

PROJECT REPORT ON
SMART HAND GLOVES USING MQTT

A report submitted in
Partial fulfillment of the requirements

For the

PG-DIPLOMA IN EMBEDDED SYSTEM AND DESIGN

OFFERED BY

C-DAC HYDERABAD

BY

RAKESH HONNAPPAGOL (230950330033)

RANVIR SINGH (230950330034)

SAMARTH PATIL (230950330035)

SANDESH KALE (230950330036)

SANJAY CHAURASIA (230950330037)



ADVANCED COMPUTING TRAINING SCHOOL

C-DAC

HYDERABAD 500005

SEPT 2023

CERTIFICATE

This is certify that Rakesh Honnappagol, Ranvir Singh, Samarth Patil, Sandesh Kale, Sanjay Chaurasia has carried out the project work presented in this report entitled "Smart Hand Gloves" for the award of PG-DIPLOMA IN EMBEDDED SYSTEM AND DESIGN at **CDAC, HYDERABAD(TS)** under the guidance of Mr. V J Shravan .The project work and studies carried out by students themselves and the contents of the report do not form the basis for the award of any other degree to the candidate or to anybody else.

Mr. V J Shravan

(Project Mentor)

ACKNOWLEDGEMENT

SMART HAND GLOVES project has presented, an objective, a goal, a challenge accepted to help the society. This project marks the final hurdle that we tackle, of hopefully what would be one of the many challenges we have taken upon and am yet to take. However, we could not have made it without the support and guidance from the following. Firstly, I want to take this opportunity to have special thanks to our guide **Mr. V J Shravan** who helped us throughout this project by providing valuable guidance and advice as well as acquiring all components needed for this project to become a success.

(PG-DESD SEPT 2023)

Rakesh Honnappagol (230950330033)

Ranvir Singh (230950330034)

Samarth Patil (230950330035)

Sandesh Kale (230950330036)

Sanjay Chaurasia (230950330037)

TABLE OF CONTENTS

TOPIC	PAGE
Abstract	1
1 Introduction	1
1.1 Objectives	2
1.2 Hardware Requirements	2
2 Theory	3
2.1 IOT(Internet of things)	4
2.2 Features of IOT	6
2.3 Hardware Description	6
2.3.1 ESP-32	6
2.3.1.1 Pin Configuration	6-7
2.3.2 Flex Sensor	7-8
2.3.3 DF mini Player MP3	9
2.3.3.1 DF mini Player MP3 Pin Configuration	9-10
2.3.4 OLED	10-11
2.3.4.1 OLED Pin Configuration	11-12
2.3.5 Speaker	12
2.4 Software Tools Used	13
2.4.1 Arduini Ide	13
2.5 MQTT	14
3 Implementation	15
3.1 Block Diagram	15
3.2 Circuit Diagram	16
3.3 Working	16
3.4 Flow Chart	17
3.5 API'S	19
3.6 Applications	20
4 Conclusion	
4.1 Result	21
4.2 Limitations	21
4.3 Future Scope	21
4.4 Conclusion	21
5 References	22

ABSTRACT

Smart gloves represent a promising innovation in wearable technology, offering users a seamless interface to interact with digital devices and the Internet of Things (IoT) ecosystem. This abstract explores the development and implementation of smart gloves integrated with the Message Queuing Telemetry Transport (MQTT) protocol, aiming to facilitate intuitive communication and control in various domains

The integration of MQTT protocol into smart gloves establishes a robust communication framework, enabling seamless interaction between the gloves and MQTT-enabled devices or services. MQTT, known for its lightweight and efficient publish-subscribe messaging model, ensures real-time data transmission with minimal latency, ideal for the dynamic nature of smart glove applications.

Key functionalities of smart gloves leveraging MQTT encompass gesture recognition, sensor data collection, and device control. Through sophisticated algorithms and machine learning techniques, these gloves interpret hand gestures to trigger specific actions, such as controlling smart home devices, navigating virtual environments, or interacting with augmented reality interfaces.

Furthermore, MQTT facilitates bidirectional communication, allowing smart gloves to transmit sensor data to MQTT brokers for analysis and decision-making processes. This capability enhances user experience by providing contextual feedback or enabling remote monitoring in applications such as healthcare, sports performance analysis, or industrial automation.

The proposed implementation emphasizes interoperability and scalability, leveraging MQTT's widespread adoption across IoT ecosystems. By adhering to MQTT standards, smart gloves can seamlessly integrate with existing MQTT infrastructures, enabling cross-platform compatibility and interoperability with diverse IoT devices and services.

In conclusion, the integration of MQTT protocol into smart gloves presents a promising avenue for enhancing interaction and communication in various domains. By leveraging MQTT's lightweight and efficient messaging model, smart gloves offer users intuitive control and seamless integration with IoT ecosystems, unlocking new possibilities for immersive and interactive experiences.

CHAPTER 1

1 INTRODUCTION

"Introducing our revolutionary Smart Hand Gloves tailored for paralyzed patients, a pioneering solution designed to redefine accessibility and independence. Powered by advanced technology including MQTT for seamless communication, DFmini MP3 player for personalized audio feedback, ESP32 microcontroller for connectivity, and a flex sensor for precise gesture recognition, these gloves offer unparalleled functionality. With the inclusion of an OLED display and speaker, users receive real-time visual and auditory cues, enhancing their interaction with the world. Our gloves prioritize user empowerment, comfort, and ease of use, promising a transformative experience for paralyzed individuals seeking greater control and engagement in their daily lives."

