Project Report

On

Sensor Anomaly Detection

Submitted
In partial fulfillment
For the award of the Degree of

PG-Diploma in Embedded Systems and Design (PG-DESD)

C-DAC , ACTS (Pune)

Guided by

Mr. Sripad Deshpande

SubmittedBy:

Mr. AKHILESH SRIVASTAVA	230940130004
Mr. AKHILESH YADAV	230940130005
Mr. ANGAD CHAUHAN	230940130011
Mr. PRADEEP KUMAR	230940130039
Mr. SHASHANK GIRI	230940130058

Centre for Development of Advanced Computing (C-DAC), ACTS

Acknowledgement

This is to acknowledge our indebtedness to our Project Guide **Mr. Sripad Deshpande** C-DAC ACTS, Pune for other constant guidance and helpful suggestion for preparing his project **SENSOR ANOMALY DETECTION.** We express our deep gratitude towards her for inspiration, personal involvement, constructive criticism that she provided us along with technical guidance during the course of this project.

We take this opportunity to thank Head of the department **Mr. Gaur Sundar** for providing us such a great infrastructure and environment for our overall development.

We express thanks to **Mrs. Namrata Ailwar** for their kind cooperation and extendable support towards the completion of our project.

It is our great pleasure in expressing sincere and deep gratitude towards **Mrs. Risha P R (Program Head)** and **Mrs. Srujana Bhamidi** (Course Coordinator, PG-DESD) for their valuable guidance and constant support throughout this work and help to pursue additional studies.

Also, our warm thanks to **C-DAC ACTS Pune**, which provided us this opportunity to carry out, this prestigious Project and enhance our learning in various technical fields.

AKHILESH SHRIVATVA	230940130004
AKHILESH YADAV	230940130005
ANGAD CHAUHAN	230940130011
SHASHANK GIRI	230940130058
PRADEEP KUMAR	230940130039

ABSTRACT

There has been increasing interest in cooperating sensing systems into objects or the environment for monitoring purposes. In this work we compare approaches to performing fully-distributed anomaly detection as a means of detecting security threats for objects equipped with sensing and communication ability. In our project we used four sensors (temperature, pressure, smog and vibration sensor). We gave conclude result after co-relating sensors. We used STM32F407VGT6 micro-controller board, and ESP32 board.

All four sensors connected on STM32 board and send the data to ESP32 via CAN BUS, after receiving the data on ESP32 board, we send the data at cloud by using with the help of MQTT. We are using Thingsboard platform for this observation we use TFT display for observing data. .

Table of Contents

List of Figure

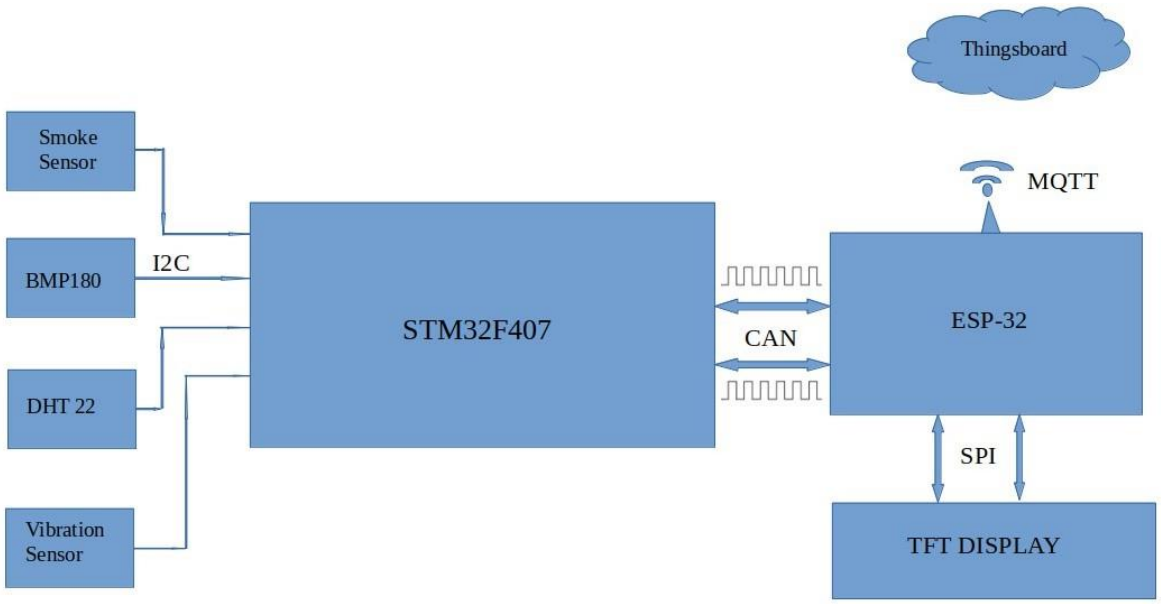
Chapter 1

Introduction

Overview

In industry we often see there is very technology or method used to detect fault and as well as for checking health condition of the component but that was not showing much enough or exact value on basis of real time and environment condition. So for resolving this problem we made Anomaly Sensor Detection method for detecting fault condition and checking health condition of devices. it gives us exact condition of devices in form of data with the help of co-relating different sensors.

1.1 Block Diagram–



1.1 Block Diagram of Project

Hardware Setup

The system is based on Cortex-M3/M4 (STM32f407VG) Microcontroller boards and ESP32 board. The STM board is interfaced with 4 sensors (Temperature , Pressure, smog and vibration sensor).The STM 32 interfacing with BMP180 with i2c bus and other sensor using serial wire . We used CAN bus for sending data which is coming from different sensors on ESP32 BOARD. Apart from this the SPI communication used between TFT display and ESP32 for showing the sensor data . Thingsboard for sending data on cloud by MQTT.

We used co-relating method for giving accurate data after concluding various type of data from different sensors. There is two method for correlation the first one is the simple correlation method and second is z-score method by this method we obtained the conclusion that what should be the temprature ,pressure, humidity, and other essential data for a device or circuit. We sent data of sensors on cloud and stored continuously last 24 hour data of sensors.We used circular queue method for storing that large amount of data in buffer. After each 24 hour data will be overwritten. At last we will find concluding data which help to detect fault of devices and real time condition of devices.

Chapter 2

LITERATURE REVIEW

Anomaly detection has attracted considerable attention from the research community in the past few years due to the advancement of sensor monitoring technologies, low-cost solutions, and high impact in diverse application domains. Sensors generate a huge amount of data while monitoring the physical spaces and objects. These huge collected data streams can be analyzed to identify unhealthy behaviors. It may reduce functional risks, avoid unseen problems, and prevent downtime of the systems. Many research methodologies have been designed and developed to determine such anomalous behaviors in security and risk analysis domains. In this project, we present the results of a systematic literature review about anomaly detection techniques except for these dominant research areas. We focus on the studies published from 2000 to 2018 in the application areas of intelligent inhabitant environments, transportation systems, health care systems, smart objects, and industrial systems. We have identified a number of research gaps related to the data collection, the analysis of imbalanced large datasets, limitations of statistical methods to process the huge sensory data, and few research articles in abnormal behavior prediction in real scenarios. Based on our analysis, researchers and practitioners can acquaint themselves with the existing approaches, use them to solve real problems, and/or further contribute to developing novel techniques for anomaly detection, prediction, and analysis. Proposed system The proposed system is sensor (Temperature, pressure, smog, vibration sensors) based technique for detecting fault and health condition checking for devices which is used in industry. In this system we used STM32F407VGT6 and ESP-WROOM-32E. We concluded data on the basis of co-relating sensors. Main objective The main purpose of the project is as follows... 1. To detect fault and monitoring devices on the basis of real time. 2. We will give real time running condition of devices with exact value.There are two method, we used for correlating sensor data.

1. **Simple condition rule :** In this method we use simple condition rule and found which range of sensor data should run on industry purpose in respect of other sensors data.

We use BMP180 sensor for pressure measuring . First we convert the pascal value to hecto pascal after dividing by 100. Then we find what should be altitude ,temprature range and vibration. We have decided by using this algorithm.

```
void rule_1 ( void )
{
  if ( ( pressure >= 900 ) && ( pressure <= 1100 ) )
  {
    if (temperature < 20 || temperature > 30)
    {
      Serial.println("Error: Temperature anomaly when pressure is between 900 and 1100 hPa\n");
      tft.println("temprature anomaly :");
      tft.println("press b/w 900-1100");
    }
  }

  if ( (pressure < 900) && (pressure > 1000))
  {
   if(altitude > 1000)
   {
    tft.println("Altitude anomaly detected");
   }
  }
```

2. **Z-score method :** With the help of z-score value we detect anomaly based on sensor reading .In this method first we find the mean of sensor ratings and then we calculate variance and standard deviation and with the help of standard deviation and

mean we find z-score value for every sensor reading. After calculating enough z-score value of every sensor reading we decide the threshold value for every sensor (like for temperature, Humidity, pressure etc we calculate different z-score value).And if any sensor crosses its threshold z-score value we find the anomaly in that sensor.

Z score = (x – x̄) / $\sqrt{(\sum(x-\bar{x})}$ (x − x ¯) ^{2} /n)

.

Chapter 3

System Design

This chapter explains in detail about hardware that are being used, their features and applications.

3.1 STM32f407VG Microcontroller

The STM32F4 Discovery board is small devices based on STMF407 ARM microcontroller, which is a high-performance microcontroller. This board allows users to develop and design applications. It has multiple modules within itself which allows the user to communicate and design the interface of different kinds without relying on any third device. The board has all the modern system modules peripherals like DAC, ADC, audio port, UART, etc which makes it one of the best-developing devices. The device may be for developing modern applications but some protocols will need to be followed to use the device, like the compiler, voltage potential, etc. The STM32F407xx family is based on the high-performance ARM® Cortex®-M4 32-bit RISC core with FPU operating at a frequency of up to 72 MHz, and embedding a floating point unit (FPU), a memory protection unit (MPU) and an embedded trace macro cell (ETM). The family incorporates high-speed embedded memories (up to 1 Mbytes of Flash memory, up to 192 Kbytes of RAM) and an extensive range of enhanced I/O ’s and peripherals connected to two APB buses. They also feature standard and advanced communication interfaces: up to two I2Cs, up to three SPIs (two SPIs are with multiplexed full-duplex I2Ss), three USARTs, up to two UARTs, CAN and USB. To achieve audio class accuracy, the I2S peripherals can be clocked via an external PLL. The STM32F303xB/STM32F303xC family operates in the -40 to +85°C and -40 to +105 °C temperature ranges from a 2.0 to 3.6 V power supply. A comprehensive set of power-saving mode allows the design of low-power applications.

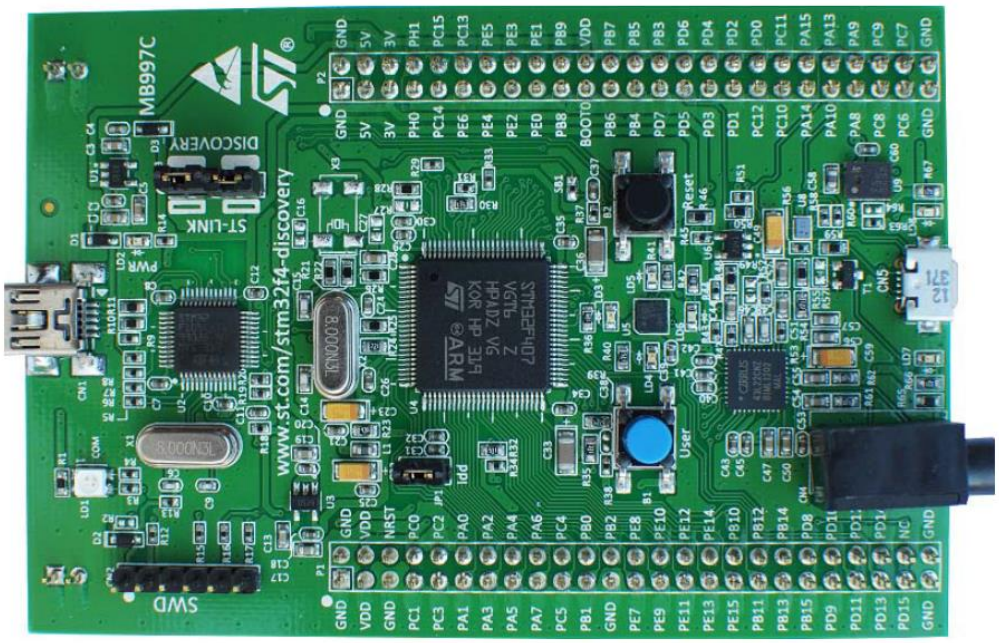


Figure 3.1: STM32f407VG

Microcontroller Board 3.1.1 Power supply Pins This board provides onboard power pins to give power to external sensors or circuits. Power Input: The STM32F4 has multiple power pins are all pins can be used any 5-V power supply to power up the device. The power-up the whole device only pins can be used, power input through USB won't be able to activate all modules of the device. All Power input pin is given below. In P1 Header: VDD – Pin3, Pin4 In P2 Header: 5V – Pin3, Pin4 Power Output: Different power levels have always been a requirement for every device, and to fulfill the requirement there is an internal regulator within the STM32F4. It has two power pin one is 5Volt and the second one is 3V3. 5V pin is directly connected to the power input and 3V3 is connected through the regulator. Both pins are in header P2: 5V – Pin3, Pin4

3V3 – Pin5, Pin6, GPIO15, GPIO16

Ground: In the case of multiple external devices the ground becomes the basic requirement for each device to make the common ground. This device has multiple ground pins in both headers which can be used individually. All of them are connected internally and they are listed below: In P1 Header: GND – Pin1, Pin2, Pin5, Pin23, Pin49, Pin50 In P2 Header: GND – Pin1, Pin2, Pin49, Pin50

3.1.2 STM32F4 Oscillator Pins

STM32F4 may be larger and smarter but it doesn't have any internal crystal for clock pulse. It has four external clock pulse pins two they are used for 32KHz crystal and the other two are used for High-frequency crystal. The crystal can be used up to 50MHz on that pin but after 25MHz it becomes tricky to handle the crystal. These crystal pins can be used as GPIO, so it should be instructed within the program about the use of an oscillator. Both oscillators can't be used at the same time and both are in the same header P2.

