**ΕΡΓΑΣΤΗΡΙΟ ΠΡΟΗΓΜΕΝΟΙ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ**

**Εργαστηριακή Άσκηση 4**

Δούρου Βασιλική Ευαγγελία – Α.Μ: 1072633

Πουρής Βασίλειος – Α.Μ.: 1072485

**1.** Κατά τη συγγραφή των παρακάτω έγινε η ακόλουθη παραδοχή:

* Σε περίπτωση που πατηθεί ταυτόχρονα και το PIN5 και το PIN6 του PORTF, τότε θα μετρηθεί ως λάθος σειρά ψηφίων.

Ο κώδικας του πρώτου ερωτήματος της άσκησης για την αναμονή εισαγωγής σωστού συνδυασμού και την ενεργοποίηση του χρονιστή είναι ο ακόλουθος:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#define en\_alarm 20

int y=0; // flag for portf interrupt

int x=0; //flag for timer interrupt

int pr=0;

int main() {

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

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

sei();

while(y==0){

}

if (y==1){

TCA0.SINGLE.CTRLB = 0;

TCA0.SINGLE.CNT = 0;

TCA0.SINGLE.CMP0 = en\_alarm;

TCA0.SINGLE.CTRLA = TCA\_SPLIT\_CLKSEL\_DIV1024\_gc | TCA\_SPLIT\_ENABLE\_bm;

TCA0.SINGLE.INTCTRL = TCA\_SINGLE\_CMP0\_bm; // CMP0 INTERRUPT ENABLE

while (x==0) {

}

y=0;

x=0;

}

cli();

}

ISR(TCA0\_CMP0\_vect){

//clear flag

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS=intflags;

x=1;

}

ISR(PORTF\_PORT\_vect){

//clear the interrupt flag

if(PORTF.INTFLAGS==0x20){

if(pr==0 || pr==2){

pr++;

}

else{

pr=0; //an patithei 5,5

}

}

if(PORTF.INTFLAGS==0x40){

if(pr==1 || pr==3){

pr++;

}

else{

pr=0;

}

}

if(pr==4){

pr=0;

y=1;

}

int intflags = PORTF.INTFLAGS;

PORTF.INTFLAGS=intflags;

}

Ο κώδικας του δεύτερου ερωτήματος μετά την προσθήκη του ADC, του LED0, του χρονιστή και τον έλεγχο εισαγωγής σωστού συνδυασμού σε τρεις προσπάθειες μέσα στο χρονικό περιθώριο που ορίζει ο χρονιστής είναι ο παρακάτω:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#define en\_alarm 20

#define time\_for\_dis 80

int y=0; // flag for portf interrupt

int x=0; //flag for timer interrupt

int pr=0;

int m=0; //mistakes counter

int z=0; // flag for adc interrupt

int main() {

PORTD.DIR |= PIN0\_bm; //PIN0 is output for ADC

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

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

sei();

while(y==0){

}

x=0;

TCA0.SINGLE.CTRLB = 0;

TCA0.SINGLE.CNT = 0;

TCA0.SINGLE.CMP0 = en\_alarm;

TCA0.SINGLE.CTRLA = TCA\_SPLIT\_CLKSEL\_DIV1024\_gc | TCA\_SPLIT\_ENABLE\_bm;

TCA0.SINGLE.INTCTRL = TCA\_SINGLE\_CMP0\_bm; // CMP0 INTERRUPT ENABLE

while (x==0) {

}

x=0;

//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 |= 4; //Set threshold

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

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

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

while (z==0) {

}

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

TCA0.SINGLE.CTRLB = 0;

TCA0.SINGLE.CNT = 0;

TCA0.SINGLE.CMP0 = time\_for\_dis;

TCA0.SINGLE.CTRLA = TCA\_SPLIT\_CLKSEL\_DIV1024\_gc | TCA\_SPLIT\_ENABLE\_bm;

TCA0.SINGLE.INTCTRL = TCA\_SINGLE\_CMP0\_bm; // CMP0 INTERRUPT ENABLE

while(x==0){

}

y=0;

x=0;

cli();

}

ISR(TCA0\_CMP0\_vect){

//clear flag

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS=intflags;

x=1;

