ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

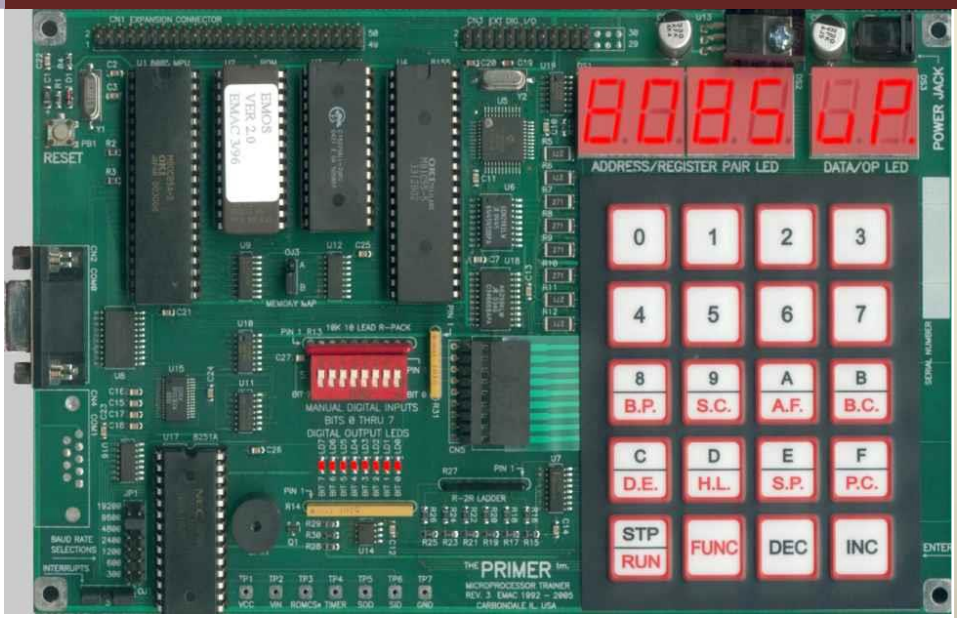
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

&

ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Εργαστήριο Μικροϋπολογιστών

8η Σειρά Ασκήσεων



7ο Εξάμηνο

ΡΟΗ Υ

Αθανασίου Νικόλαος

ΑΜ 03112074

Βαβουλιώτης Γεώργιος

ΑΜ 03112083

Γιαννούλας Βασίλειος

ΑΜ 03112117

Ζήτημα 4.1

Σκοπός της άσκησης αυτής είναι να υλοποιηθεί ένα πρόγραμμα που να απεικονίζει στην οθόνη LCD(πραγματοποιείται με την συνάρτηση lcd_data) το πλήκτρο που πατήθηκε τελευταίο στο keypad(πραγματοποιείται το διάβασμα με την συνάρτηση scan_keypad_rising_edge) . Μέχρι να πατηθεί κάποιο πλήκτρο, η οθόνη πρέπει να εμφανίζει την ένδειξη **NONE**. Τόσο η ένδειξη **NONE** όσο και το πλήκτρο που πατήθηκε εμφανίζονται στην πάνω αριστερή θέση της οθόνης. Ο κέρσορας δεν είναι ορατός γεγονός που το καθορίζουμε στην συνάρτηση lcd_init θέτωντας τον καταχωρητη r24 -> 0x0C.