- OSC (IN) – GPIO7 ˆ
- OSC (OUT) – GPIO ˆ
- OSC RTC (IN) – GPIO9 ˆ
- OSC RTC (OUT) – GPIO10

3.1.3 STM32F4 Discovery Board GPIO Pins

These are a total of six ports (A, B, C, D, E, H) in the device and all of them come with an internal pull-up resistor and can be used for input/output function. Those pins also support some other functions and it can be controlled using programming. All I/O pins are given below.

In P1 Header: GPIO7 – GPIO22, GPIO24 – GPIO47

In P2 Header: GPIO7 – GPIO21, GPIO23 – GPIO48

3.1.4 STM32F407VG Discovery Board USART Communication Pins

There is multiple serial communication within the STM32F4 and USART is one of them due to its popularity and easiness. The USART uses only two wires and the rest.

of the protocol depends on the controllers. In STM32F4 all USART communications are the same but each one of them should be initialized within the program to avoid any kind of conflict. All USART pins of STM32F4 are given below.

In P1 Header: ˆ

- USART2 TX – GPIO14 ˆ
- USART2 RX – GPIO13 ˆ
- USART3 TX – GPIO34, GPIO40 ˆ
- USART3 RX – GPIO37, GPIO42 ˆ
- USART4 TX – GPIO12 ˆ
- USART4 RX – GPIO1

In P2 Header: ˆ

- USART1 TX – GPIO23, GPIO44 ˆ
- USART1 RX – GPIO24, GPIO41 ˆ
- USART2 TX – GPIO29 ˆ
- USART2 RX – GPIO30 ˆ
- USART3 TX – GPIO37 ˆ
- USART3 RX – GPIO38 ˆ
- USART4 TX – GPIO37 ˆ
- USART4 RX – GPIO38 ˆ
- USART5 TX – GPIO35 ˆ
- USART5 RX – GPIO34 ˆ
- USART6 TX – GPIO47 ˆ

- USART6 RX – GPIO48 1

3.1.5 STM32F4 Discovery Board I2C Channels

There are multiple modules and motors which only follow the I2C communication due to their internal structure. In that case, the device can only communicate with them with I2C protocol and multiple GPIO Pins support this communication only through programming. All these pins are...

In P1 Header: ˆ

- I2C2 SCL – GPIO34 ˆ
- I2C2 SDA – GPIO35

In P2 Header: ˆ

- I2C1 SCL – GPIO19 ˆ
- I2C1 SDA – GPIO20 ˆ
- I2C1 SCL – GPIO23 ˆ
- I2C1 SDA – GPIO24 ˆ
- I2C3 SCL – GPIO43 ˆ
- I2C3 SDA – GPIO46

3.1.6 STM32F4 Discovery Board ADC Pins

There are multiple analogs to digital converter in the STM32F4. All these converters can be used individually with any TTL or ST output device to convert the analog data to digital. And ADC channels only exist in header P1.GPIO7 – GPIO22

STM32F4 Discovery Board DAC Pins

The device also supports two digitals to analog converting channels which convert any digital input to its voltage level. All DAC pins in STM32F4 are in header P1: GPIO15

3.1.7 STM32F4 Audio Input Channel Pins

Audio communication is increasing now a day and STM32F4 supports multiple audio pins. In these audio pins, some of them have an internal D speaker Driver. These audio pins only use to receive the audio from the device, they can't be used for audio in. All audio pins are given below:

In P1 Header: ˆ

- GPIO15 ˆ
- GPIO16 ˆ
- GPI38

In P2 Header only simple Audio exist: ˆ

- GPIO20 ˆ
- GPIO32 ˆ
- GPIO35 ˆ
- GPIO47

3.1.8 MIC:

To send the audio signal to the device a mic pin can be used which allows the user to send the audio by interfacing the mic with the STM32F4. There is only one mic pin within the whole device and it is given in GPIO9

3.1.9 Pulse Width Modulation Channels:

STM32F4 can also generate a PWM signal. In this device, the number of PWM signals is much greater than most of the devices. All PWM pin can be used individually for output signal and all of them are given below:

In Header P1: ˆ

- GPIO11 – GPIO18 ˆ
- GPIO21, GPIO22 ˆ
- GPIO26 – GPIO32 ˆ
- GPIO34-GPIO35 ˆ

- GPIO37–GPIO39
- GPIO44 – GPIO47

In header P2: ˆ

- GPIO11 ˆ
- GPIO14 ˆ
- GPIO19 ˆ
- GPIO20 ˆ
- GPIO23 – GPIO26 ˆ
- GPIO28 ˆ GPIO41 ˆ
- GPIO43 – GPIO48

3.1.10 MEMS Sensor Interface Channels:

The devices support the electro-mechanical sensors interfacing with itself. It can use the MEMS pins to connect those devices with itself. All those pins are listed below:

In P1 Header: ˆ

- GPIO15 ˆ
- GPIO17 ˆ
- GPIO18
- In P2 Header: ˆ
- GPIO16 ˆ
- GPIO17 ˆ
- GPIO18

3.1.11 RESET:

STM32F4 also has a reset pin in header P1 which can be used externally to reset the device by giving a pulse: NRST – Pin6

3.1.12 Features ˆ

STM32 microcontroller with LQFP64 package. ˆ

- Two types of extension resources. ˆ
- mbed-enabled (mbed.org) ˆ
- On-board ST-LINK/V2-1
- debugger/programmer with SWD connector. ˆ
- Flexible board power supply. ˆ
- Three LEDs. ˆ
- Two push buttons: USER and RESET. ˆ
- USB re-enumeration capability: three different interfaces supported on USB, etc.

3.1.13 Application ˆ

- The device is being widely used for developing projects like robots or cars ˆ
- Developers also like to use the SMT32F4 in industrial projects due to its multifunctional. ˆ
- It is used mostly in GPS logger and voice communication projects.

3.2 Controller Area Network:

The Controller Area Network bus is an International Standardization Organization (ISO) defined serial communications bus originally developed for the automotive industry to replace the complex wiring harness with a two-wire bus. The Controller Area Network bus, or CAN bus, is a vehicle bus standard designed to allow devices and microcontrollers to communicate with each other within a vehicle without a host computer. CAN is a reliable, real-time protocol that implements a multicast, data-push, publisher/subscriber model. CAN messages are short (data payloads are a maximum of 8 bytes, headers are 11 or 29 bits), so there is no centralized master or hub to be a single point of failure and it is flexible in size. Its real-time features include deterministic message delivery time and global priority through the use of prioritized message IDs. Bus arbitration is accomplished in CAN using bit dominance, a process

where nodes begin to transmit their message headers on the bus, then drop out o f the "competition" when a dominant bit is detected on the bus, indicating a message ID

of higher priority being transmitted elsewhere. This means bus arbitration does not add overhead because once the bus is "won" the node simply continues sending its message. Because there is no time lost to collisions on a heavily loadednetwork.CAN is ideal for periodic traffic. Lastly, CAN's reliability features are suited for typically harsh embedded environments. 14 CAN was designed for a typical embedded system with periodic, busty traffic, where every node transmits at regular intervals. Ethernet, on the other hand, was designed for aperiodic, light traffic and allow number of active transmitters. More significantly, CAN was designed to achieve tight, real-time schedules, unlike Ethernet. 3.2.1 CAN Network: A CAN network consists of a number of CAN nodes which are linked via a physical transmission medium (CAN bus). In practice, the CAN network is usually based on a Line Topology with a linear bus to which a number of electronic control units are each connected via CAN interface. The passive star topology may be used as an alternative. An unshielded twisted two-wire line is the physical transmission medium used most frequently in applications (Unshielded Twisted Pair — UTP), over which Symmetrical Signal transmission occurs. The maximum data rate is 1 Mbit/s . A maximum network extensionofabout40metersisallowed.AttheendsoftheCANnetwork, BusTerminatio n Resistors contribute to preventing transient phenomena (reflections). ISO 11898 specifies the maximum number of CAN nodes as 32.

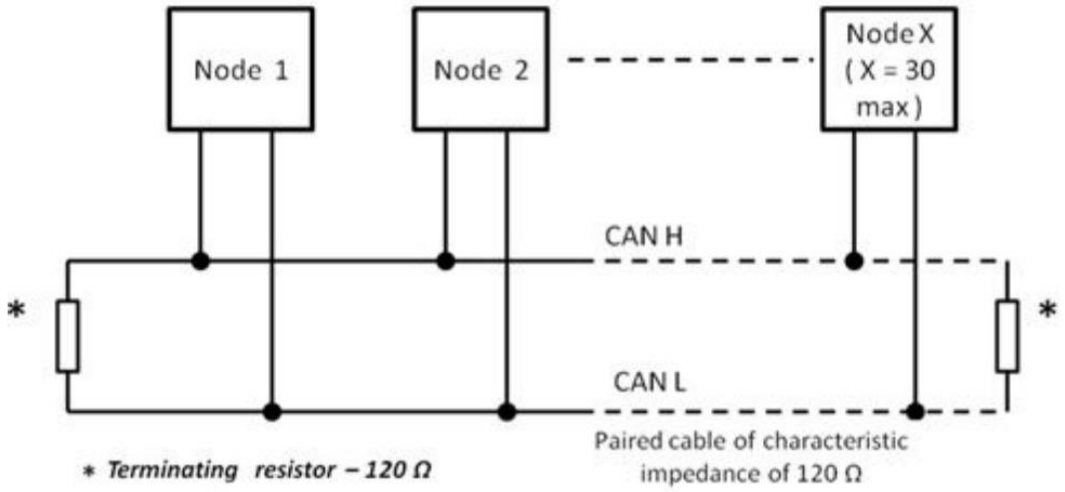


Figure 3.2 : CAN Controller

3.2.2 CAN Bus:

Physical signal transmission in a CAN network is based on transmission of differential voltages (differential signal transmission). This effectively eliminates the negative effects of interference voltages induced by motors, ignition systems and switch contacts. Consequently, the transmission medium (CAN bus) consists of two lines: CAN High and CAN Low.

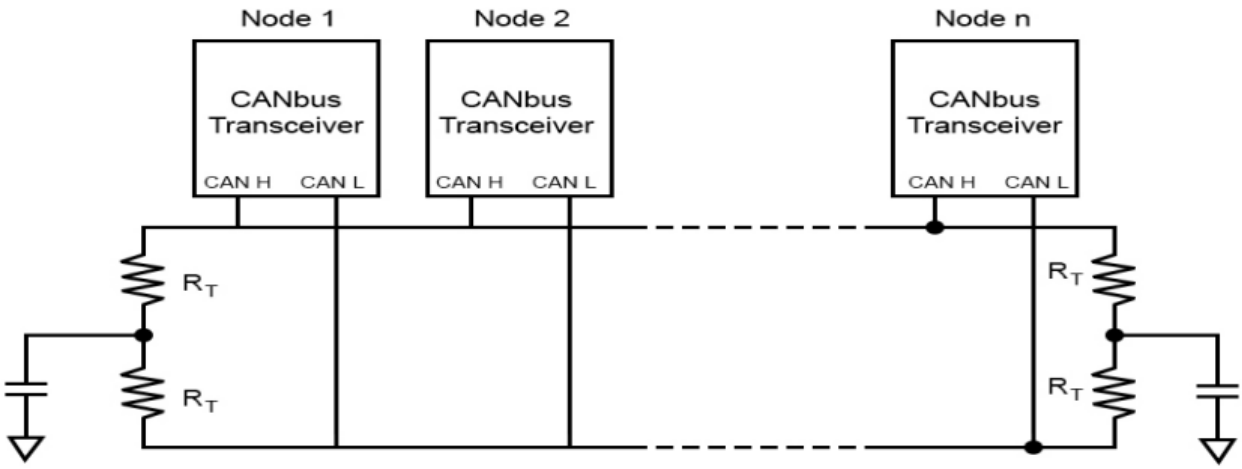


Figure 3.3 CAN BUS

Twisting of the two lines reduces the magnetic field considerably. Therefore, in practice twisted pair conductors are generally used as the physical transmission medium. Due to finite signal propagation speed, the effects of transient phenomena (reflections) grow with increasing data rate and bus extension. Terminating the ends of the communication channel using termination resistors (simulation of the electrical properties of the transmission medium) prevents reflections in a high-speed CAN network. The key parameter for the bus termination resistor is the so called characteristic impedance of the electrical line. This is 120 Ohm. In contrast to ISO 11898- 2, ISO 11898-3 (low-speed CAN) does not specify any bus termination resistors due to the low maximum data rate of 125 Kbit/s.

