**Εργαστήριο Προηγμένοι Μικροεπεξεργαστές**

Εργαστηριακή Άσκηση 4

Σκάγκος Ιωάννης 1072611

Στεργίου Γεώργιος 1072503

**Ερώτημα 1**

Κώδικας ερωτήματος 1:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

int sw5=0;

int sw6=0;

int flag=0;

#define ped 20

void TCA0\_init(void)

{

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;

sei();

}

int main(void)

{

//sei();

PORTF.PIN5CTRL |= PORT\_PULLUPEN\_bm | PORT\_ISC\_BOTHEDGES\_gc;

PORTF.PIN6CTRL |= PORT\_PULLUPEN\_bm | PORT\_ISC\_BOTHEDGES\_gc;

sei();

while(1)

{

if (sw5==10 && sw6==12)

{

TCA0\_init();

sw5=0;

sw6=0;

}

}

}

ISR(PORTF\_PORT\_vect){

int y = PORTF.INTFLAGS;

flag++;

if (y==32)

{

if (flag==1)

{

sw5=sw5+5;

}

else if (flag==3 && sw5==5)

{

sw5=sw5+5;

}

else

{

flag=0;

sw5=0;

sw6=0;

}

}

else if(y==64)

{

if (flag==2 && sw5==5)

{

sw6=sw6+6;

}

else if (flag==4 && sw6==6)

{

sw6=sw6+6;

}

else

{

flag=0;

sw5=0;

sw6=0;

}

}

PORTF.INTFLAGS=y;

}

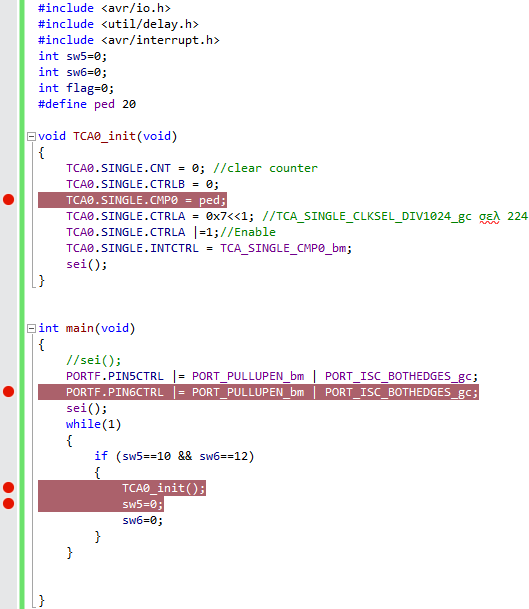
ISR(TCA0\_CMP0\_vect){

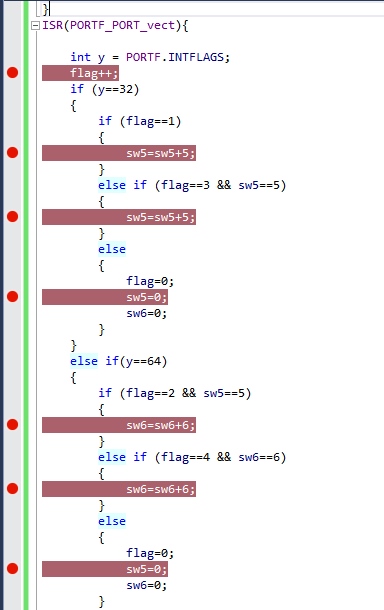
TCA0.SINGLE.CTRLA = 0;

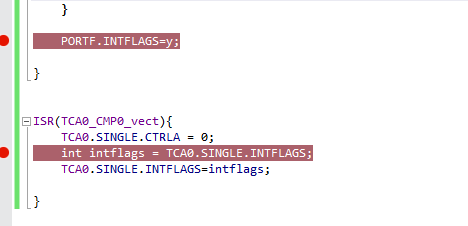
int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS=intflags;

}







Σχόλια – Επεξήγηση κώδικα:

**main**

Αρχικά στην main θέτουμε τα δύο switch και μέσα στην while(1) μια συνθήκη ενεργοποίησης του timer ώστε να ενεργοποιείται μόνο όταν η μεταβλητές sw5 = 10 και sw6 = 12.