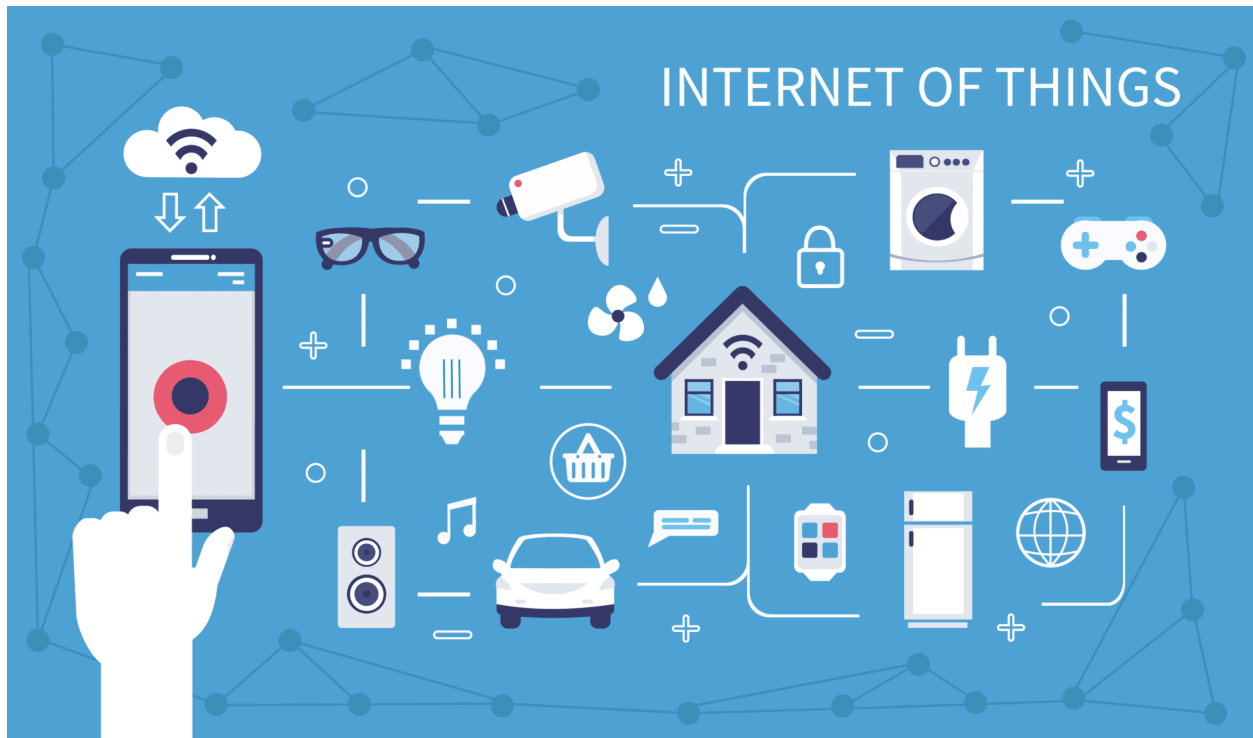


FIG. 1

1.1 OBJECTIVES

- Design a comfortable and ergonomic glove prototype capable of accommodating flex sensors.
- Integrate flex sensors into the gloves to accurately detect and measure finger and hand movements
- Develop software algorithms for processing sensor data and translating it into actionable commands
- Design and fabricate smart gloves with embedded flex sensors capable of detecting hand gestures accurately.
- Integrate MQTT protocol into the glove system for establishing communication with MQTT-enabled devices and systems
- Develop firmware and software for processing sensor data, translating gestures into MQTT messages, and handling MQTT communication
- Demonstrate the practical applications of gesture-based device control using MQTT protocol in real-world scenarios

1.2 HARDWARE REQUIREMENTS:

- ESP-32(WROOM DEV KIT)
- DF MINI PLAYER MP3
- FLEX SENSOR
- SPEAKER

CHAPTER 2

THEORY

2.1 IOT (INTERNET OF THINGS):

IOT as a term has evolved long way as a result of convergence of multiple technologies, machine learning, embedded systems and commodity sensors. IOT is a system of interconnected devices assigned a UIDS, enabling data transfer and control of devices over a network. It reduced the necessity of actual interaction in order to control a device. IOT is an advanced automation and analytics system which exploits networking, sensing, big data, and artificial intelligence technology to deliver complete systems for a product or service. These systems allow greater transparency, control, and performance when applied to any industry or system.

2.2 Features of IOT

2.2.1 Intelligence

IOT comes with the combination of algorithms and computation, software & hardware that makes it smart. Ambient intelligence in IOT enhances its capabilities which facilitate the things to respond in an intelligent way to a particular situation and supports them in carrying out specific tasks. In spite of all the popularity of smart technologies, intelligence in IOT is only concerned as a means of interaction between devices, while user and device interaction are achieved by standard input methods and graphical user interface

2.2.2 Connectivity

Connectivity empowers the Internet of Things by bringing together everyday objects. Connectivity of these objects is pivotal because simple object level interactions contribute towards collective intelligence in the IOT network. It enables network accessibility and compatibility in the things. With this connectivity, new market opportunities for the Internet of things can be created by the networking of smart things and applications

2.2.3 Dynamic Nature

The primary activity of Internet of Things is to collect data from its environment, this is achieved with the dynamic changes that take place around the devices. The state of these devices change dynamically, example sleeping and waking up, connected and/or disconnected as well as the context of devices including temperature, location and speed. In addition to the state of the device, the number of devices also changes dynamically with a person, place and time .

2.2.4 Enormous Scale

The number of devices that need to be managed and that communicate with each other will be much larger than the devices connected to the current Internet. The management of data generated from these devices and their interpretation for application purposes becomes more critical. Gartner (2015) confirms the enormous scale of IOT in the estimated report where it stated that 5.5 million new things will get connected every day and 6.4 billion connected things will be in use worldwide in 2016, which is up by 30 percent from 2015. The report also forecasts that the number of connected devices will reach 20.8 billion by 2020

2.2.5 Sensing

IOT wouldn't be possible without sensors that will detect or measure any changes in the environment to generate data that can report on their status or even interact with the environment. Sensing technologies provide the means to create capabilities that reflect a true awareness of the physical world and the people in it. The sensing information is simply the analog input from the physical world, but it can provide a rich understanding of our complex world

2.2.6 Heterogeneity

Heterogeneity in Internet of Things as one of the key characteristics Devices in IOT are based on different hardware platforms and networks and can interact with other devices or service platforms through different networks. IOT architecture should support direct network connectivity between heterogeneous networks. The key design requirements for heterogeneous things and their environments in IOT are scalabilities, modularity, extensibility and interoperability.

