ESP32-S3-ETH-8DI-8RO-C

From Waveshare Wiki
Jump to: navigation, search

Introduction

[Collapse]

ESP32-S3-ETH-8DI-8RO-C

(https://www.waveshare.com/esp32-s3-eth-8di-8ro-c.htm)

8-ch, WiFi/CAN/USB

Introduction

ESP32-S3-ETH-8DI-8RO-C is an industrial-grade 8-channel WiFi network relay based on ESP32-S3 main controller and supports WiFi, Bluetooth, CAN, network port and other peripheral interfaces. It has built-in protection circuits such as power supply isolation and optocoupler isolation, which is safe, stable and reliable, and suitable for the AIoT field.

Features

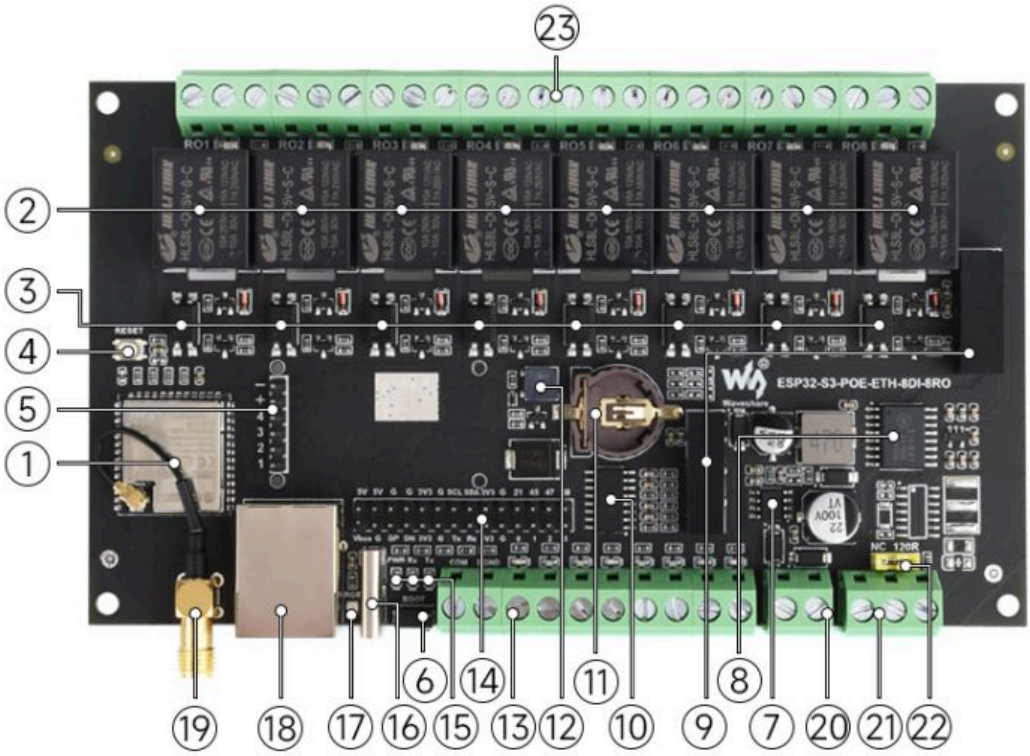
- Based on ESP32-S3 microcontroller with Xtensa 32-bit LX7 dual-core processor with frequency up to 240MHz
- Integrated 2.4GHz WiFi and Bluetooth LE dual-mode wireless communication, with superior RF performance
- High quality relay, contact rating: ≤10A 250V AC or ≤10A 30V DC
- Onboard isolated CAN interface, for connecting to various CAN expansion modules or sensors
- Onboard GPIO pin header interface, for expand other devices such as RS232 / sensor and other functions
- Onboard USB Type-C port for power supply, firmware downloading and debugging, making development more convenient
- Onboard power supply screw terminal, supports 7~36V wide voltage input, suitable for industrial applications
- Onboard optocoupler isolation to prevent interference with control chip from external high-voltage circuit connected to the relay
- Onboard digital isolation to avoid external signal interference with the control chip
- Onboard integrated power isolation, providing stable isolation voltage, no extra power supply required for the isolated terminal
- Built-in buzzer, RGB colorful LED, power supply and CAN TX/RX indicators for monitoring the operating status of the module

