International Journal of Innovative Research in Management Studies (IJIRMS)
Volume 7, Issue 6, July 2025. pp.286-315.

OFFLINE AUTOMATED VOICE CONTROL MULTI-TASKING HOME AUTOMATION USING AI THINKER VC-02 MODULE

D.Vengadeshwari[1], Dr.A.Vengatesh A[2], R.Praveen Kumar[3], Dr.M.Malarvizhi[4]

[1]PG Scholar, Department of Embedded System Technologies, Gnanamani College of Technology (Autonomous), Namakkal. Email ID: keerthirthi317@gmail
[2, 3, 4]Assistant Professor, Department of EEE, Gnanamani College of Technology (Autonomous), Namakkal.

Abstract—*The AI Thinker VC-02 is an advanced offline voice control module tailored for smart home applications. Powered by AI-based speech recognition technology and a high-performance DSP chip, it supports local voice command processing without internet connectivity. This eliminates latency and privacy concerns typical of cloud-dependent systems. The study proposes a multitasking smart home automation system using the VC-02 module, focusing on seamless device control, energy efficiency, and offline operability. The system proves ideal for modern smart homes, offering a cost-effective, secure, and scalable automation solution.*

Keywords: *AI Thinker, Offline Voice Control Module, DSP Chip, Smart Home Automation System, Modern Smart Homes.*

INTRODUCTION

The AI Thinker VC-02 is a cutting-edge voice control module that plays a transformative role in the field of home automation. As the demand for smart homes continues to grow, voice control has emerged as a key interface for creating seamless and intuitive user experiences. The VC-02 module offers a powerful offline voice recognition system, enabling users to control appliances, lighting, security systems, and more through simple voice commands—without relying on internet connectivity. Designed with a high-performance Digital Signal Processor (DSP), the VC-02 delivers fast and accurate recognition of predefined commands, ensuring reliability and privacy in voice-controlled systems. Its offline functionality not only eliminates latency issues common with cloud-based solutions but also enhances data security by processing commands locally. These features make it particularly well-suited for smart homes where consistent and secure operation is essential. The VC-02 integrates easily into home automation platforms, enabling devices to respond to commands like turning on lights, adjusting the thermostat, opening curtains, or activating security systems. It supports customizable wake words and commands, allowing users to tailor their smart home experiences according to personal preferences. With its compact size, low power consumption, and developer-friendly environment, the VC-02 is a versatile solution for manufacturers and developers aiming to create innovative and user-centric home automation products. This document delves into the features and benefits of the AI Thinker VC-02, focusing on its role in enhancing home automation. It highlights technical specifications, such as its low power consumption, voice recognition accuracy, and compatibility with popular microcontrollers like ESP32 and STM32. The VC-02 serves as an essential building block for developers and manufacturers aiming to create innovative, voice-controlled smart home devices that improve convenience, security, and personalization for end users.

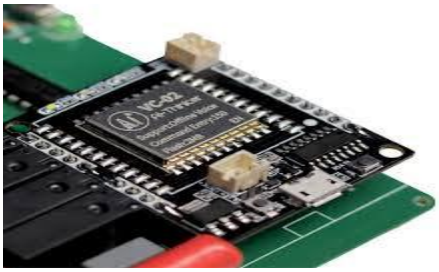


fig 1.1 AI Thinker VC 02

OBJECTIVES

1. **Enhance User Convenience**

   Enable seamless voice control for smart home devices, reducing the need for manual interaction and making everyday tasks more intuitive and efficient.

2. **Enable Offline Voice Recognition**

   Provide accurate and fast voice command processing without relying on cloud-based systems, ensuring consistent performance in environments with limited or no internet connectivity.

3. **Improve Data Privacy and Security**

   Process all voice commands locally on the device to protect user data and eliminate privacy concerns associated with cloud-based solutions.

4. **Facilitate Easy Integration**

   Offer a compact and developer-friendly module that can be easily embedded into a variety of home automation devices, such as smart lights, appliances, thermostats, and security systems.

5. **Support Customization**

   Allow developers to configure wake words and voice commands to meet specific user requirements, creating personalized and adaptive smart home experiences.

6. **Promote Energy Efficiency**

   Provide a low-power voice control solution that is suitable for battery-operated and energy-conscious devices within the smart home ecosystem.

7. **Expand Smart Home Functionality**

   Enable voice control for a wide range of devices, such as motorized blinds, smart locks, entertainment systems, and more, fostering a fully integrated smart home environment.

8. **Simplify Prototyping and Development**

   Offer tools and libraries for rapid development, enabling manufacturers to bring innovative, voice-controlled smart home devices to market quickly.

9. **Ensure Scalability**

   Design the module to support future upgrades and additional commands, making it adaptable for evolving technologies and user demands in the smart home industry.

10. **Drive Adoption of Voice AI**

    Serve as a cost-effective and versatile solution for manufacturers and developers, accelerating the adoption of voice-controlled technologies in the smart home market.

EXISTING SYSTEM

Home Automation Using Ardunio

Arduino UNO will control devices and reads sensor data. The figure "Room Architecture" depicts how the Arduino UNO will connects with the devices and sensors. Room have multiple controllable devices(i.e. Light(s), Fan, Wall Socket(s), etc.), one PassiveIR (to detect human presence in the room), one temperature sensor (LM35 to collect room temperature) and LDR (to detect light intensity near room window).

The current systems for home automation and appliance control largely rely on manual or semi-automated operations. These systems may include traditional wired automation setups or standalone smart devices, but they often have several limitations:

1. **Manual or Semi-Automated Control**

   Existing systems mostly require manual intervention to control home appliances, either through physical switches or limited remote-control functionality via basic devices like IR remotes. This approach is neither efficient nor user-friendly.

2. **Lack of Integration**

   Many existing home automation systems operate in isolation. For example, standalone devices like smart lights, thermostats, or security cameras function independently without a central platform for unified control.

3. **Limited Connectivity**

   Traditional systems often lack the capability to connect to the internet or communicate with other devices. This restricts remote accessibility and real-time monitoring.

4. **No Real-Time Feedback**

   Existing setups rarely provide real-time status updates of appliances or environmental conditions, making it difficult for users to stay informed about their home environment when they are away.

5. **Low Energy Efficiency**

   Without intelligent automation based on sensor data, energy is often wasted. For instance, lights or appliances may remain on unnecessarily when no one is present.

LITERATURE SURVEY

Voice recognition and control systems have been gaining significant traction as integral components of modern home automation technologies. The integration of voice AI into smart home devices has been extensively studied and implemented in recent years, focusing on improving accuracy, reliability, and user convenience. This literature survey highlights previous works and developments relevant to the **AI Thinker VC-02** module's role in offline voice recognition for home automation.

1. Evolution of Voice-Controlled Home Automation

Research on voice-based home automation systems dates back to the early 2000s, where initial prototypes relied heavily on cloud computing for processing. Cloud-dependent systems like Amazon Alexa and Google Assistant demonstrated the potential of natural language processing (NLP) in controlling home appliances. However, concerns regarding data privacy, latency, and dependency on constant internet connectivity have led to the development of offline, edge-based voice recognition solutions.

- **Study by Shen et al. (2019):**

  Proposed an offline voice control framework for smart home devices using lightweight DSP chips to enable real-time command processing. This approach significantly reduced latency compared to cloud-based systems, making it suitable for low-power IoT devices.

- **Application of Embedded Systems (Wu et al., 2020):**

  Highlighted the integration of voice modules into embedded systems such as ESP32 and STM32 for home automation. The study emphasized the importance of low-power consumption and modularity in creating scalable voice control solutions.

2. Offline Voice Recognition Technology

Offline voice recognition has emerged as a promising alternative to cloud-based systems, addressing critical challenges such as privacy, reliability, and cost-efficiency. Key developments in this area include:

- **Speech Recognition Accuracy (Zhao et al., 2021):**

  Explored DSP-based speech recognition algorithms that achieve high accuracy even in noisy environments. Their work highlighted the role of preprocessing techniques like noise filtering and dynamic thresholding for effective command recognition.

- **Customizable Wake Words (Singh et al., 2022):**

  Investigated user-configurable wake word systems to personalize voice recognition modules for specific applications. The study provided insights into balancing user customization and system performance.

3. Integration of Voice Control in Home Automation

The literature also examines various methods of integrating voice recognition modules into home automation systems:

- **Scenario-Based Automation (Kim et al., 2018):**

  Proposed linking voice commands to multiple device actions, such as activating "movie mode" to simultaneously dim lights, lower blinds, and start media playback. Their study demonstrated how voice recognition modules can create cohesive smart home environments.

