ELECTRIC VEHICLE BATTERY FAULT MONITORING USING IOT

A PROJECT REPORT

Submitted by

R.INDUMATHI (110318106013)

K.RAMYA (110318106032)

L.YOGESHWARI (110318106049)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

GRT INSTITUTE OF ENGINEERING AND TECHNOLOGY

TIRUTTANI

ANNA UNIVERSITY: CHENNAI 600 025

JUNE-2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**ELECTRIC VEHICLE BATTERY FAULT MONITORING USING IOT**" is the bonafide work of the **"R.INDUMATHI (110318106013), K.RAMYA (110318106032), L.YOGESHWARI (110318106049)"** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.P.SIVAKUMAR, M.Tech.,Ph.D.,	**D.SARATHY, M.E.,**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Professor	**Assistant Professor**
Electronics & Communication Engineering	Electronics & Communication Engineering
GRT Institute of Engineering & Technology	GRT Institute of Engineering & Technology
Tiruttani.	Tiruttani.

Submitted for the Project Viva-Voce examination held on ___________

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

First and foremost we place this project work on the feet of **GOD ALMIGHTY** and **OUR PARENTS** who is the power of strength in each step of progress towards the successful completion of our project.

Our sincere thanks to honorable chairman, **Shri.G.RAJENDRAN** and Managing Director **Shri.G.R.ANANTHAPADMANABHAN** for creating a wonderful atmosphere inside the campus.

We express our thanks to our Dean **Dr.S.MEENAKSHI SUNDARAM Ph.D**(MANAGEMENT&AGRI.EXT)**,MBA**(HR&MARKETING),**Msc**(AGRI.EXT),**PGDBM,PGDJMC,PGDTD,DST…**for his great support and blessings.

We express our thanks to our principal, **Dr.S.ARUMUGAM, Ph.D.,** and our Administrative officer **Mr.K.SASIKUMAR** who helped us in providing required facilities to complete this project.

We like to express our deep gratitude to **Dr.P.SIVAKUMAR, M.Tech., Ph.D.,** Professor, Head of the Department of Electronics and Communication Engineering, for giving the opportunity and facilities to complete our work in time.

We like to express our special gratitude to the project coordinator **Mr.C.E.MOHANKUMAR, M.E.,** Assistant professor, Department of Electronics and Communication Engineering, for his kind co-operation and guidance in doing our project work.

We would like to express our deep sense of profound gratitude and thanks to our project supervisor **D.SARATHY,M.E,.** Assistant Professor, Department of Electronics and Communication Engineering, for her enthusiastic guidance, suggestions and constant encouragement in completing project.

Finally, we thank our faculty members and friends for their moral support and valuable suggestions in this project.

ABSTRACT

Our system monitor current, voltage and temperature condition which is collected from the LI-ION battery's like voltage sensor, current sensor, temperature sensor and the remaining charge capacity in a real-time scenario. The information collected from all the associated battery clients in the system's is analysis. The malfunction of the battery status are continuously monitored based on sudden charge and discharge voltage of battery bank by using of this sensors and battery conditions are viewed in the cloud and mobile application with the help of IOT module. To send the notification to mobile app to alert user of the vehicle .By using the sensors, continuously monitoring the battery and displayed the voltage, current and temperature values in the OLED screen to detect the fault and prevent the vehicle. If any tedious find the battery automatically motor off.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

- **OLED** -ORGANIC LIGHT EMITTING DIODE
- **IOT** - INTERNET OF THINGS
- **GPS** - GLOBAL POSITIONING SERVICE
- **LI-ION** - LITHIUM ION BATTERY
- **SOC** - STATE OF CHARGE
- **EV** - ELECTRIC VEHICLE
- **BMS** - BATTERY MANAGEMENT SYSTEM
- **SOH** -STATE OF HEALTH
- **GPRS** - GENERAL PACKET RADIO SYSTEM
- **DC** - DIRECT CURRENT
- **LCD** - LIQUID CRYSTAL DISPLAY
- **SOA** - SAFE OPERATION AREA

CHAPTER 1

INTRODUCTION

Nowadays, electric vehicle (EV) is becoming popular since the fuel prices becoming more expensive. Due to the scenario, many vehicle manufacture looking for alternatives of energy sources other than gas. The use of electrical energy sources may improve the environment since there is less pollution. In addition, EV produces great advantages in terms of energy saving and environmental protection. Most EVs used rechargeable battery which is lithium ion battery. It is smaller to be compared with lead acid. In fact, it has a constant power, and energy's life cycle is 6 to 10 times greater compared with lead acid battery. Lithium ion battery life cycle can be shortened by some reasons such as overcharging and deep discharges. On the other hand, EV usually has limited range of travelling due to battery size and body structure.

Now, an important reason that limits the application of EV is the safety of existing battery technology. For example, overcharging battery not only could significantly shorten the life of the battery, but also cause serious safety accidents such as fire. Therefore, a battery monitoring system for EV that can notify the user about battery condition is necessary to prevent the stated problems. Previous battery monitoring system only monitoring and detect the condition of the battery and alarmed the user via battery indicator inside the vehicle. Due to the advancement of the design of notification system, internet of things (IOT) technology can be used to notify the manufacturer and users regarding the battery status. This can be considered as one of the maintenance support procedure that can be done by the manufacturer. IOT utilizes internet connectivity beyond traditional application, where diverse range of devices and everyday things can be connected via the internet, making the world is at the user’s finger tips. Motivating by the stated problems, in this work, the design and development of a battery monitoring system using IOT technology is proposed.

1.1 EXISTING METHOD

Battery is the most essential component of any vehicle. So, perfect maintenance of any battery is very much essential for it to function properly. Lead Acid batteries which are more commonly used in the vehicle battery is need to be efficiently monitored, for it to perform better under all circumstances. So, a more systematic battery management system needs to be implemented so that the performance of the battery can be monitored continuously. When it comes to battery, the two most important parameters are the State Of Charging (SOC) and State of Health (SOH) of the battery. There are several coherent methods to calculate these parameters. But these methods cannot provide correct results, as the battery materials, atmosphere surrounding the battery, the load put on to the battery, will affect these parameters. Overcharging of the battery leads to emission of gases like Hydrogen, Oxygen etc. This Battery Management System (BMS) aims at detecting the emission of these gases from the battery, when it is overcharged, and monitors the other basic parameters such as Voltage, Current, Temperature of the battery using STM controller and sensors but the parameters are not displayed to the clients. It is also equipped with GPS module, which enables tracking of vehicles.

1.1.1 EXISTING SYSTEM DISADVANTAGES

1. There is no SOC estimation and cannot display the condition in it.
2. There is no current measurement.
3. There is no user interface.
4. There is no temperature measurement.
5. There is no cooling system.
6. IOT is not present which would be easy to get information through mobile

1.2 OBJECTIVE

- The periodical monitoring/observations are required for battery source to provide continuous power to the load without any interruption.
- Our proposed system monitors and stores parameters that provide an indication of the lithium ion state of charge, voltage, current, temperature, and the remaining charge capacity in a real-time scenario. Wireless local area network is used as the backbone network.
- The information collect from all the associated battery clients in the system is analysed.
- The malfunction of the battery status are continuously monitored based on sudden charge & discharge voltage of battery bank and battery conditions are viewed in the cloud with help of IOT module.

CHAPTER 2

LITERATURE SURVEY

2.1 INFERENCE ON STATE OF CHARGE

Title 1: An optimal nonlinear observer for state-of-charge estimation of lithium-ion batteries.

Authors: Yong Tian, Dong Li, Jindong Tian

Year: 2015

Description:

State-of-Charge (SOC) information is very crucial for the control, diagnostics and monitoring of Li-ion cells/batteries. Compared to conventional data-driven or equivalent circuit models often employed in battery management systems, electrochemical battery models have the potential to give physically accurate the SOC information by tracking the Li-ion concentration in each electrode. In this project, two nonlinear observer designs are presented to estimate Li-ion battery State-of-Charge based on reductions of an electrochemical model. The first observer design uses a constant gain Luenberger structure whereas the second one improves it by weighing the gain with the output Jacobian. For both observer designs, Lyapunov's direct method is applied and the design problems are converted to solving LMIs. Simulation results are included to demonstrate the effectiveness of both observer designs.

2.2 INFERENCE ON OVERCHARGE AND DISCHARGE

Title 2: Research on overcharge and over discharge effect on Lithium-ion batteries.

Authors:C. Wu, J. L. Sun, C. B Zhu, Y. W. Ge, Y. P. Zhao

Year: 2016

Description:

Diagnose and analyse the overcharge and over discharge fault of lithium ion battery. Through the dynamic simulation model, the phenomenon of overcharge and over discharge fault for automotive lithium-ion battery (LIB) was discussed, and the fault diagnosis effect was summarized. The results of this study show that the fault diagnosis analysis of LIB can achieve good results. It is of certain application value for diagnosis of LIB with different parameters. Therefore, it’s concluded that the overcharge and over discharge faults of automotive LIB are likely to jeopardize the using effect of the batteries, which should attract more attention of relevant automobile manufacturers.

2.3 INFERENCE ON DYNAMIC MODEL

Title3: Battery management system based on battery nonlinear dynamics modelling.

Authors: Szumanowski and Y. Chang

Year: 2017

Description:

To determining electromotive force and battery internal resistance as time functions, which are depicted as functions of state of charge (SOC) because the model is based on battery discharge and charge characteristics under different constant currents that are tested by a laboratory experiment. This method of determining the battery SOC according to a battery modelling result. The influence of temperature on battery performance is analyzed according to laboratory-tested data, and the theoretical background for calculating the SOC is obtained. The algorithm of battery SOC indication is depicted in detail. The algorithm of the battery SOC indication considering the influence of temperature can be easily used in practice by a microprocessor. An battery is used to depict the modelling method. In fact, the method can also be used for different types of contemporary batteries, as well as Li-ion batteries, if the required test data are available.