Ο κώδικας σε assembly avr φαίνεται παρακάτω :

```
.DSEG
_tmp_:.byte 2
.CSEG
.include "m16def.inc"
.def temp=r16

start:
        ldi temp,high(RAMEND)                   ;stoiva
        out sph,temp
        ldi temp,low(RAMEND)
        out spl,temp

        ;arxikopoihsh keypad
        ldi temp,(1 << PC7)|(1 << PC6)|(1 << PC5)|(1 << PC4)
        out DDRC,temp
        ; arxikopoihsh othonis
        ldi temp,(1 << PD7)|(1 << PD6)|(1 << PD5)|(1 << PD4)|(1 << PD3)|(1 << PD2)
        out DDRD,temp

        rcall lcd_init                          ;arxikopoihseis othonhs
```

```
        rcall none



read:
        ldi r24,low(20)                         ;xronoi gia
        ldi r25,high(20)                        ;spinthhrismous
        rcall scan_keypad_rising_edge   ;anagnosh keypad
        rcall keypad_to_ascii                   ;epistrofh ascii plhktrou pou paththhke
        cpi r24,0                                       ;an den diavasei tipota,
ksanadiavase
        breq read
        push r24
        push r25
        rcall lcd_init                          ;katharise thn othonh
        pop r25
        pop r24
        rcall lcd_data                          ;display
        rjmp read



none:
        ldi r24,'N'             ;N
        rcall lcd_data  ; kalw thn lcd_data poy moy emfanizei sthn o8onh to dedomeno pou
einai apo8hkeumeno ston r24
        ldi r24,'O'             ;O
        rcall lcd_data
```

```
        ldi r24,'N'             ;N
        rcall lcd_data
        ldi r24,'E'             ;E
        rcall lcd_data
        ret

scan_row:
        ldi r25 ,0x08
back_:
        lsl r25
        dec r24
        brne back_
        out PORTC ,r25
        nop
        nop
        in r24 ,PINC
        andi r24 ,0x0f
        ret.

scan_keypad:
        ldi r24 ,0x01
        rcall scan_row
        swap r24
        mov r27 ,r24
        ldi r24 ,0x02
        rcall scan_row
        add r27 ,r24
        ldi r24 ,0x03
```

```
        rcall scan_row
        swap r24
        mov r26 ,r24
        ldi r24 ,0x04
        rcall scan_row
        add r26 ,r24
        movw r24 ,r26
        ret

scan_keypad_rising_edge:
        mov r22 ,r24
        rcall scan_keypad
        push r24
        push r25
        mov r24 ,r22
        ldi r25 ,0
        rcall wait_msec
        rcall scan_keypad
        pop r23
        pop r22
        and r24 ,r22
        and r25 ,r23
        ldi r26 ,low(_tmp_)
        ldi r27 ,high(_tmp_)
        ld r23 ,X+
        ld r22 ,X
        st X ,r24
        st -X ,r25
```

```
        com r23
