# Introduction

The objective of the project was to use interrupts and timers to monitor and trigger different devices. Interrupts are methods that services the device when triggered without tying up the CPU to always monitor the status. This will help in higher efficiency of the CPU. Timers and counters are methods that can create delays and counts.

# Procedure

Project Proposal:

Specific tasks where to turn both the servo and stepper motor in clockwise direction when an object is 20 inches or greater from the ping sensor. If the object is at a distance between 5 and 20 inches, blink an LED (or all of them). The blinking speed of the LED is determined by the distanceand should blink slowly starting at 20 inches and incrementally “speed up” as you reach 5 inches. The servo and stepper motors should not turn onin this distance range. And If the object is 5 inches or closer towards the ping sensor, the LED should light up solid and both the servo and stepper motor should rotate counter clockwise with no delay and should display in inches on the LCD the distance. The servo should turn off when the object is further than 10 inches away. The LED(s) should return back to a blink if it is between 5 inches and 20 inches and should turn off for distances greater than 20 inches.

Main Operational Method:

Two interrupt service routines were used in the code for this project. One was Timer2 overflow interrupt and the other was pin change interrupt 3. The Timer2 overflow interrupt was triggered by the output compare flag and the pin change interrupt 3 service routine was triggered by the change in pin change interrupt flag.

Flow Chart:

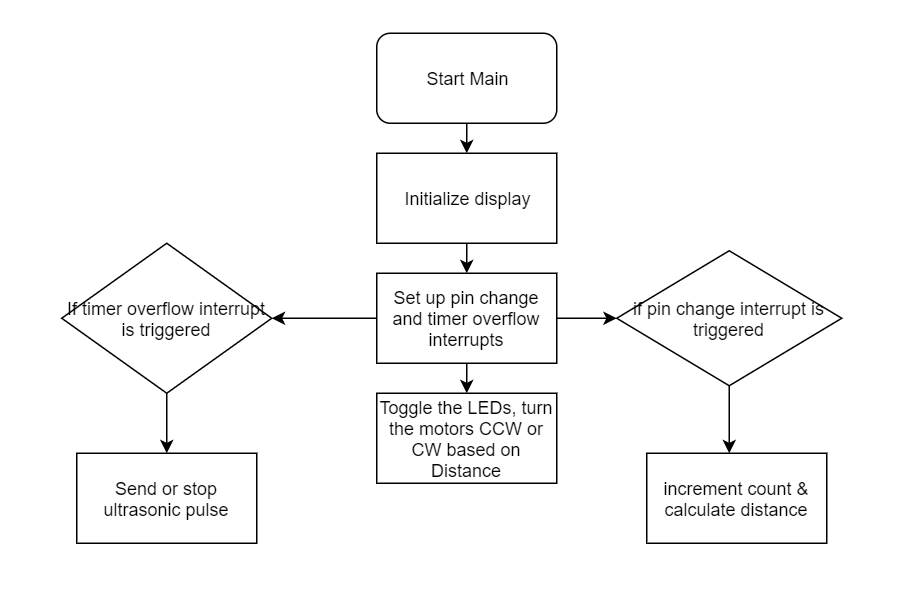


Figure 1. Flow chart

# Results

The system of devices worked as a unit to produce the expected the result. When an object was placed beyond the 20 inches mark from the Ping sensor, both the servo motor and stepper motor turned in the clockwise direction and as the object was moved to between the distance the distance of 20 and 5 inches, the PORTA LEDs started blinking with increasing speed. Meanwhile, the Servo motor and stepper motor stopped turning. And when the object was moved inside the 5 inches mark, the Servo motor and the stepper motor started turning counter clockwise. Meanwhile, the LEDs were light up solid.

# Discussion and Conclusion

The goal of the project was to use interrupts and timers effectively to monitor and trigger multiple devices all at the same time. The results were as expected.

# Appendix

Code:

#define *F\_CPU* 16000000UL // Define CPU frequency for delay.h

#include <util/delay.h>

#include <avr/interrupt.h>

#include "SSD1306.h"

//#include "SSD1306.c" //Not sure why I need to include this file here

// Pin definitions

#define PINGPin PD0

// Variable declarations

volatile int dist = 0;

char str [100];

int count = 0;

int main(void) {

*int8\_t* displayInverted = 1;

uint8\_t dispNum = 0x00;

char text[] = "TEMPLE MADE ECE";

char text1[] = "GO EAGLES!";

OLED\_Init(); //initialize the OLED

*\_delay\_ms*(1);

OLED\_Clear(); //clear the display (for good measure)

