**ΕΡΓΑΣΤΗΡΙΟ ΠΡΟΗΓΜΕΝΟΙ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ**

**Εργαστηριακή Άσκηση 1**

Δούρου Βασιλική Ευαγγελία – Α.Μ: 1072633

Πουρής Βασίλειος – Α.Μ.: 1072485

**1.** Ο κώδικας του πρώτου ερωτήματος της άσκησης για τη διαχείριση των φαναριών του μεγάλου δρόμου και των πεζών είναι ο ακόλουθος:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#define pushagain 20

#define ped 10

int x=0;

int button=0;

int main() {

PORTD.DIR |= PIN0\_bm; //PIN0 IS OUTPUT

PORTD.OUT |= PIN0\_bm; //LED IS OFF (KOKKINO FANARI PEZWN=PIN0)

PORTD.DIR |= PIN2\_bm; //PIN2 IS OUTPUT

PORTD.OUT |= PIN2\_bm; //LED IS OFF (KOKKINO FANARI DROMOU=PIN2)

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

while (1){

sei(); //begin accepting interrupt signals

if (button==1) //koumpi pezwn patithike

{

PORTD.OUT |= PIN2\_bm; //kokkino gia ta autokinhta

PORTD.OUTCLR = PIN0\_bm; //prasino gia pezous

TCA0.SINGLE.CNT=0;

TCA0.SINGLE.CTRLB=0;

TCA0.SINGLE.CMP0=ped;

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

TCA0.SINGLE.CTRLA |=1;

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

while (x==0) {

}

PORTD.OUT |=PIN0\_bm; //kokkino to fanari twn pezwn

PORTD.OUTCLR = PIN2\_bm; //prasino gia autokinhta

x=0;

TCA0.SINGLE.CNT = 0;

TCA0.SINGLE.CMP0 = pushagain;

while (x==0) {

}

x=0;

}

else{

PORTD.OUTCLR = PIN2\_bm; //prasino dromou

}

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

int y = PORTF.INTFLAGS;

PORTF.INTFLAGS=y;

button++;

}

Ο κώδικας του δεύτερου ερωτήματος της άσκησης, μετά την προσθήκη της λειτουργίας του τραμ, είναι ο παρακάτω:

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#define pushagain 20

#define ped 10

#define tram 50

int y=0;

int x=0;

int button=0;

int main() {

PORTD.DIR |= PIN0\_bm; //PIN0 IS OUTPUT

PORTD.OUT |= PIN0\_bm; //LED IS OFF (KOKKINO FANARI PEZWN=PIN0)

PORTD.DIR |= PIN2\_bm; //PIN2 IS OUTPUT

PORTD.OUT |= PIN2\_bm; //LED IS OFF (KOKKINO FANARI DROMOU=PIN2)

PORTD.DIR |= PIN1\_bm; //PIN IS OUTPUT

PORTD.OUT |= PIN1\_bm; //LED IS OFF (KOKKINO FANARI TRAM=PIN1)

while (1){

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

TCA0.SPLIT.CTRLD = TCA\_SPLIT\_SPLITM\_bm; //ENABLE TCA SPLIT MODE

TCA0.SPLIT.CTRLB = TCA\_SPLIT\_LCMP0EN\_bm|TCA\_SPLIT\_HCMP0EN\_bm;

TCA0.SPLIT.LCNT = tram;

TCA0.SPLIT.LCMP0 = 0x00;

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

TCA0.SPLIT.INTCTRL =0x10; // LCMP0 INTERRUPT ENABLE

sei();

while(y==0){

sei();

if (button==1) //koumpi pezwn patithike

{

PORTD.OUT |= PIN2\_bm; //kokkino gia ta autokinhta

PORTD.OUTCLR = PIN0\_bm; //prasino gia pezous

TCA0.SPLIT.HCNT = ped;

TCA0.SPLIT.HCMP0 = 0X00;

TCA0.SPLIT.INTCTRL =0x02; // HUNF INTERRUPT ENABLE

sei();

while (x==0) {

}

cli();

x=0;

PORTD.OUT |=PIN0\_bm; //kokkino to fanari twn pezwn

PORTD.OUTCLR = PIN2\_bm; //prasino gia autokinhta

TCA0.SPLIT.HCNT = pushagain;