- **Interoperability Challenges (Gupta & Li, 2021):**

  Discussed the importance of ensuring compatibility between voice control modules and existing communication protocols like Wi-Fi, Zigbee, and Bluetooth.

- **Energy Efficiency (Rana et al., 2020):**

  Explored strategies for minimizing power consumption in embedded voice modules for battery-operated smart home devices.

4. Privacy and Security Concerns in Voice Control Systems

Privacy has been a critical area of focus in literature, particularly for systems like the AI Thinker VC-02, which operate offline to enhance data security:

- **Local Voice Processing (Patel et al., 2021):**

  Examined the effectiveness of on-device voice processing in protecting sensitive user data. Their findings emphasized the benefits of edge AI in mitigating privacy risks associated with cloud-based systems.

- **Security Challenges (Ahmed et al., 2019):**

  Addressed the risks of unauthorized command injection in voice control systems, proposing encryption methods and voice authentication to enhance security.

5. Applications in Real-World Scenarios

- **Smart Home Devices (Lee et al., 2020):**

  Reviewed the integration of voice modules into appliances such as smart thermostats, motorized curtains, and lighting systems. Their study showcased the versatility of voice recognition in enhancing everyday convenience.

- **Accessibility Solutions (Kumar & Zhao, 2022):**

  Investigated the role of voice control systems in enabling independent living for individuals with mobility challenges or disabilities, emphasizing its potential for social impact.

Sensor-Based Automation

The role of sensors in automation is critical for achieving intelligent, adaptive systems. Key sensors such as motion detectors, temperature sensors, and gas sensors have been widely studied:

- Motion Sensors: Studies by Liu et al. (2017) show that motion detection is effective in enhancing home security and optimizing energy use by activating lights or appliances only when motion is detected.
- Temperature Sensors: Research by Sharma et al. (2019) highlights the use of temperature sensors for controlling HVAC systems, ensuring energy efficiency and user comfort.
- Gas Sensors: Gupta et al. (2018) explored the use of gas sensors for detecting hazardous gases like LPG or carbon monoxide, significantly improving home safety by providing real-time alerts.

Smart Appliance Control

Smart appliances form the backbone of home automation systems. Researchers have explored different methods for controlling appliances:

- Wireless Control: Ahmad et al. (2015) demonstrated the use of Wi-Fi and Zigbee for controlling appliances remotely. These technologies are cost-effective and easy to implement but face limitations in network coverage.
- Voice-Activated Systems: Ramesh and Kumar (2020) studied the integration of voice assistants like Amazon Alexa and Google Home, which allow hands-free control of appliances but depend heavily on internet connectivity.

Energy Efficiency in Home Automation

Energy conservation is one of the primary motivations for adopting smart home systems.

- A study by Rahman et al. (2016) proposed an IoT-based energy monitoring system that reduces energy wastage by analyzing usage patterns and automating appliances.
- Han et al. (2018) demonstrated that using real-time data from sensors, such as temperature and occupancy data, can lead to energy savings of up to 30% in smart homes.

Conclusion

The literature survey reveals that while significant progress has been made in home automation, there are still challenges that need to be addressed, including system scalability, energy efficiency, and security. By leveraging advancements in IoT, AI, and edge computing, future systems can overcome these limitations and provide a more intelligent, efficient, and secure solution.

EXISTING SYSTEM

Human beings are very lazy creatures, we want to do all of our work by just sitting at a one place and not bothering to go out and socialize. Emerging technology has on the other hand is trying its best to fulfil our demand by everyday introducing something new. One of the emerging and interesting field is Internet of Things.

Under Internet of Things we have a small domain called Home automation in which focus is on automating almost every device of our homes. We can make our own security system just by using sensors and doing some code. We can control the devices wirelessly by using Bluetooth or Wi-Fi module. It is all about connecting out devices with the network so that we can control almost anything from anywhere. Unlimited amount of sensors are available in the market and our task is only to deeply understand their functioning and implement them in various areas of concern. One of the examples is of water level sensor, we can use it in our water tanks to measure the level of water and automatically the motor will be turned off when the level exceeds some threshold value.

Need to automate the device can come to any of us but we do not have enough of the knowledge of how to do the things. So the need to study the basic concepts of Internet of Things are important. We can get various ideas related to the innovation by constantly surveying the people by emails or sms etc.

Today Internet of Things is an important part of innovation where constant researches are going on and in Home automation everyday something new is being generated. In our project we will be testing some of the basic sensors and then try to control some of the devices using Bluetooth and Wi-Fi module.

Major issue now a days is the issue of privacy. With recent events like Facebook Cambridge analytica case, privacy of users is becoming of great concern to the state authorities in position. Data from several websites and the data stored on various clouds also has an issue of privacy due to increasing number of cyber-attacks and increasing cases of data being stolen without the consent of the user. Data from several servers is being stolen and sold to big corporate so that they could easily target their audience and make money out of it.

Figure 3.1: Smart home

IOT FOR SMART TECHNIQUE

Control: We can easily control all the network connected devices almost everywhere around the world, for example we did not turn off the lights but our light system is connected to the network via some sensors and constantly monitoring the status of the lights and giving us the report via sms or any other means. By seeing the status of the lights we can turn it off by sending some command.

Security: This can be achieved by using Wi-Fi enabled cameras, motion sensor which will sense the motion or any activity of any object and by using this in collaboration of alarm we can easily monitor what is going on in the house.

Safety: It can be achieved by using sensors like water sensor which can detect any leakage in the water tank, smoke sensors which can detect fire etc. Using these sensors and connecting them to the network can prevent many tragedies from encountering.

Convenience: This is the most desirable feature or advantage of smart homes. We do not have to do any kind of hard work rather everything happens automatically just by sitting at a corner and giving commands to the devices. Example in voice controlled homes we just need to say that turn off lights and the lights will switch off, or led display on fridge which will display that milk is in the fridge or we need to buy it or if we are out of it can order itself online.

Figure 3.2: Uses of Internet of Things

DISCUSSION OF LITERATURE SURVEY

Objective

To develop an automated home controlled using Arduino.

- The objective is to conserve energy resources.
- To implement services such as switching on/off lights via mobile, smart car parking etc
- To use Bluetooth to make it wireless.
- To use Wi-Fi module esp8266 to wirelessly control the devices.

Methodology

In this smart home automation we would have a system which can identify the owner of the house. When someone is at the door it sends the picture of the person to the owner and door automatically opens or closes on the command of the owner. When you enter the room the lightning of the room goes as specified by the user according to his mood.

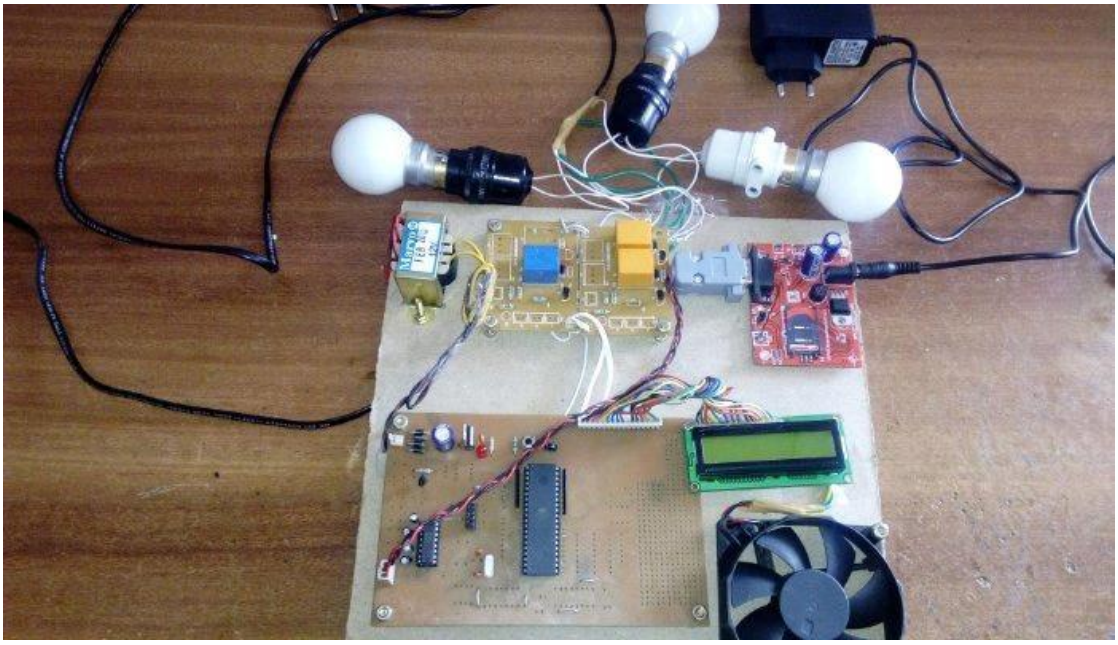


Figure 3.3: Methodology

To achieve all of the above we will build an html page accessible from every device which will all the functionalities needed to automate the home.