**Πάτημα switch5 (if y == 32)**

Στην ρουτίνα διακοπής των switch χρησιμοποιούμε τις τιμές των y flags ώστε να διαφοροποιείται πιο switch πατήθηκε (switch5 y=2^5=32, switch6 y=2^6=64).Επίσης θέτουμε και μια μεταβλητή flag η οποία μετράει σε πιο πάτημα κουμπιού βρίσκεται το σύστημα(η μεταβλητή flag μετράει – αυξάνεται κατά 1 όταν πατηθεί είτε το switch5 είτε το switch6). Την πρώτη φορά που θα πατηθεί το switch5 η μεταβλητή sw5 θα γίνει 5 ενώ την 2η (flag=3) φορά που θα πατηθεί (και ενώ εχει ήδη πατηθεί μια φορά το switch6 )η μεταβλητή sw5 θα γίνει 10. Σε οποιαδήποτε άλλη περίπτωση όλες οι μεταβλητές θα μηδενιστούν αφού ο κωδικός εισαγωγής θα είναι ήδη λάθος.

**Πάτημα switch5 (else if y == 64)**

Όταν πατηθεί για πρώτη φορά το switch6 και εάν προηγούμενος έχει πατηθεί μια φορά το switch5 (flag==2 && sw5==5) η μεταβλητή sw6 θα αυξήσει την τιμή της κατά 6, εάν πατηθεί χωρίς να έχει πατηθεί προηγούμενος το switch5 τότε όλες οι μεταβλητές θα μηδενιστούν αφού ο κωδικός εισαγωγής θα είναι ήδη λάθος. Την δεύτερη φορα που θα πατηθεί το switch6 και εάν προηγουμένως έχει πατηθεί η σωστή ακολουθία (δηλαδή flag==4 και sw6=6) η μεταβλητή sw6 θα αυξήσει την τιμή της κατά 6. Σε οποιαδήποτε άλλη περίπτωση όλες οι μεταβλητές θα μηδενιστούν αφού ο κωδικός εισαγωγής θα είναι ήδη λάθος.

**Ερώτημα 2**

Κώδικας ερωτήματος 2:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

int sw5=0;

int sw6=0;

int flag=0;

int failflag=0;

int mode = 0;

#define ped 20

void TCA0\_init(void)

{

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;

sei();

}

void ADC0\_init(void)

{

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

ADC0.DBGCTRL |= ADC\_DBGRUN\_bm;

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;

}

int main(void)

{

//sei();

PORTD.DIR |= PIN0\_bm ;

PORTF.PIN5CTRL |= PORT\_PULLUPEN\_bm | PORT\_ISC\_BOTHEDGES\_gc;

PORTF.PIN6CTRL |= PORT\_PULLUPEN\_bm | PORT\_ISC\_BOTHEDGES\_gc;

sei();

while(1)

{

if (mode==0 && sw5==10 && sw6==12)

{

TCA0\_init();

sw5=0;

sw6=0;

flag=0;

}

else if(mode==1 && sw5==10 && sw6==12){

TCA0.SPLIT.CTRLA &= ~TCA\_SPLIT\_ENABLE\_bm;//disable the timer

PORTD.OUT |= PIN0\_bm;

sw5=0;

sw6=0;

flag=0;

mode==0;

}

else if (failflag=3) {

PORTD.OUTCLR= PIN0\_bm; //LED is on

break();

}

}

}

ISR(PORTF\_PORT\_vect){

//clear the interrupt flag

int y = PORTF.INTFLAGS;

flag++;

if (mode==0) {

if (y==32)

{

if (flag==1)

{

sw5=sw5+5;

}

else if (flag==3 && sw5==5)

{

sw5=sw5+5;

}

else

{

flag=0;

sw5=0;

sw6=0;

}

}

else if(y==64)

{

if (flag==2 && sw5==5)

{

sw6=sw6+6;

}

else if (flag==4 && sw6==6)

{

sw6=sw6+6;

}

else

{

flag=0;

sw5=0;

sw6=0;

}

}

}

else if (mode==1) {

if (y==32)

{

if (flag==1)

{

sw5=sw5+5;

}

else if (flag==3 && sw5==5)

{

sw5=sw5+5;

}

else

{

flag=0;

sw5=0;

sw6=0;

failflag=++;

