Εργαστήριο Μικρουπολογιστών

Παπαδόπουλος Κωνσταντίνος el20152
Σκούρτης Παύλος el20052

Άσκηση 1A-B

```
.include "m328PBdef.inc"

.equ FOSC_MHZ=16     ;MHz
.equ DEL_mS=500      ;mS
.equ DEL_NU500= FOSC_MHZ*DEL_mS
.equ DEL_NU5= FOSC_MHZ*5
.def interupts = r20

.cseg
   .org 0x00              ;Reset vector
   rjmp reset             ;Jump to reset handler

   .org 0x4               ;Start of interrupt vectors
   rjmp ISR1              ;Jump to INT1 interrupt handler


reset:
   ldi r24,low(RAMEND)
   out SPL,r24
   ldi r24,high(RAMEND)
   out SPH,r24

;init PORTB as output
ser r26
out DDRB, r26     ;PORTB=out
out DDRC, r26     ;PORTC=out
```

```
clr r26
out DDRD,r26      ;PortD == in

ldi r26,(1<<ISC11)|(0<<ISC10)     ;Falling edge generates interrupt
sts EICRA,r26
ldi r26,(1<<INT1)                 ;Configure INT1
out EIMSK,r26                     ;Enable INT1 interrupt

sei                               ;Enable global interrupts

ldi interupts, 0x00
out PORTC, interupts


loop1:
   clr r26
loop2:
   out PORTB, r26
   ldi r24, low(DEL_NU500-1)
   ldi r25, high(DEL_NU500-1)
   rcall delay_mS

   inc r26

   cpi r26, 16
   breq loop1
   rjmp loop2
```

delay_mS:	Η ρουτίνα καθυστέρησης όπως έγινε στην ασκ1
nop	
nop	
ldi r22, 248 ; (1 cycle)	
loop_out:	
dec r22 ; 1 cycle	
nop ; 1 cycle	
brne loop_out ; 1 or 2 cycles	
;total group delay 996 cycles	
delay_inner:	
ldi r23, 249 ; (1 cycle)	
loop_inn:	
dec r23 ; 1 cycle	
nop ; 1 cycle	
brne loop_inn ; 1 or 2 cycles	
sbiw r24 ,1 ; 2 cycles	
brne delay_inner ; 1 or 2 cycles	
ret ;4 cycles	
ISR1:	Ρουτίνα χειρισμού διακοπής

flashover:	Σπινθηρισμός
ldi r17, (1<<INTF1)	
out EIFR, r17	
ldi r24, low(DEL_NU5)	Wait 5ms
ldi r25, high(DEL_NU5)	
rcall delay_mS	
in r17, EIFR	Εάν έχει ισορροπήσει στο λογικό 0 εξυπηρέτησε την διακοπή
sbrc r17, INTF1	
rjmp flashover	Διαφορετικά έχει υπάρξει σπινθηρισμός, οπότε έλεγξε ξανά από την αρχή
in r17,PIND	Εάν έχει πατηθεί το κουμπί PD6, μην κάνεις κάτι
sbrs r17, 6	
rjmp skip2	
inc interupts	Διαφορετικά αύξησε ένα τα led που εμφανίζουν το πλήθος των interrupts
cpi interupts, 32	
breq skip	
out PORTC,interupts	
reti	
skip:	
ldi interupts,0x00	
reti	
skip2:	
nop	
nop	
reti	