3.2.3 CAN Bus Levels: Physical signal transmission in a CAN network is based on differential signal transmission. The specific differential voltages depend on the bus interface that is used. A distinction is made here between the high-speed CAN bus interface (ISO 11898-2) and the low-speed bus interface

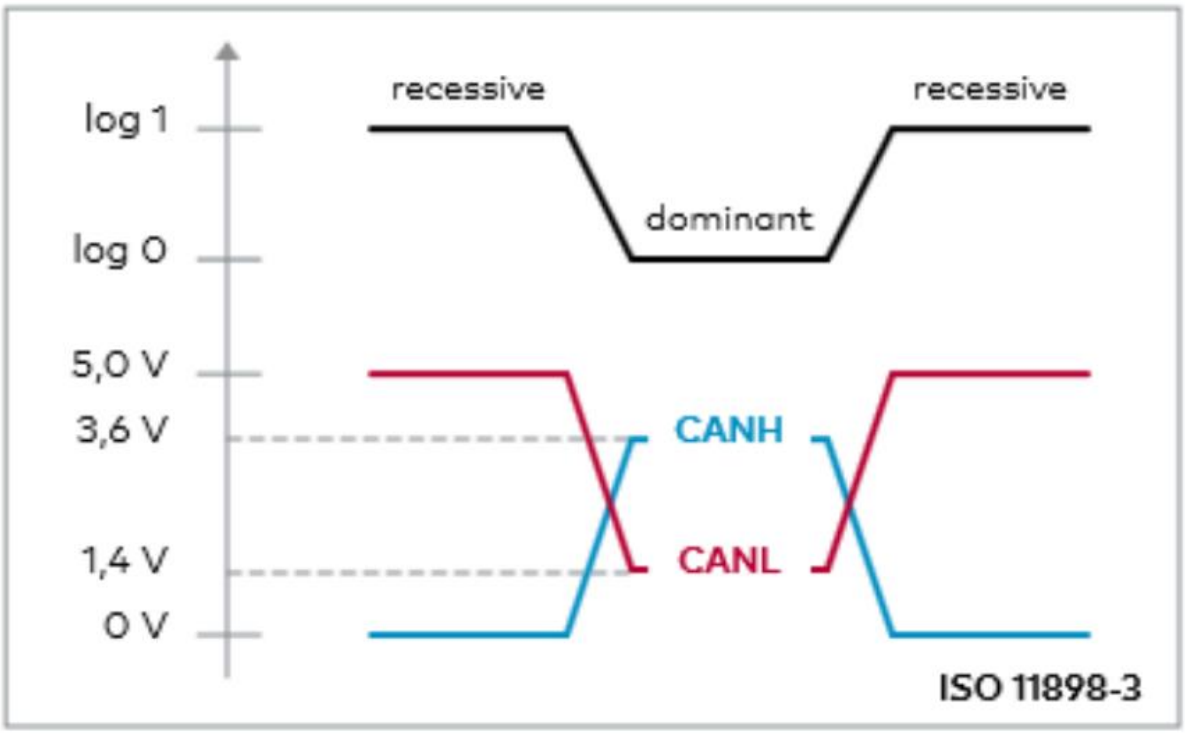


Figure 3.4 : CAN BUS LEVELS

transceivers interpret a differential voltage of more than 0.9 Volt as a dominant level within the common mode operating range, typically between 12 Volt and -12 Volts. Below 0.5 Volt, however, the differential voltage is interpreted as a recessive level. A hysteresis circuit increases immunity to interference voltages. ISO 11898-3 assigns a typical differential voltage of 5 Volt to logical "1", and a typical differential voltage of 2 Volt corresponds to logical "0". The figure "High-Speed CAN Bus Levels" and the figure "Low -Speed CAN Bus Levels" depict the different voltage relationships on the CAN bus.

3.2.4 CAN Layers:

Data Link Layer: Most of the CAN standard applies to the transfer layer. The transfer layer receives messages from the physical layer and transmits those messages to the object layer. The transfer layer is responsible for bit timing and synchronization, message framing, arbitration, acknowledgement, error detection and signaling, and

fault confinement. It performs....

- Error Detection ˆ
- Message Validation
- ˆ Acknowledgement ˆ
- Arbitration ˆ
- Message Framing

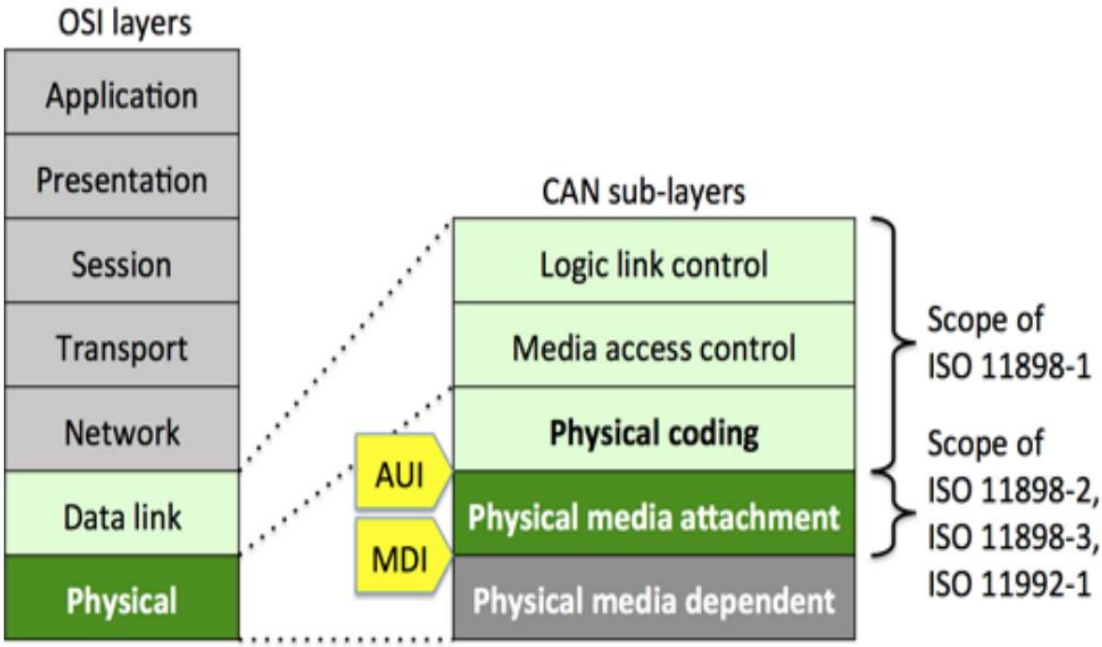


Figure 3.5 CAN Layers

Physical Layer:

CAN bus specify the link layer protocol with only abstract requirements for the physical layer. As a result, implementation often employs custom connector with various sorts of cables, of which two are the CAN bus lines. Noise immunity is achieved by maintaining the differential impedance of the bus at a low level with low-value resistors (120 ohms) at each end of the bus.

3.2.5 CAN Frames

CAN frames are as follows....

1. **DataFrame** The data frame is the most common message type, and comprises the Arbitration Field, the Data Field, the CRC Field, and the Acknowledgment Field. The Arbitration Field contains an 11-bit identifier and the RTR bit, which is dominant for data frames. Next is the Data Field which contains zero to eight bytes of data, and the CRC Fieldwhich contains the 16-bit checksum used for error detection. Last is the Acknowledgment Field.

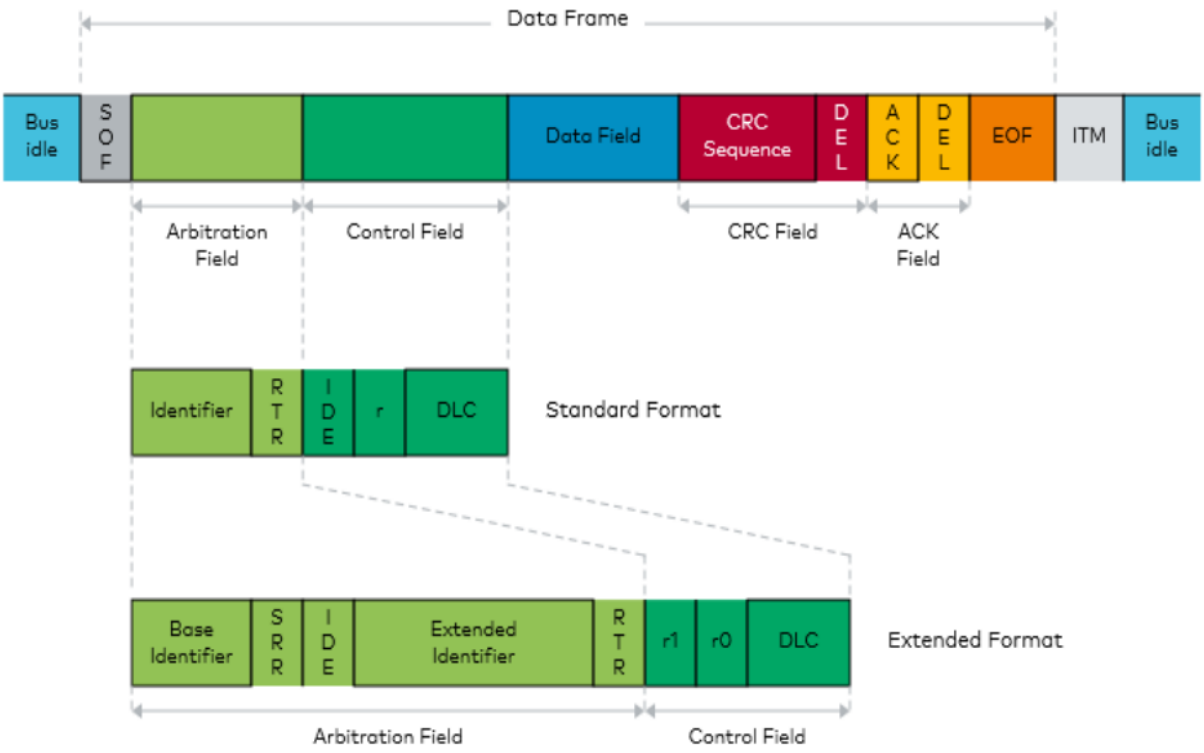


Figure 3.6 : CAN Standard Data Frame

2. Overload Frame

This frame Consists of two bit fields

- ˆ overload flag ˆ
- overload delimiter

Three different overload conditions lead to the transmission of the overload frame are....

- ˆ Internal conditions of a receiver require a delay of the next data frame or remote frame. ˆ At least one node detects a dominant bit during intermission. ˆ
- A CAN node samples a dominant bit at the eighth bit (i.e., the last bit) of an error delimiter or overload delimiter

3. Remote Frame

The intended purpose of the remote frame is to solicit the transmission of data from another node. The remote frame is similar to the data frame, with two important differences. First,this type of message is explicitly marked as a remote frame by a recessive RTR bit in the arbitration field, and secondly, there is no data.

