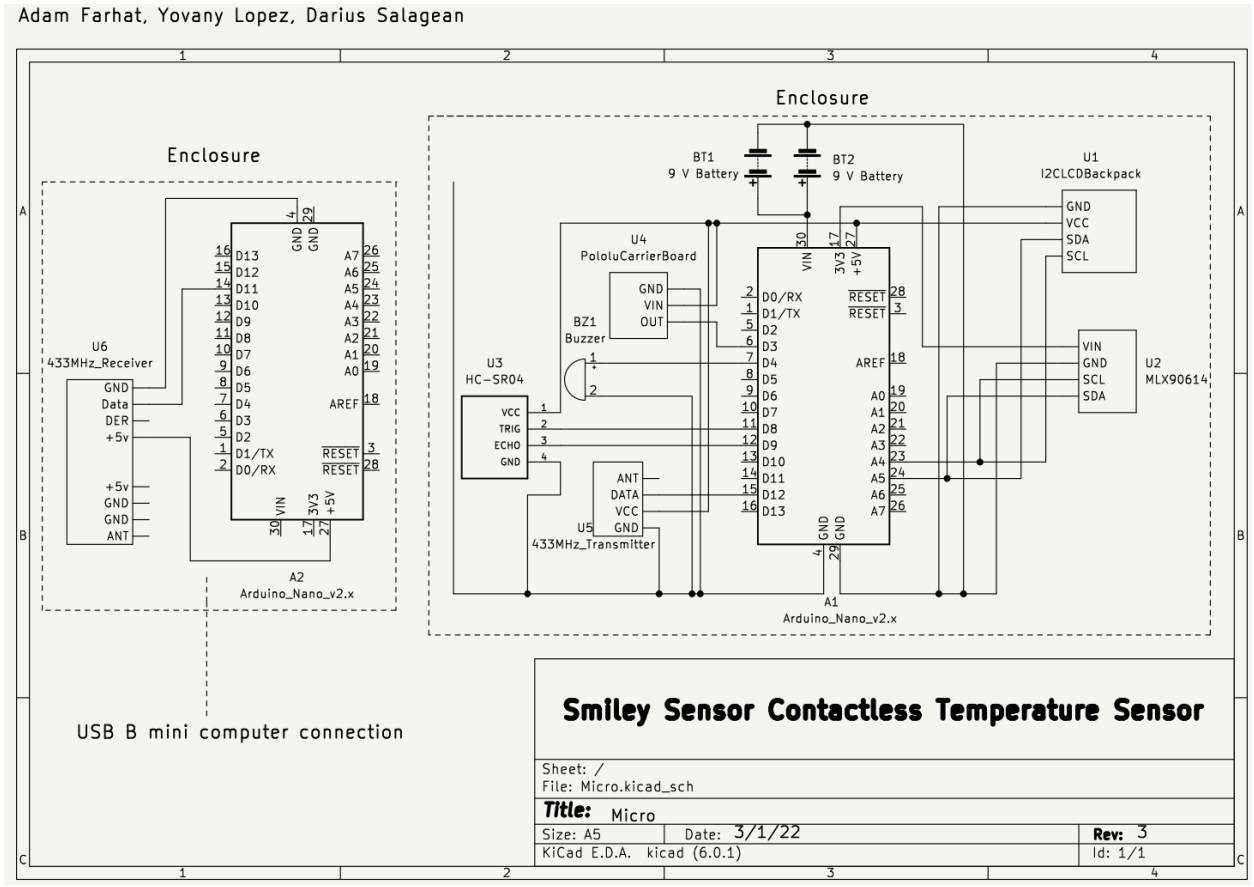
Smiley Scanner

Darius Salagean, Adam Farhat, Yovany Lopez

March 4th, 2022
ECE 342 Junior Design II
Group: Temp 7

Electrical Schematic:



Code:

Filecode (Python):

```
import serial # Import for communication through USB
import time
from tkinter import *
from tkinter.ttk import *
from threading import Thread # Import for multiple threads
from datetime import datetime # Import for getting the date and time
import os


class Controller(object):
   def __init__(self, arduino_port, baud, fileName): # Take in basic
information
       self.fileName = fileName
       self.playing = False
       self.ser = serial.Serial(arduino_port, baud)
       print("Connected to Arduino port: " + arduino_port) # Connect to Arduino
       self.file = open(self.fileName, "a")# Creates file, the 'a' means append
so it will append to the file if it is already created
       print("Created file")
       self.data = str(self.ser.readline())[2:][:-5] # Parse the information on
the serial plotter in the arduino
       dt_string2 = now.strftime(",PROGRAM START: %H:%M:%S")
       print(self.data + ",Time Taken" + dt_string2)
       self.file.write(self.data + ",Time Taken" + dt_string2 + "\n") # Write
the information into the csv
       self.file.close()

   def record(self):
       if self.playing:
           self.stop()
       else:
           self.file = open(self.fileName, "a")
           self.playing = True

       def play():
           print("Started Data Collection")
           while self.playing:
# Same information as above but in a loop for the information while its
gathered.
               self.data = str(self.ser.readline())[2:][:-5]
               now = datetime.now()
               dt_string2 = now.strftime(",%H:%M:%S")
               print(self.data + dt_string2)
               self.file.write(self.data + dt_string2 + "\n")

               time.sleep(1)
```

```
        self.thread = Thread(target=play) # We use multiple threads because we
need the user to still have access to the buttons
        self.thread.start()

    def stop(self):
        if self.playing:
            self.playing = False
            print("Data collection complete!")
            self.thread.join(0.01)
            self.file.close()
            os._exit(0)

# Everything below is for creating the UI
master = Tk()
master.geometry("250x150")

now = datetime.now()
dt_string = now.strftime("%d-%m-%Y.csv")
controller = Controller("COM5", 9600, dt_string) # Send basic information to
the CLASS

master.title("Main Menu")

label = Label(master,
              text="Transmitter/Receiver")

label.pack(pady=10)

btn = Button(master,
             text="Record Data",
             command=controller.record)

btn2 = Button(master,
              text="Stop",
              command=controller.stop)

btn.pack(pady=10)

btn2.pack(pady=10)

mainloop()
```

