ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

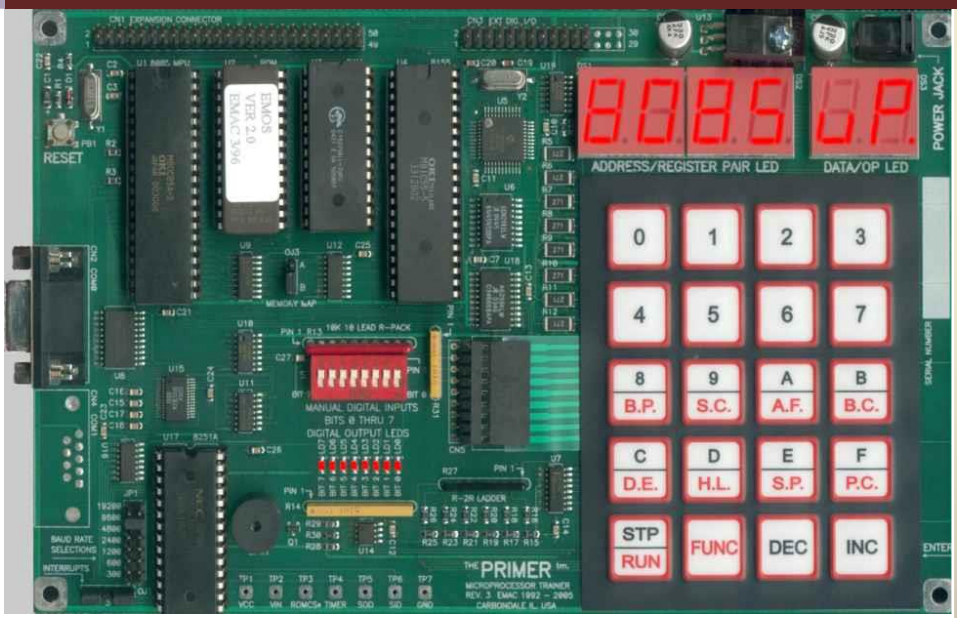
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

