EE344: Electronic Design Lab
Project 18: IoT-based Biogas Plant Health Monitoring
Group: TUE-23

Jujhaar Singh 200110052 | Rohan Kalbag 20D170033 | Sankalp Bhamare 200110096

Faculty Mentors: *Prof. Anil Kottantharayil* and *Prof. Debanjan Bhowmik*

Abstract

This project proposes an IoT-based system that uses gas sensors to monitor humidity and carbon dioxide gas concentrations in biogas. The collected data is stored on a cloud database and accessed remotely via a web application. Real-time updates and analysis of the data enable the optimization of biogas production processes and the improvement of biogas quality. This system has the potential to contribute to sustainable biogas production and reduce greenhouse gas emissions.

Contents

1 Acknowledgments

We would like to thank the *Department of Electrical Engineering* at *IIT Bombay* for including a course such as *EE 344 - Electronics Design Laboratory* in our curriculum. Our learnings in this course have been immensely fruitful and will be of significant use when we enter the industry as engineers. We would like to extend our immense gratitude to the faculty associated with EDL, our faculty mentors, *Prof. Anil Kottantharayil* and *Prof. Debanjan Bhowmik*, the staff and faculty at *Wadhwani Electronics Laboratory* and *PCB Lab* for the seamless curation of course logistics, and their continued guidance and support. We would like to specially thank *Maheshwar Sir* and *Ankur Sir* for their assistance in the creation of the PCB and Enclosure CAD. We would also like to thank our RA, *G Sravani*, for her help throughout this course.

2 Description of Design

2.1 Overview

The proposed system is an IoT-based monitoring system designed to continuously measure and display various parameters of a biogas plant on an LCD screen. The system consists of Internet-of-Things (IoT) nodes that are equipped with sensors capable of measuring the concentrations of gaseous products of decomposition, such as methane (CH4), carbon dioxide (CO2), carbon monoxide (CO), and hydrogen sulphide (H2S), as well as humidity and temperature.

The nodes sample at a specified rate, which can be adjusted, and send their latest samples to a central server periodically. The collected data is then stored in a cloud database, which enables remote access and analysis through a web application. The web application provides a user-friendly interface for visualization and interpretation of the collected data, enabling continuous monitoring of the biogas plant parameters.

By analyzing the collected data, the system can provide valuable information for optimizing biogas production and identifying potential issues that may impact production efficiency. For instance, capturing the parameters such as the concentration of gases produced can help determine which changes in conditions allow for a change in the composition of the biogas produced. Typically the concentration of biogas is about 65% CH4 and around 30% CO2 with small amounts of the other gases and some moisture. If we can increase the concentration of methane in the biogas and reduce the concentration of other gases and water vapour, the energy that can be produced overall from the produced biogas can be significantly increased.

Furthermore, some of the gases in the mixture of biogas, like CO and H2S, can potentially irritate and, in worse cases, be fatal to humans. Thus, their levels should also be monitored. The system provides real-time updates on the status of biogas production, enabling plant operators to tune the conditions to optimize the products they obtain.

In summary, the proposed system is a monitoring system that utilizes IoT nodes with sensors to continuously measure and display various parameters of a biogas plant, providing valuable information for optimizing biogas production, identifying potential issues, and enhancing safety. In addition to this, our system is designed in a modular way using Inter-Integrated Circuit (I2C) protocol, this allows for plug-and-play support for other I2C enabled gas sensors.

2.2 Similarities with other EDL P-18 groups

There are in total three groups working on the same project. After mutual discussion as well as consultation with the faculty mentor. The similarities between various groups are as follows

- Usage of sensors of order of the range of hundreds/thousands of ppm.
- Similar pressurised gas chamber will be used by all the teams to test the various sensors.
- All the teams will develop web applications capable of storing the sensed readings into a cloud database as well as capable of triggering alerts regarding the health of the biogas plan.

2.3 Differences from other EDL P-18 groups

- Our team shall use the **ESP32 WROOM 32E SMD** chip as the MCU.
- Our team shall sense and monitor CO_2 and moisture content present in biogas, one of the other groups will monitor CO and CH_4 and the other will monitor H_2S and temperature
- Our team will develop an LCD interface to continuously display the various sensor readings.
- Our team will use a Buzzer based alarm to alert if concentrations of gases are not in desired range.
- Our team will use a USB type B port input for power supply which can be later connected to an adapter or power bank to power the entire design.
- The other teams have decided to implement data analysis on collected sensor data, layered filters to perform selective gas sensing and LED based alerts, automated control of plant

2.4 Modular Design of System

Each group is focusing on 2 sensors that are to be used. Our group will be working on the sensors that are measuring the concentration of CO_2 and the humidity level. We have decided to design the system in a modular fashion using **I2C** such that other sensors can be integrated into our designs with ease.