}

}

else if(y==64)

{

if (flag==2 && sw5==5)

{

sw6=sw6+6;

}

else if (flag==4 && sw6==6)

{

sw6=sw6+6;

}

else

{

flag=0;

sw5=0;

sw6=0;

failflag=++;

}

}

}

PORTF.INTFLAGS=y;

}

ISR(TCA0\_CMP0\_vect){

TCA0.SINGLE.CTRLA = 0; //Disable

//clear flag

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS=intflags;

if (mode==0)

{

ADC0\_init();

}

else

{

PORTD.OUTCLR= PIN0\_bm; //LED is on

break();

}

}

ISR(ADC0\_WCOMP\_vect){

int intflags = ADC0.INTFLAGS;

ADC0.INTFLAGS = intflags;

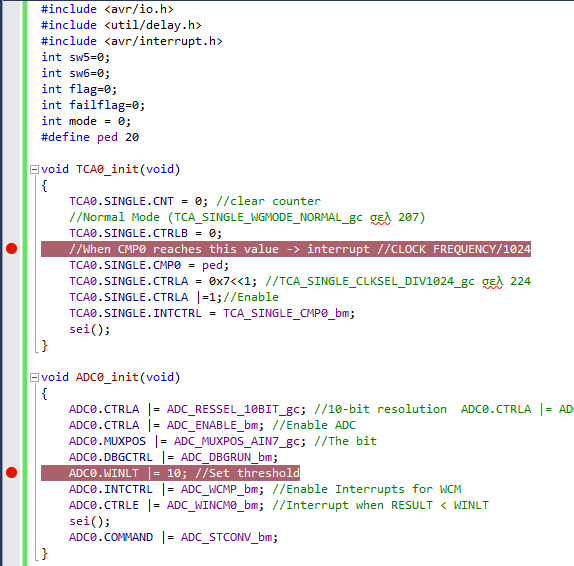
PORTD.OUTCLR= PIN0\_bm; //LED is on

mode=1;

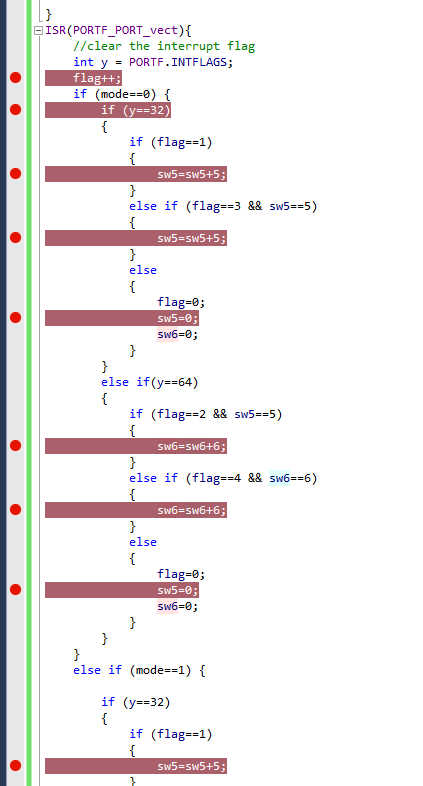
TCA0\_init();

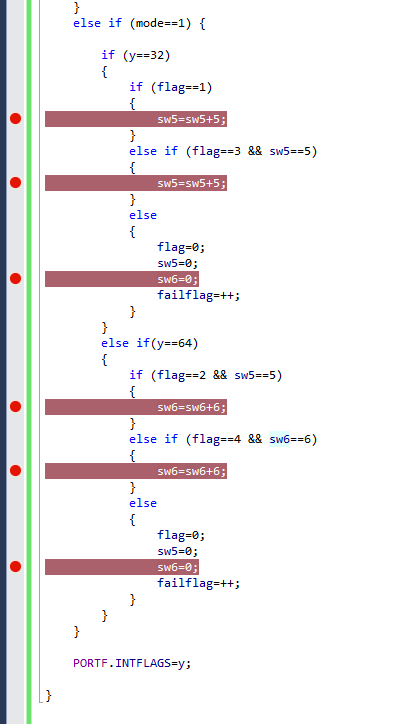
}

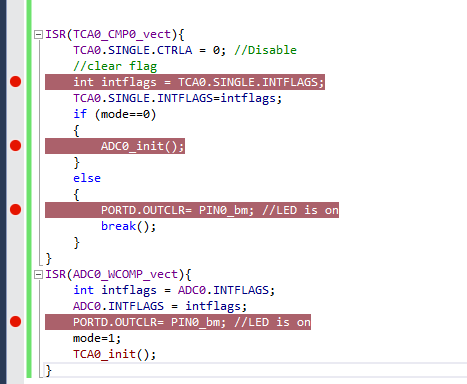
Κώδικας με breakpoints:











Σχόλια – Επεξήγηση κώδικα:

**Προσθήκη Μεταβλητών**

Για το δεύτερο ερώτημα προσθέτουμε στον κώδικα τις μεταβλητές failflag , ώστε να μετράμε τις

αποτυχημένες προσπάθειες εισαγωγής του κωδικού όταν το σύστημα βρίσκεται στην δεύτερη φάση ,

και mode ώστε να διαχωρίσουμε αν το σύστημα βρίσκεται στην αρχική φάση (mode=0) ή στην δεύτερη

φάση (mode=1).

**Επεξήγηση των Mode**

Το πρόγραμμα ξεκινάει με mode=0 και αναμένει την σωστή εισαγωγή κωδικού όπως στο προηγούμενο

ερώτημα αυτή την φορά όμως υπάρχει ακόμα μία if στην ISR του κουμπιού η οποία ελέγχει σε τι mode

βρίσκεται. Έτσι με την σωστή εισαγωγή κωδικού οδηγούμαστε στην while όπου για mode=0 και όλες τις

άλλες παραμέτρους ενεργοποιείται ο timer, όταν ο timer μπει στην ISR για mode=0 ενεργοποιεί τον

ADC ο οποίος έχει τεθεί έτσι ώστε μόλις το threshold πάρει τιμή μικρότερη από 10 να ενεργοποιηθεί η

ISR του η οποία θέτει το σύστημα σε mode=1 και ξανά ενεργοποιεί τον timer. Για mode=1 η ISR των

switches για κάθε λανθασμένη προσπάθεια ανεβάζουν την τιμή του failflag κατά 1 μόλις το failflag

φτάσει την τιμή 3 οδηγούμαστε στην while και στην εκδοχή όπου το failflag=3 όπου εκεί παραμένει

αναμμένο το λαμπάκι και τερματίζει το πρόγραμμα (υπό συνθήκη επειδή δεν έχουμε βάλει ακόμα το

PWM). Αν όμως εισαχθεί σωστά ο κωδικός πριν το failflag=3 τότε οδηγούμαστε στην while και στην

εκδοχή για mode=1 όπου το mode γίνεται πάλι 0 και αναμένει το σύστημα την σωστή εισαγωγή

κωδικού. Αν όμως δεν προλάβει ο κωδικός να εισαχθεί πριν την λήξει του timer ο timer θα μπει μέσα

στην ISR και για mode=1 και παραμένει αναμμένο το λαμπάκι και τερματίζει το πρόγραμμα (υπό

συνθήκη επειδή δεν έχουμε βάλει ακόμα το PWM).

**Ερώτημα 3**

Κώδικας ερωτήματος 3:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

int sw5=0;

int sw6=0;

int flag=0;

int failflag=0;

int mode = 0;

#define ped 20

void TCA0\_init(void)

{

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;

sei();

}

void ADC0\_init(void)

{

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

ADC0.DBGCTRL |= ADC\_DBGRUN\_bm;

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;

}

void TCB0\_init(void){

TCB0.CTRLB = TCB\_CLKSEL\_CLKDIV1024\_gc; // Set the clock source and prescaler

// Set PWM period

TCB0.CCMP = 254;

// Set duty cycle

TCB0.CCMPH = 127;

// Set PWM output on PD1

TCB0.CTRLB |= TCB\_CCMPEN\_bm | TCB\_CNTMODE\_PWM8\_gc;

// Enable interrupt overflow

TCB0.INTCTRL |= TCB\_CAPT\_bm;

// Enable TCB0

TCB0.CTRLA |= TCB\_ENABLE\_bm;

// Enable global interrupts

sei();

}

int main(void)

{

//sei();