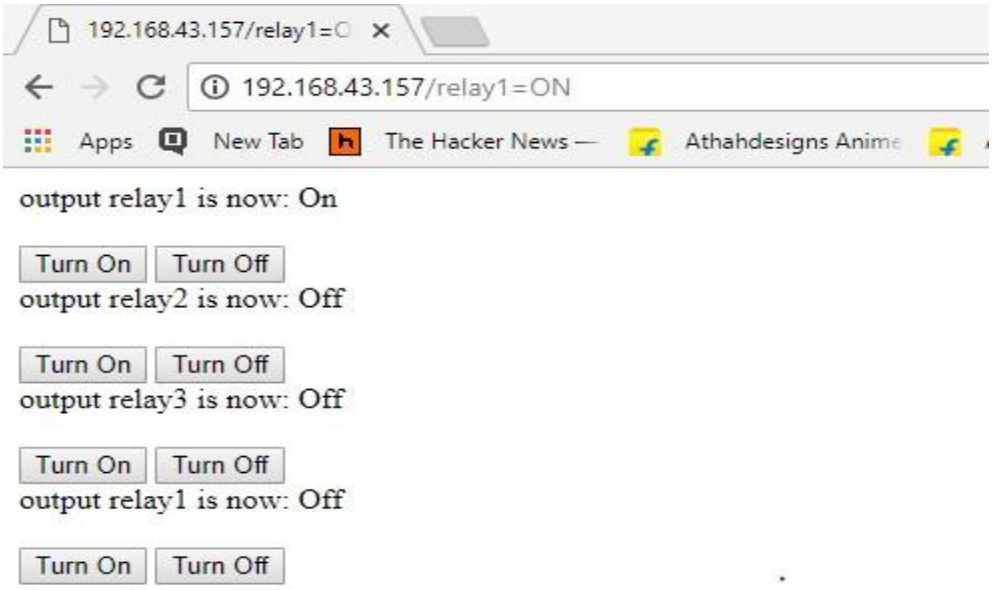


Table 1: Technical Specifications

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage	7-12V
Digital pins	14
PWM Digital Pins	6
Analog I/P Pins	6
DC current per pin	20mA
Flash Memory	32KB
SRAM	2KB
EEPROM	2KB
Clock speed	16MHz
Length	68.6mm
Width	53.4mm
Weight	25g

Figure3.4: Arduino IDE

2. Proteus8 Professional

Proteus is a software for doing the simulation of the project. The software comes with some of the built in sensors and other electronic components and other non-inclusive components are too included explicitly by downloading certain libraries and including in proteus's library section. It is a great tool to simulate various projects before implementing in hardware.

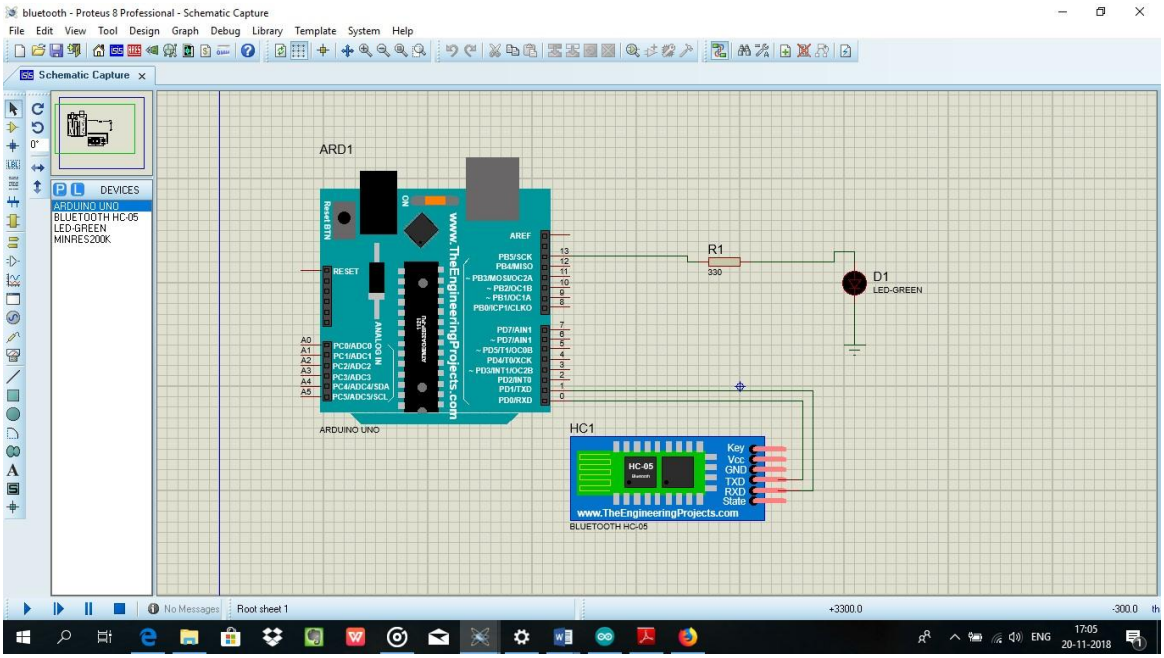


Figure 3.5: Proteus8 Professional

PROPOSED SYSTEM

The AI Thinker VC-02 is a cutting-edge voice control module that plays a transformative role in the field of home automation. As the demand for smart homes continues to grow, voice control has emerged as a key interface for creating seamless and intuitive user experiences. The VC-02 module offers a powerful offline voice recognition system, enabling users to control appliances, lighting, security systems, and more through simple voice commands—without relying on internet connectivity.

Designed with a high-performance Digital Signal Processor (DSP), the VC-02 delivers fast and accurate recognition of predefined commands, ensuring reliability and privacy in voice-controlled systems. Its offline functionality not only eliminates latency issues common with cloud-based solutions but also enhances data security by processing commands locally. These features make it particularly well-suited for smart homes where consistent and secure operation is essential.

The VC-02 integrates easily into home automation platforms, enabling devices to respond to commands like turning on lights, adjusting the thermostat, opening curtains, or activating security systems. It supports customizable wake words and commands, allowing users to tailor their smart home experiences according to personal preferences. With its compact size, low power consumption, and developer-friendly environment, the VC-02 is a versatile solution for manufacturers and developers aiming to create innovative and user-centric home automation products.

This document delves into the features and benefits of the AI Thinker VC-02, focusing on its role in enhancing home automation. It highlights technical specifications, such as its low power consumption, voice recognition accuracy, and compatibility with popular microcontrollers like ESP32 and STM32. The VC-02 serves as an essential building block for developers and manufacturers aiming to create innovative, voice-controlled smart home devices that improve convenience, security, and personalization for end users.

METHODOLOGY:

The methodology for implementing the **AI Thinker VC-02** module in a home automation system involves a systematic approach to hardware integration, voice command customization, and functional deployment. Below is an outline of the key steps and components in the methodology:

1. System Architecture Design

- **Objective:** Define the architecture of the home automation system integrating the VC-02 module.
- **Steps:**
    1. Identify the devices to be controlled, such as smart lights, fans, air conditioners, and security systems.
    2. Select a suitable microcontroller, such as **ESP32**, for interfacing with the VC-02 module.
    3. Define communication protocols (e.g., Wi-Fi, Bluetooth, Zigbee) for interaction between devices.
    4. Design a power management system to ensure energy-efficient operation of the VC-02 module.

2. Hardware Setup

- **Objective:** Physically integrate the VC-02 module with the home automation devices.
- **Components:**
    - **AI Thinker VC-02 module**: Core voice recognition hardware.
    - **Microcontroller unit (MCU)**: Acts as the central controller for processing commands and controlling devices.
    - **Relay modules**: For switching on/off electrical appliances.
    - **Power supply unit**: Provides appropriate voltage to the VC-02 and other components.
- **Steps:**

Connect the VC-02 module to the selected microcontroller using UART communication.

1. Link the microcontroller to the respective home automation devices via GPIO pins or relay modules.
2. Ensure the hardware setup is compact, efficient, and safe for household use.