2.4 INFERENCE ON TRACKING SYSTEM

Title 4: Design and implementation of real time tracking system based on arduino intel galileo.

Authors:O. A. Mohamad, R. T. Hameed, N. Tăapus

Year: 2018

Description:

Tracking and monitoring vehicles are coming vastly utilized based on Global Positioning System (GPS). In this paper a real time tracking system is proposed. The proposed framework would make great utilization of new innovation that basis on embedded board denotation Arduino Intel Galileo. This system acts on Global System for Mobile Communication (GSM), Global Positioning System (GPS) and General Packet Radio System (GPRS) which are utilized for vehicle tracking and monitoring. The SIM908 Module is applied which incorporates three technic to be specific GPS, GPRS and GSM. GPS gives the vehicle location coordinates, GPRS transmits these data to the server and finally the GSM transmits warning message to the vehicle owner phone. This exhibited the evolution of the vehicle tracking system prototype which is used in the vehicle. In particular, the framework will use GPS to acquire a vehicle location coordinates and send it utilizing GSM modem to the owner phone and to the web server. After that, the browser can carry on the PHP site page that utilizes Google maps to display the place in a real time. To define the location accuracy of the suggested system, we compared the system proposed results with the different commercial GPS devices.

2.5 INFERENCE ON BATTERY MANAGEMENT SYSTEM

Title 5: IOT Based Battery Management System

Authors: Harish N , Prashal V and Dr. D. Sivakumar

Year: 2019

Description:

Lead Acid batteries which are more commonly used in the vehicle battery to need to be efficiently monitored. So, a more systematic battery management system needs to be implemented so that the performance of the battery can be monitored continuously. When it comes to battery, the two most important parameters are the State Of Charging (SOC) and State of Health (SOH) of the battery. There are several coherent methods to calculate these parameters. But these methods cannot provide correct results, as the battery materials, atmosphere surrounding the battery, the load put on to the battery, will affect these parameters.
Overcharging of the battery leads to emission of gases like Hydrogen, Oxygen etc. It is also equipped with GPS module, which enables tracking of vehicles. Also these values are displayed in Cloud, which brings the concept of Internet of Things (IOT).

2.6 INFERENCE ON DEVELOPMENT OF BATTERY

Title 6: Present Status and Development Trend of Batteries for Electric Vehicles.

Authors: Mohd Helmy Abd Wahab, Nur Imanina Mohamad Anuar , Radzi Ambar, AslinaBaharum, Shanoor Shanta1.

Year: 2020

Description:

The application of Internet-of-things to monitoring the performance of electric vehicle battery. It is clear that an electrical vehicle totally depends on the source of energy from a battery. However, the amount of energy supplied to the vehicle is decreasing gradually that leads to the performance degradation. This is a major concern for battery manufacture. In this work, the idea of monitoring the performance of the vehicle using IOT techniques is proposed, so that the monitoring can be done directly. The proposed IOT-based battery monitoring system is consists of two major parts i) monitoring device and ii) user interface. Based on experimental results, the system is capable to detect degraded battery performance and sends notification messages to the user for further action.

2.7 INFERENCE ON GENERALIZED BATTERY MANAGEMENT SYSTEM

Title 7: Designing a new generalized battery management system.

Authors: Sandro Martin, Passoukwende Minoungou, Eugene Moss, Sagarkuma rPatel .

Year: 2021

Description:

Design a generator based on size and power output, and designed dependent systems for heating the batteries, charging them, powering the motor from the generator, and switching between circuits. The battery heating is accomplished via heating pads that activate to warm the batteries to a safe temperature while the vehicle is in use, extending the operable temperature range of the design. When the battery state of charge runs low, the user may activate the generator to recharge them, allowing for continued operation beyond what would normally be achievable. They added power supplies to the design to allow the generator to drive the vehicle, and microcontroller-driven relays to ensure only the appropriate systems would activate when called upon, and to isolate the batteries from the motor when the generator is active. All of the systems in the electric vehicle are tested and functional, save for the state of charge determination, and the vehicle is capable of running using either the batteries or the generator.

CHAPTER 3

SYSTEM DESIGN

3.1 PROPOSED SYSTEM

In this proposed system, EV battery fault monitoring system. Temperature sensor, voltage sensor, current sensor are connected and used to monitor the battery condition and the lithium ion batteries are arranged in series and parallel are connect to the charger module to get the power supply to the vehicle .While charging, parameters collected information with use of all three sensor is fetched by Node MCU and triggers the relay for cutoff charging.The information collect from all the associated battery clients in the system is analyzed. The malfunction of the battery status are continuously monitored based on sudden charge & discharge voltage of battery bank. If any tedious find the battery automatically motor off and battery conditions are viewed in the cloud and mobile application with help of IOT module.When our device detects any malfunction. The condition will be monitored real-time through mobile application and the client gets the notification through the mobile application.

If the temperature, current and voltage are displayed in the OLED and the buzzer is alarmed and by using the relay to cut off the motor and stop the vehicle.

3.2 PROPOSED SYSTEM ADVANTAGES

- Use the unlicensed ISM frequency band.
- It is a flexible solution that can be easily adapted.
- It is scalable.
- It supports bi-directional communication.
- Provides a high level of security due to encryption algorithms.
- Provides energy efficiency.