        com r22
        and r24 ,r22
        and r25 ,r23
        ret

keypad_to_ascii:
        movw r26 ,r24
        ldi r24 ,'*'
        sbrc r26 ,0
        ret
        ldi r24 ,'0'
        sbrc r26 ,1
        ret
        ldi r24 ,'#'
        sbrc r26 ,2
        ret
        ldi r24 ,'D'
        sbrc r26 ,3
        ret .
        ldi r24 ,'7'
        sbrc r26 ,4
        ret
        ldi r24 ,'8'
        sbrc r26 ,5
        ret
        ldi r24 ,'9'
        sbrc r26 ,6
```

```
ret
ldi r24 ,'C'
sbrc r26 ,7
ret
ldi r24 ,'4'
sbrc r27 ,0
ret
ldi r24 ,'5'
sbrc r27 ,1
ret
ldi r24 ,'6'
sbrc r27 ,2
ret
ldi r24 ,'B'
sbrc r27 ,3
ret
ldi r24 ,'1'
sbrc r27 ,4
ret
ldi r24 ,'2'
sbrc r27 ,5
ret
ldi r24 ,'3'
sbrc r27 ,6
ret
ldi r24 ,'A'
sbrc r27 ,7
ret
```

```
        clr r24
        ret

wait_usec:
        sbiw r24 ,1
        nop ; 1
        nop ; 1
        nop ; 1
        nop ; 1
        brne wait_usec ; 1
        ret ; 4

wait_msec:
        push r24
        push r25
        ldi r24 , low(998)
        ldi r25 , high(998)
        rcall wait_usec
        pop r25
        pop r24
        sbiw r24 , 1
        brne wait_msec
        ret

write_2_nibbles:
                push r24
                in r25,PIND
                andi r25,0x0f
```

```
        andi r24,0xf0
        add r24,r25
        out PORTD,r24
        sbi PORTD,PD3
        cbi PORTD,PD3
        pop r24
        swap r24
        andi r24,0xf0
        add r24,r25
        out PORTD,r24
        sbi PORTD,PD3
        cbi PORTD,PD3
        ret


lcd_data:
        sbi PORTD,PD2
        rcall write_2_nibbles
        ldi r24,43
        ldi r25,0
        rcall wait_usec
        ret

lcd_command:
        cbi PORTD,PD2
        rcall write_2_nibbles
        ldi r24,39
        ldi r25,0
```

```
        rcall wait_usec
        ret

lcd_init:
        ldi r24,40
        ldi r25,0
        rcall wait_msec

        ldi r24,0x30
        out PORTD,r24
        sbi PORTD,PD3
        cbi PORTD,PD3
        ldi r24,39
        ldi r25,0
        rcall wait_usec

        ldi r24,0x30
        out PORTD,r24
        sbi PORTD,PD3
        cbi PORTD,PD3
        ldi r24,39
        ldi r25,0
        rcall wait_usec

        ldi r24,0x20
        out PORTD,r24
        sbi PORTD,PD3
        cbi PORTD,PD3
```

```
ldi r24,39
ldi r25,0
rcall wait_usec

ldi r24,0x28
rcall lcd_command

ldi r24,0x0c
rcall lcd_command

ldi r24,0x01
rcall lcd_command
ldi r24,low(1530)
ldi r25,high(1530)
rcall wait_usec

ldi r24,0x06
rcall lcd_command
ldi r24,low(3900)
ldi r25,high(3900)
rcall wait_usec

ret
```

