

Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

**ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ**

**OMAΔΑ Β1 - ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3**

**Στοιχεία Φοιτητών:**

Αλευράς Ηλίας [up1069667@upnet.gr](mailto:up1069667@upnet.gr) 1069667

Σάββας Στυλιανού [up1069661@upnet.gr](mailto:up1069661@upnet.gr) 1069661

Για τον υπολογισμό της περιόδου χρησιμοποιούμε την πράξη που μας δίνεται από το **ΑΤmega4808/4809 Data Sheet(σελίδα.192)** όπου:

Aρχικά, έχουμε για την **κυκλική κίνηση της βάσης** ότι :

* **Παλμός περιόδου:**
* **Κύκλος λειτουργίας:**

Επιπρόσθετα, για την **κυκλική κίνηση των λεπίδων** ότι:

* **Παλμός περιόδου:**
* **Κύκλος λειτουργίας:**

**Διάγραμμα ροής**

A picture containing text, diagram, technical drawing, plan

Description automatically generated

**Κώδικας υλοποίησης και των 3ων ερωτημάτων**

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

int wait=0;

int x=1;

int button=0;

int main(void){

//PIN is output

PORTD.DIR |= 0b00000001; //PIN0\_bm lepides

//PIN is output

PORTD.DIR |= 0b00000010; //PIN1\_bm vasi

//PIN is output

PORTD.DIR |= 0b00000100; //PIN2\_bm adc

//LED is off

PORTD.OUT |= 0b00000001; //PIN0\_bm

//LED is off

PORTD.OUT |= 0b00000010; //PIN1\_bm

//LED is off

PORTD.OUT |= 0b00000100; //PIN2\_bm

//pullup enable and Interrupt enabled with sense on both edges

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

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();

while(x==1){

lepides\_off();

vasi\_off();

while(wait==0){

;

}

if (button==1){//energopiite o anemistiras normal mode

ADC0.CTRLA |= ADC\_ENABLE\_bm; //start adc

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

//TCA SPLIT MODE

//prescaler=1024

TCA0.SPLIT.CTRLD = 1;

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

TCA0.SPLIT.LPER = 19; //select the resolution/lepides

TCA0.SPLIT.LCMP0 = 10; //select the duty cycle/lepides 50%

TCA0.SPLIT.LCNT = 0; //Low Counter

TCA0.SPLIT.HPER = 38; //select the resolution/vasi

TCA0.SPLIT.HCMP0 = 15; //select the duty cycle/vasi 40%

TCA0.SPLIT.HCNT = 10; //High Counter // 10 gia kathisterisi na min simvenoun interrupt mazi

//select Single\_Slope\_PWM

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

//enable interrupt Overflow

TCA0.SPLIT.INTCTRL = TCA\_SPLIT\_HUNF\_bm | TCA\_SPLIT\_LUNF\_bm;

//enable interrupt CMP0

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

while (button==1){

;

}

}else if (button==2){//energopiite o anemistiras x2 mode

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

TCA0.SPLIT.CTRLD = 1;

TCA0.SPLIT.LPER = 38; //select the resolution/lepides(2x)

TCA0.SPLIT.LCMP0 = 20; //select the duty cycle/lepides 50%

TCA0.SPLIT.LCNT = 0; //Low Counter

TCA0.SPLIT.HPER = 38; //select the resolution/vasi

TCA0.SPLIT.HCMP0 = 15; //select the duty cycle/vasi 40%

TCA0.SPLIT.HCNT = 10; //High Counter 10 gia kathisterisi na min simvenoun interrupt mazi

//select Single\_Slope\_PWM

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

//enable interrupt Overflow

TCA0.SPLIT.INTCTRL = TCA\_SPLIT\_HUNF\_bm | TCA\_SPLIT\_LUNF\_bm;

//enable interrupt CMP0

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

while(button==2){

;

}

}else if (button==3){//apenergopiite o anemistiras

wait=0;

button=0;

//apenergopiisi tca adc

ADC0.CTRLA &= ~ADC\_ENABLE\_bm; //stop conversion

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

}

}

}

void lepides\_on(){

//on

PORTD.OUTCLR= 0b00000001; //PIN0\_bm

}

void lepides\_off(){

//LED is off

PORTD.OUT |= 0b00000001; //PIN0\_bm

}

void vasi\_on(){

//on

PORTD.OUTCLR= 0b00000010; //PIN1\_bm

}

void vasi\_off(){

//LED is off

PORTD.OUT |= 0b00000010; //PIN1\_bm

}

void adc\_on(){

//on

PORTD.OUTCLR= 0b00000100; //PIN2\_bm

}

void adc\_off(){

//LED is off

PORTD.OUT |= 0b00000100; //PIN2\_bm

}

ISR(PORTF\_PORT\_vect){

//clear the interrupt flag

int y = PORTF.INTFLAGS;

PORTF.INTFLAGS=y;

adc\_off();

wait=1;

button++;