- Rail-mounted ABS protective enclosure, easy to install, safe to use

Specifications

Item	Parameter
Supply voltage	7~36V (or 5V/1A Type-C port)
Relay channels	8 channels
Digital input channels	8 channels
Contact Form	1NO 1NC
Wiring port	Type-C
Communication protocol	USB protocol
Dimensions	90 (H) x 175 (V) (mm)

Onboard Resources



(/wiki/File:ESP32-S3-ETH-8DI-8RO-C-details-inter.jpg)

1. ESP32-S3

Up to 240MHz operating frequency, with 2.4GHz WiFi and BLE

2. High-quality relay

Contact rating per channel: ≤10A 250V AC or ≤10A 30V DC

3. Optocoupler isolation

Prevents interference with the control chip from external high-voltage circuit connected to the relay

4. RESET button

5. PoE port

Can be connected to PoE module (PoE port version only)

6. BOOT button

7. Power chip

8. Digital isolation

Prevents interference with the control chip from external signal

9. Power isolation

Provides a stable isolation voltage, needs no extra power supply for the isolated terminal

10. Bidirectional optocoupler isolation

11. RTC battery header

12. Buzzer

13. Digital input interface

14. Pin header interface

Can be connected to other devices

15. Operating status indicator

PWR: Power indicator

RXD: CAN RX indicator

TXD: CAN TX indicator

16. USB Type-C interface

For module power supply, firmware downloading and USB communication

17. WS2812 RGB colorful LED

Controllable via GPIO38 pin

18. Network port

19. External antenna socket

SMA female connector, for WiFi and Bluetooth wireless communication

20. Power supply screw terminal

Supports 7~36V DC wide voltage range power supply

21. CAN communication interface

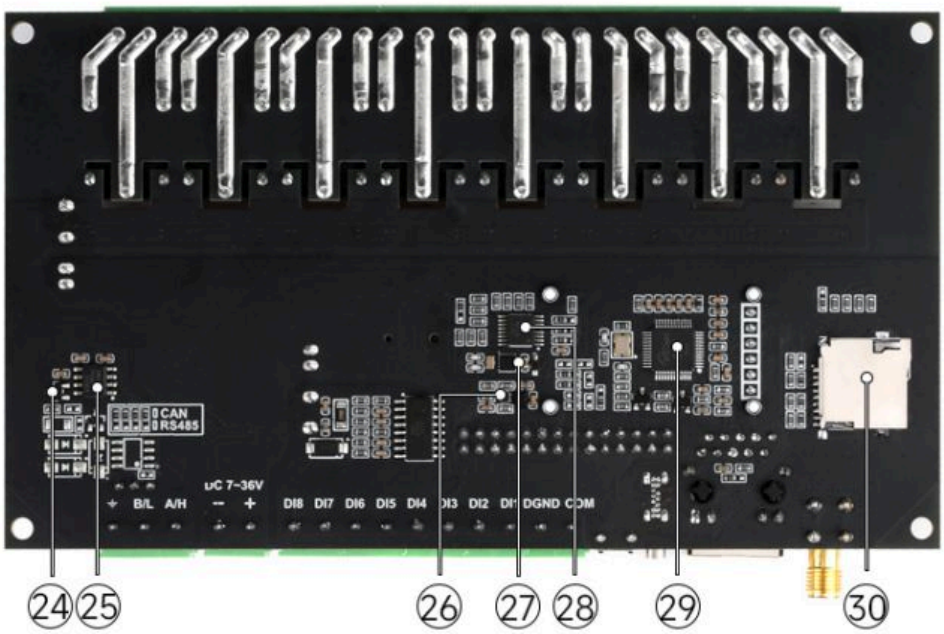
Supports connecting external CAN devices

22. CAN matching resistor

Onboard reserved 120R matching resistor, enabled via jumper

23. Relay screw terminal

Convenient for users to connect devices



(/wiki/File:ESP32-S3-ETH-8DI-8RO-C-details-inter-1.jpg)