3.3 BLOCK DIAGRAM

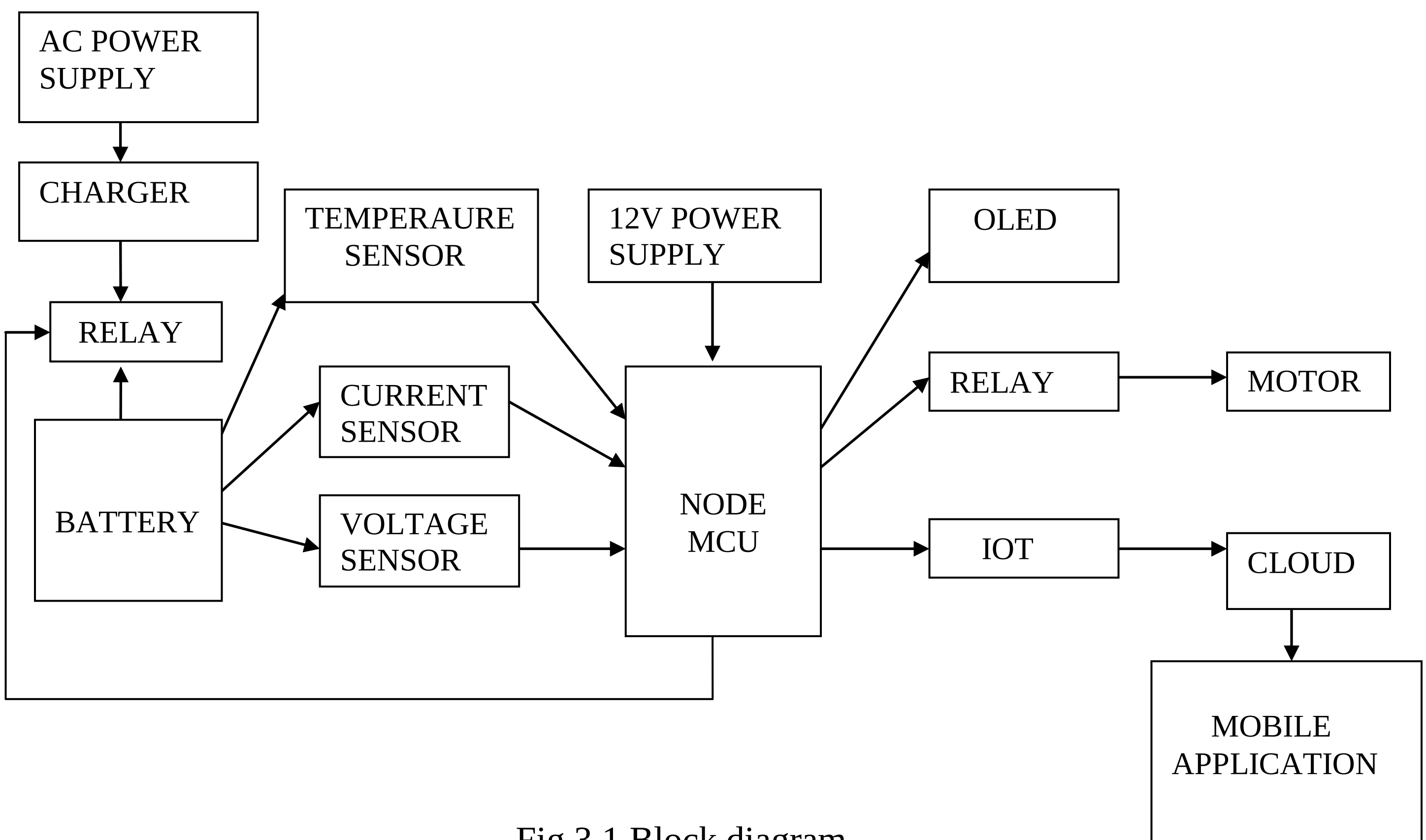

Fig 3.1 Block diagram

3.3.1 CIRCUIT DIAGRAM

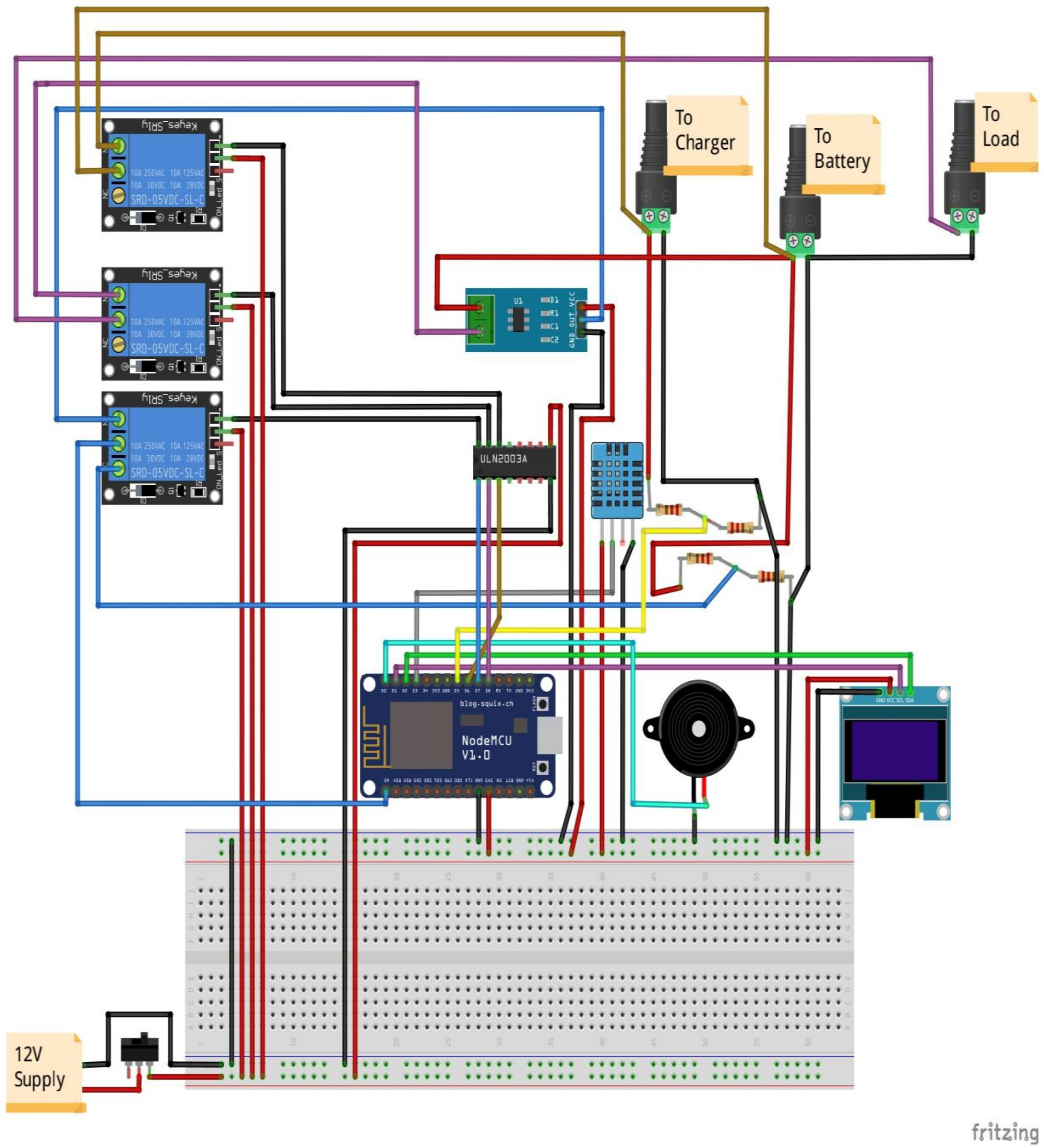

Fig 3.2 Circuit diagram

3.3.5 BLOCK DIAGRAM EXPLANATION

- The battery is connected with the temperature sensor, voltage sensor and current sensor.
- The sensors are connected with the node MCU and the 5v power is supplied to the node.
- The two relay are connected and one relay is connected between node and battery. Second relay is connected between the node and motor.
- The AC power is supplied to the charger module and the charger module provides the constant current to the battery and the battery is connected with the sensor to detect the temperature, current and voltage.
- The information are collected from the sensors and transferred to the node.
- If the temperature, voltage, current is high and the information is send to the node and relay is cutoff. The motor is off and the buzzer is alarmed.
- The battery is overcharged and the relay is automatically to turn off the charger module.
- If the parameters is high by using the IOT to send the notification to the mobile app. For the purpose of monitoring the battery and reduce the heat of the battery.

3.4 HARDWARE REQUIREMENTS

- NODE MCU(ESP32)
- POWER SUPPLY
- OLED
- BATTERY (LI-ION)
- TEMPERATURE SENSOR
- CURRENT SENSOR
- VOLTAGE ENSOR
- BUZZER
- RELAY
- DC-MOTOR
- CHARGER MODULE

3.5 SOFTWARE REQUIREMENTS

- ARDUINO IDE
- MIT APP INVENTER

3.4 HARDWARE IMPLEMENTATION

3.4.1 NODE MCU(ESP32)

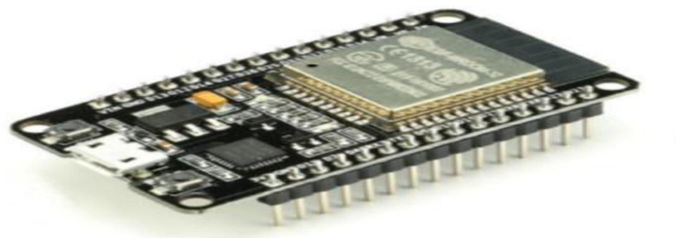

Fig 3.3 Node MCU

Node MCU is an open source Lua based firmware and development board specially targeted for IOT based Applications. It includes firmware that runs on the ESP32 Wi-Fi SOC from Espressif Systems and hardware which is based on the ESP-32 module. The Node MCU ESP32Development board comes with the ESP-32 module containing ESP32 chip having Tensilica Xtensa 32-bit LX106 RISC microprocessor. This Microprocessor supports RTOS and operates at 80MHz to 160MHz adjustable clock frequency. Node MCU has 128 KB RAM and 4MB of flash memory to store data and programs. Its high processing power within built Wi-Fi / Bluetooth and Deep Sleep Operating features make it ideal for IOT projects.

Node MCU can be powered using Micro USB jack and VIN pin(External Supply Pin).It supports UART,SPI and 12c interface.

Programming Node MCU ESP32

The Node MCU Development Board can be easily programmed with Arduino IDE since it is easy to use .Programming Node MCU with the Arduino IDE will hardly take 5-10 minutes. All we need is the Arduino IDE, a USB cable and the Node MCU board itself.

Uploading the first program

Once Arduino IDE is installed on the computer, connect the board with the computer using the USB cable. Now open the Arduino IDE and choose the correct board by selecting Tools>Boards>Node MCU and choose the correct Port by selecting Tools>Port. To get it started with the Node MCU board and blink the built in OLED, load the example code by selecting Files>Examples>Basics>blink. Once the example code is loaded into your IDE, click on the upload button given on the top bar. Once the upload is finished, you should see the built in OLED of the board blinking.

ESP32 Arduino core

As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to supports alternate tool chains to allow Arduino C/C++ to be complied down to these new processors. They did this with the introduction of the Board Manager and the SAM core. A “Core” is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file down to the target MCU’s machines language. Some creative ESP32 enthusiasts have developed an Arduino core for the ESP32 WiFi SOC that is available at the

GitHub ESP32 core web pages. This is what is popularly called the"ESP32 Core for the Arduino IDE" and it has becomes one of the leading software development platforms for the various ESP32 based modules and developments boards, including Node MCUs.

The Button

The Button is a Wi-Fi connected push button designed by Peter R Jennings. The Button is designed for single-purpose, internet-enabled functions. When the button is pressed, a connection is made to a web server which will perform the desired task. Applications include a doorbell or panic button**.**

Node USB

Node USB is an open IOT platform about the size of a standard USB stick. It was designed to leverage Node MCU (Lua) for easy programming and has the extra features of the USB capability. It is ideal for Plug-in-Play solutions, allowing easy prototyping for the developers.