2.2.7 Security

IOT devices are naturally vulnerable to security threats. As we gain efficiencies, novel experiences, and other benefits from the IOT, it would be a mistake to forget about security concerns associated with it. There is a high level of transparency and privacy issues with IOT. It is important to secure the endpoints, the networks, and the data that is transferred across all of it means creating a security paradigm.

2.3 HARDWARE DESCRIPTION:

2.3.1 ESP-32:

Node MCU is an open source Lua based firmware for the ESP32 and ESP8266 WiFi SOC from Espressif and uses an on-module

Flash - based SPIFFS file system. Node MCU is implemented in C and is layered on the Espressif ESP-IDF.

The firmware was initially developed as is a companion project to the popular ESP8266-based Node MCU development modules, but the project is now community-supported, and the firmware can now be run on *any* ESP module.

Support for the new ESP32 WiFi/BlueTooth SOC from Espressif is under way.

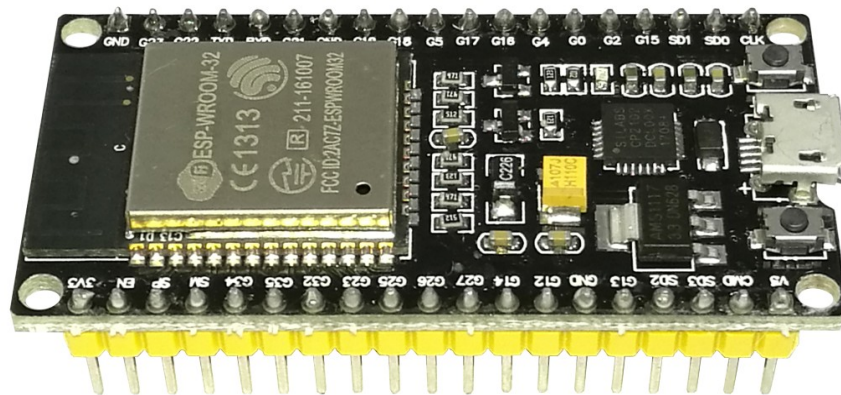


FIG. 2.3.1

2.3.1.1 Pin Configuration:

ESP-32 dev kit has 38 pins and 19 on each side of the board as shown in the picture above. It has 34 GPIO pins and each pin has multiple functionalities which can be configured using specific registers. There are many types of

GPIOs available like digital input, digital output, 7 analog input, and analog output, capacitive touch, UART communication and many other features mentioned above.

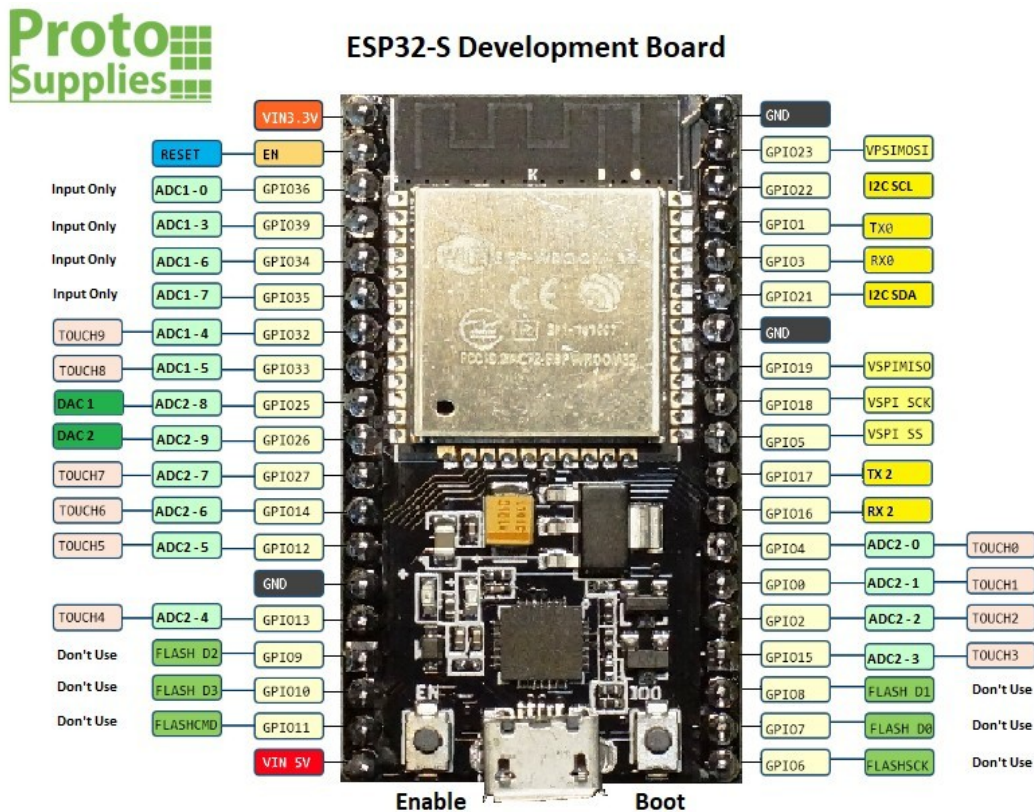


FIG. 2.3.1.1

2.3.2 FLEX SENSOR:

The conductive ink on the sensor acts as a resistive element. In its straight position, the sensor exhibits a resistance of approximately 32.5k ohms.

