



# ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΜ&ΜΥ  
Εργαστήριο Μικροϋπολογιστών

3<sup>η</sup> Εργαστηριακή Άσκηση  
Ακ. έτος 2011-2012

*Ομάδα C07:*

Ελένη Ευαγγελάτου	A.M.: 03108050
Γρηγόρης Λύρας	A.M.: 03109687
Βασιλεία Φραγκιαδάκη	A.M.: 03108026

28 Δεκεμβρίου 2011

## Άσκηση (i)

Σε αυτή την άσκηση ζητείται να διαβάζεται από το πληκτρολόγιο ένας δυαδικός αριθμός των δέκα bits και στη συνέχεια να εκτυπώνεται ο αντίστοιχος δεκαδικός. Γι' αυτό το σκοπό, διαβάζουμε τον δυαδικό αριθμό ανά ψηφίο και τον αποθηκεύουμε στον διπλό καταχωρητή DX, με ένα loop που εκτελείται δέκα φορές και κάθε φορά ολισθαίνει κατά μία θέση προς τα αριστερά τον DX προσθέτοντας ένα μόνο εάν το ψηφίο που διάβασε είναι 1. Στη συνέχεια ο δυαδικός αριθμός μετατρέπεται στο δεκαδικό σύστημα με διαίρεση αρχικά με 1000 για την εύρεση των χιλιάδων, με διαίρεση του υπολοίπου από την προηγούμενη διαίρεση με 100 για την εύρεση των εκατοντάδων και διαίρεση του υπολοίπου με 10 για την εύρεση των δεκάδων. Το υπόλοιπο είναι οι μονάδες. Κάθε φορά εκτυπώνεται ο αντίστοιχος αριθμός (με τον αντίστοιχο χαρακτήρα ASCII).

