SUMMER INTERNSHIP

(3170001)

A PROJECT REPORT

Submitted by

ANURAAG PAL

190130111026

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

Department of **Electronics and Communication Engineering**

Government Engineering College, Gandhinagar

Gujarat Technological University, Ahmedabad

July, 2022

Government Engineering College

Sector-28, Gandhinagar

CERTIFICATE

This is to certify that the project report submitted along with the project titled **Summer Internship** has been carried out by **Anuraag Pal** under my guidance in partial fulfillment for the degree of Bachelor of Engineering in Electronics and Communication Engineering, **7th *Semester*** of Gujarat Technological University, Ahmedabad during the academic year 2022-23.

Avani A Vithalani
Internal Guide

Tejas B. Sheth
Head of the Department

Government Engineering College

Sector-28, Gandhinagar

DECLARATION

We hereby declare that the Internship/Project report submitted along with the Internship/Project entitled **Summer Internship** submitted in partial fulfillment for the degree of Bachelor of Engineering in Electronics and Communication Engineering to Gujarat Technological University, Ahmedabad, is a bonafide record of original project work carried out by me/us at **ELECTROGENICS INNOVATIONS PRIVATE LIMITED** under the supervision of **Mr. Manish Mandaviya** and that no part of this report has been directly copied from any students' reports or taken from any other source, without providing due reference.

Name of the student	**Sign of student**
1. ANURAAG PAL	_____

ACKNOWLEDGEMENT

First, I would like to thank **Mr. Manish Mandaviya** Sir, **Director of Electrogenics Innovations**, Ahmedabad for allowing me to do an internship within the organization. In addition, I would like to **thank all the people** with whom I worked at Electrogenics Innovations, Ahmedabad. With their patience and openness, they created an enjoyable working environment. It is with great pleasure and a deep sense of gratitude that I acknowledge the assistance of these individuals.

I am highly indebted to **Prof. Avani A Vithalani** and **Principal Dr. Shweta P. Dave**, for the facilities provided to accomplish this internship. I would like to thank my Head of the Department **Prof. Tejas B. Sheth** for his constructive criticism throughout my internship. I would like to thank **Prof. Devendra Patel**, College internship coordinator Department of Electronics and Communication Engineering.

ABSTRACT

This report is a detailed overview of my internship journey at Electrogenics Innovations Private Limited. During my internship, I have learned a lot about how the industry of software and hardware development actually works. I have learned what are the parameters, how to work on an actual project, and how to work in a flow of teamwork. I have known about the workflow of IoT developers and Product Development roles and responsibilities. I have learned to work in a corporate space which not only enriched me professionally but also helped me grow personally as well. My contribution was appreciated by my supervisor and other members of the department. The career path I would be selecting for myself is quite influenced from my internship as I have had a great opportunity to practically see how startups are building products and their working and evolving around the entire Globe. However, this report has been written in a short time. I have tried my level best to make it meaningful by reflecting on my work at the Company. Also, I have summarized my overall experience, with my learning and challenges faced as an intern.

List of Figures

Table of Contents

Chapter 1 COMPANY PROFILE

1.1 Overview

Technology First, that's our approach which drives our company as innovation studio. We develop technology that helps our client get what they want. At Electrogenics Innovations we focus on understanding requirement of clients, develop innovative solutions and product.

With unparalleled expertise in field of medical electronics, industrial automations and customized Smart Home, we are committed in designing the applications and products using advance and internationally benchmarked technology which assures world class quality. The most obvious advantage in investing in technology is quality, speed and efficiency, and that's what we offer to our clients.

Manufacture of medical appliances and instruments and appliances for measuring, checking, testing, navigating and other purposes except optical instruments.

1.2 History of Company

Started in the year 2020 our company is using our creative potential in numerous challenging processes - from generating new ideas of improvements to integrating them by means of business plans. Year by Year company's growth is

increasing in a such a good way. While each of our projects is based on deep market study, we apply our knowledge about it to improve ourselves to produce better ideas for other companies.

1.3 Our Product

Figure 1.1 Advanced Neonatal Respiratory Support Device

We are coming with an innovative technology for Neonates. An Advanced Neonatal Respiratory Support Device ***" Breath.ei "***

1.3.1 Key Features

1. 🔳 Multiple Modes: Nasal CPAP, Bubble CPAP, HFNC, Auto Resuscitation
2. 🔀 Built-in Air Source
3. 🔋 90-120min battery Backup
4. ⚖️ Light Weight <3Kg

Many More....