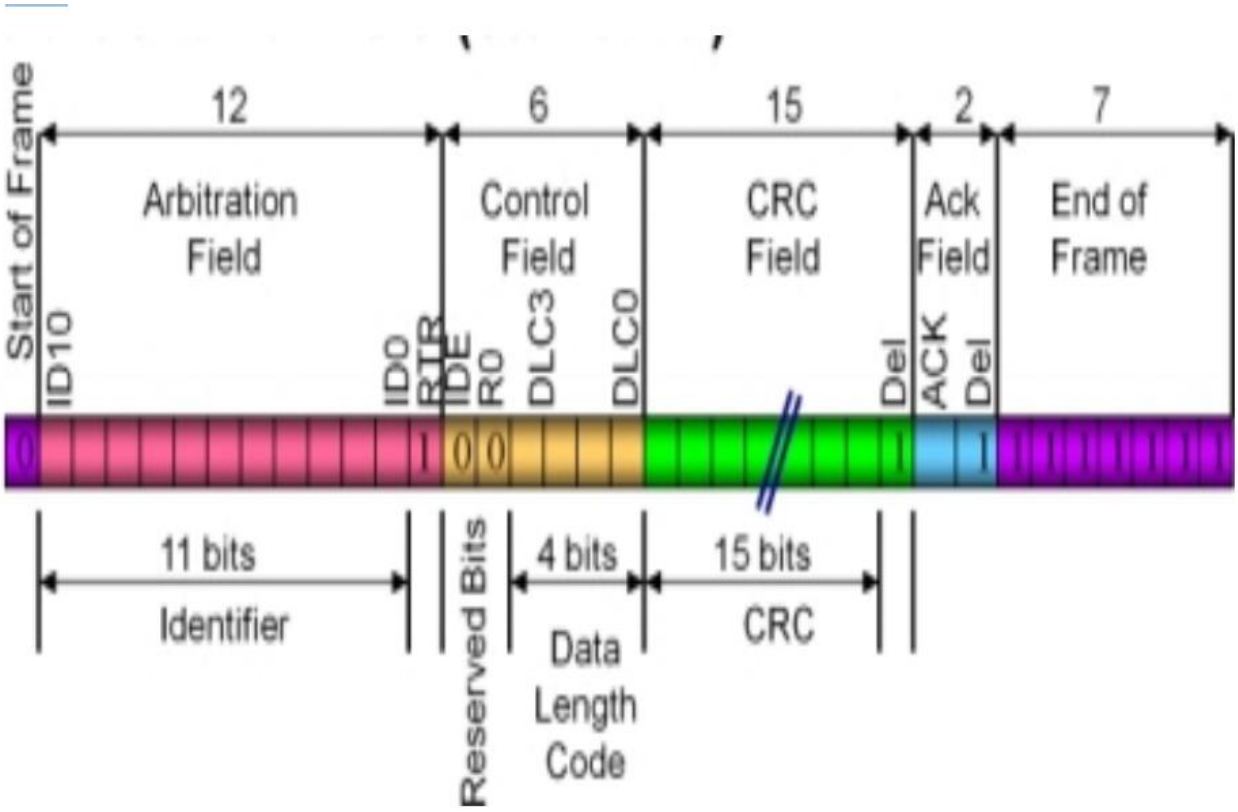


Figure 3.7 : CAN Remote

4. Error Frame

The error frame is a special message that violates the formatting rules of a CAN message. It is transmitted when a node detects an error in a message, and causes all other nodes in the network to send an error frame as well. The original transmitter then automatically retransmits the message. An elaborate system of error counters in the CAN controller ensures that a node cannot tie up a bus by repeatedly transmitting error frames.

Bus Arbitration:

The CAN communication protocol is a carrier-sense, multiple-access protocol with collision detection and arbitration on message priority (CSMA/CD+AMP). CSMA means that each node on a bus must wait for a prescribed period of inactivity before attempting to send a message. CD+AMP mean that collisions are resolved through.

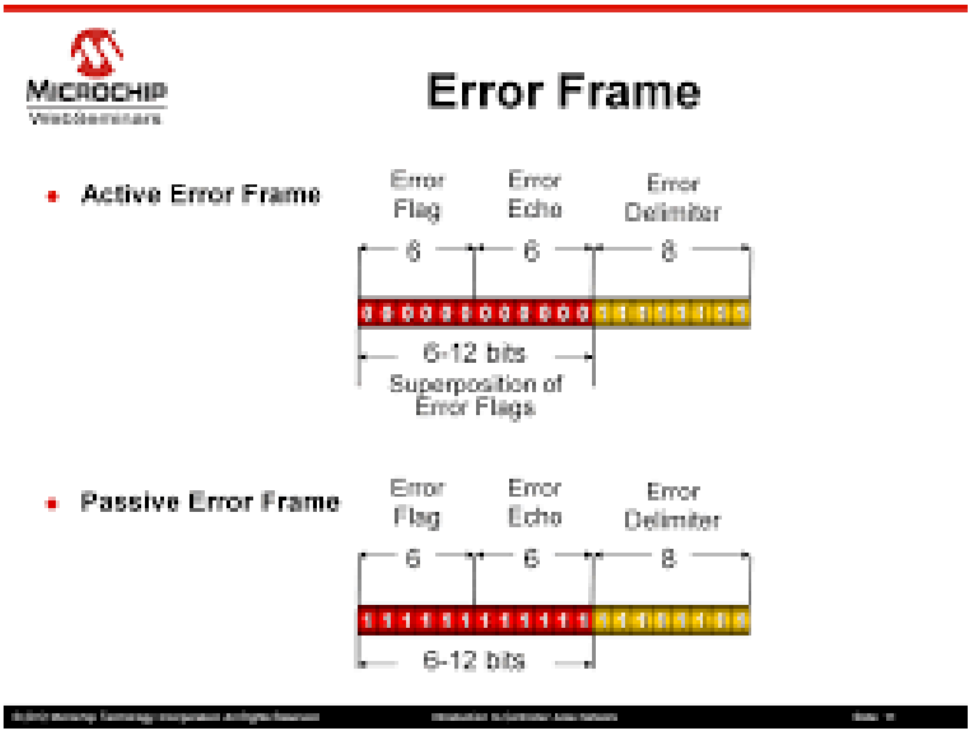


Figure 3.8: CAN Error Frame

a bit-wise arbitration, based on a pre-programmed priority of each message in the identifier field of a message. The higher priority identifier always wins bus access. That is, the last logic-high in the identifier keeps on transmitting because it is the highestpriority.Whenever the bus is free, any unit may start to transmit a message. If two or more units start transmitting messages at the same time, the bus access conflict is resolved by bitwise arbitration using the Identifier. The mechanism of arbitration guarantees that neither information nor time is lost. If a data frame and a remote frame with the same identifier are initiated at the same time, the data frame prevails over the remote frame. During arbitration every transmitter compares the level of the bit transmitted with the level that is monitored on the bus.

If these levels are equal the unit may continue to send. When a 'recessive' level is sent and a 'dominant' level is monitored, the unit has lost arbitration and must withdraw without sending one more bit.

2.2 ESP32-WROOM32-E

Introduction :

The ESP32 WROOM-32E is a versatile and powerful module built around Espressif's ESP32 chipset. It offers dual-core processing, integrated Wi-Fi and Bluetooth connectivity, and boasts a wide range of peripheral interfaces. Known for its low-power consumption, the module is ideal for IoT applications, enabling smart connectivity and robust performance in compact form factors.

The ESP32-WROOM-32E is a powerful and versatile microcontroller module developed by Espress if Systems. It is based on the ESP32 series, which is a family of low-cost, low-power system-on-chip (SoC) microcontrollers with integrated Wi-Fi and Bluetooth capabilities. The ESP32-WROOM-32E module specifically features the ESP32-D0WDQ6 chip.

Feature : Here are some key features of the ESP32-WROOM-32E:

- **Dual-Core Processor:** The ESP32-WROOM-32E is equipped with a dual-core Tensilica Xtensa LX6 processor, providing efficient multitasking capabilities.
- **Wireless Connectivity:** It supports both Wi-Fi 802.11 b/g/n and Bluetooth 4.2/BLE, making it suitable for a wide range of IoT (Internet of Things) applications.

Peripheral Interfaces: The module includes a variety of peripheral interfaces such as SPI, I2C, UART, I2S, PWM, and GPIOs, allowing for seamless integration with various sensors, displays, and other devices.

- **Memory:** It comes with 520 KB SRAM and up to 16 MB Flash memory, providing sufficient space for program storage and data.
    - **Ultra-Low Power Consumption:** The ESP32 series is known for its low-power capabilities, making it suitable for battery-powered and energy-efficient applications.
    - **Security Features:** The ESP32-WROOM-32E incorporates security features such as hardware-based encryption, secure boot, and secure flash.
    -

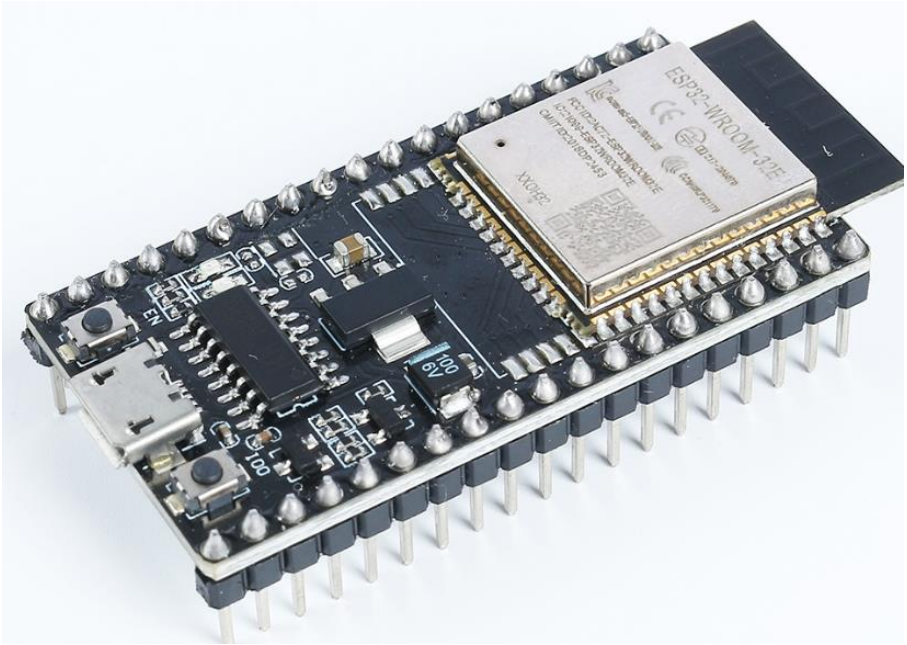


Figure 3.9 ESP32-WROOM-32E

- **Development Environment:** Espressif provides an open-source development framework called ESP-IDF (Espressif IoT Development Framework) for programming the ESP32-WROOM-32E. Additionally, it can be programmed using the Arduino IDE, PlatformIO, and other popular development environments.

- **Versatility:** Due to its capabilities and features, the ESP32-WROOM-32E is widely used in various applications, including home automation, smart devices, industrial automation, and more.
- **CAN BUS Support :** The ESP32-WROOM-32E has a CAN bus port with a transceiver. It also has an integrated CAN controller, so you don't need an external controller.

Pinout Diagram : The ESP32 has some pin usage limitations due to various functionalities sharing certain pins. When designing a project, it's a good practice to carefully plan the pin usage and cross-check for potential conflicts to ensure proper functioning and avoid issues.

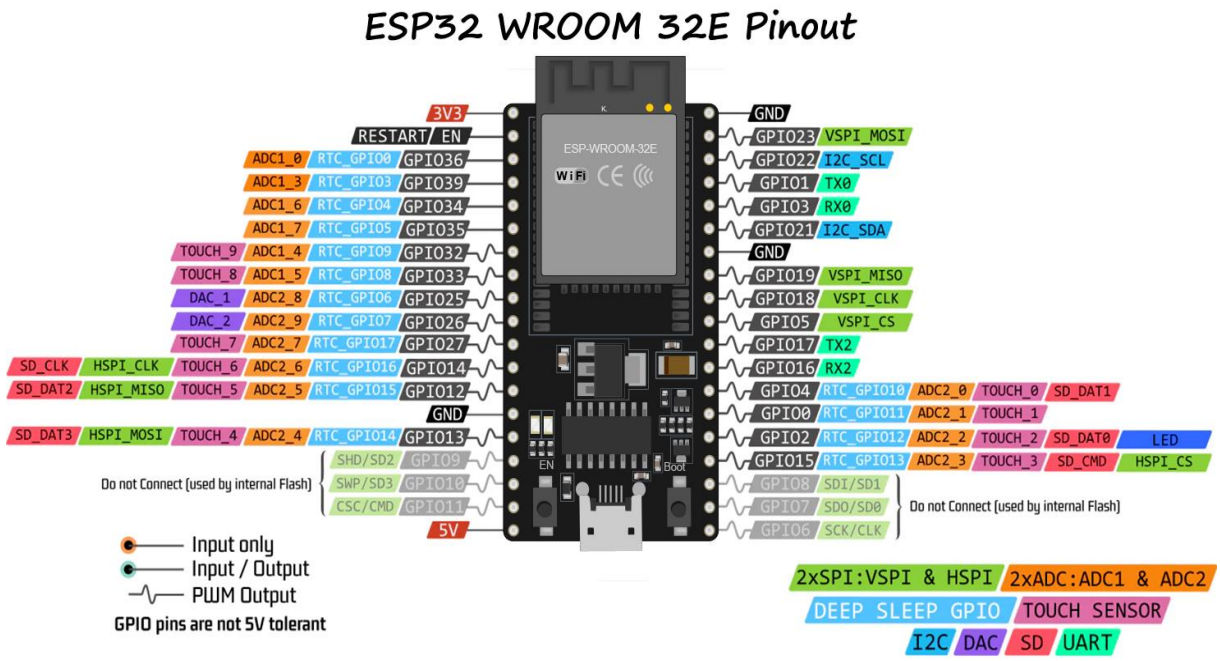


Figure 3.10 ESP32-WROOM-32E Pin Diagram

- **ADC1 and ADC2**: ADC2 cannot be used when WiFi or Bluetooth is active. However, ADC1 can be used without any restrictions.
- **Bootstrapping Pins**: GPIO0, GPIO2, GPIO5, GPIO12, and GPIO15 are used for bootstrapping during the boot process. Care should be taken not to connect external components that could interfere with the boot process on these pins.