Κυρίως κώδικας:

```
1  INCLUDE MACROS.TXT
2  ;=====DATA SEGMENT=====
3  DATA SEGMENT
4      MESSAGE1 DB "GIVE A 10-BINARY NUMBER: $"
5      MESSAGE2 DB "DECIMAL: $"
6      PKEY DB "PRESS ANY KEY TO START OR Q IN ORDER TO EXIT. $"
7      NEW_LINE DB 0AH, 0DH, '$' ;OI ASCII KWDIKOI GIA ALLAGH GRAMMHS
8  ENDS
9  ;=====STACK SEGMENT=====
10 STACK SEGMENT
11     DW 128 DUP(?)
12 ENDS
13
14 ;=====CODE SEGMENT=====
15 CODE SEGMENT
16     ASSUME CS:CODE, SS:STACK, DS:DATA, ES:DATA
17
18
19 MAIN PROC FAR
20
21 ; SET SEGMENT REGISTERS:
22     MOV AX, DATA
23     MOV DS, AX
24     MOV ES, AX
25
26 START:
27     PRINT_STRING MESSAGE1 ;PRINT_STRING ME XRHSH TOY MACRO
28     CALL BIN_KEYBOARD ;O XRHSTHS DINEI TON BINARY
29     PUSH DX
30     PRINT_STRING NEW_LINE ;ALLAGH GRAMMHS
31     PRINT_STRING MESSAGE2
32     POP DX
33     CALL DEC_CONVERSION ;TON METATREPW SE DEKADIKO
34     PRINT_STRING NEW_LINE ;ALLAGH GRAMMHS
35     PRINT_STRING PKEY ;ODHGIES PROS TON XRHSTH GIA TO TI NA PATHSEI
36     READ ;DIABAZEI AUTO POU EDWSE O XRHSTHS
37     CMP AL, 'Q' ;AN PATHSHKE TO Q
38     JE QUIT ;TELOS PROGRAMMATOS
39     CMP AL, 'q' ;AN PATHSHKE TO q
40     JE QUIT ;TELOS PROGRAMMATOS
41     PRINT_STRING NEW_LINE ;ALLAGH GRAMMHS
42     JMP START
43 QUIT:
44     MOV AL, 0H
45     EXIT ;APO TO MACRO
46
47
48 MAIN ENDP
49 ;=====DIABASMA TOY BINARY APO TO KEYBOARD=====
50 BIN_KEYBOARD PROC NEAR
51     MOV DX, 0
52     MOV CX, 10 ;O CX EINAI DEFAULT COUNTER GIA LOOPS. SELW 10 NOUMERA NA DIABASW
53
54 IGNORE:
55     READ ;DIABAZEI KARAKTHRA APO PLHKTROLOGIO XWRIS NA TO TUPWSEI
56     CMP AL, 'Q' ;BLEPW AN EINAI Q
57     JE QUIT ;AN EINAI TOTE KANOUME EXIT
58     CMP AL, 'q' ;BLEPW AN EINAI q
59     JE QUIT ;AN EINAI TOTE KANOUME EXIT
```

```

60     SHL DX,1
61     CMP AL, '0' ;ALLIWS,BLEPW AN EINAI 0
62     JE ZERO
63     CMP AL, '1' ;ALLIWS,BLEPW AN EINAI 1
64     JE ONE
65     JMP IGNORE
66 ONE:
67     INC DX ;GIA NA DIABASW 10 ARI8MOUS...
68 ZERO:
69     LOOP IGNORE
70 ADDR2:
71     RET
72
73 BIN_KEYBOARD ENDP
74
75 ;=====METATROPH K PRINT SE DEKADIKO=====
76 DEC_CONVERSION PROC NEAR
77     MOV AX,DX
78     MOV DX,0
79     ;0 DX::AX EINAI 0 DEFAULT DIAIRETEOS
80     MOV BX, 1000
81     DIV BX ; DIAIRW ME 1000
82     PRINT_NUM AL ; TO AL EXEI TO PHLIKO DHLADH THN XILIADA
83     MOV AX,DX ; DIAIRETHS 8A GINEI TO PROHGOUMENO UPOLOIPO
84     MOV DX,0
85     MOV BX, 100 ; DIAIRW ME 100
86     DIV BL
87     PRINT_NUM AL ; TO AL 8A EXEI TO PHLIKO POU 8A NAI OI EKATONTADES
88     MOV AL,AH ; DIAIRETHS 8A GINEI TO PROHGOUMENO UPOLOIPO
89     MOV AH,0
90     MOV BX, 10 ; DIAIRW ME 10
91     DIV BL
92     PRINT_NUM AL ; TO PHLIKO EDW EXEI TIS DEKADES
93     PRINT_NUM AH ; TO UPOLOIPO EDW EXEI TIS MONADES
94
95     RET
96
97 DEC_CONVERSION ENDP
98 ;=====
99 CODE ENDS
100
101 END MAIN

```

Τα macros που χρησιμοποιήσαμε:

```

1 ;This macro change registers AH,AL
2 READ MACRO
3     MOV AH,1
4     INT 21H
5 ENDM
6
7 ;This macro changes registers AH,DL
8 PRINT_MACRO CHAR
9     PUSH AX
10    PUSH DX
11    MOV DL,CHAR
12    MOV AH,02H
13    INT 21H
14    POP DX
15    POP AX
16 ENDM
17
18 ;This macro change registers AH,DX
19 PRINT_STRING MACRO STRING
20     PUSH AX
21     PUSH DX
22     MOV DX,OFFSET STRING ;Assume that string is a variable or constant, NOT an address
23     MOV AH,09H
24     INT 21H
25     POP DX
26     POP AX
27 ENDM
28
29 PRINT_NUM MACRO CHAR
30     PUSH DX
31     PUSH AX

```

```

32     MOV DL, CHAR
33     ADD DL, 30H
34     MOV AH, 2
35     INT 21H
36     POP AX
37     POP DX
38 ENDM
39
40 PAUSE MACRO
41     PUSH AX
42     PUSH DX
43     LEA DX,PKEY           ;<=>MOV DX, OFFSET PKEY;GIVES THE OFFSET OF PKEY TO DX
44     MOV AH,9
45     INT 21H               ;OUTPUT STRING AT DS:DX
46     MOV AH,8              ;WAIT FOR PRESSING OF A KEY
47     INT 21H               ;WITHOUT ECHO->8
48     PRINT OAH
49     PRINT ODH
50     POP DX
51     POP AX
52 ENDM
53
54 EXIT MACRO
55     MOV AH,4CH
56     INT 21H
57 ENDM

