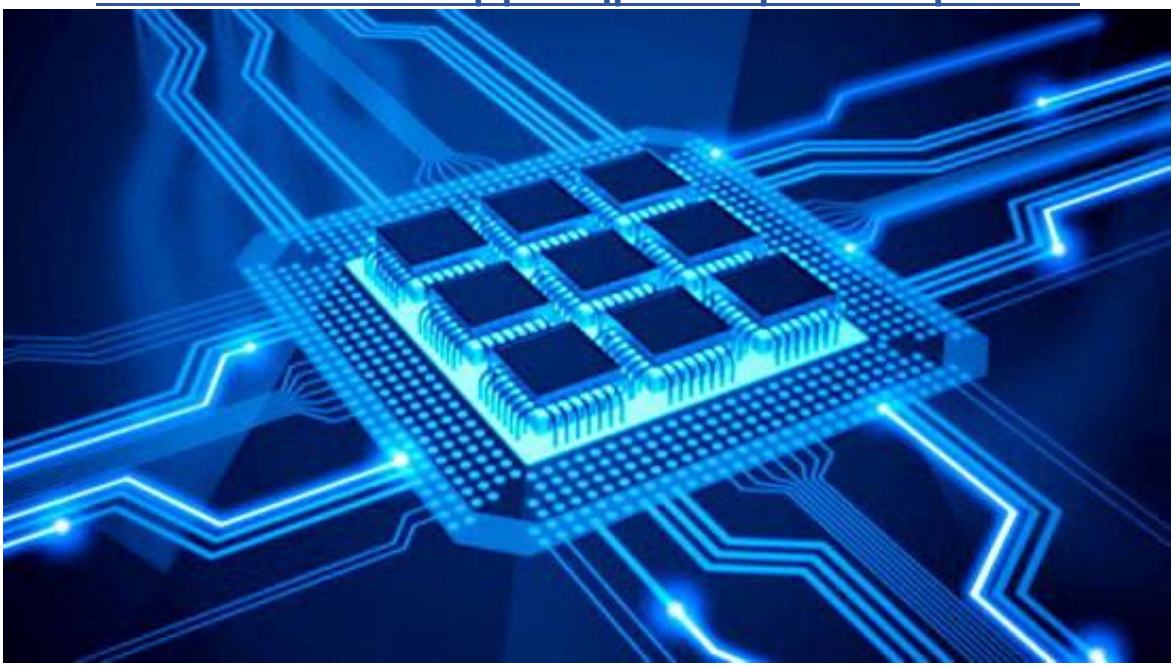
5η ΕΡΓΑΣΤΗΡΙΑΚΗ ΆΣΚΗΣΗ
ΓΙΑ ΤΟ ΜΑΘΗΜΑ "Εργαστήριο Μικροϋπολογιστών"



Ομάδα: 16
Ημερομηνία επίδειξης: 22/11/2017
Μέλη:
Κερασιώτης Ιωάννης, Α.Μ.:03114951
Πευκιανάκης Κωνσταντίνος, Α.Μ.: 03114897
Ραφτόπουλος Ευάγγελος, Α.Μ.:03114743

ΑΣΚΗΣΗ 1:

Η άσκηση αυτή ζητάει να εμφανίζει συχνώς ένα μετρητή ο οποίος αυξάνεται και εμφανίζεται ανά 0.2sec στην θύρα εξόδου PORTB. Επίσης όταν γίνεται διακοπή INT1 και όταν είναι ανεβασμένο το LSB των dip switch της θύρας D τότε προσμετράει την διακοπή αλλιώς όχι. Το πλήθος των διακοπών εμφανίζεται στην θύρα εξόδου PORTA.

Στην αρχή αρχικοποιούμε τον δείκτη της στοίβας αφού θα γίνει κλήση ρουτίνας και ορίζουμε τις θύρες A,B ως εξόδους και την D ως είσοδο. Έπειτα ενεργοποιούμε τις διακοπές INT1 μέσω της σημαίας καταχωρητών GICR και ορίζουμε να παίρνει το σήμα στην θετική ακμή μέσω της σημαίας MCUCR.

Η κύρια επανάληψη το μόνο που κάνει είναι να αυξάνει τον μετρητή του αριθμού που ζητάει να εμφανίζει και να το εμφανίζει μαζί με τον μετρητή των διακοπών. Αυτά τα εμφανίζει ανά 0.2sec και γιαυτό καλούμε την ρουτίνα χρονοκαθυστέρησης.