Pin Definitions

- **JTAG Pins**: GPIO12, GPIO13, GPIO14, and GPIO15 can be used as JTAG pins for debugging purposes. If JTAG debugging is not required, these pins can be used as regular GPIOs.
- **Touch Pins**: Some pins support touch functionalities. These pins should be used carefully if you intend to use them for touch sensing.
- **Power Pins**: Some pins are reserved for power-related functions and should be used accordingly. For example, avoid drawing excessive current from power supply pins like 3V3 and GND.
- **Input-only Pins**: Some pins are input-only and should not be used as outputs.

Strapping Pins¶

ESP32 has five strapping pins:

Strapping Pin	Description
IO5	Defaults to pull-up, the voltage level of IO5 and IO15 affects the Timing of SDIO Slave.
IO0	Defaults to pull-up, if pulled low, it enters download mode.
IO2	Defaults to pull-down, IO0 and IO2 will make ESP32 enter download mode
IO12(MTDI)	Defaults to pull-down, if pulled high, ESP32 will fail to boot up normally.
IO15(MTDO)	Defaults to pull-up, if pulled low, debug log will not be visible. Additionally, the voltage

	level of IO5 and IO15 affects the Timing of SDIO Slave.

Software can read the values of these five bits from register “GPIO_STRAPPING”. During the chip’s system reset release (power-on-reset, RTC watchdog reset and brownout reset), the latches of the strapping pins sample the voltage level as strapping bits of “0” or “1”, and hold these bits until the chip is powered down or shut down. The strapping bits configure the device’s boot mode, the operating voltage of VDD_SDIO and other initial system settings.

Each strapping pin is connected to its internal pull-up/pull-down during the chip reset. Consequently, if a strapping pin is unconnected or the connected external circuit is high-impedance, the internal weak pull-up/pull-down will determine the default input level of the strapping pins.

To change the strapping bit values, users can apply the external pull-down/pull-up resistances, or use the host MCU’s GPIOs to control the voltage level of these pins when powering on ESP32.

After reset release, the strapping pins work as normal-function pins. Refer to following table for a detailed boot-mode configuration by strapping pins.

3.4 Smog Sensor

Introduction¶

The MQ-2 sensor is a versatile gas sensor capable of detecting a wide range of gases including alcohol, carbon monoxide, hydrogen, isobutene, liquefied petroleum gas, methane, propane, and smoke. It is popular among beginners due to its low cost and easy-to-use features.

Figure 3.11 MQ2 Smoke Sensor

Working Principle¶

The MQ-2 sensor works on the principle of resistance changes in the presence of different gases. When the target gas comes in contact with the heated MOS(Metal Oxide Semiconductor) material, it undergoes oxidation or reduction reactions that change the resistance of the MOS material. It is noteworthy that the MQ2 gas sensor is capable of detecting multiple gases, but lacks the ability to differentiate between them. This is a common characteristic of most gas sensors.

The sensor has a built-in potentiometer that allows you to adjust the sensor digital output (D0) threshold. When the concentration of gas in the air exceeds a certain threshold value, the resistance of the sensor changes. This change in resistance is then converted into an electrical signal that can be read by an Arduino board.

Usage¶

Hardware components

- Arduino Uno R4 or R3 board * 1
- Gas Sensor Module(MQ2) * 1
- Jumper Wires

Pin Diagram

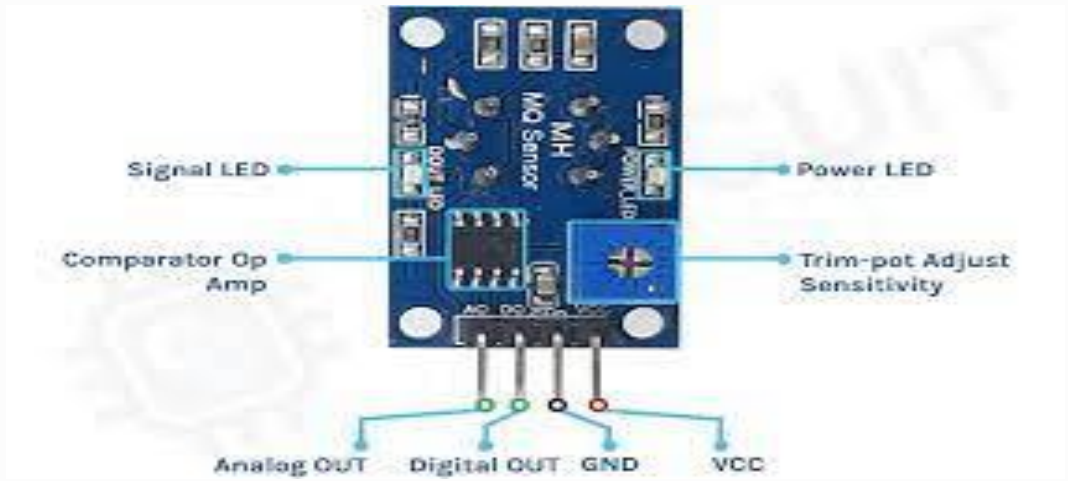


Figure 3.12 MQ2 Smoke Sensor Pin Diagram

3.5. BMP 180 Sensor

BMP180 general description The BMP180 is the function compatible successor of the BMP085, a new generation of high precision digital pressure sensors for consumer applications. The ultra-low power, low voltage electronics of the BMP180 is optimized for use in mobile phones, PDAs, GPS navigation devices and outdoor equipment. With a low altitude noise of merely 0.25m at fast conversion time, the BMP180 offers superior performance. The I2C interface allows for easy system integration with a microcontroller. The BMP180 is based on piezo-resistive technology for EMC robustness, high accuracy and linearity as well as long term stability. Robert Bosch is the world market leader for pressure sensors in automotive applications. Based on the experience of over 400 million pressure sensors in the field, the BMP180 continues a new generation of micro-machined pressure sensors.

Figure 3.13 BMP180 Sensor

Feature of BMP180 Sensor

The BMP180 sensor is a barometric pressure sensor designed to measure atmospheric pressure and temperature. It is manufactured by Bosch Sensor tec and is commonly used in various applications, including weather stations, altitude measurement, and drones. Here's a brief explanation of its key features and functionality:

1. **Pressure Measurement:**

- The BMP180 sensor is primarily known for its ability to measure barometric pressure. It employs a piezo-resistive sensor to detect changes in pressure, providing accurate readings in Pascals (Pa) or other pressure units.

2. **Temperature Measurement:**

- In addition to pressure, the BMP180 also includes a temperature sensor. It can provide temperature readings in degrees Celsius. The temperature data is often used to compensate for changes in pressure due to temperature variations.

3. **I2C Communication:**

- The BMP180 communicates with other devices, such as microcontrollers, using the I2C (Inter-Integrated Circuit) protocol. This makes it easy to interface with various platforms and microcontrollers.

4. **Compact Size:**

- The BMP180 is designed to be small and lightweight, making it suitable for applications where space is limited, such as in portable devices or IoT (Internet of Things) projects.

5. **Low Power Consumption:**

- The sensor is designed to operate with low power consumption, making it suitable for battery-powered applications.

6. **Pressure and Altitude Calculation:**

- The BMP180 can calculate altitude based on the atmospheric pressure. It is commonly used in altimeters to determine the height above sea level.

7. **Calibration Data:**

- To enhance accuracy, the BMP180 sensor is often calibrated during manufacturing. Calibration data is stored in the sensor, and it can be used to compensate for variations and improve the accuracy of pressure and temperature readings.

8. **Usage in Various Applications:**

- The BMP180 is widely used in meteorological instruments, navigation systems, and any application where accurate pressure and temperature data is required.

Pin Diagram of BMP180

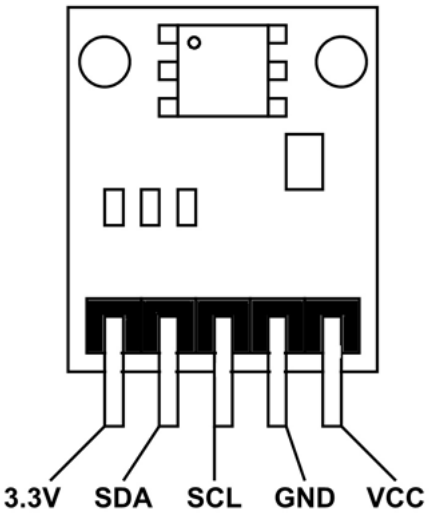


Figure 3.13 Pin Diagram of BMP180

The BMP180 barometric pressure sensor has an I2C interface, which means it communicates with the Arduino using only two pins. The BMP180's I2C interface has a hardwired address of 0x77HEX.

Here are some other features of the BMP180:

- Operating voltage: 1.8V to 3.6V, with 3.3V being the recommended voltage
- Input voltage: 3.3V to 5.5V
- Peak current: 1000uA
- Measuring range: 300 to 1100hPa
- Accuracy: Up to 0.02 hPa
- Low power consumption: 5 µA in standard mode and 0.1 µA in standby mode
- Small and compact: Can be used in weather stations or drones
- Fully calibrated: Has very low noise

3.6 Temperature and Humidity Sensor (DHT22)

DHT22 output calibrated digital signal. It utilizes exclusive digital-signal-collecting-technique and humidity sensing technology, assuring its reliability and stability.Its sensing elements is connected with 8-bit single-chip computer. Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of program in OTP memory, when the sensor is detecting, it will cite coefficient from memory. Small size & low consumption & long transmission distance (20m) enable DHT22 to be suited in all kinds of harsh application occasions. Single-row packaged with four pins, making the connection very convenient.

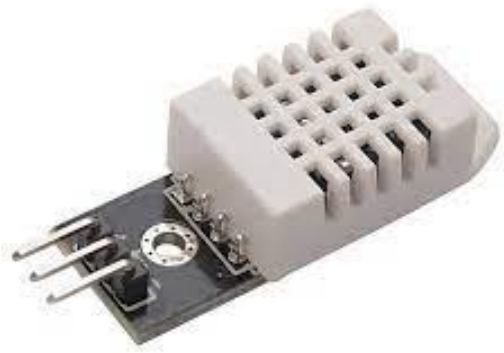


Figure 3.14 DHT22 Sensor

The DHT22 sensor, also known as the AM2302, is a digital temperature and humidity sensor. It is commonly used in various applications where monitoring environmental conditions is necessary. Here's a brief definition of the DHT22 sensor:

1. **Digital Temperature and Humidity Measurement:**

- The DHT22 is designed to measure both temperature and humidity in the surrounding environment. It provides digital outputs for both of these parameters, making it easy to interface with microcontrollers and other digital devices.

2. **Accuracy and Precision:**

- The sensor is known for its reasonable accuracy and precision in measuring temperature and humidity. It has a specified accuracy level for both parameters.

3. **Wide Operating Range:**

- The DHT22 sensor can operate over a broad temperature and humidity range, making it suitable for various environmental conditions.

4. **Single-Wire Digital Interface:**

- The DHT22 communicates with external devices, such as microcontrollers, through a single-wire digital interface. This simplifies the wiring and integration process.

5. **Calibrated Sensor:**

- The DHT22 is pre-calibrated during manufacturing, meaning that it doesn't require additional calibration by the user. This makes it relatively easy to use out of the box.

6. **Relative Humidity (RH) and Temperature Outputs:**

- The sensor provides relative humidity (RH) readings as a percentage and temperature readings in degrees Celsius or Fahrenheit, depending on the model.

7. **Low Power Consumption:**

- The DHT22 sensor is designed to operate with low power consumption, making it suitable for battery-powered applications and other projects where energy efficiency is important.

8. **Applications:**

- The DHT22 sensor is commonly used in various applications, including home automation, weather stations, industrial monitoring, and other projects where monitoring and controlling temperature and humidity are essential.