```

## Άσκηση (ii)

Στην άσκηση αυτή ζητείται τυπώνοντας τα κατάλληλα μηνύματα να διαβάζουμε ένα δεκαδικό αριθμό τεσσάρων ψηφίων, να τον εκτυπώνουμε και αν πατηθεί <enter> να τον μετατρέπουμε στον αντίστοιχο δεκαεξαδικό τον οποίο και να εκτυπώνουμε . Για το σκοπό αυτό, διαβάζουμε τον αριθμό ανά ψηφίο και τον αποθηκεύουμε τελικά στον καταχωρητή DP, με ένα loop που εκτελείται τέσσερις φορές (μία για κάθε ψηφίο), όπου δημιουργούμε τον αριθμό πολλαπλασιάζοντας κάθε φορά το προηγούμενο αποτέλεσμα επί 10 και προσθέτοντας το τρέχον ψηφίο. Στη συνέχεια, αφού πατηθεί <enter> τυπώνεται ο αντίστοιχος δεκαεξαδικός, τον οποίο υπολογίζουμε απομονώνοντας κάθε φορά στον καταχωρητή BL τα εκάστοτε τέσσερα ψηφία τα οποία αφού μετατρέψουμε στον αντίστοιχο δεκαεξαδικό χαρακτήρα τον εκτυπώνουμε στην οθόνη. Ξεκινάμε από τα τέσσερα MSB και προχωράμε προς τα τέσσερα LSB του BP. Με το χαρακτήρα “Q” ή “q” το πρόγραμμα τερματίζεται.

Κυρίως κώδικας:

```

1  INCLUDE MACROS.TXT
2
3  DATA SEGMENT
4      ; ADD YOUR DATA HERE!
5      PKEY DB "INSERT 4 DECIMAL NUMS AND THEN <ENTER>...$"
6      MESSAGE1 DB "GIVE FOUR NUMBERS: $"
7      MESSAGE2 DB "HEX = $"
8      NEW_LINE DB OAH, ODH, '$' ;OI ASCII KWDIKOI GIA ALLAGH GRAMMHS
9  ENDS
10
11 STACK SEGMENT
12     DW 128 DUP(?)
13 ENDS
14
15 CODE SEGMENT
16
17 MAIN PROC FAR
18 ; SET SEGMENT REGISTERS:
19     MOV AX, DATA
20     MOV DS, AX
21     MOV ES, AX
22
23 START:
24
25     PRINT_STRING MESSAGE1
26     CALL DEC_KEYBOARD ; KATEUSEIAN ME TO READ TA BAZEIS STON BP A8ROIZONTAS
27     PRINT_STRING NEW_LINE
28 BCK:
29     READ
30     CMP AL,ODH ;koita gia enter
31     JE CNT
32     CMP AL,'Q'
33     JE QUIT

```

```

34     CMP AL,'q'
35     JE QUIT
36     JMP BCK
37
38 CNT:
39     PRINT_STRING MESSAGE2
40     ;TUPWSE TA HEX TOU 16BITOU BP
41     CALL DIGITS_TO_HEX5
42     PRINT_STRING NEW_LINE
43     JMP START
44
45
46 QUIT:
47     MOV AL,0H
48     EXIT
49
50 MAIN ENDP
51
52
53 ;=====PROCEDURES=====
54 DEC_KEYBOARD PROC NEAR
55     MOV DX, 0
56     MOV CX, 4 ;GIATI 8A DEXTW 4 ARI8MOUS
57
58 IGNORE:
59     READ
60     CMP AL,'Q'
61     JE QUIT
62     CMP AL,'q'
63     JE QUIT
64     CMP AL,'0'
65     JL IGNORE
66     CMP AL,'9'
67     JG IGNORE
68     SUB AL, 30H
69     MOV BL,AL ;APO8UKEUSE TO TREXON STON BL
70     MOV BH,0
71     MOV AX,DX ; FORTWSE TO PROHGOUMENO A8ROISMA APO DX
72     MOV DX,10 ;BALE STO DL 10
73     MUL DX ;AX=AX*10
74     ADD AX,BX ;+BL
75     MOV DX,AX ;KAI BALTO PALI STO DX
76     LOOP IGNORE
77     MOV BP,DX
78     RET
79
80 DEC_KEYBOARD ENDP
81
82
83
84 ;=====MAKE 16 BITS TO HEX=====
85
86 DIGITS_TO_HEX5 PROC NEAR
87     ; KSEKINAME APO TA MSB KAI TYPWNOYME HEX ANA 4DES
88     MOV BX, BP
89     MOV BL, BH ;APOMONWNW TA 4 MSB
90     SHR BL, 4 ;OLIS8HSE TA STIS 4 LEAST SIGNIF 8ESEIS
91     CALL PRINT_HEX
92     MOV BX, BP
93     MOV BL,BH
94     AND BL, 0FH
95     CALL PRINT_HEX
96     MOV BX, BP
97     AND BL, 0F0H
98     SHR BL, 4
99     CALL PRINT_HEX
100    MOV BX, BP
101    AND BL, 0FH
102    CALL PRINT_HEX
103
104
105    RET
106 DIGITS_TO_HEX5 ENDP
107
108 PRINT_HEX PROC NEAR

