Συστήματα Μικρουπολογιστών

4η Ομάδα Ασκήσεων

Μετζάκης Ιωάννης

Α.Μ.: 03116202

ΣΗΜΜΥ 8ο

1η ΑΣΚΗΣΗ

Εδώ τον έλεγχο του αν έχω φτάσει σε κάποια γωνία, τον κάνω μέσω ενός μετρητή, όπου ξεκινώντας από 7(θέση LSB), μειώνεται κατά 1 για κάθε shift αριστερά, και ελέγχω αν έχει γίνει 0(θέση MSB), όπου αλλάζω κατεύθυνση και αντίστοιχα αυξάνω τον μετρητή κατά 1 σε κάθε shift.
Τα breakpoints τα έχω βάλει για την προσομοίωση στο AVR Studio.

Παρακάτω ο κώδικας σε **assembly** με τα απαραίτητα σχόλια:

```
 .INCLUDE "m16def.inc"

.def counter=r20
.def input=r23
.org 0

reset:
    ldi r24,low(RAMEND)     ;itilialize stack pointer
    out SPL,r24
    ldi r24,high(RAMEND)
    out SPH,r24
    clr r27

main:
    clr r28
    out DDRC,r28            ;portC input
    ser r28
    out DDRB,r28            ;portB output

    ldi r27,0x01
    out PORTB,r27
    ldi counter,7           ;set counter value 7


left:
    in input,PINC           ;diavazw eisodo
    andi input,0x04
    cpi input,0x04
    breq left

    clc
    rol r27
    out PORTB,r27
    dec counter
    cpi counter,0       ;an 0 exw ftasei gwnia, opote phgaine right
    breq right
    rjmp left

right:
    in input,PINC
    andi input,0x04
    cpi input,0x04
    breq right
    clc
    ror r27
    out PORTB,r27
    inc counter
    cpi counter,7
    breq left
    rjmp right
```

2η ΑΣΚΗΣΗ

Εδώ έχω απλά υλοποιήσει τις λογικές συναρτήσεις της εκφώνησης, χρησιμοποιώντας την XOR με το 1, για να υλοποιήσω την NOT.

Παρακάτω ο κώδικας σε **assembly** με τα απαραίτητα σχόλια:

```
.INCLUDE "m16def.inc"

.def input=r18
.def A=r20
.def B=r21
.def C=r22
.def D=r23
.def F0=r24
.def F1=r25

main:
    ldi r27,1
    clr r28
    out DDRB,r28        ;portB for input
    ser r28
    out DDRA,r28        ;portA for output

    in input,PINB
    mov A,input     ;A
    andi A,0x01

    mov B,input     ;B
    andi B,0x02
    ror B

    mov C,input     ;C
    andi C,0x04
    ror C
    ror C

    mov D,input     ;D
    andi D,0x08
    ror D
    ror D
    ror D

    mov F0,A
    mov r29,B       ;F0
    eor r29,r27
    and F0,r29
    mov r29,C
    eor r29,r27;    ;Xor gia to not
    and r29,D
    and r29,B
    or F0,r29
    eor F0,r27

    mov F1,A        ;F1
    or F1,C
    mov r29,B
    or r29,D
    and F1,r29

    clc
    rol F1
    add F0,F1
    out PORTA,F0

    rjmp main
```

3η ΑΣΚΗΣΗ

Εδώ ελέγχω κάθε φορά αν έχει πατηθεί κάποιος από τους 4 ενδιαφερόμενους διακόπτες, και σε αυτή την περίπτωση, μένω σε μία while λούπα μέχρις ότου να αφηθεί ο διακόπτης, όπου και προχωράω στην επιθυμητή κάθε φορά λειτουργία.

Παρακάτω ο κώδικας σε **c** με τα απαραίτητα σχόλια:

```c
#include <avr/io.h>
unsigned char x;

int main(void){
    DDRB = 0xff;    //portB output
    DDRA = 0x00;    //portA input

    x = 1;
    PORTB = x;
    while(1){
        if((PINA & 0x01) == 1){ //1o
            while((PINA & 0x01) == 1){} //wait to release
            if(x == 0x01){
                x = 0x80;
            }
            else{
                x = x>>1;
            }
        }
        if((PINA & 0x02) == 2){ //2o
            while((PINA & 0x02) == 2){}

            if(x == 0x80){
                x = 0x01;
            }
            else{
                x = x<<1;
            }
        }
        if((PINA & 0x04) == 4){ //3o
            while((PINA & 0x04) == 4){}
            x = 0x01;
        }
        if((PINA & 0x08) == 8){ //4o
            while((PINA & 0x08) == 8){}
            x = 0x80;
        }
        PORTB = x;
    }
    return 0;
}
```

Διευκρίνιση:

Παρά το ότι στις ομάδες στο mycources η ομάδα που είμαι εγγεγραμένος (44) ήμασταν αρχικά 3 άτομα, το ένα είχε εν τέλη πραγματοποιήσει το περυσινό εργαστήριο, οπότε και βγήκε από την ομάδα, και με το άλλο ενώ προσπάθησα να επικοινωνήσω μαζί του, δεν τα κατάφερα, αλλά για κάποιο λόγο είναι ακόμα εγγεγραμένος στην ομάδα.
Συνεπώς, η αναφορά αυτή είναι ατομική, όπως και η επίδειξη της άσκησης στις 28/5, όπου εξετάστηκα προσωπικά.
Τέλος, όλοι οι κώδικες της παρούσας αναφοράς, επισυνάπτονται στο παραδοτέο zip αρχείο.