3 Block Diagram

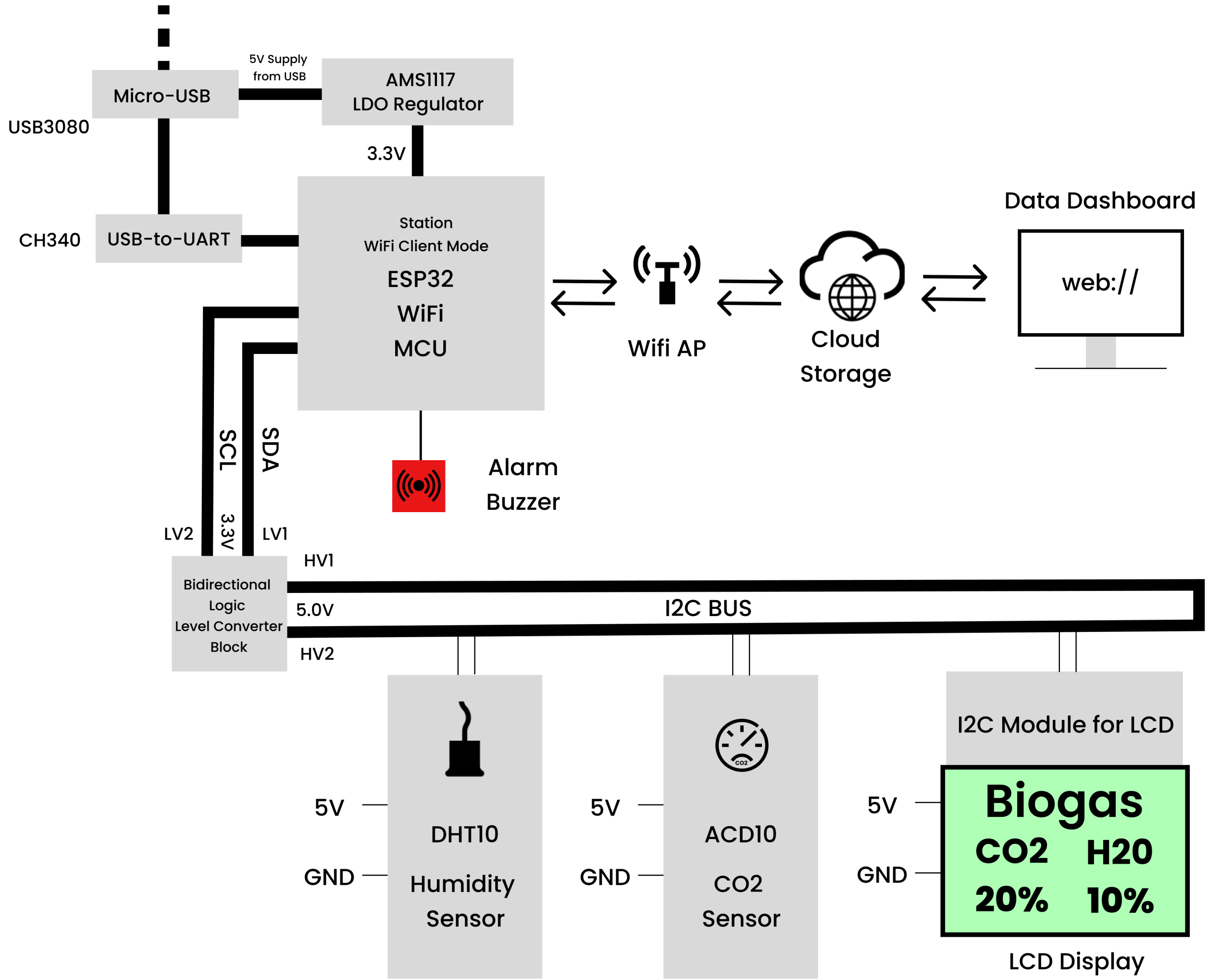


Figure 1: Overview of Subsystems

4 Principle of Operation of each Subsystem

The system will comprise of a micro-controller interfaced **ESP32 WROOM 32E Module SMD chip** will be used as the microcontroller unit, will be used to collect the data from the sensors using serial communication interfaces and then transfer it over the internet using WiFi capabilities which is contained in the ESP32 module itself, to the server, where the data will be stored and analyzed.

4.1 Microcontroller Block

We are using the ESP32 Module called ESP32WROOM32E, which will be soldered onto the PCB. We have to provide external power to the microcontroller, which we achieve through a USB port. This will also be used for programming the microcontroller using the Espressif IDE in embedded C.

Since the output voltage from the USB port is 5V, we require a voltage converter to provide 3.3V to the ESP32. For this, we use the AMS1117 LDO regulator. To allow communication between the USB and the microcontroller, we have to provide a USB to UART bridge (support for external programmer).

The ESP32WROOM32E module has wifi capabilities with an antenna onboard. We use this to connect with a local wifi access point and transfer the data from each node to the cloud.

4.2 LCD and Sensors

We are using sensors with lower max concentration levels (around 10,000 ppm) compared to the nominal concentration of gases like CO_2, close to 30%, and CH_4, close to 65%, in biogas. For this reason we will be using a separate chamber in order to dilute the gas with a mixer gas so that readings can be obtained.

The sensor used for finding the concentration of CO_2 is the *ACD10* by ASAIR. This IC offers a I2C interface through which the readings will be sent to the microcontroller.

The *DHT20* by ASAIR will be used to monitor the humidity and temperature levels. An I2C interface is available on the IC in order to communicate with the microcontroller.

There will be a common bus for all I2C peripherals which will be the LCD as well as the two sensors. An I2C conversion module for the LCD is present to allow communication with it via I2C.

4.3 Buzzer Alarm

We will use a buzzer that is connected GPIO pin 12 of the MCU. The buzzer will be activated if any of the safety norms are breached (if CO_2 conc. exceed the thresholds).

4.4 Bidirectional Logic Level Converter

Since the ESP32 works on 3.3V HIGH and 0 LOW GPIO pins, we will need a Bidirectional Logic Level Converter Block implemented using resistances and *BSS138 NMOS*, to interface the 5V peripherals with the 3.3V I2C bus

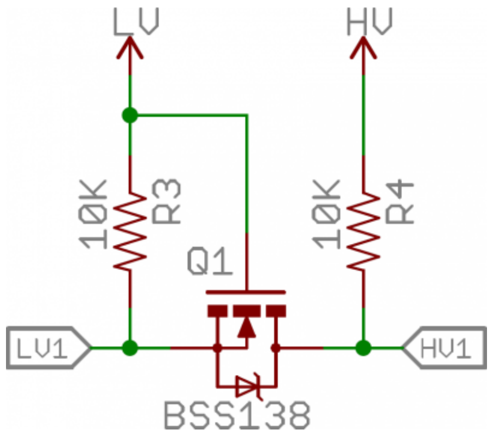


Figure 2: Bidirectional Logic Level Converter Schematic

The I2C signals such as SDA and SCL can be connected across HV1 and LV1, with the high voltage side on HV1 and low voltage side on LV1. The reference high voltage (5 V) and reference low voltage (3.3V) should be connected to HV and LV respectively.