Node MCU ESP32 Specification & Features

SPECIFICATION

1. MODEL-Node MCU ESP32
2. TYPE-ESP32
3. PROCESSOR-Tensilica LX6 Dual-Core
4. CLOCK FREQUENCY-240 MHz
5. SRAM-512 KB
6. MEMORY-4 MB
7. WIRELESS STANDARD-802.11 b/g/n
8. FREQUENCY-2,4 GHz

9. BT WIRELESS CONNECTION-Classic / LE

10.DATA INTERFACES-UART / I2C / SPI / DAC / ADC

11.OPERATING VOLTAGE-3,3V (operable via 5V-microUSB)

12.OPERATING TEMPERATURE-40°C - 125°C

13.DIMENSIONS-48 x 26 x 11,5 mm

14.WEIGHT-10 g

FEATURES

- Processors:
  - CPU: Xtensa dual-core (or single-core) 32-bit LX6 microprocessor, operating at 160 or 240 MHz and performing at up to 600 DMIPS
  - Ultra low power (ULP) co-processor
  - Memory: 320 KB RAM, 448 KB ROM
- Wireless connectivity:
  - Wi-Fi: 802.11 b/g/n
  - Bluetooth: v4.2 BR/EDR and BLE (shares the radio with Wi-Fi)
- Peripheral interfaces:
  - 34 × programmable GPIOs
  - 12-bit SAR ADC up to 18 channels
  - 2 × 8-bit DACs
  - 10 × touch sensors (capacitive sensing GPIOs)
  - 4 × SPI
  - 2 × I²S interfaces
  - 2 × I²C interfaces
  - 3 × UART
  - SD/SDIO/CE-ATA/MMC/EMMC host controller
  - SDIO/SPI slave controller

- Ethernet MAC interface with dedicated DMA and planned IEEE 1588 Precision Time Protocol support
- CAN bus 2.0
- Infrared remote controller (TX/RX, up to 8 channels)
- Motor PWM
- LED PWM (up to 16 channels)
- Hall effect sensor
- Ultra low power analog pre-amplifier

- Security:
    - IEEE 802.11 standard security features all supported, including WPA, WPA2, WPA3 (depending on version)[5] and WLAN Authentication and Privacy Infrastructure (WAPI)
    - Secure boot
    - Flash encryption
    - 1024-bit OTP, up to 768-bit for customers
    - Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG)
- Power management:
    - Internal low-dropout regulator
    - Individual power domain for RTC
    - 5 μA deep sleep current
    - Wake up from GPIO interrupt, timer, ADC measurements, capacitive touch sensor interrupt

Applications of Node MCU

- Prototyping of IOT devices
- Low power battery operated applications
- Network projects
- Project requiring multiple I/O interfaces with Wi-Fi and Bluetooth functionalities.

3.4.2 OLED (Organic Light Emitting Diode)

Fig 3.4 Oled

An organic light-emitting diode (OLED or organic LED), also known as organic electroluminescent (organic EL) diode, is a light-emitting diode (LED) in which the emissive electroluminescent layer is a film of organic compound that emits light in response to an electric current. This organic layer is situated between two electrodes; typically, at least one of these electrodes is transparent. OLEDs are used to create digital displays in devices such as television screens, computer monitors, and portable systems such as smart phones and handheld game consoles. A major area of research is the development of white OLED devices for use in solid-state lighting applications. OLED is fundamentally different from LED which is based on a p-n diode structure. In LEDs doping is used to create p- and n- regions by changing the conductivity of the host semiconductor. OLEDs do not employ a p-n structure. Doping of OLEDs is used to increase radioactive efficiency by direct modification of the quantum-mechanical optical recombination rate. Doping is additionally used to determine the wavelength of photon emission. An OLED display works without a backlight because it emits visible light. Thus, it can display deep black levels and can be thinner and lighter than a liquid crystal display (LCD). In low ambient light conditions (such as a dark room), an OLED screen can achieve a higher contrast ratio than an LCD, regardless of whether the LCD uses cold cathode fluorescent lamps or an LED backlight.

FEATURES

OLED displays have many outstanding characteristics:

- Very saturated, vivid colors
- Deepest black, highest contrast ratio
- Wide viewing angle (180°)
- Low power consumption
- Extremely thin form factor, flexible
- Response time in microseconds for very crisp motion picture quality and 3D applications.

3.4.3 LITHIUM ION BATTERY

Fig 3.5 Lithium ion battery

Lithium is the lightest metal with the greatest electrochemical potential and the largest energy density per weight of all metals found in nature. Using lithium as the anode, rechargeable batteries could provide high voltage, excellent capacity and a remarkably high-energy density. However, lithium is inherently unstable, especially during charge. Therefore, lithium ions have replaced lithium metals in many applications because they are safer than lithium metals with only slightly lower energy density. Nevertheless, certain precautions should be made during charging and discharging. The Sony Corporation was the first company to commercialize the lithium-ion battery in 1991, which has since become popular and remains the best choice for rechargeable batteries. The lithium-ion battery requires almost no maintenance during its lifecycle, which is an advantage that other batteries do not have. No scheduled cycling is required, and there is no memory effect in the battery. Furthermore, the lithium-ion battery is well suited for electric vehicles because its self-discharge rate is less than half of the discharge rate of lead-acid and NiMH batteries. Despite the advantages of lithium-ion batteries, they also have certain drawbacks. Lithium ions are brittle. To maintain the safe operation of these batteries, they require a protective device to be built into each pack. This device, also referred to as the battery management system (BMS), limits the peak voltage of each cell during charging and prevents the cell voltage from dropping below a threshold during discharging. The BMS also controls the maximum charging and discharging currents and monitors the cell temperature.

LITHIUM BATTERY CHALLENGES

The operating temperature and voltage are the most important parameters that determine the performance of lithium-ion cells .Figure 1-1 and 1-2 shows that the cell operating voltage, current and temperature must be maintained within the area indicated by the green box labeled “Safe Operation Area” (SOA) at all times. The cell could be permanently damaged if it is operated outside the safety zone.

On the other hand, overly discharging the cells or storing the cells for extended periods of time would cause the cell voltage to fall below its lower limit, typically 2.5 V. This could progressively break down the electrode.

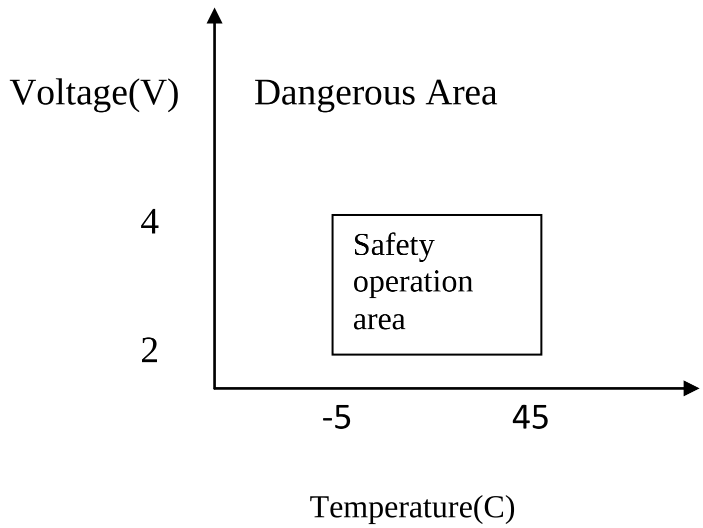

Figure 1-1 Lithium-ion cell operation window (Voltage)

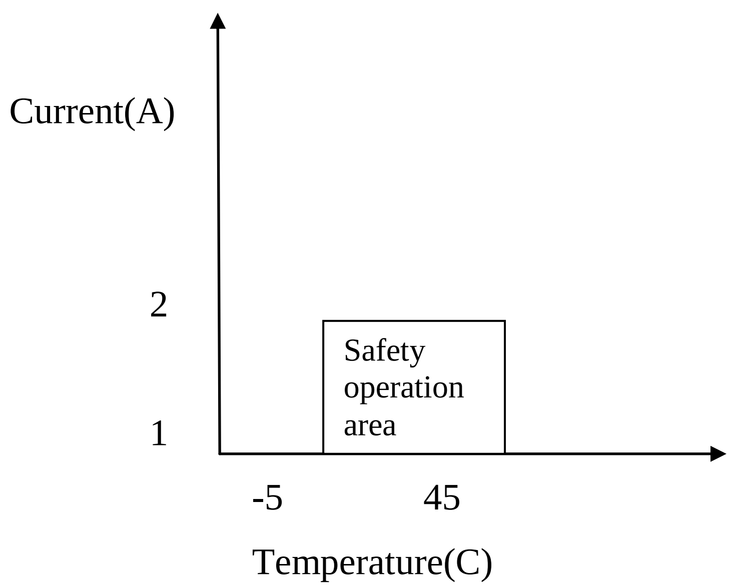

Figure 1-2 Lithium-ion cell operation window (Current)

The operating temperature of lithium-ion cells should be carefully controlled because excessively high or low temperatures could damage the cell. Temperature-related damages could be grouped into three types: low-temperature operational impact, high temperature operational impact and thermal runaway. While the effects of voltage and temperature on cell failures are immediately apparent, their effects on the lifecycle of the cells are not as obvious. However, the cumulative effects of these digressions may affect the lifetime of the cells. Figure 1-3 shows that the lifecycles of the cell would be reduced if its operating temperature falls below approximately 10 °C. Similarly, their lifecycles would be reduced if the cells were operated above 40 °C. Furthermore, thermal runaway would occur when the temperature reached 60 °C. The thermal management system, which is part of the BMS, must be designed to keep the cells operating within its limitation at all times.

