Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Εθνικό Μετσόβιο Πολυτεχνείο

Συστήματα Μικροϋπολογιστών

Έκτη ομάδα ασκήσεων

Σπουδαστές
Κατσάμπουλα Χριστίνα Σοφία (Α.Μ.: 03114910)
Παπασκαρλάτος Αλέξανδρος (Α.Μ.: 03111097)

Ημερομηνία Υποβολής Αναφοράς: 2 Ιουλίου 2018

Άσκηση 1

```
.include “m16def.inc”
.def temp = r16
.def delay = r15                ; Πόσες φορές θα κληθεί η ρουτίνα Delay10

clr temp                        ; PortD είσοδος
out DDRD, temp

ser temp
out DDRB, temp                  ; B έξοδος
out PORTD, temp                 ; Ενεργοποίηση pull-up αντιστάσεων (αν χρειάζεται)

eternal_loop:
        ldi delay, 50           ; if MSB(PORTD)=1 then delay=50 else delay=150
        sbis PIND, 7
        ldi delay, 150

        ser temp                ; Ανάβουμε τα LEDs
        out PORTB, temp
        mov temp, delay
```

```
del1:                       ; Καθυστέρηση ίση με delay*10ms
        rcall Delay10       ; δηλαδή: 50*10ms = 0.5 sec αν, αρχικά, delay = 50
        dec delay           ; 150*10ms = 1.5 sec αν, αρχικά, delay = 150
        brne del1
        out PORTB, delay    ; σβήσιμο των leds (βγαίνοντας από το loop, είναι σίγουρα
                            ; delay=0x00)
        ldi delay,200   ; Στις δύο αυτές γραμμές, κάνουμε delay = 200 - temp
                        ; με (temp = παλιά τιμή του delay)
        sub delay,temp
                        ; Αν temp=50, τότε delay=150, ενώ αν temp=150, delay=50

del2:                       ; Καθυστέρηση ίση με 0.5 sec ή 1.5 sec
        rcall Delay10
        dec delay
        brne del2
        rjmp eternal_loop   ; επιστροφή στην αρχή

Delay10:            ; Κατασκευή χρονοκαθυστέρησης 10msec, με την υπόθεση
                    ; λειτουργίας του ρολογιού του επεξεργαστή στη συχνότητα των
                    ; 4 MHz
        ldi r24,0xF0    ; Ανάθεση της δεκαδικής τιμής: 55536 στο ζεύγος καταχωρητών,
        ldi r25,0xD8    ; διαδικασία που απαιτεί 2 κύκλους ρολογιού.

delay_loop:         ; Κάθε επανάληψη, ΕΚΤΟΣ της τελευταίας εκτελείται σε 4
                    ; κύκλους, ενώ η τελευταία σε 3 κύκλους λόγω του ότι δε
                    ; γίνεται διακλάδωση ροής.
        adiw r24,1
        brne delay_loop
        ret
```

Άσκηση 2

Πρόγραμμα σε assembly

```
.include "m16def.inc"  ;Ορισμός μικροελεγκτή AVR ATmega16
.def temp=r10
.def A=r11
.def AN=r12
.def B=r13
.def CN=r14
.def D=r15
.def DN=r16
.def E=r17
.def G=r18
.def F0=r19
.def F1=r20
.def F2=r21

orismosE/E:
clr temp
out DDRD,temp       ;Η θύρα D ορίζεται ως είσοδος
ser temp
out PORTD,temp      ;Ενεργοποίηση pull-up αντιστάσεων (ΜΗ απαραίτητο)
out DDRB,temp       ;Η θύρα B ορίζεται ως έξοδος


log_var:
in temp,PIND        ;Διάβασμα εισόδου

mov A,temp          ;Το A στο LSB του καταχωρητή A
mov AN,temp
com AN
lsr temp
mov B,temp          ;Το B στο LSB του καταχωρητή B
lsr temp
mov CN,temp         ;Το C στο LSB του καταχωρητή C
com CN              ;συμπληρώματος C'
lsr temp
mov D,temp          ;Το D στο LSB του καταχωρητή D
mov DN,D            ;Δημιουργία του
com DN              ;συμπληρώματος D'
lsr temp
mov E,temp          ;Το E στο LSB του καταχωρητή E
lsr temp
mov G,temp          ;Το G στο LSB του καταχωρητή F
```

```
log_func:
mov F0,A            ;F0=A
mov temp,B          ;temp=B
and temp,CN         ;temp=BC’
and temp,DN         ;temp=BC’D’
or F0,temp          ;F0=A+BC’D’
and F0,0x01         ;Απομόνωση του LSB (F0=0000000?), με ?=0 ή 1

mov F1,AN
and F1,B            ;F1=A’B
and F1,CN           ;F1=A’BC'
and F1,D            ;F1=A’BC'D
mov temp,E          ;temp=E
and temp,G          ;temp=EG
or F1,temp          ;X1=A’BC'D+EG
and F1,0x01         ;Απομόνωση του LSB

mov F2,F0
and F2,F1           ;F2=F0 F1 (F2=0000000?)

Eksodos:
mov temp,F2         ;Βάζουμε τα F2,F1,F0 στον temp
lsl temp            ;Στις σωστές θέσεις
or temp,F1
lsl temp
or temp,F0
lsl temp
lsl temp
lsl temp
lsl temp
lsl temp

out PORTB,temp      ;Και εξάγουμε στον B

 rjmp log_var       ;Πρόγραμμα διαρκούς λειτουργίας
```

