System Verification Report

Non-Contact Temperature Scanner

Group 3

Junior Design II

Preston Hang, Jia Wei Cheng, Budsakol Tiangdah, Sultan Aldhaheri

Table of Contents

Top Level Diagram

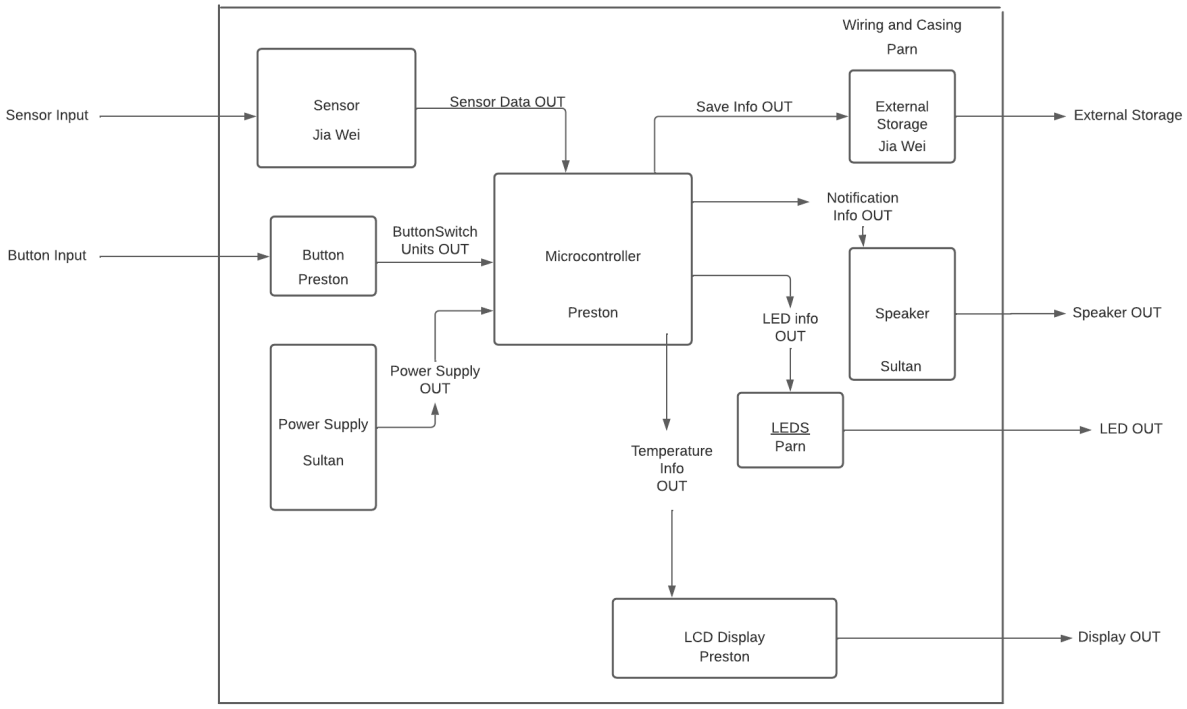

Figure 1: Top Level Diagram

System Interfaces and Properties

Interface	**Definition**
Sensor Input	Inputs: - 5V input Voltage - GND Connection - Sensor Analog Temperature Sensing
Button Input	Inputs - 5V input Voltage - GND Connection - Analog Button Press

	- Button Active Low
Sensor Data OUT	Outputs: - Sensor should pick up the temperature and send that information to the microcontroller. - SDA, serial data connection to arduino. - SCL, serial clock connection to arduino.
Temperature Info OUT	Outputs: - Microcontroller outputs the recorded temperature and sends it to the LCD display. - SDA, serial data connection to arduino. - SCL, serial clock connection to arduino.

LED Info OUT	Outputs:
	- Depending on the recorded temperature, the microcontroller sends different voltage values to change LED colors.
	  - **Red (Fever):** 104.5℉ (38℃) or higher
	  - **Green (Healthy):** Less than 104.5℉ (38℃), greater than 95℉ (35℃)
	  - **Blue (Hypothermia):** 95℉ (35℃) or lower

Notification Info OUT	Outputs: - If the temperature is at Fever level, the microcontroller sends a signal to play a noise from the speaker and display corresponding color in the LED.

Display OUT	Outputs:
	- Measured temperature is outputted to the user with the LCD display.
	  - LCD input voltage: 3.1-3.5V
	  - Supply Current Max: 2.5mA

LED OUT	Outputs: - RGB Light color is outputted to the user with the LED depending on the measured temperature.
Speaker OUT	Outputs: - Notification sound is outputted to the user with the speaker if the measured temperature is above 105.5℉.
Save Info OUT	Outputs: - Measured temperatures should be saved and logged onto micro-SD card.
Enclosure	Durable, able to fit in all the components, has holes for sensor, button, RGB LED, LCD display, speaker, easy to open and close for maintenance Using 3D printing method, consist of 2 parts: back and front hold them together by bolts and nuts