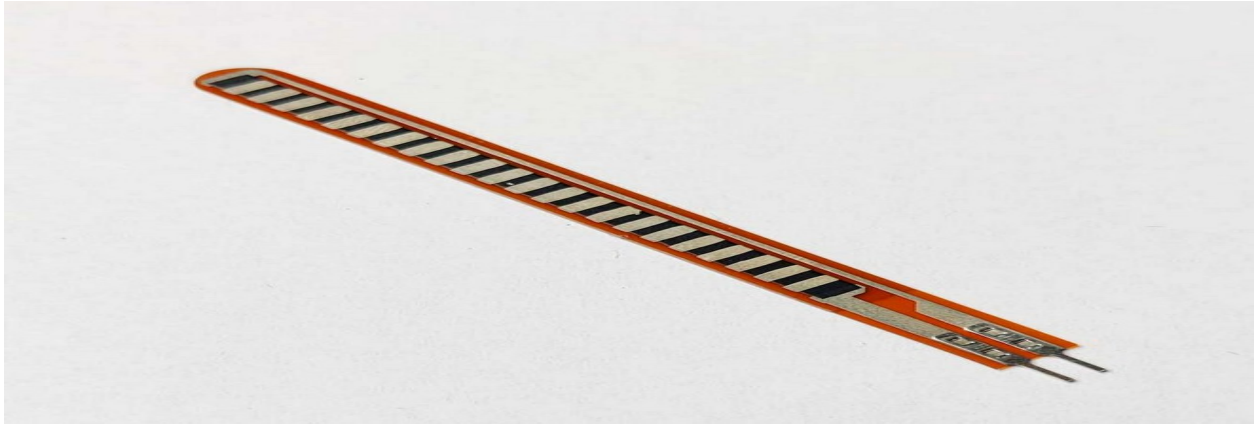
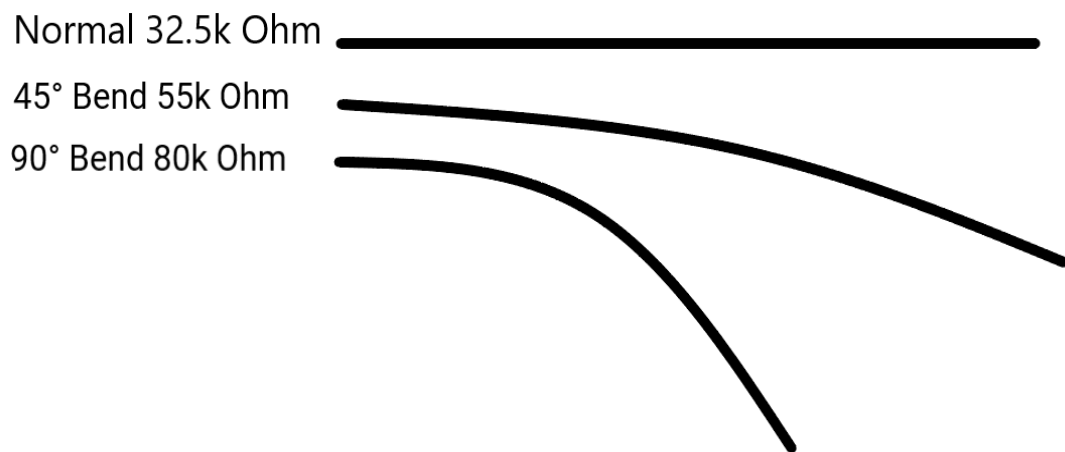


FIG. 2.3.2

Bending the sensor causes the conductive layer to stretch, leading to a decrease in cross-sectional area and an increase in resistance. At a 45° angle, this resistance is approximately 55K. At a 90° angle, this resistance is approximately 80K. When the sensor is straightened, the resistance reverts back to its initial value. By measuring the resistance, it is possible to determine the degree of bending of the sensor.



2.3.3 DF MINI PLAYER MP3:

The DFPlayer Mini module is a compact, serial MP3 player that offers seamless integration and hardware decoding for MP3 and WMV files. Its software supports TF card drivers and is compatible with FAT16 and FAT32 file systems. With straightforward serial commands, users can effortlessly specify music playback and access various functions without dealing with complex underlying operations. Its key features include ease of use, stability, and reliability, making it an ideal choice for projects requiring audio playback with minimal setup and hassle

