

Πανεπιστήμιο Πατρών

Τμήμα Μηχανικών Η/Υ και Πληροφορικής

**Εργαστηριακή Άσκηση 5**

Συνεργάτες:

Θεόφραστος Παξιμάδης 1093460

Κωνσταντίνος Αναστασόπουλος 1093320

Τμήμα: Β4

Ομάδα: 1

Μέρος 1: Κώδικας

/\*

\* lab5.c

\*

\* Created: 30/04/2025 16:16:52

\* Author : Theo & Dean

\*/

#include <avr/io.h>

#include <avr/interrupt.h>

int loop\_control = 0; // Chooses the block to be executed in the loop

int rising\_edges = 0; // PWM rising edges counter

int humidity = 2; // 0 -> humidity low, 1 -> humidity high, 2 -> humidity ok

int error = 0; // Error indicator

int main() {

// Leds setup

PORTD.DIR |= PIN0\_bm | PIN1\_bm | PIN2\_bm; // Pins 0, 1 and 2 are outputs

PORTD.OUT |= PIN0\_bm | PIN1\_bm | PIN2\_bm; // All initially turned off

// Setup ADC for free running mode

ADC0.CTRLA |= ADC\_RESSEL\_10BIT\_gc; // 10-bit resolution

ADC0.CTRLA |= ADC\_FREERUN\_bm; // Free-Running mode enabled

ADC0.CTRLE = ADC\_WINCM\_OUTSIDE\_gc; // Interrupt outside of the thresholds

ADC0.MUXPOS |= ADC\_MUXPOS\_AIN7\_gc; // Pin 7 used as input for ADC

ADC0.DBGCTRL |= ADC\_DBGRUN\_bm; // Enable debug mode

// Window comparator mode setup

ADC0.WINLT |= 8; // Low threshold

ADC0.WINHT |= 64; // High threshold

ADC0.INTCTRL |= ADC\_WCMP\_bm; // Enable Interrupts for WCM

// Pullup enable and sense on both edges for pins 5 and 6 (setup buttons)

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

PORTF.PIN6CTRL |= PORT\_PULLUPEN\_bm | PORT\_ISC\_BOTHEDGES\_gc;

// Timer/counter setup

TCA0.SINGLE.CTRLB = TCA\_SINGLE\_WGMODE\_NORMAL\_gc; // Normal mode

TCA0.SINGLE.CTRLA = TCA\_SINGLE\_CLKSEL\_DIV1024\_gc; // 1024 prescaler value

sei(); // Start accepting interrupts

while(1) { // Main loop

PORTD.OUT |= 0b00000000; // BREAKPOINT command

switch (humidity) { // Reset pins

case 0:

PORTD.OUT &= ~PIN0\_bm; // Turn on led 0 for low humidity

break;

case 1:

PORTD.OUT &= ~PIN1\_bm; // Turn on led 1 for high humidity

break;

default:

break;

}

if (loop\_control == 0) { // Reset, enable ADC again

loop\_control = 3; // Don't execute any block in the next iteration

if (error == 1) { // Reset the pins after error

PORTD.OUT |= PIN0\_bm | PIN1\_bm | PIN2\_bm; // Pins 0, 1 and 2 are turned off

error = 0; // Reset error variable

}

PORTD.OUT |= 0b00000000; // BREAKPOINT command

ADC0.CTRLA |= ADC\_ENABLE\_bm; // Enable ADC

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

}

if (loop\_control == 1) { // Handle low humidity

loop\_control = 3; // Don't execute any block in the next iteration

TCA0.SINGLE.INTCTRL = TCA\_SINGLE\_CMP0\_bm; // Compare channel 0 interrupt enable

TCA0.SINGLE.CTRLB = TCA\_SINGLE\_WGMODE\_NORMAL\_gc; // Normal mode

TCA0.SINGLE.CNT = 0; // Clear counter

int wait\_time = ADC0.WINLT - ADC0.RES; // Time window to water the plants

TCA0.SINGLE.CMP0 = wait\_time; // When CMP0 reaches this value -> interrupt BREAKPOINT command