Electrical Schematic

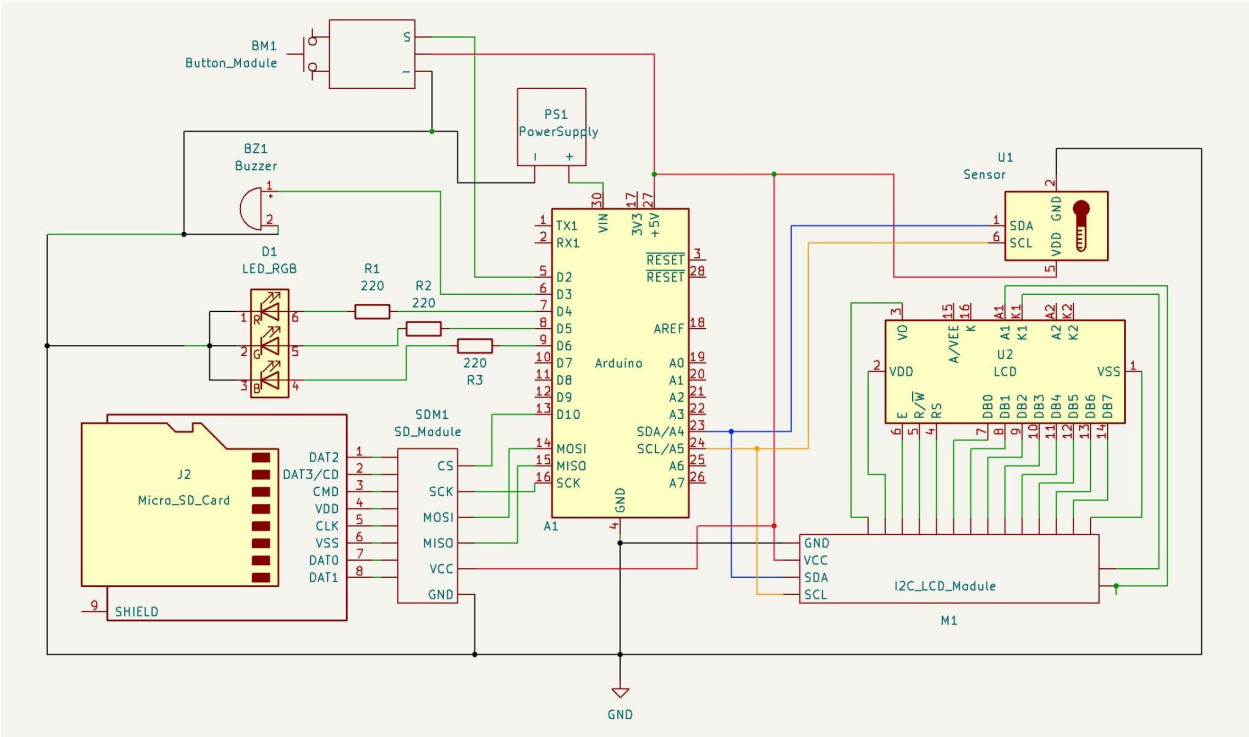

Figure 2: KiCAD Schematic

Mechanical Drawing

Sensor

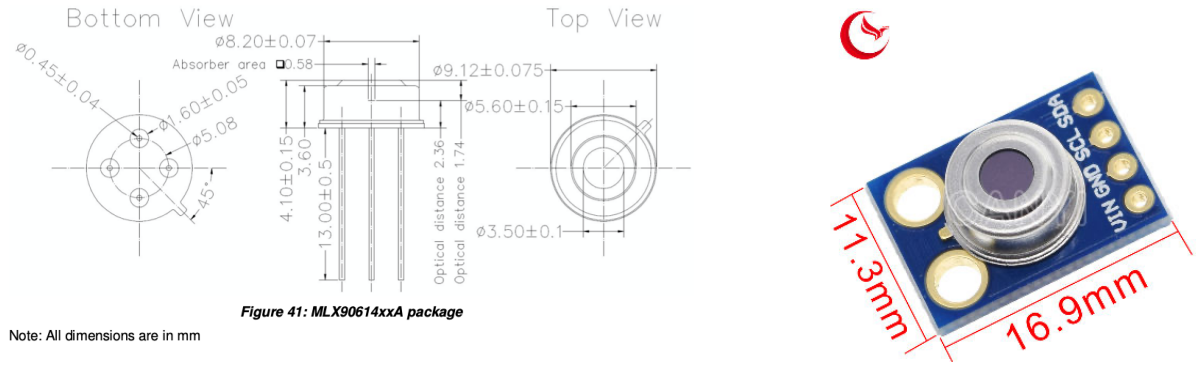

Figure 3: Sensor Mechanical Drawing

LCD

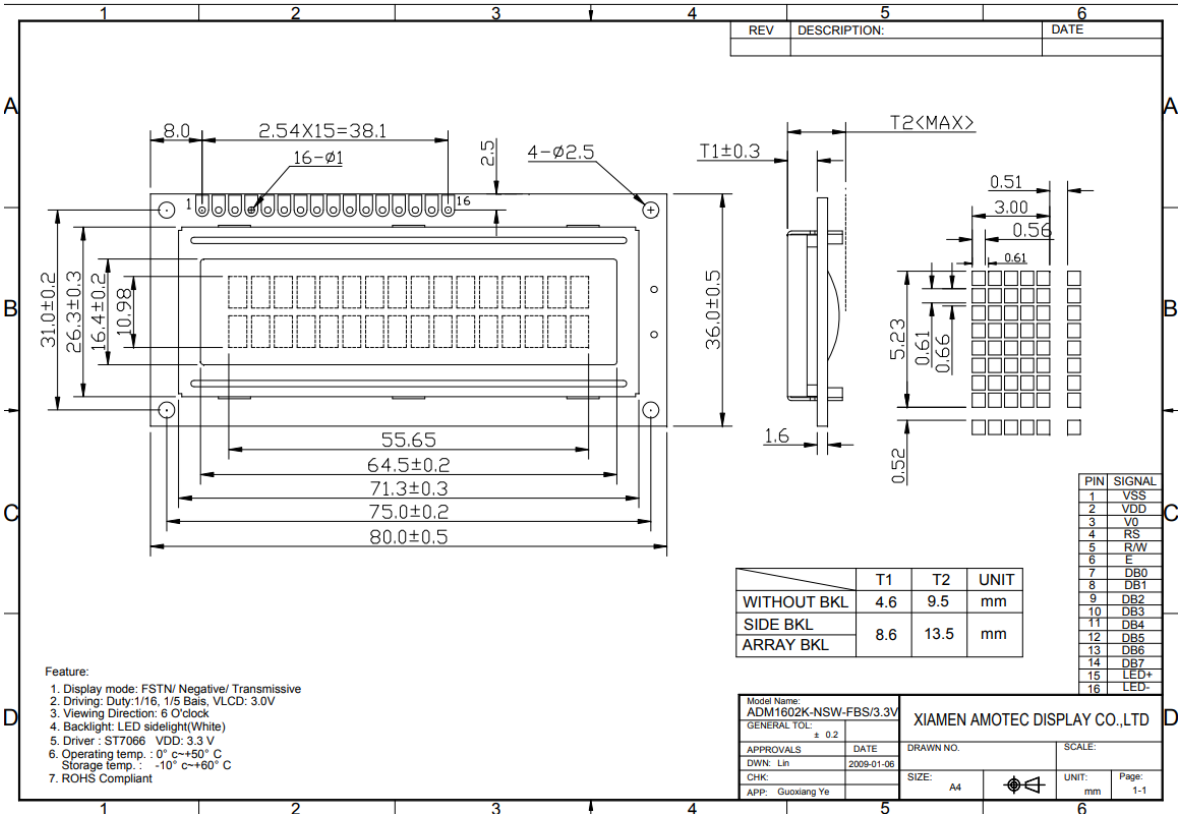

Figure 4: LCD Mechanical Drawing

LED

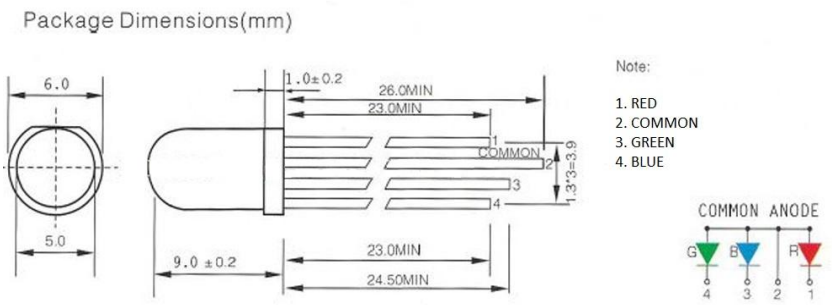

Figure 5: LED Mechanical Drawing

Button

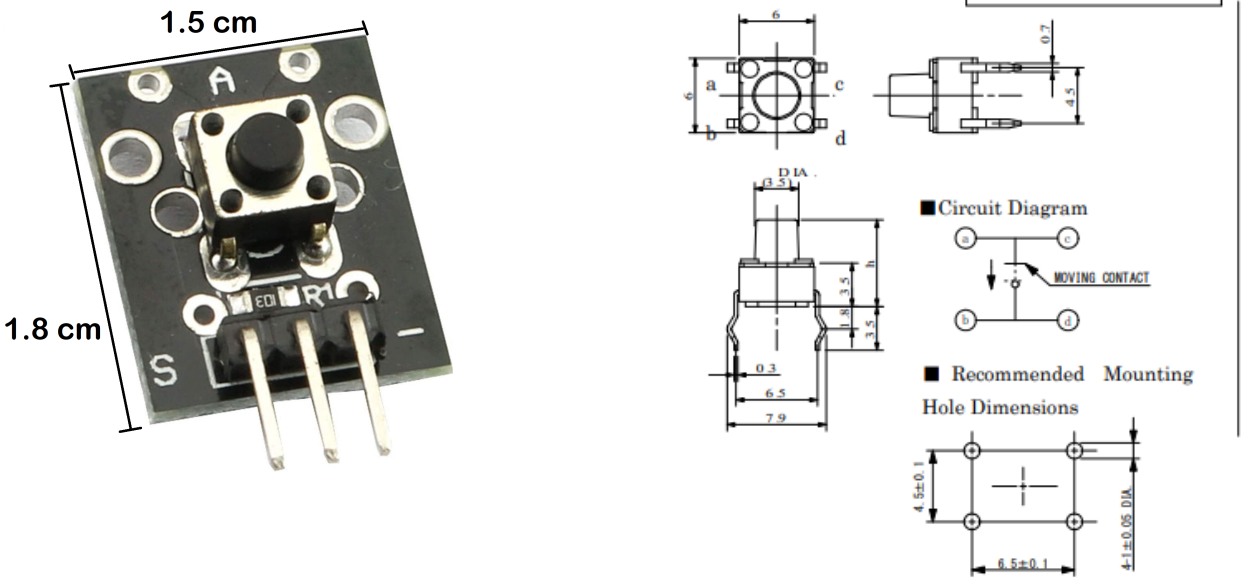

Figure 6: Button and Button Module Mechanical Drawing

Speaker

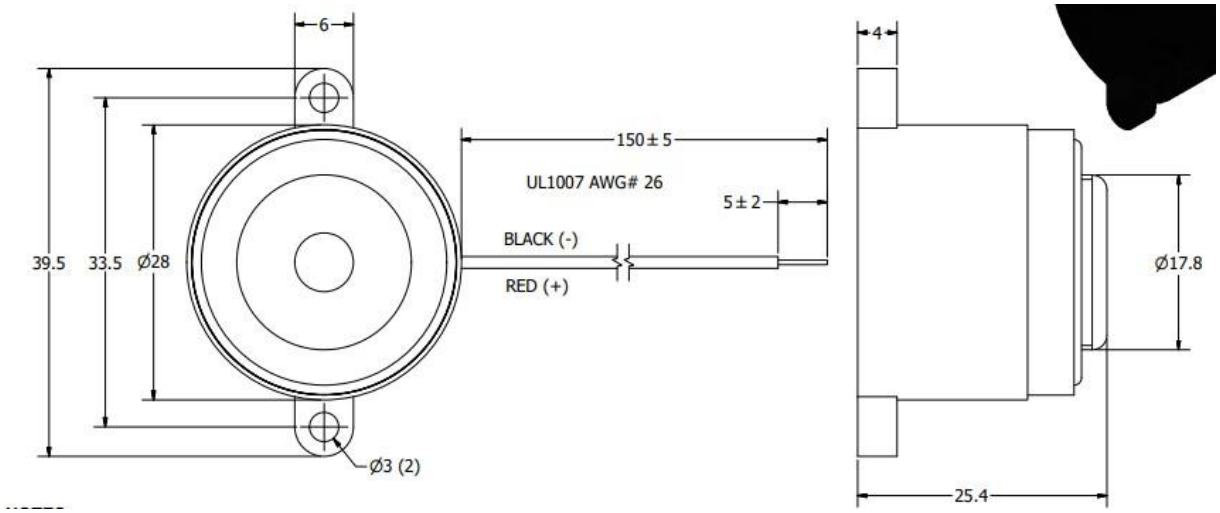

Figure 7: Speaker Mechanical Drawing

SD Card

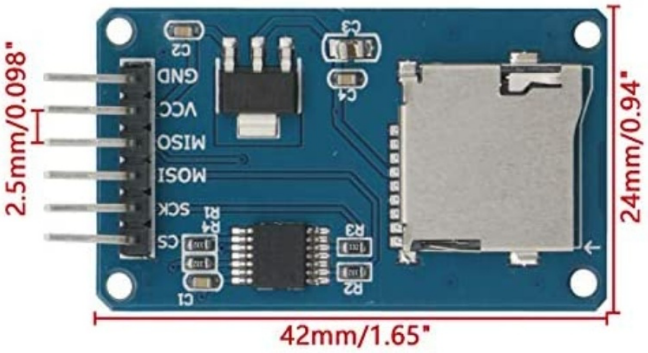

Figure 8: SD Card Module Mechanical Drawing

Battery

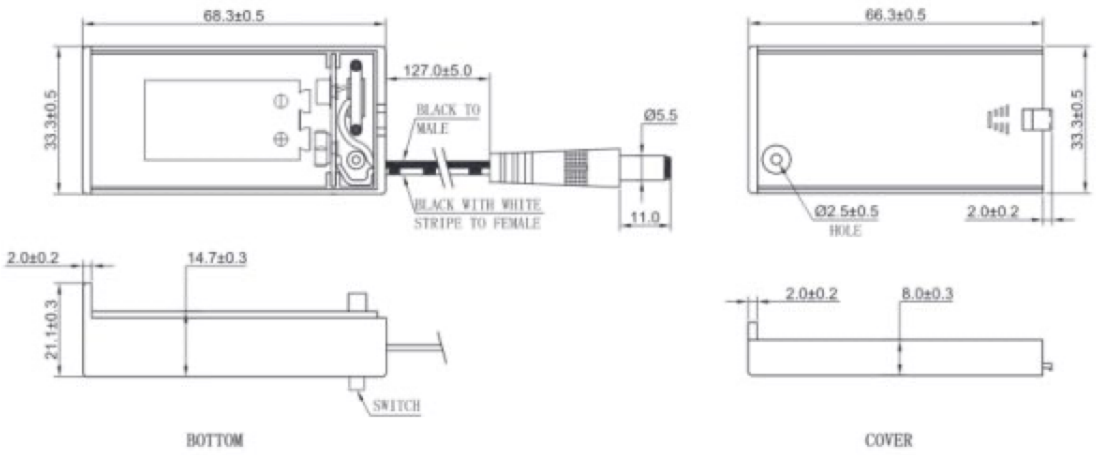

Figure 9: 9V Power Supply Mechanical Drawing

Enclosure

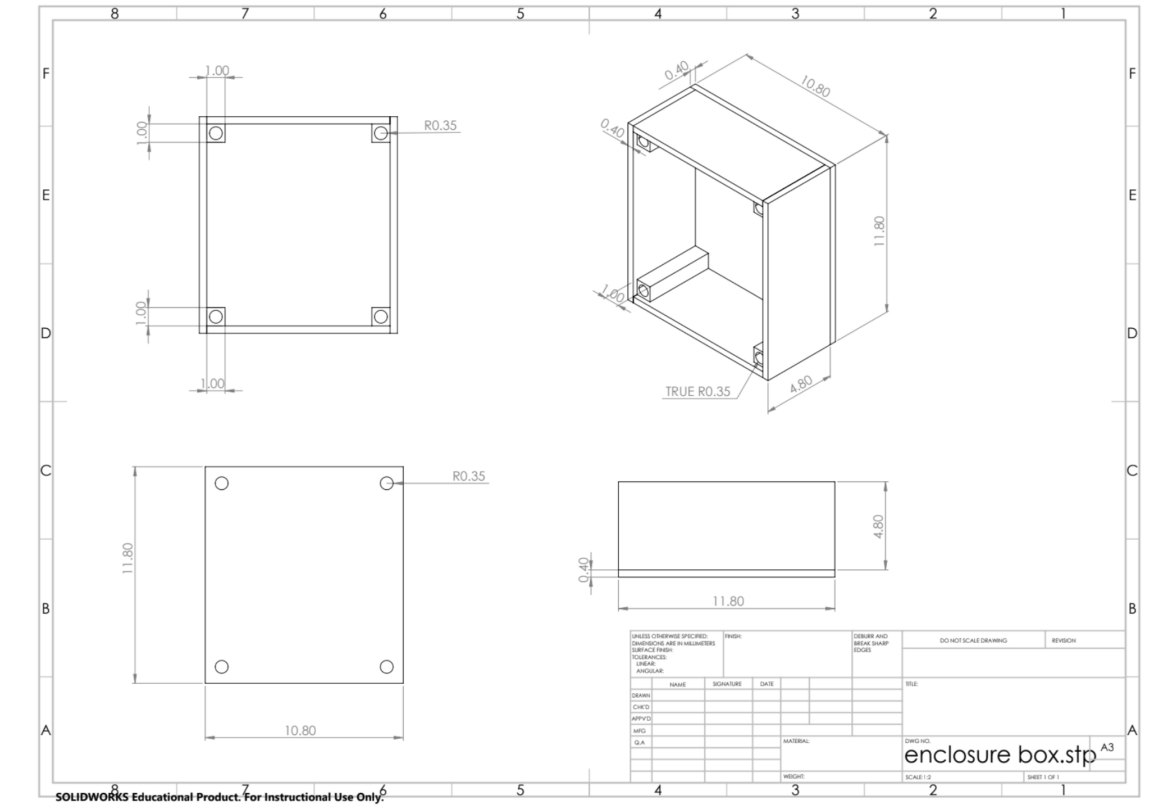

Figure 10: Mechanical drawing of the box part

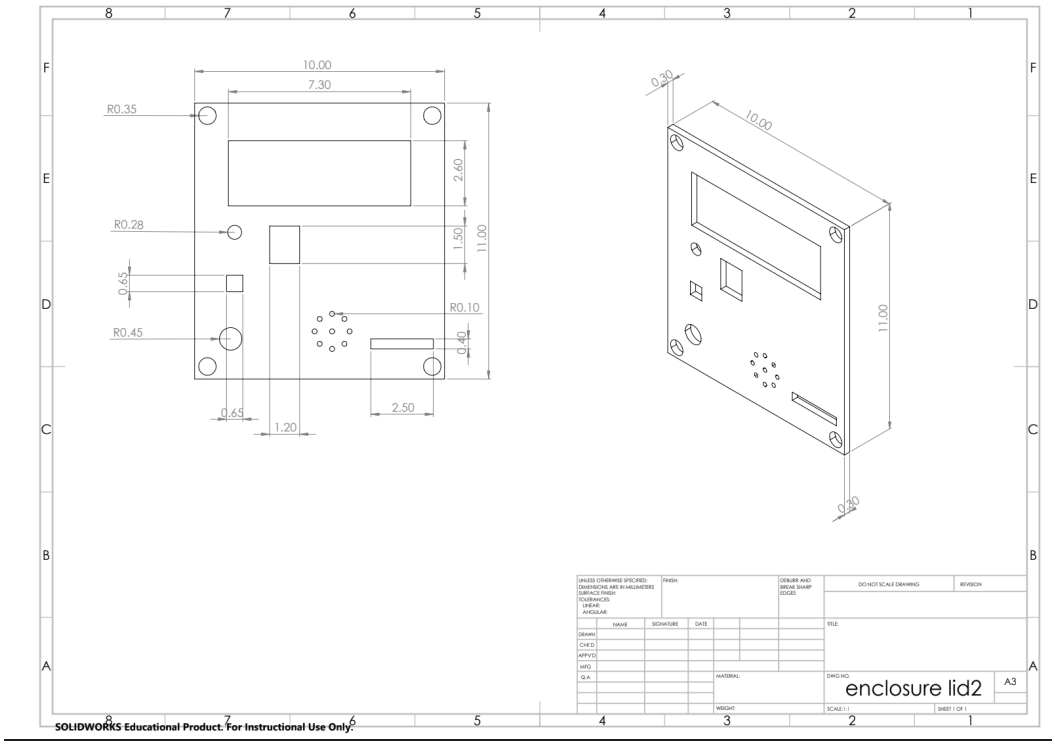

Figure 11: Mechanical drawing of the front part

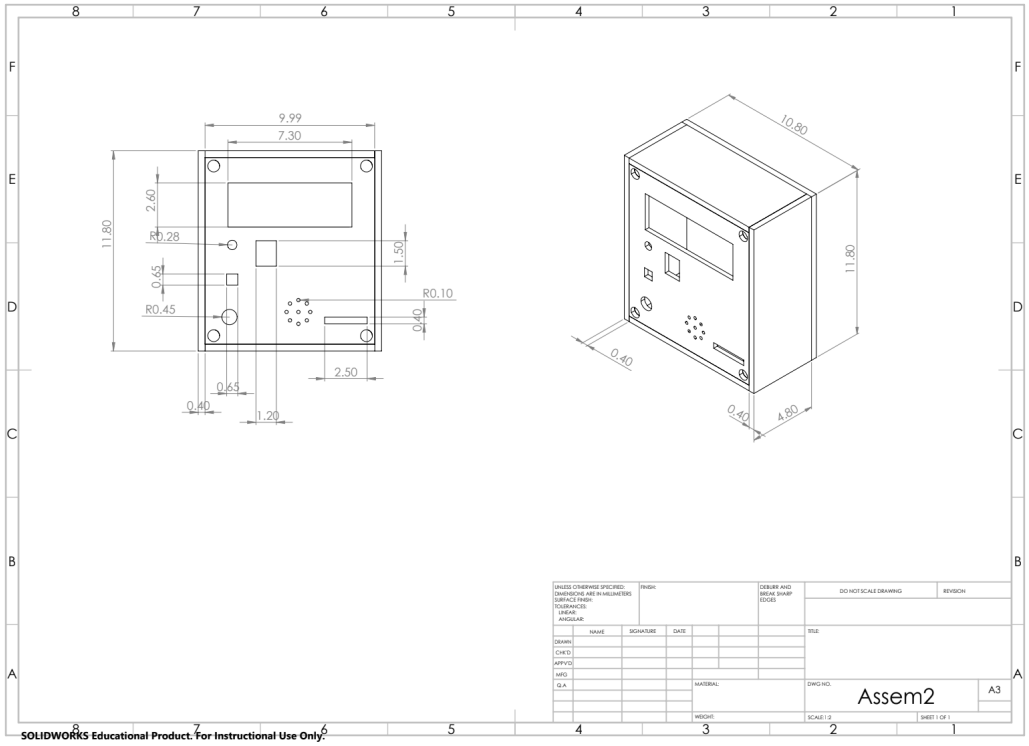

Figure 12: Mechanical drawing of the whole enclosure

Nuts and Bolts

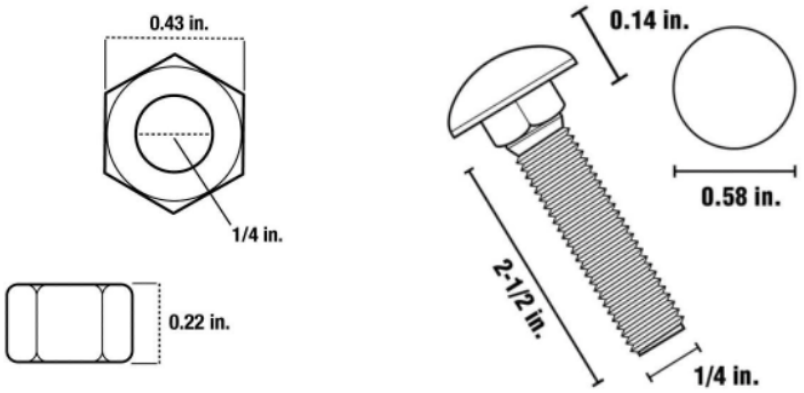

Figure13 : Dimensions of Nuts and Bolts

PCB Layers

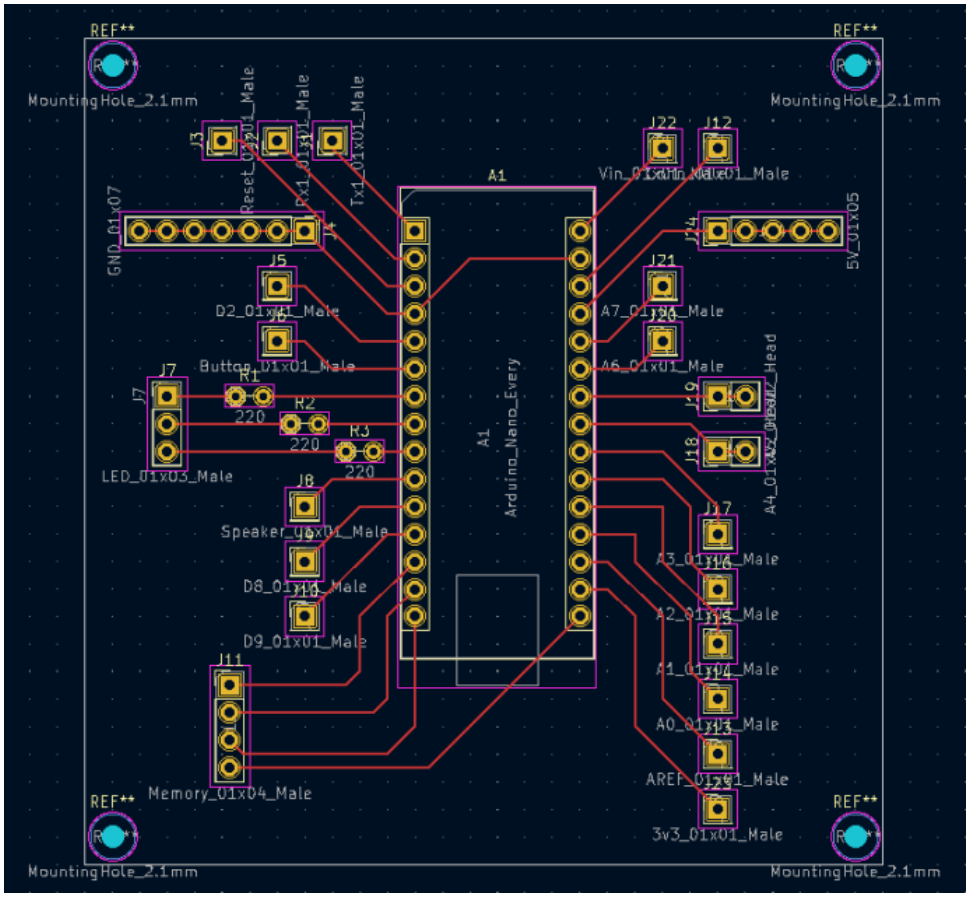

Figure14 : Detailed PCB Design

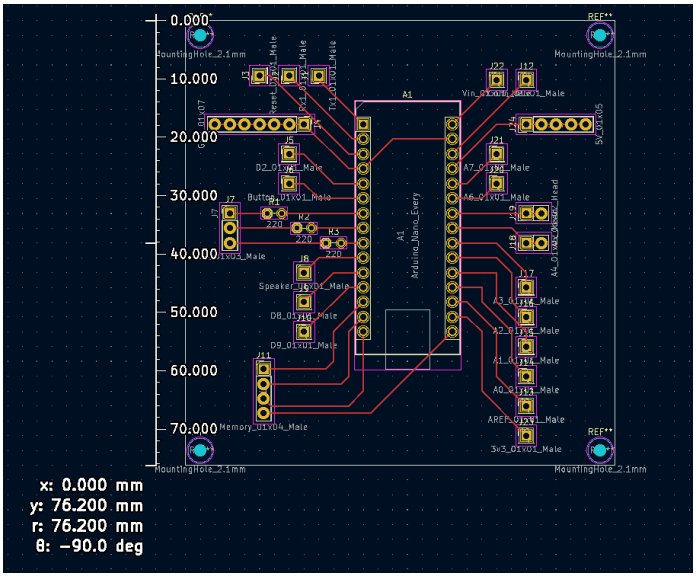