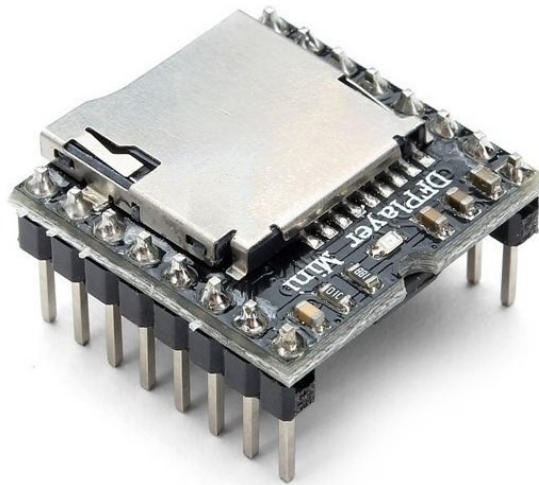


FIG. 2.3.3

2.3.3.1 DF MINI PLAYER MP3 PIN CONFIGURATION:

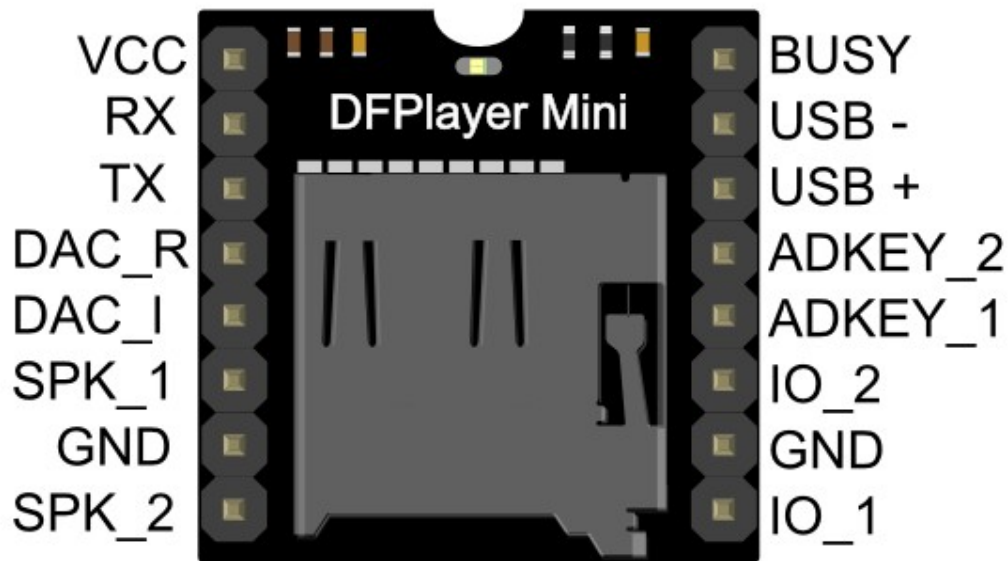


FIG 2.3.3.1

2.3.4 OLED:

OLED (Organic Light-Emitting Diode) displays utilize organic compounds to emit light when an electric current is applied. Unlike traditional LCD displays, OLEDs do not require a backlight; instead, each pixel emits its own light. This allows for true blacks, vibrant colors, and improved contrast ratios. OLED displays can be thin, flexible, and energy-efficient, consuming power only when emitting light. There are two main types: passive matrix and active matrix OLEDs (AMOLED), with AMOLED being commonly used in modern devices like smartphones and TVs due to faster response times and superior image quality. Overall, OLED displays offer benefits including vibrant colors, high contrast ratios, energy efficiency, and thin form factors, making them suitable for various applications.



FIG. 2.3.4

2.3.4.1 OLED PIN CONFIGURATION:

- Total pins are 4.
- GND,VCC,SCL,SDA
- Operating voltage 3~5v

SSD1306 Pinout

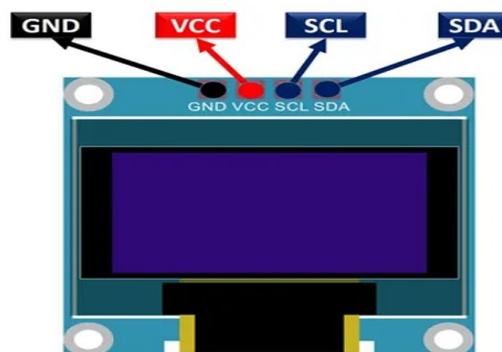


FIG. 2.3.4.1

2.3.5 SPEAKER:

- 3 Watt speaker
- Operating current 1 ohm speaker



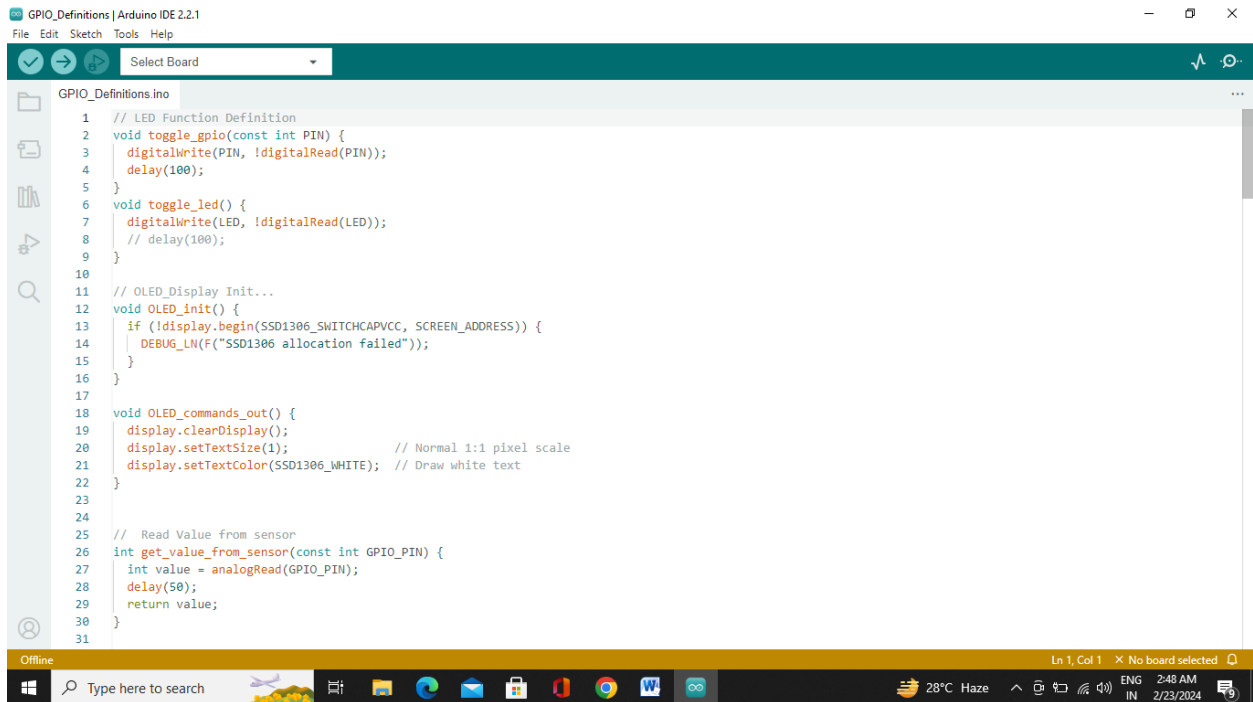
FIG. 2.3.5

2.4 SOFTWARE TOOLS USED:

2.4.1 Arduino ide:

The Arduino Integrated Development Environment (IDE) is a software application used to write, compile, and upload code to Arduino microcontroller boards. The Arduino IDE provides an easy-to-use interface for programming Arduino boards, and it is available for Windows, Mac