Ζήτημα 4.2

Σκοπός της άσκησης είναι να υλοποιηθεί ένα πρόγραμμα που να εξομοιώνει ένα σύστημα συναγερμού. Τα push buttons PA0-PA7 αντιστοιχούν στις εξόδους των αισθητήρων. Αν κάποιο από αυτά πατηθεί τότε μέσα σε 4 sec (χρησιμοποιώντας τον timer1) πρέπει να

πληκτρολογηθεί στο keypad η βάρδια μας D και ο αριθμός της ομάδας μας 06. Αν είναι σωστά εμφανίζεται το μήνυμα **ALARM OFF** στο LCD display. Αλλιώς τίθεται ο συναγερμός με το συνεχόμενο άναμμα-σβήσιμο των leds PB0-PB7 με περίοδο ~0.2 sec που υποθέτουμε ότι αντιστοιχεί στην ενεργοποίηση της σειρήνας. Ταυτόχρονα εμφανίζεται στο LCD display το μήνυμα **ALARM ON**. Κατά την εισαγωγή του κωδικού εμφανίζονται τα πλήκτρα που πατάμε στο LCD display καθώς και ο κέρσορας. Όλα τα μηνύματα ξεκινούν από την πάνω αριστερή θέση του display, ενώ το υπόλοιπο να είναι κενό.
Ο κώδικας σε assembly avr φαίνεται παρακάτω :

```
.DSEG
_tmp_:.byte 2
.CSEG
.include "m16def.inc"
.def temp=r16
.def cnt=r17
.def leds=r18
.org 0x0
	jmp start
.org 0x10
	jmp ISR_TIMER1_OVF	; etiketa sthn opoia 8a paei to programma otan teleiwsoun ta 4 sec tou timer

start:
	ldi temp,high(RAMEND)				;stoiva
	out sph,temp
	ldi temp,low(RAMEND)
	out spl,temp

	;arxikopoihsh keypad
	ldi temp,(1 << PC7)|(1 << PC6)|(1 << PC5)|(1 << PC4)
	out DDRC,temp
	;arxikopoihsh lcd
	ldi temp,(1 << PD7)|(1 << PD6)|(1 << PD5)|(1 << PD4)|(1 << PD3)|(1 << PD2)
	out DDRD,temp

	clr temp					;orismos A san eisodo gia na elegxoume tous diakoptes
	out DDRA,temp
	ser temp
	out DDRB,temp					;orismos B san eksodo sta leds

	rcall lcd_init					;arxikopoihseis othonhs

activation:
	in temp,PINA
	cpi temp, 0x00					;elegxos an exoume energopoihsh
	ldi temp,0x85		;arxikopoihsh oste meta apo 4 sec na kanei uperxeilhsh
	out TCNT1H,temp					;65536 - 4*7812.5=  34.286 kai se hex einai    iso me 85EE
	ldi temp,0xEE
	out TCNT1L,temp
```

```
        breq activation                                 ;an den exei energopoihthei
perimenei

        ldi temp,(1 << TOIE1)   ;xrhsh timer
        out TIMSK, temp
        ldi temp, (1 << CS12)|(0 << CS11)|(1 << CS10) ;CLK/1024
        out TCCR1B ,temp                ;f=8MHz--->8/1024=7812.5 Hz
        sei


        ;rcall clean
        clr cnt                                 ;metrhths gia 3 arithmous
read:
        ldi r24,low(20)
        ldi r25,high(20)
        rcall scan_keypad_rising_edge
        rcall keypad_to_ascii
        cpi r24,0                       ;an den paththei tipota
        breq read
        push r24
        rcall lcd_data
        pop r24
        inc cnt
        cpi cnt,0x03            ;an den exoun diavastei kai ta 3 stoixeia sunexise
        brne cont
        clr cnt                         ;an exouun diavastei 3 stoixeia, clear ton counter kai
katharise thn othonh an teleiwsei o xronos twn 4 sec kai den exw to swsto apotelesma xtypaei
o sunagermos
        ;rcall lcd_init
        rjmp read
cont:
        cpi r24,'D'             ;  elegxw an exei patithei to plhktro D ths bardias mas
        breq wait3
        rjmp read

wait3:
        ldi r24,low(20)
        ldi r25,high(20)
        rcall scan_keypad_rising_edge
        rcall keypad_to_ascii
        cpi r24,0                       ;an den paththei tipota
        breq wait3
        push r24
        rcall lcd_data
        pop r24
        inc cnt
        cpi cnt,0x03
        brne cont2
        clr cnt
        ;rcall lcd_init
cont2:
        cpi r24, '0'
        breq wait4                      ; an exei patithei to plhktro 0
        rjmp read
```

```
wait4:
        ldi r24,low(20)
        ldi r25,high(20)
        rcall scan_keypad_rising_edge
        rcall keypad_to_ascii
        cpi r24,0                       ;an den paththei tipota
        breq wait4
        push r24