Figure15 : Length Measurement of PCB

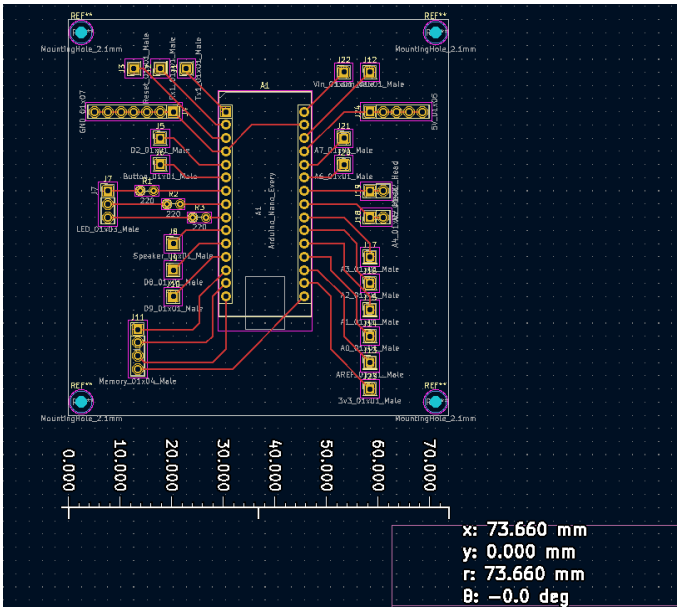

Figure16 : Width Measurement of PCB

Source Code

```
/*Libraries needed for Sensor*/
#include <Wire.h>
#include <Adafruit_MLX90614.h>
#include <LiquidCrystal_I2C.h>
#include <SPI.h>
#include <SD.h>
#include <math.h>
#include <TimeLib.h>

#define CS_PIN 10
#define SAMPLE_SIZE 60

Adafruit_MLX90614 sensor = Adafruit_MLX90614();
LiquidCrystal_I2C lcd(0x27, 16, 2);
File file;
String t_filename = "";
const int chipSelect = CS_PIN;

int button = 2;                 /*set int button to designated digital pin*/
int buttonCurrent;              /*integer of button's current state (LOW or HIGH)*/
int buttonPrevious;             /*integer of button's previous state (LOW or HIGH)*/
int printstatus = 0;            /*integer used to determine toggle status*/

int buzzer = 3;                 /*setting up buzzer and LEDS to designated digital pin*/
int red_pin = 4;
int green_pin = 6;
int blue_pin = 5;

void setup() {                  /*Initialization of LCD, sensor, memory unit*/
  lcd.init();
  lcd.backlight();
  Serial.begin(9600);
  sensor.begin();

  Serial.println("Initializing SD card...");
  if (!SD.begin()) {
    Serial.println("initialization failed!");