5 Circuit Schematic & PCB Design

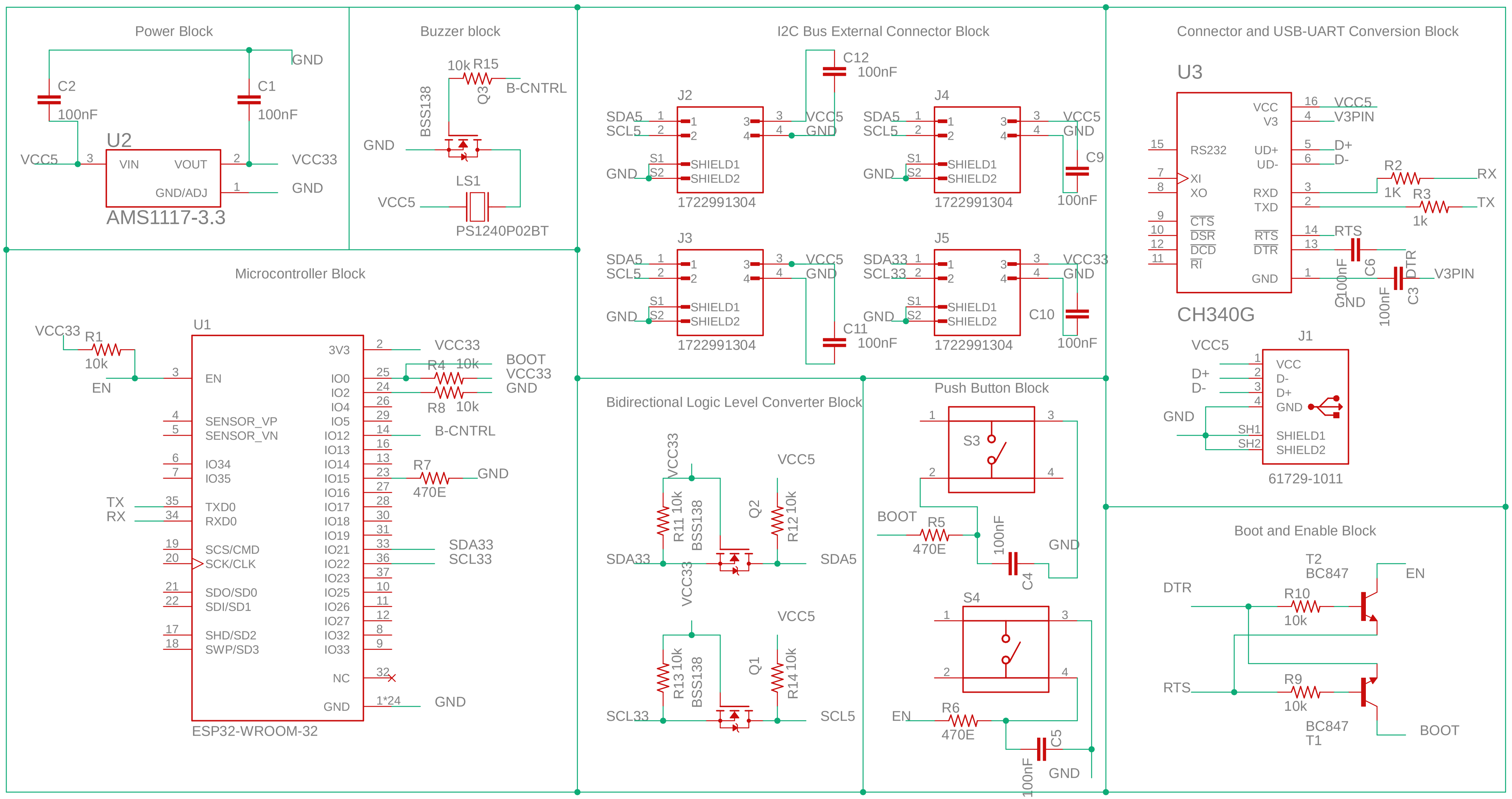
PCB circuit schematic for the project have been designed using *Fusion 360* and *Eagle* PCB design software, it consists of symbols for each component, along with the connections between the components.

5.1 Circuit Schematic

Device Name	**Symbols**
CAPACITOR SMD 100nF	C1, C2, C3, C4, C5, C6, C9, C10, C11, C12
RESISTOR SMD 10k	R1, R4, R8, R9, R10, R11, R12, R13, R14, R15
RESISTOR SMD 1k	R2, R3
RESISTOR SMD 470E	R5, R6, R7
MOLEX 1722991304	J2, J3, J4, J5
USB Type-B Conn.	J1
AMS1117-3.3	U2
BC847	T1, T2
BSS138	Q1, Q2, Q3
CH340G	U3
ESP32-WROOM-32	U1
Buzzer	LS1
Push button	S3, S4

Table 1: Schematic Symbols

The detailed circuit schematic of the system can be found in the next page



26-03-2023 01:52 C:\Users\Lenovo\Downloads\23_March_2023_BIO_GAS_UPdated\23_March_2023_BIO_GAS_UPdated\23_MARCH_2023.sch (Sheet: 1/1)

5.2 PCB Design

5.2.1 Function of the PCB

The function of the PCB is to act as a fully functioning IoT node in our system. The schematic of the PCB has been discussed and described in 5.1.

The PCB has 4 outgoing I2C connectors that will be used to interface with various other components in the system like the sensors and the LCD.

5.2.2 Component Placement and Routing

The components were placed according to their positions in the final schematic so that minimal routes can be made so as to make the design functional.

During routing, acute angles and right angles were avoided at all costs, keeping in line with PCB design best practices, so as to avoid issues in PCB printing and functioning. Extra vias were added in the PCB to avoid interference in the whole PCB design.

The PCB route width, via drill sizes, and all other parameters that come under the purview of PCB DRC checks were kept completely compliant with the standards that are required by the WEL PCB printing guidelines and requirements. For instance, the drill width and the route width are both kept at 20 mils.

The PCB design process was greatly helped by the sessions conducted by WEL and the EDL faculty. Maheshwar sir and Ankur sir played a large role in helping realize the final PCB design.

5.2.3 PCB Printing

We got our PCB (and a clone for backup) printed from PCBPower and used the same in our project. The design of the PCB was completely tested and the correct functioning of the hardware was ascertained through rigorous hands-on testing.