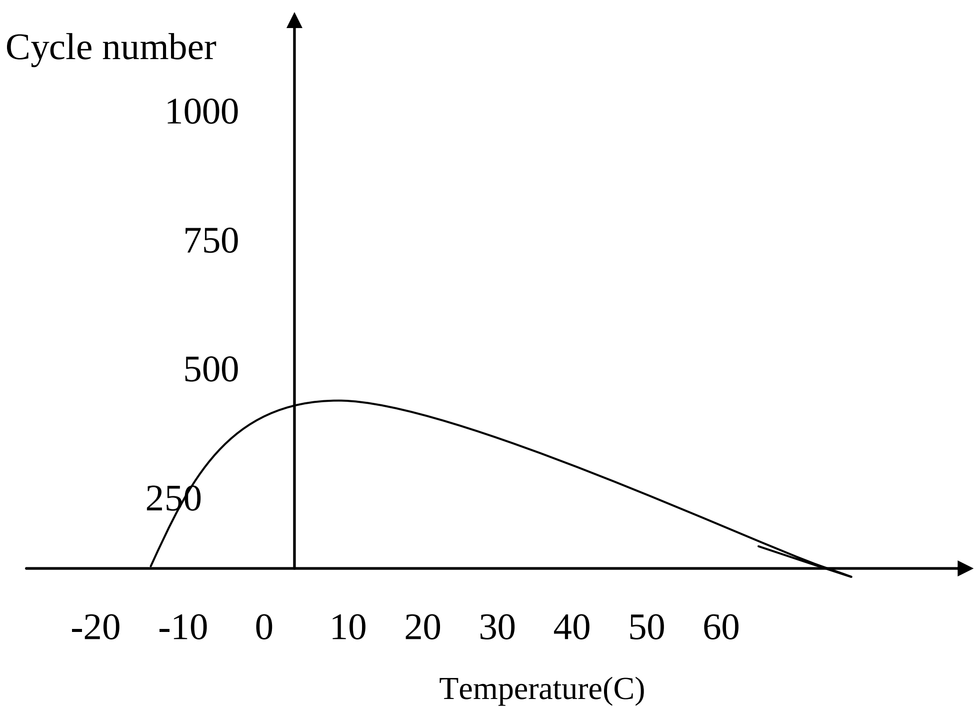

Figure 1-3 Lifecycle versus operating temperature of Li-ion cells

It is clear from the discussion above that the goal of the BMS is to keep the cells operating within their safety zone; this could be achieved using safety devices such as protection circuits and thermal management systems.

3.4.4 TEMPERATURE SENSOR

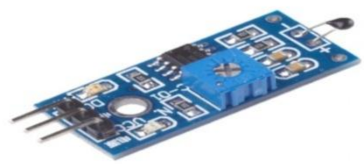

Fig 3.6 Temperature sensor

A temperature sensor is an electronic device that measures the temperature of its environment and converts the input data into electronic data to record, monitor, or signal temperature changes. Temperature sensors measure temperature readings via electrical signals. They contain two metals that generate an electrical voltage or resistance when a temperature change occurs. The sensor plays a vital role in maintaining a specific temperature for a variety of industries, including medical applications, HVAC systems, and electrical appliances in our homes. Temperature sensors are critical for accuracy and temperature control in industries like these. Temperature sensors work by measuring the voltage across the diode terminals. When the voltage increases, the temperature also increases, which is then followed by a voltage drop between the transistor terminals and the emitter (in a diode). Temperature sensors are extremely beneficial and necessary for a wide range of applications. In fact, you may even use them more often than you think! They are used for microwaves, refrigerators, and water heaters in your home. Temperature sensors are also used in the field; for example, thermometers play a vital role in geotechnical monitoring and renewable energy.

3.4.5 CURRENT SENSOR

Fig 3.7 Current sensor

A current sensor is a device that detects and converts current to an easily measurable output voltage, which is proportional to the current through the measured path. Current Sensor detects the current in a wire or conductor and generates a signal proportional to the detected current either in the form of analog voltage or digital output. Current Sensing is done in two ways – Direct sensing and Indirect Sensing. In Direct sensing, to detect current, Ohm's law is used to measure the voltage drop occurred in a wire when current flows through it.A current-carrying conductor also gives rise to a magnetic field in its surrounding. In Indirect Sensing, the current is measured by calculating this magnetic field by applying either Faraday's law or Ampere law. Here either a Transformer or Hall effect sensor or fiber optic current sensor are used to sense the magnetic field.

3.4.6 VOLTAGE SENSOR

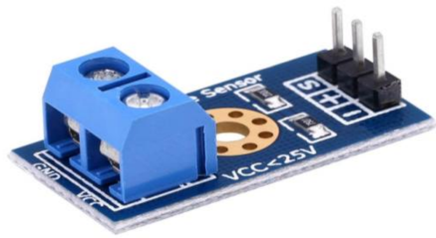

Fig 3.8 Voltage sensor

This sensor is used to monitor, calculate and determine the voltage supply. This sensor can determine the DC voltage level. The input of this sensor can be the voltage whereas the output is the switches, analog voltage signal, a current signal, an audible signal, etc. A voltage sensor can determine, monitor, and measure the supply of voltage. It can measure the AC level and/or DC voltage level. The input to the voltage sensor is the voltage itself, and the output can be analog voltage signals, switches, audible signals, analog current levels, frequency, or even frequency-modulated outputs. That is, some voltage sensors can provide sine or pulse trains as output, and others can produce amplitude modulation, pulse width modulation, or frequency modulation outputs. In voltage sensors, the measurement is based on a voltage divider.

3.4.7 RELAY

Fig 3.9 Relay

Relays are electric switches that use electromagnetism to convert small electrical stimuli into larger currents. These conversions occur when electrical inputs activate electromagnets to either form or break existing circuits.

- Relay works on the principle of electromagnetic induction.
- When the electromagnet is applied with some current, it induces a magnetic field around it.
- Above image shows working of the relay. A switch is used to apply DC current to the load.
- In the relay, Copper coil and the iron core acts as electromagnet.
- When the coil is applied with DC current, it starts attracting the contact as shown. This is called energizing of relay.
- When the supply is removed it retrieves back to the original position. This is called De energizing of relay.

There are also such relays, whose contacts are initially closed and opened when there is supply i.e. exactly to opposite to the above shown relay. Solid state relays will have sensing element to sense the input voltage and switches the output using opto-coupling.

3.4.8 BUZZER

Fig 3.10 Buzzer

The buzzer is a sounding device that can convert audio signals into sound signals. It is usually powered by DC voltage. It is widely used in alarms, computers, printers and other electronic products as sound devices. It is usually powered by DC voltage. It is widely used in alarms, computers, printers and other electronic products as sound devices. It is mainly divided into piezoelectric buzzer and electromagnetic buzzer, represented by the letter "H" or "HA" in the circuit. According to different designs and uses, the buzzer can emit various sounds such as music, siren, buzzer, alarm, and electric bell.

3.4.9 CHARGER MODULE

Fig 3.11 Charger module

A battery charger, or recharger, is a device that stores energy in a battery by running an electric current through it. The charging protocol (how much voltage or current for how long, and what to do when charging is complete) depends on the size and type of the battery being charged. Battery chargers come as simple, trickle, timer-based, intelligent, universal battery charger-analyzers, fast, pulse, inductive, USB based, solar chargers, and motion powered chargers. These battery chargers also vary depending on the applications like a mobile phone charger, battery charger for vehicles, electric vehicle batteries chargers and charge stations. Charging methods are classified into two categories: fast charge method and slow charge method. Fast charge is a system used to recharge a battery in about two hours or less than this, and the slow charge is a system used to recharge a battery throughout the night. Slow charging is advantageous as it does not require any charge detection circuit. Moreover, it is cheap as well. The only drawback of this charging system is that it takes maximum time to recharge a battery.

3.4.10 DC MOTOR

Fig 3.12 Dc motor

A DC motor is used to drive a mechanical load. In this lab, a separately excited DC generator provides the load. The load on the motor is adjusted by varying the generator field current. By increasing the field current of the DC generator, the load on the DC motor increases and thus the armature current increases. In general, DC motors are characterized by their torque speed curves. Since the measuring equipment for shaft torque is not available in the lab it is necessary to use alternative means of characterizing the DC motor. One alternative is to plot shaft speed versus armature current since torque is directly proportional to the armature current with a constant field current supplied to the motor. Shaft speed is also a function of the field current in a DC motor while maintaining a constant armature voltage as field current is directly proportional to the direct axis flux produced in the machine.

CONSTRUCTION

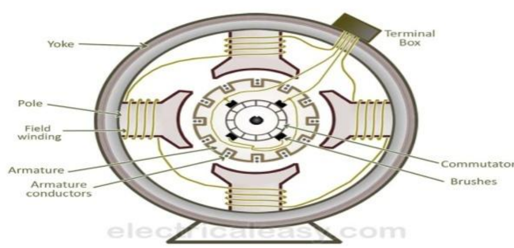

Fig 3.13 construction of dc motor

The stator of the DC motor has poles, which are excited by DC current to produce magnetic fields. In the neutral zone, in the middle between the poles, commutating poles are placed to reduce sparking of the commutator. The commutating poles are supplied by the DC current. Compensating windings are mounted on the main poles. These short-circuited windings damp rotor oscillations. The poles are mounted on an iron core that provides a closed magnetic circuit. The motor housing supports the iron core, the brushes and the bearings. The rotor has a ring shaped laminated iron core with slots. Coils with several turns are placed in the slots. The distance between the two legs of the coil is about 180 electric degrees. The coils are connected in series through the commutator segments. The ends of each coil are connected to a commutator segment. The commutator consists of insulated tube. Two brushes are placed in the neutral zone, where the magnetic field is close to zero to reduce arcing. The rotor has a ring shaped laminated iron core with slots. The commutator consists of insulated copper segments mounted on an insulted tube. Two brushes are pressed to the commutator to the adjacent coil, the switching requires the interruption of the coil current. The sudden interruption of an inductive current generates high voltages. The high voltage produces flash over and arcing between the commutator segments and the brush.

MOTOR DRIVER

Generally, even the simplest robot requires a motor to rotate a wheel or performs particular action. Since motors require more current then the microcontroller pin can typically generate, you need some type of a switch (Transistors, MOSFET, Relay etc..,) which can accept a small current, amplify it and generate a larger current, which further drives a motor. This entire process is done by what is known as a motor driver. Motor driver is basically a current amplifier which takes a low current signal from the microcontroller and gives out a proportionally higher current signal which can control and drives a motor. In most cases, a transistor can act as a switch and perform this task which drives the motor in a single direction. Turning a motor ON and OFF requires only one switch to control a single motor in a single direction. What if you want your motor to reverse its direction? The simple answer is to reverse its polarity. This can be achieved by using four switches that are arranged in an intelligent manner such that the circuit not only drives the motor, but also controls its direction.