```

```

109     CMP BL,9 ;AN O ARI8MOS EINAI METAKSU O K 9 PROS8ETW 30H
110     JG ADDR1
111     ADD BL, 30H
112     JMP ADDR2
113
114 ADDR1:
115     ADD BL, 37H ;DIAFORETIKA PROS8ETW 37H ('A' = 41H)
116 ADDR2:
117     PRINT BL
118     RET
119
120 PRINT_HEX ENDP
121 ;=====END OF MAKE 16 BITS TO HEX=====
122
123
124 END MAIN

```

Τα macros που χρησιμοποιήσαμε:

```

1 ;This macro change registers AH,AL
2 READ MACRO
3     MOV AH,1
4     INT 21H
5 ENDM
6
7 ;This macro changes registers AH,DL
8 PRINT_MACRO CHAR
9     PUSH AX
10    PUSH DX
11    MOV DL,CHAR
12    MOV AH,02H
13    INT 21H
14    POP DX
15    POP AX
16 ENDM
17
18 ;This macro change registers AH,DX
19 PRINT_STRING MACRO STRING
20     PUSH AX
21     PUSH DX
22     MOV DX,OFFSET STRING ;Assume that string is a variable or constant, NOT an address
23     MOV AH,09H
24     INT 21H
25     POP DX
26     POP AX
27 ENDM
28
29 PRINT_NUM MACRO CHAR
30     MOV DL, CHAR
31     ADD DL, 30H
32     MOV AH, 2
33     INT 21H
34 ENDM
35
36 PAUSE MACRO
37     PUSH AX
38     PUSH DX
39     LEA DX,PKEY ;<=>MOV DX, OFFSET PKEY;GIVES THE OFFSET OF PKEY TO DX
40     MOV AH,9
41     INT 21H ;OUTPUT STRING AT DS:DX
42     MOV AH,8 ;WAIT FOR PRESSING OF A KEY
43     INT 21H ;WITHOUT ECHO->8
44     PRINT OAH
45     PRINT ODH
46     POP DX
47     POP AX
48 ENDM
49
50 EXIT MACRO
51     MOV AH,4CH
52     INT 21H
53 ENDM
54
55 GETHON MACRO R
56     CALL GETHEX
57     MOV R,AX

```

```

58     CALL GETHEX
59     SHL R,4
60     OR R,AX
61     CALL GETHEX
62     SHL R,4
63     OR R,AX
64     CALL GETHEX
65     SHL R,4
66     OR R,AX
67     ENDM