5.2.4 PCB Layout

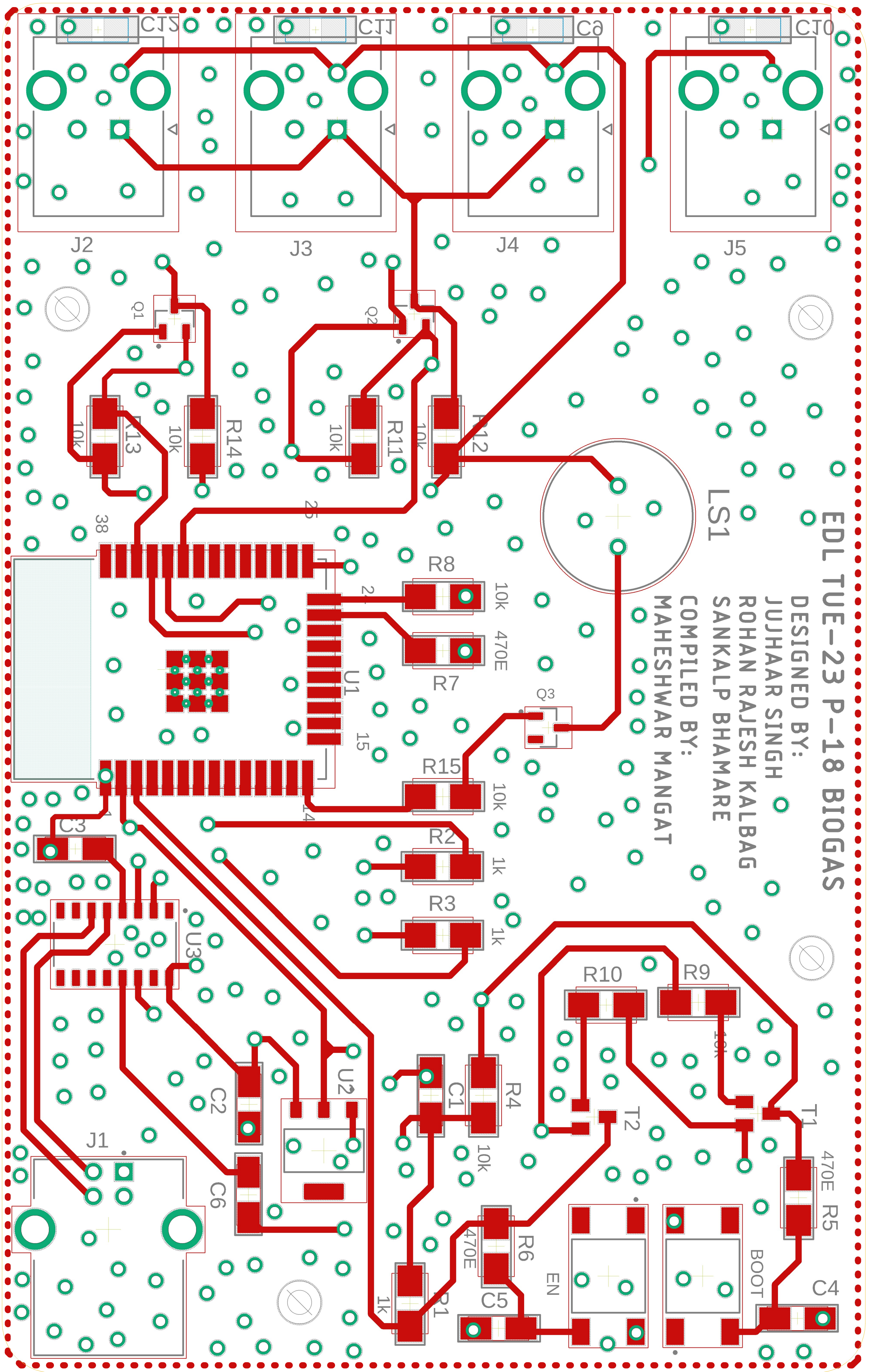
The next three pages contain layout schematic generated by Autodesk Eagle