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

}

if (loop\_control == 2) { // Handle high humidity

PORTD.OUT &= ~PIN1\_bm & ~PIN2\_bm; // Pins 1 and 2 initially turned on

loop\_control = 3; // Don't execute any block in the next iteration

TCA0.SINGLE.CTRLB |= TCA\_SINGLE\_WGMODE\_SINGLESLOPE\_gc; // Single slope PWM

TCA0.SINGLE.INTCTRL |= TCA\_SINGLE\_OVF\_bm; // Timer overflow/underflow interrupt enable

TCA0.SINGLE.INTCTRL |= TCA\_SINGLE\_CMP0\_bm; // Compare channel 0 interrupt enable

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

TCA0.SINGLE.PER = 20; // (Approximately 1ms period)

TCA0.SINGLE.CMP0 = 0.5 \* TCA0.SINGLE.PER; // 50% Duty cycle BREAKPOINT command

}

}

cli(); // Stop accepting interrupts

}

ISR(ADC0\_WCOMP\_vect) { // Humidity interrupt from the ADC

if (ADC0.RES < ADC0.WINLT) { // Humidity below threshold

PORTD.OUT &= 0b11111110; // Turn on Led 0

humidity = 0; // Low humidity

PORTD.OUT |= 0b00000000; // BREAKPOINT command

}

if (ADC0.RES > ADC0.WINLT) { // Humidity above threshold

PORTD.OUT &= 0b11111101; // Turn on led 1

humidity = 1; // High humidity

PORTD.OUT |= 0b00000000; // BREAKPOINT command

}

int intflags = ADC0.INTFLAGS; // Clear interrupt flags

ADC0.INTFLAGS = intflags;

ADC0.CTRLA &= ~ADC\_ENABLE\_bm; // Disable the ADC

}

ISR(PORTF\_PORT\_vect) { // Button interrupt from port f

*uint8\_t* humidity\_low = PORTF.INTFLAGS & PIN5\_bm; // Equals PIN5\_bm if pin 5 is pressed, else it's all 0s

*uint8\_t* humidity\_high = PORTF.INTFLAGS & PIN6\_bm; // Equals PIN6\_bm if pin 6 is pressed, else it's all 0s

if (humidity == 0 && humidity\_low) { // Pin 5 was used and humidity is low

loop\_control = 1; // Handle low humidity in the next iteration

PORTD.OUT |= 0b00000000; // BREAKPOINT command

}

if (humidity == 1 && humidity\_high) { // Pin 6 was used and humidity is high

loop\_control = 2; // Handle high humidity in the next iteration

PORTD.OUT |= 0b00000000; // BREAKPOINT command

}

if ((humidity == 0 && humidity\_high) || (humidity == 1 && humidity\_low) || (humidity == 2)) { // Wrong button pressed

loop\_control = 0; // Reset the ADC in the next iteration

PORTD.OUT &= ~PIN0\_bm & ~PIN1\_bm & ~PIN2\_bm; // Pins 0, 1 and 2 are turned on (error indicator) BREAKPOINT command

error = 1; // Set error variable BREAKPOINT command

}

int intflags = PORTF.INTFLAGS; // Clear the interrupt flags

PORTF.INTFLAGS = intflags;

}

ISR(TCA0\_CMP0\_vect) { // Time to stop watering the plants

if (humidity == 0) {

TCA0.SINGLE.CTRLA = ~TCA\_SINGLE\_ENABLE\_bm; // Disable the timer/counter

loop\_control = 0; // Reset the ADC in the next iteration

PORTD.OUT |= PIN0\_bm; // Pin 0 is turned off

humidity = 2; // Humidity is ok

int intflags = TCA0.SINGLE.INTFLAGS; // Clear the interrupt flags

TCA0.SINGLE.INTFLAGS = intflags;

}

}

ISR(TCA0\_OVF\_vect) { // Interrupt in rising edges

PORTD.OUT ^= PIN2\_bm; // Complement the value of pin 2

rising\_edges = rising\_edges + 1; // increment the rising edge counter BREAKPOINT command

if (rising\_edges == 4) { // Max number of rising edges reached

TCA0.SINGLE.CTRLA &= ~TCA\_SINGLE\_ENABLE\_bm; // Disable timer/counter

PORTD.OUT |= PIN1\_bm | PIN2\_bm; // Pins 1 and 2 are turned off BREAKPOINT command

loop\_control = 0; // Reset the ADC in the next iteration

rising\_edges = 0; // Reset the rising edges counter

humidity = 2; // Humidity is ok

}

int intflags = TCA0.SINGLE.INTFLAGS; // Clear the interrupt flags

TCA0.SINGLE.INTFLAGS = intflags;