&

ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Εργαστήριο Μικροϋπολογιστών

6η Σειρά Ασκήσεων



7ο Εξάμηνο

ΡΟΗ Υ

Αθανασίου Νικόλαος

ΑΜ 03112074

Βαβουλιώτης Γεώργιος

ΑΜ 03112083

Γιαννούλας Βασίλειος

ΑΜ 03112117

Ζήτημα 2.1

Το παρακάτω πρόγραμμα υλοποιεί στο κυρίως μέρος την λειτουργία ενός μετρητή με ταχύτητα μέτρησης 1 μέτρηση ανά 0.2 sec. Περιλαμβάνει επίσης ρουτίνα εξυπηρέτησης εξωτερικής διακοπής που όταν ενεργοποιείται απαριθμεί το πλήθος των διακοπών εφόσον το PD7 είναι ενεργοποιημένο. Αρχικά καθορίζονται ποιές θύρες θα είναι έξοδοι(A,B) και ποιές είσοδοι(D),αρχικοποιούνται οι μετρητές για τις δύο μετρήσεις,ρυθμίζονται οι παράμετροι ώστε να καθοριστεί η επιθυμητή διακοπή και τέλος αυτή ενεργοποιείται.Στη συνέχεια δίνονται κατάλληλες τιμές στους r24,r25 ώστε με την χρήση των wait_usec & wait_msec να προκληθούν κατά την εμφάνιση των αποτελεσμάτων οι επιθυμητές καθυστερήσεις. Και στις δύο μετρήσεις ελεγχέται αν έχουν φτάσει το 255 και αν αυτό συμβαίνει τότε οι μετρητές μηδενίζονται και συνεχίζεται η μέτρηση από εκεί.Στη ρουτίνα εξυπηρέτησης της διακοπής εξετάζουμε τηνπερίπτωση που έχουμε αναπηδήσεις με χρήση του αλγορίθμου του εργαστηριακού οδηγού και στη συνέχεια αν το PD7 είναι ενεργοποιημένο, ο μετρητής των διακοπών αυξάνει και εμφανίζεται η νέα τιμή του στην θύρα A. Σε αντίθετη περίπτωση η μέτρηση παραμένει η ίδια με πριν και επίσης εμφανίζεται στην θύρα A.
Ο κώδικας σε assembly avr φαίνεται παρακάτω :

```
.include "m16def.inc"
.def cnt1=r19
.def cnt2=r18
.def eight=r17
.def check=r22
.def interr=r20
.def max=r21
.def temp=r16

.org 0x00
        jmp reset

.org 0x4
        jmp interr1


reset:
        ldi temp,high(ramend)           ;arxikopoihsh g stack
        out sph,temp
        ldi temp,low(ramend)
        out spl,temp

        ser temp
        out DDRB,temp                   ;B eksodos tou counter ts main
        out DDRA,temp                   ;A eksodos tou counter ton interrupts
        clr temp                        ;D eisodos
        out DDRD,temp

        clr cnt1                        ;arxikopoihseis metrhton cnt1,cnt2
        clr cnt2
        out porta,cnt2                  ;apeikonish metrhsh interr
        ser max                 ;arxikopoihsh max--sugrish an exo ftasei st metrhsh 255
```

```
        ldi eight,0x80                  ;arxikopoihsh eight--sugrish kai apomonosh PD7
        ldi interr,( 1 << ISC11) | ( 1 << ISC10)   ;orismos stoixeion gia diakoph--- int1 sthn
anerxomenh akmh
        out MCUCR , interr
        ldi interr ,( 1 << INT1)
        out GICR , interr
        sei
        ;energopoihsh diakopon

        ldi r24,low(200)                ;orismos kathusterhshs
        ldi r25,high(200)

cont:
        out portb,cnt1
        ;arxikopoihsh metrhth main
        ldi r24,low(200)                ;orismos kathusterhshs
        ldi r25,high(200)
        rcall wait_msec
        inc cnt1
        cp cnt1,max       ;an eftase h metrhsh st 255 mhdenise prota kai meta continue, an oxi
                          ;continue amesos
        brne cont
        out portb,cnt1
        ;arxikopoihsh metrhth main
        ldi r24,low(200)                        ;orismos kathusterhshs
        ldi r25,high(200)
        rcall wait_msec
        clr cnt1
        rjmp cont

interr1:
        ;push SREG
        ;sozo reg shmaion
        push r24
        push r25
; Algorithmos gia apofygi anapidisewn tou diakopti
repeat:
        ldi r24 ,(1 << INTF1)
        out GIFR ,r24
        ldi r24,low(5)
        ldi r25,high(5)
        rcall wait_msec
        in r24,GIFR
        sbrc r24,7
        rjmp repeat

        in check,pind                           ;diavazo eisodo D
        and check,eight                         ;apomonosh pinD7
        cp check,eight
        brne labeljmp                   ;an pinD7=1 aukshse ton cnt2, allios prosperase
        inc cnt2
labeljmp:
        out porta,cnt2                  ;emfanise cnt2
```

```
        cp cnt2,max      ;an eftase h metrhsh st 255 mhdenise prota kai meta return, an oxi

                                ;return amesos
        brne return
        clr cnt2
return:
        pop r25
        pop r24
        ;pop SREG
        ;epanafora reg shmaion
        reti
        ;return interrupt

wait_usec:
        sbiw r24,1                      ; 2 κύκλοι (0.250 μsec)
        nop                             ; 1 κύκλος (0.125 μsec)
        nop                             ; 1 κύκλος (0.125 μsec)
        nop                             ; 1 κύκλος (0.125 μsec)
        nop                             ; 1 κύκλος (0.125 μsec)
        brne wait_usec                  ; 1 ή 2 κύκλοι (0.125 ή 0.250 μsec)
        ret
        ; 4 κύκλοι (0.500 μsec)

wait_msec:
        push r24                        ; 2 κύκλοι (0.250 μsec)
        push r25                        ; 2 κύκλοι
        ldi r24,low(998)   ; φόρτωσε τον καταχ. r25:r24 με 998 (1 κύκλος - 0.125 μsec)
        ldi r25,high(998)               ; 1 κύκλος (0.125 μsec)
        rcall wait_usec      ; 3 κύκλοι (0.375 μsec), προκαλεί συνολικά καθυστέρηση 998.375
;μsec
        pop r25                         ; 2 κύκλοι (0.250 μsec)
        pop r24                         ; 2 κύκλοι
        sbiw r24 , 1                    ; 2 κύκλοι
        brne wait_msec                  ; 1 ή 2 κύκλοι (0.125 ή 0.250 μsec)
        ret
        ; 4 κύκλοι (0.500 μsec)
```

