

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΡΟΗ Υ



Εργαστήριο Μικροπολογιστών

ΟΜΑΔΑ Δ06:

Βαβουλιώτης Γεώργιος (ΑΜ 03112083)

Αθανασίου Νικόλαος (ΑΜ 03112074)

Γιαννούλας Βασίλης (ΑΜ 03112117)

7ο Εξάμηνο

ΑΣΚΗΣΗ 1

Σε αυτή την άσκηση λαμβάνουμε ως είσοδο ένα δυαδικό αριθμό των 8bits και εμφανίζουμε τον αριθμό αυτό στο δεκαδικό σύστημα. Για την υλοποίηση της άσκησης χρησιμοποιείται ένας καταχωρητής-ο DX- τον οποίο μετά από κάθε αναγνώση ενός bit εισόδου τον ολισθαίνουμε προκειμένου μετά την ολοκλήρωση της εισαγωγής των δεδομένων να περιέχει όλη την είσοδο. Κατά την ανάγνωση των bits εισόδου γίνεται έλεγχος αν είναι έγκυρα και αν όχι αγνοούνται. Η ανάγνωση συνεχίζεται ώπου να ληφθούν 8 έγκυρα bits ή ώπου να πατηθεί το Q. Για τον προσδιορισμό της δεκαδικής αναπαράστασης του αριθμού με διαδοχικές αφαιρέσεις με 100 και 10 προσδιορίζονται σε 3 διαφορετικούς καταχωρητές οι αριθμοί των εκατοντάδων, των δεκάδων και των μονάδων (ό,τι περισσέψει μετά τις αφαιρέσεις είναι οι μονάδες). Τέλος, και στις δύο περιπτώσεις τυπώνονται τα περιεχόμενα των καταχωρητών των εκατοντάδων, των δεκάδων και των μονάδων. Ο κώδικας της άσκησης φαίνεται στη συνέχεια:

```
PRINT MACRO CHAR    ; makroentolh ektypwshs xarakthra
```

```
PUSH DX
```

```
PUSH AX
```

```
MOV DL,CHAR
```

```
MOV AH,2
```

```
INT 21H
```

```
POP AX
```

```
POP DX
```

```
ENDM
```

```
READ MACRO        ;makroentolh me thn opoia diabazoyme ena xarakthra apo to plhktrologio
```

```
MOV AH,8
```

```
INT 21H
```

```
ENDM
```

```
PRINT_STR MACRO STRING
```

```
MOV DX,OFFSET STRING
```

```
MOV AH,9
```

```
INT 21H
```

ENDM

EXIT MACRO ;macro exits back to DOS

MOV AX,4C00H

INT 21H

ENDM

DATA_SEG SEGMENT

NEW_LINE DB 0AH,0DH,'\$'

MSG1 DB 'GIVE AN 8-BIT BINARY NUMBER:\$'

MSG2 DB 'DECIMAL:\$'

DATA_SEG ENDS

CODE_SEG SEGMENT

ASSUME CS:CODE_SEG,DS:DATA_SEG

MAIN PROC FAR

MOV AX,DATA_SEG

MOV DS,AX

START:

MOV DX,0 ;mhdenizw ton dx opou 8a exei telika ton BCD ari8mo

MOV CX,8 ;metrhths pshfiwn

PUSH DX

PRINT_STR MSG1

POP DX

IGNORE:

```

READ

CMP AL,'Q'      ;an dwsame to 'q'na termatistei to programma
JE QUIT

CMP AL,30H      ; an einai to '0' to dexomai
JL IGNORE

CMP AL,31H      ; an einai to '1' to dexomai
JG IGNORE

JMP ADDR1


ADDR1:

PUSH DX

PRINT AL        ;typwnw to prwto bit pou diabasa apo to plhktrologio
POP DX

SUB AL,30H      ;metatroph tou ascii code se duadiko ari8mo
PUSH CX

SHL DX,1        ;ton kanw olis8hsh giati arxizw na diabazw apo to msb bit
ADD DX,AX

POP CX

LOOP IGNORE     ;diabazw ta 8 bit mexri o cx na ginei 0
PUSH DX

PRINT_STR NEW_LINE
PRINT_STR MSG2

POP DX

CONT:

MOV AL,0000H    ;EKATONTADES

MOV BL,0000H    ;DEKADES

                ;DX MONADES

