ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3

ΠΡΟΗΓΜΕΝΟΙ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ

Ομάδα

Ζήκος Σπύρος 1084581

Κυριακουλόπουλος Καλλίνικος 1084583

Αναλυτική Περιγραφή Κώδικα

Ορίζουμε σταθερές για την περίοδο των κυματομορφών και τα duty cycles. Επίσης, ορίζουμε μία μεταβλητή που υποδηλώνει την κατάσταση λειτουργίας του ανεμιστήρα (0: κλειστός, 1: κανονική λειτουργία, 2: γρήγορη λειτουργία).

“

int HPER = 40; // 2ms

int LPER = 20; // 1ms

int LPER2 = 8; // T = 1ms/2 = 0.5ms We used the formula that was given to calculate LPER2 from 0.5ms

int HCMP0 = 24; // duty cycle (60% of HPER)

int LCMP0 = 10; // duty cycle (50% of LPER)

int power\_mode = 0; // 0/1/2 times button pressed

“

int main()

Στη συνάρτηση main γίνεται η αρχικοποίηση του PIN5 του PORTF για interrupts ώστε να πατηθεί το κουμπί και να ενεργοποιηθεί ο ανεμιστήρας, δηλαδή να κληθεί η συνάρτηση setup\_tca\_split().

void setup\_tca\_split()

Με τη συνάρτηση αυτή αρχικοποιείται ο TCA0. Επιλέγουμε να λειτουργεί με clock division 1024 και σε split mode. Στη συνέχεια θέτουμε τους καταχωρητές HPER, LPER που οι τιμές τους ορίζουν την ανώτερη τιμή των κυματομορφών και τους καταχωρητές HCMP0, LCMP0 οι οποίοι ορίζουν πόσο διάστημα θα είναι high η κυματομορφή σύμφωνα με το duty cycle. Επίσης ενεργοποιούμε τα interrupts του TCA0 για HUNF και LUNF και τον ΤCA0 με λειτουργία split. Τέλος θέτουμε την κατεύθυνση για τα leds του PORTD και τα αρχικοποιούμε σβηστά.

void increase\_speed()

Όταν καλείται αυτή η συνάρτηση σημαίνει ότι έχει πατηθεί το κουμπί (SW5) για δεύτερη φορά και πρέπει να αυξηθεί η ταχύτητα των λεπίδων. Επομένως η μεταβλητή power\_mode γίνεται 2 και αλλάζει η περίοδος. Η αλλαγή της περιόδου γίνεται από την ανάθεση της νέας τιμής στους καταχωρητές LPER (max τιμή που φτάνει) και LCMP0 (νέο διάστημα που είναι high o

παλμός). Επιπλέον θα πρέπει να γίνει ένας έλεγχος για την περίπτωση όπου η τιμή που έχει ο LCNT είναι μεγαλύτερη από αυτή του LPER (μόλις άλλαξε σε μικρότερη). Αν είναι μεγαλύτερη και ο HCNT δεν έχει την ίδια τιμή με τον LPER (το θέλω για διαφορά φάσης) τότε ο LCNT παίρνει την τιμή του LPER, αλλιώς θα πάρει την τιμή του LPER μειωμένη κατά 5.

void shut\_down()

H συνάρτηση αυτή απενεργοποιεί τον timer TCA ώστε να σταματήσει να λειτουργεί ο ανεμιστήρας. Επίσης, κάνει την μεταβλητή power\_mode ίση με 0 ώστε να δηλώσει ότι το σύστημα βρίσκεται στην αρχική του κατάσταση. Επιπλέον, βάζει στις μεταβλητές LPER και LCMP0 τις αρχικές τους τιμές ώστε οι λεπίδες του ανεμιστήρα (όταν αυτός ξαναρχίσει να λειτουργεί) να έχουν την σωστή ταχύτητα και duty cycle.

ISR(TCA0\_HUNF\_vect) Η συνάρτηση αυτή διαχειρίζεται το underflow που προκαλείται από τα high bits του TCA. Κάθε φορά που γίνεται αυτό το interrupt, καθαρίζονται οι σημαίες των interrupts και επίσης αν το LED1 είναι ανοιχτό τότε το σβήνει ενώ αν είναι κλειστό το ανοίγει.

ISR(TCA0\_LUNF\_vect) Η συνάρτηση αυτή διαχειρίζεται το underflow που προκαλείται από τα low bits του TCA. Κάθε φορά που γίνεται αυτό το interrupt, καθαρίζονται οι σημαίες των interrupts και επίσης αν το LED0 είναι ανοιχτό τότε το σβήνει ενώ αν είναι κλειστό το ανοίγει.

ISR(PORTF\_PORT\_vect) Η συνάρτηση αυτή διαχειρίζεται το interrupt που προκαλείται από το πάτημα του κουμπιού (PIN5 του PORTF). Κάθε φορά που γίνεται αυτό το interrupt, καθαρίζονται οι σημαίες των interrupts.

Αν η μεταβλητή power\_mode είναι:

* 0, τότε καλείται η συνάρτηση setup\_tca\_split() η οποία αρχικοποιεί τον TCA (σε split mode) και τα 2 LEDS.
* 1, τότε καλείται η συνάρτηση increase\_speed() η οποία υποδιπλασιάζει την περίοδο λειτουργίας των λεπίδων του ανεμιστήρα.
* 2, τότε καλείται η συνάρτηση shut\_down() η οποία σταματά την λειτουργία του ανεμιστήρα.