3. Firmware Development

- **Objective:** Develop the firmware to enable communication and functionality.
- **Steps:**
    1. Configure the VC-02 module for offline voice recognition using the manufacturer's SDK.
    2. Program the microcontroller to process recognized commands and trigger appropriate device actions.
    3. Customize voice commands and wake words to suit specific user requirements (e.g., "Turn on the lights" or "Set temperature to 22°C").
    4. Implement error handling and fallback mechanisms for unrecognized commands.

4. Voice Command Customization

- **Objective:** Adapt the VC-02 module to the specific voice control needs of the system.
- **Steps:**
    1. Define a list of predefined commands and their corresponding actions (e.g., "Lights on," "Fan off").
    2. Use the provided VC-02 configuration tools to map commands to system actions.
    3. Conduct testing to ensure high recognition accuracy for each command.

4. Optimize recognition settings for noisy or distant environments.

5. System Integration

- **Objective:** Integrate the voice-controlled module into the home automation system.
- **Steps:**
    1. Connect the microcontroller to a central hub (if applicable) for controlling multiple devices.
    2. Link the VC-02 module to other smart devices in the system using communication protocols (e.g., Wi-Fi, Zigbee).
    3. Test interactions between devices, ensuring seamless execution of multi-device commands (e.g., "Goodnight" scenario).

TESTING AND OPTIMIZATION:

Testing and optimizing the AI Thinker VC-02 module for home automation ensures that the system functions reliably and efficiently under various conditions. This process focuses on verifying system performance, identifying potential issues, and implementing improvements to enhance accuracy, responsiveness, and user experience.

1. Testing Phase

1.1 Functional Testing

- **Objective:** Ensure that each component of the system works as expected.
- **Steps:**
    1. Test individual devices (e.g., lights, fans) to confirm they respond to voice commands.
    2. Validate the integration between the VC-02 module, microcontroller, and connected devices.
    3. Test predefined voice commands to ensure they trigger the appropriate device actions.
    4. Verify that multi-device scenarios (e.g., "Movie Mode") execute correctly.

1.2 Environmental Testing

- **Objective:** Evaluate system performance under different environmental conditions.
- **Scenarios to Test:**
    1. **Noise levels**: Test command recognition in quiet and noisy environments.
    2. **Distance**: Verify performance when the user is close to or far from the module.
    3. **Room acoustics**: Test in rooms with varying echo and sound absorption levels.
    4. **Light conditions**: Evaluate performance under bright and dim lighting, though primarily applicable if visual feedback is involved.

1.3 Latency Testing

- **Objective:** Measure the response time between issuing a voice command and the corresponding device action.
- **Steps:**
    1. Record the time taken for the VC-02 module to recognize a command.
    2. Measure the delay in device activation or deactivation.
    3. Compare latency under different conditions (e.g., with noise, at a distance).

1.4 Stress Testing

- **Objective:** Test the system's reliability under heavy usage or simultaneous commands.

- **Steps:**
    1. Issue multiple commands in quick succession to evaluate the module's processing capability.
    2. Simulate simultaneous commands from different users to check system behavior.

1.5 Security Testing

- **Objective:** Ensure the system is protected against unauthorized access or command injection.
- **Steps:**
    1. Test for vulnerabilities to spoofed commands or recordings.
    2. Validate the security of local voice data processing to prevent data leakage.

2. Optimization Phase

2.1 Command Accuracy Optimization

- **Objective:** Improve the recognition rate of voice commands.
- **Strategies:**
    1. Refine the voice recognition algorithm to reduce false positives and negatives.
    2. Adjust sensitivity settings for environments with high background noise.
    3. Train the module with varied accents, dialects, and speech patterns.

2.2 Latency Reduction

- **Objective:** Minimize the time between command recognition and device response.
- **Strategies:**
    1. Optimize firmware for faster processing on the microcontroller.
    2. Reduce unnecessary processing steps in the communication pipeline.
    3. Ensure efficient use of resources, such as RAM and CPU.

2.3 Power Consumption Optimization

- **Objective:** Maximize energy efficiency for long-term use, especially in battery-powered systems.
- **Strategies:**
    1. Enable low-power modes when the VC-02 module is idle.
    2. Optimize code to reduce unnecessary processor activity.
    3. Use energy-efficient components for peripheral devices.

2.4 Customization and Scalability

- **Objective:** Tailor the system to user needs and prepare it for future expansions.
- **Strategies:**
    1. Add user-configurable wake words and commands.
    2. Ensure compatibility with additional devices and communication protocols (e.g., Zigbee, Bluetooth).
    3. Implement modular updates, allowing firmware upgrades without hardware changes.

2.5 Usability Optimization

- **Objective:** Enhance user experience and accessibility.

- **Strategies:**
    1. Simplify setup procedures with user-friendly interfaces.
    2. Provide voice feedback to confirm command execution.
    3. Incorporate error messages or fallback actions when commands are not recognized.

RELAY CONFIGURATIONS

Relay Configurations for Home Automation with AI Thinker VC-02

Relay modules are essential components in home automation systems for switching electrical devices on and off using low-voltage signals from microcontrollers or voice recognition modules like the AI Thinker VC-02. Below is a comprehensive guide on configuring relay modules with the VC-02 for controlling home appliances.

1. Relay Module Overview

A relay module allows the switching of higher-voltage appliances using low-voltage control signals. It works by energizing an electromagnet that opens or closes the circuit to control devices like lights, fans, air conditioners, and more.

2. Types of Relay Modules

Common relay modules include:

- **Single-channel relay**: Controls one device at a time.
- **Multi-channel relay**: Controls multiple devices simultaneously (e.g., 2-channel, 4-channel, 8-channel).
- **Solid-state relay (SSR)**: Offers silent operation and better longevity but typically higher cost.
- **Mechanical relay**: Standard relay used in most systems; less expensive but can have wear issues over time.

For the AI Thinker VC-02 integration, you can use a **5V relay module**, which is commonly compatible with microcontrollers like ESP32.

3. Relay Configuration Setup

3.1 Wiring the Relay to Microcontroller

The relay module will act as an intermediary between the AI Thinker VC-02 module and the devices you want to control. Here's how to wire it:

Components Required:

- AI Thinker VC-02 module
- Microcontroller (e.g., ESP32 or Arduino)
- 5V relay module (single or multi-channel)
- Home appliances (e.g., lights, fans)
- Jumper wires
- External power supply (if needed)

Steps:

1. **Power Supply**: Connect the 5V and GND pins of the relay module to the corresponding power source pins (5V and GND) on the microcontroller.
2. **Control Signal Pin**: Connect the relay module's IN pin to a GPIO pin on the microcontroller (e.g., GPIO12 for Relay 1).
3. **Device Control**: Connect the Normally Open (NO) terminal of the relay to the live wire of the appliance. Connect the Common (COM) terminal to the power source (AC live wire). The Neutral wire of the appliance should be connected directly to the AC neutral wire.

Example Wiring for Single-Channel Relay:

- **Relay IN Pin → GPIO Pin on Microcontroller (e.g., GPIO12)**
- **Relay 5V Pin → 5V Power Pin on Microcontroller**
- **Relay GND Pin → GND Pin on Microcontroller**
- **Relay NO Pin → Live Wire of Appliance**
- **Relay COM Pin → AC Power Source (Live)**

3.2 Relay Control Logic in Firmware

To control the relay using voice commands from the AI Thinker VC-02, you need to integrate it with your microcontroller's firmware. Below is a basic outline of the logic for controlling a relay.

Basic Logic:

- **Relay ON**: When the VC-02 recognizes a voice command like "Turn on the lights," the microcontroller will activate the GPIO pin connected to the relay, switching the appliance on.
- **Relay OFF**: When the VC-02 recognizes a voice command like "Turn off the lights," the GPIO pin will be set to LOW, deactivating the relay and turning off the appliance.

Example Code (ESP32 with Arduino IDE):

```
#define RELAY_PIN 12  // GPIO12 connected to relay IN pin


void setup() {
  pinMode(RELAY_PIN, OUTPUT);  // Set relay pin as output
  digitalWrite(RELAY_PIN, LOW);  // Initialize relay to OFF
}


void loop() {
  // Example: Turn ON relay (e.g., if voice command "Turn on the lights" is detected)
  digitalWrite(RELAY_PIN, HIGH);  // Relay ON


  // Example: Turn OFF relay (e.g., if voice command "Turn off the lights" is detected)
  digitalWrite(RELAY_PIN, LOW);   // Relay OFF


  delay(1000);  // Delay to simulate waiting for the next command
}
```

3.3 Multi-Relay Configuration

For systems requiring control over multiple devices, you can configure multiple relays. If you're using a multi-channel relay, each channel will have its control signal pin connected to different GPIO pins on the microcontroller.