//set up Pin Change Interrupt Registers

PORTD |= (1<<0);// Set Pull up resistor PD0

PCICR |= (1<<PCIE3);// Pin Change Interrupt 1 Mask for PD0 is at PCIE3

PCIFR|=(1<<PCIF3);// Pin Change Interrupt Flag for PD0 is at PCIF3

PCMSK3 |= (1<<PCINT24) ; //Enable PCINT24

//set up TIMER2 Registers

TCCR2B = (1<<CS21);// Set TIMER2 to Normal Mode with 1:8 prescaler for 1st output pulse

TIMSK2 = (1<<TOIE2);// Unmask TIMER2 in Normal Mode

OCR2B = 255;// Set output compare register to 255

sei(); // Enable global interrupts

// set portb,a,d,e,as output

PORTB = 0xFF;

DDRA=0XFF;

DDRD=0XFF;

DDRE=0XFF;

while (1) {

dispNum++;

OLED\_SetCursor(0, 0); //set the cursor position to (0, 0)

OLED\_SetCursor(1, 0); // set the cursor position to (1, 0)

OLED\_DisplayString(text); // display text

OLED\_SetCursor(2, 0); //set the cursor position to (2, 0)

OLED\_DisplayString(text1); // display text1

OLED\_SetCursor(4, 0); //set the cursor position to (4, 0)

OLED\_Printf(" distance: ");

OLED\_DisplayNumber(C\_DECIMAL\_U8,dist,3);

if(dist >20) // if distance greater then 20

{

// servo and stepper motor run ccw

PORTD = 0b00000100;

*\_delay\_ms*(1.3);//\_delay\_ms(1.3);

PORTD = 0b00000000;

*\_delay\_ms*(20);//\_delay\_ms(20);

PORTE = 0x30;

*\_delay\_ms* (10);//\_delay\_ms(10);

PORTE = 0x60;

*\_delay\_ms* (10);//\_delay\_ms(10);

PORTE = 0x48;

*\_delay\_ms* (10);//\_delay\_ms(10);

PORTE = 0x18;

*\_delay\_ms* (10);//\_delay\_ms(10);

}

else if ( dist >5&& dist<20) //if between 5 and 20

{

// led light up

PORTA=0b11111111;

*\_delay\_ms*(dist);

PORTA=0b00000000;

*\_delay\_ms*(dist);

if (dist>10) // if distance greater then 10

{

//stepper motor run cw

PORTE = 0x30;

*\_delay\_ms* (10); //\_delay\_ms(10);

PORTE = 0x18;

*\_delay\_ms* (10);//\_delay\_ms(10);

PORTE = 0x48;

*\_delay\_ms* (10);//\_delay\_ms(10);

PORTE = 0x60;

*\_delay\_ms* (10);//\_delay\_ms(10);

}

}

else if (dist<=5) // if distance equal or smaller then 5

{

// servo motor and stepper motor run ccw

PORTA=0b11111111;

PORTE = 0x30;

*\_delay\_ms* (10); //\_delay\_ms(10);

PORTE = 0x18;

*\_delay\_ms* (10);//\_delay\_ms(10);

PORTE = 0x48;

*\_delay\_ms* (10);//\_delay\_ms(10);

PORTE = 0x60;

*\_delay\_ms* (10);//\_delay\_ms(10);

PORTD = 0b00000100;

*\_delay\_ms*(1.7); //\_delay\_ms(1.7);

PORTD = 0b00000000;

*\_delay\_ms*(20); //\_delay\_ms(20);

}

}

return 0; // never reached

}

ISR(TIMER2\_OVF\_vect) {

if (OCR2B == 255) {

TCCR2B = (1<<CS21);// Set TIMER2 to Normal Mode with 1:8 prescaler

DDRD |= (1<<PINGPin);// Set data direction of PD0 to output

PORTD |= (1<<PINGPin);// Send ultrasonic pulse

OCR2B = 3;// Set output compare register to 3

return;// Return to main

}

else if (OCR2B == 3) {

PORTD &= ~(1<<PINGPin);// Stop ultrasonic pulse

OCR2B = 2;// Set output compare register to 2

return;// Return to main

}

else if (OCR2B == 2) {

DDRD &= ~(1<<PINGPin);// Set data direction of PB0 to input

OCR2B = 255;// Set output compare register to 255

TCCR2B = (1<<CS22)|(1<<CS21);// Set TIMER2 to Normal Mode with 1:256 prescaler

return;// Return to main

}

}

ISR(PCINT3\_vect) {

count++;// Increment count

if (count == 3) TCNT2 = 0;// If ISR executed 3 times, reset timer

if (count == 4) {// If ISR executed 4 times

dist = (TCNT2\*.107);// Calculate distance

OCR2B = 255;// Set output compare register to 255

count = 0;// Reset count

return;// Return to main

}

return;// Return to main

}

Video Link:

<https://youtu.be/O0LTOHeG0xc>