1. Top Layer

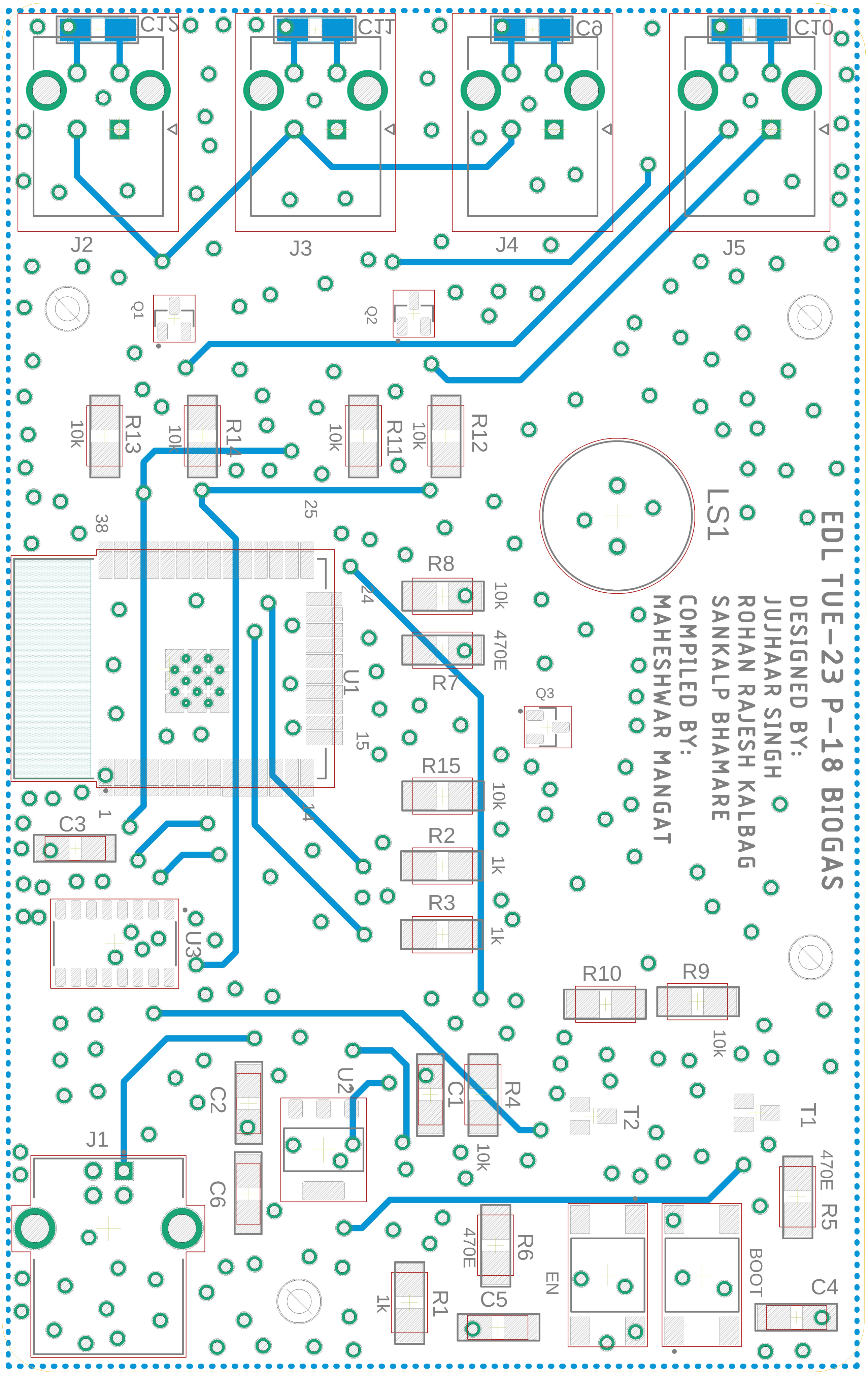
2. Bottom Layer

3. Both Layers

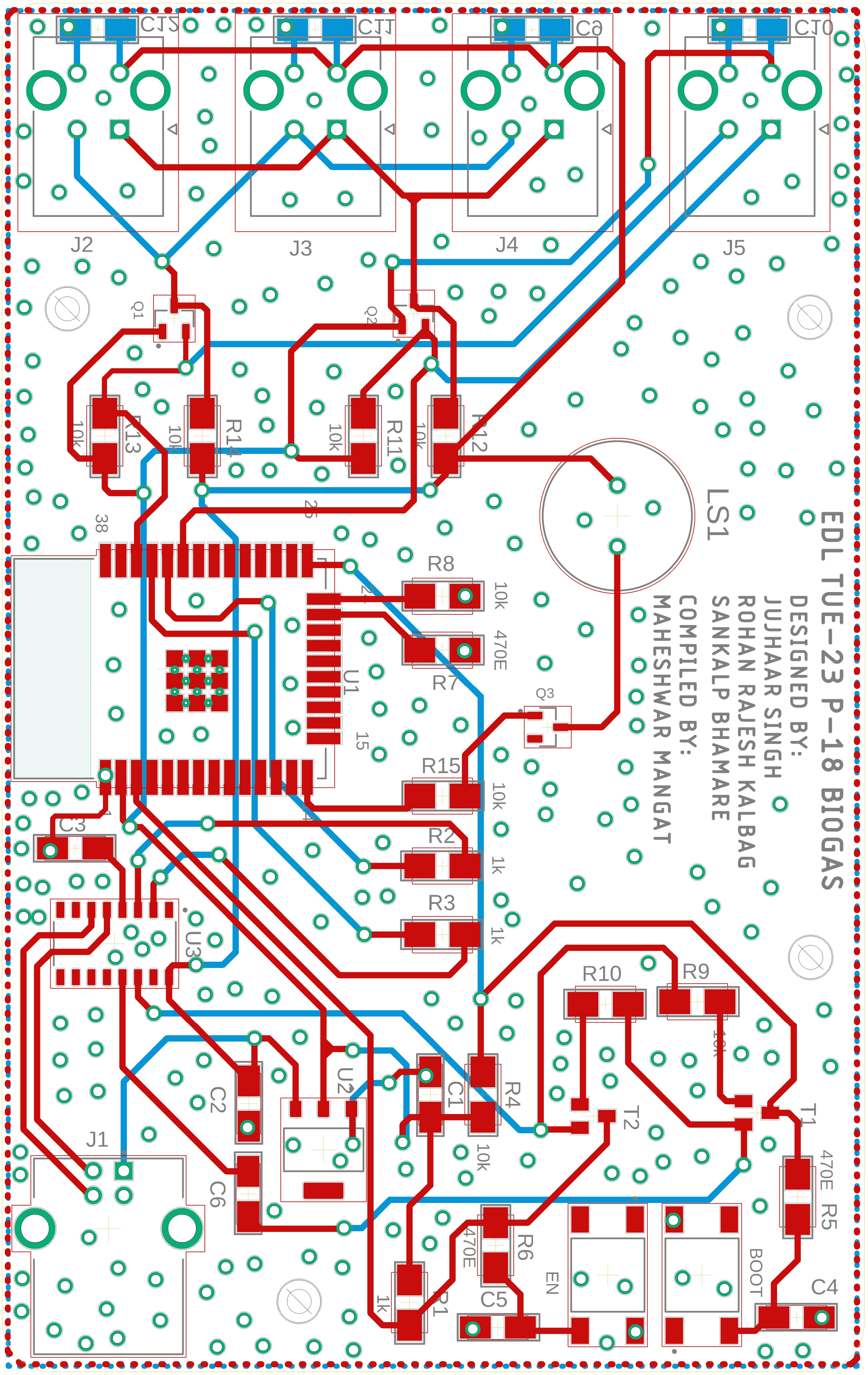
26-03-2023 20:19 f=2.40 C:\Users\Lenovo\Downloads\23_March_2023_BIO_GAS_UPdated\23_March_2023_BIO_GAS_UPdated\23_MARCH_2023.brd



26-03-2023 20:20 f=2.40 C:\Users\Lenovo\Downloads\23_March_2023_BIO_GAS_UPdated\23_March_2023_BIO_GAS_UPdated\23_MARCH_2023.brd



26-03-2023 01:47 f=2.40 C:\Users\Lenovo\Downloads\23_March_2023_BIO_GAS_UPdated\23_March_2023_BIO_GAS_UPdated\23_MARCH_2023.brd



5.3 CAD Design

The main system is enclosed in a laser-cut 3 mm acrylic sheet box (the CAD files are in `Enclosure CAD Files/`). Six finger cut faces were cut along with cutouts for the LCD display and other ports/buttons on the PCB. The final box was assembled by connecting all the faces using a glue gun, and PCB was bolted down to the bottom face and LCD to the top face.