OS X, and Linux operating systems. The IDE is based on the Processing programming language and includes a code editor, a compiler, a linker, and a serial monitor. It also includes a library manager, which allows users to easily add third-party libraries to their projects. The Arduino IDE supports a simplified version of the C++ programming language, which is easy to learn even for beginners with no programming experience. The code is uploaded to the Arduino board through a USB port or another type of serial connection. The Arduino IDE is an open-source project,

and its source code is available on GitHub for anyone to modify and contribute to. The Arduino community has created a vast number of libraries and examples, which can be accessed through the Arduino IDE's Library Manager.



```
1 // LED Function Definition
2 void toggle_gpio(const int PIN) {
3     digitalWrite(PIN, !digitalRead(PIN));
4     delay(100);
5 }
6 void toggle_led() {
7     digitalWrite(LED, !digitalRead(LED));
8     // delay(100);
9 }
10
11 // OLED_Display Init...
12 void OLED_init() {
13     if (!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
14         DEBUG_LN(F("SSD1306 allocation failed"));
15     }
16 }
17
18 void OLED_commands_out() {
19     display.clearDisplay();
20     display.setTextSize(1);           // Normal 1:1 pixel scale
21     display.setTextColor(SSD1306_WHITE); // Draw white text
22 }
23
24
25 // Read Value from sensor
26 int get_value_from_sensor(const int GPIO_PIN) {
27     int value = analogRead(GPIO_PIN);
28     delay(50);
29     return value;
30 }
31
```