Πίνακας 3

.include "m328PBdef.inc"	
.equ FOSC_MHZ=16 ;MHz	
.equ DEL_mS=1000 ;mS	
.equ DEL_NU= FOSC_MHZ*DEL_mS	
.equ DEL_NU5= FOSC_MHZ*50	
.cseg	
.org 0x00 ;Reset vector	
rjmp reset ;Jump to reset handler	
.org 0x2 ;Start of interrupt vectors	
rjmp ISR0	
reset:	
ldi r24,low(RAMEND)	
out SPL,r24	
ldi r24,high(RAMEND)	
out SPH,r24	
ser r26	
out DDRC, r26	;PORTC=out
clr r26	
out DDRB,r26	;PortB == in
ldi r26,(1<<ISC01)|(0<<ISC00) ;	;Falling edge generates interrupt
sts EICRA,r26	
ldi r26,(1<<INT0)	

out EIMSK,r26	
sei	
loop1:	
clr r26	
loop2:	
out PORTC, r26	Μέτρηση στον r26 PORTC
ldi r24, low(DEL_NU-1)	
ldi r25, high(DEL_NU-1)	
rcall delay_mS	
inc r26	
cpi r26, 31	
breq loop1	
rjmp loop2	
delay_mS:	Ρουτίνα καθυστέρησης όπως στην άσκηση 1
nop	
nop	
ldi r22, 248 ; (1 cycle)	
loop_out:	
dec r22 ; 1 cycle	
nop ; 1 cycle	
brne loop_out ; 1 or 2 cycles	
;total group delay 996 cycles	
delay_inner:	
ldi r23, 249 ; (1 cycle)	
loop_inn:	

dec r23 ; 1 cycle	
nop ; 1 cycle	
brne loop_inn ; 1 or 2 cycles	
sbiw r24 ,1 ; 2 cycles	
brne delay_inner ; 1 or 2 cycles	
ret ;4 cycles	
ISR0:	Εάν έχω διακοπή
push r25	Σώζω καταχωρητές 24-25
push r24	
in r17,PINB	R17 —> είσοδος
ldi r18,0x05	
ldi r19,0x00	
loop3:	
ror r17	C —> το LSB της εισόδου + rotate right
brcc syn	Εάν είναι 1 συν
rjmp meion	Αλλιώς μείων
syn:	Για κάθε 1 πατημένο κουμπί
inc r19	Αύξησε τον r19
meion:	Αλλιώς μη κανείς κάτι
dec r18	Επανάλαβε 5 φορές
brne loop3	
ldi r20,0x00	
cpi r19,0	Εάν δεν έχει πατηθεί κανένα κουμπί (r19=0) έξοδος
breq exit	

loop4:	Αλλίως
seC	Μέσω του κρατουμένου κάνε δεξιά περιστροφή και κάνε το πρώτο ψηφίο 1
rol r20	
dec r19	
brne loop4	
exit:	
out PORTC,r20	Εμφάνισε έξοδο
ldi r24, low(DEL_NU5-1)	
ldi r25, high(DEL_NU5-1)	
rcall delay_mS	
pop r24	
pop r25	
reti	

Άσκηση 3 σε c

#define F_CPU 16000000UL	
#include <avr/io.h>	
#include <avr/interrupt.h>	
#include <util/delay.h>	
int flag=0;	
ISR (INT1_vect){	Ρουτίνα διακοπής
flag=1;	Υπάρχει διακοπή μόνο όταν flag=1
}	
void on(){	Ρουτίνα ανοίγματος όλων των led (reset)
sei();	Enable interrupts
flag=0;	
PORTB=0x1F;	Άνοιξε όλα τα led
int i;	
for(i=0; i<10; i++){	Καθυστέρησε συνολικά 500ms (ωστε να είναι τα ολα led ανοιχτά)
_delay_ms(50);	
if (flag==1){on(); return;}	Εάν ξαναπατηθεί διακοπή, κάνε την ρουτίνα on και επέστρεψε, προκειμένου να μην υπάρξει συσσώρευση του χρόνου reset
}	
}	
int main()	
{	
EICRA=(1 << ISC10)|(1 << ISC11); // interrupt on every INT1	Αρχικοποίηση των διακοπών
EIMSK = (1 << INT1);	
sei();	