Example of 4-Channel Relay Configuration:

1. **Relay IN1 Pin → GPIO Pin 12** (for controlling device 1)
2. **Relay IN2 Pin → GPIO Pin 13** (for controlling device 2)
3. **Relay IN3 Pin → GPIO Pin 14** (for controlling device 3)

4. **Relay IN4 Pin → GPIO Pin 15** (for controlling device 4)

In the firmware, you can then control each relay individually based on the voice commands received.

Example Code for Multi-Relay (4-Channel):

```
#define RELAY1_PIN 12  // Relay 1 control pin
#define RELAY2_PIN 13  // Relay 2 control pin
#define RELAY3_PIN 14  // Relay 3 control pin
#define RELAY4_PIN 15  // Relay 4 control pin

void setup() {
  pinMode(RELAY1_PIN, OUTPUT);
  pinMode(RELAY2_PIN, OUTPUT);
  pinMode(RELAY3_PIN, OUTPUT);
  pinMode(RELAY4_PIN, OUTPUT);

  digitalWrite(RELAY1_PIN, LOW);
  digitalWrite(RELAY2_PIN, LOW);
  digitalWrite(RELAY3_PIN, LOW);
  digitalWrite(RELAY4_PIN, LOW);
}

void loop() {
  // Logic to turn on/off different relays based on voice commands

  digitalWrite(RELAY1_PIN, HIGH);  // Turn on device 1
  digitalWrite(RELAY2_PIN, LOW);   // Turn off device 2

  delay(1000);
}
```

4. Safety Considerations

- **Relay Ratings**: Ensure that the relay is rated for the voltage and current of the appliances being controlled. For example, use a 10A relay for high-power appliances (e.g., air conditioners, heaters) and a 5A relay for low-power devices (e.g., lights, fans).
- **Isolation**: For safety, ensure that the control side (low-voltage circuit) is properly isolated from the high-voltage side. Solid-state relays (SSR) offer better isolation and reliability for sensitive applications.
- **Protective Components**: Consider adding flyback diodes or snubber circuits across the relay's contacts to protect against voltage spikes when switching inductive loads like motors or compressors.

5. Optimization and Debugging

- **Debouncing**: Relays may experience contact bouncing, especially in mechanical relays. Software debouncing techniques or hardware solutions (e.g., capacitors) can help reduce the chances of misfiring.
- **GPIO Pin Management**: Ensure you manage GPIO pins carefully, especially if the microcontroller has limited GPIO availability.

HARDWARE DESCRIPTION

CHARACTERISTICS

1. Switching Capability

- **Voltage Rating:** Relay modules have specific voltage ratings, typically either for **AC** or **DC** circuits. For example:
  - **AC relays** are often rated for **110V to 220V**.
  - **DC relays** are typically rated for **5V or 12V**.
- **Current Rating:** Relay modules are rated for the amount of current they can switch. Common ratings include **5A** and **10A**, which determine the size and power of appliances that can be controlled.
- **Type of Load:** Relays can be used for various types of loads such as:
  - **Resistive loads** (e.g., lamps, heaters),
  - **Inductive loads** (e.g., motors, fans, compressors).

2. Control Voltage

- The **control voltage** is the low-voltage signal used to activate the relay. In most cases, **5V relays** are used in home automation projects, and the **AI Thinker VC-02** can be interfaced with microcontrollers like **ESP32** or **Arduino**, which provide this 5V signal.
- **Low-power operation**: The relay operates with a low voltage from the microcontroller or VC-02 module, enabling safe and easy control of high-power appliances.

3. Form Factor and Number of Channels

- **Single-Channel Relay:** Controls a single device. Used when only one appliance needs to be controlled.
- **Multi-Channel Relay:** Controls multiple devices simultaneously (e.g., **2-channel**, **4-channel**, **8-channel**). It allows for controlling multiple appliances in one configuration, suitable for complex home automation systems.
- The choice of form factor depends on the complexity and requirements of the home automation setup.

4. Isolation

- **Electrical Isolation:** Relays provide electrical isolation between the low-voltage control circuit and the high-voltage appliance. This ensures safety and protects sensitive microcontrollers or modules from high-voltage spikes.
- **Mechanical Relays** typically offer galvanic isolation by using physical contacts that open and close. **Solid-state relays** (SSRs) offer the same isolation but do so electronically, without moving parts, which can lead to greater reliability and longer lifespan.

5. Response Time

- **Activation Time:** The time it takes for a relay to switch on (activate) when the control voltage is applied. This is usually very fast (in milliseconds).
- **Deactivation Time:** The time it takes for a relay to switch off when the control signal is removed. This is also typically quick but may vary slightly depending on the type of relay.
- In home automation, a **short response time** ensures that appliances react promptly to voice commands.

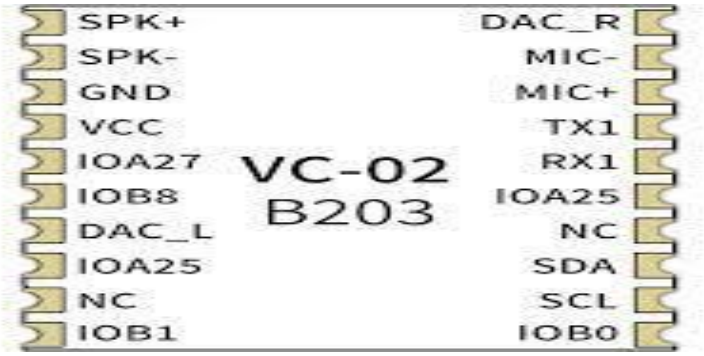


Fig 5.1 – Circuit Diagram

Fig 5.2 Development board

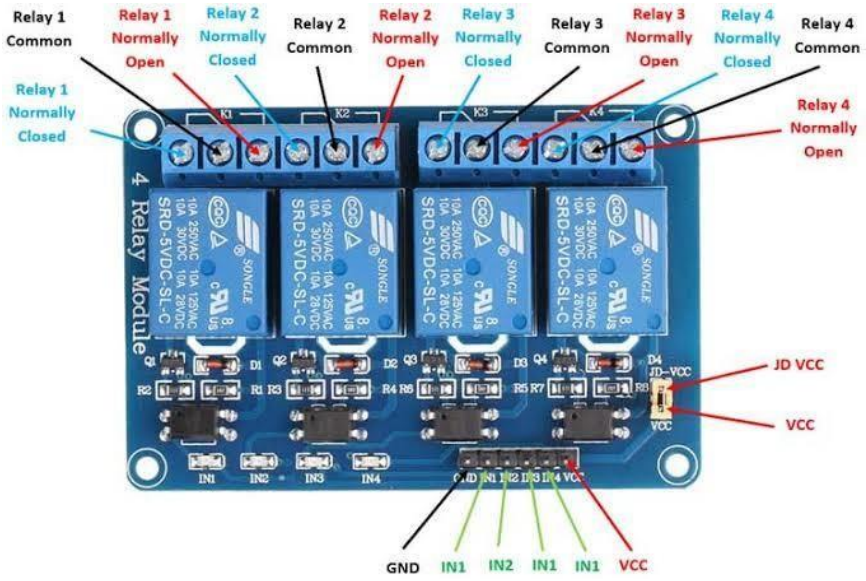


Fig 5.3 Relay

Relays control one electrical circuit by opening and closing contacts in another circuit. When a relay contact is normally open (NO), there is an open contact when the relay is not energized. In electromechanical relays (EMR), contacts are opened or closed by a magnetic force.

FUNCTIONS

1. Device Control

- **On/Off Control**: The primary function of a relay in a home automation system is to switch electrical devices on or off. The relay receives a control signal (e.g., from the AI Thinker VC-02 module via a microcontroller) to activate or deactivate connected appliances.
    - **Example**: When the user says, "Turn on the lights," the voice command triggers the microcontroller to send a signal to the relay, which closes the circuit and powers on the light.

2. Isolation of Low and High Voltage Circuits

- **Electrical Isolation**: One of the key functions of a relay is to provide **galvanic isolation** between the low-voltage control circuit (e.g., the microcontroller or voice recognition module) and the high-voltage appliances being controlled. This ensures that the low-voltage system remains safe from electrical surges, spikes, or faults in the high-voltage circuit.
    - **Example**: The relay safely isolates the **AI Thinker VC-02** module from dangerous high-voltage electricity while still enabling it to control devices such as lights or appliances.