PORTD.DIR |= PIN0\_bm ;

PORTF.PIN5CTRL |= PORT\_PULLUPEN\_bm | PORT\_ISC\_BOTHEDGES\_gc;

PORTF.PIN6CTRL |= PORT\_PULLUPEN\_bm | PORT\_ISC\_BOTHEDGES\_gc;

sei();

while(1)

{

if (mode==0 && sw5==10 && sw6==12)

{

TCA0\_init();

sw5=0;

sw6=0;

flag=0;

}

else if(mode==1 && sw5==10 && sw6==12){

TCA0.SPLIT.CTRLA &= ~TCA\_SPLIT\_ENABLE\_bm;//disable the timer

PORTD.OUT |= PIN0\_bm;

sw5=0;

sw6=0;

flag=0;

failflag=0;

mode==0;

}

else if (failflag==3 && mode==1) {

PORTD.OUTCLR= PIN0\_bm; //LED is on

sw5=0;

sw6=0;

flag=0;

failflag=0;

mode=2;

TCB0\_init();

}

else if(mode==2 && sw5==10 && sw6==12) {

sw5=0;

sw6=0;

flag=0;

failflag=0;

mode=0;

TCB0.CTRLA &= ~TCB\_ENABLE\_bm; // Disable the timer

PORTD.OUT |= PIN0\_bm;//LED is Off

}

}

}

ISR(PORTF\_PORT\_vect){

//clear the interrupt flag

int y = PORTF.INTFLAGS;

flag++;

if (mode==0) {

if (y==32)

{

if (flag==1)

{

sw5=sw5+5;

}

else if (flag==3 && sw5==5)

{

sw5=sw5+5;

}

else

{

flag=0;

sw5=0;

sw6=0;

}

}

else if(y==64)

{

if (flag==2 && sw5==5)

{

sw6=sw6+6;

}

else if (flag==4 && sw6==6)

{

sw6=sw6+6;

}

else

{

flag=0;

sw5=0;

sw6=0;

}

}

}

else if (mode==1) {

if (y==32)

{

if (flag==1)

{

sw5=sw5+5;

}

else if (flag==3 && sw5==5)

{

sw5=sw5+5;

}

else

{

flag=0;

sw5=0;

sw6=0;

failflag=++;

}

}

else if(y==64)

{

if (flag==2 && sw5==5)

{

sw6=sw6+6;

}

else if (flag==4 && sw6==6)

{

sw6=sw6+6;

}

else

{

flag=0;

sw5=0;

sw6=0;

failflag=++;

}

}

}

else if (mode==2) {

if (y==32)

{

if (flag==1)

{

sw5=sw5+5;

}

else if (flag==3 && sw5==5)

{

sw5=sw5+5;

}

else

{

flag=0;

sw5=0;

sw6=0;

}

}

else if(y==64)

{

if (flag==2 && sw5==5)

{

sw6=sw6+6;

}

else if (flag==4 && sw6==6)

{

sw6=sw6+6;

}

else

{

flag=0;

sw5=0;

sw6=0;

}

}

}

PORTF.INTFLAGS=y;

}

ISR(TCA0\_CMP0\_vect){

TCA0.SINGLE.CTRLA = 0; //Disable

//clear flag

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS=intflags;

if (mode==0)

{

ADC0\_init();

}

else if(mode==1)

{

PORTD.OUTCLR= PIN0\_bm; //LED is on

sw5=0;

sw6=0;

flag=0;

failflag=0;

mode=2;

TCB0\_init();

}

}

ISR(ADC0\_WCOMP\_vect){

int intflags = ADC0.INTFLAGS;

ADC0.INTFLAGS = intflags;

PORTD.OUTCLR= PIN0\_bm; //LED is on

mode=1;

TCA0\_init();