24. Onboard TVS (Transient voltage suppressor)

Effectively suppresses surge voltage and transient spike voltage in the circuit

25. CAN conversion chip

26. MP1605GTF-Z

DC-DC power module

27. PCF85063ATL

RTC chip for some timed tasks

28. TCA9554PWR

Expands IO for relay control

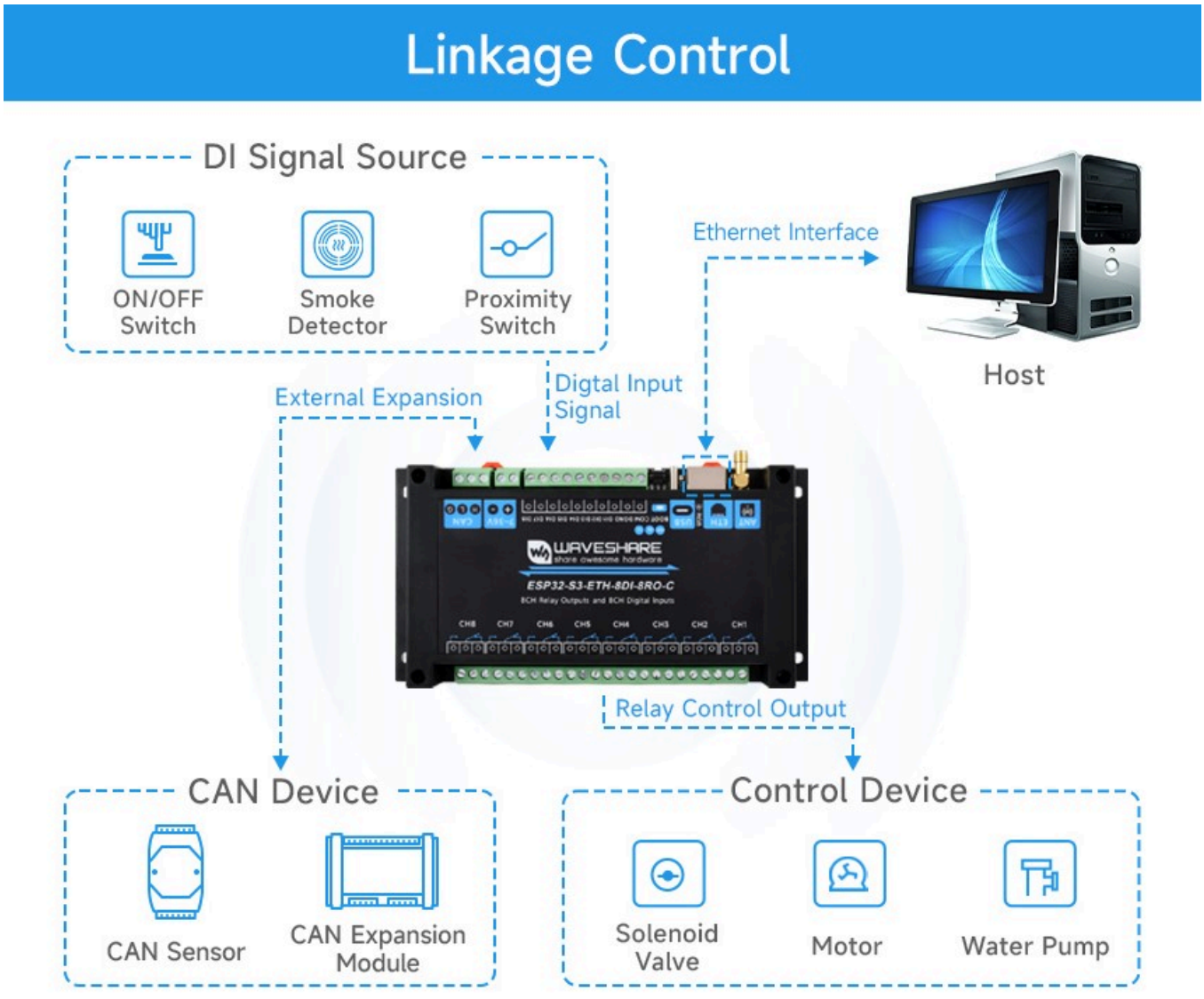
29. W5500

Expands 10/100Mbps network connection via SPI interface

30. TF card slot

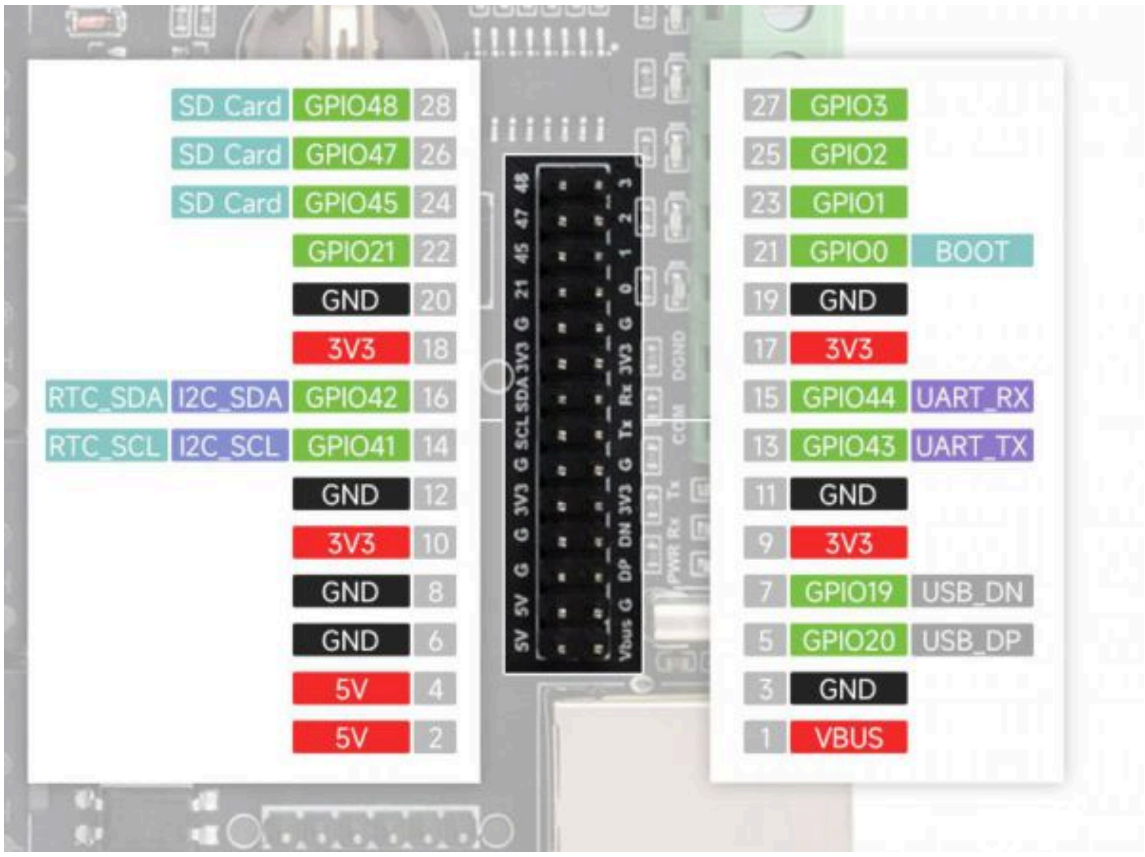
Supports external TF card storage for images and files

Interfaces



(/wiki/File:ESP32-S3-ETH-8DI-8RO-C-details-11.jpg)

- Pin header



(/wiki/File:900px-ESP32-S3-ETH-8DI-8RO-details-2_(1).jpg)

- Relay

Control EXIO	Function
EXIO1	Relay 1 control pin
EXIO2	Relay 2 control pin
EXIO3	Relay 3 control pin
EXIO4	Relay 4 control pin
EXIO5	Relay 5 control pin
EXIO6	Relay 6 control pin
EXIO7	Relay 7 control pin
EXIO8	Relay 8 control pin

- Digital input

Control EXIO	Function
GPIO4	Digital input 1 detection pin
GPIO5	Digital input 2 detection pin
GPIO6	Digital input 3 detection pin
GPIO7	Digital input 4 detection pin
GPIO8	Digital input 5 detection pin
GPIO9	Digital input 6 detection pin
GPIO10	Digital input 7 detection pin
GPIO11	Digital input 8 detection pin

- W5500 network port chip

Control GPIO	Function
GPIO12	ETH_INT
GPIO13	ETH_MOSI

GPIO14	ETH_MISO
GPIO15	ETH_SCLK
GPIO16	ETH_CS

- TF Card

Control GPIO	Function
GPIO45	MISO
GPIO47	MOSI
GPIO48	SCLK
NC	CS
NC	SD_D1
NC	SD_D2

- CAN

Control GPIO	Function
GPIO17	CAN TX pin, corresponding to TWAI TX pin
GPIO18	CAN RX pin, corresponding to TWAI RX pin