3. Automation of Household Tasks

- **Scheduling and Automation**: Relays enable the automation of everyday tasks by switching devices based on time schedules, sensor inputs, or voice commands. When combined with the AI Thinker VC-02, users can automate tasks like turning on lights when entering a room or turning off an appliance when no one is present.
    - **Example**: "Turn off the kitchen light after 10 minutes" or "Turn on the fan when the temperature exceeds 25°C."

4. Voice-Controlled Home Automation

- **Voice Command Execution**: The relay module interacts with voice recognition systems like the AI Thinker VC-02 to carry out tasks based on spoken commands. This enables hands-free control of various devices and systems within a smart home.
    - **Example**: The user can simply say, "Turn off the fan," and the relay will disconnect the power supply to the fan based on the voice command received from the VC-02 module.

5. Integration with Other Smart Devices

- **Smart Home Ecosystem Integration**: Relay modules allow the integration of traditional devices with modern smart home systems. For example, users can integrate regular household appliances (e.g., lamps, coffee makers, fans) into their home automation system, allowing them to control these devices remotely or through voice commands.
    - **Example**: "Turn on the coffee machine," or "Activate the heater in the living room" can be easily achieved through relays linked to a home automation system.

WORKING

The relay module in a home automation system is responsible for controlling high-voltage devices using low-voltage signals, such as those from a microcontroller or a voice recognition module like the **AI Thinker VC-02**. The working of a relay module in this setup can be broken down into several stages, from receiving the control signal to switching the appliance on or off.

1. Relay Module Basics

A **relay** is an electromagnetic switch that allows you to control the flow of current in a circuit without directly handling the high-voltage components. The relay consists of the following key parts:

- **Electromagnet**: When energized by a low-voltage signal, it creates a magnetic field that moves the switch mechanism.
- **Contacts**: The relay has two primary contacts: the **Common (COM)**, **Normally Open (NO)**, and **Normally Closed (NC)** contacts.
- **Coil**: When the coil is energized, it closes the circuit between the COM and NO contacts. If the coil is de-energized, the NO contact remains open, and the NC contact may close.

2. Control Signal from AI Thinker VC-02

The **AI Thinker VC-02** module is a voice recognition unit that can detect specific commands from the user, such as "Turn on the light" or "Switch off the fan." When the **VC-02** module receives a valid voice command, it sends a control signal to the microcontroller (e.g., **ESP32** or **Arduino**), which processes the command and sends the appropriate signal to the relay.

- **Voice Command Recognition**: The **AI Thinker VC-02** processes the spoken command and translates it into an actionable output, such as turning on a light, activating a fan, or switching off an appliance.

3. Relay Activation

The microcontroller (like an **ESP32**) receives the signal from the VC-02 and uses one of its General-Purpose Input/Output (GPIO) pins to send a low-voltage signal to the relay module. This is the activation signal for the relay.

- **Low-Voltage Control**: The relay module is controlled by a low-voltage signal, typically **5V** or **3.3V**, depending on the system design. This control signal is sent from the microcontroller to the relay's **IN pin**.
- **Relay Coil Energization**: When the microcontroller sends a HIGH signal to the relay's control pin, the relay's electromagnet coil is energized. This causes the relay to switch the state of its contacts, either **closing** the circuit (for NO) or **opening** the circuit (for NC).

4. Switching the Appliance

When the relay coil is energized:

- **Normally Open (NO)**: The **NO** contact, which is initially open (disconnected), closes, allowing current to flow through the connected appliance.
- **Normally Closed (NC)**: If the relay is designed to use the NC configuration, the NC contacts are initially closed, and the relay opens the circuit when triggered.

In either case, the high-voltage current is now flowing through the appliance (e.g., light, fan, heater), turning it on.

- **Relay-Actuated Switch**: This process works as a switch that activates or deactivates the high-power devices based on the low-voltage signal received from the AI Thinker VC-02.

5. Deactivation of Relay

When the command to turn off the device is given:

- The **AI Thinker VC-02** sends a signal to the microcontroller to **deactivate** the relay.
- The **GPIO pin** on the microcontroller goes LOW, de-energizing the relay coil.
- This causes the **NO contacts** to open again, cutting off the flow of electricity to the appliance, which turns off the device.
- **Safe Power-Off**: When the control signal goes LOW, the relay coil is de-energized, and the contacts revert to their default state (NO opened or NC closed)

SOFTWARE DESCRIPTION

FUNCTIONS

In a home automation system utilizing **AI Thinker VC-02** for voice recognition and relay modules for controlling high-voltage appliances, software plays a crucial role in managing communication, controlling devices, and ensuring seamless operation. The software functions can be divided into several categories, each of which ensures the smooth operation of the system.

1. Voice Command Recognition and Parsing

- **Function**: Recognizes and processes voice commands received by the **AI Thinker VC-02** module.
- **Details**:
  - The software interprets user commands like “Turn on the lights,” “Activate the fan,” or “Switch off the television.”
  - Voice recognition is handled by the AI Thinker VC-02, which processes the audio input and converts it into text or actions.
  - The system uses a **predefined set of voice commands** to identify what the user wants to control. These commands are mapped to specific functions in the software.

Example Function**:**

c

Copy code

```
// Pseudocode for voice command recognition
if (VC_02.command == "Turn on the light") {
   controlDevice(LIGHT, ON);
}
```

2. Device Control Logic

- **Function**: Sends control signals to relay modules based on voice commands.
- **Details**:
  - Once the voice command is parsed and recognized, the software triggers specific functions to control the relay module connected to the device.
  - The control is done via GPIO pins from a **microcontroller** (like **ESP32** or **Arduino**) to activate or deactivate the relay, which in turn controls the connected appliance.
  - **Relay control functions** are often simple GPIO toggling based on the state (ON/OFF) that needs to be sent to the relay.

Example Function:

c

Copy code

```
void controlDevice(int device, bool state) {
   if (state == ON) {
      digitalWrite(devicePin[device], HIGH);  // Activates the relay
   } else {
      digitalWrite(devicePin[device], LOW);   // Deactivates the relay
```

```
  }
}
```

3. System Initialization

- **Function**: Initializes hardware components (e.g., AI Thinker VC-02, relay modules, microcontroller pins) at the startup of the system.
- **Details**:
  - At startup, the microcontroller initializes the voice recognition system, setting up the **AI Thinker VC-02** module for communication.
  - The relay pins are also configured as **OUTPUT** pins for controlling connected appliances.

Example Function:

```
c
Copy code
void setup() {
  // Initialize the relay control pins
  pinMode(DEVICE_PIN_1, OUTPUT);
  pinMode(DEVICE_PIN_2, OUTPUT);

  // Initialize the voice recognition module
  VC02.begin();
}
```

4. Voice Command Feedback

- **Function**: Provides feedback to the user about the status of the command (success/failure) through audio or visual signals.
- **Details**:
  - After processing a voice command, the system might offer audio feedback or even send status updates (e.g., "The light is now on").
  - This can be achieved by sending a voice response from the VC-02 or through the microcontroller triggering a speaker or LED to indicate the device status.

Example Function:

```
c
Copy code
void provideFeedback(String message) {
  // Use voice output to notify the user
  VC02.speak(message);
}
```

5. Timing and Scheduling (Optional)

- **Function**: Allows for scheduling specific actions based on time or conditions (e.g., turning on a device at a particular time).

- **Details**:
  - The system can include functions for scheduling appliances to turn on or off at a specified time or when specific conditions are met (e.g., temperature thresholds or occupancy).
  - These functions could rely on **real-time clock (RTC)** modules or **time-based logic** within the microcontroller.

Example Function:

c

Copy code

```
void scheduleTask(int device, int hour, int minute, bool state) {
  // Example: Turn on light at 7:00 AM
  if (hour == 7 && minute == 0) {
    controlDevice(device, state);  // Turn on/off device
  }
}
```

REAL TIME USE

In a home automation system using the **AI Thinker VC-02** for voice control and relay modules to control high-voltage appliances, real-time software plays a crucial role in ensuring seamless and responsive operation. The software must manage multiple tasks simultaneously, including voice recognition, device control, feedback, and handling real-time changes in the system's environment. Below are examples of real-time software applications in this type of system:

1. Voice-Controlled Appliance Management in Real-Time

- **Use**: Voice commands are processed and executed in real-time, providing users with the ability to control appliances on-demand.
- **How it Works**:
  - The **AI Thinker VC-02** listens to user input in real-time, processing commands such as "Turn on the living room light" or "Turn off the air conditioner."
  - As soon as a command is recognized, the software sends a signal to the microcontroller (e.g., **ESP32** or **Arduino**) to activate or deactivate the corresponding relay.
  - The relay immediately switches the appliance, and feedback is provided to the user (either through voice feedback or an indicator).