Ζήτημα 2.2

Το παρακάτω πρόγραμμα περιλαμβάνει στο κυρίως πρόγραμμα την ίδια λειτουργία με την 2.1 και μια ρουτίνα εξυπηρέτησης της εξωτερικής διακοπής INT0 (PD2) που όταν ενεργοποιείται απεικονίζει στα LEDs PC0-7 ξεκινώντας απο το LSB bit τοπλήθος των διακοπτών της θύρας PORTA (PA7-PA0) που είναι ON.Αρχικά καθορίζονται οι θύρες εισόδου και εξόδου,αρχικοποιούνται οι μετρητές του κυρίου προγράμματος καιένας ακόμα που θα μετράει τα ON leds και ρυθμίζονται οι παράμετροι της διακοπής INT0 και των καθυστερήσεων στην εμφάνιση αποτελεσμάτων.Για την μέτρηση της main και πάλι κάθε φορά ελεγχέται αν έφτασε το 255 και αν αυτό συμβαίνει ο μετρητής μηδενίζεται. Στη ρουτίνα εξυπηρέτησης της διακοπής εξετάζουμε πάλι την περίπτωση ύπαρξης σπινθηρισμών. Στη συνέχεια,διαβάζεται η είσοδος A και αρχικοποιείται ένας μετρητής στο

8 ώστε να πραγματοποιηθούν 8 επαναλήψεις για τον έλεγχο όλων των leds. Αν το msb είναι ON ο μετρητής των αναμμένων leds αυξάνει και με ολίσθηση της εισόδου από την θύρα A ελεγχέται το επόμενο led. Αν το msb είναι OFF παραλείπεται η αύξηση του μετρητή.

Ο κώδικας σε assembly avr φαίνεται παρακάτω :

```
.include "m16def.inc"
 .def cnt1=r19
 .def again_cnt=r18
 .def pins_cnt=r17
 .def check=r22
 .def interr=r20
 .def max=r21
 .def temp=r16
 .def outreg=r28

 ;orismos dieuthunseon gia reset kai interrupt INT1
 .org 0x0
 jmp reset
 .org 0x2
 jmp interr0
 ;reti

 reset:
                ldi temp,high(ramend)                   ;arxikopoihsh g stack
                out sph,temp
                ldi temp,low(ramend)
                out spl,temp
;orismos eisodon kai eksodon
                ser temp
                out DDRB,temp                           ;B eksodos tou counter ts main
                out DDRC,temp                           ;C eksodos tou counter ton pins
                clr temp                        ;D eisodos gia metrhsh posa pins einai set
                out DDRD,temp

                clr cnt1                ;arxikopoihseis metrhton cnt1,pins_cnt
                clr pins_cnt            ;cnt1--> metrhsh main, pins_cnt-->metrhsh on pins
                out portc,pins_cnt                      ;apeikonish metrhsh interr
                ser max     ;arxikopoihsh max--sugrish an exo ftasei st metrhsh 255 O cnt1
                ldi pins_cnt,0                          ;arxikopoihsh metrhth set pins

                ldi r24,low(200)        ;orismos kathusterhshs gia thn metrhsh ths main
                ldi r25,high(200)

                ldi interr,( 1 << ISC00) | ( 1 << ISC01)   ; ΄    int0 sthn anerxomenh akmh
                out MCUCR , interr
                ldi interr ,( 1 << INT0)
                out GICR , interr
                sei
                ;energopoihsh diakopon
; main metrhths apo 0 ews 255
cont:
                out portb,cnt1                  ;enarksh metrhshs--- ola ta led svhsta
                ldi r24,low(200)        ;orismos kathusterhshs gia thn metrhsh ths main
                ldi r25,high(200)
```

```
                rcall wait_msec
                inc cnt1        ;aukshsh metrhth metrhshs g na deikso epomenh timh
                cp cnt1,max     ;an eftase h metrhsh st 255 mhdenise prota kai meta
continue, an oxi continue amesos
                brne cont
                out portb,cnt1          ;enarksh metrhshs--- ola ta led svhsta
                ldi r24,low(200)        ;orismos kathusterhshs gia thn metrhsh ths main
                ldi r25,high(200)
                rcall wait_msec
                clr cnt1
                rjmp cont

;DIAKOPH
interr0:
                push r24  ;sozo r24, r25 logo allaghs timon sthn klhsh ths kathusterhshs
                push r25

; to check_again einai elegxos gia spinthhrismous
check_again:
                ldi r24,(1<<INTF0)
                out GIFR,r24
                push r24
                push r25
                ldi r24,low(5)                          ;orismos kathusterhshs
                ldi r25,high(5)
                rcall wait_msec
                pop r25
                pop r24
                in r24,GIFR
                sbrc r24,6




                rjmp check_again
;telos elegxou spinthhrismon

                in check,pina                   ;diavazo eisodo a gia na metrhsh pins
                clr again_cnt   ;arxikopoihsh metrhth gia 8 epanalhpseis---8 leds gia elegxo
                clr pins_cnt
again:
                sbrc check,7            ;an to msb einai 0 skip thn aukshsh tou pins_cnt
                inc pins_cnt            ;an msb=1 aukshse pins_cnt
                lsl check               ;olisthhsh gia ton elegxo tou epomenou bit
                inc again_cnt           ;aukshsh metrhth epanalhpseon
                cpi again_cnt,8         ;an elegksa ola ta bit vgaino, allios again
                brne again

;trexw gia pins_cnt fores kai vazw tosous asous ap ta deksia pros ta aristera ston outreg
                ldi outreg,0
                cpi pins_cnt,0
                breq zero
execute:
                lsl outreg
        ;       ori outreg,1
```

```
                adiw outreg,1
                dec pins_cnt
                cpi pins_cnt,0
                brne execute

zero:
                out PORTC,outreg        ;meta thn oloklhrosh tou elegxou olon ton bit
emfanizo thn timh tou outreg sthn thura C.

return:
                pop r25     ;epanafora timon r24,r25 pou ekana push gia ton elegxo
spinthhrismon.
                pop r24
                reti



wait_usec:
                sbiw r24 ,1             ; 2 κύκλοι (0.250 μsec)
                nop                     ; 1 κύκλος (0.125 μsec)
                nop                     ; 1 κύκλος (0.125 μsec)
                nop                     ; 1 κύκλος (0.125 μsec)
                nop                     ; 1 κύκλος (0.125 μsec)
                brne wait_usec          ή 2 κύκλοι (0.125 ή 0.250 μsec)
                ret                     ; 4 κύκλοι (0.500 μsec)

wait_msec:
                push r24                ; 2 κύκλοι (0.250 μsec)
                push r25                ; 2 κύκλοι
                ldi r24 , low(998)      ; φόρτωσε τον καταχ. r25:r24 με 998 (1 κύκλος -
0.125 μsec)
                ldi r25 , high(998)     ; 1 κύκλος (0.125 μsec)
                rcall wait_usec         ; 3 κύκλοι (0.375 μsec), προκαλεί συνολικά
καθυστέρηση 998.375 μsec
                pop r25                 ; 2 κύκλοι (0.250 μsec)
                pop r24                 ; 2 κύκλοι
                sbiw r24 , 1            ; 2 κύκλοι
                brne wait_msec          ; 1 ή 2 κύκλοι (0.125 ή 0.250 μsec)
                ret                     ; 4 κύκλοι (0.500 μsec)
```