3.5 SOFTWARE IMPLEMENTATIONS

3.5.1 ARDUINO IDE

The Arduino Integrated Development Environment- or Arduino software(IDE) – contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and series of menus. It connects to the Arduino and Genuion hardware to upload programs and communicate with them. Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an OLED, publishing something online.

You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing. Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike . As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IOT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

WRITING SKETCHES

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software(IDE), including complete error messages and other information. The bottom right hand corner of the windows displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor. NB: Versions of the Arduino Software(IDE) prior to 1.0 saved sketches with the extension .pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the .ino extension on save.

Programming steps

- **Upload –** Complies your code and uploads it to the configured board. See uploading below for details.

  Note: If you are using an external programmer with your board ,you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer".
- **New** –Creates a new sketch.
- **Open** –Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

  Note: Due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the **File | Sketchbook** menu instead.
- **Save –** Save your sketch.
- **Serial Monitor –**Opens the serial monitor.

Additional commands are found within the five menus: **File, Edit, Sketch, Tools** and **Help.** The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

FILE

- ***New***

  Creates the new instance of the editor, with the bare minimum structure of a sketch already in place.

- ***Open***

  Allows loading a sketch file browsing through the computer drives and the folders.

- ***Open Recent***

  Provides a short list of the most recent sketches, ready to be opened.

- ***Sketchbook***

  Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.

- ***Examples***

  Any examples provided by the Arduino Software(IDE) or library shows up in the menu item. All the examples are structured in a tree that allows easy access by topic or library.

- ***Close***

  Closes the instances of the Arduino Software(IDE) from which it is clicked.

- ***Save***

  Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as" window.

- ***Save as***

  Allows saving the current sketch with a different name.

- ***Page setup***

  It shows the Page Setup window for printing.

- ***Print***

  Sends the current sketch to the printer according to the setting defined in Page Setup.

- ***Preferences***

  Opens the preferences window where some setting of the IDE may be customized, as the language of the IDE interface.

- ***Quit***

  Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

EDIT

- ***Undo/Redo***

  Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.

- ***Cut***

  Removes the selected text from the editor and place it into the clipboard.

- ***Copy***

  Duplicates the selected text in the editor and places it into the clipboard.

- ***Copy for Forum***

  Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with the syntax.

- ***Copy as HTML***

  Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages**.**

- ***Paste***

  Puts the contents of the clipboard at the cursor position, in the editor.

- ***Select All***

  Selects and highlights the whole content of the editor

- ***Comment/Uncomment***

  Puts or removes the comments marker at the beginning of each selected line.

- ***Increases/Decrease Indent***

  Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.

- ***Find***

  Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.

- ***Find Next***

  Highlights the previous occurrence, if any of the string specified as the search item in the Find window relative to the cursor position.

SKETCH, TOOLS AND SKETCH BOOK

- ***Verify/Compile***

  Check your sketch for errors compiling it; it will report memory usage for code and variables in the console area.

- ***Upload***

  Compiles and loads the binary file onto the configured board through the configured port.

- ***Upload Using Programmer***

  This will over write the boot loader on the board; you will need to use Tools>Burn Boot loader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a tools. Burn Boot Loader command must be executed.

- ***Export Complied Binary***

  Saves .hex file that may be kept as archive or sent to the board using other tools.

- ***Show Sketch Folder***

  Opens the current sketch folder.

- ***Include Library***

  Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see libraries below. Additionally, from this menu item you can access the Library Manger and import new libraries from .zip files.

- ***Add File***

  Adds a source file to the sketch (it will be copied from its current location).The new file appears in a new tab in the sketch window.Files can be removed from the sketched using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right side of the toolbar.

TOOLS

- ***Auto Format***

  This formats your code nicely: i.e, indents it so that opening and closing curly braces line up and that the statements inside curly braces are indented more.

- ***Archive Sketch***

  Archives a copy of the currents sketch in .zip format. The archive is placed in the same directory as the sketch.

- ***Fix Encoding& Reload***

  Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.

- ***Serial Monitor***

  Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port.This usually resetsthe board, if the board supports Reset over serial port opening.

- ***Board***

  Select the board that you are using. See below for description of the various boards.

3.5.2. INTERNET OF THINGS

IOT devices are often called “smart” devices because they have sensors and can conduct complex data analytics. IOT devices collect data using sensors and offer services to the user based on the analyses of that data and according to user-defined parameters. For example, a smart refrigerator uses sensors (e.g., cameras) to inventory stored items and can alert the user when items run low based on image recognition analyses. Sophisticated IOT devices can “learn” by recognizing patterns in user preferences and historical use data. An IOT device can become “smarter” as its program adjusts to improve its prediction capability so as to enhance user experiences or utility. IOT devices are connected to the internet: directly, through another IOT device, or both. Network connections are used for sharing information and interacting with users. The IOT creates linkages and connections between physical devices by incorporating software applications. IOT devices can enable users to access information or control devices from anywhere using a variety of internet-connected devices. For example, a smart doorbell and lock may allow a user to see and interact with the person at the door and unlock the door from anywhere using a mobile device or computer.

UNDERSTANDING OF IOT

The Internet of Things (IOT) refers to the use of intelligently connected devices and systems to leverage data gathered by embedded sensors and actuators in machines and other physical objects. IOT is expected to spread rapidly over the coming years and this convergence will unleash a new dimension of services that improve the quality of life of consumers and productivity of enterprises, unlocking an opportunity that the GSMA refers to as the ‘Connected Life’. Machine to Machine (M2M) solutions - a subset of the IOT – already use wireless networks to connect devices to each other and the Internet, with minimal direct human intervention, to deliver services that meet the needs of a wide range of industries. In 2013, M2M connections accounted for 2.8% of global mobile connections (195 million), indicating that the sector is still at a relatively early stage in its development. An evolution of M2M, the IOT represents the coordination of multiple vendors’ machines, devices and appliances connected to the Internet through multiple networks To date, mobile operators have identified the following key distinctive features:

1. The Internet of Things can enable the next wave of life-enhancing services across several fundamental sectors of the economy.
2. Meeting the needs of customers may require global distribution models and consistent global services.
3. The Internet of Things presents an opportunity for new commercial models to support mass global deployments.
4. The majority of revenue will arise from the provision of value-added services and mobile operators are building new capabilities to enable these new service areas.
5. Device and application behaviour will place new and varying demands on mobile networks.

APPLICATIONS OF IOT

- Smart Door access control system
- Smart lighting for home and office
- Automated Gate and garage
- Smart thermostats and humidity controllers

SECURITY AND PRIVACY

Internet of Things privacy is the special considerations required to protect the information of individuals from exposure in the IOT environment. The data transmitted by a given endpoint might not cause any privacy issues on its own. However, when even fragmented data from multiple endpoints is gathered, collated and analyzed, it can yield sensitive information. The idea of networking appliances and other objects is relatively new, especially in terms of the global connectivity and autonomous data transfer that are central to the Internet of Things. As such, security has not traditionally been considered in product design, which can make even everyday household objects points of vulnerability.

CHAPTER 4

RESULTS

4.1 SOFTWARE RESULT

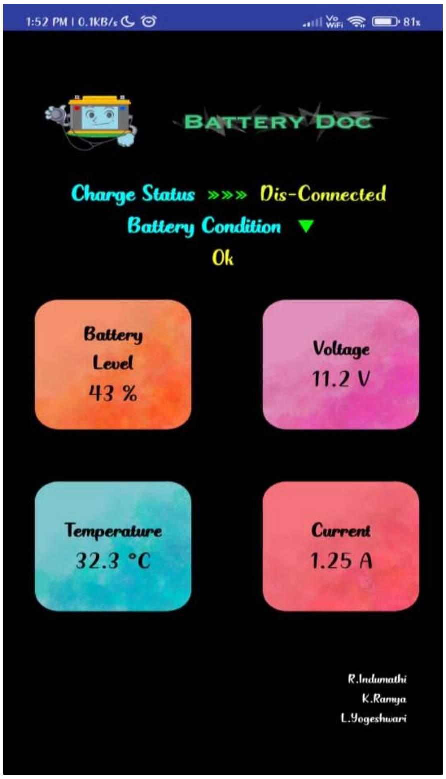

Fig.4.1 Output frame

- The information are collected from the sensors and transferred to the node.
- The temperature, voltage, current values is send to the node and the information are displayed in the mobile app.

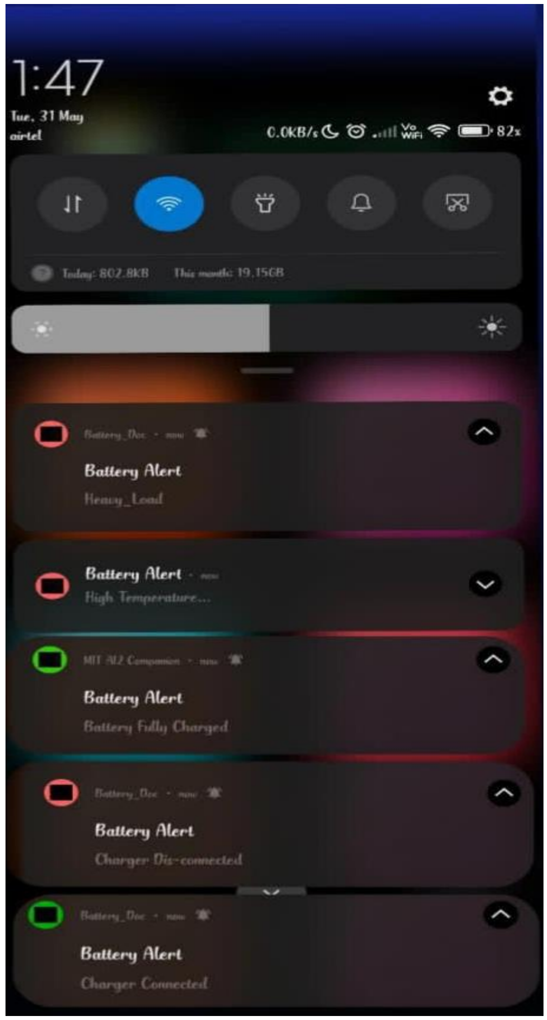

Fig 4.1 Output frame

If the parameters is high by using the IOT to send the notification to the mobile app. The purpose of monitoring the battery and reduce the heat of the battery.