Main System (Arduino):

```
/*********************************************
* Temp_Final.ino
* Darius Salagean, Adam Farhat, Yovany Lopez Hernandez
* Junior Design ECE 342
```

```
* 2/11/22
*****************************************/

#include <RH_ASK.h> // Library for RF communication
#include <SPI.h> // Library for RF communication
#include <Wire.h> // Library for I2C communication
#include <LiquidCrystal_I2C.h> // Library for LCD
#include <Adafruit_MLX90614.h> //Library for MLX temp sensor
#include <NewPing.h>  //Library for Ultrasonic Sensor

//Definition statements for all devices
RH_ASK driver; //RF transmitter
Adafruit_MLX90614 mlx = Adafruit_MLX90614(); //Temp sensor
LiquidCrystal_I2C lcd = LiquidCrystal_I2C(0x27, 20, 4); //LCD Display
NewPing sonar(9, 8, 400);  //Distance sensor
#define TRIGGER_PIN 9
#define ECHO_PIN 8

//Variable initialization
float distance, tempF, tempC, tempA;
int flag = 1, trig = 0, dis = 0, c = 0, f = 1;
float cel[20], amb[20], far[20];

//First time setp-up for devices
void setup() {
   Serial.begin(9600);
   pinMode(4, OUTPUT); //Pin D4 used to drive buzzer
   mlx.begin(); //Temp sensor initialization
   lcd.init(); //LCD initialization
   lcd.backlight(); //LCD backlight initialization
   driver.init(); //RF transmitter initialization

}

//Main program. Performs all functions of the temperature
//sensor in a continuous, never ending loop.
void loop() {
   flag = 1; // Flag used to continue the while loop is set

   lcd.clear(); // LCD is set to print out a welcome message measuring distance
   lcd.setCursor(0, 0); // Set the cursor
   lcd.print("Please stand 4cm");
   lcd.setCursor(0, 1);
   lcd.print("away from sensor");
   lcd.setCursor(0, 2);
   lcd.print("Current distance:");

   //Main loop. Continues updating the distance until user
   //is within 4cm of the sensor
```

```
    while( flag == 1){

        distance = sonar.ping_cm(); //Get the distance as a float
        dis = (int)distance; //Cast the distance to an int value
        trig = digitalRead(3); //Check the motion sensor for C to F input
        lcd.setCursor(0, 3); //Print the current distance
        lcd.print(dis);
        lcd.print(" cm  ");

        if(dis <= 3){
            dis = 0;
        }else if(dis >= 200){
            dis = 200;
        }

        //Check if on F or C mode
        if(f == 1){
            lcd.print("Reading F  ");
        }
        else{
            lcd.print("Reading C  ");
        }

        if(trig == 0){ //0 means the motion sensor was tripped
            if(c == 0){ //Swap to C output mode
                f = 0;
                c = 1;
                delay(250); //Small delay to prevent rapid inputs
            }else{   //Swap to F output mode.
                f = 1;
                c = 0;
                delay(250);
            }
        }

        delay(50); //Small delay to prevent the distance from updating too fast

        //Once the user is within 3cm of the sensor it will begin reading their
temp
        if(dis == 4){
            lcd.clear(); //Reset the LCD screen for wait message
            lcd.setCursor(0, 0);
            lcd.print("Please wait while");
            lcd.setCursor(0, 1);
            lcd.print("your temperature is");
            lcd.setCursor(0, 2);
            lcd.print("being read");

            tempF = 0; //Rest the temperature values so they can be reused
```

```
            tempC = 0;
            tempA = 0;   //Delay so that the temp starts reading after the user
            delay(1000); //acknowledges they are close enough and stops moving.

            //Take 20 readings of user temp and ambient temp in C
            //Temp in C was chosen since C is more widely used.
            for (int i = 0; i < 20; i++) {
                cel[i] = mlx.readObjectTempC();
                amb[i] = mlx.readAmbientTempC();
                delay(100); //Small delay between readings to get 2 seconds of
data
            }

            //Compute the average of the readings
            for (int i = 0; i < 20; i++){
                tempC = tempC + cel[i];
                tempA = tempA + amb[i];
            }
            tempC = tempC / 20.00;
            tempC = tempC + 4.65;
            tempA = tempA / 20.00;
            Serial.println(tempC);
            Serial.println(tempA);

            //Using modifed formula found at
https://www.medrxiv.org/content/10.1101/2020.12.04.20243923v1.full.pdf
            //and recorded data to estimate body temperature from forehead
temperature
            tempC = 2.85 + pow(((pow(tempC, 4)-(0.02*(pow(tempA, 4))))/(0.98)),
(0.25));
            tempF = (tempC*(9.000/5.000)) + 32.000; //Convert from C to F

            lcd.clear(); //Clear the LCD for temperature readout
            lcd.setCursor(0, 0); // Set the cursor
            lcd.print("Your temperature is:");
            lcd.setCursor(0, 1);
            //Display F or C temperature depending on user choice at start
            if(f == 1){
                lcd.print(tempF);
                lcd.print(" F ");
            }else{
                lcd.print(tempC);
                lcd.print(" C ");
            }

            digitalWrite(4, HIGH); //Signal to the user their temp ws taken
            delay(250);
            digitalWrite(4, LOW);
            //Check if a fever was recorded
```

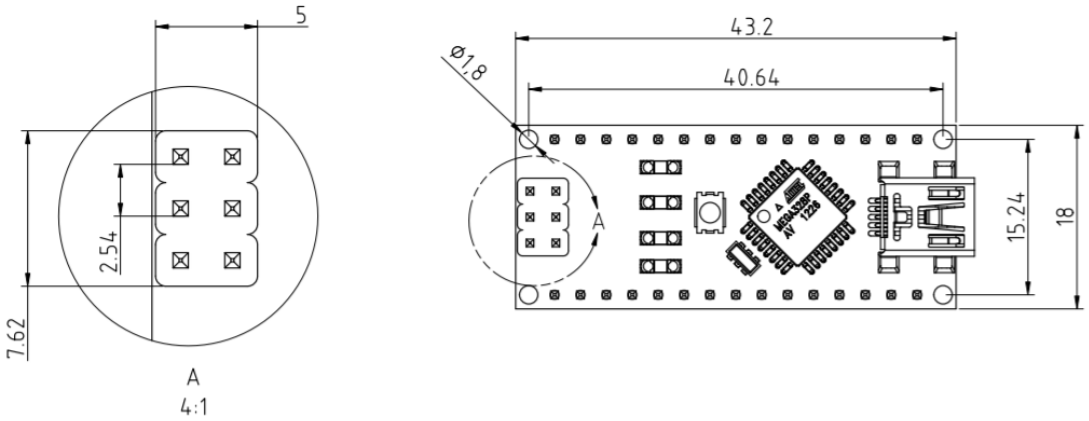
```
            if(tempF >= 100.40 || tempC >= 38.00 ){
                 lcd.setCursor(0, 2); //Let the user know on the screen
                lcd.print("You may have a fever");
                //Make the buzzer give the user an audio que.
                for(int i = 0; i <5; i++){
                     digitalWrite(4, HIGH);
                    delay(250);
                    digitalWrite(4, LOW);
                    delay(250);
                }
            }
            //Let the user know data is being saved
            lcd.setCursor(0, 3);
            lcd.print("Data is being saved");
            //Transmit the data to the receiver
            driver.send((uint8_t *)&tempC, 4); //two bytes on AVR Arduino
            driver.waitPacketSent(); //Wait for the data to be sent before
moving on

            delay(5000); //Small delay to allow the user to recognize that the
reading went through

            c = 0;
            f = 1;
            flag = 0;
        }
    }
}
```

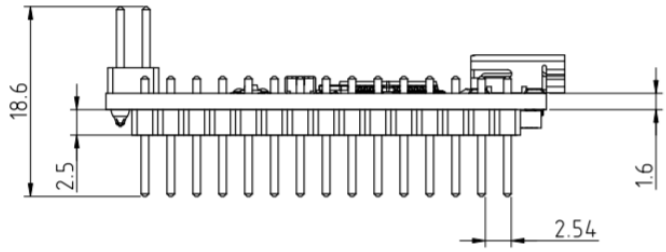
Receiver (Arduino):

```
#include <RH_ASK.h>
#include <TimeLib.h>
#include <SPI.h> // Not actually used but needed to compile

RH_ASK driver;

void setup()
{
   Serial.begin(9600);  // Debugging only
   if (!driver.init()){
        Serial.println("init failed");
   }
   //Serial.print("Date of Recording: ");
   //Serial.println(F(__DATE__));
   //Serial.print("Time of Recording: ");
   //Serial.println(F(__TIME__));
}

   //gloabls
```