Link to video of laser cutting performed at WEL

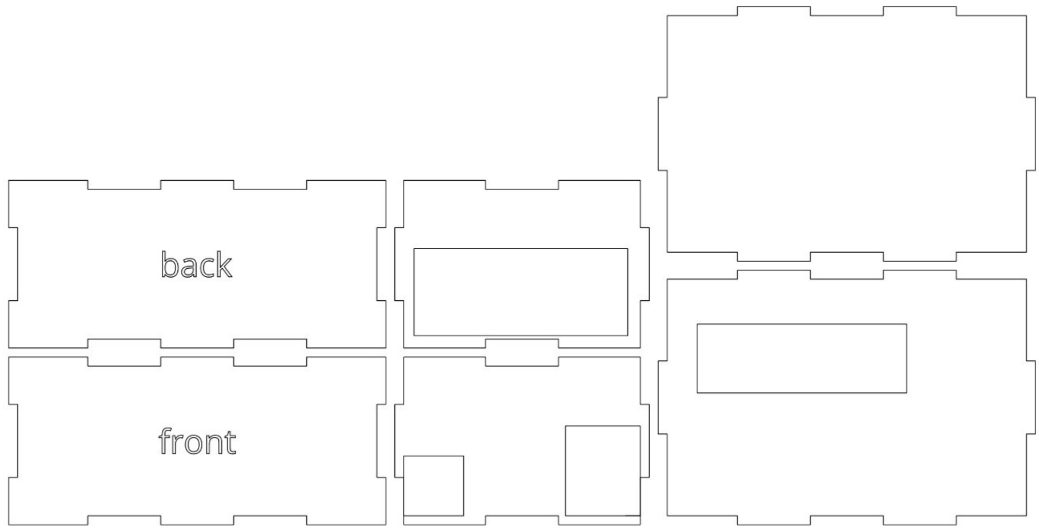


Figure 3: CAD Schematic

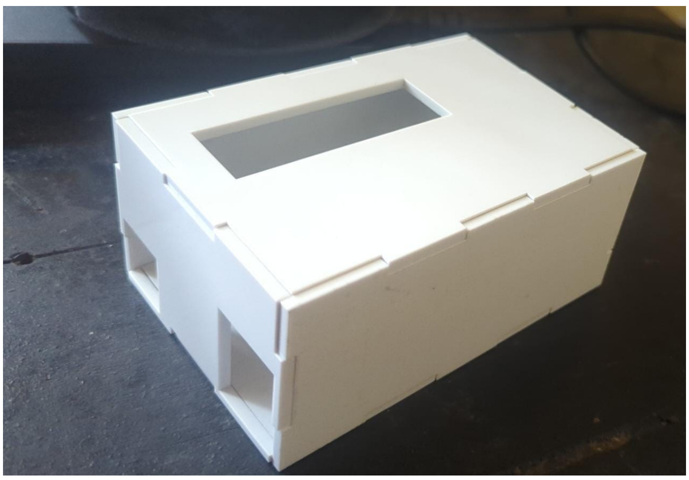


Figure 4: Assembled Enclosure Box

6 Revised Bill of Materials

The following table contains the materials finally used in our project.

Product Name	Product No	Quantity	Cost (INR)
CO_2 Sensor	ACD10	1	1,299
Humidity Sensor	1568-SEN-18364-ND	1	538
ESP32-WROOM-32E	ESP32WROOM32E	1	270
USB B Type Connector	UJ2-BH-4-TH	1	76
BC847BWQ-13-F - BJT	BC847BWQ-13-F	1	15.5
AMS1117 - LDO Regulator	AMS1117	1	40.5
BSS138 NMOS	BSS138	2	21
16x2 LCD Display	EC-0048	1	119
I2C Module for LCD	ST2001SM0062	1	86
5V Buzzer	-	1	6.9
Push Button	-	2	34
CAPACITOR SMD 100nF	-	12	22
RESISTOR SMD 10k	-	10	9
RESISTOR SMD 1k	-	2	2
RESISTOR SMD 470E	-	3	3
PCBPower Printed PCB	-	2	2800
USB-UART Module	REES52	1	499

7 Testing Setup

7.1 Preliminary In-Lab Testing

To test the various components and the final PCB, interface the microcontroller with the sensors as well as to develop a basic webserver to which the sensor readings are uploaded, also to visualize the sensor readings on a LCD.

7.2 Components used in Experimental Setup

ESP32-WROOM-32 module, USB-UART programmer module, ACD-10 sensor, DHt-20 Humidity sensor, USB to micro-USB cable, LCD module, I2C module for LCD

7.3 Procedure

We devise a set of tests for the individual components of the stack and then perform final end testing.

7.3.1 Flashing ESP32 with Embedded C Code

- Make sure **Espressif IDE** is installed on the computer for programming of the ESP32 microcontroller using Embedded C and also to flash it
- Connect the Serial TX/RX pins of the programmer with RX/TX pins of the ESP32 module on the PCB
- Connect the External USB programmer to a USB port of your laptop, and make sure to identify its port number
- Make a new empty project on **Espressif IDE** and enclose all the files `.c` and `.h` in `Source Code/firmware/`, into the main directory generated by Espressif

```
main
|_ drivers
|_ main.c
|_ server.conn.c
|_ server.conn.h
```

- Now run `flash` command on the IDE, once the IDE shows flashing status, hold `BOOT` button and then press `EN` button (repeat until it connects successfully)
- Now code is burnt to the ESP32, press EN to boot up and start up the ESP32

Configuring WiFi Endpoint Details for IoT

In order to configure the settings to our custom usage refer to the instructions below:

WiFi Endpoint: open **server_conn.h** and set the macros **ESP_WIFI_SSID** and **ESP_WIFI_PASS** to configure the hotspot name and password.

