**Εργαστήριο Προηγμένοι Μικροεπεξεργαστές**

Εργαστηριακή Άσκηση 1

Σκάγκος Ιωάννης 1072611

Στεργίου Γεώργιος 1072503

**Ερώτημα 1**

Διάγραμμα Ροής:

Εικόνα που περιέχει διάγραμμα

Περιγραφή που δημιουργήθηκε αυτόματα

Κώδικας ερωτήματος 1:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#define ped 10

int strofes=0;

int x=1;

int main(){

PORTD.DIR |= (PIN1\_bm|PIN2\_bm|PIN0\_bm); //PIN is output

//initialize the ADC for Free-Running mode

ADC0.CTRLA |= ADC\_RESSEL\_10BIT\_gc; //10-bit resolution

ADC0.CTRLA |= ADC\_FREERUN\_bm; //Free-Running mode enabled

ADC0.CTRLA |= ADC\_ENABLE\_bm; //Enable ADC

ADC0.MUXPOS |= ADC\_MUXPOS\_AIN7\_gc; //The bit //Enable Debug Mode

ADC0.DBGCTRL |= ADC\_DBGRUN\_bm; //Window Comparator Mode

ADC0.WINLT |= 10; //Set threshold

ADC0.INTCTRL |= ADC\_WCMP\_bm; //Enable Interrupts for WCM

ADC0.CTRLE |= ADC\_WINCM0\_bm; //Interrupt when RESULT < WINLT

sei();

ADC0.COMMAND |= ADC\_STCONV\_bm; //Start Conversion

while(x==1){

if (strofes>=4)

{

x=2;

}

PORTD.OUTCLR = PIN1\_bm;

sei();

}

}

ISR(ADC0\_WCOMP\_vect){

cli();

int intflags = ADC0.INTFLAGS;

ADC0.INTFLAGS = intflags;

PORTD.OUT|= PIN1\_bm;

PORTD.OUTCLR= PIN2\_bm;

TCA0.SINGLE.CNT = 0; //clear counter

TCA0.SINGLE.CTRLB = 0;

TCA0.SINGLE.CMP0 = ped;

TCA0.SINGLE.CTRLA = 0x7<<1; //TCA\_SINGLE\_CLKSEL\_DIV1024\_gc σελ 224

TCA0.SINGLE.CTRLA |=1;//Enable

TCA0.SINGLE.INTCTRL = TCA\_SINGLE\_CMP0\_bm;

strofes=strofes+1;

sei();

}

ISR(TCA0\_CMP0\_vect){

cli();

TCA0.SINGLE.CTRLA = 0; //Disable

int intflags = TCA0.SINGLE.INTFLAGS;

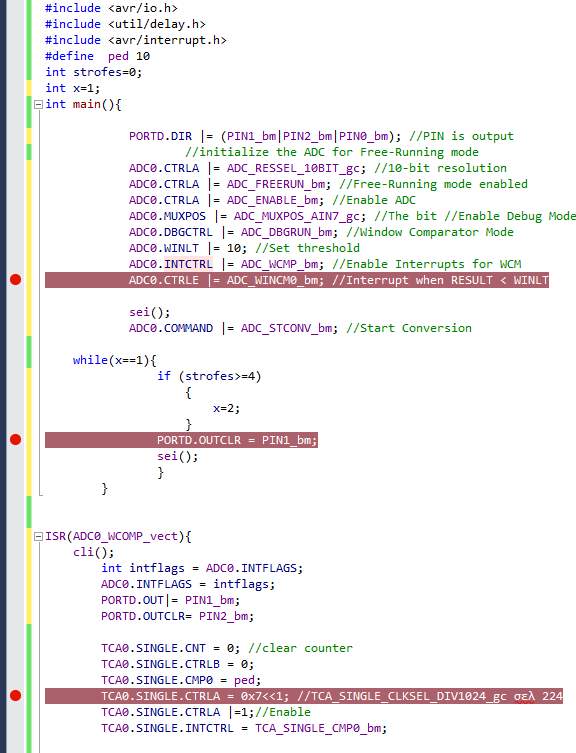
TCA0.SINGLE.INTFLAGS=intflags;

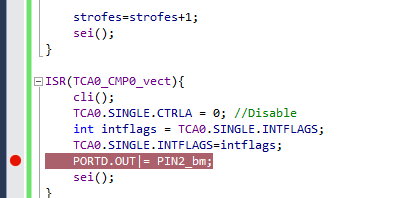
PORTD.OUT|= PIN2\_bm;

sei();

}

Screenshots με breakout points:





Σχόλια – Επεξήγηση κώδικα:

Αρχικά θέτουμε ως εξόδους τα PIN1,PIN2,PIN0 (Όπου PIN1 συμβολίζει την ευθέια πορέια της οικιακής συσκευής, PIN0 την δεξία κίνηση και PIN2 την αριστερή κίνηση). Θετουμε σε λειτουργία τον ADC σε free-runninig mode με σύγκριση low και τιμή κατωφλιού 10.