        rcall lcd_data          ;den vazo inc cnt gt gia na exo ftasei edo exo upoxreotika
diavasei 3 arithmous.
        pop r24
        cpi r24, '6'            ; an exei patithei to plhktro 6
        breq alarmoff
        clr cnt                         ;an o kodikos p exoume eisagei den einai sostos
katharise othonh kai cnt kai ksana
        ;rcall lcd_init
        rjmp read
alarmoff:
        cli                                     ;apenergopoihsh diakopon etsi wste na
apenergopoih8ei kai o timer kai na paramenei sthn o8onh to mhnuma "ALARM OFF"

off:  ;efoson path8ei o swstos sundiasmos plhktrwn anabei sthn o8onh to alarm off
        rcall lcd_init
        ldi r24,'a'             ;A
        rcall lcd_data
        ldi r24,'l'             ;L
        rcall lcd_data
        ldi r24,'a'             ;A
        rcall lcd_data
        ldi r24,'r'             ;R
        rcall lcd_data
        ldi r24,'m'             ;M
        rcall lcd_data
        ldi r24,' '             ;(space)
        rcall lcd_data
        ldi r24,'o'             ;O
        rcall lcd_data
        ldi r24,'f'             ;F
        rcall lcd_data
        ldi r24,'f'             ;F
        rcall lcd_data
        rjmp off

ISR_TIMER1_OVF:     ; molis teleiwsoun ta 4 sec tou timer kai den exei path8ei o swstos
sundiasmos plhktrwn bgazw sthn o8onh to mhnuma " ALARM ON "
        rcall lcd_init
        ldi r24,'a'             ;A
        rcall lcd_data
        ldi r24,'l'             ;L
        rcall lcd_data
        ldi r24,'a'             ;A
        rcall lcd_data
        ldi r24,'r'             ;R
        rcall lcd_data
        ldi r24,'m'             ;M
```

```
        rcall lcd_data
        ldi r24,' '             ;(space)
        rcall lcd_data
        ldi r24,'o'             ;O
        rcall lcd_data
        ldi r24,'n'             ;N
        rcall lcd_data

on_off:
        ser leds
        out PORTB,leds
        ldi r24,low(200)
        ldi r25,high(200)
        rcall wait_msec
        clr leds
        out PORTB,leds
        ldi r24,low(200)
        ldi r25,high(200)
        rcall wait_msec
        rjmp on_off
        reti

scan_row:

        ldi r25 ,0x08

back_:

        lsl r25

        dec r24

        brne back_

        out PORTC ,r25

        nop

        nop

        in r24 ,PINC

        andi r24 ,0x0f

        ret.


scan_keypad:

        ldi r24 ,0x01

        rcall scan_row

        swap r24
```

```
        mov r27 ,r24
        ldi r24 ,0x02
        rcall scan_row
        add r27 ,r24
        ldi r24 ,0x03
        rcall scan_row
        swap r24
        mov r26 ,r24
        ldi r24 ,0x04
        rcall scan_row
        add r26 ,r24
        movw r24 ,r26
        ret

scan_keypad_rising_edge:
        mov r22 ,r24
        rcall scan_keypad
        push r24
        push r25
        mov r24 ,r22
        ldi r25 ,0
        rcall wait_msec
        rcall scan_keypad
        pop r23
        pop r22
        and r24 ,r22
        and r25 ,r23
        ldi r26 ,low(_tmp_)
```

```
        ldi r27 ,high(_tmp_)
        ld r23 ,X+
        ld r22 ,X
        st X ,r24
        st -X ,r25
        com r23
        com r22
        and r24 ,r22
        and r25 ,r23
        ret

keypad_to_ascii:
        movw r26 ,r24
        ldi r24 ,'*'
        sbrc r26 ,0
        ret
        ldi r24 ,'0'
        sbrc r26 ,1
        ret
        ldi r24 ,'#'
        sbrc r26 ,2
        ret
        ldi r24 ,'D'
        sbrc r26 ,3
        ret .
        ldi r24 ,'7'
        sbrc r26 ,4
        ret
```

```
ldi r24 ,'8'
sbrc r26 ,5
ret
ldi r24 ,'9'
sbrc r26 ,6
ret
ldi r24 ,'C'
sbrc r26 ,7
ret
ldi r24 ,'4'
sbrc r27 ,0
ret
ldi r24 ,'5'
sbrc r27 ,1
ret
ldi r24 ,'6'
sbrc r27 ,2
ret
ldi r24 ,'B'
sbrc r27 ,3
ret
ldi r24 ,'1'
sbrc r27 ,4
ret
ldi r24 ,'2'
sbrc r27 ,5
ret
ldi r24 ,'3'
```