1.3.2 Product Gallery

Figure 1.2 Product Features

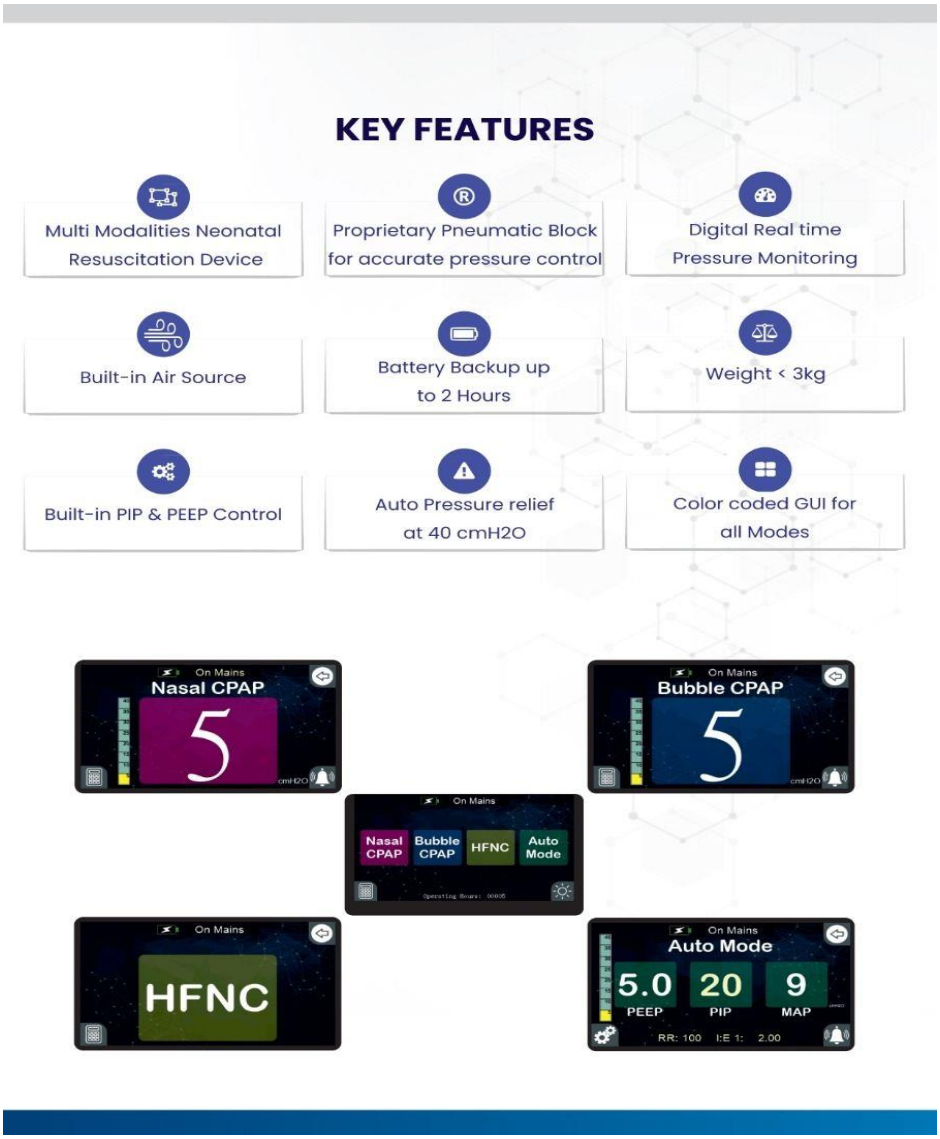

Figure 1.3 Product Display

Chapter 2 ACCESS HEXADECIMAL DATA USING UART COMMUNICATION

2.1 Overview

In this project, we are accessing Hexadecimal data using UART communication. This project is a part of a larger project that will be integrated into a product later.

As we know, UARTs transmit data asynchronously. This means there is no clock signal to synchronize the output of bits from the sending UART to the sampling of bits by the receiving UART. Instead of a clock signal, the transmitting UART adds start and stop bits to the transferred data packet.

And, we used UART communication not only for data transfer between devices but also between Serial Monitors and STM32F103 Microcontroller through a transistor-transistor logic (TTL).

In this Hex-Data Accessing and Distributing System, we will utilize the STM32F103 Blue pill board to access the hexadecimal data received from a device (but here we use Arduino Mega as the Transmitter of the hex data frame) and after filtering the data frame, we print the data on the Arduino IDE's Serial Monitor. Using the Arduino Serial Monitor, data will be displayed in hexadecimal based on the category of the data frame and data frame error detection and correction. When a Data Frame changes, the receiver detects data on a character-category basis and displays it on the Serial Monitor in hexadecimal format. Thus, we can monitor the accessed hexadecimal data from one device to another device.