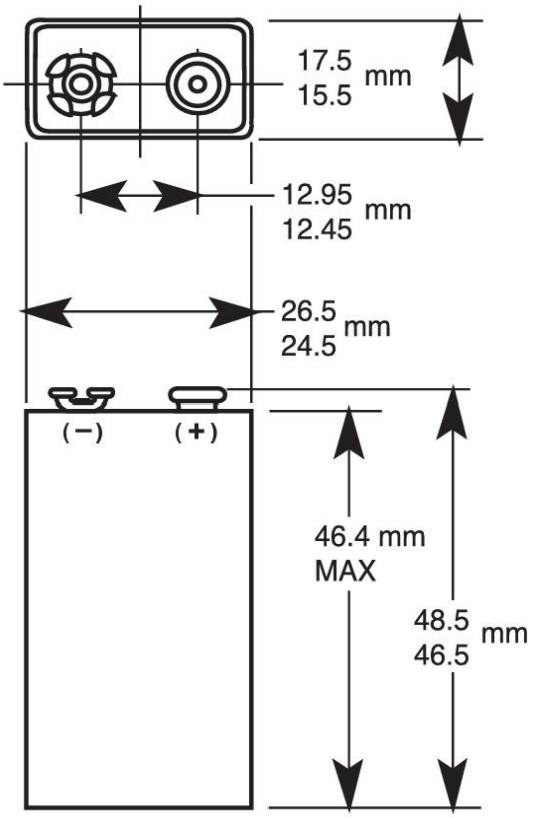
```
   String dataLabel1 = "Fahrenheit";
   String dataLabel2 = "Celsius";
   float fahrenheit;
   float celsius;
   bool label = true;

void loop()
{
   while(label){ //runs once
       Serial.print(dataLabel2);
       Serial.print(",");
       Serial.println(dataLabel1);
       label=false;
 }

   uint8_t buflen = 4; // max number of bytes allowed in message
   if (driver.recv((uint8_t *) &celsius, &buflen)) // Non-blocking
   {
     if (buflen == 4)  //got correct message size?
    {
     // Message with a good checksum received, dump it.
     //Serial.print("TEMP: ");
     Serial.print(celsius);
     Serial.print(" C");

     Serial.print(",");
     fahrenheit = (celsius *(9.000/5.000)) + 32.000;

     Serial.print(fahrenheit);
     Serial.println(" F");

     delay(1000); // ms
    }
   }
}
```

Mechanical Drawings:

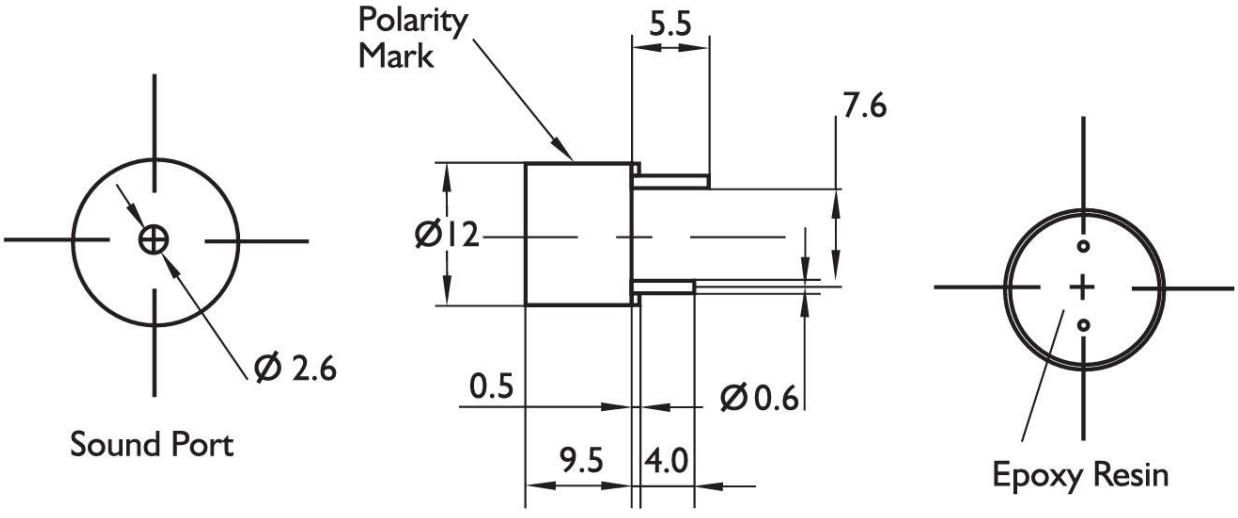




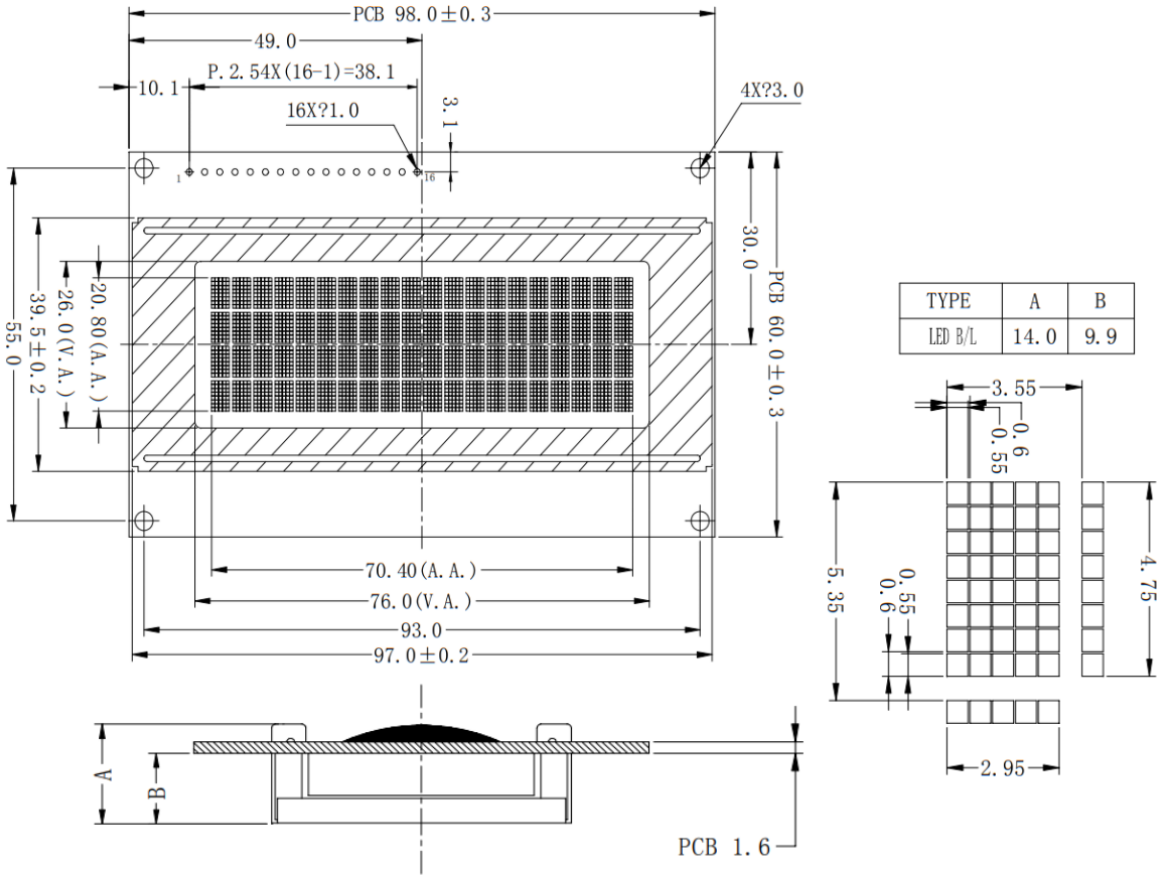
Arduino Nano



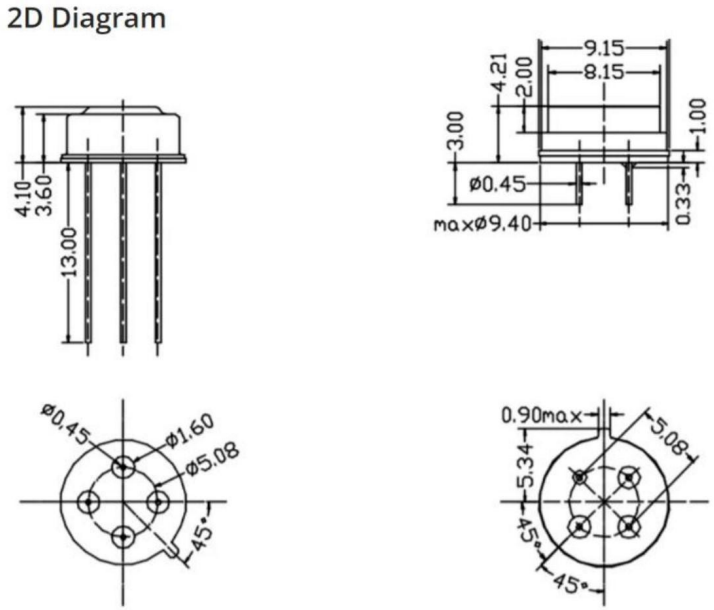
9v Battery



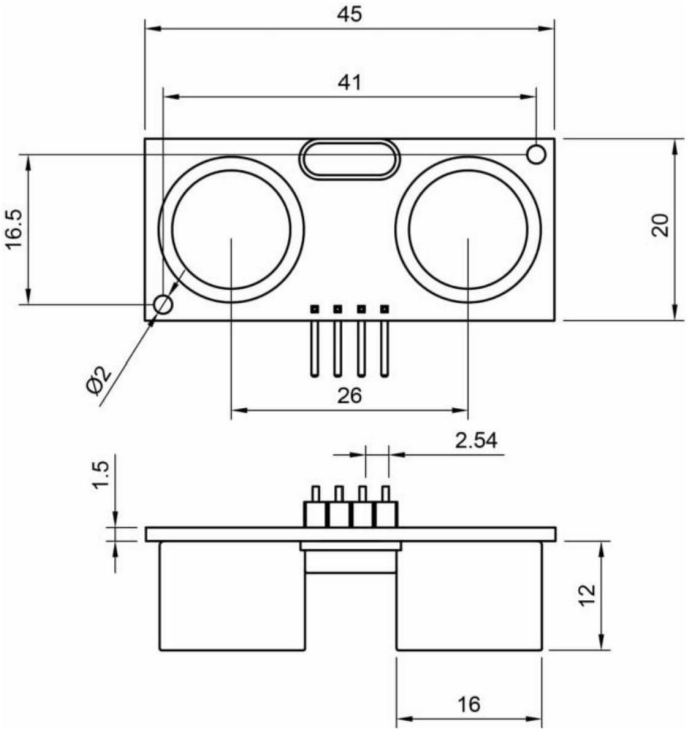
ABI-009-RC Buzzer



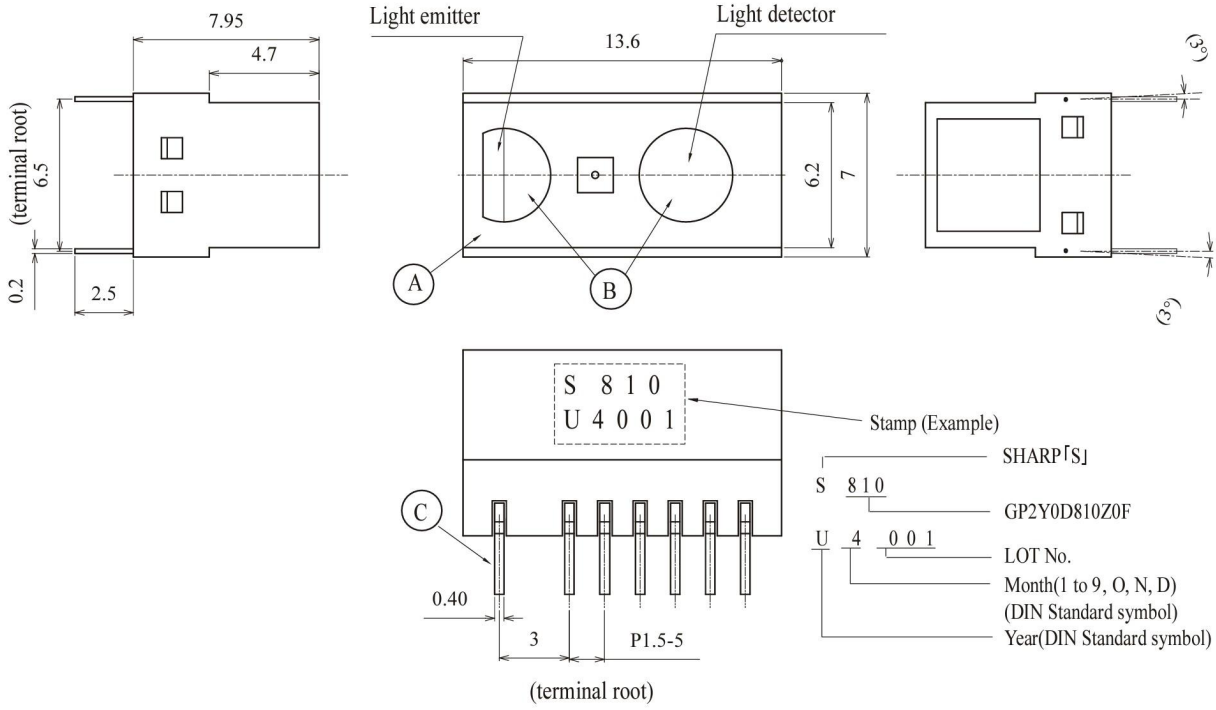
TC2004A-01 LCD Display