```

## Άσκηση (iii)

Εδώ ζητείται να διαβάζουμε το πολύ 20 χαρακτήρες από το πληκτρολόγιο και μετά το πατημα του <enter>, να τους εκτυπώνουμε στην έξοδο ομαδοποιημένους κατά αριθμούς, μικρά και κεφαλαία γράμματα αγνοώντας τα κενά. Έτσι, δεσμεύουμε χώρο 21 byte για κάθε μία από τις τρεις "ομάδες", που αρχικοποιούμε στο χαρακτήρα τερματισμού string "\$" (το 21ο byte χρησιμεύει για να έχουμε χαρακτήρα τερματισμού στην περίπτωση που διαβάσουμε 20 χαρακτήρες ίδιας ομάδας). Στη συνέχεια διαβάζουμε τους χαρακτήρες αγνοώντας τους μη επιθυμητούς, μέσα σε ένα loop 20 επαναλήψεων, το οποίο σταματάει νωρίτερα σε περίπτωση που δεχτεί <enter>, και κάθε έναν τον αποθηκεύουμε στον αντίστοιχο πίνακα, αυξάνοντας κάθε φορά μία μεταβλητή που χρησιμεύει σαν δείκτης σ' αυτό το χώρο. Στη συνέχεια εκτυπώνουμε κάθε πίνακα διαδοχικά χρησιμοποιώντας τη μακροεντολή print\_string. Το πρόγραμμα τερματίζεται αν δοθεί ο χαρακτήρας "/".

Κυρίως κώδικας:

```

1     INCLUDE MACROS.TXT
2
3     STACK_SEG SEGMENT STACK
4         DW 128 DUP(?)
5     ENDS
6
7
8     DATA_SEG SEGMENT
9         MSG DB "GIMME <=20 CHARS END PRESS RETURN '/' TO QUIT",0AH,0DH,"$"
10        MSG2 DB " => $"
11        SPACE DB " "
12        LINE DB 0AH,0DH,"$"
13        NUMS DB 21 DUP("$")
14        NCNT DW 0
15        LOWC DB 21 DUP("$")
16        LCNT DW 0
17        UPRC DB 21 DUP("$")
18        UCNT DW 0
19
20
21     ENDS
22
23     CODE_SEG SEGMENT
24         ASSUME CS:CODE_SEG,SS:STACK_SEG,DS:DATA_SEG,ES:DATA_SEG
25
26     MAIN PROC FAR
27         ;FOR SEGMENT REGISTERS
28         MOV AX,DATA_SEG
29         MOV DS,AX
30         MOV ES,AX
31
32     START:
33         PRINT_STRING MSG
34         MOV DX,0
35         MOV BX,0
36         CALL GET_INPUT
37     CNT:
38         PRINT_STRING MSG2
39         PRINT_STRING NUMS
40         PRINT SPACE
41         PRINT_STRING LOWC
42         PRINT SPACE
43         PRINT_STRING UPRC
44         PRINT_STRING LINE
45         JMP START
46
47     EX:
48         MOV AL,0H
49         EXIT

```

```

50  MAIN ENDP
51
52
53  GET_INPUT PROC NEAR
54      MOV DX,0
55      MOV CX,20
56  READL:
57      READ
58      CMP AL,0DH
59      JE CNT
60      CMP AL,'/'
61      JE EX
62      CMP AL,30H ;0
63      JL READL
64      CMP AL,40H ;9+1
65      JL NUMBERS
66      CMP AL,41H ;A
67      JL READL
68      CMP AL,5BH ;Z+1
69      JL ULETTER
70      CMP AL,61H ;a
71      JL READL
72      CMP AL,7BH ;z+1
73      JL LLETTER
74      JMP READL
75  NUMBERS:
76      MOV BX,OFFSET NUMS
77      ADD BX,NCNT
78      MOV [BX],AL
79      INC NCNT
80      LOOP READL
81      RET
82  LLETTER:
83      MOV BX,OFFSET LOWC
84      ADD BX,LCNT
85      MOV [BX],AL
86      INC LCNT
87      LOOP READL
88      RET
89  ULETTER:
90      MOV BX,OFFSET UPRC
91      ADD BX,UCNT
92      MOV [BX],AL
93      INC UCNT
94      LOOP READL
95      RET
96  GET_INPUT ENDP
97
98
99  CODE_SEG ENDS
100
101  END MAIN

```

Τα macros που χρησιμοποιήσαμε:

```

1  ;This macro change registers AH,AL
2  READ MACRO
3      MOV AH,1
4      INT 21H
5  ENDM
6
7  ;This macro changes registers AH,DL
8  PRINT MACRO CHAR
9      PUSH AX
10     PUSH DX
11     MOV DL,CHAR
12     MOV AH,02H
13     INT 21H
14     POP DX
15     POP AX
16  ENDM
17
18 ;This macro change registers AH,DX
19 PRINT_STRING MACRO STRING
20     PUSH AX
21     PUSH DX