- RTC

Control GPIO	Function
GPIO41	RTC_SCL, I2C clock line
GPIO42	RTC_SDA, I2C data line

- RGB light beads

Control GPIO	Function
GPIO38	RGB control pin

- Buzzer

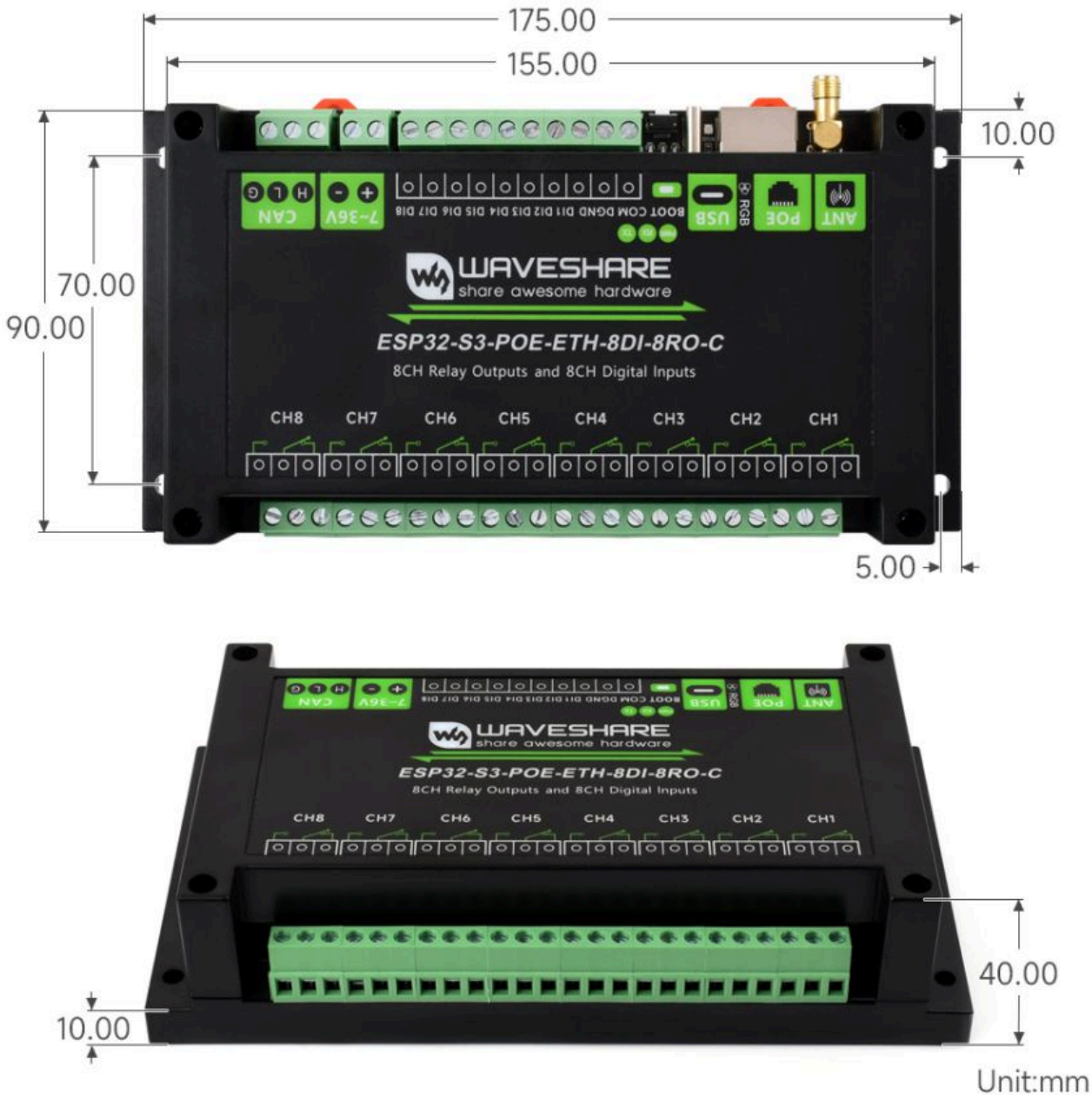
Control GPIO	Function
GPIO46	Buzzer control pin

- BOOT button

Control GPIO	Function

GPIO0	BOOT button control pin

Dimensions



(/wiki/File:ESP32-S3-ETH-8DI-8RO-C-details-size-1.jpg)

Electrical and Relay Safety Instructions

- This product must be operated by professional electricians or qualified personnel. During use, ensure electrical safety, leakage protection, and proper insulation.
- Before installing, maintaining, or replacing the relay device, always turn off the power and unplug the device.
- Do not attempt to disassemble the relay device to avoid damage or the risk of electric shock.