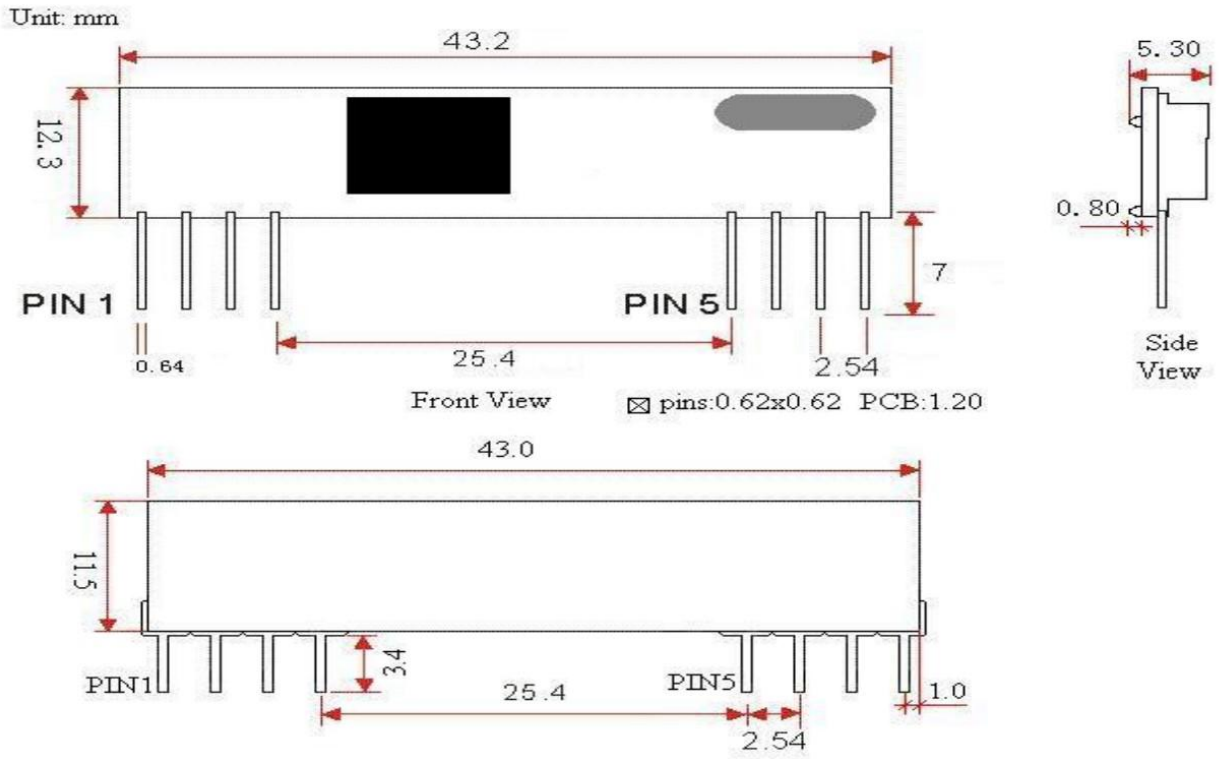
MLX90614 Temperature Sensor



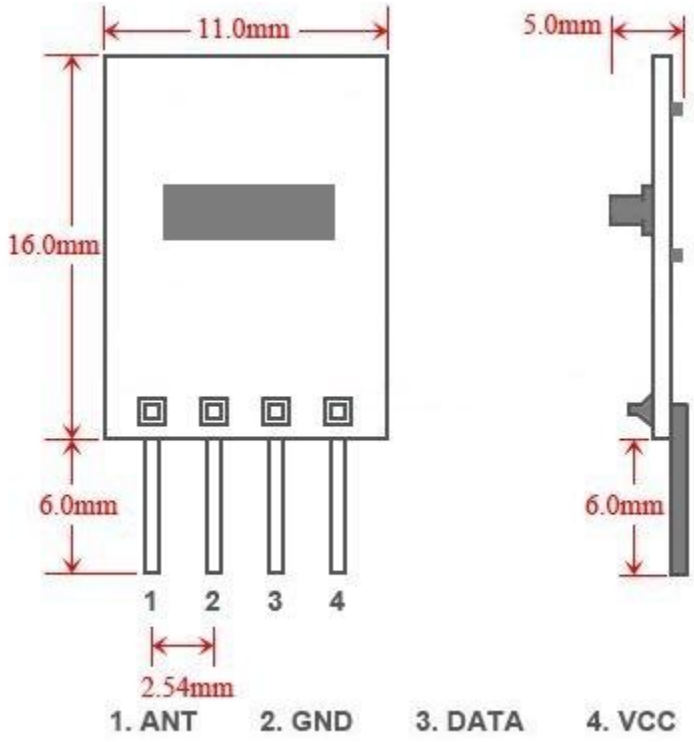
HC-SR04 Ultrasonic Sensor



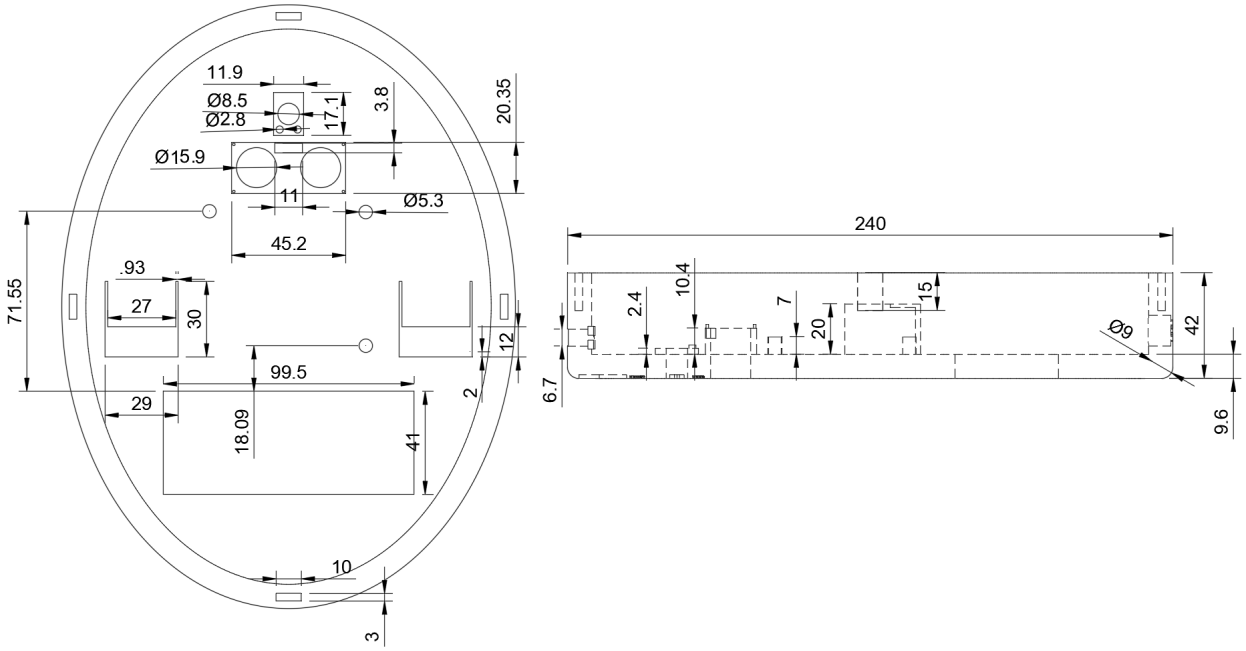
B01MRY3OK5 Motion Sensor



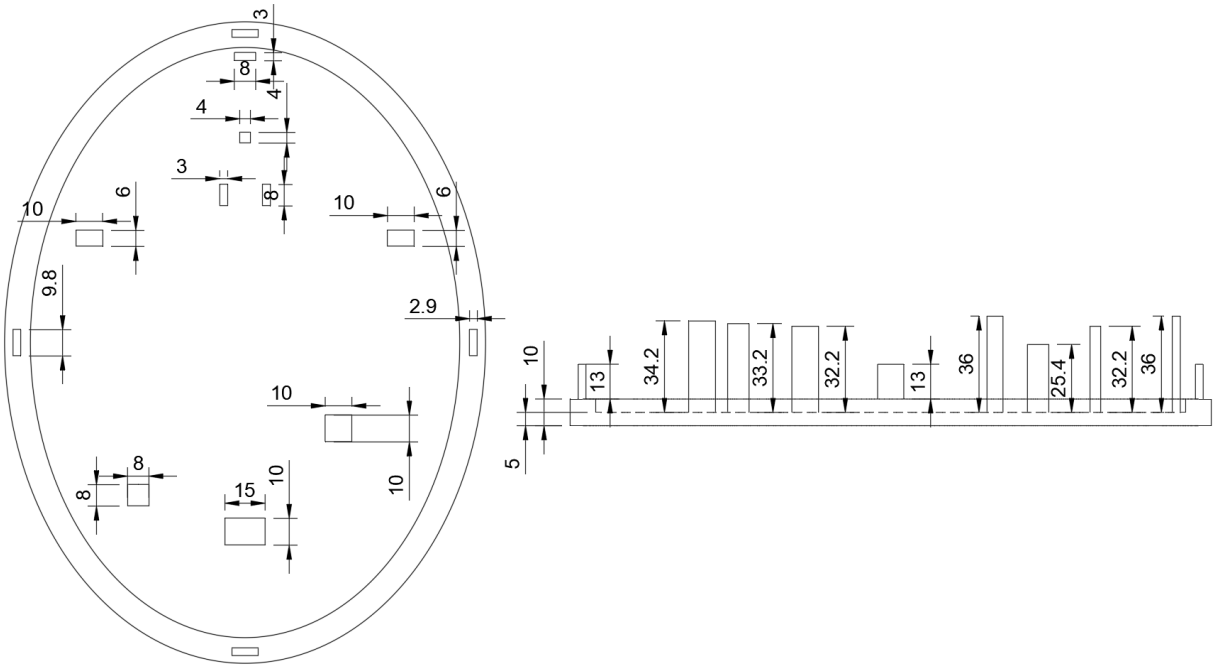
RBX96 Receiver



