υποτίθεται ότι η θερμοκρασία λαμβάνεται μέσω μιας 16-bit θύρας εισόδου σε δυαδική μορφή των 12 bit. Η τάση που παρέχεται από τον αισθητήρα θερμοκρασίας έχει την χαρακτηριστική καμπύλη του παρακάτω σχήματος (Θερμοκρασία/ Τάση εξόδου) και ακολουθείται από ένα μετατροπέα από Αναλογική τιμή σε Ψηφιακή (ADC) των 12 bits (βλ. σχήμα με χαρακτηριστική Τάση εισόδου ADC/Έξοδος ADC). Για να προσομοιώσετε την θύρα εισόδου, τα δεδομένα της (12 bits) να τα εισάγετε μέσω του πληκτρολογίου σε μορφή 3ων HEX ψηφίων (να λαμβάνονται κάθε φορά τα 3 πρώτα έγκυρα). Το πρόγραμμα να αρχίζει με το μήνυμα “START(Y,N):” που εμφανίζεται μια φορά στην έναρξη του προγράμματος και ανάλογα με το χαρακτήρα που δίνεται να ξεκινάει ή να τερματίζεται. Μετά την εκκίνηση να αναμένει 3 HEX ψηφία όπως αναφέρθηκε και να είναι συνεχούς λειτουργίας δηλαδή να εμφανίζει νέα τιμή θερμοκρασίας σε κάθε νέα τριάδα HEX ψηφίων που δίνεται από το πληκτρολόγιο. Επίσης και στη φάση της λειτουργίας να τερματίζεται αν δοθεί οποιαδήποτε στιγμή ο χαρακτήρας N (χωρίς να εμφανίζεται το μήνυμα START(Y,N):). Για τιμές μεγαλύτερες από 1200°C να εμφανίζεται το μήνυμα σφάλματος “ERROR”.



Λύση

Με τη χρήση του προσομοιωτή παίρνουμε το εξής αποτέλεσμα για διάφορες εισόδους:

Οι συναρτήσεις των 2 κλάδων και ο τρόπος υπολογισμού των κλασματικών μερών φαίνονται παρακάτω:

$$\text{1ος κλάδος: } T = \frac{800V}{4095}$$
$$\text{2ος κλάδος: } T = \frac{3200V}{4095} - 1200$$
$$\text{κλασματικόςμέρος} = \frac{10 * \text{υπόλοιπο}}{4095}$$

όπου T η ζητούμενη θερμοκρασία και V η τάση εξόδου του ADC

Κώδικας σε ASSEMBLY

```
INCLUDE macros.asm

DATA SEGMENT
STARTPROMPT DB "START(Y,N):$"       ;starter message
ERRORMSG DB "ERROR$"                ;error message
ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA

MAIN PROC FAR
        MOV AX,DATA
        MOV DS,AX
        PRINTSTR STARTPROMPT
    START:                          ;starter char
        READCH
        CMP AL,'N'                  ;= N ?
        JE FINISH                   ;finish
        CMP AL,'Y'                  ;= Y ?
        JE CONT                     ;begin
        JMP START
    CONT:
        PRINTCH AL                  ;print starter char
        PRINTLN
        PRINTLN
    NEWTEMP:
        MOV DX,0
        MOV CX,3                     ;3 hex digit
    READTEMP:                        ;input
        CALL HEX_KEYB                ;put digit
        CMP AL,'N'                   ;check for N to end
        JE FINISH                    ;all digit in DX
        PUSH CX
        DEC CL                       ;rol
        ROL CL,2
        MOV AH,0
        ROL AX,CL                    ;rol left 8,4,0 digit
        OR DX,AX
        POP CX
        LOOP READTEMP
        PRINTTAB
        MOV AX,DX
        CMP AX,2047                 ;V<=2 ?
        JBE BRANCH1
        CMP AX,3071                 ;V<=3 ?
        JBE BRANCH2
        PRINTSTR ERRORMSG           ;V>3
        PRINTLN
        JMP NEWTEMP
    BRANCH1:                        ;1o: V<=2, T=(800*V) div 4095
        MOV BX,800
        MUL BX
        MOV BX,4095
        DIV BX
        JMP SHOWTEMP
    BRANCH2:                        ;2o: 2<V<=3, T=((3200*V) div 4095)-1200
        MOV BX,3200
        MUL BX
        MOV BX,4095
        DIV BX
        SUB AX,1200
    SHOWTEMP:
        CALL PRINT_DEC16            ;print integer number  (AX)
                                    ;klasma = (upolipo*10) div 4095
        MOV AX,DX
        MOV BX,10
        MUL BX
        MOV BX,4095
        DIV BX
        PRINTCH ','                 ;upodiastoli
        ADD AL,48                   ;ASCII
        PRINTCH AL                  ;print klasma
        PRINTLN
        JMP NEWTEMP
    FINISH:
        PRINTCH AL
        EXIT
MAIN ENDP

                                    ;hex (in AL)
```

```
HEX_KEYB PROC NEAR              ;80x86_programs.pdf selida 20-21
    READ:
        READCH
        CMP AL,'N'
        JE RETURN
        CMP AL,48               ;<0 ?
        JL READ
        CMP AL,57               ;>9 ?
        JG LETTER
        PRINTCH AL
        SUB AL,48               ; ASCII
        JMP RETURN
    LETTER:                     ;A...F
        CMP AL,'A'              ;<A ?
        JL READ
        CMP AL,'F'              ;>F ?
        JG READ
        PRINTCH AL
        SUB AL,55               ;ASCII
    RETURN:
        RET
HEX_KEYB ENDP

                                ;16 bit dec AX
PRINT_DEC16 PROC NEAR           ;80x86_programs.pdf selida 26-27
        PUSH DX
        MOV BX,10               ;div 10
        MOV CX,0                ;count
    GETDEC:                     ;output digit
        MOV DX,0                ;number mod 10 (upolipo)
        DIV BX                  ;div 10
        PUSH DX                 ;save
        INC CL
        CMP AX,0                ;number div 10 = 0 ? (piliko)
        JNE GETDEC
    PRINTDEC:                   ;print digit
        POP DX
        ADD DL,48               ; ASCII
        PRINTCH DL
        LOOP PRINTDEC
        POP DX
        RET
PRINT_DEC16 ENDP

CODE ENDS
END MAIN
```

Αρχείο macros.asm

```
PRINTCH MACRO CHAR
    PUSH AX
    PUSH DX
    MOV DL,CHAR
    MOV AH,2
    INT 21H
    POP DX
    POP AX
ENDM


PRINTSTR MACRO STRING
    PUSH AX
    PUSH DX
    MOV DX,OFFSET STRING
    MOV AH,9
    INT 21H
    POP DX
    POP AX
ENDM


PRINTLN MACRO
    PUSH AX
    PUSH DX
    MOV DL,13
    MOV AH,2
    INT 21H
    MOV DL,10
    MOV AH,2
    INT 21H
    POP DX
    POP AX
ENDM


PRINTTAB MACRO
    PUSH AX
    PUSH DX
    MOV DL,9
    MOV AH,2
    INT 21H
    POP DX
    POP AX
ENDM


READCH MACRO
    MOV AH,8
    INT 21H
ENDM


READNPRINTCH MACRO
    MOV AH,1
    INT 21H
ENDM


EXIT MACRO
    MOV AX,4C00H
    INT 21H
ENDM
```