    return;
  }
  Serial.println("initialization done.");
  pinMode(button, INPUT);       /*initialize button as input (active LOW)*/

  pinMode(A0, INPUT);           /*set up inputs and outputs*/
  pinMode(buzzer, OUTPUT);
  pinMode(red_pin, OUTPUT);
  pinMode(green_pin, OUTPUT);
  pinMode(blue_pin, OUTPUT);
  t_filename = "log" + String(month()) + "_" + String(day()) + ".txt";    /*set up file for SD
card*/
  // read_from_file(t_filename);
}
```

```
void loop() {

  buttonPrevious = buttonCurrent;       /*save previous state of button*/
  buttonCurrent = digitalRead(button);  /*read button's current state*/

  if (buttonCurrent == HIGH && buttonPrevious == LOW) {
    if (printstatus == 0)               /*when button is pressed, reset print status*/
      printstatus = 1;                  /*printstatus = 1, so now Celsius will print*/
    else {
      printstatus = 0;                  //if printstatus = 1 (reading Celsius), set back to 0
                                       //(now will read fahrenheit)
    }
  }
  if (printstatus == 0) {               //if printstatus = 0, print Fahrenheit values (call the
                                       //sensor to print fahrenheit
    display_temperature_data_F();
  }
  else {
    display_temperature_data_C();
  }
  write_temperature_to_file(t_filename, printstatus);
  delay(100);                           /*set delay between each temperature reading*/
}
/*****************************
   FUNCTIONS

 *****************************/
/*function to get temperature in fahrenheit */
double get_temperature_F(int sample_size = SAMPLE_SIZE) {
  double temperature = 0;
  /*code for adjusting sensor readings to improve accuracy*/
  for (int i = 0; i < sample_size; ++i) temperature += sensor.readObjectTempF();
  double corrected_temperature = 53.3 * pow(temperature / 60.0, 0.125) + 1;
  return corrected_temperature;
}

/*function to get temperature in celsius*/
double get_temperature_C(int sample_size = SAMPLE_SIZE) {
  double temperature = 0;
  for (int i = 0; i < sample_size; ++i) temperature += sensor.readObjectTempF();
  double corrected_temperature = get_temperature_F();
  return (corrected_temperature - 32) * 5.0 / 9.0;
}

void read_from_file(String filename) {    /*reading from external files*/
  file = SD.open(filename, FILE_READ);
  Serial.println(filename);
  while (file.available()) {
    Serial.write(file.read());
  }
  file.close();
}
```

```
// type 0 == C, type 1 == F
/*writing temperature readings to external file*/