Πρόγραμμα σε C

```
#include <mega16.h> //Φόρτωση κατάλληλου αρχείου κεφαλίδας
unsigned char temp, A, B, C, D, E, G, F0, F1, F2;

void main(void)
{
orismosE/E:
DDRD = 0x00;          //Η θύρα D ορίζεται ως είσοδος
PORTD = 0xFF;         //Ενεργοποίηση pull-up αντιστάσεων (ΜΗ απαραίτητο)
DDRB = 0xFF;          //Η θύρα B ορίζεται ως έξοδος

while (1)             //Ατέρμων βρόγχος (Η συνθήκη είναι πάντα αληθής)
{
log_var:
temp = PIND;  //Διάβασμα εισόδου

A = temp;       //Το A στο LSB της μεταβλητής A
temp >> 1;
B = temp;       //Το B στο LSB της μεταβλητής B
temp >> 1;
C = temp;       //Το C στο LSB της μεταβλητής C
temp >> 1;
D = temp;       //Το D στο LSB της μεταβλητής D
temp >> 1;
E = temp;       //Το E στο LSB της μεταβλητής E
temp >> 1;
G = temp;       //Το F στο LSB της μεταβλητής G


log_func:
F0 = (A) | (B & ~C & ~D);       //F0 = A + B C' D'
F0 = F0 & 0x01;                 //Απομόνωση του LSB (X0=0000000?), με ?=0 ή 1
F1 = (~A & B & ~C & D) | (E & G);     //F1=A' B C' D + E G
F1 = F1 & 0x01;                       //Απομόνωση του LSB
F2 = X0 & X1;                         //F2=F0 F1 (X2=0000000?)

Eksodos:
F0 << 5
F1 << 6;        //Τοποθέτηση στην κατάλληλη θέση
F2 << 7;
temp = F0 | F1 | F2;
PORTB = temp ;
};
}
```

Άσκηση 3

```
.include "m16def.inc" ;Ορισμός μικροελεγκτή AVR ATmega16
.def temp=r10
.def input=r11
.def output=r12
.def output2=r13      ;Θα χρησιμεύσει για να κρατάμε την τιμή για τις περιστροφές

OrismosE/E:
clr temp
out DDRD,temp         ;Η θύρα D ορίζεται ως είσοδος
ser temp
out PORTD,temp        ;Ενεργοποίηση pull-up αντιστάσεων (ΜΗ απαραίτητο)
out DDRB,temp         ;Η θύρα B ορίζεται ως έξοδος
ldi output2,0xFE
out PORTB,output2     ;PORTB=1111 1110

main:
in input,PIND
lsl input
lsl input
lsl input

sw4:
lsl input
brcs sw3              ;Εάν PIND=XXX1 XXXX, τότε κάνε άλμα στην sw3
ldi output,0xFF       ;Αλλιώς
out PORTB,output      ;PORTB=1111 1111
rjmp main

sw3:
lsl input
brcs sw2              ;Εάν PIND=XXX1 1XXX, τότε κάνε άλμα στην sw2
rol output2           ;Αλλιώς, κάνε αριστερή περιστροφή των bits
out PORTB,output2
call Delay500         ;με χρονοκαθυστέρηση 0,5s ώστε να γίνει αισθητή η περιστροφή
rjmp main

sw2:
lsl input
brcs sw1              ;Εάν PIND=XXX1 11XX, τότε κάνε άλμα στην sw1
bst output2,0         ;(Υπερβολή που μάλλον δε ζητείται)
ror output2           ;Αλλιώς, κάνε δεξιά περιστροφή των bits
bld output2,7         ;(Δεύτερο μέρος υπερβολής)
out PORTB,output2
call Delay500         ;με χρονοκαθυστέρηση 0,5s ώστε να γίνει αισθητή η περιστροφή
rjmp main
```

```
sw1:
lsl input
brcs sw0            ;Εάν PIND= XXX1 111X, τότε κάνε άλμα στην sw0
ldi output,0x0F     ;Αλλιώς, άναψε τα led 4-7 και σβήσε τα led 0-3
out PORTB,output
rjmp main

sw0:
brcs main           ;Εάν PIND=XXX1 1111, τότε κάνε άλμα στην MAIN
ldi output,0xF0     ;Αλλιώς, άναψε τα led 0-3 και σβήσε τα led 4-7 (αρνητικής λογικής)
out PORTB,output
rjmp main
```