Pin Diagram

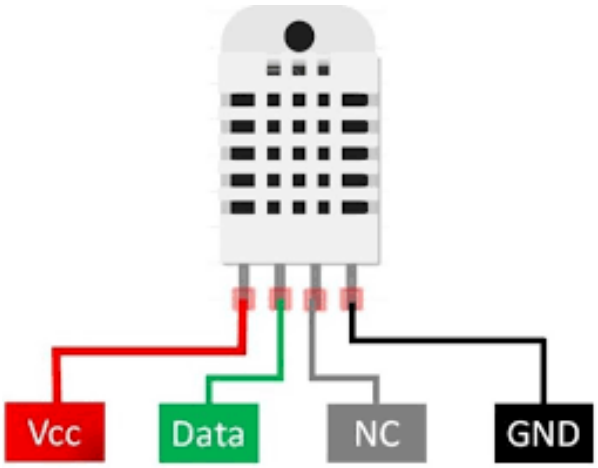


Figure 3.15 DHT22 Sensor

3.7 Vibration Sensor (SW-420)

The SW420 is a type of vibration sensor that is commonly used to detect vibrations and movements in various electronic projects. It is a simple module that consists of a vibration-sensitive component and an amplifier. Here's how the SW420 vibration sensor typically works:

Figure 3.15 SW-420 (Vibration Sensor)

1. **Vibration Detection:**

- The SW420 sensor has a small metallic spring inside, and when it experiences vibration or movement, the spring flexes or deforms.
- This deformation causes the electrical resistance of the spring to change.

2. **Variable Resistance:**

- The change in resistance is then converted into a voltage signal by an internal amplifier circuit in the SW420 module.
- The module usually has a potentiometer that allows you to adjust the sensitivity of the sensor.

3. **Output Signal:**

- The output is a digital signal that changes state when a certain level of vibration or movement is detected.
- When no vibration is present, the output is typically high (logical 1).
- When vibration is detected and crosses the set threshold, the output switches to low (logical 0).

4. **Interface with Microcontrollers or other Electronics:**

- The digital output signal can be interfaced with microcontrollers, Arduino boards, Raspberry Pi, or other electronic devices.
- Program your microcontroller to take specific actions or trigger events based on the sensor's output state change.

Pin Diagram

Figure 3.15 SW-420 Sensor Pin Diagram

5. **Applications:**

- SW420 vibration sensors are commonly used in alarm systems, home security systems, robotics, and industrial applications to detect unauthorized movement or vibrations.
- They are also used in projects where triggering an action based on physical movement is required.

6. **Usage Tips:**

- Adjust the sensitivity using the potentiometer according to your project requirements.
- Some sensors may have an onboard LED to indicate the detection of vibration.

3.8 2.8 inch SPI Screen Module TFT Interface 240 x 320

This TFT display is big (2.8″ diagonal) bright and colorful! 240×320 pixels with individual RGB pixel control, this has way more resolution than a black and white 128×64 display.

The display can be used in two modes: 8-bit and SPI. For 8-bit mode, you'll need 8 digital data lines and 4 or 5 digital control lines to read and write to the display (12

lines total). SPI mode requires only 5 pins total (SPI data in, data out, clock, select, and d/c) but is slower than the 8-bit mode. In addition, 4 pins are required for the Non-Touchscreen (2 digital, 2 analogs). This 2.8 inch SPI Non-Touch Screen Module is wrapped up into an easy-to-use breakout board, with SPI connections on one end and 8-bit on the other. Both are 3-5V compliant with high-speed level shifters so you can use it with any microcontroller. If you're going with SPI mode, you can also take advantage of the onboard MicroSD card socket to display images.

3.16 Figure of 2.08 inch SPI Screen Module TFT

Specification :

The specifications for a 2.8-inch TFT LCD with touch display can vary depending on the specific model or manufacturer. However, I can provide you with a general overview of the common pin specifications for such displays. Please note that you should refer to the datasheet or documentation provided by the manufacturer for precise details. Here is a typical configuration:

1. **VCC/Power Supply** : Power input for the display. Usually requires a voltage within a specified range (e.g., 3.3V or 5V).

2. **GND/Ground**: Ground reference for the power supply.
3. **LED+ (Backlight)**: This pin is for connecting to the positive terminal of the LED backlight. The backlight may require a separate power supply or be driven directly from the display's power.
4. **LED- (Backlight)**: This is the ground or negative terminal of the LED backlight.
5. **RESET**: Reset pin for initializing the display.
6. **DC/RS (Data/Command)**: Data/Command selection pin. It is used to indicate whether the data on the data bus is a command or display data.
7. **SDI/MOSI (Serial Data In/Master Out Slave In)**: Serial Data Input pin for sending data to the display.
8. **SCK (Serial Clock)**: Serial Clock pin for synchronizing data transfer.
9. **CS (Chip Select)**: Chip Select pin for enabling communication with the display.
10. **SDO/MISO (Serial Data Out/Master In Slave Out)**: Serial Data Output pin for receiving data from the display.
11. **T_IRQ (Touch Screen Interrupt)**: Touch screen interrupt pin. It is used to indicate when a touch has been detected.
12. **T_CLK (Touch Screen Clock)**: Touch screen clock signal.
13. **T_CS (Touch Screen Chip Select)**: Touch screen chip select pin.
14. **T_DIN (Touch Screen Data In)**: Touch screen data input pin.
15. **T_DO (Touch Screen Data Out)**: Touch screen data output pin.
16. **T_PEN (Touch Screen Pen Down)**: Touch screen pen-down signal.

Pin Diagram of SPI Screen Module TFT

3.17 Pin Diagram of 2.8 inch SPI Screen Module TFT

Application of SPI Screen Module TFT

A 2.8-inch TFT touch display can be used in various applications where a graphical user interface (GUI) or touch input is required. Here are some common applications for a 2.8-inch TFT touch display with their corresponding pin functionalities:

1. **Embedded Systems**: These displays are often used in embedded systems for projects like Arduino or Raspberry Pi-based devices. The display allows you to create interactive interfaces for your projects, such as home automation systems, weather stations, or electronic gadgets.
2. **Industrial Control Panels**: The touch display can be integrated into industrial control panels for machinery and equipment. Operators can interact with the control panel using the touch interface to monitor and control various parameters.
3. **Medical Devices**: TFT touch displays are utilized in medical devices, such as patient monitoring systems or portable diagnostic equipment. The touch interface makes it

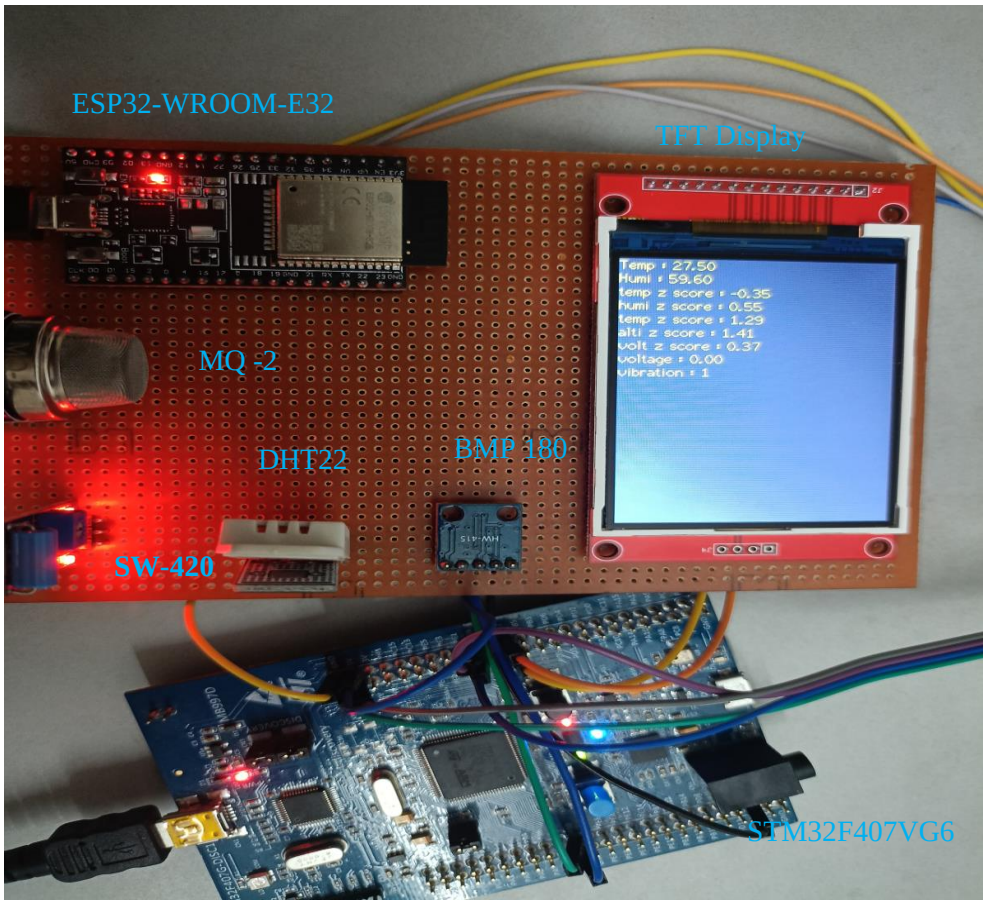
easy for healthcare professionals to interact with the device and access critical information.

4. **Automotive Systems**: These displays can be integrated into automotive applications for in-car entertainment systems, navigation, and vehicle control interfaces. The touch functionality enables a user-friendly interface for drivers and passengers.
5. **Smart Home Devices**: Smart home devices, like thermostats, lighting controls, or security systems, can benefit from a touch display to provide users with an intuitive interface for managing and controlling their smart home features.
6. **Consumer Electronics**: Portable electronic devices, like MP3 players, digital cameras, or handheld gaming consoles, can incorporate these displays to enhance user interaction and provide visual feedback.
7. **Test and Measurement Equipment**: Instruments used for testing and measurement purposes, such as oscilloscopes or multimeter, can employ TFT touch displays to provide a user-friendly interface for configuring and interpreting measurements.
8. **Educational Kits and Learning Platforms**: These displays can be used in educational settings for creating interactive learning kits or platforms where students can engage with visual content through touch interaction.

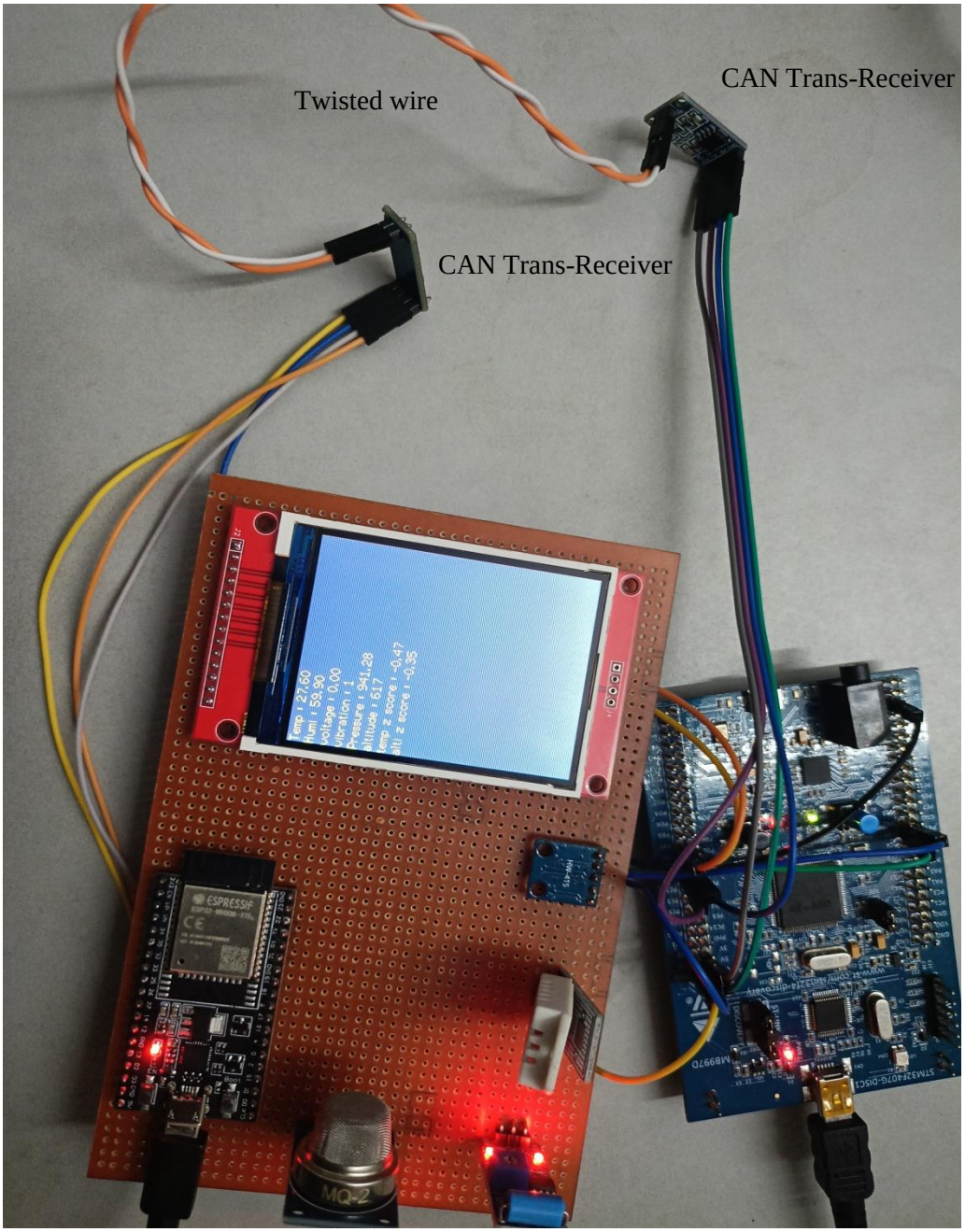
Chapter 4

Implementation

1. We have already discussed about the different type of component such as Stm32f407vg ,ESP32-WROOM-E32, BMP180 , DHT22 Sensor,MQ-2,and SW-420 sensor with their specification and use in this section we are designing the circuit by the help of such component. Apart from this we will write the code in arduino ide. In addition of this we will write the code in STM32 ide using real time operating system (FreeRTOS) . but first of all we made the circuit. We have used spi bus for communication between display and ESP32 , i2c bus BMP18between BMP180 and CAN bus for communicating between STM32 and ESP 32.