}

//Aneveni pros psili stathmi

ISR(TCA0\_LUNF\_vect){

//clear the interrupt flag

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS = intflags;

//Kinisi Vasis On

lepides\_on();

//Kinisi Lepidon On

vasi\_on();

}

//Proxora pros xamili stathmi

ISR(TCA0\_HUNF\_vect){

//clear the interrupt flag

int intflags = TCA0.SINGLE.INTFLAGS;

TCA0.SINGLE.INTFLAGS = intflags;

//Kinisi Vasis Off

lepides\_off();

//Kinisi Lepidon Off

lepides\_off();

}

ISR(ADC0\_WCOMP\_vect){

int intflags = ADC0.INTFLAGS;

ADC0.INTFLAGS = intflags;

adc\_on();

wait=0;

button=0;

ADC0.CTRLA &= ~ADC\_ENABLE\_bm; //stop conversion

//apenergopiisi adc kai tca

ADC0.CTRLA &= ~ADC\_ENABLE\_bm; //stop conversion

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

}

**Αναφορά:**

Μας ζητήθηκε να υλοποιήσουμε την κίνηση ενός ανεμιστήρα ο οποίος θα αποτελείται από δύο περιστροφικές κινήσεις, την κίνηση των λεπιδών και την κίνηση της βάσης του ανεμιστήρα. Αυτές οι δύο περιστροφικές κινήσεις θα καθορίζονται από δύο διαφορετικούς Παλμοευρικούς Διαμορφωτές.(PWMs), με LED0(Λεπίδα),LED1(Βάση).

Αρχικά, χρησιμοποιήσαμε ένα TCA0 SPLIT 16bit-χρονιστή όπου στα 8 low bit βάλαμε τη λειτουργία της λεπίδας και στα 8 high bit τη λειτουργία της βάσης.

Το πρώτο ερώτημα μας ζήταγε να ενεργοποιήσουμε τα δύο LEDακια με την ενεργοποίηση για πρώτη φορά του SWITCH5 του PORTF. Για να πετύχουμε αυτή τη λειτουργία υπολογίσαμε και χρησιμοποιήσαμε τα PER και CMP0 για την κίνηση της λεπίδας και της βάσης και έτσι με δύο ISR το ένα για LUNF όπου αφορά τη λειτουργία της λεπίδας και το HUNF για τη λειτουργία της βάσης. Για να αποφύγουμε την παράλληλη ανερχόμενη παρυφή έχουμε χρησιμοποιήσει την χρονοκαθυστέρηση στους δύο παλμούς έτσι ώστε να πετύχουμε τη λειτουργία τους. Το πετύχαμε με τη χρήση αυτών του εντολών:

* TCA0.SPLIT.LCNT = 0; //Low Counter
* TCA0.SPLIT.HCNT = 10; //High Counter

Επιπρόσθετα, στο δεύτερο ερώτημα ζήταγε τη προσθήκη του ADC μετατροπέα ο οποίος αν εντοπίσει μια τιμή μικρότερη από το κατώφλι.(δηλαδή, ότι υπάρχει κάποιο αντικείμενο κοντά στις λεπίδες, άρα πρέπει να σταματήσουν να περιστρέφονται).Η τιμή που έχουμε ορίσει για το κατώφλι είναι : ADC0.WINLT |= 10; //Set threshold ,τότε θα σταματήσει η λειτουργία των λεπίδων, απενεργοποίηση LED0 LED1 και ενεργοποίηση LED2. Αυτό το πετύχαμε με τη χρήση ενός ISR για το ADC και έχουμε ενεργοποιήσει του LED2 και απενεργοποίηση του TCA και ADC.

Επιπλέον, για τη ενεργοποίηση του ανεμιστήρα για δεύτερη φορά με τη χρήση Switch5 του PORTF μας ζητήθηκε να διπλασιάσουμε την περίοδο της κυκλικής κίνησης των λεπίδων ,αυτό το πετύχαμε με τη χρήση ενός button, όπου αν πατηθεί το κουμπί ενεργοποιείται η ISR του PORTF και έτσι το button=2, όπου εκεί διπλασιάσαμε το PER και CMP0 για το LUNF όπου αφορούσε την κίνηση της λεπίδας και πετύχαμε αυτό που ήθελε το ερώτημα.

Tέλος, το τρίτο ερώτημα μας ζήταγε να υλοποιήσουμε την απενεργοποίησή του ανεμιστήρα με το πάτημα του κουμπιού για 3η φορά. Με το ίδιο σκεπτικό αν το button=3 όπου αυτό το πετύχαμε με την ενεργοποίηση του ISR PORTF όπου έχουμε βάλει ένα counter, όποτε πατηθεί το κουμπί, και έτσι εισέρχεται σε ένα loop όπου ο μετρητής αυτός γίνεται 0 όπου θα απενεργοποιήσει το LES0,LED1,LED2 τον μετατροπέα ADC και τον ΤCA