void write_temperature_to_file(String filename, bool type) {
  Serial.println(filename);
  file = SD.open(filename, FILE_WRITE);
  Serial.print("Writing Data\n");
  String data = "";
  if (type == 0) {
    data += String(get_temperature_F(), 2) + " F";
  } else if (type == 1) {
    data += String(get_temperature_C(), 2) + " C";
  }
  file.println(data);
  file.close();
}


/*display sensor readings in fahrenheit onto LCD screen*/
void display_temperature_data_F() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Fahrenheit: ");
  lcd.setCursor(0, 1);
  lcd.print(get_temperature_F());           /*call get_temperature_F function*/
  lcd.print(" F");
  /*if statements that tell LED to turn on specific color based on temp reading*/
  if (get_temperature_F() >= 100.5) {
    /*display red on LED when fahrenheit temp is 100.5 or higher*/
    RGB_color(255, 0, 0);
    tone(buzzer, 1000);                     /*buzz noise*/
    delay(200);
  }
  /*display green on LED when fahrenheit temp is between 95 and 100.4*/
  else if (get_temperature_F() < 100.5 && get_temperature_F() >= 95) {
    RGB_color(0, 255, 0); // Green
    noTone(buzzer);
  }
  /*display blue on LED when fahrenheit temperature is under 95*/
  else {
