ΕΡΓΑΣΤΗΡΙΟ ΠΡΟΗΓΜΕΝΟΙ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ

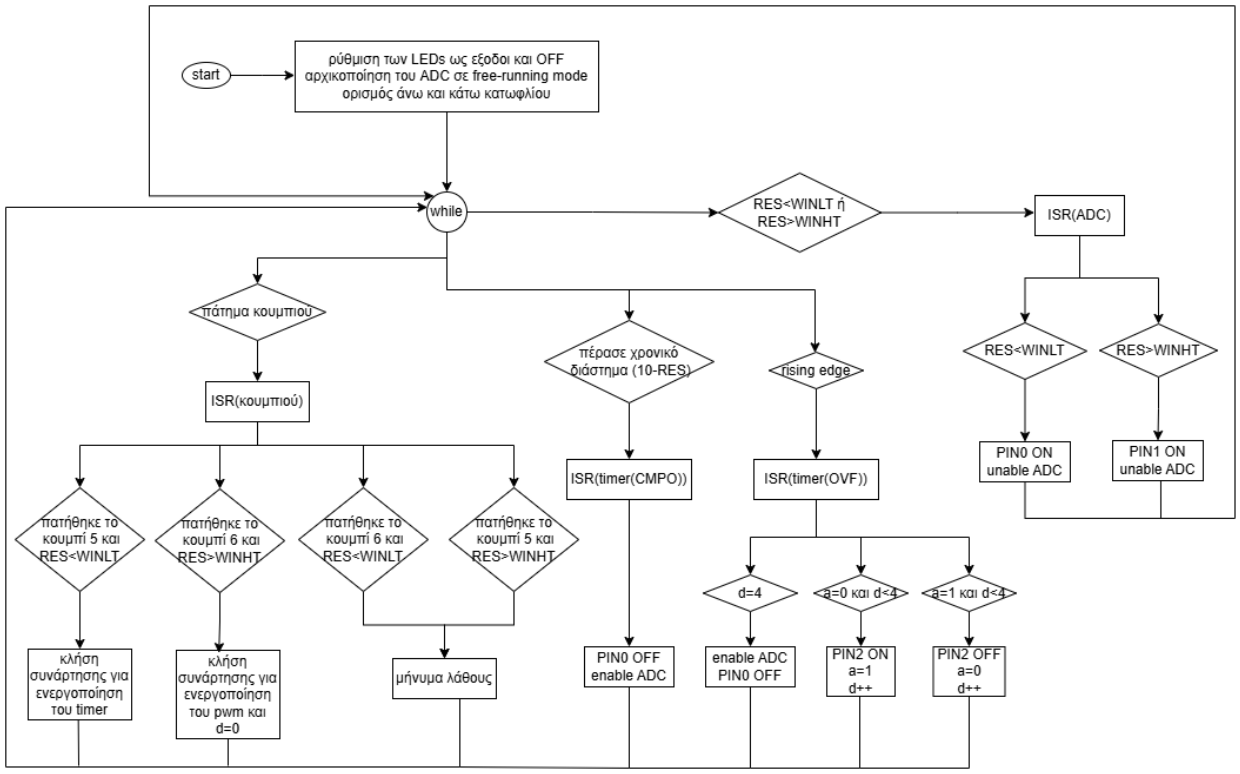
Εργαστηριακή Άσκηση 05:

Λειτουργία Έξυπνου Θερμοκηπίου

Χαράλαμπος Κωνσταντακόπουλος, 1090059

Εβελίνα Σενή, 1080416

<u>Διάγραμμα ροής</u>:



<u>Κώδικας:</u>

```
#include <avr/io.h>
#include <avr/interrupt.h>

int y=0; //irrelevant
int a=0; //on off leds ston palmo
int d=0; //counter gia ta rising edges

void Timer(value){
	//initialize timer
	TCA0.SINGLE.CNT = 0; //clear counter
	TCA0.SINGLE.CTRLB = 0;			//Normal Mode
```

```
	TCA0.SINGLE.CMP0 = value;   //When CMP0 reaches this value -> interrupt
	TCA0.SINGLE.CTRLA=TCA_SINGLE_CLKSEL_DIV1024_gc;
	TCA0.SINGLE.INTCTRL = TCA_SINGLE_CMP0_bm; //Interrupt Enable (=0x10)
	TCA0.SINGLE.CTRLA |= 1;            //Enable
}

void PWM(){
	//prescaler=1024
	TCA0.SINGLE.CTRLA=TCA_SINGLE_CLKSEL_DIV1024_gc;
	TCA0.SINGLE.PER = 18; //select the resolution
	TCA0.SINGLE.CMP0 = 9; //select the duty cycle
	TCA0.SINGLE.CTRLB |= TCA_SINGLE_WGMODE_SINGLESLOPE_gc;//select Single_Slope_PWM
	TCA0.SINGLE.INTCTRL |= TCA_SINGLE_OVF_bm;//enable interrupt Overflow
	TCA0.SINGLE.INTCTRL &= ~TCA_SINGLE_CMP0_bm;//disable interrupt CMP0
	TCA0.SINGLE.CTRLA |= TCA_SINGLE_ENABLE_bm; //Enable
	d=0; //reset rising edge counter
}
int main() {

	PORTD.DIR |= PIN0_bm | PIN1_bm | PIN2_bm ; //PINs are output
	PORTD.OUT |= PIN0_bm | PIN1_bm | PIN2_bm;  //LEDs are off

	//pullup enable and Interrupt enabled with sense on both edges
	PORTF.PIN5CTRL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
	PORTF.PIN6CTRL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;

	//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
	ADC0.DBGCTRL |= ADC_DBGRUN_bm;            //Enable Debug Mode

	//Window Comparator Mode
	ADC0.WINLT |= 10;                         //Set low threshold
	ADC0.WINHT |= 50;                         //Set high threshold
	ADC0.INTCTRL |= ADC_WCMP_bm;       //Enable Interrupts for WCM
	ADC0.CTRLE |= 0x4;          //Interrupt when RESULT < WINLT  or RESULT > WINHT

	ADC0.COMMAND |= ADC_STCONV_bm; //Start Conversion

	sei();
	while (y==0) {
		;
	}
	cli();
}

ISR(PORTF_PORT_vect){
	cli();

	if(PORTF.INTFLAGS & (1 << 5) && ADC0.RES > ADC0.WINHT){
		PORTD.OUTCLR |= PIN0_bm | PIN1_bm | PIN2_bm; //LEDs are on
		PORTD.OUT |= PIN0_bm | PIN1_bm | PIN2_bm;  //LEDs are off
	}else if (PORTF.INTFLAGS & (1 << 6)&& ADC0.RES < ADC0.WINLT){
		PORTD.OUTCLR |= PIN0_bm | PIN1_bm | PIN2_bm; //LEDs are on
		PORTD.OUT |= PIN0_bm | PIN1_bm | PIN2_bm;  //LEDs are off
	}else if(PORTF.INTFLAGS & (1 << 5) && ADC0.RES < ADC0.WINLT){
```

```
		Timer(10 - ADC0.RES);
	}else if(PORTF.INTFLAGS & (1 << 6) && ADC0.RES > ADC0.WINHT){
		PWM();
	}

	int y = PORTF.INTFLAGS; //Procedure to
	PORTF.INTFLAGS=y;
	sei();
}

ISR(TCA0_CMP0_vect){
	cli();

	TCA0.SINGLE.CTRLA = 0;//Disable
	int intflags = TCA0.SINGLE.INTFLAGS;
	TCA0.SINGLE.INTFLAGS=intflags;

	ADC0.CTRLA |= ADC_ENABLE_bm;
	ADC0.COMMAND |= ADC_STCONV_bm; //Start Conversion
	PORTD.OUT |= PIN0_bm;
	sei();
}

ISR(ADC0_WCOMP_vect){
	cli();

	ADC0.CTRLA &= ~ADC_ENABLE_bm; //disable ADC

	int intflags = ADC0.INTFLAGS;			//Procedure to
	ADC0.INTFLAGS =intflags;			//clear interrupt flag

	if(ADC0.RES < ADC0.WINLT){
			PORTD.OUTCLR= PIN0_bm; //sima anagkis potismatos

	}else if(ADC0.RES > ADC0.WINHT){
			PORTD.OUTCLR= PIN1_bm; //sima anagkis aerismou
	}
	sei();
}

ISR(TCA0_OVF_vect){
	cli();

	//clear the interrupt flag
	int intflags = TCA0.SINGLE.INTFLAGS;
	TCA0.SINGLE.INTFLAGS = intflags;

	if(d==4){
		ADC0.CTRLA |= ADC_ENABLE_bm;		//Enable ADC
		ADC0.COMMAND |= ADC_STCONV_bm;	//Start Conversion
		PORTD.OUT |= PIN0_bm; //OFF
		TCA0.SINGLE.INTCTRL &= ~TCA_SINGLE_OVF_bm;//disable interrupt Overflow
	}
	else if(a==0 && d<4){
		PORTD.OUTCLR |= PIN2_bm; //ON
		a=1;
		d++;
	}
```

```
        else if(a==1 && d<4){
                PORTD.OUT |= PIN2_bm; //OFF
                a=0;
                d++;
        }
        sei();
}
```

Αναφορά:

Στην main(), αρχικά ρυθμίζονται οι ακροδέκτες PD0, PD1 και PD2 του PORTD ως έξοδοι, στους οποίους είναι συνδεδεμένα τα τρία LEDs του συστήματος. Αρχικά όλα τα LEDs τίθενται σε λογικό 1, δηλαδή βρίσκονται σε κατάσταση OFF. Έπειτα, ρυθμίζονται οι διακόπτες SW5 και SW6 του PORTF (PF5 και PF6 αντίστοιχα), με ενεργοποίηση της pull-up αντίστασης και ανίχνευση διακοπής σε κάθε ακμή. Ακολουθεί η ενεργοποίηση και αρχικοποίηση του ADC σε λειτουργία Free-Running mode,. Ο ADC ρυθμίζεται να λειτουργεί με δύο threshold: το κατώτερο WINLT = 10 και το ανώτερο WINHT = 50. Αν η τιμή του ADC βγει εκτός αυτών των ορίων, ενεργοποιείται αυτόματα η διακοπή ADC0_WCOMP_vect. Το πρόγραμμα εισέρχεται στον βρόχο αναμονής while και μένει εκεί μέχρι να ενεργοποιηθεί κάποια διακοπή, η οποία μπορεί να αλλάξει την τρέχουσα κατάσταση. Από αυτό το σημείο και μετά, η οποιαδήποτε αλλαγή γίνεται μέσω και των ISR. Όταν η τιμή του ADC βγει εκτός threshold, ενεργοποιείται η διακοπή ADC0_WCOMP_vect. Αν η τιμή RES είναι μικρότερη του WINLT, δηλαδή πολύ χαμηλή υγρασία, ενεργοποιείται το LED0, το οποίο αντιστοιχεί σε ένδειξη ανάγκης ποτίσματος. Αν RES είναι μεγαλύτερη του WINHT, δηλαδή έχουμε υπερβολική υγρασία, τότε ενεργοποιείται το LED1 που υποδεικνύει την ανάγκη αερισμού. Σε κάθε περίπτωση, ο ADC απενεργοποιείται προσωρινά ώστε να περιμένει την απόκριση του χρήστη. Η επόμενη ISR που ενεργοποιείται είναι η PORTF_PORT_vect, στην οποία ο χρήστης μπορεί να πατήσει κάποιον από τους δύο διακόπτες. Το πρόγραμμα ελέγχει ποιος διακόπτης πατήθηκε και ποια είναι η κατάσταση της υγρασίας από την προηγούμενη μετατροπή. Αν πατηθεί το SW5 και η υγρασία είναι πολύ χαμηλή, δηλαδή RES < WINLT, τότε ενεργοποιείται η διαδικασία ποτίσματος. Καλείται η συνάρτηση Timer(10 - ADC.RES), η οποία ρυθμίζει τον Timer TCA0 σε Normal Mode, ώστε να δημιουργήσει μια καθυστέρηση που είναι ανάλογη της έλειψης υγρασίας. Όταν ο Timer φτάσει την καθορισμένη τιμή CMP0, ενεργοποιείται η διακοπή TCA0_CMP0_vect. Εκεί απενεργοποιείται ο Timer και ο ADC ενεργοποιείται ξανά για νέες μετρήσεις, ενώ το LED ποτίσματος (PD0) σβήνει. Αν ο χρήστης πατήσει το SW6 ενώ η υγρασία είναι υψηλή (RES > WINHT), ενεργοποιείται η λειτουργία αερισμού με τη χρήση του PWM. Καλείται η συνάρτηση PWM(), η οποία ρυθμίζει τον TCA0 σε λειτουργία Single Mode PWM, με περίοδο 1ms και duty cycle 50% (PER = 18, CMP0 = 9). Η διακοπή TCA0_OVF_vect ενεργοποιείται σε κάθε overflow του PWM, δηλαδή κάθε 1ms. Εκεί μετριούνται οι ανερχόμενες ακμές με έναν μετρητή d. Σε κάθε ακμή, το LED2 (PD2) αναβοσβήνει, και όταν φτάσουμε την τέταρτη ακμή, η PWM λειτουργία απενεργοποιείται και ο ADC ενεργοποιείται ξανά. Αν ο χρήστης πατήσει το λάθος κουμπί, για παράδειγμα SW5 ενώ η υγρασία είναι υψηλή, τότε ενεργοποιούνται και τα τρία LEDs ταυτόχρονα. Πρόκειται για μηχανισμό προειδοποίησης λανθασμένης ενέργειας. Στη συνέχεια, τα LEDs σβήνουν και το σύστημα επιστρέφει σε κατάσταση αναμονής, περιμένοντας την επόμενη έξοδο του ADC.