```

CHECK1:

CMP DL,64H ; sygkrish me to 100

JAE PLUS_100

CHECK2:

CMP DL,0AH ; sygkrish me to 10

JAE PLUS_10

PRINT_OUTPUT:

ADD AL,30H ; metatroph dyadiko se ascii code

PRINT AL

ADD BL,30H ; metatroph bin se ascii code

PRINT BL

ADD DL,30H ; metatroph bin se ascii code

PRINT DL

PRINT_STR NEW_LINE

JMP START

QUIT:

EXIT

MAIN ENDP

PLUS_10:

INC BL

SUB DL,0AH

JMP CHECK2

PLUS_100:

INC AL

```
SUB DL,64H

JMP CHECK1

CODE_SEG ENDS

END MAIN
```

ΑΣΚΗΣΗ 2

```
EXIT    MACRO                                ;macro exits back to DOS

        MOV AX,4C00H

        INT 21H

ENDM
```

```
PRINT   MACRO CHAR                          ;macro to print a char

        PUSH DX

        PUSH AX

        MOV DL,CHAR

        MOV AH,2

        INT 21H

        POP AX

        POP DX

ENDM
```

```
READ    MACRO                                ;macro to read a char from keyboard

        MOV AH,8

        INT 21H

ENDM
```

PRINT_STRING MACRO STRING ;macro that prints a whole string

PUSH DX

PUSH AX

LEA DX,STRING

MOV AH,09H

INT 21H

POP AX

POP DX

ENDM

DATA_SEG SEGMENT

MSG1 DB "GIVE DECIMAL DIGITS: ", '\$'

MSG2 DB "HEX= ", '\$'

LINEFEED DB 0DH,0AH, '\$'

DATA_SEG ENDS

STACK_SEG SEGMENT STACK

DB 50 DUP(?) ;50 bytes for stack

STACK_SEG ENDS

CODE_SEG SEGMENT

ASSUME CS:CODE_SEG,DS:DATA_SEG,SS:STACK_SEG

PRINT_HEX PROC NEAR

```
CMP DL,9
JLE ADDR3
ADD DL,37H
JMP ADDR4
ADDR3:ADD DL,30H
ADDR4:PRINT DL
RET
PRINT_HEX ENDP
```

```
MAIN PROC FAR

    MOV AX,DATA_SEG

    MOV DS,AX                ;DS <- DATA_SEG

    MOV AX,DATA_SEG

    MOV SS,AX                ;SS <- STACK_SEG

    JMP START
```

TERMINATE_CHECK_FIRST_4: ;tsekarw efoson eimai stis 4 prwtes epanalhpseis na m er8ei h akolou8ia D15 opote kai termatizw. an den er8ei oi ari8moi pou isws dw8oun meta to D apo8hkeyontai kanonika san tous 4 prwtous

```
READ

MOV AH,0

CMP AL,'0'

JNE CHECK

PRINT AL

SUB AL,48D

CMP CX,4D

JNE SKIP
```


PRINT ','

SKIP:

PUSH AX

DEC CX

READ

MOV AH,0

CMP AL,'6'

JE TERMINATE

JMP CHECK

TERMINATE_CHECK_GENERAL:

READ

MOV AH,0

CMP AL,'0'

JNE CHECK2

PRINT AL

SUB AL,48D ;ftiakse ston AL ton teleytaio ari8mo pou diavases

MOV BH,BL

MOV BL,CH

MOV CH,CL

MOV CL,AL ;kanw update gia na exw ta swsta 4 teleytaia noumera gia ton ypologismo mou

READ

MOV AH,0

CMP AL,'6'