Άσκηση 4

Πρόγραμμα σε assembly

```
.include “m16def.inc”
.def temp=r15
.def zero=r16
.def timer=r17
.def output=r18

reset:
ldi zero,0x00              ;Κύριο πρόγραμμα
ldi temp,high(RAMEND)
out SPH,temp               ;Θέτουμε το δείκτη στοίβας στη RAM
ldi temp,low(RAMEND)
out SPL,temp
ldi temp,0xFF
out DDRC,temp              ;Η θύρα C ορίζεται ως έξοδος
ldi temp,1<<INT1           ;10000000 (INT1=7)
out GIMSK,temp             ;Ενεργοποίηση εξωτερικής διακοπής 1
ldi temp,0b00001100              ;Η διακοπή INT1 προκαλείται στην
out MCUCR,temp             ;ακμή ανόδου του σήματος
SEI                        ;Ενεργοποίηση των διακοπών

LOOP:
rjmp loop                  ;Αναμονή εξωτερικής διακοπής

interrupt1:
ldi timer,0x05             ;Timer=counter (5 φορές)

loop_counter:              ;Αναβοσβήνουμε τα LEDs 5 φορές
clr output
out PORTC,output
call Delay
ser output
out PORTC,output
dec timer
brne loop_counter

ldi timer, 120             ;Αρχικοποίηση του timer στο 120 (120*0.5=60)
ldi output,0x00            ;Άναμμα των LEDs
out PORTC,output

delay_120:                 ;Καθυστέρηση ίση με 60 sec περίπου. Αν ωστόσο
call Delay                 ;προκληθεί ξανά διακοπή, αυτή θα εκτελεστεί
dec timer                  ;κανονικά, κρατώντας αναμμένα τα LEDs για
cpse timer,zero            ;60 sec, όπως θα θέλαμε, ενώ, όταν επιστρέψει
rjmp delay_120             ;ο καταχωρητής timer θα έχει την τιμή 0, οπότε θα τερματιστεί
                           ;αμέσως, χωρίς ανεπιθύμητη καθυστέρηση
ldi output,0b11111111      ;Σβήσιμο των LEDs μετά τα
out PORTC,output           ;60s.
RETI
```

Πρόγραμμα σε C

```
#include <mega16.h> //Φόρτωση κατάλληλου αρχείου κεφαλίδας
int timer;
INTERRUPT [EXT_INT1] VOID EXT_INT1_ISR(void){
        timer=5
        while (timer>0){                //Ανάβω-σβήνω τα LEDs 5 φορές
                PORTA=0x00;
                Delay();
                PORTA=0xFF;
                timer--;
        };

        timer=120;
        PORTA=0x00;         //Άναμμα των LEDs
        while (timer>0){
                Delay();    //Καθυστέρηση 0.5 sec
                timer--;
        };
        PORTA=0xFF;         //Σβήσιμο των LEDs μετά από 60 (=120*0.5) sec
}

void main(void){
        DDRA=0xFF;          //Η θύρα Α ορίζεται ως έξοδος
        PORTA=0x00;
        GIMSK=0x80;         //Ενεργοποίηση της διακοπής
        INT1
        MCUCR=0x08;         //Η διακοπή ενεργοποιείται στην ακμή πτώσης
        #ASM("SEI")         //Ενεργοποίηση συνολικά των διακοπών
        while (1){          //Αναμονή διακοπής
        };
}
```