TCA0.SPLIT.HCMP0 = 0X00;

TCA0.SPLIT.INTCTRL =0x02; // HUNF INTERRUPT ENABLE

sei();

while (x==0) {

}

cli();

x=0;

button=0;

}

else{

PORTD.OUTCLR = PIN2\_bm; //prasino dromou

PORTD.OUT |=PIN0\_bm; //kokkino to fanari twn pezwn

}

cli();

}

if (y==1){

PORTD.OUT |= PIN2\_bm; //kokkino gia ta autokinhta

PORTD.OUTCLR = PIN0\_bm; //prasino gia pezous

PORTD.OUTCLR = PIN1\_bm; //prasino gia tram

TCA0.SPLIT.HCNT = ped;

TCA0.SPLIT.HCMP0 = 0X00;

TCA0.SPLIT.INTCTRL =0x02; // HUNF INTERRUPT ENABLE

sei();

while (x==0) {

}

cli();

y=0;

PORTD.OUT |=PIN0\_bm; //kokkino to fanari twn pezwn

PORTD.OUT |=PIN1\_bm; //kokkino to fanari tou tram

PORTD.OUTCLR = PIN2\_bm; //prasino gia autokinhta

x=0;

}

cli();

}

}

ISR(TCA0\_LCMP0\_vect){

//clear flag

int intflags = TCA0.SPLIT.INTFLAGS;

TCA0.SPLIT.INTFLAGS = intflags;

y=1;

}

ISR(TCA0\_HUNF\_vect){

//clear flag

int intflags = TCA0.SPLIT.INTFLAGS;

TCA0.SPLIT.INTFLAGS=intflags;

x=1;

}

ISR(PORTF\_PORT\_vect){

//clear the interrupt flag

int y = PORTF.INTFLAGS;

PORTF.INTFLAGS=y;

button++;

}

**2.** Στον κώδικα του πρώτου ερωτήματος, αρχικά, ορίζουμε δύο μεταβλητές x και button, οι οποίες αρχικοποιούνται με μηδέν, και χρησιμοποιούνται ως flags για το πότε έχει περάσει ο χρόνος του timer και πότε έχει πατηθεί το κουμπί των πεζών αντίστοιχα. Επίσης, ορίζουμε το PIN0 του PORTD ως έξοδο και το αρχικοποιούμε με 1, ώστε το LED για τους πεζούς να παραμείνει σβηστό. Έπειτα, ορίζουμε το PIN2 του PORTD ως έξοδο και το αρχικοποιούμε με 1, ώστε και το LED για τον μεγάλο δρόμο να παραμείνει σβηστό. Στη συνέχεια, κάνουμε enable το pullup και το interrupt για το PIN5 του PORTF, ώστε αυτό να χρησιμοποιηθεί ως το κουμπί των πεζών.