IoT Server Endpoint: open **server_conn.h** and set the macros **WEB_SERVER** and **WEB_PORT** to configure the server hostname and port.

- After configuring the macros in the above textbox then build the files and flash it into the microcontroller using **Espressif IDE**.

7.3.2 ACD-10 CO_2 Sensor

Before implementing the stack it will be useful if we can check the functioning of individual sensors, since the CO_2 sensor supports UART communication it can be connected to laptop via UART chip and the data can be monitored using the serial interface.
Using the `monitor.py`(5) we can monitor the readings of the sensor and check if it is working.

```python
import serial
import time
ser = serial.Serial('<device-portname>',
                    baudrate=1200)

packet = bytearray([0xFE , 0xA6 ,
                    0x00 , 0x01, 0xA7])
for i in range(1000):
  ser.write(packet)
  time.sleep(1)
  data = ser.read(9)
  val = data[4:9]
  print("\co2(ppm):",int(val[0]) * 256 + val[1],
        end="")
  time.sleep(0.5)
ser.close()
```

Figure 5: `monitor.py`

Now once we have verified our sensor is working we can proceed with integrating it in our stack. The GPIOs **22** and **21** were used for the SCL, SDA signals in I2C. The LCD and the ACD10 sensor's SCL and SDA will be connected here and this will be used as a common bus.

7.3.3 Basic Setup of LCD

- Connect the LCD to the I2C Module
- Connect the SDA and SCL signals to the PCB I2C connector VCC to VCC and GND to GND

7.3.4 Basic Setup of Sensors and Webserver

For testing the working of sensors and end to end working of the system a prototype was setup as follows:

- Connect the sensors to the on-board I2C connectors on the PCB (Connect SDA, SCL, GND, 5V lines).
- Power up the board using external USB
- Setup a WiFi AP with SSID name inputted in (7.3.1) (In our example we set up a wireless hotspot **RnD** and password: **12345678** for ESP-32 to connect to IoT cloud for uploading data.
- Make sure your python has the `flask` library for web server development installed, if not do the following `pip install flask`
- Create a terminal in a directory containing `script.py` and a directory `templates` containing the file `index.html` these files are in `Source Code/server/` and follow the following directory structure

```
|_ script.py
|_ templates
     |_ index.html
```

- Enter the following in the terminal `python3 script.py`
- Now open your browser and enter the following in the url `localhost:5000`

7.3.5 Verify Working of Sensor and Buzzer

- Wait for the ESP 32 to boot up and connect to the web server.
- Now open the webserver dashboard on `localhost:5000`
- After few seconds the web server will start polling the data and display a plot of CO2(ppm) and Humidity live readings.
- The live readings of CO2 and Humidity sensor should be seen in the LCD.
- Now start blowing air on the ACD-10 sensor this will cause the CO_2 concentration to increase and the readings will start rising. (this can be observed by the rising graph). The buzzer should start ringing when the CO_2 readings cross the set threshold which is 3000 ppm.
- Now stop blowing air and wait for the readings to again fall back to initial levels. The buzzer should stop ringing when reading goes below threshold.

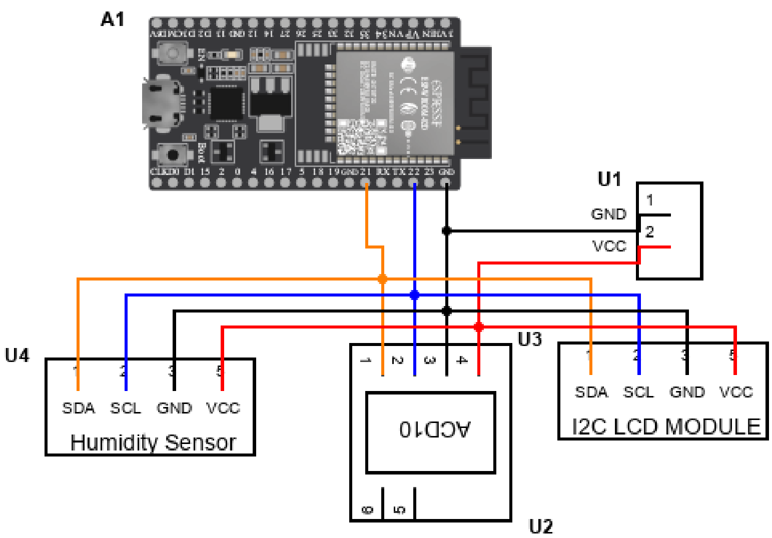


Figure 6: Connection Schematics

7.4 Final Testing Setup at Biogas plant

- The final PCB was assembled with the enclosure and parts as shown in the below figure. A small general purpose perforated PCB was used to connect all peripherals to the common I2C bus and all the sensors were connected.
- The external REES52 USB-UART programmer was used to flash the code onto the board (only once).
- The board is made to connect to the hotspot RnD. The server was set up via localhost on a laptop.