}

ISR(PORTF\_PORT\_vect){

//clear the interrupt flag

if(PORTF.INTFLAGS==0x20){

if(pr==0 || pr==2){

pr++;

}

else{

pr=0; //an patithei 5,5

m++;

}

}

if(PORTF.INTFLAGS==0x40){

if(pr==1 || pr==3){

pr++;

}

else{

pr=0;

m++;

}

}

if(m==3){

TCA0.SINGLE.CTRLA =0 ;

pr=0;

x=1;

m=0;

}

if(pr==4){

pr=0;

y=1;

m=0;

}

int intflags = PORTF.INTFLAGS;

PORTF.INTFLAGS=intflags;

}

ISR(ADC0\_WCOMP\_vect){

ADC0.CTRLA =0 ;

int intflags = ADC0.INTFLAGS;

ADC0.INTFLAGS = intflags;

z=1;

}

Ο κώδικας του τρίτου ερωτήματος μετά την ενεργοποίηση της σειρήνας είναι ο ακόλουθος:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#define en\_alarm 20

#define time\_for\_dis 80

int y=0; // flag for portf interrupt

int x=0; // flag for timer interrupt

int pr=0; // elegxei thn swsth seira twn pshfiwn

int m=0; //mistakes counter

int z=0; // flag for adc interrupt

int alarm=0; // gia to pwm

int main() {

PORTD.DIR |= PIN0\_bm; //PIN0 is output for ADC

PORTD.OUT |= PIN0\_bm; //LED IS OFF

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

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

while (1){

sei();

while(y==0){

}

x=0;

TCA0.SINGLE.CTRLB = 0;

TCA0.SINGLE.CNT = 0;

TCA0.SINGLE.CMP0 = en\_alarm;

TCA0.SINGLE.CTRLA = TCA\_SPLIT\_CLKSEL\_DIV1024\_gc | TCA\_SPLIT\_ENABLE\_bm;

TCA0.SINGLE.INTCTRL = TCA\_SINGLE\_CMP0\_bm; // CMP0 INTERRUPT ENABLE

while (x==0) {

}

x=0;

//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 |= 4; //Set threshold

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

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

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

while (z==0) {

}

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

TCA0.SINGLE.CTRLB = 0;

TCA0.SINGLE.CNT = 0;

TCA0.SINGLE.CMP0 = time\_for\_dis;

TCA0.SINGLE.CTRLA = TCA\_SPLIT\_CLKSEL\_DIV1024\_gc | TCA\_SPLIT\_ENABLE\_bm;

TCA0.SINGLE.INTCTRL = TCA\_SINGLE\_CMP0\_bm; // CMP0 INTERRUPT ENABLE

while(x==0){

}

TCA0.SINGLE.CTRLA=0;

y=0;

x=0;

if(alarm==1){

TCA0.SINGLE.CTRLA=TCA\_SINGLE\_CLKSEL\_DIV1024\_gc;

TCA0.SINGLE.PER = 254; //select the resolution

TCA0.SINGLE.CMP0 = 127; //select the duty cycle

//select Single\_Slope\_PWM

TCA0.SINGLE.CTRLB = TCA\_SINGLE\_WGMODE\_SINGLESLOPE\_gc;

//enable interrupt Overflow

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

//enable interrupt CMP0

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

TCA0.SINGLE.CTRLA |= TCA\_SINGLE\_ENABLE\_bm; //Enable

while(y==0){

}

PORTD.OUT |= PIN0\_bm; //PIN is off

alarm=0;

}

y=0;

x=0;

cli();

}

}

ISR(TCA0\_CMP0\_vect){

//clear flag

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS=intflags;

x=1;

alarm=1;

PORTD.OUT |= PIN0\_bm; //PIN is off

}

ISR(TCA0\_OVF\_vect){

//clear the interrupt flag

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS = intflags;

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

}

ISR(PORTF\_PORT\_vect){

//clear the interrupt flag

if(PORTF.INTFLAGS==0x20){

if(pr==0 || pr==2){

pr++;

}

else{

pr=0; //an patithei 5,5

m++;