2.2 Components Used

Sr. No	Component	Quantity
1.	STM32F103C8T6 Blue Pill Microcontroller	1
2.	Arduino Mega Microcontroller	1
3.	Transistor-Transistor Logic PL2303	1
4.	Jumper Wires	Some

2.3 Data Frame Format to be accessed by receiver

Start Character	**Packet Length**	**Parameter Type**	**Packet Type**	**Packet ID**	**Serial number**	**Data Content**	**Check sum**
0xFA	0x0D	0x03	0x04	0x84	0x00, 0x00, 0x00, 0x2F	Byte1-A (ex. 0x28) Byte2-B (ex. 0x00) Byte3-C (ex. 0x09)	Ex. 0xD9
0xFA	0x11	0x03	0x04	0x85	0x00, 0x00, 0x00, 0x2F	Byte1-Low 8 Bit D Byte2-High 8 Bit D Byte3-E Byte4-Low 8 Bit F Byte5-High 8 Bit F Byte6-G Byte7-H	?
0xFA	0x0D	0x03	0x04	0x86	0x00, 0x00, 0x00, 0x2F	Byte1-I Byte2 Byte3	?
0xFA	0x0C	0x03	0x04	0x87	0x00, 0x00, 0x00, 0x2F	Byte1-Low 8 Bit J Byte2-High 8 Bit J	?

2.4 Hardware Description

2.4.1 STM32F103C8T6

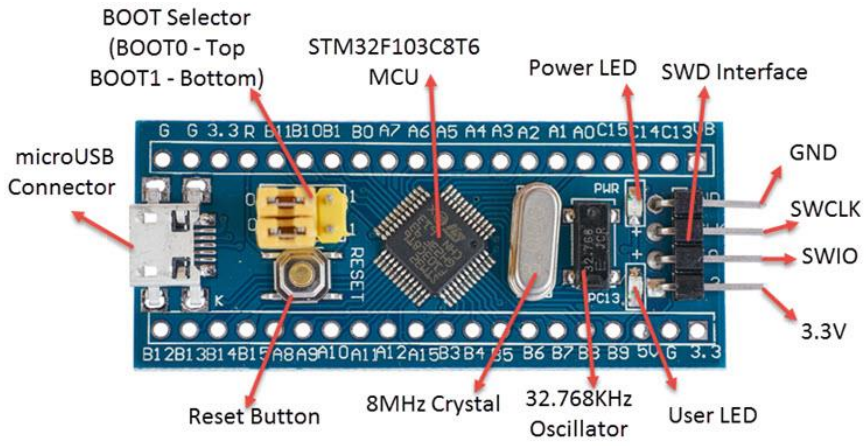

Figure 2.1 Hardware Description of STM32F103C8T6 (STM32 Blue Pill)

The blue pill, also known as the **STM32F103C8T6** is by far the most popular one that shows up when I am searching for starter stm32 boards and honestly, I think it is just because it has a nickname, the blue pill. Unlike Arduino, STM32 doesn't really have a whole lot of nicknames they're almost always referred to by the part number themselves. The part number tells us a lot about the chip. I think the most important parts are these first few digits here the F-103 and everything after that is kind of more of the technicalities like how many pins there are or what the footprint looks like. So it is pretty common to see people reference shorter part numbers to abbreviate or generalize them.

2.4.2 STM32 with Arduino IDE using TTL

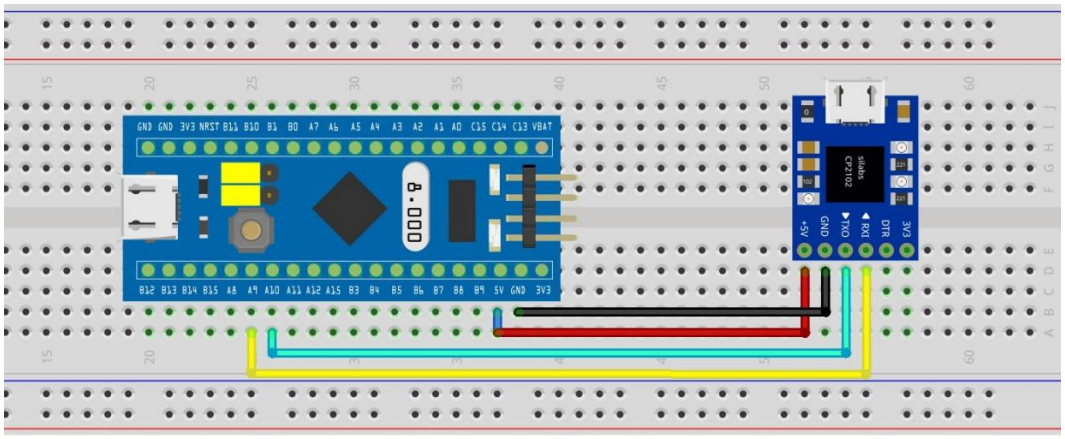

Figure 2.2 Connection Between STM32F103C8T6 and Transistor-Transistor Logic

We can use the 5-volt pin and connect it to 5 volt of USB to TTL and ground to ground. The PA9 which is TxD this goes to RxD of USB to TTL and the PA10 that is our RxD pin which goes to TxD of USB to TTL. No separate power source is required when you plug on the USB to PC. It will supply the voltage you can see the yellow line in the stm32 pinout image these all pins are 5 volts tolerant. So you can safely use this USB to TTL device. After you have connected your USB to TTL converter with the STM32. You will need to do one more change. You can see these Yellow colour **Boot Jumpers**, you will need to change the position of the jumper marked with Red colour. So simply unplug this and connect it again but this time the right leg and the middle leg should short. So, at this point, now you are all set. Now you can connect your USB to TTL converter with the computer or laptop through a cable and the LED will light up.