}

ISR(TCB0\_INT\_vect) {

// Clear the interrupt flag

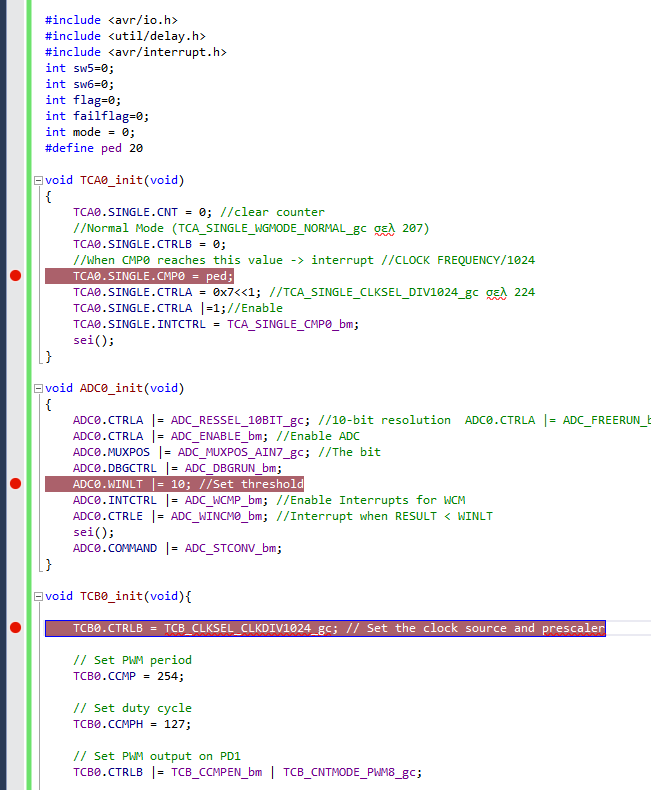
TCB0.INTFLAGS |= TCB\_CAPT\_bm;

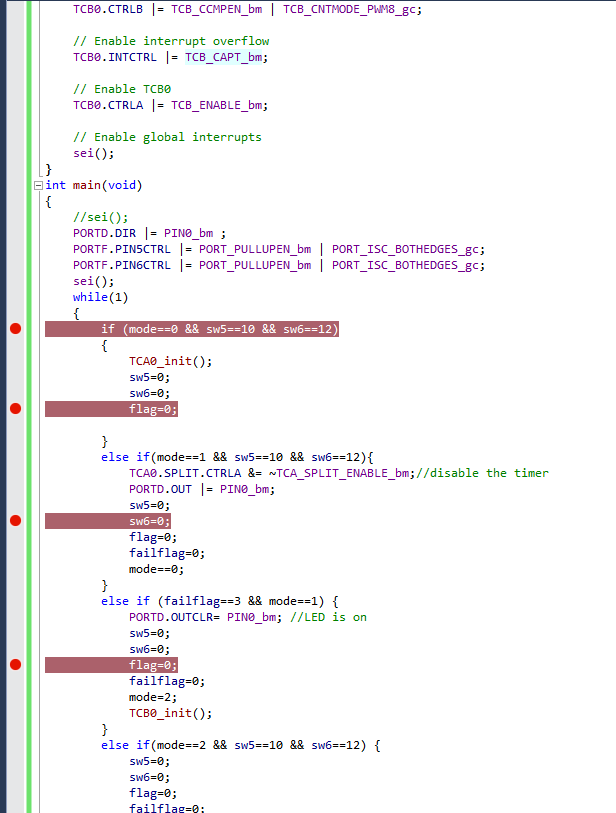
// Toggle the pin state

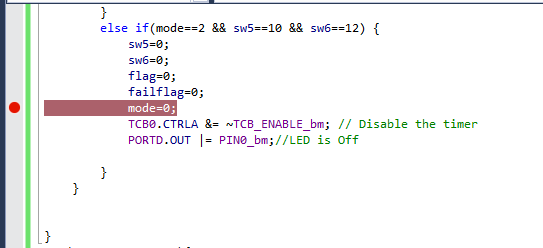
PORTD.OUTTGL |= PIN0\_bm;