- Properly install and place the relay device. Do not use it in humid, overheated, flammable, or explosive environments to prevent accidents caused by improper installation or use.

1. Load Matching

- Ensure the relay's rated voltage and current match the load. Do not exceed the rated capacity.
- For inductive loads (motors, coils, lamps, etc.), the starting current may be much higher than the rated current. Choose a relay with sufficient current margin.

2. Short Circuit and Overcurrent Protection

- Install a **fuse** or **circuit breaker** in the relay circuit to prevent damage due to short circuits or accidental overcurrent.
- Ensure the load circuit has no short circuits during wiring, and select protection components with appropriate current ratings if necessary.

3. Arc and Switching Protection

- Relay switching generates arcs, which can cause contact wear or welding.
- For inductive loads, it is recommended to use **RC snubber circuits** or **varistors** for arc suppression.

4. Installation Environment

- Do not use the relay in humid, high-temperature, flammable, explosive, or dusty environments.
- Install the relay securely to avoid vibrations or shocks that may cause misoperation or damage.

5. Power-Off Operation

- Always cut off power before maintenance, wiring, or replacing the relay to ensure personnel and device safety.
- Latching relays are only powered when changing state. Avoid strong vibrations or strong magnetic fields while the relay is unpowered.

6. Status Confirmation

- After powering on, confirm or reset the relay status as needed to prevent abnormal operation caused by transportation, installation, or external disturbances.
- Avoid power interruption during relay operation to prevent uncertain status or contact damage.

7. Regular Inspection

- Periodically inspect relay contacts, terminals, and insulation to ensure proper operation.
- If abnormal heating, odor, or burn marks are detected, immediately cut off power and replace the relay.

Version Description

This development board includes both common network port and POE Ethernet port versions. Please click on the corresponding product to view the usage instructions.

(https://www.waveshare.com/esp32-s3-eth-8di-8ro-c.htm?sku=31282)

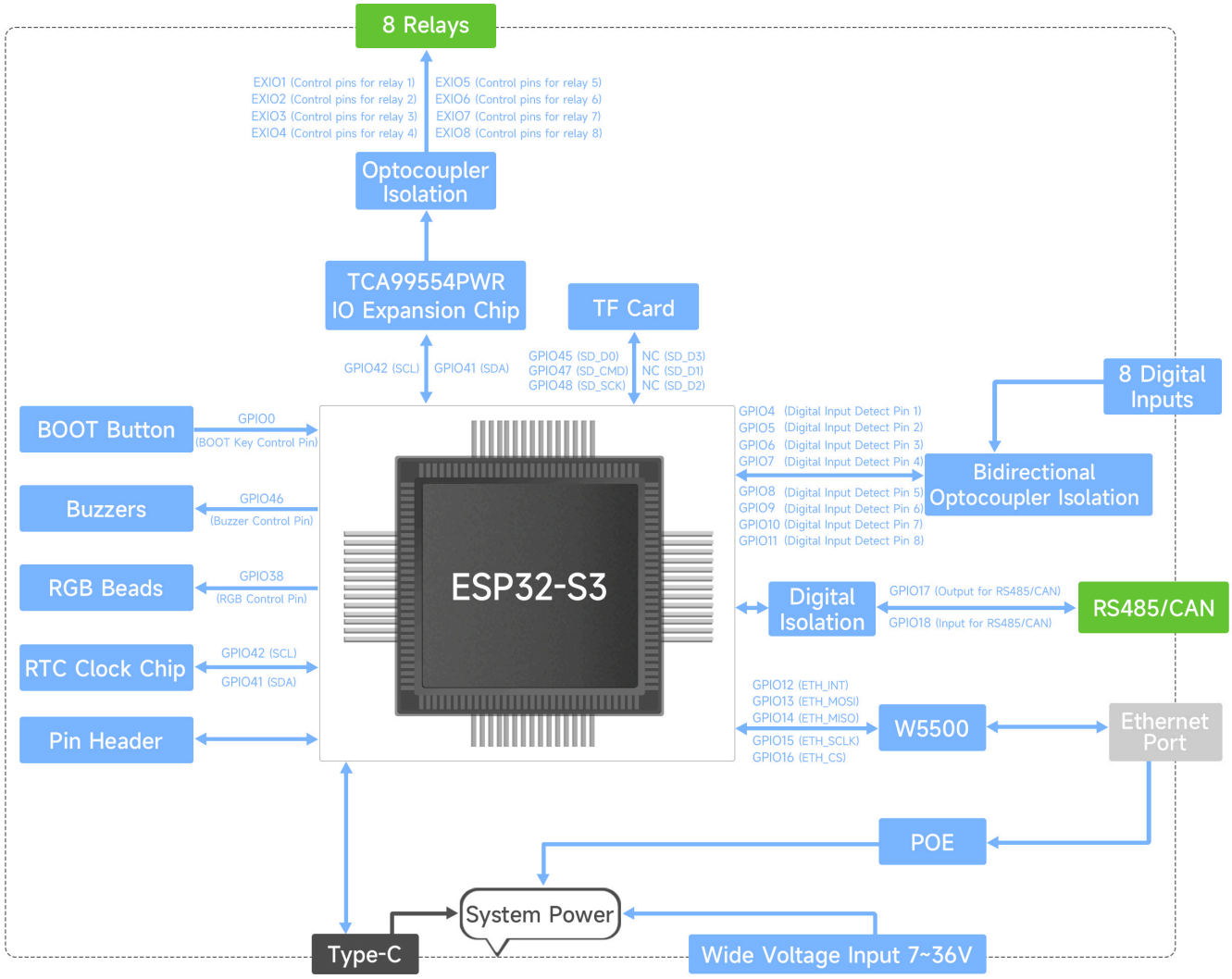
ESP32-S3-ETH-8DI-8RO-C
Common network port