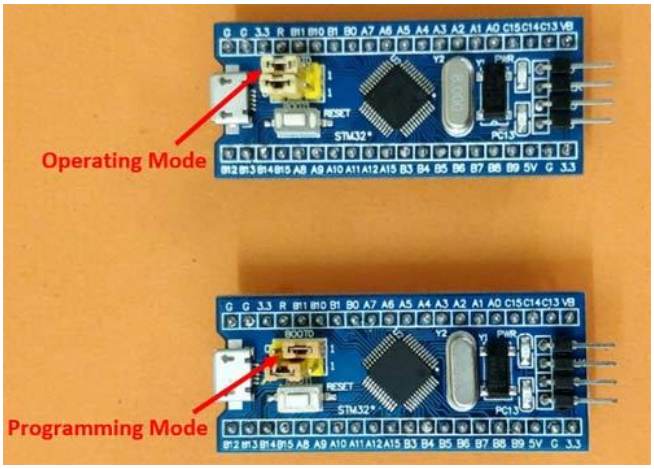

Figure 2.3 Operating Mode and Programming Mode

2.4.3 Arduino Mega and Arduino IDE

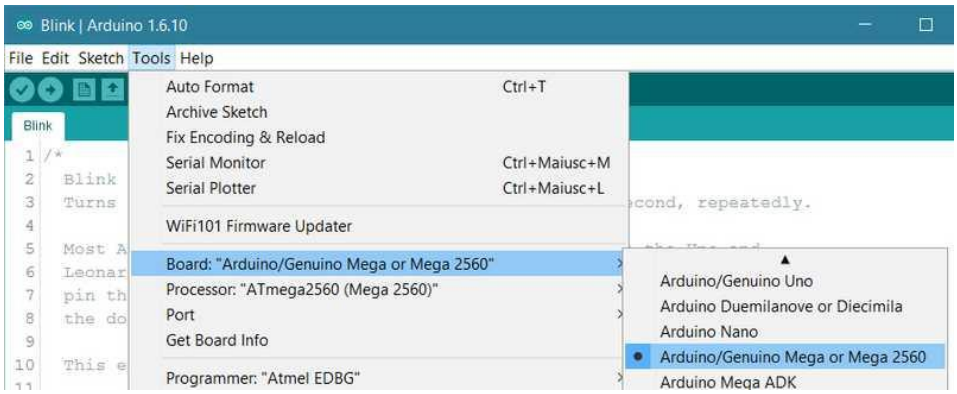

Figure 2.4 Board Selection in IDE: Arduino Mega

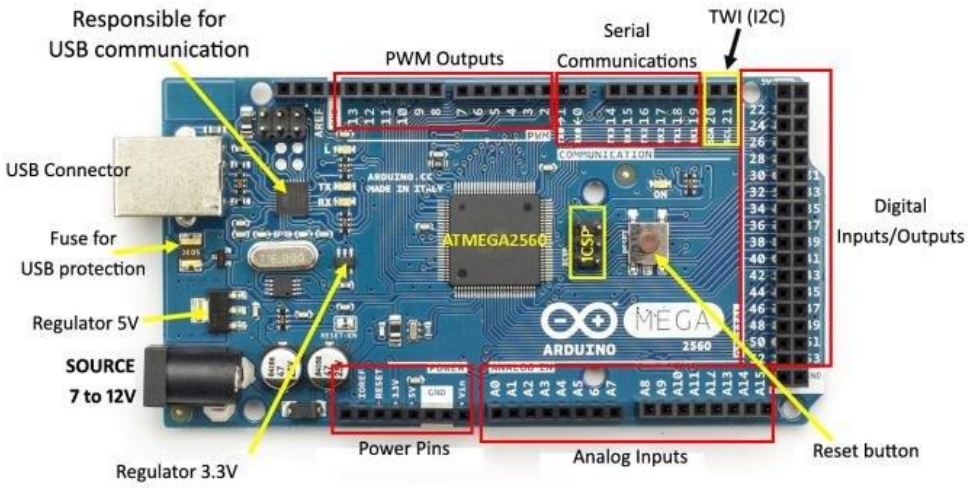

Figure 2.5 Arduino Mega Board Description

2.4.4 UART Communication

The UART that is going to transmit data receives the data from a data bus. The data bus is used to send data to the UART by another device like a CPU, memory, or microcontroller. Data is transferred from the data bus to the transmitting UART in parallel form. After the transmitting UART gets the parallel data from the data bus, it adds a start bit, a parity bit, and a stop bit, creating the data packet. Next, the data packet is output serially, bit by bit at the TxD pin. The receiving UART reads the data packet bit by bit at its Rx pin. The receiving UART then converts the data back into parallel form and removes the start bit, parity bit, and stop bits. Finally, the receiving UART transfers the data packet in parallel to the data bus on the receiving end.