Our project is implemented using C++ programming and the software is used to connect with the Node MCU (ESP32). The remaining charge capacity in a real-time scenario. The information collect from all the associated battery clients in the system is analyzed. The malfunction of the battery status are continuously monitored based on sudden charge & discharge voltage of battery bank and battery conditions are viewed in the cloud and mobile application with help of IOT module.

4.2 HARDWARE RESULTS

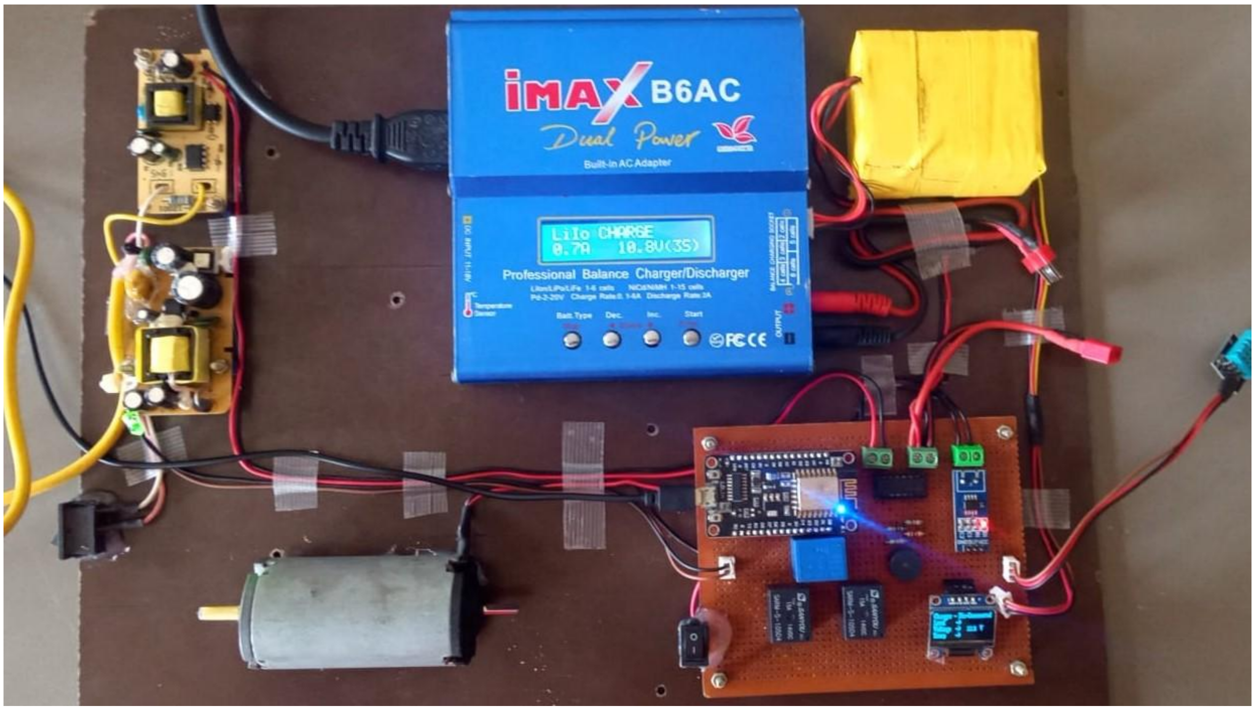

Fig 4.2 Input frame

The work has been implemented in real time using Node MCU (ESP32) module and OLED and project is working successfully.

CONCLUSION AND FUTURE WORK

After the implementation of smart battery system into the vehicle, it is easy for customer to view the battery parameters like voltage, temperature and charge capacity in the display on the battery. Mobile application and real time data base as well. Our smart battery system is going to enhance the battery life by monitoring each cells of the battery individually while charging. Its switching the charger when detecting the malfunction like overcharging, over discharging, overheating and it will push notification to the mobile app.Also it cut off the vehicle power when detecting over heating of battery pack and also it offs the charging when battery gets full.

Developments in the EV space are taking place so rapidly, and will continue to, that potential investors will have to make measured decisions so as to ensure their returns are not only limited to the near term. Not only is each stride towards truly zero-emission vehicles (ZEVs) a remarkable feat but also a step towards reducing the impact that human activity has had on the planet. Sustainable solutions towards energy storage on the whole, as well as EVs in particular, will offer resounding returns for generations to come and pave the way for a cleaner energy future wherein energy losses will be minimized to a great extent. We are talking about a future wherein sustainably generated energy powers electric vehicles for ranges of 1,000 km (621 miles) without ever having to be parked for more than a few minutes at a time. The same vehicle could then supplant the energy needs of the home or grid it draws power from. And all this from a battery pack that could possibly be as thin as the body panels of current cars and could prove even safer!

REFERENCES

1. Yong Tian, Dong Li, Jindong Tian, “An optimal nonlinear observer for state-of-charge estimation of lithium-ion batteries”, Industrial Electronics and Applications (ICIEA), 2017 12th IEEE Conference.

2. C. Wu, J. L. Sun, C. B Zhu, Y. W. Ge, Y. P. Zhao, "Research on overcharge and over discharge effect on Lithium-ion batteries", Proc. IEEE Vehicle Power Propulsions Conf., pp. 1-6, 2015.

3. H. V. Venkatasetty and Y. U. Jeong, “Recent advances in lithium-ion and lithium-polymer batteries,” in Proc. 17th Annu. Battery Conferences Applications and Advances, Jan. 2002, pp. 173–178.

4. S. Nakano, Y. Goto, K. Yokosawa, K. Tsukada, "Hydrogen Gas Detection System Prototype with Wireless Sensor Networks", Digest Tech. Papers IEEE Sensors 2005, pp. 159-162, October 31–November 3, 2005.

5. Fraiwan, L.; Lweesy, K.; Bani-Salma, A.; Mani, N. “A wireless home safety gas leakage detection system” in Proceedings of the 1st Middle East Conference on Biomedical Engineering (MECBME), Sharjah, United Arab Emirates, 21–24 February 2011; pp. 11–14.

6. O. A. Mohamad, R. T. Hameed, N. Tăapus, "Design and implementation of real time tracking system based on arduino intel galileo", Electronics Computers and Artificial Intelligence (ECAI) 2016 8th International Conference on, pp. 1-6, 2016.

7. Rasool Haaris, Rasool Aazim, Rasool Urfa, Raza Ali, Ahmad Waqar, "Centralized Environment and Battery Monitoring System for server rooms",Transportation Electrification Asia-Pacific (ITEC Asia-Pacific) 2014 IEEE Conference and Expo, pp. 1-5, Aug. 31 2014- Sept. 3 2014.

8.Atzori, L.; Iera, A.; Morabito, G. Understanding the Internet of Things: Definition, potentials, and societal role of a fast evolving paradigm. Ad Hoc Network 2017, 56, 122–140.

9.Back, J.A.; Tedesco, L.P.; Molz, R.F.; Nara, E.O.B. An embedded system approach for energy monitoring and analysis in industrial processes. Energy 2016, 115, 811–819.

10. Velandia, D.M.S.; Kaur, N.; Whittow, W.G.; Conway, P.P.; West, A.A. Towards industrial internet of things: Crankshaft monitoring, traceability and tracking using RFID. Robot. Computer Integration Manuf. 2016, 41, 66–77.

11.Xia, M.; Li, T.; Zhang, Y.; de Silva, C.W. Closed-loop design evolution of engineering system using condition monitoring through internet of things and cloud computing. Computer Network 2016, 101.

12. López-Benítez, M.; Drysdale, T.D.; Hadfield, S.; Maricar, M.I. Prototype for Multidisciplinary Research in the context of the Internet of Things. J. Network Computer Appl. 2017, 78, 146–161.

13.Xia, Z.; Su, H.; Liu, T. Remote Monitoring System of Lead-Acid Battery Group Based on GPRS. In Proceedings of the 2010 International Conference on Electrical and Control Engineering (ICECE), Wuhan, China, 25–27 June 2010; pp. 4023–4026.