4.1 Implemented Circuit

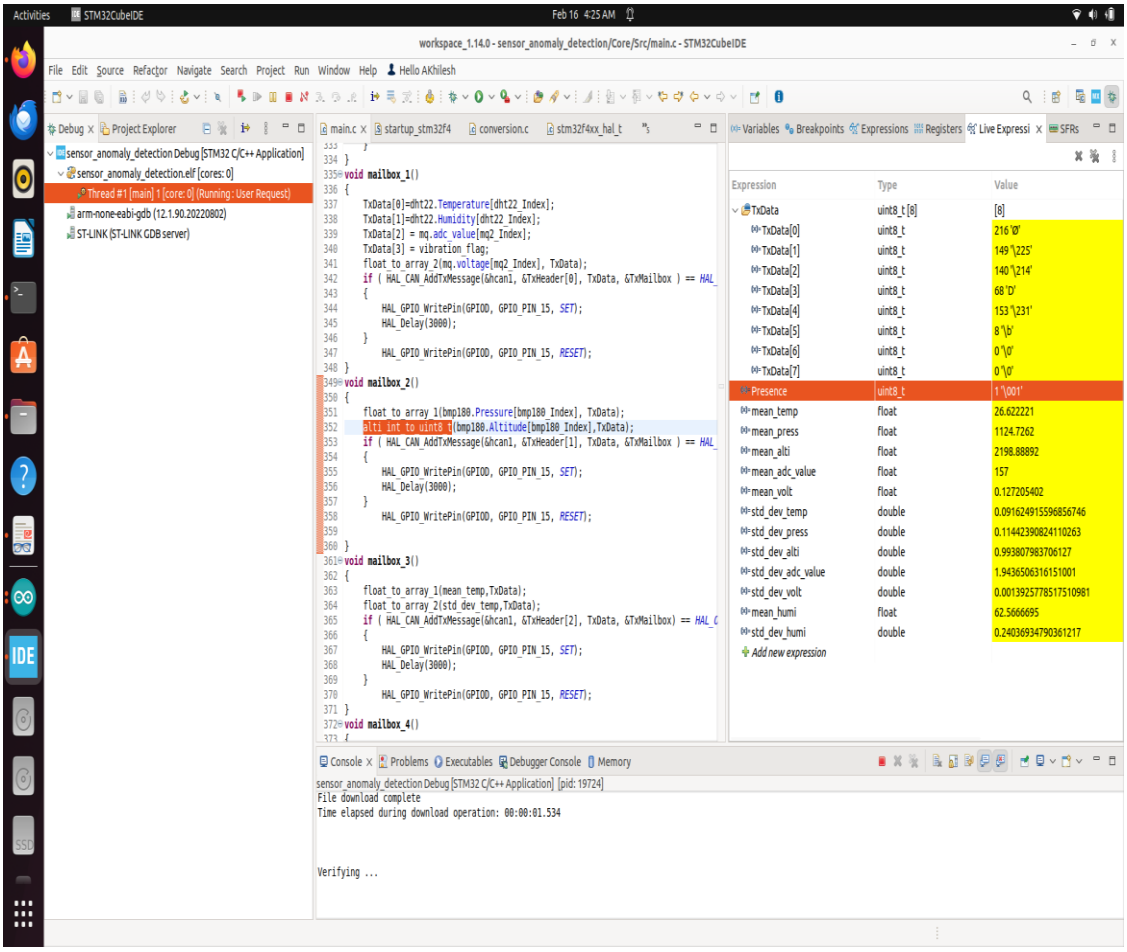


4.2 CAN Bus Implementation

Chapter : 5

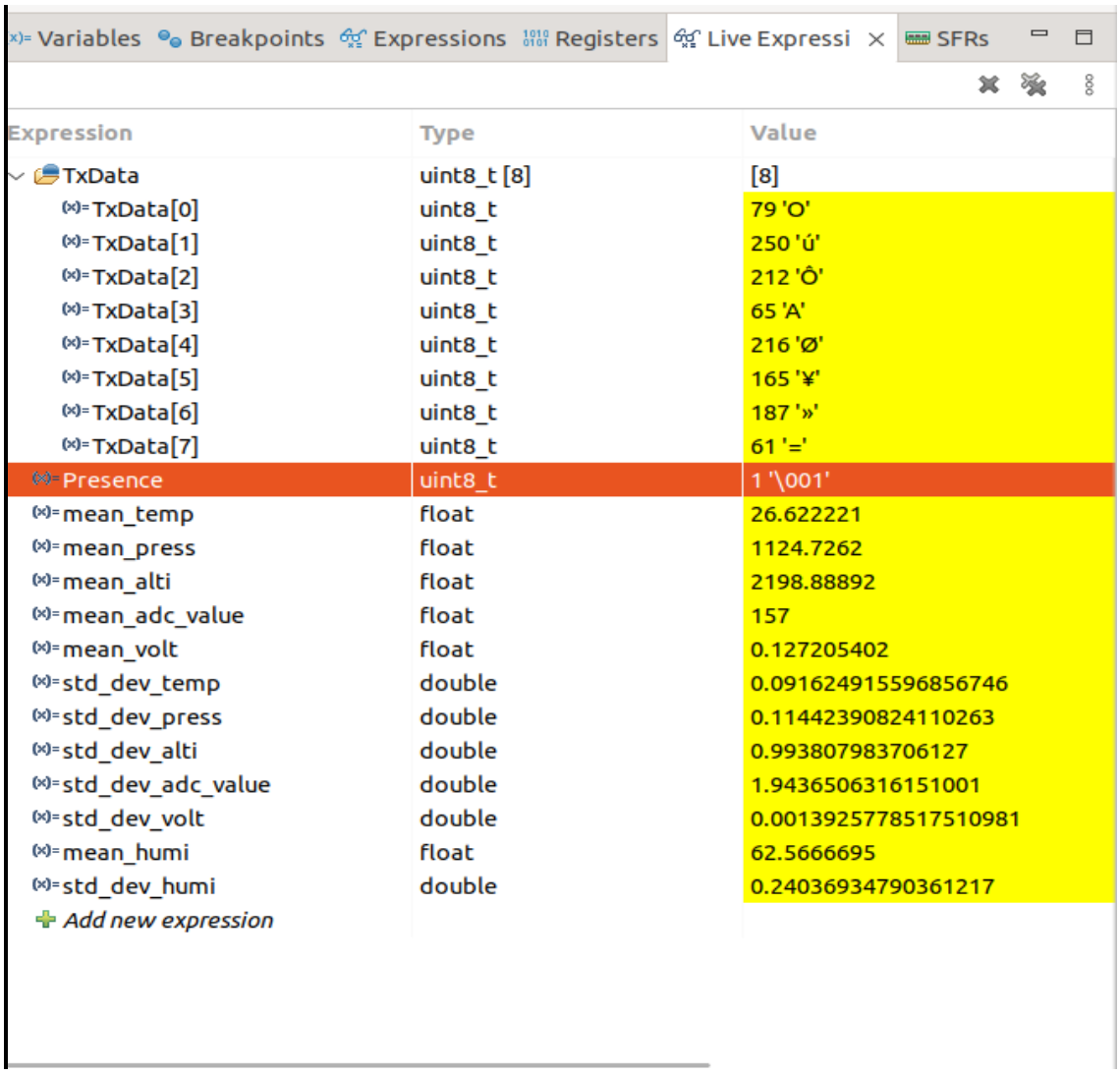
Results

Below figure shows the implementation results of our project that are carried out using STM32Cube IDE, arduino ide and Thingsboard for sending sensor data to cloud we use MQTT. One figure also shows the detailed connection diagram of CAN Connections for communicating between two boards. On stm32 ide we have written code in RTOS and then compile it then declare the some variable for live expression which has been given below.



5.1 STM32 Task output with variables

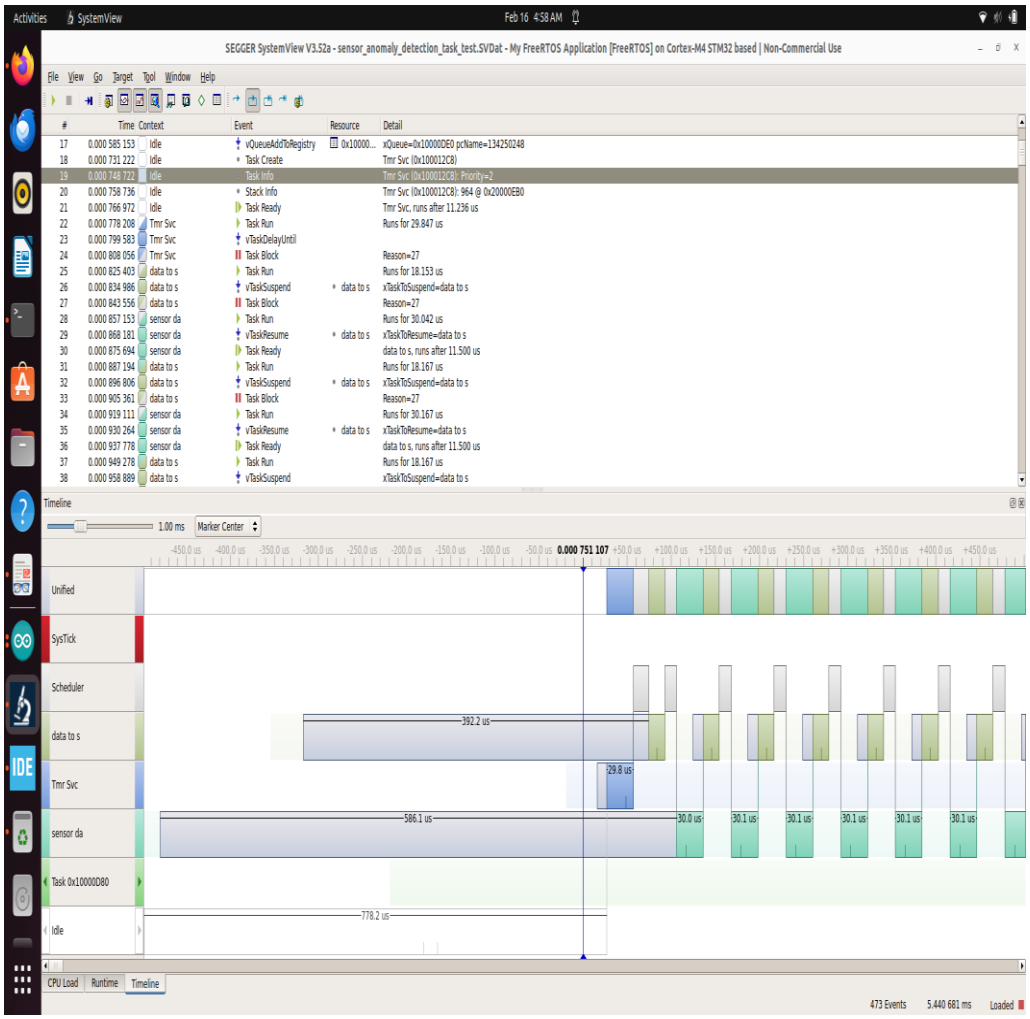
This is some global variable on the basis of this we are applying a correlation between Sensors for their threshold value . There is two method for this the first one is simple correlation method and second is z-score method.