FIG. 2.4.1

2.5 MQTT:

MQTT is a lightweight messaging protocol for IOT, featuring a publish - subscribe model with a central broker. Messages are organized into topics, and subscribers receive messages based on their topic interests. It supports three levels of Quality of Service (QOS) for message delivery reliability and operates efficiently in resource-constrained environments. MQTT includes features like a keep-alive mechanism, TLS security, and retained messages for enhanced functionality and reliability.

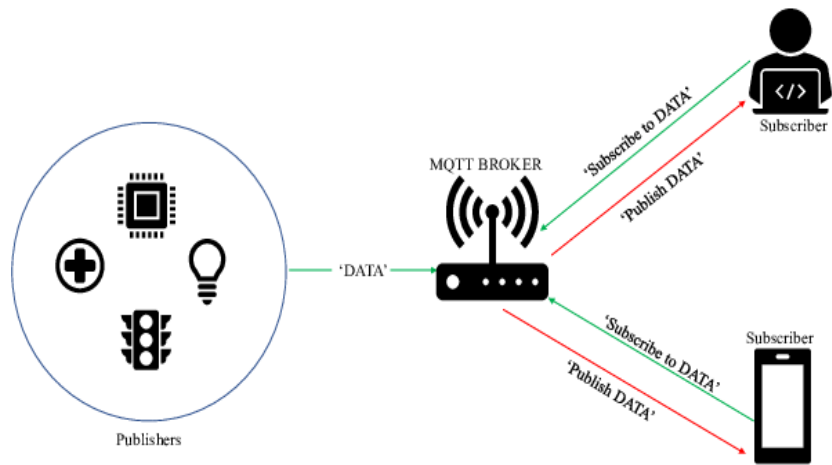


FIG. 2.5

CHAPTER 3

Implementation

3.1 BLOCK DIAGRAM:

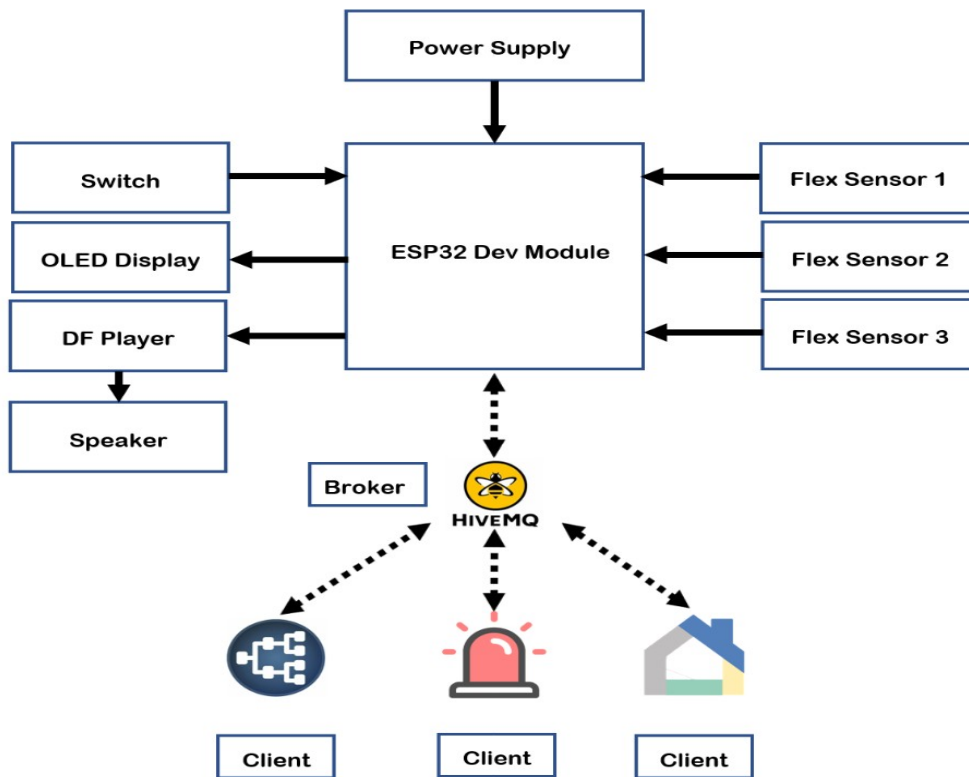


FIG. 3.1

3.2 CIRCUIT DIAGRAM:

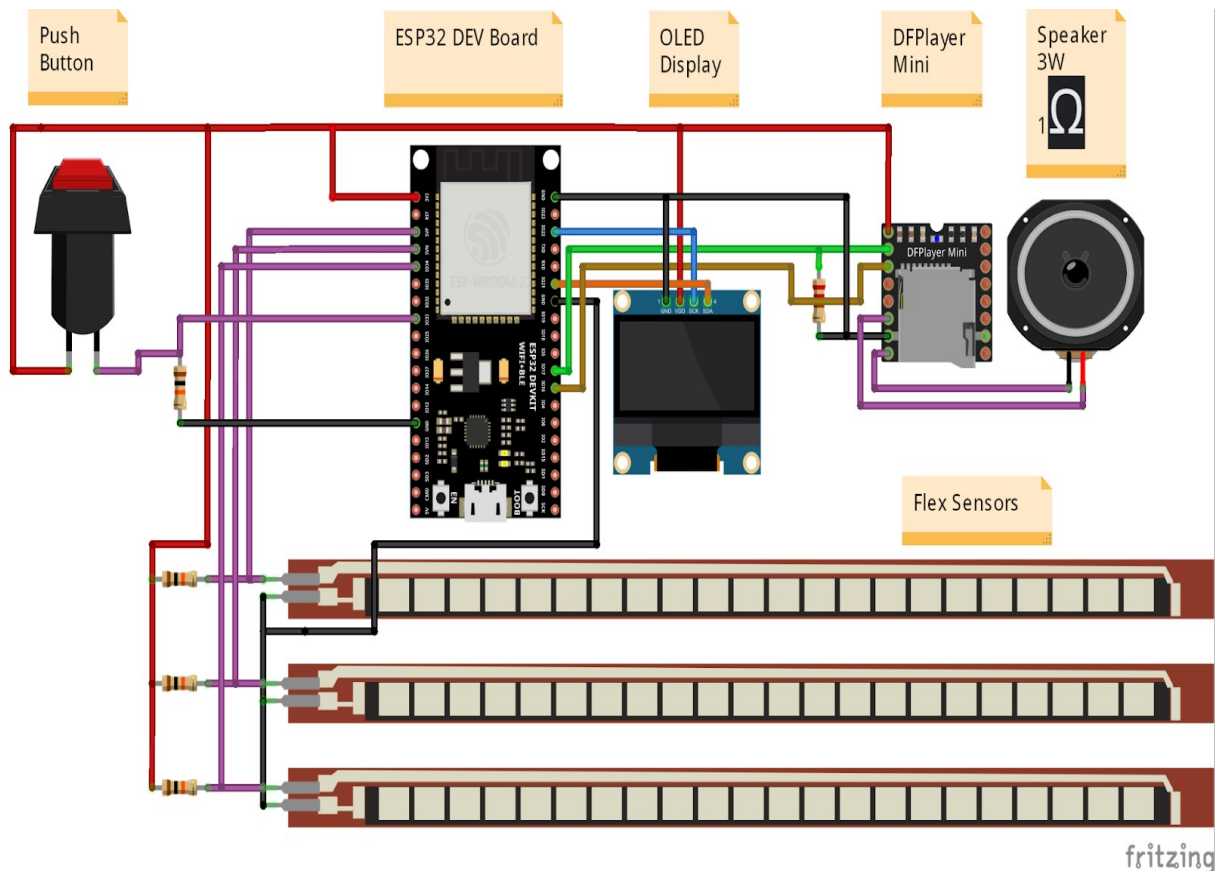


FIG. 3.2

3.3 WORKING:

The ESP32 microcontroller, embedded within smart hand gloves, coordinates communication between components. It interfaces with the DF Mini Player MP3 module to manage audio playback stored on an SD card, delivering sound through a speaker. An OLED display integrated into the gloves provides visual feedback and instructions. MQTT, via the Mosquitto broker, facilitates remote control and monitoring of the gloves' functions, allowing caregivers to adjust settings or receive alerts remotely. This system assists

paralyzed patients by providing audio cues and visual prompts, enhancing communication and accessibility.

