**Lab 1:**

**Display a Virtual Recipe on an LCD**

**Mechatronics/Intel Mach Engr, ECE 4370-001 Spring 2020**

**Date Submitted:** January 31, 2020

**Coauthors:** Joey Franklin

Levi Carroll

**Signatures: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Recipient:** Dr. Stephen Canfield

**Executive Summary**

In this lab we were instructed to make a measurement device using an inferred (IR) transmitter/receiver module. The IR module blasts focused IR beam straight in from of it and measures the intensity of light that it receives back. Since light scatters in a predictable way the amount of scattered IR radiation it receives will give a reasonably actuate measurement of distance. There are however some problems with this system. Light will scatter differently when reflecting off of different colors. This can interfere with your measurements. For example, we calibrated the module using a highly reflective light blue folder, using that folder our measurements are within two tenths of a centimeter, however when we took measurements using my matte black phone case our readings were off by almost two centimeters. This is because more light was absorbed by the phone case that by the folder.

Another consideration of the IR module is that it is very sensitive to changes in it’s angle while you hold it and to interference, either due to atmospheric conditions or to the presence of unexpected IR radiation. The easiest and most effective way to overcome this is to take several measurements and then sort those measurement to find the median, or the average. We chose the median as this yielded the most accurate result however the average yields a more consistent and stable measurement.

**Circuit Diagram**

A screenshot of a cell phone

Description automatically generated

**Program Flowchart**

A close up of a sign

Description automatically generated

**Program Listing**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*  ECE4370 - Mechatronics                                      \*

\*  Dr. Stephen Canfield                                        \*

\*  LAB 4: Electronic tape measure using Sharp IR sensor        \*

\*         through ATD conversion                               \*

\* Written by: Joey Franklin                                    \*

\*        and  Levi Carroll                                     \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <**LiquidCrystal**.h>

// initialize the library by providing LCD pin locations

**LiquidCrystal** lcd(8,9,7,6,5,4);

// Declare Functions

void lcd\_display(String units, int val);

float find\_median(int n, float x[]);

// Variables to Hold Readings and Median of them

float ir\_volt[11] ={0,0,0,0,0,0,0,0,0,0,0};

float ir\_volt\_median = 0;

float measurement = 0;

const int Button = 0x01; //PIN 22

bool run\_measure= false;

bool lcd\_backlight = false;

// Setup

void setup() {

 //Set PA0 as input

 DDRA=0b00000000;

 /////////// Set up AtoD Converter //////////

 // Set Voltage Reference to 1024 = AVREF = 2.56 --> bit 7, bit 6

 // Left Justify Register Read For Easier Calculations --> bit 5

 // Use PF0 --> bit 4, bit 3, bit 2, bit 1, bit 0

 ADMUX = 0b01100000;

 // Enable AtoD Conversion --> bit 7

 // Start AtoD Conversions --> bit 6

 // Dont Use Auto Trigger --> bit 5

 // Clear Interrupt Flag --> bit 4

 // Enable Interrupts --> bit 3

 // Set pre-scaler to 16 --> bit 2, bit 1, bit0

 ADCSRA = 0b10000100;

 // Enable AtoD Conversion

 // Start AtoD Conversions

  // Initialize the Serial

**Serial**.begin(9600);

 // Wait for Serial to be ready

 while (! **Serial**);

 // print to the Serial port

**Serial**.println("Serial is ready");

 //Set LCD Size

 lcd.begin(16,2);

}

void loop() {

if(PINA&Button){

 delay(200);

 run\_measure = !run\_measure;

}

while(run\_measure){

 //// Select AtoD Channel (ADC0,PF0,pin97) ////

 ADMUX &= 0b11110000;

 ADMUX |= 0b00000000;

 //// Start AtoD Conversions (bit 6) ////

 ADCSRA |= 0b01000000;

 //// Wait until Conversion Complete Flag is set (bit 4) ////

 while(!(ADCSRA & 0b00010000));

 // Read High Register Data to Fill Pre-Sized Array //

 for(int num\_readings = 0; num\_readings <= 10; num\_readings++){

   ir\_volt[num\_readings] = ADCH;

 }

// Set the Value to be displayed on the LCD //

 ir\_volt\_median = find\_median(11,ir\_volt);

  // Convert Voltage to cm with curve fit from Excel 175.95e-0.026x

 measurement = (2772.8\*pow(ir\_volt\_median/2, -1.2045))/2;

// Print Value to the LCD //

 lcd\_display ("cm", measurement);

 if(PINA&Button){

   delay(200);

 run\_measure = !run\_measure;

 }

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Function Name: find\_median         \*

\* Recieves an Array to be            \*

\* sorted and its size                \*

\* Written By: Joey Franklin          \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

float find\_median(int n, float x[]) {

   int temp;

   int i, j;

   // the following two loops sort the array x in ascending order

   for(i=0; i<n-1; i++) {

       for(j=i+1; j<n; j++) {

           if(x[j] < x[i]) {

               // swap elements

               temp = x[i];

               x[i] = x[j];

               x[j] = temp;

           }

       }

   }

   if(n%2==0) {

       // if there is an even number of elements, return mean of the two elements in the middle

       return((x[n/2] + x[n/2 - 1]) / 2.0);

   } else {

       // else return the element in the middle

       return x[n/2];

   }

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Function Name: lcd\_display         \*

\* Revieves a number and string to    \*

\* display on the LCD                 \*

\* Written By: Joey Franklin          \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void lcd\_display(String units, float val){

   // print to the LCD

   lcd.setCursor(4,0);

   lcd.print("Distance:");

   lcd.setCursor(6,1);

   lcd.print(val,2);

   lcd.setCursor(14,1);

   lcd.print(units);

   // delay

   delay(300);

   lcd.clear();

  }