ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ

ΟΜΑΔΑ Β1 - ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3

Στοιχεία Φοιτητών:

Αλευράς Ηλίας up1069667@upnet.gr 1069667

Σάββας Στυλιανού up1069661@upnet.gr 1069661

Για τον υπολογισμό της περιόδου χρησιμοποιούμε την πράξη που μας δίνεται από το **ATmega4808/4809 Data Sheet(σελίδα.192)** όπου:

$$f_{PWM\_SS} = \frac{f_{CLK\_PER}}{N(PER+1)}$$

- $f_{CLK_{PER}} = 20MHz$
- $N = 1024$

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

- **Παλμός περιόδου:$\boldsymbol{Tb = 2ms}$**
- **Κύκλος λειτουργίας: $\boldsymbol{Db = 40\%}$**

$$\frac{1}{2ms} = \frac{20MHz}{1024(PER+1)} =>$$

$$1024(PER+1) = 20MHz * 2ms =>$$

$$PER = \frac{20MHz * 2ms - 1024}{1024} => \boldsymbol{PER = 38.06 \approx 38}$$

$$\boldsymbol{Κύκλος\ Λειτουργίας(Db)} = 40\% = 38 * 0.4 = \boldsymbol{15.2 \approx 15}$$

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

- **Παλμός περιόδου:$\boldsymbol{Tl = 1ms}$**
- **Κύκλος λειτουργίας: $\boldsymbol{Dl = 50\%}$**

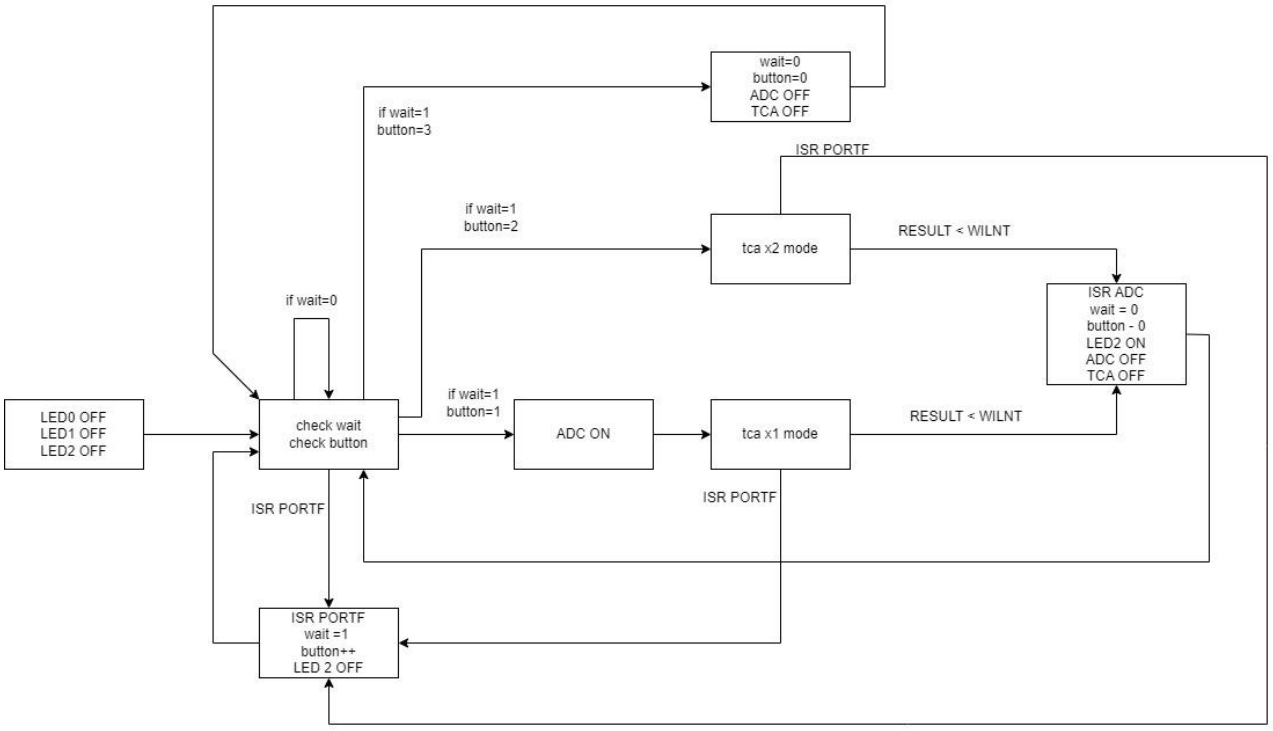
$$\frac{1}{2ms} = \frac{20MHz}{1024(PER+1)} =>$$

$$1024(PER+1) = 20MHz * 1ms =>$$

$$PER = \frac{20MHz * 1ms - 1024}{1024} => \boldsymbol{PER = 18.53 \approx 19}$$

$$\boldsymbol{Κύκλος\ Λειτουργίας(Db)} = 50\% = 19 * 0.5 = \boldsymbol{9.5 \approx 10}$$

Διάγραμμα ροής



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.

Τέλος, το τρίτο ερώτημα μας ζήταγε να υλοποιήσουμε την απενεργοποίησή του ανεμιστήρα με το πάτημα του κουμπιού για $3^{η}$ 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 TCA