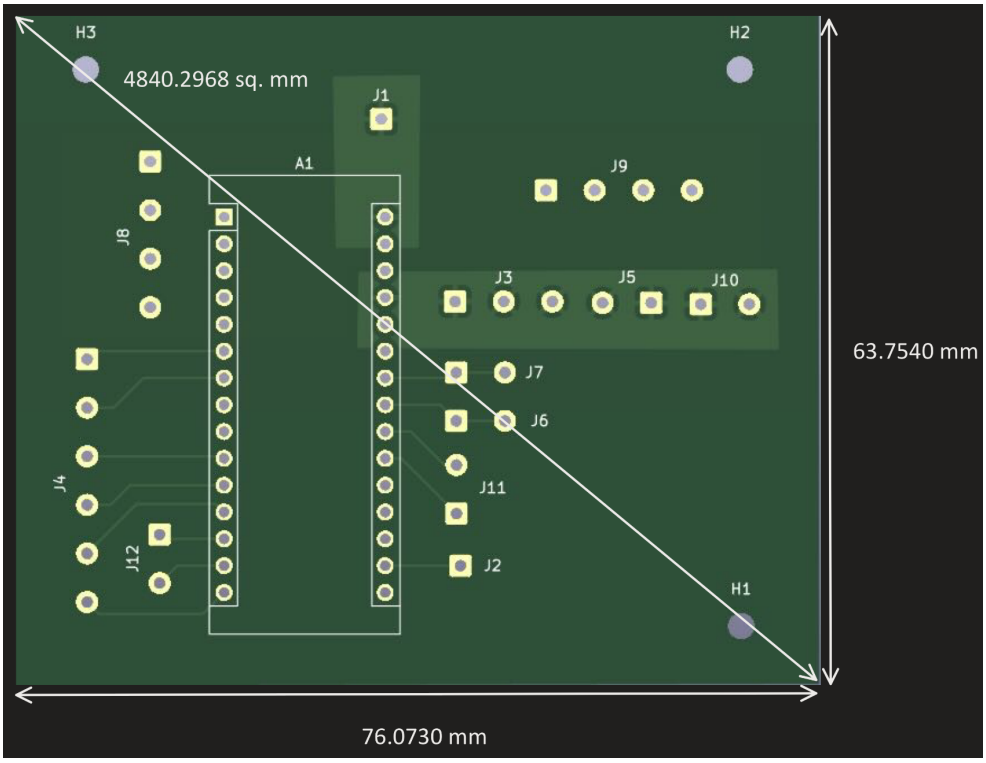
433 MHz Transmitter



Enclosure



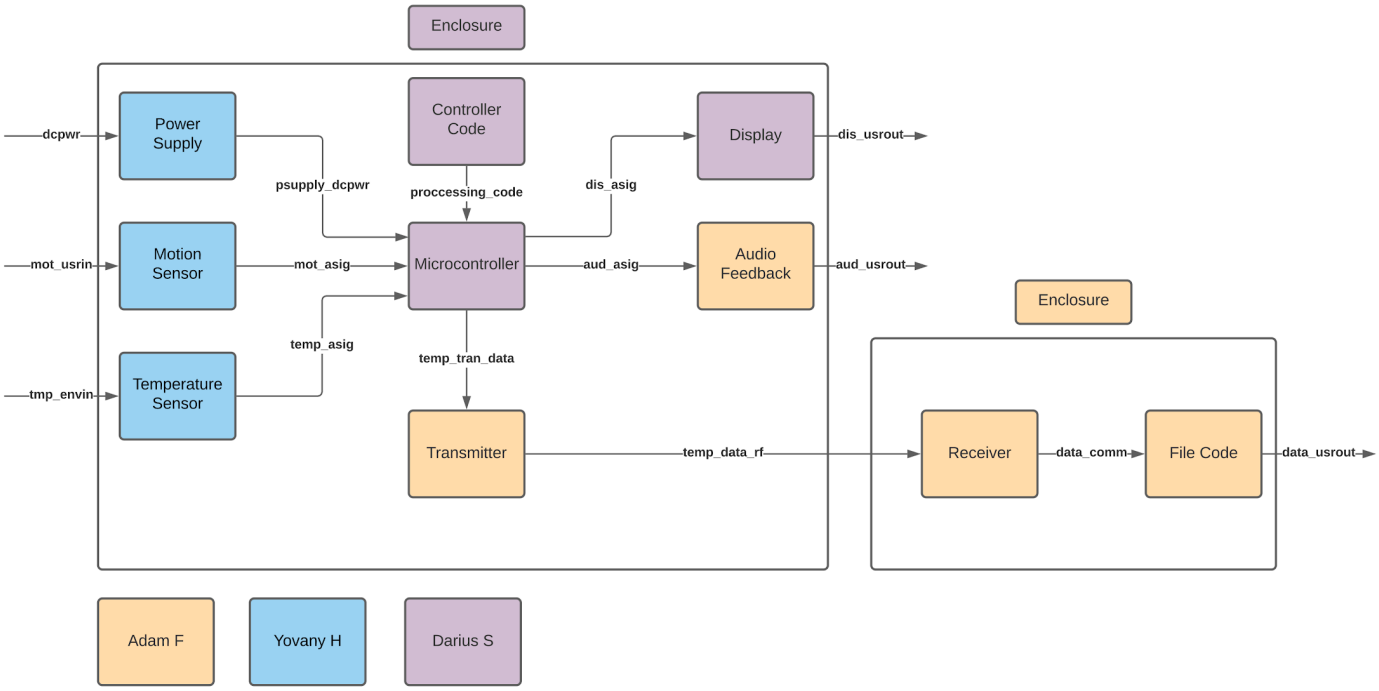
Enclosure Cover



Custom PCB Dimensions

Custom PCB Thickness of Board

Top Level Diagram:

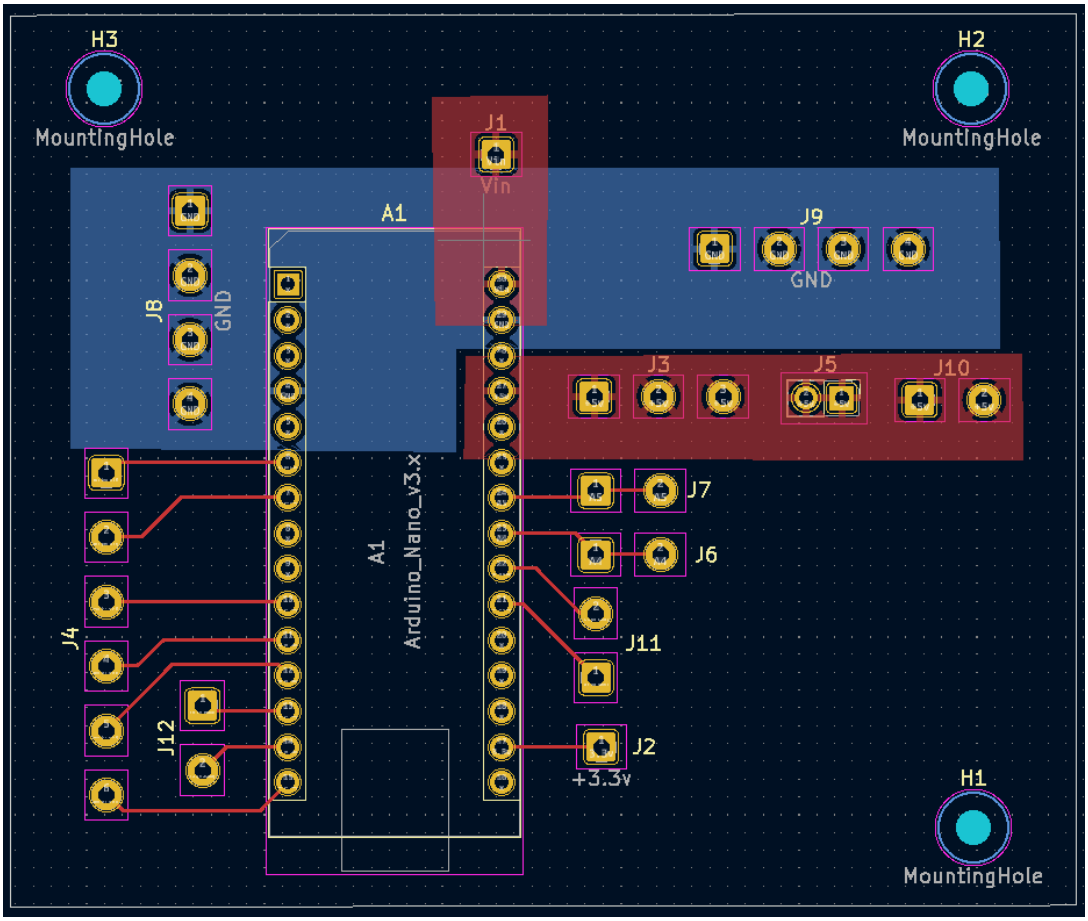


Interface and properties:

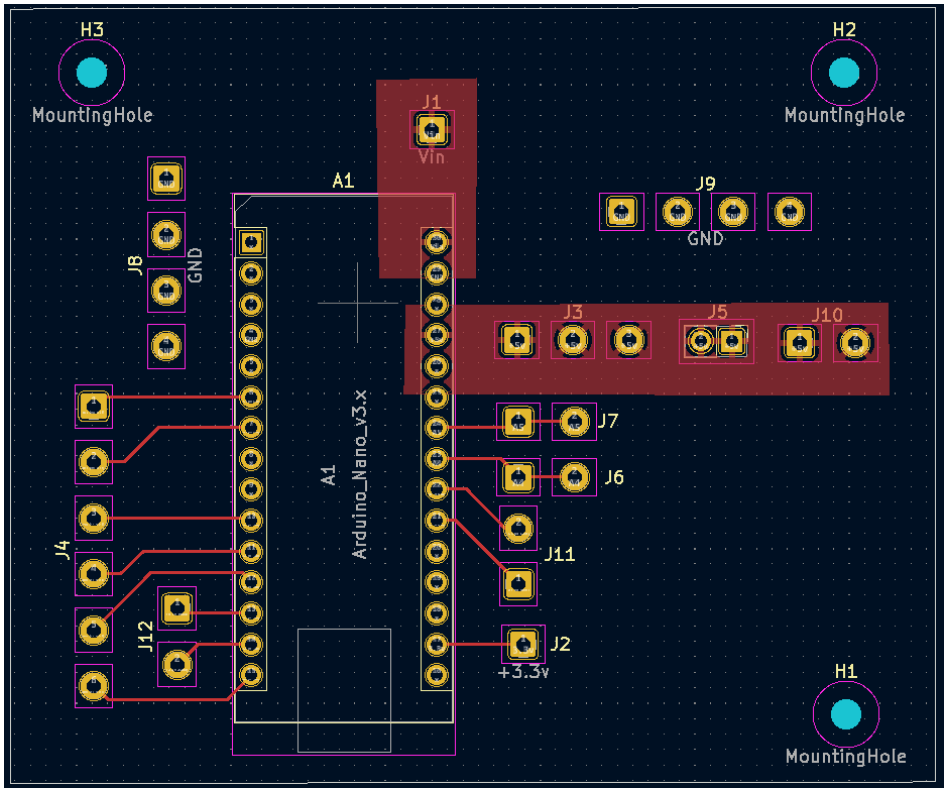
Interface Name	Interface Type	Specifics
dcpwr	DC Power	● V_{max} = 9.0 V ● I_{max} = 550 mA ● I_{min} = 120 mA
psupply_dcpwr	DC Power	● V_{max} = 12.0 V ● V_{min} = 7.0 V ● I_{max} = 250 mA ● I_{min} = 120 mA
mot_usrin	User Input	● Use of ultrasonic sound to detect distance. ● Detects how close the user is within a range of 0 - 200 cm.
mot_asig	Analog Signal	● V_{range} = 0 - 5 V ● Distance = (ECHO high level time X ultrasonic velocity (Speed of Sound in air 340m/sec) / 2 ● $I_{nominal}$ = 15mA
tmp_envin	Environmental Input	● Use of infrared waves to detect temperature. ● Detects human body temperature without contact. ● Detects ambient temperature.
tmp_asig	Analog Signal	● V_{range} = 3.3v - 5 V ● Temperature: 0V = 0ºC / 5V = 40ºC (1.25% Tolerance) ● Temperature: 0V = 32ºF / 5V = 104ºF (1% Tolerance) ● $I_{nominal}$ = 15mA
processing_code	Code	● Transmits data to a receiver. ● Activates buzzer when temperature is over 100.4ºF/38ºC. ● Measures temperature of user through temperature sensor. ● Mesures motion of user through motion sensor. ● Determines internal body temperature. ● Size less then 20 kb
filemake_code	Code	● Output to notepad file on desktop of

		computer ● Receives data from internal microcontroller ● Size less than 10 kb
dis_asig	Analog Signal	● V_{max} = 11 V ● V_{min} = 5 V ● I_{max} = 600uA ● I_{min} = 350uA
dis_usrout	User Output	● Output 20 characters in 4 rows. ● Display current temperature. ● Display fever alert if over 100.4ºF/38ºC. ● Display a welcome message instructing the user on how close they are to the device.
aud_asig	Analog Signal	● V_{max} = 8.0 V ● V_{min} = 4.0 V ● I_{max} = <25 mA
aud_usrout	User Output	● dB_{range} 55 - 65 dB ● $Freq_{range}$ 400 - 600 Hz ● Activates when temperature is over 100.4ºF/38ºC. ● Plays 5, .25 second beeps.
temp_tran_data	Data	● V_{max} = 12.0 V ● V_{min} = 3.0 V ● I_{max} = 95 mA ● I_{min} = 2 mA ● Float value containing the temperature reading in Celcius.
temp_data_rf	Data	● 433 MHz signal. ● 0 - 30ft range.
data_comm	Data	● USB connection from arduino nano to computer. ● File code connects to USB to draw received information from the transmitter.
data_usrout	User Output	● File named the current date: (xx-xx-xxxx) ● File contains: Temperature in fahrenheit/celsius and time temperature was taken.

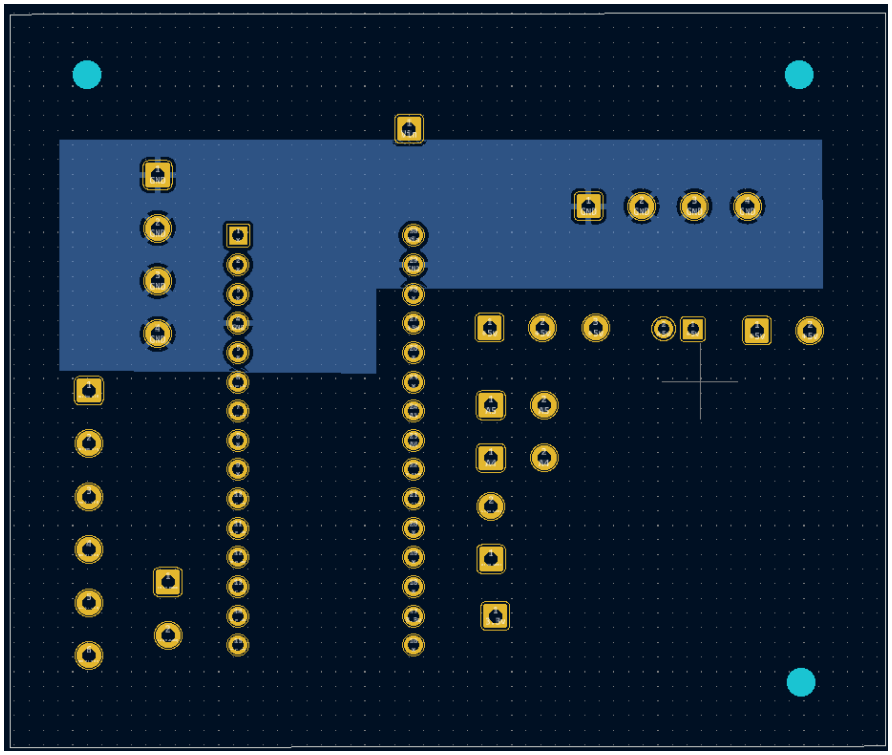
PCB Layers:



PCB With All Layers



Top Layer PCB



Bottom Layer PCB

Bill of Materials:

Part	Block Identifier	Product Name	Part Number	Quantity	Price	Datasheet	Purchase link
Microcontroller	A1 & A2	Arduino Nano	A000005	2	$49.98 (24.99)x2	Datasheet	Amazon
Battery	BT1 & BT2	Duracell CopperTop 9V Alkaline Batteries	MN1604	2	$8.54	Datasheet	Amazon
Buzzer	BZ1	Cylewet 5V Active Buzzer	CYT1036	1	$6.98 (Pack of 10)	Datasheet	Amazon
Display	U1	GeeekPi 20x4 LCD Module with I2C Interface Adapter Blue Backlight	LCD2004	1	$9.99	LCD I2C Adapter	Amazon
Temperature Sensor	U2	Songhe Non Contact IR Infrared Temperature Sensor Module	GY-906 MLX90614ESF-BAA	1	$11.88	Datasheet	Amazon
Ultrasonic Sensor	U3	ELEGOO HC-SR04 Ultrasonic Module Distance Sensor	EL-SM-001	1	$9.99 (Pack of 5)	Datasheet	Amazon
Motion Sensor	U4	Pololu Carrier with Sharp GP2Y0D810Z0F Digital Distance Sensor	POLOLU-1134	1	$6.99	Datasheet	Amazon
Transmitter and Receiver	U5 & U6	433MHz Superheterodyne RF Link Transmitter and Receiver Kits 3400	619636107597	1	$9.99	Datasheet	Amazon

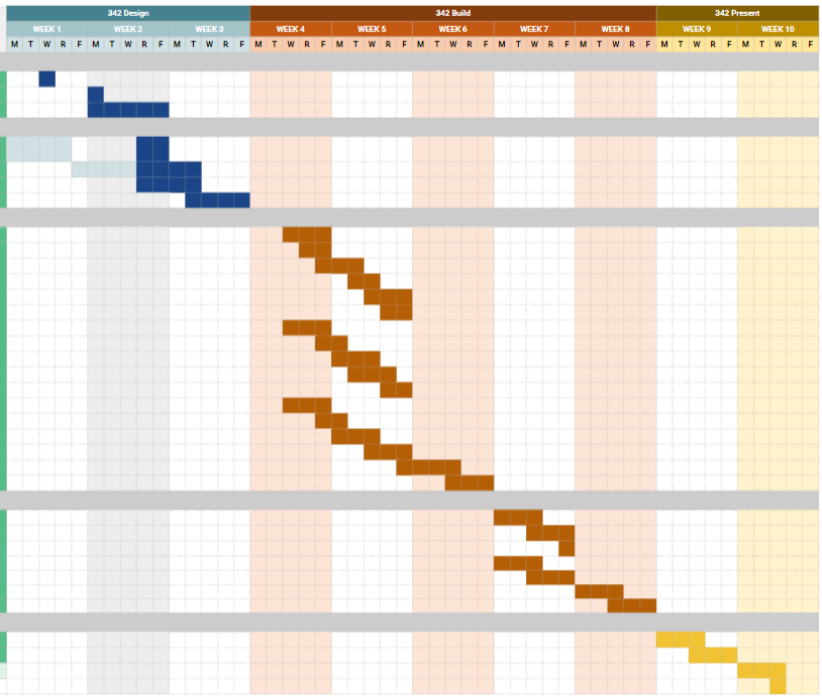
Link to full spreadsheet: Bill of Materials

Time Report:

342 GANTT CHART

PROJECT TITLE	Contactless Temperature Sensor	COMPANY NAME	ECE 342 Junior Design II
PROJECT MANAGER	Adam Farhat, Yovany Lopez, Darius Salagean	DATE	3/3/22

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
1	**Project Initiation**					
1.1	Project Charter	Given	1/6/22	1/6/22	1	100%
1.2	Requirements	Everyone	1/10/22	1/10/22	1	100%
1.3	Design Research	Everyone	1/10/22	1/14/22	5	100%
2	**Block Definition**					
2.1	Block Diagram and Interface Definitions Draft	Everyone	1/13/22	1/14/22	4	100%
2.2	Part list	Everyone	1/13/22	1/18/22	3	100%
2.3	Budget	Everyone	1/13/22	1/18/22	6	100%
2.4	Ordering of Parts	Adam F	1/18/22	1/21/22	7	100%
3	**Initial Prototyping**					
3.1	Temperature Sensor Bulding	Yovany H	1/26/22	1/28/22	3	100%
3.1.1	Temperature Sensor Testing	Yovany H	1/27/22	1/28/22	2	100%
3.2	Motion Sensor Building	Yovany H	1/28/22	2/1/22	5	100%
3.2.1	Motion Sensor Testing	Yovany H	2/1/22	2/2/22	2	100%
3.3	Power Supply Bulding	Yovany H	2/2/22	2/4/22	3	100%
3.3.1	Power Supply Testing	Yovany H	2/3/22	2/4/22	2	100%
3.4	Receiver/Transmitter Bulding	Adam F	1/26/22	1/28/22	3	100%
3.4.1	Receiver/Transmitter Code Creation	Adam F	1/26/22	1/31/22	4	100%
3.4.2	Receiver/Transmitter Testing	Adam F	1/31/22	2/2/22	3	100%
3.5	Audio Feedback Bulding	Adam F	2/1/22	2/4/22	3	100%
3.5.1	Audio Feedback Testing	Adam F	2/3/22	2/4/22	2	100%
3.6	Display Bulding	Darius S	1/26/22	1/28/22	3	100%
3.6.1	Display Testing	Darius S	1/28/22	1/31/22	4	100%
3.7	Microprocessor Code Creation	Darius S	1/31/22	2/2/22	3	100%
3.7.1	Microprocessor Testing	Darius S	2/2/22	2/3/22	3	100%
3.8	Bread Board Prototype Creation	Everyone	2/3/22	2/9/22	6	100%
3.8.1	Bread Board Prototype Testing	Everyone	2/9/22	2/11/22	3	100%
4	**Final Protoyping**					
4.1	PCB Schematic Creation	Everyone	2/14/22	2/16/22	3	100%
4.1.1	PCB Design	Yovany H	2/16/22	2/18/22	3	100%
4.1.2	PCB Ordering	Yovany H	2/18/22	2/18/22	1	100%
4.2	Enclosure Design	Darius S	2/14/22	2/16/22	3	100%
4.2.1	Enclosure Creation	Darius S	2/16/22	2/18/22	3	100%
4.3	Final Prototype Creation	Everyone	2/21/22	2/23/22	3	100%
4.3.1	Final Prototype testing	Everyone	2/23/22	2/25/22	3	100%
5	**Project Presentation**					
5.1	Poster Creation	Everyone	2/28/22	3/2/22	3	100%
5.2	Report	Everyone	3/2/22	3/4/22	3	100%
5.3	Presentation Preperation	Everyone	3/7/22	3/9/22	3	20%
5.3	Project Presentation	Everyone	3/9/22	3/9/22	1	0%



Link to file: [Time Report]