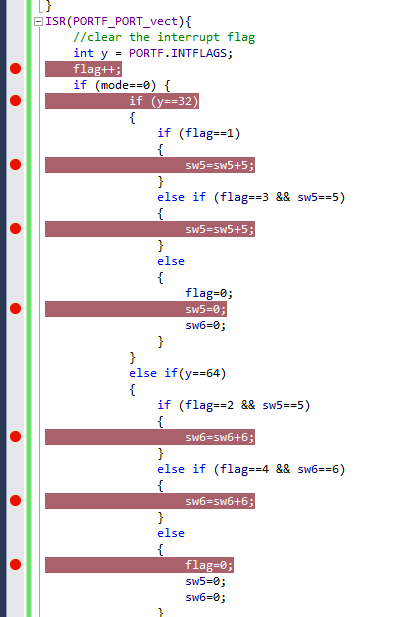
}

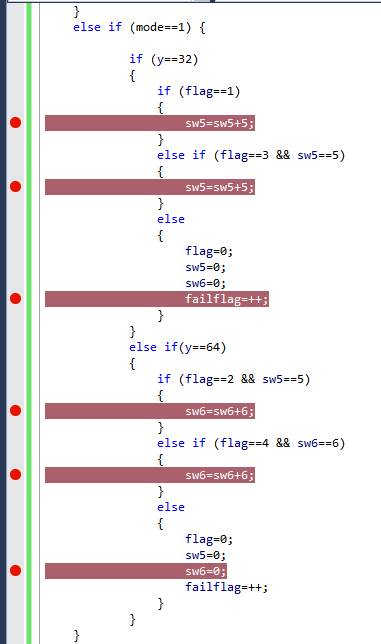
Κώδικας με breakpoints:

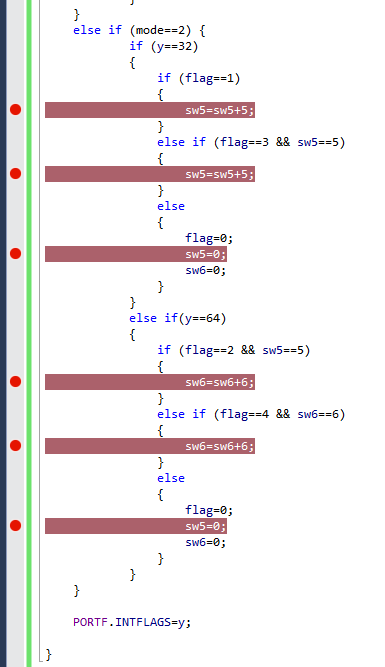


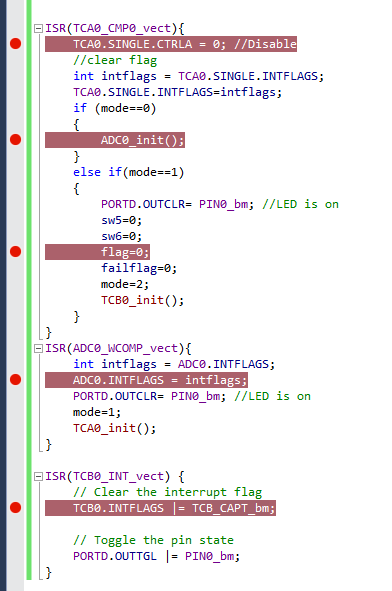












Σχόλια – Επεξήγηση κώδικα:

Στο τρίτο ερώτημα προσθέσαμε τον TCB ώστε να δώσει τον PWM παλμό όπου χρειάζεται το λαμπάκι για να εκτελέσει TOGGLE , επίσης προσθέσαμε ακόμα ένα mode με αριθμό 2. Πιο συγκεκριμένα ορίσαμε ότι στο mode=1 όταν πατηθεί λάθος κωδικός τρεις φορές ή όταν λήξει ο timer για mode=1 να ενεργοποιείται ο TCB και να γίνεται mode=2 . Στην ISR του TCB αναβοσβήνει το λαμπάκι με τις συχνότητες όπου θέσαμε στην ανερχόμενη παρυφή και ο μόνος τρόπος να σταματήσει είναι να εισαχθεί ο σωστός κωδικός . Όπως φαίνεται και στην ISR του κουμπιού για mode=2 αρκεί να ενταχθεί σωστά ο κωδικός μια φορά ώστε να μπει στην εντολή[ else if(mode==2 && sw5==10 && sw6==12) ] της while όπου εκεί πέρα μηδενίζονται όλα τα flags το mode γίνεται ξανά 0 και απενεργοποιείται και ο TCB και το λαμπάκι. Έτσι το πρόγραμμα αρχίζει ξανά από την αρχή όπου για mode=0 περιμένει την σωστή εισαγωγή κωδικού ώστε να ξανά αρχίσει η λειτουργία του συναγερμού.