}

Μέρος 2: Επεξήγηση Κώδικα και Διάγραμμα Ροής

Στην αρχή του κώδικα, αρχικοποιούνται τα leds του PORTD (led0, led1, led2), καθώς και τα στοιχεία του Atmega 4808 που θα χρησιμοποιηθούν στην συνέχεια. Ειδικότερα, αρχικοποιείται ο ADC σε free-running mode και ενεργοποιούνται οι διακοπές όταν οι τιμές του καταχωρητή RES είναι εκτός του ορίου των δύο thresholds, ενεργοποιούνται οι διακοπές από τα PIN5, PIN6 του PORTF και τέλος θέτουμε τον TCA0 σε normal mode.

Στην συνέχεια η main, επειδή αρχικά η global μεταβλητή loop\_control είναι 0, τρέχει το κομμάτι if (loop\_control == 0), στο οποίο ενεργοποιείται ο ADC και ξεκινάει το conversion. Εάν ο καταχωρητής RES του ADC πάρει τιμή μικρότερη ή μεγαλύτερη των ορίων του threshold, τότε καλείται η ISR(ADC0\_WCOMP\_vect). Εάν έχει παραβιαστεί το κάτω όριο (if (ADC0.RES < ADC0.WINLT)), τότε ανοίγει το led0 και η global μεταβλητή humidity γίνεται 0, που δηλώνει ότι πρόκειται για χαμηλή υγρασία. Επίσης, γίνεται disabled o ADC για να μην λάβει άλλες τιμές. Ομοίως για όταν παραβιάζεται το άνω threshold και η μεταβλητή humidity γίνεται 1. Στην συνέχεια, η ροή επιστρέφει στην main όπου αναμένεται να πατηθεί είτε το PIN5 είτε το PIN6 του PORTF. Στην περίπτωση οποιουδήποτε κουμπιού, καλείται η ISR(PORTF\_PORT\_vect). Εάν πατηθεί το λάθος κουμπί από αυτό που αναμένεται, τότε με την χρήση του bitmasking στα PORTF.INTFLAGS και την μεταβλητή humidity (if ((humidity == 0 && humidity\_high) || (humidity == 1 && humidity\_low))), ενεργοποιούνται όλα τα led για μία στιγμή, η global μεταβλητή loop\_control γίνεται 0 και εκτελείται πάλι η ίδια διαδικασία μέχρι ο χρήστης να πατήσει το σωστό κουμπί. Όταν ο χρήστης επιλέξει το σωστό κουμπί, τότε η μεταβλητή loop\_control γίνεται 1 εάν πρόκειται για χαμηλή υγρασία, αλλιώς 2 για υψηλή.

Επιστρέφοντας στην main, εάν το loop\_control είναι 1, τότε για χρονικό διάστημα ίσο με την διαφορά του χαμηλού threshold από την τιμή του καταχωρητή RES, μετράει ο TCA0. Όταν περάσει αυτό το χρονικό διάστημα, καλείται η ISR(TCA0\_CMP0\_vect), όπου σβήνει το led0, η μεταβλητή loop\_control γίνεται 0 και ξεκινάει όλη η διαδικασία από την αρχή.

Εάν το loop\_control είναι 2, τότε στην main, ενεργοποιείται το led2, και ρυθμίζεται ο TCA0 για την δημιουργία παλμού μήκους 1 ms και duty cycle 50 % () Κάθε ένα ms, καλείται η ISR(TCA0\_OVF\_vect), στην οποία αντιστρέφεται το LED2 και όταν κληθεί 4 φορές (if (rising\_edges == 4)), τότε το loop\_control γίνεται 0 και ξεκινάει όλη η διαδικασία από την αρχή.

A diagram of a flowchart

AI-generated content may be incorrect.

Διάγραμμα Ροής Λειτουργίας Συστήματος Θερμοκηπίου