Ζήτημα 2.3

Το παρακάτω προγράμμα προσομοίωνει έναν αυτοματισμό για τη λειτουργία ενός φωτιστικού σώματος. Ορίζουμε τη θύρα D ως είσοδο αφού από εκεί λαμβάνεται η αίτηση διακοπής (INT1)και στη θύρα A το bit 8(PA7)ορίζεται ως είσοδος ενώ το PB0 ορίζεται ως έξοδος που αντιπροσωπεύει το φωτιστικό σώμα και είναι ανοιχτό όταν έχει την τιμή 1. Αρχικά, ορίζουμε τις παραμέτρους για τον χρονιστή, ο οποίος όποτε ενεργοποιείται σβήνει το φωτιστικό σώμα. Κάθε φορά που πιέζουμε το PD3 ή το PA7 ανανεώνεται ο χρόνος ενεργοποίησης του timer ώστε το φωτιστικό να σβήσει από τότε σε 2 sec, αν όμως ξαναπατηθούν μέσα στο χρονικό διάστημα των 2 sec ένα από τα παραπάνω κουμπία τότε θα πρέπει να ανάβουν για τα πρώτα 0.5 sec όλα τα leds της θύρας PORTB και στην συνέχεια σβήνουν εκτός από το PB0 που παραμένει συνολικά για 2 sec εκτός και αν ανανεωθεί .