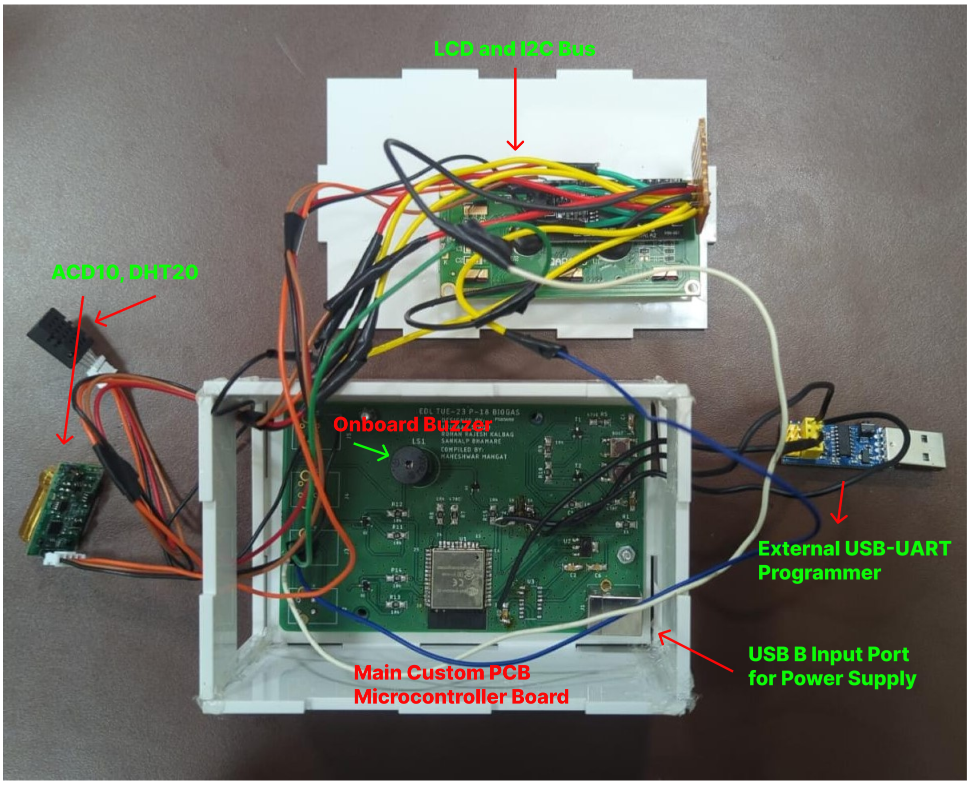


Figure 7: Contents of Final System

8 Results and Observations

8.1 In-Lab Testing

- A real-time graph of CO_2 readings in ppm and humidity was obtained on the webserver as expected which can be seen in the image below.
- Live readings were obtained on the LCD in the desired format as expected.
- As we breath on top of the sensors, CO_2 and humidity readings start rising (as our breath has high concentration of CO_2 and is moist).
- On excessive breathing, CO_2 concentrations rises above 3000 ppm, triggering the buzzer alert system.

Figure 8: Web Dashboard with Live Readings Obtained During In-Lab Testing

8.2 Biogas Plant On-Site Testing

The system was setup near a Biogas plant with the humidity and CO_2 sensors placed near the nozzle of the plant.

- The ambient readings of the sensors were noted (nozzle was closed), around 400 ppm CO_2 (conc.) and 50 % humidity, agreeing with the expected atmospheric levels in the testing environment (Powai, Mumbai) indicating the reliability of the sensor readings.

- Now nozzle was opened, and the readings were allowed to stablise we noted around 500 ppm CO_2 (conc.) and an increase in humidity to around 54% as expected, since CO_2, H_2O are a significant fraction of Biogas.

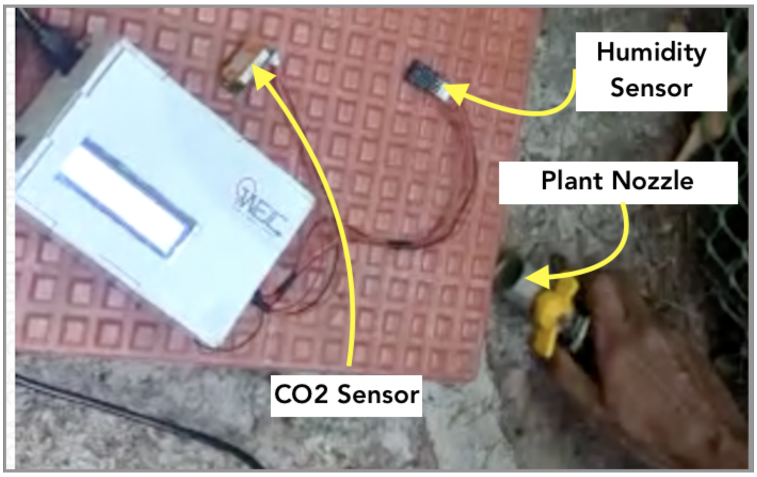


Figure 9: Sensors Placed in Front of the Biogas Plant Nozzle

8.3 Demonstration Videos

- Link to Breadboard Web Server Demonstration Video
- Link to Breadboard LCD Demonstration Video
- Link to Final Presentation Video with Biogas On-site Testing

9 Risk Mitigation

Being a hardware system, we had anticipated the following problems during the design phase and here is how we dealt with them:

- **Failure of USB-UART IC**: The IC used for USB-UART conversion (CH340C) to program the microcontroller was unreliable. Thus, we used an external USB-UART module for programming.
- **Power Supply Insufficiency**: Used wall adapters with the appropriate rating (5V/1A)

- **Connector Robustness**: Successful communication with sensors over the connectors to obtain valid readings and plots
- **On-board Microcontroller**: The reliability of the standalone on-board microcontroller was verified after testing performed on the PCBpower PCB.
- **Multiple IC I2C Bus Testing**: System is working with multiple I2C sensors as well as the LCD module on a single bus

10 Conclusion

- Developed a completely packaged, modular IoT product to monitor various parameters of a Biogas plant.
- The system developed by us is able to monitor CO_2 readings in ppm, humidity in relative %, display the live readings on a web server using IoT, and is **modular**, allowing easy integration of other I2C compatible gas sensors and is capable of monitoring the health of a Biogas plant.
- Additionally, we were able to mitigate all of our anticipated problems during the development and prototyping stages.

11 Future Work

The set targets for the project were successfully achieved, some of the key future work in plan include

- Developing a comprehensive system for the biogas plant that can measure the biogas produced and actuate gas valves for safety mechanisms.
- Creating a machine learning model to optimize CO_2 and humidity levels using data gathered from the biogas plant system.
- Exploring other potential applications for our system in related industries, such as remote gas monitoring, as this system offers a low-cost alternative to the typically expensive commercial solutions.

Implementing the remote gas monitoring solution to improve safety and efficiency in various industries beyond biogas production.