Example Real-Time Function:

c

Copy code

```
void processVoiceCommand(String command) {
  if (command == "Turn on the light") {
    controlDevice(LIGHT, ON);  // Activates the relay and turns on the light
    provideFeedback("The light is now on.");
  } else if (command == "Turn off the fan") {
    controlDevice(FAN, OFF);   // Deactivates the relay and turns off the fan
    provideFeedback("The fan is now off.");
  }
```

```
}
```

2. Real-Time Scheduling and Timed Actions

- **Use**: Real-time scheduling allows users to automate appliance actions based on time.
- **How it Works**:
  - The system can be configured to turn devices on or off at specific times or intervals without manual intervention.
  - This functionality is useful for activities like setting a coffee maker to start brewing at 7 AM or scheduling lights to turn on when the sun sets.

Example Real-Time Function:

c

Copy code

```
void checkSchedule() {
   int currentHour = getCurrentHour();
   int currentMinute = getCurrentMinute();

   // Turn on the living room light at 6:00 PM
   if (currentHour == 18 && currentMinute == 0) {
      controlDevice(LIGHT, ON);
      provideFeedback("The living room light is now on.");
   }

   // Turn off the heater at 10:00 PM
   if (currentHour == 22 && currentMinute == 0) {
      controlDevice(HEATER, OFF);
      provideFeedback("The heater is now off.");
   }
}
```

3. Real-Time Environment Monitoring and Response

- **Use**: In home automation, real-time responses are often required to environmental changes, such as temperature or motion detection.
- **How it Works**:
  - The system can use sensors (e.g., temperature sensors, motion sensors, humidity sensors) that continuously feed data to the microcontroller.
  - Based on sensor readings, the software can take immediate action, such as turning on the air conditioner when the temperature exceeds a threshold or activating lights when motion is detected.

Example Real-Time Function:

c

Copy code

```
void monitorTemperature() {
  int currentTemperature = readTemperature();  // Assuming function to read temperature

  // Turn on the fan if the temperature exceeds 30°C
  if (currentTemperature > 30) {
    controlDevice(FAN, ON);
    provideFeedback("The fan is now on, as the temperature is above 30°C.");
  }

  // Turn off the fan if the temperature goes below 25°C
  if (currentTemperature < 25) {
    controlDevice(FAN, OFF);
    provideFeedback("The fan is now off, as the temperature is below 25°C.");
  }
}
```

4. Real-Time Error Handling and Recovery

- **Use**: The system must be able to detect and recover from errors in real time, ensuring continuous operation.
- **How it Works**:
  - The software continuously monitors the status of devices (e.g., whether the relay has switched properly) and looks for potential errors such as communication failures or hardware malfunctions.
  - If an error is detected, the system either retries the operation or informs the user through feedback.

Example Real-Time Function:

c

Copy code

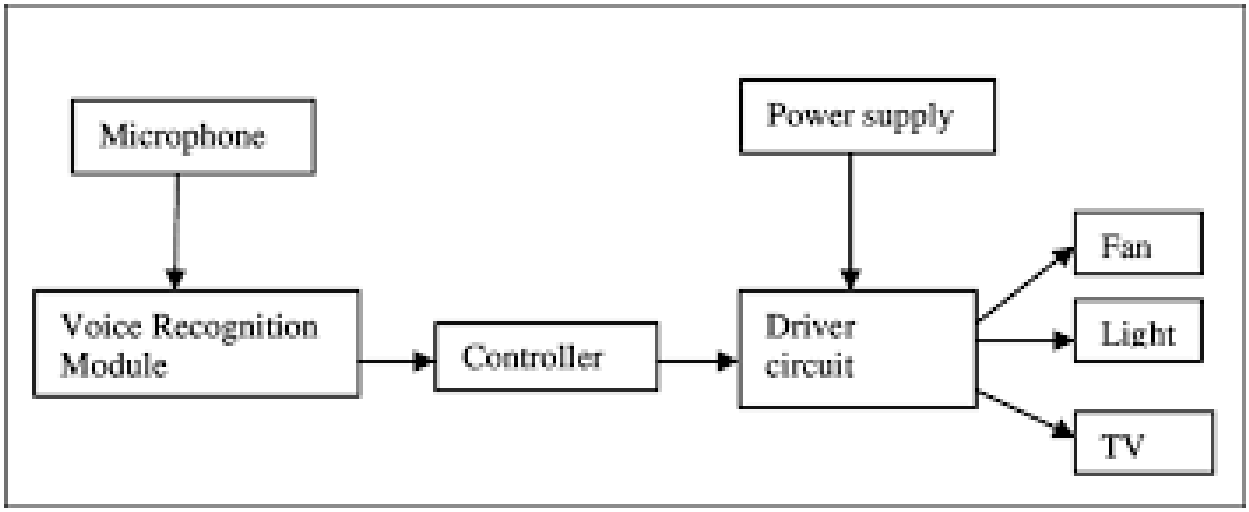
```
void checkDeviceStatus() {
  bool deviceStatus = digitalRead(RELAY_PIN);  // Read relay status

  // If the relay didn't activate as expected, try again
  if (deviceStatus == LOW) {
    provideFeedback("There was an issue turning on the light. Retrying...");
    digitalWrite(RELAY_PIN, HIGH);  // Attempt to activate the relay again
  }
}
```

5. Voice Feedback in Real-Time

- **Use**: Real-time voice feedback ensures users are aware of the system's actions and status without needing to visually monitor the devices.
- **How it Works**:

- After executing a voice command or performing an action, the system uses the **AI Thinker VC-02** to give immediate feedback about the result (e.g., "The light is on" or "The fan is off").
- This feature makes the system interactive and responsive, providing users with confirmation and updates about the current state of their appliances.

Example Real-Time Function:



c

Copy code

```
void provideVoiceFeedback(String message) {
   VC02.speak(message);  // Voice output through AI Thinker VC-02
```

BLOCK DIAGRAM

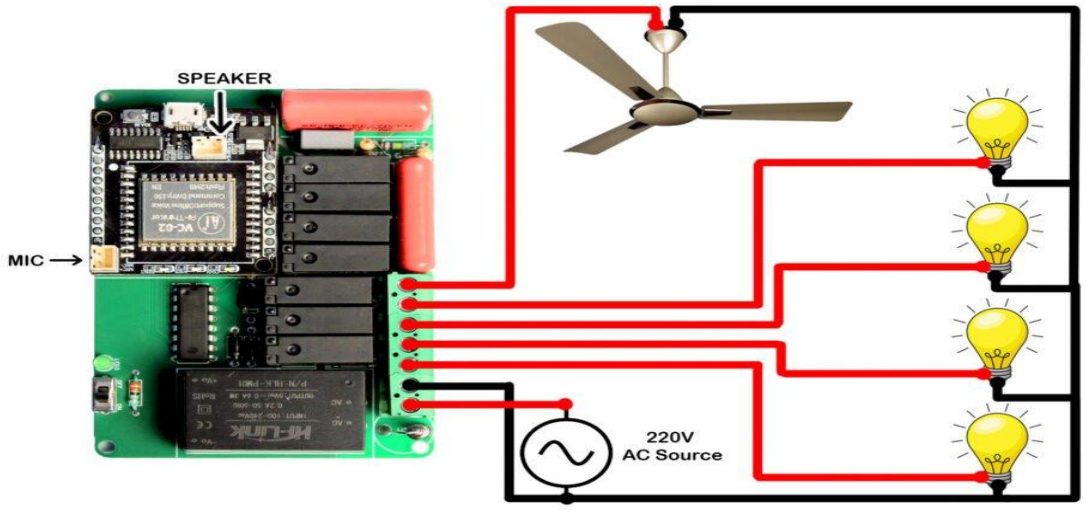


Fig 6.1 Circuit Diagram

OUTPUTS AND WAVEFORM

OBJECTIVES

1. Accurate Control of Appliances and Devices

- **Objective**: To reliably control home appliances (such as lights, fans, or air conditioners) through precise signal outputs.
- **Waveform Type**: Digital outputs in the form of **HIGH (ON)** and **LOW (OFF)** states.
- **Purpose**: Ensure that commands (like turning on/off a device) are executed correctly and consistently. The square waveform (representing digital signals) controls relays to activate or deactivate appliances.