Έπειτα εκτελείται μια while που ανάβει το PIN1 (ευθεία πορεία) για x=1 το οποίο αλλάζει (x=2;) μόνο όταν η μεταβλητή strofes πάρει τιμή μεγαλύτερη ή ίση του 4 (τετράγωνο δωμάτιο) με σκοπό να τερματίσει την λειτουργία της οικιακής συσκευής (για strofes=4). Όταν ο καταχωρητής RES πάρει τιμή μικρότρη του 10 τότε το πρόγραμμα μπαίνει στην ISR του ADC σβήνει το PIN1 (τέλος ευθείας πορείας) και ανάβει το PIN2 (έναρξη αριστερής στροφής) επίσης θέτει σε λειτουργία τον timer και αυξάνει την τιμή της μεταβλητής strofes κατά 1.

Στην αρχη της ρουτίνας διακοπής του ADC απενεργοποιούμε τις διακοπές ώστε να προλάβουν να εκτελεστούν όλες οι εντολές το ISR πριν λήξει ο timer. Τέλος στην ISR του timer σβήνει το PIN2 (τελειώνει η αριστερή στροφή).

**Ερώτημα 2**

Διάγραμμα Ροής:

Εικόνα που περιέχει διάγραμμα

Περιγραφή που δημιουργήθηκε αυτόματα

Κώδικας ερωτήματος 2:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#define ped 10

void mprosta();

void aristeristrofh();

void dexiastrofh();

int strofes=0;

int dexiaFlag=0;

int adcFlag=0;

int tcaFlag=0;

int tcaFlagg=0;

int main()

{

PORTD.DIR |= (PIN1\_bm|PIN2\_bm|PIN0\_bm); //PIN is output

PORTD.OUT = (PIN1\_bm|PIN2\_bm|PIN0\_bm);

sei();

while (strofes<8){

TCA0.SINGLE.CNT = 0; //clear counter

//Normal Mode (TCA\_SINGLE\_WGMODE\_NORMAL\_gc σελ 207)

TCA0.SINGLE.CTRLB = 0;

//When CMP0 reaches this value -> interrupt //CLOCK FREQUENCY/1024

TCA0.SINGLE.CMP0 = ped;

TCA0.SINGLE.CTRLA = 0x7<<1; //TCA\_SINGLE\_CLKSEL\_DIV1024\_gc σελ 224

TCA0.SINGLE.CTRLA |=1;//Enable

TCA0.SINGLE.INTCTRL = TCA\_SINGLE\_CMP0\_bm;

ADC0.CTRLA = ADC\_RESSEL\_10BIT\_gc; //10-bit resolution

ADC0.CTRLA = ADC\_FREERUN\_bm; //Free-Running mode enabled

ADC0.CTRLA = ADC\_ENABLE\_bm; //Enable ADC

ADC0.MUXPOS = ADC\_MUXPOS\_AIN7\_gc; //The bit Enable Debug Mode

ADC0.DBGCTRL = ADC\_DBGRUN\_bm; //Window Comparator Mode

ADC0.WINLT = 10; //Set threshold

ADC0.INTCTRL = ADC\_WCMP\_bm; //Enable Interrupts for WCM

ADC0.CTRLE = ADC\_WINCM\_ABOVE\_gc; //Interrupt when RESULT WINLT

ADC0.COMMAND = ADC\_STCONV\_bm; //Start Conversion

mprosta();

*\_delay\_ms*(3000);

if(dexiaFlag)

{

dexiastrofh();

strofes=strofes+1;

}

ADC0.CTRLA = 0;

ADC0.CTRLA = ADC\_RESSEL\_10BIT\_gc; //10-bit resolution

ADC0.CTRLA = ADC\_ENABLE\_bm; //Enable ADC

ADC0.MUXPOS = ADC\_MUXPOS\_AIN7\_gc; //The bit

ADC0.DBGCTRL = ADC\_DBGRUN\_bm; //Window Comparator Mode

ADC0.WINLT = 10;