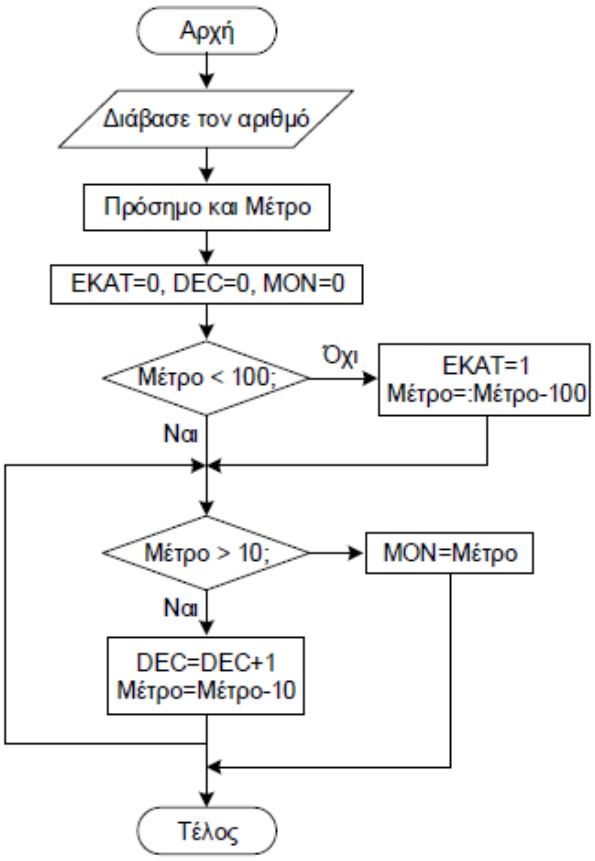
```
        sbrc r27 ,6
        ret
        ldi r24 ,'A'
        sbrc r27 ,7
        ret
        clr r24
        ret

wait_usec:
        sbiw r24 ,1
        nop ; 1
        nop ; 1
        nop ; 1
        nop ; 1
        brne wait_usec ; 1
        ret ; 4

wait_msec:
        push r24
        push r25
        ldi r24 , low(998)
        ldi r25 , high(998)
        rcall wait_usec
        pop r25
        pop r24
        sbiw r24 , 1
        brne wait_msec
        ret
```

```
write_2_nibbles:
                push r24
                in r25,PIND
                andi r25,0x0f
                andi r24,0xf0
                add r24,r25
                out PORTD,r24
                sbi PORTD,PD3
                cbi PORTD,PD3
                pop r24
                swap r24
                andi r24,0xf0
                add r24,r25
                out PORTD,r24
                sbi PORTD,PD3
                cbi PORTD,PD3
                ret


lcd_data:
        sbi PORTD,PD2
        rcall write_2_nibbles
        ldi r24,43
        ldi r25,0
        rcall wait_usec
        ret

lcd_command:
        cbi PORTD,PD2
        rcall write_2_nibbles
        ldi r24,39
        ldi r25,0
        rcall wait_usec
        ret

lcd_init:
        ldi r24,40
        ldi r25,0
        rcall wait_msec

        ldi r24,0x30
        out PORTD,r24
        sbi PORTD,PD3
        cbi PORTD,PD3
        ldi r24,39
        ldi r25,0
        rcall wait_usec

        ldi r24,0x30
        out PORTD,r24
        sbi PORTD,PD3
        cbi PORTD,PD3
        ldi r24,39
```

```
ldi r25,0
rcall wait_usec

ldi r24,0x20
out PORTD,r24
sbi PORTD,PD3
cbi PORTD,PD3
ldi r24,39
ldi r25,0
rcall wait_usec

ldi r24,0x28
rcall lcd_command

ldi r24,0x0d
rcall lcd_command

ldi r24,0x01
rcall lcd_command
ldi r24,low(1530)
ldi r25,high(1530)
rcall wait_usec

ldi r24,0x06
rcall lcd_command
ldi r24,low(3900)
ldi r25,high(3900)
rcall wait_usec

ret
```

