Ε.Μ.Π. - ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧ. & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΫΠΟΛΟΓΙΣΤΩΝ & ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΑΘΗΝΑ 25. 6. 2021

ΓΡΑΠΤΗ ΕΞΕΤΑΣΗ ΣΤΟ ΜΑΘΗΜΑ "***Συστήματα Μικροϋπολογιστών***"

(ΘΕΜΑ 3ο – ΣΥΝΟΛΟ 2 Μονάδες)

Έναρξη 13:40' - ΔΙΑΡΚΕΙΑ 30' + 10' Παράδοση: 14:20'

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: ΓΙΩΡΓΟΣ ΚΥΡΙΑΚΟΠΟΥΛΟΣ - el18153

ΘΕΜΑ 3ο: (2 ΜΟΝΑΔΕΣ): Σε ένα προσωπικό υπολογιστή, να γραφεί πρόγραμμα σε Assembly μΕ 80x86 που να δέχεται από το πληκτρολόγιο τέσσερις (4) δεκαδικούς αριθμούς (D_3, D_2, D_1, D_0 με τη σειρά αυτή) για να αποτελέσουν ένα διψήφιο και δυο μονοψήφιους δεκαδικούς αριθμούς και να κάνει τον εξής υπολογισμό: $P=(D_3\times 10 + D_2)\times(D_1 + D_0)$. Το πρόγραμμα τυπώνει στην οθόνη τα μηνύματα εισόδου και τους εισαγόμενους αριθμούς. Όταν συμπληρωθούν 4 έγκυροι δεκαδικοί αριθμοί να αναμένει τον χαρακτήρα 'h' και μετά να τυπώνει το αποτέλεσμα σε δεκαεξαδική μορφή 3 ψηφίων αν είναι <400Hex, αλλιώς το μήνυμα yperx, αυστηρά όπως φαίνεται παρακάτω:

DOSE 1ο ARITHMO = *58*
DOSE 2ο ARITHMO = *7*
DOSE 3ο ARITHMO = *9*
APOTELESMA = *3A0* ή APOTELESMA = yperx

Να θεωρήσετε δεδομένες τις μακροεντολές (σελ. 361-2, 373) του βιβλίου και μπορείτε να κάνετε χρήση των ρουτινών DEC_KEYB και PRINT_HEX χωρίς να συμπεριλάβετε τον κώδικά τους. Για την διευκόλυνσή σας, δίνονται οι πρώτες εντολές που αποτελούν τον 'σκελετό' του ζητούμενου προγράμματος.

ΑΠΑΝΤΗΣΗ

```
INCLUDE     MACROS

DATA_SEG    SEGMENT
  MSG1      DB 0AH,0DH, 'DOSE 1ο ARITHMO =  $'
  MSG2      DB 0AH,0DH, 'DOSE 2ο ARITHMO = $'
  MSG3      DB 0AH,0DH, 'DOSE 3ο ARITHMO = $'
  MSG4      DB 0AH,0DH, 'APOTELESMA = $'
DATA_SEG    ENDS

CODE_SEG    SEGMENT
  ASSUME   CS:CODE_SEG, DS:DATA_SEG

MAIN  PROC  FAR
      MOV   AX, DATA_SEG
      MOV   DS, AX
. . .
```

```
INCLUDE     MACROS

DATA_SEG  SEGMENT
       MSG1 DB 0AH,0DH, 'DOSE 1ο ARITHMO = $'
       MSG2 DB 0AH,0DH, 'DOSE 2ο ARITHMO = $'
  MSG3      DB 0AH,0DH, 'DOSE 3ο ARITHMO = $'
  MSG4      DB 0AH,0DH, 'APOTELESMA = $'
  MSG5   DB 0AH,0DH, 'APOTELESMA = yperx'
DATA_SEG  ENDS
```

```
CODE_SEG  SEGMENT
        ASSUME   CS:CODE_SEG, DS:DATA_SEG

MAIN PROC FAR
   MOV        AX, DATA_SEG
   MOV        DS, AX

ADDR1:
   PRINT_STR MSG1
   CALL DEC_KEYB
   CMP AL, 'Q'
   JE QUITMAIN
   MOV BL, 10
   MUL BL          ; AL = D3*10
   MOV BL, AL      ; BL = D3*10

   CALL DEC_KEYB
   CMP AL, 'Q'
   JE QUITMAIN
   ADD AL, BL      ; AL = D3*10 + D2
   MOV BL, AL      ; BL = D3*10 + D2

   PRINT_STR MSG2
   CALL DEC_KEYB
   CMP AL, 'Q'
   JE QUITMAIN
   MOV CL, AL      ; CL = D1

   CALL DEC_KEYB
   CMP AL, 'Q'
   JE QUITMAIN
   ADD AL, CL      ; AL = D1 + D0

   MUL BL          ; AL = (D3*10 + D2) * (D1 + D0)

   CPM 0400H, AX
   JGE YPERX

CHECK:
   CALL WAIT_H   ; Wait for H
   CMP AL, 'H'
   JE ADDR2
   JMP CHECK

ADDR2:
   ROL AX, 1       ; 4 left rotate for msbs to become lsbs
   ROL AX, 1
   ROL AX, 1
   ROL AX, 1
   MOV DL, AL        ; keep 4 lsbs of dl that are the msbs of exit
   AND DL, 0FH
   PUSH AX
   PRINT_STR MSG4
   CALL PRINT_HEX
   POP AX
   LOOP ADDR2
   JMP ADDR1

YPERX:
   PRINT_STR MSG5
   JMP ADDR1
```

```
QUITMAIN:
    EXIT

    MAIN ENDP
    CODE_SEG ENDS
    END MAIN


WAIT_H PROC NEAR
    IGNORE:
    READ
    CMP AL, 'Q'
    JE QUIT
    CMP AL, 'H'
    JNE IGNORE
    QUIT:
    RET
WAIT_H ENDP
CODE_SEG ENDS
```