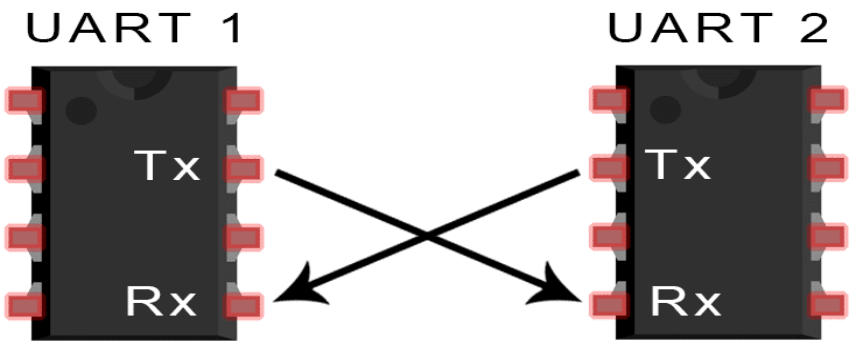

Figure 2.6 UART Communication: Data Flow from TxD to RxD

2.5 Pinout Connections

	TTL	-	**STM32F103C8T6**
1.	+5V	-	Vin
2.	GND	-	GND
3.	RxD	-	A9
4.	TxD	-	A10

	Arduino Mega	-	**STM32F103C8T6**
1.	TX1	-	A3
2.	RX1	-	A4

2.6 STM32F103C8T6 Data Accessing Code

```
byte arr[50];
byte A, B, C, D, Di, E, F1, F2, G, H, I, J1, J2;


void setup() {
 // put your setup code here, to run once:
 Serial2.begin(115200);
 Serial.begin(9600);
}
void loop () {
 // put your main code here, to run repeatedly:
 while (Serial2.available() == 0);
 arr[0] = Serial2.read();
 if (arr[0] == 250) {
 for (int i = 1; i < 5; i++) {
```

```
    arr[i] = Serial2.read();
  }
if (arr[4] == 132) {
    for (int i = 5; i < 13; i++) {
      arr[i] = Serial2.read();
    }
  A = arr[9];
 B = arr[10];
 C = arr[11];
  delay (10);
}
  else if (arr[4] == 133){
    for (int i = 5; i < 17; i++) {
      arr[i] = Serial2.read();
  }
    D = arr[9];
    Di = arr[10];
    E = arr[11];
    F1 = arr[12];
    F2 = arr[13];
    G = arr[14];
    H = arr[15];
    delay(10);
  }
```

```
  else if (arr[4] == 134) {
    for (int i = 5; i < 13; i++) {
      arr[i] = Serial2.read();
    }
    I = arr[9];
    delay(10);
  }
  else if (arr[4] == 135)
  {
    for (int i = 5; i < 13; i++) {
      arr[i] = Serial2.read();
    }
    J1 = arr[9];
    J2 = arr[10];
    delay(10);
  }
 delay(10);
//Printing the Data Accessed by STM32 on Serial Monitor:
  Serial.print(A, HEX);
  Serial.print(" ");
  Serial.print(B, HEX);
  Serial.print(" ");
  Serial.print(C, HEX);
  Serial.print(" ");
```

```
    Serial.print(((D << 8) | (Di)), HEX);
    Serial.print(" ");
    Serial.print(E, HEX);
    Serial.print(" ");
    Serial.print(((F1 << 8) | (F2)), HEX);
    Serial.print(" ");
    Serial.print(G, HEX);
    Serial.print(" ");
    Serial.print(H, HEX);
    Serial.print(" ");
    Serial.print(I, HEX);
    Serial.print(" ");
    Serial.println(((J1 << 8) | (J2)), HEX);
  }
}
```

2.7 Arduino Mega Data Sending Code

```
void setup () {
  // put your setup code here, to run once:
  Serial1.begin(115200);
}
void loop () {
  char data [] = {0xfa, 0x0d, 0x03, 0x04, 0x84, 0x00, 0x00, 0x00, 0x2f, 0x28, 0x00,
0x09, 0xd9
```

```
        , 0xfa, 0x11, 0x03, 0x04, 0x85, 0x00, 0x00, 0x00, 0x2f, 0x29, 0x01, 0x10,
0x13, 0x14, 0x15, 0x16, 0xf4

        , 0xfa, 0x0d, 0x03, 0x04, 0x86, 0x00, 0x00, 0x00, 0x2f, 0x30, 0x02, 0x11,
0xf6

        , 0xfa, 0x0c, 0x03, 0x03, 0x87, 0x00, 0x00, 0x00, 0x2f, 0x31, 0x03, 0xf7

        , 0xfa, 0x0d, 0x03, 0x04, 0x84, 0x00, 0x00, 0x00, 0x2f, 0x51, 0x04, 0x12,
0xda

        , 0xfa, 0x11, 0x03, 0x04, 0x85, 0x00, 0x00, 0x00, 0x2f, 0x52, 0x06, 0x13,
0x14, 0x15, 0x16, 0x17, 0xf5

        , 0xfa, 0x0d, 0x03, 0x04, 0x86, 0x00, 0x00, 0x00, 0x2f, 0x53, 0x07, 0x14,
0xf7

        , 0xfa, 0x0c, 0x03, 0x03, 0x87, 0x00, 0x00, 0x00, 0x2f, 0x54, 0x08, 0xf9

};

 for (int i = 0; i < 109; i++)

  {

    Serial1.write(data[i]);

    delayMicroseconds (100);

  }

}
```