Ο υπόλοιπος κώδικας επαναλαμβάνεται συνεχώς, στη διάρκεια του οποίου γίνονται enable οι διακοπές και υπάρχουν δύο διαφορετικές περιπτώσεις. Στη πρώτη περίπτωση, αν το button είναι ίσο με 1, δηλαδή αν έχει πατηθεί το κουμπί των πεζών, τότε θέτουμε με 1 το PIN2 του PORTD και με 0 το PIN0 του PORTD, ώστε να είναι σβηστό το φανάρι του μεγάλου δρόμου και ανοιχτό το φανάρι των πεζών. Έπειτα, γίνεται clear ο counter του TCA, θέτουμε τον TCA σε normal mode, θέτουμε την τιμή που θέλουμε να γίνει το interrupt, δηλαδή ο χρόνος που θέλουμε να έχουν οι πεζοί για να περάσουν τον δρόμο, ορίζουμε τη τιμή του prescaler κα κάνουμε enable τον timer και τα interrupts. Μετά, όσο ο timer δεν έχει φτάσει ακόμη στη τιμή που έχουμε ορίσει, θα μείνει το πρόγραμμα σε ένα while loop. Όταν ο timer φτάσει στην τιμή που θέλουμε, θα γίνει διακοπή και θα καλεστεί η ISR του, στη διάρκεια της οποίας θα καθαριστούν τα flags και η τιμή του x θα αλλάξει προκειμένου να βγούμε από το while loop. Τότε, θα τεθεί ίσο με 1 το PIN0 του PORTD και με 0 το PIN2 του PORTD, ώστε να σβήσει το φανάρι των πεζών και να ανάψει το φανάρι των αυτοκινήτων. Τότε, καθαρίζει ο counter του timer και αρχίζει να τρέχει ο timer που καθορίζει πόσο χρόνο θα περιμένουν οι πεζοί μέχρι να μπορούν να ξαναπατήσουν το κουμπί. Όσο ο timer δεν έχει φτάσει ακόμη στη τιμή που έχουμε ορίσει, θα μείνει το πρόγραμμα σε ένα while loop. Όταν ο timer φτάσει στην τιμή που θέλουμε, θα γίνει διακοπή και θα καλεστεί η ISR του, στη διάρκεια της οποίας θα καθαριστούν τα flags και η τιμή του x θα αλλάξει προκειμένου να βγούμε από το while loop. Στη δεύτερη περίπτωση, αν το button είναι ίσο με 0, δηλαδή αν δεν έχει πατηθεί το κουμπί των πεζών, τότε θέτουμε με 0 το PIN2 του PORTD, ώστε να είναι ανοιχτό το φανάρι του μεγάλου δρόμου.

Στο δεύτερο ερώτημα κάναμε την υπόθεση πως, αφού δεν διαλέγουν οι πεζοί πότε θα περάσει το τραμ, δεν θα αρχίσει να μετριέται χρόνος αναμονής για να πατηθεί το κουμπί των πεζών αν μόλις έχει περάσει το τραμ και οι πεζοί δεν έχουν πατήσει το κουμπί.