3.4 FLOW CHART:

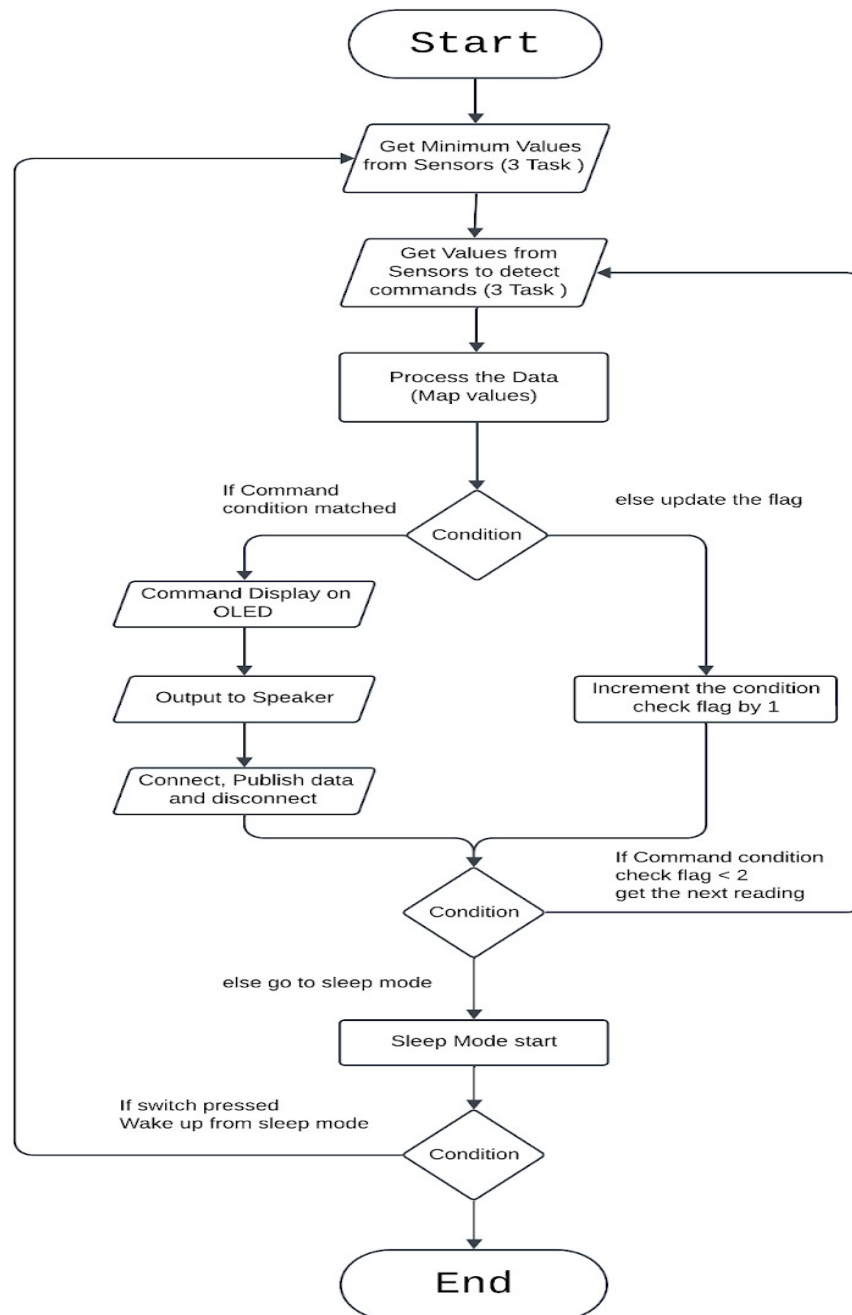


FIG. 3.4

3.5 API'S:

- Functions toggle GPIO pins and LEDs, respectively.

```
toggle_gpio()  
toggle_led()
```

- Function initializes the OLED display.

```
OLED_init()
```

- Function prepares the OLED display for output by clearing any existing content and setting text size and color parameters to default values. This function is typically called before displaying any information on the OLED screen.

```
OLED_commands_out()
```

- Function reads the analog value from a sensor connected to the specified GPIO_PIN.

```
get_value_from_sensor()
```

- Function initializes smart gloves by setting up tasks to obtain minimum values from sensors.

```
init_smart_gloves()
```

- This message serves to provide information to developers or users about the current state or operation of the program.

```
DEBUG_LN("Initialize Minimum Value for Sensor");
```

- Function is used to create a new task that will execute concurrently with other tasks in the system.

```
xTaskCreatePinnedToCore(get_test_value_from_sensor_1,  
"Get Sensor 1 Min Value", 1024, &MIN_1, 5, &task_1,  
CORE);
```

- To Map Values

```
map(value_1, MIN_1, Sense_MAX, 0, MAX_MAPPED);
```

- Function evaluates the sensor readings to determine the conditions for motion detection. Based on the sensor values, it decides which track to play via the DFPlayer.

```
check_conditions_for_motions()
```

- The "RTC_DATA_ATTR" attribute is typically used in ESP32 projects to store variables in the RTC (Real-Time Clock) memory, which retains data even when the device is powered off. This allows the variable "value" to retain its value across power cycles or resets.

```
#include "Library.h"
RTC_DATA_ATTR int value = 0;
```

- configures the ESP32 to wake up from deep sleep mode upon receiving a signal on GPIO pin 33. The second argument, 1, specifies that the wake-up signal should trigger on the rising edge of the signal.

```
esp_sleep_enable_ext0_wakeup(GPIO_NUM_33, 1);
```

- instructs the ESP32 microcontroller to enter deep sleep mode. Once called, the microcontroller will enter a low-power state where most of its operations are halted, consuming minimal power.

```
esp_deep_sleep_start();
```

- Function is a callback handler typically used in MQTT (Message Queuing Telemetry Transport) implementations. It is invoked when a message is received from the MQTT broker. Here's a brief explanation of its parameters:

```
callback(char* topic, byte* payload, unsigned int
length)
```

- Defines the MQTT topic for the message as "command"

```
String topic = "command";
```

- Converts the message from a string (in the "messages" array at index "i") to a C-style string (const char*) using the `c_str()` method.

```
const char* message = messages[i].c_str();
```

3.6 Applications:

- **Government Sector:** The smart glove can be used in banks, hospitals, railway stations, airports and restaurants where the hand gestures are recognized and real time translation is made which helps in easier communication.
- **Medical Sector:** The gesture recognition project can be used for communicating with the sick and disabled patients with minimal effort from them. It reduces the need for 24/7 presence of a helper around them since it can be used to monitor patients' needs remotely
- **Educational Sector:** Differently-abled children can learn sentence formations, prepositions, grammar, with the help of this glove.
- **Industrial Sector:** Vast industrial floors often find themselves being extremely unfavourable for vocal or visual communication. Our gloves help solve this issue by helping people communicate about the various activities that happen on the floor regardless of the distance or background noise levels.
- **Virtual Reality Sector:** The sensory glove is implemented in interactive applications which also include much other virtual reality technology, such as a computer games, simulated environment, wearable mouse glove, virtual musical appliances, wearable keyboard glove, etc

CHAPTER 4

CONCLUSION:

4.1 Result:

The smart hand gloves prototype successfully integrated MQTT communication, enabling using switch (**for wake up**). The circuit diagram was faithfully implemented, leading to expected results.

4.2 Limitation:

Smart hand gloves for paralyzed patients using MQTT face challenges such as complexity in setup, high costs, reliability issues with Wi-Fi, power dependency, mobility limitations, speech recognition accuracy, privacy/security concerns, compatibility issues, and maintenance needs.

4.3 Future Scope:

Future scope for smart hand gloves for paralyzed patients using MQTT includes advancements in simplicity, cost-effectiveness, reliability, mobility assistance, improved speech recognition, enhanced privacy/security measures, broader device compatibility, and streamlined maintenance protocols.

4.4 Conclusion:

In conclusion, smart hand gloves for paralyzed patients using MQTT offer promising potential in enhancing accessibility and control over devices. However, challenges such as complexity, cost, reliability, and mobility limitations must be addressed for widespread adoption and effectiveness.

REFERENCES

1. Basics of Arduino <https://www.arduino.cc/reference/en/>
2. Basics of MQTT <https://mqtt.org/>
3. Basics of ESP-32
<https://www.espressif.com/en/products/socs/esp32>
4. Interface flex sensor
<https://circuitdigest.com/microcontroller-projects/interfacing-flex-sensor-with-arduino>
5. https://www.youtube.com/watch?v=cHGcoPfpP_w
6. Interface DF mini player MP3
<https://maker.pro/arduino/projects/how-to-use-the-dfmini-player-mp3-module-with-arduino>
7. https://wiki.dfrobot.com/DFPlayer_Mini_SKU_DFR0299
8. For simulation <https://wokwi.com/>