2.8 Conclusion

After Hardware Connections and successful upload of the program, open the Serial Monitor at the baud rate of 115200 on the COM attached with TTL. Using the serial monitor, we can see the accessed hexadecimal data across the different categories without errors. This project will help the company a lot in sending and checking coded hexadecimal data between devices for their future projects.

Chapter 3 AIR PRESSURE MONITORING SYSTEM

3.1 Overview

In this project, we built an Air Pressure Status Monitoring System**.** This project is a part of a larger project that will be integrated into a product later.

As we know, the Air Pressure Sensor in any system or device is used for measuring Air Pressure as it's one of the main components. As a result, Air Pressure measurement is necessary for those devices. Hence, we need to monitor the air pressure level of the device environment or tank/cylinder.

These days, there are highly embedded air pressure monitoring systems that monitor the pressure, humidity, temperature, and buzzer signals of the surrounding.

In this IoT-based Air Pressure Monitoring System, we will use the STM32F103C8T6 microcontroller board to send the pressure data to the Arduino Serial Monitor via Transistor-Transistor Logic (TTL). In the previous chapter, section 2.3.2 describes the connection between STM32F103C8T6 and TTL. Air Sensor MSP40-GDF is not directly connected to STM32 but to ADC HX711, which converts the value into analog. The Arduino Serial Monitor will display the Air Pressure Values in kilopascal in every environment. As the air pressure changes the Air Sensor value changes and with that, the ADC maps itself and sends the analog data of air pressure to STM32. Then, with TTL and UART communication between them, we can see that the value of the environment is getting changed. Thus, we can monitor the Air Pressure.

3.2 Components Used

Sr. No	Component	Quantity
1.	STM32F103C8T6 Microcontroller	1
2.	Analog to Digital Converter HX711	1
3.	Air Pressure Sensor MSP40-GDF	1
4.	Transistor-Transistor Logic PL2303	1
5.	Jumper Wires	Some

3.3 Hardware Description

3.3.1 Analog-to-Digital Converter HX711

Figure 3.1 Pinout of HX711 IC (Analog-to-Digital Converter [ADC])

The HX711 module is a Load Cell Amplifier breakout board for the HX711 IC that allows you to easily read load cells to measure weight. This module uses **24 high-precision A/D converter chip HX711**. With two analog input channels, as well as the internal integration of 128 times the programmable gain amplifier, it is designed for high precision electronic scale design. The input circuit can be configured to provide a bridge type pressure bridge (such as pressure, weighing sensor mode), is of high precision, low cost, and is an ideal sampling front-end module.

The HX711 uses a **two-wire** interface (Clock and Data) for communication. In comparison to other chips, HX711 has several advantages such as high integration, fast response, immunity, and other features that improve performance and reliability. Finally, it's one of the best choices for electronic enthusiasts. The chip lowers the cost of the electronic scale, at the same time, improving performance and reliability.

Differential input voltage: ±40mV (Full-scale differential input voltage is ± 40mV)

- **Data accuracy:** 24 bit (24 bit A / D converter chip.)
- **Refresh frequency:** 10/80 Hz
- **Operating Voltage:** 2.7V to 5VDC
- **Operating current:** <10 mA

3.3.2 Air Pressure Sensor MSP40-GDF

Figure 3.2 Air Pressure Sensor MSP40-GDF with range 0-5.8 psi

Pressure range: 40kpa (differential pressure)

Product Features:

- Solid, MEMS technology, high reliability
- Low cost Wide monitoring and control media
- The application of a wide temperature range
- Easy to use, choose from a variety of range. Easy to use, easy to install in OEM equipment

Application areas:

- Automotive: tire pressure, car air pump, MAP sensor, diagnostic equipment, automotive sensors.
- Industry: Air brake switch, portable pressure gauge, such as digital pressure gauge, environmental monitoring, consumer and sports
- Health care: patient monitoring and diagnostic equipment, such as blood pressure monitors, medical instrumentation and monitoring

3.4 Pinout Connections

	ADC HX711	-	**STM32F103C8T6**
1.	+5V	-	Vin
2.	GND	-	GND
3.	DT	-	A3
4.	SCK	-	A4

	MSP40-GDF	-	ADC HX711
1.	1	-	(A-)
2.	2	-	(E+)
3.	4	-	(A+)
4.	5	-	(A+ or E-)
5.	6	-	(E-)
6.	7	-	(A-)