Στον κώδικα του δεύτερου ερωτήματος, αρχικά, ορίζουμε τρεις μεταβλητές x, y και button, οι οποίες αρχικοποιούνται με μηδέν, και χρησιμοποιούνται ως flags για το πότε έχει περάσει ο χρόνος του timer για το διάστημα t2 και t3, ο χρόνος του timer για το διάστημα t1 και πότε έχει πατηθεί το κουμπί των πεζών αντίστοιχα. Όπως και προηγουμένως, τίθενται ως έξοδοι τα PIN0, PIN1 και PIN2 του PORTD και αρχικοποιούνται με 1, ώστε να είναι και τα τρία LED σβηστά. Ο υπόλοιπος κώδικας επαναλαμβάνεται συνεχώς. Στη διάρκεια του, κάνουμε enable το pullup και το interrupt για το PIN5 του PORTF, ώστε αυτό να χρησιμοποιηθεί ως το κουμπί των πεζών και κάνουμε enable το split mode στον timer TCA, ώστε με τα low bytes του να ελέγχουμε τον χρόνο του τραμ και με τα high bytes του να ελέγχουμε τον χρόνο που έχουν οι πεζοί για να περάσουν και τον χρόνο στη διάρκεια του οποίου δεν μπορεί να ξαναπατηθεί το κουμπί των πεζών. Επιπλέον, διαλέγουμε η σύγκριση των τιμών των δύο 8-bit timers να γίνεται στα channels HCMP0 και LCMP0, αρχικοποιούμε τον low byte counter με τον χρόνο που χρειάζεται το τραμ για να περάσει και την τιμή σύγκρισης με μηδέν, αφού στο split mode οι timers κάνουν count-down, ορίζουμε τη τιμή του prescaler και κάνουμε enable τον timer και το interrupt για το LCMP0. Αρχικοποιούμε τα interrupts και όσο δεν έχει φτάσει ο χρόνος του τραμ υπάρχουν δύο περιπτώσεις. Στη πρώτη περίπτωση, αν το button είναι ίσο με 1, δηλαδή αν έχει πατηθεί το κουμπί των πεζών, τότε θέτουμε με 1 το PIN2 του PORTD και με 0 το PIN0 του PORTD, ώστε να είναι σβηστό το φανάρι του μεγάλου δρόμου και ανοιχτό το φανάρι των πεζών. Έπειτα, θέτουμε το high byte counter του TCA ίσο με το χρονικό διάστημα που μπορούν να περάσουν οι πεζοί, την τιμή σύγκρισης στο high byte counter ίση με το 0 και ενεργοποιούμε το interrupt του TCA για underflow στα high bytes. Μετά, όσο ο timer δεν έχει φτάσει ακόμη στη τιμή που έχουμε ορίσει, θα μείνει το πρόγραμμα σε ένα while loop. Όταν ο timer φτάσει στην τιμή που θέλουμε, θα γίνει διακοπή και θα καλεστεί η ISR του high byte timer για high byte underflow, στη διάρκεια της οποίας θα καθαριστούν τα flags και η τιμή του x θα αλλάξει προκειμένου να βγούμε από το while loop. Τότε, θα τεθεί ίσο με 1 το PIN0 του PORTD και με 0 το PIN2 του PORTD, ώστε να σβήσει το φανάρι των πεζών και να ανάψει το φανάρι των αυτοκινήτων. Τότε, θέτουμε το high byte counter του TCA ίσο με το χρονικό διάστημα που χρειάζεται να περάσει μέχρι να μπορέσουν να ξαναπατήσουν το κουμπί για το φανάρι οι πεζοί, την τιμή σύγκρισης στο high byte counter ίση με το 0 και ενεργοποιούμε το interrupt του TCA για underflow στα high bytes. Όσο ο timer δεν έχει φτάσει ακόμη στη τιμή που έχουμε ορίσει, θα μείνει το πρόγραμμα σε ένα while loop. Όταν ο timer φτάσει στην τιμή που θέλουμε, θα γίνει διακοπή και θα καλεστεί η ISR του high byte timer για high byte underflow, στη διάρκεια της οποίας θα καθαριστούν τα flags και η τιμή του x θα αλλάξει προκειμένου να βγούμε από το while loop. Στη δεύτερη περίπτωση, αν το button είναι ίσο με 0, δηλαδή αν δεν έχει πατηθεί το κουμπί των πεζών, τότε θέτουμε με 0 το PIN2 του PORTD, ώστε να είναι ανοιχτό το φανάρι του μεγάλου δρόμου. Αν περάσει ο χρόνος του τραμ, τότε θα τεθεί 1 στο PIN2 του PORTD και 0 στο PIN0 και στο PIN1 του PORTD, για να σβήσει το φανάρι του μεγάλου δρόμου και να ανάψει το φανάρι για τους πεζούς και το τραμ. Έπειτα, θέτουμε το high byte counter του TCA ίσο με το χρονικό διάστημα που μπορούν να περάσουν οι πεζοί, την τιμή σύγκρισης στο high byte counter ίση με το 0 και ενεργοποιούμε το interrupt του TCA για underflow στα high bytes. Μετά, όσο ο timer δεν έχει φτάσει ακόμη στη τιμή που έχουμε ορίσει, θα μείνει το πρόγραμμα σε ένα while loop. Όταν ο timer φτάσει στην τιμή που θέλουμε, θα γίνει διακοπή και θα καλεστεί η ISR του high byte timer για high byte underflow, στη διάρκεια της οποίας θα καθαριστούν τα flags και η τιμή του x θα αλλάξει προκειμένου να βγούμε από το while loop. Τότε, θα τεθεί ίσο με 1 το PIN0 και το PIN1 του PORTD και με 0 το PIN2 του PORTD, ώστε να σβήσει το φανάρι των πεζών και του τραμ και να ανάψει το φανάρι των αυτοκινήτων.

Τα διαγράμματα ροής των ερωτημάτων είναι τα ακόλουθα:

1ου ερωτήματος:

Diagram

Description automatically generated

2ου ερωτήματος:

Diagram

Description automatically generated

Οι τιμές που επιλέχθηκαν ως t1, t2 και t3 είναι οι 50, 10 και 20 αντίστοιχα. Οι τιμές αυτές προέκυψαν από τον τύπο value=T\*ftimer, όπου Τ είναι ο χρόνος του χρονιστή και ftimer είναι η συχνότητα του ρολογιού, όπως προκύπτει από τον τύπο , με fsystem=20MHz και Ν=1024.