Ζήτημα 4.3

Σκοπός της άσκησης είναι ναυλοποιειθεί ένα πρόγραμμα που να απεικονίζει στο LCD display την δυαδική τιμή (σε συμπλήρωμα **ως προς 2**) που διαβάζει από την θύρα PORTB και τη δεκαδική τιμή του σε μορφή τριών ψηφίων με το πρόσημο. Η διαδικασία να είναι συνεχόμενη. Η μετατροπή της δυαδικής τιμής σε δεκαδική έγινε με βάση το ακόλουθο διάγραμμα ροής.



Ο κώδικας σε assembly avr φαίνεται παρακάτω :

```
.include "m16def.inc"
.def temp = r16
.def input = r17
.def ekat = r18
.def dek = r19
.def mon = r20
.def pros = r21
.def zero = r22
.def one = r23

        ldi r24 , low(RAMEND) ; initialize stack pointer
        out SPL , r24
        ldi r24 , high(RAMEND)
        out SPH , r24
        ldi one,1
        clr temp                        ; orismos PortB ws thura eisodou
        out DDRB,temp
        ser temp
        out DDRC,temp
        ldi zero,0
```

```
        ; arxikopoihsh lcd display
        ldi temp,(1 << PD7)|(1 << PD6)|(1 << PD5)|(1 << PD4)|(1 << PD3)|(1 << PD2)
        out DDRD,temp

        ;rcall lcd_init          ; arxikopoiisi lcd othonis
        ;ldi r24,'!'
        ;rcall lcd_data
main:
        rcall lcd_init          ; arxikopoiisi lcd othonis
        in input,PINB
        ;out PORTC,input
        mov temp,input
        andi temp,0x80
        cpi temp,0x80
        breq negative           ; ean o arithmos einai arnhtikos phgaine sto negative
        ldi pros,'+'        ; emfanise to prosimo +
        mov temp,input
        rjmp compute            ; kai sunexise se upologismous
negative:
        ldi pros,'-'        ; alliws o arnhtikos exei -
        mov temp,input
        com temp
        add temp,one                    ; vres to sumpl. ws pros 2 tou ari8mou  dhadh antestrepse ton kai pros8ese to 1 etsi wste na ginei sthn synexeia h swsth metatroph tou bin se dec kata apoluth timh
        rjmp compute            ; kai sunexise se upologismous


; Apeikonish dyadikou arithmou se dekadikh morfh
compute:
        ldi ekat,0
        ldi dek,0
        ldi mon,0
        cpi temp,100
        brlo dekades            ; an temp < 100 kane upologismo dekadwn
        inc ekat
        subi temp,100
dekades:
        cpi temp,10
        brlo monades
        inc dek
        subi temp,10
        rjmp dekades
monades:
        mov mon,temp            ; oti perissepse einai monades
        ldi temp,8

display:    ;emfanish ths duadikhs timhs tou ari8mou ka8e fora gia ta 8 bit kanw ena shift aristera na apomonwsw to ka8e bit kai to emfanizw seiriaka sthn o8onh
        ldi r24,'0' ; bazw ston r24 arxika ton ascii code tou 0 (8elw o r24 na exei eite ton ascii code tou 1 eite tou 0)
        lsl input
        adc r24,zero
        rcall lcd_data
        dec temp
```

```
        breq next
        rjmp display
next:           ; h synarthsh lcd_data pernei ton ascii code pou exei o kataxwrhths r24 kai ton emfanizei sthn o8onh gi auto kai prin apo ka8e emfanish apo8hkeuw ton ascii code tou '0'ston r24 etsi wste telika na dhmiourgh8ei o swstos ascii code tou ari8mou pou 8elw na emfanisw
        ldi r24,'='
        rcall lcd_data
        mov r24,pros
        rcall lcd_data
        ldi r24,'0'
        add r24,ekat
        rcall lcd_data
        ldi r24,'0'
        add r24,dek
        rcall lcd_data
        ldi r24,'0'
        add r24,mon
        rcall lcd_data
        rjmp main


wait_usec:
                sbiw r24 ,1
                nop
                nop
                nop
                nop
                brne wait_usec
                ret

wait_msec:
                push r24
                push r25
                ldi r24 , low(998)
                ldi r25 , high(998)
                rcall wait_usec
                pop r25
                pop r24
                sbiw r24 , 1
                brne wait_msec
                ret

write_2_nibbles:
        push r24
        in r25 ,PIND
        andi r25 ,0x0f
        andi r24 ,0xf0
        add r24 ,r25
        out PORTD ,r24
        sbi PORTD ,PD3
        cbi PORTD ,PD3
        pop r24
        swap r24
        andi r24 ,0xf0
        add r24 ,r25
```

```
        out PORTD ,r24
        sbi PORTD ,PD3
        cbi PORTD ,PD3
        ret

lcd_data:
        sbi PORTD ,PD2
        rcall write_2_nibbles
        ldi r24 ,43
        ldi r25 ,0
        rcall wait_usec
        ret

lcd_command:
        cbi PORTD ,PD2
        rcall write_2_nibbles
        ldi r24 ,39
        ldi r25 ,0
        rcall wait_usec
        ret

lcd_init:
        ldi r24 ,40
        ldi r25 ,0
        rcall wait_msec
        ldi r24 ,0x30
        out PORTD ,r24
        sbi PORTD ,PD3
        cbi PORTD ,PD3
        ldi r24 ,39
        ldi r25 ,0
        rcall wait_usec
        ldi r24 ,0x30
        out PORTD ,r24
        sbi PORTD ,PD3
        cbi PORTD ,PD3
        ldi r24 ,39
        ldi r25 ,0
        rcall wait_usec
        ldi r24 ,0x20
        out PORTD ,r24
        sbi PORTD ,PD3
        cbi PORTD ,PD3
        ldi r24 ,39
        ldi r25 ,0
        rcall wait_usec
        ldi r24 ,0x28
        rcall lcd_command
        ldi r24 ,0x0c
        rcall lcd_command
        ldi r24 ,0x01
        rcall lcd_command
        ldi r24 ,low(1530)
        ldi r25 ,high(1530)
        rcall wait_usec
```

```
ldi r24 ,0x06
rcall lcd_command
ret
```