```



```

22     MOV DX,OFFSET STRING ;Assume that string is a variable or constant, NOT an address
23     MOV AH,09H
24     INT 21H
25     POP DX
26     POP AX
27 ENDM
28
29 PRINT_NUM MACRO CHAR
30     MOV DL, CHAR
31     ADD DL, 30H
32     MOV AH, 2
33     INT 21H
34 ENDM
35
36 PAUSE MACRO
37     PUSH AX
38     PUSH DX
39     LEA DX,PKEY           ;<=>MOV DX, OFFSET PKEY;GIVES THE OFFSET OF PKEY TO DX
40     MOV AH,9
41     INT 21H              ;OUTPUT STRING AT DS:DX
42     MOV AH,8             ;WAIT FOR PRESSING OF A KEY
43     INT 21H              ;WITHOUT ECHO->8
44     PRINT OAH
45     PRINT ODH
46     POP DX
47     POP AX
48 ENDM
49
50 EXIT MACRO
51     MOV AH,4CH
52     INT 21H
53 ENDM

```

## Άσκηση (iv)

Κυρίως κώδικας:

```

1  INCLUDE MACROS.TXT
2
3  STACK_SEG SEGMENT STACK
4      DW 128 DUP(?)
5  ENDS
6
7
8  DATA_SEG SEGMENT
9      FIRST DB "First number: $"
10     SECOND DB "Second number: $"
11     SPACE DB " "
12     LINE DB OAH,ODH,"$"
13
14
15 ENDS
16
17 CODE_SEG SEGMENT
18     ASSUME CS:CODE_SEG,SS:STACK_SEG,DS:DATA_SEG,ES:DATA_SEG
19
20 MAIN PROC FAR
21     MOV AX,DATA_SEG
22     MOV DS,AX
23     MOV ES,AX
24     CALL GET_INPUT
25     MOV AX,BX      ;AX=X0
26     MOV DX,0       ;DX=0
27     MUL SI         ;DX::AX = X0*Y0
28     MOV BP,AX
29     ; == 16 LSB in BP
30     PUSH BP        ; now pushed
31     PUSH CX        ; don't need X1 right now
32     MOV CX,DX      ; so use it as buffer
33     MOV AX,BX      ;AX=X0
34     MOV DX,0       ;DX=0
35     MUL DI         ;DX::AX = X0*Y1
36     ADD AX,CX      ;AX+=previous DX
37     JNC NOTOVF1
38     INC DX         ;if carry increase DX

```

```

39 NOTOVF1:
40     POP CX          ; will need X1
41     MOV BX,DX        ; X0 is no longer needed
42     PUSH BX         ; save DX
43     MOV BX,AX        ; save AX
44     MOV AX,CX        ; AX = X1
45     MOV DX,0         ; DX = 0
46     MUL SI          ; DX::AX = X1*Y0
47     ADD AX,BX        ; AX+=previous AX
48     JNC NOTOVF2     ; if carry increase DX
49     INC DX
50 NOTOVF2:
51     POP BX
52     MOV BP,AX        ;2ND DIGIT
53     PUSH BP
54     MOV AX,CX
55     MOV CX,DX        ; REALLY??
56     MOV DX,0
57     MUL DI
58     ADD AX,BX
59     JNC NOTOVF3
60     INC DX
61 NOTOVF3:
62     ADD AX,CX
63     JNC NOTOVF4
64     INC DX
65 NOTOVF4:
66     MOV BP,AX
67     PUSH BP
68     MOV BP,DX
69     ;now BP has the answer
70     CALL DIGITS_TO_HEXS
71     POP BP
72     CALL DIGITS_TO_HEXS
73     POP BP
74     CALL DIGITS_TO_HEXS
75     POP BP
76     CALL DIGITS_TO_HEXS
77
78
79     MOV AL,0H
80     EXIT
81 MAIN ENDP
82
83 GET_INPUT PROC NEAR
84     PRINT_STRING FIRST
85     GETON CX
86     GETON BX
87     PRINT_STRING LINE
88     PRINT_STRING SECOND
89     GETON DI
90     GETON SI
91     PRINT_STRING LINE
92     RET
93 GET_INPUT ENDP
94
95
96 GETHEX PROC NEAR
97 R:  READ
98     MOV AH,0
99     CMP AL,30H ;0
100    JL R
101    CMP AL,40H ;9+1
102    JL NUM
103    CMP AL,41H ;A
104    JL R
105    CMP AL,47H ;F+1
106    JL CAPS
107    CMP AL,61H ;a
108    JL R
109    CMP AL,67H ;f+1
110    JL SMALL
111    JMP R
112 NUM:
113     SUB AL,30H