JE TERMINATE

JMP CHECK2

TERMINATE:

EXIT

START:

PRINT_STRING MSG1

MOV CX,4D ;Arxikopoiisi metriti sto 4 (4 toulaxiston)

MOV DX,0D ;o DX tha exei ton arithmo pou diavazetai (

MOV BX,0D ;tha mou xreiastei gia to swsimo tw'n arithmwn

READ_DEC:

READ ;diavase apo to pliktrologio

MOV AH,0 ;o kwdikos vrisketai sto AL kai etsi vazw sto

AH 0

CHECK:

CMP AL,'D' ;an diavases D checkare an prepei na
termatiseis to programma

JE TERMINATE_CHECK_FIRST_4

CMP AL,'0 ' ;an diavases ascii xaraktira me kwdiko < '0' ksanadiavase

JL READ_DEC

CMP AL,'9' ;an einai panw apo to '9' mh egkuro psifio, ksanadiavase

JG READ_DEC ;alliws synexise

ADDR1:

PRINT AL ;tupwse to xaraktira pou diavases

SUB AL,30H ;kai metetrepse ton ston antistoixο arithmo

CMP CX,4D ;afou diavasw ton prwto egkyro kai ton typwsw typwnw meta to ','
ka8ws se default mode 8a pairnw enan 4pshfio

JL READY ;epomenws ekei tha einai oi xiliades tou
PRINT ','

READY:

PUSH AX ;pusharw se ka8e lounpa ta stoixeia tou AX gia na krathsw meta apo ta 4
pshfia to ka8e ena ksexwrista kai na kanw update

LOOP READ_DEC ;gia na diavasw toulaxiston 4 pshfia

RESTORE_VALUES:

POP AX ;o AL exei thn timh tou 4ou pshfiou pou phra

MOV CL,AL ;o CL exei ton teleytaio ari8mo pou phra

POP AX ;O AL exei thn timh tou 3ou pshfiou pou phra

MOV CH,AL ;o CH exei ton proteleytaio ari8mo apo tous 4 prwtous pou phra (ton
3o dld)

POP AX ;o AL exei thn timh tou 2ou pshfiou ek tw'n 4 prwtwn pou mou
do8hke

MOV BL,AL ;o BL exei ton 2o ari8mo apo tous 4 prwtous pou mou do8hkan

POP AX ;o AL exei thn timh tou 1ou pshfiou ek tw'n 4 prwtwn pou mou
do8hke

MOV BH,AL ;o BH exei ton 1o ari8mo apo tous 4 prwtous pou mou do8hkan

;h diataksh ayth 8a parameinei

;dhladh o ari8mos pou 8a exw na typwsw sto telos otan path8ei enter
8a einai

;BH,BL,CH,CL

CONTINUE_READ:

READ ;diavase apo to pliktrologio

MOV AH,0

CHECK2:

CMP AL,0DH ;checkarw gia enter opote paw sthn eksodo

```

JE ENTER_PRESSED          ;o kwdikos vrisketai sto AL kai etsi vazw sto AH 0

CMP AL,'D'                 ;an diavases D checkare gia termatismo

JE TERMINATE_CHECK_GENERAL

CMP AL,'0'                 ;an diavases ascii xaraktira me kwdiko < '0'
ksanadiavase (lathos xaraktiras)

JL CONTINUE_READ

CMP AL,'9'                 ;an einai panw apo to 9 mh egkuro
psifio,ksanadiavase

JG CONTINUE_READ

```

UPDATE:

```

PRINT AL

SUB AL,48D                 ;ftiakse ston AL ton teleytaio ari8mo pou diavases

MOV BH,BL

MOV BL,CH

MOV CH,CL

MOV CL,AL                 ;kanw update gia na exw ta swsta 4 teleytaia noumera gia ton
ypologismo mou

JMP CONTINUE_READ