3.5 STM32F103C8T6 Data Accessing Code

```
#include <Q2HX711.h>

const byte hx711_data_pin = 3;

const byte hx711_clock_pin = 4;

Q2HX711 hx711(hx711_data_pin, hx711_clock_pin);

void setup () {

  Serial.begin(115200);

}

void loop () {

  float x = hx711.read();

  Serial.print(x);

  float y = 5.5*x/2147483520;

  Serial.print("     =     ");

  Serial.print(y);

  Serial.print("     =     ");

  Serial.println((y * 101.325)/0.3);

  delay (200);

}
```

3.6 Conclusion

After Hardware Connections and successful upload of the program, open the Serial Monitor at the baud rate of 115200 on the COM attached with TTL. We can see the Air Pressure Value on the Serial Monitor Based on the Varying containers/cylinders/environment's Air Pressure. This project will help the company a lot in measuring the air pressure of their future projects.

Chapter 4 BATTERY DRAINAGE MONITORING SYSTEM

4.1 Overview

In this project, we had built a Battery Status Monitoring System**.** This project is a part of larger project which will be integrated to a product at a later stage.

As we know, the battery in any system or device is the main component because it powers the entire system. Hence, we need to monitor the voltage level of the battery. We all know that an improper system of charging and discharging may lead to battery damage or system failure. Most of the electrical/electronic devices have a Battery Management System (BMS).

Actually, BMS monitors all the properties of the battery like the voltage, current, temperature & auto cut-off system. To ensure the proper safety and handling of Lithium-Ion or Lithium Polymer batteries.

In this IoT-based Battery Monitoring System, we will use the Arduino UNO board to send the battery status data to the Arduino Serial Monitor. The Arduino Serial Monitor will display the battery voltage along with the battery percentage in both the charging and discharging conditions. As we change the voltage transfer using the potentiometer, we are able to see that the status of battery percentage is getting changed. Thus, we can monitor the status of battery.

4.2 Components Used

Sr. No	Component	Quantity
1.	Arduino UNO	1
2.	Potentiometer	1
3.	9V Battery	2
4.	Multimeter	3
5.	Connectors	2
6.	Jumper Wires	Some
7.	22k ohm resister	1
8.	6.8k ohm resister	1

4.3 Circuit Diagram

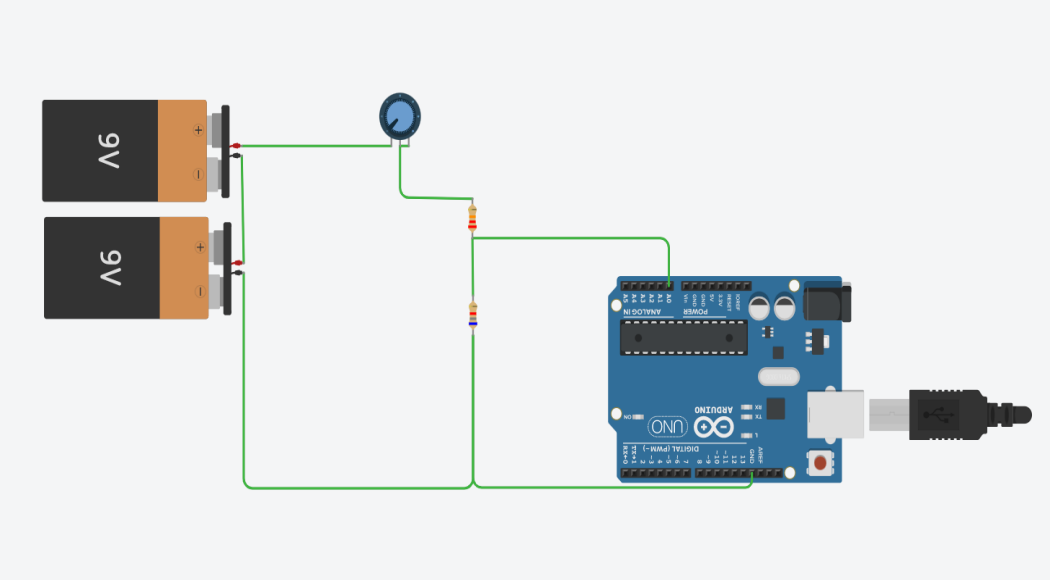

Figure 4.1 Hardware Configuration of BMS

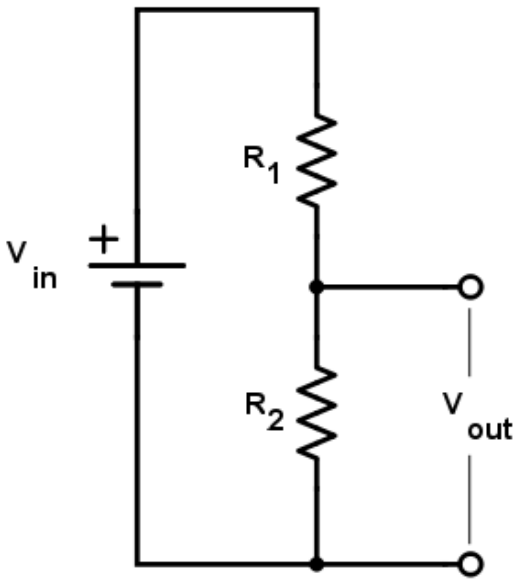

Figure 4.2 Voltage Divider Circuit

4.4 Connections

1.	+18V of Battery	-	Vcc of Potentiometer
2.	GND of Pot	-	Wiper/Input of Pot
3.	Wiper/Input of Pot	-	22k ohm resistor