APPENDIX

```
#include <SPI.h>
#include <Wire.h>
#include <DHT.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include "COOPBL5pt7b.h"
#include "SHOWG5pt7b.h"
#include "MTCORSVA5pt7b.h"

#define DHTPIN D3
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

#define OLED_RESET LED_BUILTIN //D4
Adafruit_SSD1306 display(OLED_RESET); //D4
#if (SSD1306_LCDHEIGHT != 32)
#error("Height incorrect, please fix Adafruit_SSD1306.h!");
#endif

#include <ESP8266WiFi.h>
#include <FirebaseArduino.h>
#define FIREBASE_HOST "batterydoc-3074c-default-rtdb.firebaseio.com"
#define FIREBASE_AUTH
"QZh6pMDYle9jG3YkLeZxVcrvuW5QoARLRPCmSVZ9"
#define WIFI_SSID "Battery_Doc" // Change the name of your WIFI
```

```
#defineWIFI_PASSWORD "12345678" // Change the password of your WIFI
//-------------------------------->GPIO_PINS
int Buzzer = D0;
//D1->OLED->SCK or SCL
//D2->OLED->SDA
//D3->Temperature Sensor
int Internet_Status = D4;
int Charge_Status = D5;
int Charge_Control= D6;
int VoltageCurr = D7;
int Battery_Control = D8;
//-------------------------------->Variables
int Read_Charge_Status;
float Read_Voltage;
float Read_Current;
const int OLED_Variables_Pos = 0;
const int OLED_Arrows_Pos = 50;
const int OLED_Values_Pos = 75;
String Status_String;

unsigned int total; // holds <= 64 analogReads
byte numReadings =64;
float offset = 575; // calibrate zero current
float span = 0.05; // calibrate max current | ~0.07315 is for 30A sensor
float currentin; // holds final current
float current_2Decimal;
```

```
float V_Max = 12.50;
int Battery_Level;
void setup()
{ pinMode(Internet_Status,OUTPUT);
pinMode(Battery_Control,OUTPUT);
pinMode(Charge_Control,OUTPUT);
pinMode(Charge_Status,INPUT);
pinMode(VoltageCurr,OUTPUT);

pinMode(Charge_Status,INPUT);
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(WHITE);
display.setFont(&COOPBL5pt7b);
dht.begin();

Serial.begin(115200);
pinMode(Charge_Status, INPUT);
pinMode(VoltageCurr, OUTPUT);
pinMode(Buzzer, OUTPUT);
digitalWrite(Buzzer, LOW);
WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
Serial.print("Connecting to ");
Serial.print(WIFI_SSID);
while (WiFi.status() != WL_CONNECTED)
  {Serial.print("."); delay(500);}
Serial.println();
```

```
Serial.print("Connected");
Serial.print("IP Address: ");
Serial.println(WiFi.localIP());
  //Firebase.begin(FIREBASE_HOST);
Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
}
int Read_Voltage_Interval()
 {
digitalWrite(VoltageCurr, HIGH);
delay(200);
    return analogRead(0);
}

int Read_Current_Interval() {
digitalWrite(VoltageCurr, LOW);
delay(200);
total = 0;
for (int i = 0; i<numReadings; i++) total += analogRead(A0);
    //  for (int i = 0; i<numReadings; i++) total += Read_Current_Interval();
currentin = (total / numReadings - offset) * span;
    return currentin;
}

void loop()
{
  if (WiFi.status() != WL_CONNECTED){
    //Serial.println("WiFi Not Connected");
```

```
digitalWrite(Internet_Status, HIGH); //Turn off WiFi LED
  }
else{
   //Serial.println("WiFi Connected");
digitalWrite(Internet_Status, LOW); //Turn on WiFi LED
//-------------------------DISPLAY VARIABLES & ARROWS-------------------------
}display.clearDisplay();
{
display.setCursor(OLED_Variables_Pos,6);
display.print("Charger");
display.setCursor(OLED_Variables_Pos,14);
display.print("Level");
display.setCursor(OLED_Variables_Pos,22);
display.print("Voltage");
display.setCursor(OLED_Variables_Pos,30);
display.print("Temp");
}
{
display.setCursor(45,6);
display.print("--");
display.setCursor(OLED_Arrows_Pos,14);
display.print("-->");
display.setCursor(OLED_Arrows_Pos,22);
display.print("-->");
display.setCursor(OLED_Arrows_Pos,30);
display.print("-->");
}
```

```
//----------------------------------------Charge Status-------------------------------------
   if (digitalRead(Charge_Status) == HIGH)
      {Status_String="Connected";
Firebase.setString ("Charge Status",Status_String);};
 if (digitalRead(Charge_Status) == LOW)
      {Status_String="Dis-Connected";
Firebase.setString ("Charge Status",Status_String);};
delay(100);
//----------------------------------Charge Status------------------------------------------
display.setCursor(55,6);
display.print(Status_String);
display.display();
delay(10);
//-----------------------------------------------------------------------------------------
  //Voltage Read
//-----------------------------------------------------------------------------------------
Read_Voltage = Read_Voltage_Interval();
delay(200);
Read_Voltage = Read_Voltage/77.5423729;
   float Voltage_1Decimcal = (roundf(Read_Voltage*10))/10;
   float Voltage_2Decimcal = (roundf(Read_Voltage*100))/100;
Firebase.setFloat ("Voltage",Voltage_1Decimcal);
delay(200);

//-----------------------------------------------------------------------------------------
//----------------------------------------Battery Level--------------------------------------
```

```
display.setCursor(OLED_Values_Pos,22);
display.print(Voltage_1Decimcal,1);
display.setCursor(OLED_Values_Pos+28,22);
display.print("V");
display.display();
delay(10);

//-------------------------------------------------------------------------------------------
  //Battery Level
//-------------------------------------------------------------------------------------------
if (Voltage_1Decimcal >= 10.50)
{ Battery_Level = 100-((V_Max - Voltage_2Decimcal)/.0235);
Firebase.setFloat ("Battery Level",Battery_Level);};
if (Voltage_1Decimcal < 10.50)
{ Battery_Level = 0;
Firebase.setFloat ("Battery Level",Battery_Level);};
//-------------------------------------------------------------------------------------------
//-----------------------------------Battery Level--------------------------------------
display.setCursor(OLED_Values_Pos,14);
display.print(Battery_Level);
display.setCursor(OLED_Values_Pos+18,14);
display.print("%");
display.display();                delay(10);

//-------------------------------------------------------------------------------------------
   //Temperature Read
```

```
//------------------------------------------------------------------------------------------------
float Read_Temperature = dht.readTemperature();
Firebase.setFloat ("Temperature",Read_Temperature);
delay(100);
//------------------------------------------------------------------------------------------------
//-----------------------------------------Temperature Read---------------------------------

display.setCursor(OLED_Values_Pos,30);
display.print(Read_Temperature,1);
display.setCursor(OLED_Values_Pos+28,30);
display.print("*C");
display.display();
delay(10);
//------------------------------------------------------------------------------------------------
  //Battery Level Functions
//------------------------------------------------------------------------------------------------
   if (Voltage_1Decimcal >= 12.5)
{ digitalWrite(Buzzer, HIGH);
digitalWrite(Charge_Control, LOW);
Firebase.setString ("Battery Condition","Fully_Charged");
delay (5000);};
if (Voltage_1Decimcal == 10.5)
{ digitalWrite(Buzzer, HIGH);
Firebase.setString ("Battery Condition","LOW_Battery");
delay (5000);};
 if (12.5 > Voltage_1Decimcal > 10.5)
```

```
{ digitalWrite(Buzzer, LOW);
Firebase.setString ("Battery Condition","Ok");
digitalWrite(Battery_Control, LOW);    //MOTOR ON
delay (5000);};
if (Voltage_1Decimcal < 10.5)
{ digitalWrite(Buzzer, HIGH);
digitalWrite(Battery_Control, HIGH);    //MOTOR OFF
Firebase.setString ("Battery Condition","Too_Low→Disconnected");
delay (5000);
digitalWrite(Battery_Control, HIGH);    //MOTOR OFF
    };
    if ((Voltage_1Decimcal < 12.5) && (digitalRead(Charge_Status) == HIGH))
    {digitalWrite(Charge_Control, HIGH);}
else{digitalWrite(Charge_Control, LOW);};
//--------------------------------------------------------------------------------------------
  //Temperature Functions
//--------------------------------------------------------------------------------------------
   if (Read_Temperature>= 36)
    {digitalWrite(Buzzer, HIGH);
Firebase.setString ("Battery Condition","High_Temperature");
    delay (5000);
digitalWrite(Battery_Control, HIGH);    //MOTOR OFF
digitalWrite(Charge_Control, LOW); // CHARGER OFF
digitalWrite(Buzzer, LOW);
    };
    if (Read_Temperature< 36)
    {digitalWrite(Buzzer, LOW);
```

```
digitalWrite(Battery_Control, LOW);
Firebase.setString ("Battery Condition","Ok");
};delay (20);
//----------------------------------------------------------------------------------------------
 //Current Read
//----------------------------------------------------------------------------------------------
Read_Current = Read_Current_Interval();
Serial.print(Read_Current);Serial.print("----");
currentin = currentin +1.4;
Serial.print("----");Serial.print(currentin);
if (currentin<0 ) {currentin = 0; Firebase.setFloat ("Current",currentin);};
if (currentin> 0)
    {
currentin = 5 * currentin;
Serial.print("----");Serial.print(currentin);
current_2Decimal = (roundf(currentin*100))/100;
Serial.print("----");Serial.println(current_2Decimal);
Firebase.setFloat ("Current",current_2Decimal);
    }; delay(200);
   //Current Functions
//----------------------------------------------------------------------------------------------
   if (current_2Decimal >= 2)
{ //Serial.println("-lkjhij--fedf-");
digitalWrite(Buzzer, HIGH);
Firebase.setString ("Battery Condition","Heavy_Load...Disconnecting");
delay (5000);
```

```
digitalWrite(Battery_Control, HIGH); //motor off
    };
if (current_2Decimal < 2)
    {digitalWrite(Buzzer, LOW);
Firebase.setString ("Battery Condition","Ok");
    // digitalWrite(Battery_Control, LOW);//motor on
     };
//---------------------------------------------------------------------------------------------
}
```