2. Real-Time Feedback to Users

- **Objective**: To provide immediate feedback to users about the status of their voice commands or device states (e.g., whether a light is on, a fan is running, or a command has been understood).
- **Waveform Type**: **Analog audio signals** generated by the **AI Thinker VC-02** module or **LED visual outputs** using PWM or digital signals.
- **Purpose**: Ensure that users receive a clear and immediate response to their commands. This feedback enhances user experience and confirms actions, such as "The light is on" or "Turning off the fan."

3. Efficient Power Management and Device Control

- **Objective**: To manage the power usage of devices efficiently, especially when multiple devices are controlled simultaneously or based on scheduled timings.
- **Waveform Type**: **PWM signals** for controlling device brightness (LEDs) or fan speeds.
- **Purpose**: Provide smooth control over devices that require variable power, such as adjusting the intensity of light or controlling the speed of motors, based on real-time needs. The PWM waveform ensures fine-grained control of device behavior.

COMPONENTS OF AI SIMULATION

Digital Outputs for Relay Control

- **Output Type**: **Digital Output** (HIGH/LOW)
- **Function**: The microcontroller (e.g., **ESP32** or **Arduino**) sends digital HIGH or LOW signals to control the relay module. The relay acts as a switch to control the flow of electricity to an appliance.
    - **HIGH Signal (3.3V/5V)**: This sends power to the relay coil, causing it to activate (close the NO contacts), thereby turning on the connected device (e.g., light, fan, etc.).
    - **LOW Signal (0V)**: This de-energizes the relay coil, causing the relay to deactivate (open the NO contacts), turning off the appliance.

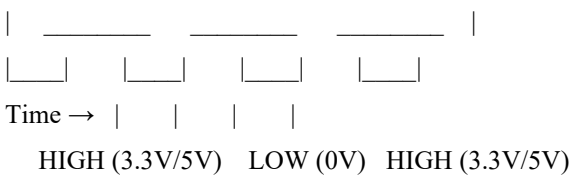
Waveform:

- The waveform of the digital output is a **square wave**, where the state alternates between HIGH (on) and LOW (off).
- A typical **HIGH signal** (e.g., 3.3V or 5V) is applied to the relay's control input, and the duration of the HIGH signal determines how long the device is activated.

Example of a square waveform:

markdown

Copy code



2. Output for Voice Feedback (Audio)

- **Output Type**: **Audio Output**
- **Function**: The **AI Thinker VC-02** provides **audio feedback** to the user based on the command processed. This feedback might be in the form of voice output ("The light is on," "Turning off the fan," etc.).
    - The VC-02 generates an **analog audio signal** that is sent to a connected speaker or audio system to provide real-time feedback to the user.

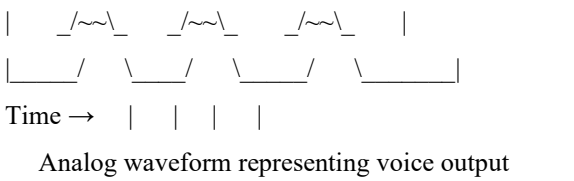
Waveform:

- The audio output is an **analog waveform** that corresponds to the sound signal being played by the VC-02 module.
- This waveform would typically represent the **frequency modulations** of the voice signal.

Example of an audio waveform (simplified):

markdown

Copy code



Analog waveform representing voice output

3. Relay Switching Waveform

- **Output Type**: **Relay Switching Action**
- **Function**: The relay switches the **high-voltage AC or DC** circuit connected to the appliance.
    - When the relay is **activated (HIGH signal)**, it closes the NO contact, allowing current to flow to the appliance.
    - When the relay is **deactivated (LOW signal)**, it opens the NO contact, interrupting the current flow and turning off the appliance.

Waveform:

- The relay switching waveform could be viewed as a **square wave** that represents the activation and deactivation of the relay.
    - **HIGH (Relay ON)**: Current flows through the appliance, and the device is powered on.
    - **LOW (Relay OFF)**: Current is cut off, and the device is powered off.

Example of a relay switching waveform:

markdown

Copy code

```
|  ________     ________     ________   |
|____|     |____|     |____|     |____|
Time →  |     |     |     |
   HIGH (Relay ON)   LOW (Relay OFF)
```

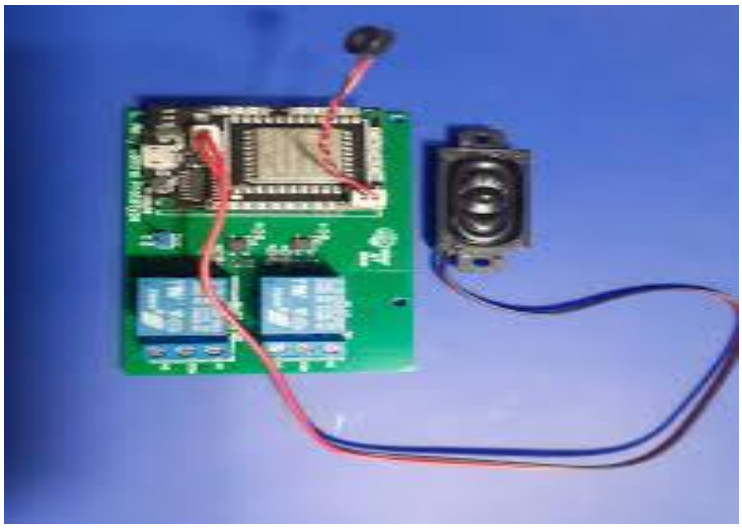


Fig 8.1 Stimulations

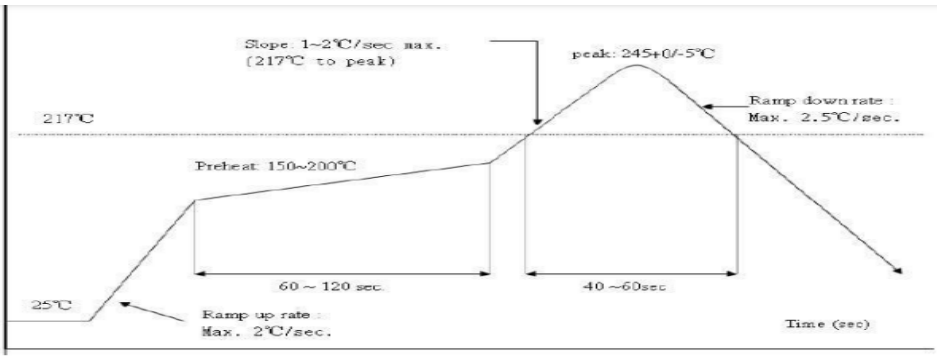


Fig 8.2 wave forms

REFERENCES

1. "Arduino Relay Control Tutorial." Adafruit. Retrieved from https://learn.adafruit.com
2. "Pulse Width Modulation (PWM) for Beginners." All About Circuits. Retrieved from https://www.allaboutcircuits.com
3. "How Relay Modules Work." SparkFun. Retrieved from https://www.sparkfun.com
4. "Using the AI Thinker VC-02 for Voice Control." Random Nerd Tutorials. Retrieved from https://randomnerdtutorials.com
5. "Introduction to PWM." Microchip. Retrieved from https://www.microchip.com

6. "Real-Time Home Automation with IoT." IEEE Xplore. Retrieved from https://ieeexplore.ieee.org

Standards and Specifications

7. IEEE. (2007). *IEEE 1451: Smart Transducer Interface Standard*. IEEE.
8. ISO/IEC. (2001). *ISO/IEC 14543-3: Home and Building Electronic Systems (HBES)*. International Organization for Standardization.

Research Papers and Technical Reports

9. Gupta, A., & Singh, S. (2020). "Voice-Controlled Home Automation System Using Artificial Intelligence." *Springer Link*. https://doi.org/10.1007/978-3-030-37434-7_21
10. Kumar, R., & Saini, H. (2019). "The Role of PWM in Home Automation." *Journal of Electrical Engineering*, 30(4), 212-223.
11. Sharma, M., & Rao, S. (2018). "IoT-Based Home Automation Using Voice Commands." *International Journal of Advanced Research in Computer Science*, 9(2), 88-94.
12. Sharma, A., & Sharma, M. (2021). "Smart Home Automation System Using ESP32 and Alexa." *International Journal of Engineering Research & Technology*, 10(7), 123-130.

Product Documentation and Tutorials

13. AI Thinker. (2021). *AI Thinker VC-02 Documentation*. AI Thinker. Retrieved from https://www.ai-thinker.com
14. Roushan, P. (2020). "ESP32 Relay Control Tutorial." Random Nerd Tutorials. Retrieved from https://randomnerdtutorials.com
15. "Relay Control with ESP8266." Instructables. Retrieved from https://www.instructables.com

*****