Ο κώδικας σε assembly avr φαίνεται παρακάτω :

```
.include "m16def.inc"
.def temp = r16
;8MHz(sunxothta plaketas)/1024(ana 1024cc auksanetai kata 1 o timer) = 7812,5
;ara 2x7812,5 = 15625 kai
;Επειδή η υπερχείλιση γίνεται όταν μετρήσει 65536 κύκλους (16 ψηφία),
;θα πρέπει η αρχική τιμή που θα του δοθεί πριν αρχίσει να μετράει προς τα πάνω να είναι
;65536-15625=49911=0xC2F7
.equ highTimerStart = 0xC2
.equ lowTimerStart = 0xF7
.def flag=r22 ;me flag=1 exw ananewsh enw me flag=0 den exw
.def flag2=r23;

.org 0x0
        jmp start
.org 0x4
        jmp int1_routine
.org 0x10
        jmp timer1_routine

start:

        ldi temp,high(ramend)
        out sph,temp
        ldi temp,low(ramend)
        out spl,temp

        ; Orismos eisodwn kai eksodwn
        clr temp
        out DDRD,temp                                   ; PortD ws eisodos
        clr temp
        out DDRA,temp                                   ; PortA ws eisodos (to A7)
        ser temp
        out DDRB,temp                                   ; PortB ws eksodos

        ; Orismos k energopoiisi diakopis xronisti timer1
        ldi temp,(1 << TOIE1)           ; Energopoiisi diakopis uperxeilisis
        out TIMSK,temp                                  ; gia ton timer1
        ldi temp,(1 << CS12)|(0 << CS11)|(1 << CS10)  ; CK/1024
        out TCCR1B,temp

         ; Orismos k energopoiisi diakopis INT1
        ldi temp,(1 << ISC10)|(1 << ISC11)      ; Anagnwrisi diakopis INT1 se
        out MCUCR,temp                          ; thetiki akmi
        ldi temp,(1 << INT1)                            ; Sima gia energopoiisi ths INT1
        out GICR,temp
        sei                                                                     ;
Energopoiisi diakopwn

        ldi flag,0
main:
        in temp,PINA                                            ; Diavase tin ta bits tis PortA
        andi temp,0x80                                          ; kai apomonwse to bit7
        breq main
```

```
continue:                                          ; An einai 0 epanelave
        in temp,PINA
        andi temp,0x80
        brne continue

        ldi temp,highTimerStart                    ; Arxikopoiise ek neou ton
        out TCNT1H,temp                                    ; xronisti
        ldi temp,lowTimerStart
        out TCNT1L,temp
        cpi flag,1
        breq setall2
        jmp fix1


setall1:
        ser temp
        out PORTB,temp
        ldi r24,low(500)          ;orismos kathusterhshs gia thn metrhsh ths main
        ldi r25,high(500)
        rcall wait_msec

fix1:
        ldi flag,1
        ldi temp,1                ; Proetoimase tin eksodo
        out PORTB,temp            ; kai emfanise tin
        rjmp main

int1_routine:
; Elegxos gia spinthirismo
repeat:
        ldi r24 ,(1 << INTF1)
        out GIFR ,r24
        ldi r24,low(5)
        ldi r25,high(5)
        rcall wait_msec
        in r24,GIFR
        sbrc r24,7 ; sbrc r24,6



        rjmp repeat

        ldi temp,highTimerStart           ; Arxikopoiise ek neou ton
        out TCNT1H,temp                   ; xronisti
        ldi temp,lowTimerStart
        out TCNT1L,temp
        cpi flag,1
        breq setall2
        jmp fix2

setall2:
        ser temp
        out PORTB,temp
        ldi r24,low(500)          ;orismos kathusterhshs gia thn metrhsh ths main
        ldi r25,high(500)
```

```
        rcall wait_msec

fix2:
        ldi flag,1
        ldi temp,1                  ; Proetoimase tin eksodo
        out PORTB,temp              ; kai emfanise tin
        reti

timer1_routine:
        ldi flag,0
        clr temp
        out PORTB,temp
        reti

wait_usec:
                sbiw r24 ,1         ; 2 κύκλοι (0.250 μsec)
                nop                 ; 1 κύκλος (0.125 μsec)
                nop                 ; 1 κύκλος (0.125 μsec)
                nop                 ; 1 κύκλος (0.125 μsec)
                nop                 ; 1 κύκλος (0.125 μsec)
                brne wait_usec      ; 1 ή 2 κύκλοι (0.125 ή 0.250 μsec)
                ret                 ; 4 κύκλοι (0.500 μsec)

wait_msec:
                push r24            ; 2 κύκλοι (0.250 μsec)
                push r25            ; 2 κύκλοι
                ldi r24 , low(998)  ; φόρτωσε τον καταχ. r25:r24 με 998 (1 κύκλος -
0.125 μsec)
                ldi r25 , high(998) ; 1 κύκλος (0.125 μsec)
                rcall wait_usec     ; 3 κύκλοι (0.375 μsec), προκαλεί συνολικά
καθυστέρηση 998.375 μsec
                pop r25             ; 2 κύκλοι (0.250 μsec)
                pop r24             ; 2 κύκλοι
                sbiw r24 , 1        ; 2 κύκλοι
                brne wait_msec      ; 1 ή 2 κύκλοι (0.125 ή 0.250 μsec)
                ret                 ; 4 κύκλοι (0.500 μsec)
```