Η ρουτίνα χρονοκαθυστέρησης ελέγχει στην αρχή το αν έγινε σπινθηρισμός. Θέτουμε στο MSB του GIFR το 1 και εκτελούμε καθυστέρηση 0.5 ms. Έπειτα ελέγχουμε αν εξακολουθεί να είναι 1. Αν όντως είναι 1 τότε συνέβη το φαινόμενο του σπινθηρισμού και αγνοούμε την επιπρόσθετη διακοπή. Αλλιώς συνεχίζουμε κανονικά στην ρουτίνα διακοπής και ελέγχουμε αν το PD7 είναι στο λογικό 1. Αν είναι τότε προσμετράται η διακοπή αλλιώς αγνοείται και επιστρέφουμε στην κύρια επανάληψη από όπου και συνέβη η διακοπή.
Ο κώδικας της άσκησης δίνεται παρακάτω:

```
.include "m16def.inc"
.def temp=r20
.def m1=r16     //metrhths kurios programmatos
.def m2=r18     //plhthos diakopon
```

```
jmp start
.org 0x4
jmp ISR1
reti

start:
ldi temp,low(RAMEND)    //arxikopoihsh tou deikth stibas afou ginetai klhsh routinas
out SPL,temp
ldi temp,high(RAMEND)
out SPH,temp
ser temp
out DDRB,temp     //h A,B os e9odoi
out DDRA,temp
clr temp            //o D einai h eisodos
out DDRD,temp
ldi temp, (1<<ISC11) | (1<<ISC10)  //shma thetikhs akmhs
out MCUCR,temp
ldi temp, (1<<INT1)   //energopoihsh INT1
out GICR,temp
sei                 // enegropoihsh diakopon
clr m2
clr m1
loop:
rcall Delay      //kalo thn xronokathisterish 0,2sec
inc m1           //auksanetai o metrhths
out PORTB,m1     //emfanise metrhth kai plithos diakopon
out PORTA,m2
jmp loop

ISR1:
loop1:
        ldi temp, 0b10000000
        out GIFR, temp
msec5Delay:                     ;texnhth xronokathusterhsh 5 msec opos dothike stis ekfoniseis ths prohgoumenhs
                                ;seiras
        ldi r24,low(5)
        ldi r25,high(5)
m_sec1:
        push r24
        push r25
        ldi r24,low(998)
        ldi r25,high(998)
wait_usecBlock1:
        sbiw r24,1
        nop
        nop
        nop
        nop
        brne wait_usecBlock1
        pop r25
        pop r24
        sbiw r24,1
        brne m_sec1
```

```
        in temp, GIFR
andi temp, 0b10000000    //elegxos gia to an egine klhsh diakophs se ligotero apo 5msec. an egine simenei oti egine
                         //spinthirismos ara thn agnooume.
cpi temp, 0b10000000
breq loop1
in temp,PIND             //elegxos gia to PD7. an einai 1 tote prosmetrountai oi diakopes allios agnoountai
sbrc temp,7
inc m2                   //aukshsh tou metrith

out PORTA,m2
reti

Delay:                   //xronokathisterish 0,2 sec
        ldi r24,low(200)
        ldi r25,high(200)
m_sec:
        push r24
        push r25
        ldi r24,low(998)
        ldi r25,high(998)
wait_usecBlock:
        sbiw r24,1
        nop
        nop
        nop
        nop
        brne wait_usecBlock
        pop r25
        pop r24
        sbiw r24,1
        brne m_sec
ret
```

ΑΣΚΗΣΗ 2:

Αυτή η άσκηση ζητάει πάλι ένα μετρητή που να αυξάνεται και να εμφανίζεται στην θύρα PORTB. Όμως τώρα όταν προκαλείται διακοπή τύπου INTR0 τότε να εμφανίζει στην θύρα PORTC τόσα LEDs όσο είναι το πλήθος των λογικών 1 στην θύρα εισόδου A, ξεκινώντας από το LSB.

Στην αρχή ορίζουμε τον δείκτη στοίβας αφού θα γίνει κλήση ρουτίνας. Έπειτα ορίζουμε τις εξόδους B,C και την είσοδο A. Τέλος ενεργοποιούμε τις διακοπές τύπου INT0 και θέτουμε θετική ακμή. Στην κύρια επανάληψη πάλι εμφανίζουμε τον μετρητή του προγράμματος με καθυστέρηση 0,2sec και τον αυξάνουμε και εμφανίζουμε τα LEDs της θύρας C που πρέπει να ανοίξουν.