DDRB= 0xFF; //PORTB as output	
PORTB=0x00; //start with LEDs off	
int i;	
while(1){	
if (flag==0)PORTB=0x00;	Εάν flag= 0 μη κάνεις κάτι (ξαναέλγξε για αλλαγή του)
if (flag == 1){	Αλλιώς πατήθηκε διακοπή
_delay_ms(10);	
flag=0;	Πέσε το flag=0
_delay_ms(10);	
PORTB=0x01;	Άνοιξε 1 led
PORTC = flag;	
for(i=0; i<50; i++){	Για 3 sec κράτα ανοιχτό το 1 led
//PORTB=0x0F;	
_delay_ms(60);	
if (flag==1) {on(); flag=1; break;}	Εάν ξαναπατηθεί διακοπή, κάνε την ρουτίνα on θέσε το flag=1 και ξαναάρχισε τα 3 sec
}	
}	
}	
}	

Άσκηση 3 σε assembly

```
.include "m328PBdef.inc"

.equ FOSC_MHZ=16      ;MHz
.equ DEL_mS1=30       ;mS
.equ DEL_mS2=50=      ;mS
.equ DEL_NU1= FOSC_MHZ*DEL_mS1
.equ DEL_NU2= FOSC_MHZ*DEL_mS2
.equ DEL_NU5= FOSC_MHZ*500
.def led = r16
.DEF ledall= r17
.DEF counter=r18
.def flag= r19
.cseg
   .org 0x00          ;Reset vector
   rjmp reset         ;Jump to reset handler

   .org 0x4           ;Start of interrupt vectors
   rjmp ISR1          ;Jump to INT1 interrupt handler

reset:
   ldi r24,low(RAMEND)
   out SPL,r24
   ldi r24,high(RAMEND)
   out SPH,r24

ldi flag,0x00         ;PORTB=out
```

;init PORTB as output	**;PortD == in**
ser r26	
out DDRB, r26 ;PORTB=out	**;Falling edge generates interrupt**
clr r26	
out DDRD,r26 ;PortD == in	
ldi r26,(1<<ISC11)|(0<<ISC10) ;Initialize external interrupt 1 (INT1)	**;Enable global interrupts**
sts EICRA,r26 ;Falling edge generates interrupt	
ldi r26,(1<<INT1) ;Configure INT1	
out EIMSK,r26 ;Enable INT1 interrupt	
sei ;Enable global interrupts	
ldi ledall,0xff	
ldi led,0x00	
main_loop:	
cpi flag,0	Εάν flag=1 open (πατήθηκε διακοπή)
brne open	
out PORTB,led	Σβήσε τα led εξόδου
rjmp main_loop	
open:	
ldi led,0x01	Άνοιξε το πρώτο led
out PORTB,led	
ldi r24, low(DEL_NU1-1)	
ldi r25, high(DEL_NU1-1)	
rcall delay_mS	
dec counter	Καθυστέρηση 100 επά 30ms = 3sec

brne open	
ldi flag,0x00	
ldi led,0x00	
out PORTB, led	
rjmp main_loop	
ISR1:	
push r25	Σώσε καταχωρητές εισόδου
push r24	
in r20, SREG	
push r20	
cpi flag,0	Εάν flag=1 σημαίνει ότι ξαναπατήθηκε διακοπή και άρα άναψε όλα τα led
brne all	
ldi flag,0x01	Διαφορετικά πέσε το flag 1
ldi counter,0x64	Θέσε τον μετρητή 100
pop r20	Έξοδος
out SREG, r20	
pop r24	
pop r25	
reti	
all:	Διαφορετικά άναψε όλα τα led (κώδικας αντίστοιχος της c)
out PORTB,ledall	
ldi r24, low(DEL_NU2-1)	
ldi r25, high(DEL_NU2-1)	
rcall delay_mS	
ldi counter,0x64	
pop r20	

```
    out SREG, r20
    pop r24
    pop r25
    reti

delay_mS:
    ldi r23, 249
loop_inn:
    dec r23          ; 1 cycle
    nop              ; 1 cycle
    brne loop_inn    ; 1 or 2 cycles

    sbiw r24 ,1      ; 2 cycles
    brne delay_mS    ; 1 or 2 cycles

    ret              ;4 cycle
```