    RGB_color(0, 0, 255); // Blue
     noTone(buzzer);
  }
  delay(1000);
}
void display_temperature_data_C() {       /*display sensor readings in celsius onto LCD screen*/
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Celsius: ");
  lcd.setCursor(0, 1);
  lcd.print(get_temperature_C());         /*call get_temperature_C function*/
  lcd.print(" C");
  /*if statements that tell LED to turn on specific color based on temp reading*/
```

```
  if (get_temperature_C() >= 38.05) {
    /*display red on LED when celsius temp is 38.05 or higher*/
    RGB_color(255, 0, 0);
    tone(buzzer, 1000);                   /*buzz noise*/
    delay(200);
  }
  /*display green on LED when celsius temp is between 35 and 38.05*/
  else if (get_temperature_C() < 38.05 && get_temperature_C() >= 35) {
    RGB_color(0, 255, 0); // Green
    noTone(buzzer);
  }
  /*display blue on LED when celsius temperature is under 35*/
  else {
    RGB_color(0, 0, 255); // Blue
    noTone(buzzer);
  }
  delay(1000);
}
/*code to display color on LED by inputting values from parameters*/
void RGB_color(int red_value, int green_value, int blue_value)
{
  analogWrite(red_pin, red_value);
  analogWrite(green_pin, green_value);
  analogWrite(blue_pin, blue_value);
}
```