```

ENTER_PRESSED:

```

PRINT_STRING LINEFEED

PRINT_STRING MSG2

MOV AX,1000               ;BH periexei xiliades,BL periexei ekatontades,CH periexei
dekades,CL periexei monades

MOV DX,BX

AND DX,0F00H              ;krataw mono ton BH me tis xiliades !!!!!!!1

ROL DX,8

MUL DX                    ;vriskw poses xiliades exw

```

```

MOV DX,AX      ;apothikeyw to apotelesma ston DX

MOV AX,100     ;pros8etw tis BL ekatontades

MUL BL        ;vriskw tis 100ades AX=100 * BL=ekatontades

ADD AX,DX     ;prosthetw ton DX ston AX ara AX exei xiliades + ekatontades

MOV DX,AX     ;o DX exei to apotelesma meta thn pros8esh xiliadwn kai
ekatontadwn

MOV AX,10     ;pros8etw tis CH dekades

MOV BX,CX     ;o B twra de mou xreiazetai, xrhsimopoihsa to periexomeno tou

AND BX,0F00H  ;krataw ton CH ston B, dhladh tis dekades  !!!!!!!!!!!

ROL BX,8

MUL BL

ADD AX,DX     ;prosthetw tis ekatontades kai tis xiliades ston AX pou twra periexei
tis 10ades

AND CX,00FFH  ;krataw ton CL pou periexei tis monades

ADD AX,CX     ;pros8etw tis CL monades o AX exei twra to apotelesma pou prepei
na typwsw

MOV BX,AX     ;twra o BX exei ton arithmo mou se dyadikh morph

MOV CX,4

PRINT_LOOP:

ROL BX,4

MOV DX,BX

AND DX,000FH

CALL PRINT_HEX

LOOP PRINT_LOOP

PRINT_STRING LINEFEED

JMP START

MAIN ENDP

```

CODE_SEG ENDS

END MAIN

ΑΣΚΗΣΗ 3

Στην άσκηση αυτή διαβάζεται από το πληκτρολόγιο μια σειρά από χαρακτήρες, αριθμούς και κενά και στην συνέχεια εμφανίζονται στην επόμενη γραμμή τα πεζά, οι αριθμοί και τέλος τα κεφαλαία γράμματα χωρισμένα με κενά και με τη σειρά που πληκτρολογήθηκαν και στην αμέσως επόμενη οι δυο μεγαλύτεροι αριθμοί που δώθηκαν. Για να επιτευχθούν τα παραπάνω αποθηκεύουμε ότι διαβάζουμε (εκτός από τα κενά) στη μνήμη και του αριθμούς επιπλέον σε έναν ακόμα πίνακα. Αυτό συμβαίνει μέχρι να πατηθεί ENTER, αφήνοντας όμως περιθώριο να πληκτρολογηθούν μέχρι 14 αριθμοί, χαρακτήρες και κενά συνολικά. Στη συνέχεια, μέσω της συνάρτησης PRINTSOME διατρέχουμε τον πίνακα που έχουμε αποθηκεύσει όλα όσα διαβάστηκαν και εμφανίζουμε όλα τα στοιχεία του που βρίσκονται εντός ενός συγκεκριμένου εύρους τιμών στον πίνακα ASCII. Η αρχή και το τέλος αυτού του εύρους δίνονται ως παράμετροι μέσω των CH και CL αντίστοιχα. Έτσι καλώντας τρεις φορές την παραπάνω εκτυπώνουμε πρώτα όλα τα πεζά, μετά όλους τους αριθμούς και τέλος όλα τα κεφαλαία με τον χαρακτήρα " " να τυπώνεται ανάμεσα στην κάθε ομάδα.