```

```

114     RET
115 CAPS:
116     SUB AL,37H
117     RET
118 SMALL:
119     SUB AL,57H
120     RET
121
122 GETHEX ENDP
123
124
125 ;=====MAKE 16 BITS TO HEX=====
126
127 DIGITS_TO_HEXS PROC NEAR
128     MOV BX, BP
129     MOV BL, BH ;APOMONWNW TA 4 MSB
130     SHR BL, 4 ;OLIS8HSE TA STIS 4 LEAST SIGNIF 8ESEIS
131     CALL PRINT_HEX
132     MOV BX, BP
133     MOV BL,BH
134     AND BL, 0FH
135     CALL PRINT_HEX
136     MOV BX, BP
137     AND BL, 0F0H
138     SHR BL, 4
139     CALL PRINT_HEX
140     MOV BX, BP
141     AND BL, 0FH
142     CALL PRINT_HEX
143
144
145     RET
146 DIGITS_TO_HEXS ENDP
147
148 PRINT_HEX PROC NEAR
149     CMP BL,9 ;AN O ARI8MOS EINAI METAKSU 0 K 9 PROS8ETW 30H
150     JG ADDR1
151     ADD BL, 30H
152     JMP ADDR2
153
154 ADDR1:
155     ADD BL, 37H ;DIAFORETIKA PROS8ETW 37H ('A' = 41H)
156 ADDR2:
157     PRINT BL
158     RET
159
160 PRINT_HEX ENDP
161 ;=====END OF MAKE 16 BITS TO HEX=====
162
163
164
165
166
167
168
169 CODE_SEG ENDS
170
171 END MAIN

```

Τα macros που χρησιμοποιήσαμε:

```

1 ;This macro change registers AH,AL
2 READ MACRO
3     MOV AH,1
4     INT 21H
5 ENDM
6
7 ;This macro changes registers AH,DL
8 PRINT MACRO CHAR
9     PUSH AX
10    PUSH DX
11    MOV DL,CHAR
12    MOV AH,02H
13    INT 21H
14    POP DX
15    POP AX

```

```

16  ENDM
17
18  ;This macro change registers AH,DX
19  PRINT_STRING MACRO STRING
20      PUSH AX
21      PUSH DX
22      MOV DX,OFFSET STRING ;Assume that string is a variable or constant, NOT an address
23      MOV AH,09H
24      INT 21H
25      POP DX
26      POP AX
27  ENDM
28
29  PRINT_NUM MACRO CHAR
30      MOV DL, CHAR
31      ADD DL, 30H
32      MOV AH, 2
33      INT 21H
34  ENDM
35
36  PAUSE MACRO
37      PUSH AX
38      PUSH DX
39      LEA DX,PKEY          ;<=>MOV DX, OFFSET PKEY;GIVES THE OFFSET OF PKEY TO DX
40      MOV AH,9
41      INT 21H              ;OUTPUT STRING AT DS:DX
42      MOV AH,8              ;WAIT FOR PRESSING OF A KEY
43      INT 21H              ;WITHOUT ECHO->8
44      PRINT OAH
45      PRINT ODH
46      POP DX
47      POP AX
48  ENDM
49
50  EXIT MACRO
51      MOV AH,4CH
52      INT 21H
53  ENDM
54
55  GETHON MACRO R
56      CALL GETHEX
57      MOV R,AX
58      CALL GETHEX
59      SHL R,4
60      OR R,AX
61      CALL GETHEX
62      SHL R,4
63      OR R,AX
64      CALL GETHEX
65      SHL R,4
66      OR R,AX
67  ENDM

```