5.2 STM32 Task output with variables

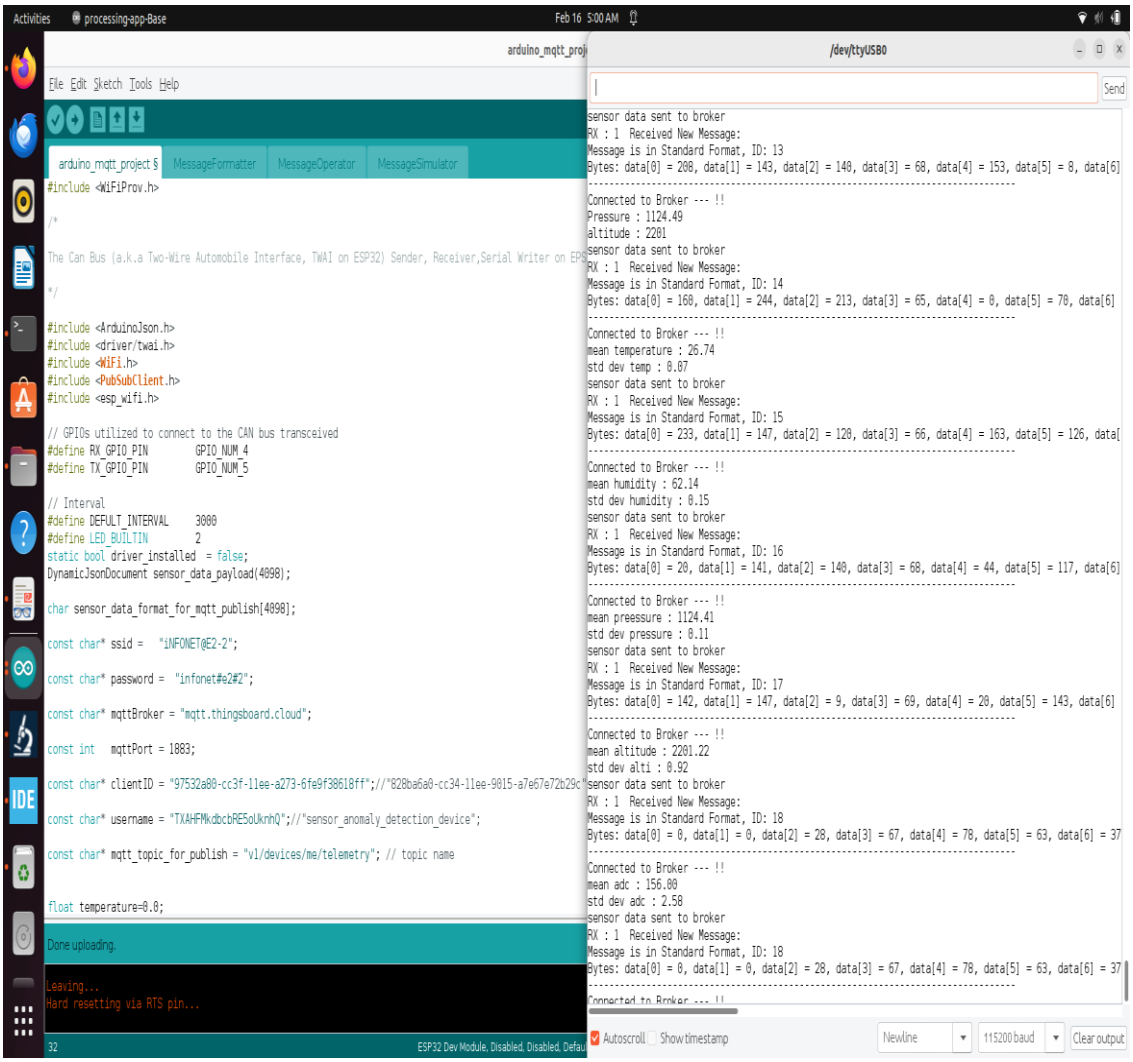
There is two Task the first one is (data_to_send) for sending sensor data to esp32 by using CAN bus and second is (Sensor data) which is takig dat from different type of sensor such as BMP180,SW-420,MQ-2 and DHT22.

The priority of task who is sending the data to ESP2 having higher priority than the task who is taking the sensor data. In this figure we can observe the behaviour of each task.First of all task sende_data will create then the task2 (data_to_send) will created. Then after SVC invoke the scheduler it schedule first task (data_to_send)who have higher priority after this task 2 will execute its for only debugging.



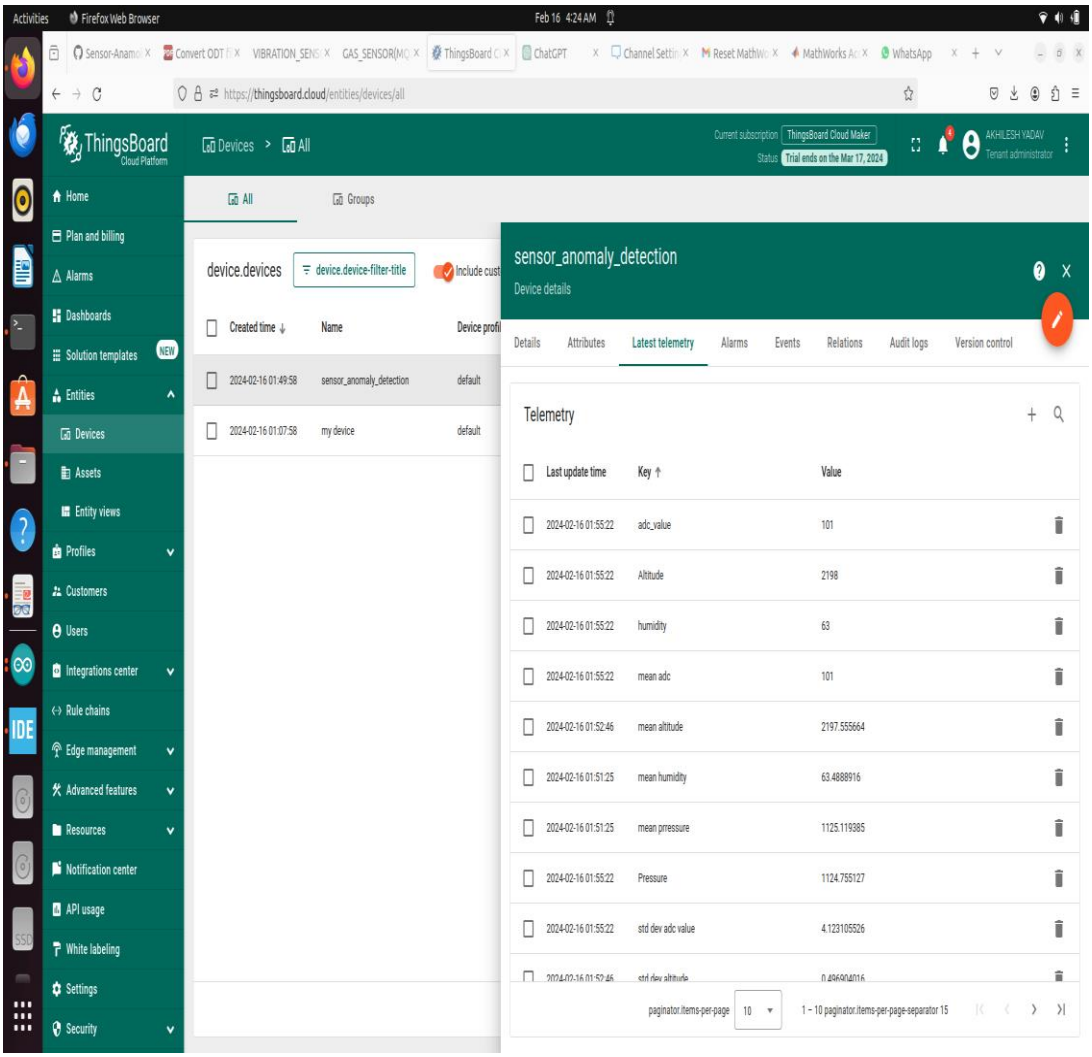
5.3 Segger View of Task

In the arduino ide we wrote code for MQTT and receive the data from STM32 using CAN bus compiled it and upload on arduino then we got a couple of data on arduino ide .



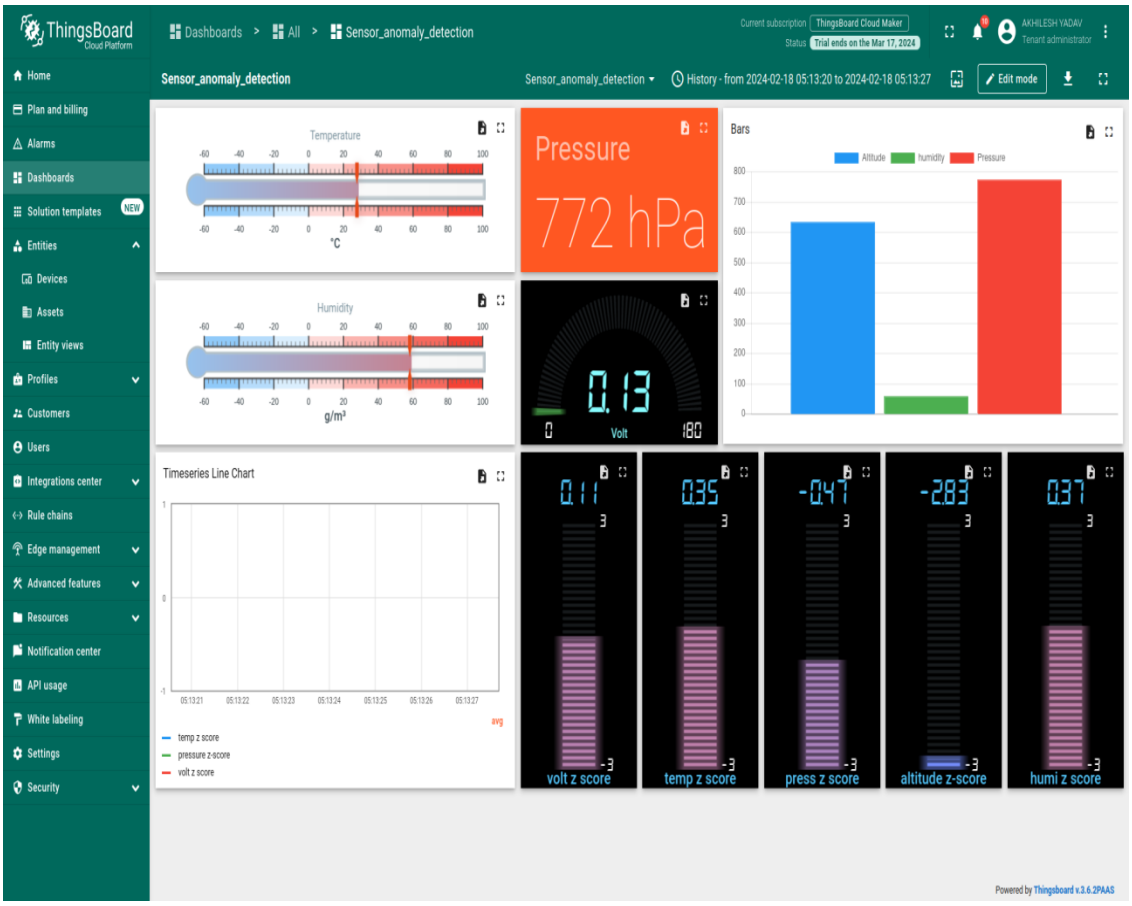
5.4 Arduino ide output

We made a thingsboard account and send the data from esp32 so that user can observe the better output with graph and widgets. In below image Thingsboard got sensor data such as temperature ,humidity, pressure ,altitude ,mean altitude ,mean adc,and other essential values.



5.5 Data on ThingsBoard

This is widgets showing the sensor data in the graph or some special form.so that user can observe the behavior of their devices



5.6 Data on widgets

Conclusion

In conclusion, sensor anomaly detection is a crucial technology with numerous advantages across various domains. By continuously monitoring and analyzing sensor data, anomaly detection systems offer early insights into abnormal patterns, deviations, or faults. This contributes to enhanced system reliability, operational efficiency, and cost savings. The proactive nature of anomaly detection allows for preventive maintenance, reducing downtime and extending the lifespan of equipment.

Moreover, the ability to differentiate between normal variations and genuine anomalies minimizes false alarms, providing a more accurate and reliable monitoring system. This is particularly beneficial in critical applications such as industrial processes, healthcare, security, and autonomous systems.

References

1. About STM32 Board https://www.st.com/resource/en/usermanual/

2. About DHT 22 interfacing with stm32f407VG6

https://controllerstech.com/temperature-measurement-using-dht22-in-stm32/

3.BMP180 interfacing with stm32f407VG6

https://controllerstech.com/interface-bmp180-with-stm32/

4. About interfacing MQ-2 sensor with stm32f407VG6

https://www.academia.edu/32790153/Design_of_Smoke_Alarm_System_Based_on_STM32

5. CAN bus implementation

https://community.simplefoc.com/t/can-bus-library-for-esp32-and-stm32/2478

https://copperhilltech.com/blog/can-bus-development-with-esp32wroom32-development-board/

6. About Controlled Area Network on
https://www.ccontrols.com/pdf/CANtutorial.pdf

7. Interfacing ESP32 sensor with TFT Dilplay

https://electropeak.com/learn/interfacing-2-8-inch-tft-lcd-touch-screen-with-esp32/

8.Library for CAN controller for ESP32

https://docs.espressif.com/projects/esp-idf/en/stable/esp32/apireference/peripherals/twai.html

9 .thingsBoard reference https://thingsboard.io/