}

}

if(PORTF.INTFLAGS==0x40){

if(pr==1 || pr==3){

pr++;

}

else{

pr=0;

m++;

}

}

if(PORTF.INTFLAGS==0x60){

m++;

}

if(m==3){

pr=0;

x=1;

m=0;

alarm=1;

}

if(pr==4){

TCA0.SINGLE.CTRLA=0;

x=1;

pr=0;

y=1;

m=0;

}

int intflags = PORTF.INTFLAGS;

PORTF.INTFLAGS=intflags;

}

ISR(ADC0\_WCOMP\_vect){

ADC0.CTRLA =0 ;

int intflags = ADC0.INTFLAGS;

ADC0.INTFLAGS = intflags;

z=1;

}

**2.** Το διάγραμμα ροής του πρώτου ερωτήματος είναι το ακόλουθο:

Εικόνα που περιέχει διάγραμμα, κείμενο, τεχνικό σχέδιο, Σχέδιο

Περιγραφή που δημιουργήθηκε αυτόματα

Το διάγραμμα ροής του δεύτερου ερωτήματος είναι το εξής:

Εικόνα που περιέχει διάγραμμα, τεχνικό σχέδιο, σκίτσο/σχέδιο, Σχέδιο

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει διάγραμμα, ασπρόμαυρο, τεχνικό σχέδιο, κείμενο

Περιγραφή που δημιουργήθηκε αυτόματαΤο διάγραμμα ροής του τρίτου ερωτήματος είναι το ακόλουθο:

Στον κώδικα του πρώτου ερωτήματος, αρχικά ορίζονται οι τρεις μεταβλητές x, y και pr και αρχικοποιούνται με μηδέν. Οι δύο πρώτες χρησιμοποιούνται ως flags για τα interrupts του timer και του PORTF αντίστοιχα, ενώ η τρίτη χρησιμοποιείται για τον υπολογισμό της σωστής σειράς εισαγωγής των ψηφίων. Πιο συγκεκριμένα, κάθε φορά που πατιέται ένα ψηφίο του κωδικού με τη σωστή σειρά, αυξάνεται κατά 1. Αν ένα ψηφίο πατηθεί σε λάθος σειρά, θεωρείται όλος ο κωδικός λανθασμένος και αυτή μηδενίζεται. Έπειτα, γίνεται enable το pullup και το interrupt για το PIN5 και το PIN6 του PORTF. Στη συνέχεια, όσο δεν έχει πατηθεί κάποιο κουμπί, η ροή του προγράμματος παραμένει σε ένα while loop. Όταν πατηθεί ένα κουμπί, θα γίνει διακοπή και θα καλεστεί η ISR του PORTF. Εκεί, ελέγχεται ποιο κουμπί προκάλεσε τη διακοπή. Αν πατήθηκε μόνο το 5, τότε ελέγχεται αν πατήθηκε ως το 1ο ή το 3ο ψηφίο, μέσω της μεταβλητής pr. Αν ισχύει αυτό, τότε αυξάνεται η pr κατά 1, αλλιώς γίνεται ίση με 0. Αν πατήθηκε μόνο το 6, τότε ελέγχεται αν πατήθηκε ως το 2ο ή το 4ο ψηφίο, μέσω της μεταβλητής pr. Αν ισχύει αυτό, τότε αυξάνεται η pr κατά 1, αλλιώς γίνεται ίση με 0. Σε κάθε άλλη περίπτωση, η μεταβλητή pr γίνεται ίση με το μηδέν. Αν γίνει ίση με 4, δηλαδή αν πατηθούν όλα τα ψηφία στη σωστή σειρά, τότε το y γίνεται ίσο με 1 και η ροή του προγράμματος βγαίνει από το while loop. Τότε, γίνεται enable ο TCA σε single mode και παραμένει η ροή του προγράμματος σε ένα άλλο while loop. Μόλις γίνει interrupt από τον timer, τότε βγαίνει από το while loop και τελειώνει η ροή του προγράμματος.