A screenshot of a computer flowchart

Description automatically generatedΔΙΑΓΡΑΜΜΑ ΡΟΗΣ

Using: <https://app.diagrams.net>

Κυματομορφές

Fpwm\_ss(high) = = => 40\*1000 = 1024\*PER + 1024 =>

=> PER = 40000-1024 / 1024 => PER = 38.0625 ~ 40 (για να δίνει ακέραιο αριθμό όταν πολλαπλασιάζεται με το 0.6)

Fpwm\_ss(low) = = => 20\*1000 = 1024\*PER + 1024 =>

=> PER = 20000-1024 / 1024 => PER = 18.53125 ~ 20 (επειδή μπορεί να διαιρεθεί δύο φορές με το 2 ώστε την πρώτη να πάρουμε το PER για την επιτάχυνση και την δεύτερη το LCMP0 για την επιτάχυνση)

Fpwm\_ss(low-speedup) = = => 10\*1000 = 1024\*PER + 1024 =>

=> PER = 10000-1024 / 1024 => PER = 8.765625 ~ 8 (επειδή διαιρείται με το 2)

A black screen with white lines

Description automatically generated

Using: <https://app.diagrams.net>

Κώδικας

#include<avr/io.h>

#include<avr/interrupt.h>

int HPER = 40; // 2ms

int LPER = 20; // 1ms

int LPER2 = 8; // T = 1ms/2 = 0.5ms We used the formula that was given to calculate LPER2 from 0.5ms

int HCMP0 = 24; // duty cycle (60% of HPER)

int LCMP0 = 10; // duty cycle (50% of LPER)

int power\_mode = 0; // 0/1/2 times button pressed

// TCA high = led1 = basi = 60% duty cycle

// TCA low = led0 = lepides = 50% duty cycle

int main(){

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

sei();

while(1){sei();}

}

void setup\_tca\_split() {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Timer/PWM Setup \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

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

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

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

TCA0.SPLIT.LPER = LPER;

TCA0.SPLIT.HPER = HPER;

TCA0.SPLIT.LCMP0 = LCMP0; // select the duty cycle

TCA0.SPLIT.HCMP0 = HCMP0; // select the duty cycle

TCA0.SPLIT.LCNT = 5; // random init

TCA0.SPLIT.HCNT = 15; // random init (different from LCNT so we have different phases)

TCA0.SPLIT.INTCTRL = TCA\_SPLIT\_HUNF\_bm; // enable hunf interrupt

TCA0.SPLIT.INTCTRL |=TCA\_SPLIT\_LUNF\_bm; // enable lunf interrupt

TCA0.SPLIT.CTRLA |= TCA\_SPLIT\_ENABLE\_bm; // start counting

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* PORTD Setup \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PORTD.DIR |= 0b00000011; // enable LED0 and LED1

PORTD.OUT |= 0b00000000; // initially LED0 and LED1 are on (0)

// change mode

power\_mode = 1;

}

void increase\_speed(){

power\_mode = 2; // change mode

TCA0.SPLIT.LPER = LPER2; // Make period half

TCA0.SPLIT.LCMP0 = LPER2/2; // Make clock cycle 50%

// Normally LCNT is between 0 and LPER

// But when (above) we do LPER=LPER/2 then LCNT must be between 0 and LPER/2

// So if LCNT > LPER/2 then make it equal to LPER/2

// But if LPER/2 is equal to HCNT then you should change the value of LCNT (decrease it by 5) so that they have a phase difference

if (TCA0.SPLIT.LCNT > TCA0.SPLIT.LPER) {

if (TCA0.SPLIT.LPER != TCA0.SPLIT.HCNT) {

TCA0.SPLIT.LCNT = TCA0.SPLIT.LPER;

} else {

TCA0.SPLIT.LCNT = TCA0.SPLIT.LPER - 5;

}

}

}

void shut\_down() {

TCA0.SPLIT.CTRLA = 0; // disable timer

power\_mode = 0; // change mode

// reset variables so that it starts at normal mode and not with increased speed

TCA0.SPLIT.LPER = LPER;

TCA0.SPLIT.LCMP0 = LCMP0;

}

ISR(TCA0\_HUNF\_vect){

cli();

int intflags = TCA0.SPLIT.INTFLAGS;

TCA0.SPLIT.INTFLAGS = intflags;

if ((PORTD.OUT & 0b00000010) == 0b00000010) { // check LED1

PORTD.OUTCLR = PIN1\_bm; // LED is on

} else {

PORTD.OUT |= PIN1\_bm; // LED is off

}

sei();

}

ISR(TCA0\_LUNF\_vect){

cli();

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS = intflags;

if ((PORTD.OUT & 0b00000001) == 0b00000001) { // check LED0

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

} else {

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

}

sei();

}

/\* Button \*/

ISR(PORTF\_PORT\_vect){

cli();

int y = PORTF.INTFLAGS;

PORTF.INTFLAGS=y;

if (power\_mode == 0) { // when button first time pressed, setup tca

setup\_tca\_split();

} else if (power\_mode == 1) { // when button second time pressed, increase speed of blades

increase\_speed();

} else if (power\_mode == 2) { // when button third time pressed, shut down the fan

shut\_down();

}

sei();

}