ADC0.INTCTRL = ADC\_WCMP\_bm;

ADC0.CTRLE = ADC\_WINCM\_BELOW\_gc;//Interrupt when RESULT WINLT

ADC0.COMMAND = ADC\_STCONV\_bm; //Start Conversion

mprosta();

*\_delay\_ms*(2000);

if(adcFlag)

{

aristeristrofh();

strofes=strofes+1;

}

}

}

void mprosta()

{

PORTD.OUT |=PIN0\_bm;

PORTD.OUT |=PIN2\_bm;

PORTD.OUTCLR = PIN1\_bm;

}

void aristeristrofh()

{

PORTD.OUT |=PIN0\_bm;

PORTD.OUT |=PIN1\_bm;

PORTD.OUTCLR = PIN2\_bm;

while (tcaFlag==0) {

}

tcaFlag=0;

adcFlag=0;

}

void dexiastrofh()

{

PORTD.OUT |=PIN1\_bm;

PORTD.OUT |=PIN2\_bm;

PORTD.OUTCLR = PIN0\_bm;

while (tcaFlagg==0) {

}

tcaFlagg=0;

dexiaFlag=0;

}

ISR(ADC0\_WCOMP\_vect)

{

int intflags = ADC0.INTFLAGS;

ADC0.INTFLAGS = intflags;

if (ADC0.RES > ADC0.WINLT)

{

dexiaFlag=1;

}

if (ADC0.RES < ADC0.WINLT)

{

adcFlag=1;

}

}

ISR(TCA0\_CMP0\_vect){

TCA0.SINGLE.CTRLA = 0; //Disable

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS=intflags;

if (tcaFlag==0)

{

tcaFlag=1;

}

if (tcaFlagg==0)

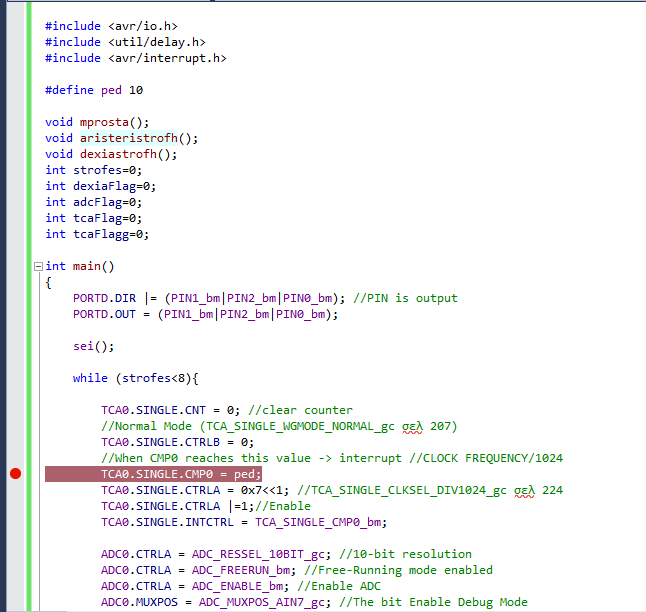
{

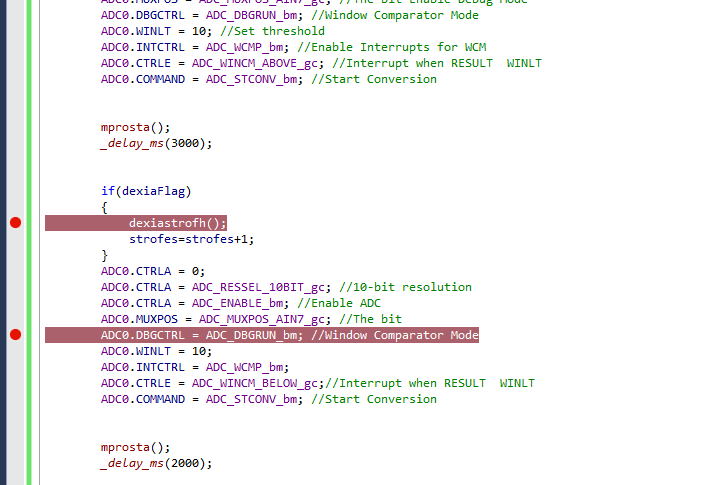
tcaFlagg=1;