Στο δεύτερο ερώτημα, ο κώδικας είναι όπως στο πρώτο με την εξαίρεση ότι αρχικά ορίζονται και δύο επιπλέον μεταβλητές, η m και z, οι οποίες αρχικοποιούνται με μηδέν. H m μετράει τον αριθμό λανθασμένων προσπαθειών, ενώ η z λειτουργεί ως flag για τον ADC. Επιπλέον, αφού γίνουν όσα έγιναν στο πρώτο ερώτημα και γίνει interrupt από τον TCA, τότε αρχικοποιείται ο ADC σε free-running mode και ενεργοποιούνται οι διακοπές όταν ισχύει RES<WINLT. Αρχίζει η μετατροπή στον ADC και μέχρι να γίνει interrupt μένουμε σε ένα while loop. Μόλις γίνει interrupt από τον ADC, τότε καλείται η ISR του, στη διάρκεια της οποίας απενεργοποιείται ο ADC. Στη συνέχεια, ενεργοποιείται το PIN0 του PORTD και ενεργοποιείται ο TCA σε single mode. Όσο δεν γίνεται interrupt από τον timer, το πρόγραμμα μένει σε ένα while loop. Στη διάρκεια του, αν πατηθούν με λάθος σειρά τα πλήκτρα του PORTF με τον έλεγχο που γίνεται όπως στο 1ο ερώτημα, τότε αυξάνεται κατά 1 η μεταβλητή m. Αν η μεταβλητή m γίνει ίση με 3 ή αν μπει σωστός κωδικός, τότε βγαίνει η ροή του προγράμματος από το while loop και τερματίζει ο κώδικας του 2ου ερωτήματος. Εναλλακτικά, αν καλεστεί η ISR του timer, τότε θα βγει η ροή από το loop.

Στο κώδικα του τρίτου ερωτήματος, ο κώδικας είναι όπως στο δεύτερο με την εξαίρεση ότι αρχικά ορίζεται και μία ακόμη μεταβλητή, η alarm, η οποία αρχικοποιείται με μηδέν και έχει σχέση με το αν χρειάζεται να ενεργοποιηθεί η σειρήνα. Γίνεται ίση με 1 μέσα από την ISR του timer ή του PORTF σε περίπτωση που έχουν υπάρξει 3 λανθασμένες προσπάθειες. Επιπλέον, αφού γίνουν όσα έγιναν στο δεύτερο ερώτημα, αν έχει μπει σωστός συνδυασμός, τότε η μεταβλητή alarm παραμένει ίση με το 0 και η ροή του προγράμματος επιστρέφει στην αρχή, ενώ αν έχει γίνει interrupt από τον timer ή αν η μεταβλητή m γίνει ίση με 3, τότε αρχικοποιείται η περίοδος και το CMP0 του TCA με το 50% της τιμής της περιόδου (duty cycle 50%). Τέλος, ορίζεται η τιμή του prescaler, γίνεται enable o timer και το interrupt για το OVF και το CMP0 και η ροή του προγράμματος μένει σε ένα while loop. Αν γίνει OVF interrupt, δηλαδή όταν ανεβαίνουμε στη υψηλή στάθμη, η τιμή του PIN0 του PORTD θα γίνει ίση με 0, ενώ αν γίνει interrupt στο CMP0, δηλαδή όταν πηγαίνουμε στη χαμηλή στάθμη, η τιμή του PIN0 του PORTD θα γίνει ίση με 1. Σε περίπτωση που εισαχθεί σωστός κωδικός, μέσω του ελέγχου που γίνεται στο PORTF όπως στα προηγούμενα ερωτήματα, τότε γίνεται disable ο TCA, η τιμή του PIN0 του PORTD θα γίνει ίση με 1, η τιμή του alarm γίνεται ίση με 0 και η ροή του προγράμματος βγαίνει από το while loop και επανέρχεται στην αρχή.

Οι τιμές που επιλέχθηκαν για τον χρόνο αναμονής ενεργοποίησης του συναγερμού και εισαγωγής σωστού συνδυασμού είναι οι ακόλουθοι:

⇒και ⇒αντίστοιχα. Οι τιμές αυτές προέκυψαν από τον τύπο , όπου Τ είναι ο χρόνος του χρονιστή και είναι η συχνότητα του ρολογιού, όπως προκύπτει από τον τύπο , με =20MHz και Ν=1024.