Bill of Materials

Component	Model	Manufacturer	Reference Designator	Cost
Microcontroller	Arduino Nano version 2.3 (Arduino Nano Datasheet)	Oregon State University (Vendor)	A1	$4.50
Infrared Sensor	Songhe GY-906 MLX90614 Infrared Sensor (Sensor Datasheet)	Songhe	U1	$11.88
Micro-SD Adapter	Micro SD TF Card Adapter Reader Module	HiLetgo	SDM1	$6.99
Micro-SD Card	SanDisk Ultra Lite microSDXC 128GB	SanDisk	J2	$14.48

Button	Youliang KY-004 3 Pin Button Key Tactile Switch Sensor Breakout Board Module	Youliang	BM1	$3.98
LED	NTE30115 RGB LED	NTE Electronics, Inc.	D1	$1.50
Power Supply	9V-Switch	LAMPVPATH	PS1	$2.98
LCD Screen Adapter	I2C SPI LCD Display Arduino Adapter	HiLetgo	M1	$2.98
LCD Screen	16x2 Character Liquid Crystal Display	Xiamen Amotec Display Co.	U2	$6.00
Speaker	Piezo Buzzer	PUI Audio, Inc.	BZ1	$6.68
Enclosure (3D print)	Material: White PLA	Tekbots		32.6
Bolts	carriage bolts Zinc (AGA) (1/4”x 2-1/2”)	Everbilt		1.08
Nuts	313740 Zinc hex jam nuts coarse (1/4 in-20)	Everbilt		0.64

Total Cost:	**$93.31**

Time Report

ECE 342 Non-contact Temperature Scanner

PROJECT TITLE	Non-contact Temperature Scanner	CLASS	ECE 342: Junior Design II
TEAM MEMBERS	Preston Hang, Jia Wei Cheng, Budsakol Tiangdah(Parn), Sultan Aldhaheri	DATE	1/14/22

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
1	**Project Conception and Research**					
1.1	Block Diagram Design	Sultan	1/14/22	1/14/22	0	100%
1.2	Hardware Research	Sultan, Parn	1/14/22	1/19/22	5	100%
1.3	Software Research	Preston, Jia Wei	1/14/22	1/19/22	5	100%
1.4	Research Evaluation	Preston	1/19/22	1/26/22	7	100%
2	**Project Design and Implementation**					
2.1	Parts Purchasing	Sultan	1/19/22	1/26/22	7	100%
2.3	Casing Design	Parn	1/26/22	2/2/22	6	100%
2.4	LCD Display Sofware	Preston	1/24/22	2/2/22	8	100%
2.5	Temperature Sensor Software	Jia Wei	1/28/22	2/2/22	4	100%
2.6	Power Source Research	Sultan	1/27/22	2/4/22	7	100%
2.7	External Storage Software	Jia Wei	2/2/22	2/9/22	7	100%
2.8	LED Software	Parn	2/2/22	2/9/22	7	100%
3	**Project Testing**	**First day of Testing : February 9th**				
3.1	Sensor and Storage Code Testing	Jia Wei	2/11/22	2/18/22	7	100%
3.2	LCD Testing	Preston	2/10/22	2/15/22	5	100%
3.3	Assembling	Sultan, Parn	2/16/22	2/23/22	7	100%
4	**Project Presentation**					
4.1	Poster Design	Jia Wei, Preston	2/25/22	2/28/22	3	100%
4.2	Design Report	Everyone	2/25/22	3/4/22	9	100%

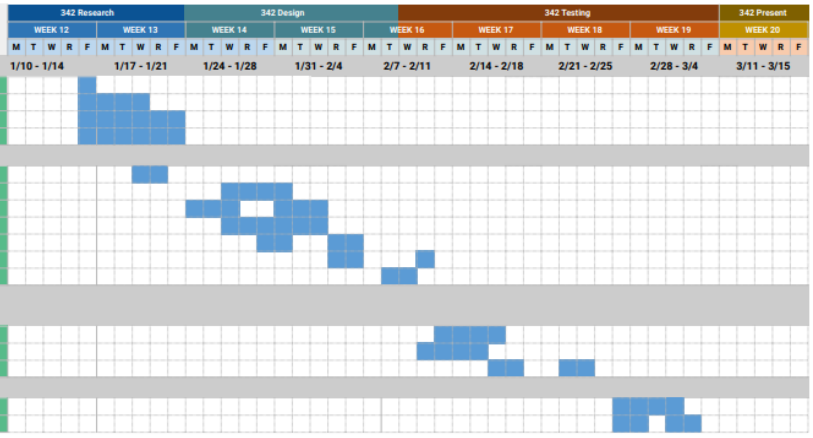

Figure : Project Timeline