Στην ρουτίνα διακοπής μετράμε το πλήθος των λογικών 1 έστω N που έχει η είσοδος A. Αυτό το κάνουμε με 8 επαναλήψεις στις οποίες ολισθαίνουμε την είσοδο και ελέγχουμε κάθε φορά αν είναι στο λογικό 1. Στην συνέχεια για να εμφανίσουμε τα σωστά LEDS στην έξοδο C δουλεύουμε ως εξής. Έχουμε δύο καταχωρητές ο ένας ξεκινάει από την τιμή 0χ00 και ο άλλος από την τιμή 0χ01. Ο $1^{ος}$ καταχωρητής είναι και αυτός που θα δώσει την έξοδο. Θα κάνουμε N επαναλήψεις όπου σε κάθε μια θα ολισθαίνουμε τον $2^{ο}$ καταχωρητή μια θέση αριστερά και έπειτα θα κάνουμε την λογική πράξη OR μεταξύ αυτού και του $1^{ου}$. Έτσι θα σχηματιστεί η επιθυμητή έξοδος στην θύρα C δηλαδή θα ανάψουν N LEDs ξεκινώντας από το LSB.

Ο κώδικας την άσκησης αυτής είναι ο εξής :

```
.include "m16def.inc"
.def temp=r20
.def m1=r16        //m1 einai o metriths gia to kurio meros ths askhshs. emfanizetai sta PB7-PB0
.def m2=r18        //einai o metriths
.def reg=r21
.def temp1=r19
.def temp2=r17
jmp start
.org 0x4
jmp ISR2
reti

start:
ldi temp,low(RAMEND)   //  arxikopoihsh tou deikth ths stibas afou kaleitai routina
out SPL,temp
ldi temp,high(RAMEND)
out SPH,temp
ser temp             //oi D,C einai oi e9odoi
out DDRB,temp
out DDRC,temp
clr temp            //o A einai h eisodos
out DDRA,temp

ldi temp, (1<<ISC01) | (1<<ISC00)    //shma thetikhs akmhs
out MCUCR,temp
ldi temp, (1<<INT0)                 //energopoihsh diakophs INT0
out GICR,temp
sei                            //energopoihsh diakopon
clr m1
clr m2

loop:
rcall Delay          //kalo thn xronokathisterish 0,2 sec
inc m1
out PORTB,m1        //emfanise ton metrhth m1 sthn eksodo B
out PORTC,temp2      //emfanise ton temp2 sthn e9odo C
jmp loop

ISR2:
clr reg
ldi r22, 8      //8 epanalipseis thelo
clr r23        //metraei posa "1" exo dosei apo to PINA
in reg,PINA
loop1:         //metrao to plhthos ton "1"
lsr reg
brcc next
inc r23
next:
dec r22
brne loop1
ldi temp1,1
```

```
ldi temp2,0
loop2:      //sxhmatizo thn e9odo sto C me thn methodo tou OR
cpi r23,0
breq end
or temp2,temp1
lsl temp1
dec r23
jmp loop2

end:            //emfanise kai return
out PORTC, temp2
reti

Delay:
        ldi r24,low(200)
        ldi r25,high(200)
m_sec:
        push r24
        push r25
        ldi r24,low(998)
        ldi r25,high(998)
wait_usecBlock:
        sbiw r24,1
        nop
        nop
        nop
        nop
        brne wait_usecBlock
        pop r25
        pop r24
        sbiw r24,1
        brne m_sec
reti
```

ΆΣΚΗΣΗ 3:

Σε αυτή την άσκηση αρχικά ενεργοποιούμε τις διακοπές και τις θύρες εισόδου A,D. Το πρόβλημα που παρουσιάζεται είναι το εξής: όταν πατάμε το κουμπί PA7 όταν όλα τα LEDs είναι σβηστά τότε θα ανάψουν όλα τα LEDs σαν να έχει κάνει ανανέωση χρόνου. Αυτό συμβαίνει γιατί καθώς όσο πατάμε το κουμπί λαμβάνει πολλές φορέας διαφορετικά πατήματα και έτσι λειτουργεί σαν να ανανεώνουμε τον χρόνο. Αυτό το πρόβλημα το αντιμετωπίζουμε με την εισαγωγή μια σημαίας flag. Στην συνέχεια είτε πατηθεί το κουμπί διακοπής PD3 είτε το κουμπί εισόδου PA7 τότε πάμε στην ρουτίνα ledupdate η οποία είναι υπεύθυνη για το άναμμά είτε για 4sec είτε των 0,5sec του 1^{ου} led ανάλογα με την κατάσταση στην οποία θα βρισκόταν όταν πατήθηκε το κουμπί. Ορίζουμε τον timer στην κατάλληλη τιμή και τον ενεργοποιούμε. Έπειτα με το που τελειώσει ο χρόνος του timer προκαλείται μία διακοπή και πηγαίνουμε στην ρουτίνα του timer. Αυτή η ρουτίνα είναι υπεύθυνη για να ελέγξει το αν θα ανάψει το 1^{ο} led για άλλα 3.5 sec αφού ήδη έχει ανάψει για 0,5sec από την ledupdate. Αν χρειάζεται να ανάψει τότε αρχικοποιήται πάλι η timer με τον αντίστοιχο ακέραιο που θα δώσει τα 3,5 sec.