}

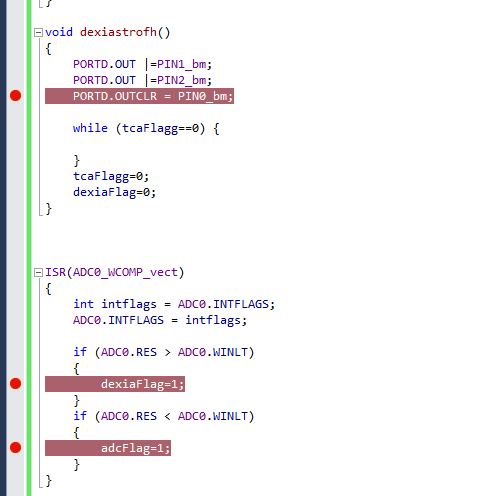
}

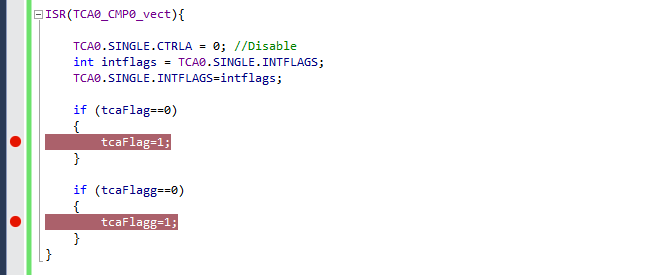
Screenshots με breakout points:











Σχόλια – Επεξήγηση κώδικα:

Αφού το δωμάτιο εχει 2 αμβλείες γωνίες το προγραμμα θα εκτελείτε μονο εφόσον οι strofes < 8.

Ορίζουμε timer που μετρά την διάρκεια των στροφών. Όταν λήξη η μέτρηση του ενεργοποιεί την ρουτινα διακοπής η οποία αλλάζει την τιμή όποιου tca flag είναι απενεργοποιημένο σε ενεργοποιημένο.

Σε πρώτη φάση θετουμε σε λειτουργία τον ADC σε free-runninig mode με σύγκριση high και τιμή κατωφλιού 10 και η συνάρτηση mprosta εκτελείτε μονο αν κανένα flag δεν είναι ενεργό δηλαδή μόνο όσο ο ADC δεν εχει ανιχνεύσει τιμή RES>10.

Το dexiaFlag ενεργοποιήται όταν ο ADC δεχτεί τιμή RES μεγαλύτερη από 10. Εξαιτίας της ενεργοποίησης του dexiaFlag το πρόγραμμα εισάγετε στην συνάρτηση dexiastrofh() όπου ανάβει το PIN0 (έναρξη δεξιας στροφής) και σβήνει τα υπόλοιπα. Περιμένει να ενεργοποιήθεί το flag tcaFlagg (μεχρι να λήξει ο timer που μετρά την διάρκεια της στροφής) και μετά κάνει reset τα flags για να συνεχίσει το πρόγραμμα. Τέλος αυξάνει την τιμή της μεταβλητής strofes κατά +1.

Σε δεύτερη φάση αλλάζει η λειτουργία του ADC σε single mode και πραγματοποιεί μόνο μια συγκριση για τα επόμενα 2 sec.

Το adcFlag ενεργοποιήται όταν ο ADC δεχτεί τιμή RES μικρότερη από 10. Εξαιτίας της ενεργοποίησης του adcFlag το πρόγραμμα εισάγετε στην συνάρτηση aristeristrofh() όπου ανάβει το PIN2 (έναρξη αριστερής στροφής) και σβήνει τα υπόλοιπα. Περιμένει να ενεργοποιήθεί το flag tcaFlag (μεχρι να λήξει ο timer που μετρά την διάρκεια της στροφής) και μετά κάνει reset τα flags για να συνεχίσει το πρόγραμμα.Τέλος αυξάνει την τιμή της μεταβλητής strofes κατά +1.

Παρατήρηση:

Καθώς δεν μπορέσαμε να πραγματοποιήσουμε μέτρηση με TIMER για την αλλαγή λειτουργίας του ADC όπως φένεται στο διαγραμμα ροής (timers T1,T2) για το ερώτημα 2 χρησημοποιήσαμε την εντολή delay για τα αντίστοιχα χρονικά διαστήματα που ζητούνται από την εκφώνηση.