4.	22k ohm resistor	-	• A0 of Arduino Uno • 6.8k ohm resistor
5.	6.8k ohm resistor	-	GND of Battery

4.5 Source Code

```
// C++ code
//
int analogInPin = A0;
float sensorValue;


void setup ()
{
Serial.begin(9600);
        delay (500);
        //pinMode(LED_BUILTIN, OUTPUT);
}


void loop ()
{
        sensorValue= analogRead(analogInPin);
        float voltage = ((sensorValue*5)/1023) *4.23;


//2.8V as Battery Cut off Voltage & 4.2V as Maximum Voltage
int bat_percentage = mapfloat (voltage, 13.34, 17.95, 0, 100);
```

```
if (bat_percentage >= 100)
{
bat_percentage = 100;
}

if (bat_percentage <= 0)
{
bat_percentage = 1;
}
Serial.print("Analog Value = ");
Serial.print(sensorValue);
Serial.print("\t Output Voltage = ");
Serial.print(voltage);
Serial.print("\t Battery Percentage = ");
Serial.println(bat_percentage);
delay (1000);
}
float mapfloat (float x, float in_min, float in_max, float out_min, float out_max)
{
return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
```

4.6 Conclusion

After successful upload of the program, open the Serial Monitor at the baud rate of 9600. So, this is how we made an IoT Based Battery Status Monitoring System using Arduino Uno and TinkerCAD at Electrogenics Innovations Pvt Ltd. This project will help the company a lot in monitoring the battery status of their future projects.

CONCLUSION

My internship Experience:

"I was intrigued by the family culture at my company after speaking to some of my seniors who had gone through the summer internship programme. As my company is the small professional services network in the India, I felt that an internship at my company was a great platform for me to gain deeper insights into the accounting and audit industry. My Company is also known for its focus on diversity as well as career development programmes which I believe provide numerous learning opportunities for the interns.

What I like about my internship experience

I like how interns are treated just like full-time associates and get assigned actual and meaningful tasks which are crucial and beneficial to the industry. It feels great to be a part of the actual IoT development team! The most memorable part of my internship was when my engagement team completed our project for a major client. Even though I only joined the team for the final two weeks, I still felt a strong sense of achievement after the signing-off of my first engagement. The best part of the internship was to work on a project with the team. The time spent working on a project with other interns from various departments for the Mentor Appreciation Day Innovation Challenge, as well as other corporate events, form some of the most memorable parts of my internship!"

Challenges that I met during my internship

Being in the Research and Development department, I handled specialised task which demanded skill sets which had not been taught in the general classroom curriculum in college. Hence, this was challenging at times when I was faced with new problems which I had not seen before. However, the seniors and mentors were always ready and willing to give their advice and guidance to help me overcome these challenges."

What I learnt from my internship

I gained deeper insights into the software industry and it has helped me significantly with my career planning. I developed a better understanding of the career path of internship, helping me to make an informed career decision. My biggest takeaway would be the new long-term relationships and connections which I had forged with my colleagues and fellow interns at my company."

My advice for internship

i. Always take initiative in everything that you do.
ii. Make a log of everything which you have learnt at the end of each day of internship.
iii. Talk to new people and make connections, be it managers, seniors or fellow interns.

REFERENCES

1. Program STM32 Blue Pill (STM32F103C8T6) with Arduino IDE. Retrieved July 15, 2022, from https://www.sgbotic.com/index.php?dispatch=pages.view&page_id=48
2. How to use ADC in STM32F103C8 STM32 Microcontroller Board. Retrieved July 15, 2022, from https://circuitdigest.com/microcontroller-projects/how-to-use-adc-in-stm32f103c8-stm32-blue-pill-board
3. How to Program the STM32 "Blue Pill" with Arduino IDE - Maker Pro. Retrieved July 15, 2022, from https://maker.pro/arduino/tutorial/how-to-program-the-stm32-blue-pill-with-arduino-ide
4. HX711 Modulo Conversor Analogico Digital 24Bits Sensor Peso. Retrieved July 15, 2022, from https://solectroshop.com/es/sensores-de-presion/1060-hx711-modulo-conversor-analogico-digital-24bits-sensor-peso-carga.html
5. Battery Level Monitor Using an Arduino | Arduino | Maker Pro. Retrieved July 15, 2022, from https://maker.pro/arduino/projects/battery-level-monitor-using-an-arduino