Ο κώδικα είναι ο εξής:

```
.include "m16def.inc"               ;prosthhkh arxeiou kefalidas gia xeirismo thurwn I/0 mesw twn.def flag = r19
.def temp = r16
.def portAValue = r17               ;apothikeuoume thn eisodo A
```

```
.def ledValue = r18
.equ fourSec = 34286                    ;orismos statherwn timwn pou tha odhghthoun ston 16bito xronisth Timer 1
.equ halfSec = 61629                    ;me vash thn parapanw sullogistikh poreia
.equ threeAndHalfSec = 38192
.org 0
jmp reset
.org 4
jmp int1Routine
.org 0x10
jmp timer1Routine
reset:
        ldi temp, high(ramend)
        out sph, temp
        ldi temp, low(ramend)
        out spl, temp
        clr temp
        out DDRA, temp          ;oi thures A kai D thures eisodou dedomenwn
        out DDRD, temp
        out PORTA, temp         ;me tautoxrono pull - up twn antistasewn tous
        out PORTD, temp
        ser temp
        out DDRB, temp          ;h thura B thura eksodou dedomenwn
                                //Setarisma anagnwrishs diakopwn tupou INT1
        ldi temp, 0b10000000
        out gicr, temp            ;epitrepsh ekswterikwn diakopwn MONO tupou INT1
        ldi temp, 0b00001100
        out mcucr, temp           ;ALLAGH h diakoph INT1 orizoume na prokaleitai sthn akmh ptwshs
        ldi temp,0b00000101
        out TCCR1B, temp
        sei                     ;Energopoihsh GIE bit tou SR
        ser flag                //shmaia pou apotrepei to na pairnei to pathma tou PA7 perissoterew apo mia
                                //fores.
        clr ledValue
main:
        in portAValue, PINA             ;diavasma thuras eisodou A
        and flag, portAValue
        lsl portAvalue                  ;meleth MSB
        brcc main                       ;MSB = 0? An nai, kane alma sthn main
        sbrc flag, 7                    //an h flag einai 0 agnohse to kaiphgaine apo thn arxh
        rjmp main
        ser flag                       //ksanakane thn flag 1

        rcall ledupdate

        rjmp main                       ;programma diarkous leitourgias

int1Routine:
        rcall ledupdate                 ;energopoihsh maskas diakophs timer 1
        reti

timer1Routine:
        sbrs ledValue,2                 //einai kapoio apo to led anoikto ektos tou 1ou?
        rjmp turnoff                    //an nai tote krathse to 1o led gia akoma 3.5sec anoikto. an oxi kleise ta
```

```
        andi ledValue, 0x01
        //set timer 3.5 seconds
        ldi temp, high(threeAndHalfSec)         ;arxikopoihsh tou TCNT1
        out TCNT1H, temp                        ;gia uperxeilish meta apo 4 sec
        ldi temp, low(threeAndHalfSec)
        out TCNT1L, temp
        ldi temp, 1<<TOIE1
        out timsk, temp

        rjmp timer1Routine_end
turnoff:
        clr ledValue

timer1Routine_end:
        out PORTB, ledValue
        reti                    ;den xreiazetai na epanasetaroume ton xrono treksimatos ths routinas logw tou oti
                                ginetai apo th main kai th diakoph

ledupdate:
        sbrc ledValue,0                 //elegxoume an einai anoikto to 1o led. an den einai tote tha meinei anoikto
gia 4sec, allios tha meinoun ola gia 0,5sec
        rjmp renew

        //set timer 4 seconds
        ldi temp, high(fourSec)         ;arxikopoihsh tou TCNT1
        out TCNT1H, temp                ;gia uperxeilish meta apo 4 sec
        ldi temp, low(fourSec)
        out TCNT1L, temp
        ldi temp, 1<<TOIE1
        out timsk, temp

        ldi ledValue,0x01
        rjmp ledupdate_end
renew:
        // 0.5 seconds
        ldi temp, high(halfSec)         ;arxikopoihsh tou TCNT1
        out TCNT1H, temp                ;gia uperxeilish meta apo 4 sec
        ldi temp, low(halfSec)
        out TCNT1L, temp
        ldi temp, 1<<TOIE1
        out timsk, temp

        ldi ledValue,0xFF
ledupdate_end:
        out PORTB,ledValue
        ret
```