EXIT MACRO ; macro exits back to DOS

MOV AX,4C00H

INT 21H

ENDM

PRINT MACRO CHAR ; tupwma xarakthra

PUSH DX

PUSH AX

MOV DL,CHAR

MOV AH,2

INT 21H

POP AX

POP DX

ENDM

READ MACRO ; diabasma apo to plhktrolgio

MOV AH,8

INT 21H

ENDM

PRINT_STRING MACRO STRING ; typwma oloklhrou string

PUSH DX

PUSH AX

LEA DX,STRING

MOV AH,09H

INT 21H

POP AX

POP DX

ENDM

STACK_SEG SEGMENT STACK

DB 50 DUP(?) ; 50 bytes for stack

STACK_SEG ENDS

DATA_SEG SEGMENT

INPUT DB 15 DUP(?)

;NUMS DB 15 DUP(?)

NEWLINE DB 0AH,0DH,'\$'

DATA_SEG ENDS

CODE_SEG SEGMENT

ASSUME CS:CODE_SEG,DS:DATA_SEG,SS:STACK_SEG

MAIN PROC FAR

```
MOV AX,DATA_SEG

MOV DS,AX ;DS <- DATA_SEG

MOV AX,STACK_SEG

MOV SS,AX ;SS <- STACK_SEG

JMP START

TERMINATE:

EXIT
```

```
START:

MOV CX,14D ;metrhths xarakthrwn

LEA BX,INPUT

;LEA DX,NUMS

MOV AH,0D
```

```
READNUM:

READ

CMP AL,'='

JE TERMINATE

CMP AL,' '

JNE CHECK_ENTER

PRINT ' '

JMP READ_NEXT2
```

```
CHECK_ENTER:

CMP AL,0DH

JNE CHECK_DIGIT

JMP NEXT
```


CHECK_DIGIT:

CMP AL,'0'

JL READNUM

CMP AL,'9'

JG CHECK_CAPITAL

PRINT AL

MOV BYTE PTR[BX],AL

JMP READ_NEXT

CHECK_CAPITAL:

CMP AL,'A'

JL READNUM

CMP AL,'Z'

JG CHECK_SMALL

PRINT AL

MOV BYTE PTR[BX],AL

JMP READ_NEXT

CHECK_SMALL:

CMP AL,'a'

JL READNUM

CMP AL,'z'

JG READNUM

PRINT AL

MOV BYTE PTR[BX],AL

READ_NEXT:

INC BX

READ_NEXT2:

LOOP READNUM

WAIT_ENTER:

READ

CMP AL,0DH

JNE WAIT_ENTER

NEXT:

MOV BYTE PTR[BX],0DH

PRINT_STRING NEWLINE

MOV CH,'a'

MOV CL,'z'

LEA BX,INPUT

CALL PRINTSOME

PRINT ' '

MOV CH,'0'

MOV CL,'9'

LEA BX,INPUT

CALL PRINTSOME

PRINT ' '

MOV CH,'A'

MOV CL,'Z'

LEA BX,INPUT

CALL PRINTSOME

PRINT_STRING NEWLINE

MOV CH,'0'

MOV CL,'9'

LEA BX,INPUT

CALL FINDMAX

PRINT_STRING NEWLINE

JMP START

FINDMAX PROC NEAR

REPEAT3232:

MOV DL,0D

MOV DH,0D

REPEAT1:

MOV AL,[BX]

CMP AL,0DH

JE EXITP1

CMP AL,CH

JL OTHER1

CMP AL,CL

```
JG OTHER1  
CMP AL,DL  
JG RENEW  
CMP AL,DL  
JE RENEW  
CMP AL,DH  
JG RENEWDH  
OTHER1:  
INC BX  
JMP REPEAT1
```

```
RENEWDH:  
MOV DH,AL  
JMP OTHER1
```

```
RENEW:  
MOV DH,DL  
MOV DL,AL  
JMP OTHER1
```

```
EXITP1:  
PRINT DL  
PRINT DH  
RET  
ENDP
```

PRINTSOME PROC NEAR

REPEAT:

MOV AL,[BX]

CMP AL,0DH

JE EXITP

CMP AL,CH

JL OTHER

CMP AL,CL

JG OTHER

PRINT AL

OTHER:

INC BX

JMP REPEAT

EXITP:

RET

ENDP

CODE_SEG ENDS

END MAIN