(https://www.waveshare.com/esp32-s3-eth-8di-8ro-c.htm?sku=31284)

ESP32-S3-POE-ETH-8DI-8RO-C (/wiki/ESP32-S3-POE-ETH-8DI-8RO-C)
POE Ethernet port

Note: When transmitting data via CAN, a frame is only considered complete after looping back to itself. Therefore, during data transmission, both the TX (green light) and RX (blue light) indicator lights may flash simultaneously, which is normal behavior.

Implementation Logic



(/wiki/File:Implementation_logic.jpg)

Usage Instructions

ESP32-S3-ETH-8DI-8RO-C currently provides the **Arduino IDE** development tool and

framework.

Development Tool

(/wiki/File:270px-Arduino-IDE-logo.jpg)

Arduino IDE

Arduino IDE is an open source electronic prototyping platform, convenient and flexible, easy to get started. After a simple learning, you can start to develop quickly. At the same time, Arduino has a large global user community, providing an abundance of open source code, project examples and tutorials, as well as rich library resources, encapsulating complex functions, allowing developers to quickly implement various functions.

Each of these two development approaches has its own advantages, and developers can choose according to their needs and skill levels. Arduino and MicroPython are suitable for beginners and non-professionals because they are easy to learn and quick to get started.

Components Preparation

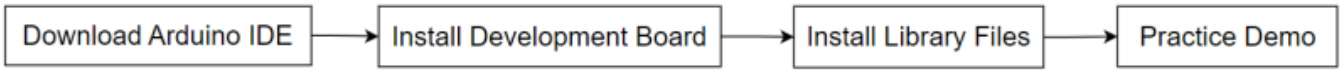
- ESP32-S3-ETH-8DI-8RO-C x1
- USB cable (Type-A male to Type-C male) x1
- Modbus RTU Relay (https://www.waveshare.com/Modbus-RTU-Relay.htm) x 1
- USB TO RS232/485 (https://www.waveshare.com/USB-TO-RS485-B.htm) x 1
- USB-CAN-A (https://www.waveshare.com/USB-CAN-A.htm) x 1

Before operating, it is recommended to browse the table of contents to quickly understand the document structure. For smooth operation, please read the FAQ carefully to understand possible problems in advance. All resources in the document are provided with hyperlinks for easy download.

Working with Arduino

This chapter introduces setting up the Arduino environment, including the Arduino IDE, management of ESP32 boards, installation of related libraries, program compilation and downloading, as well as testing demos. It aims to help users master the development board

and facilitate secondary development.

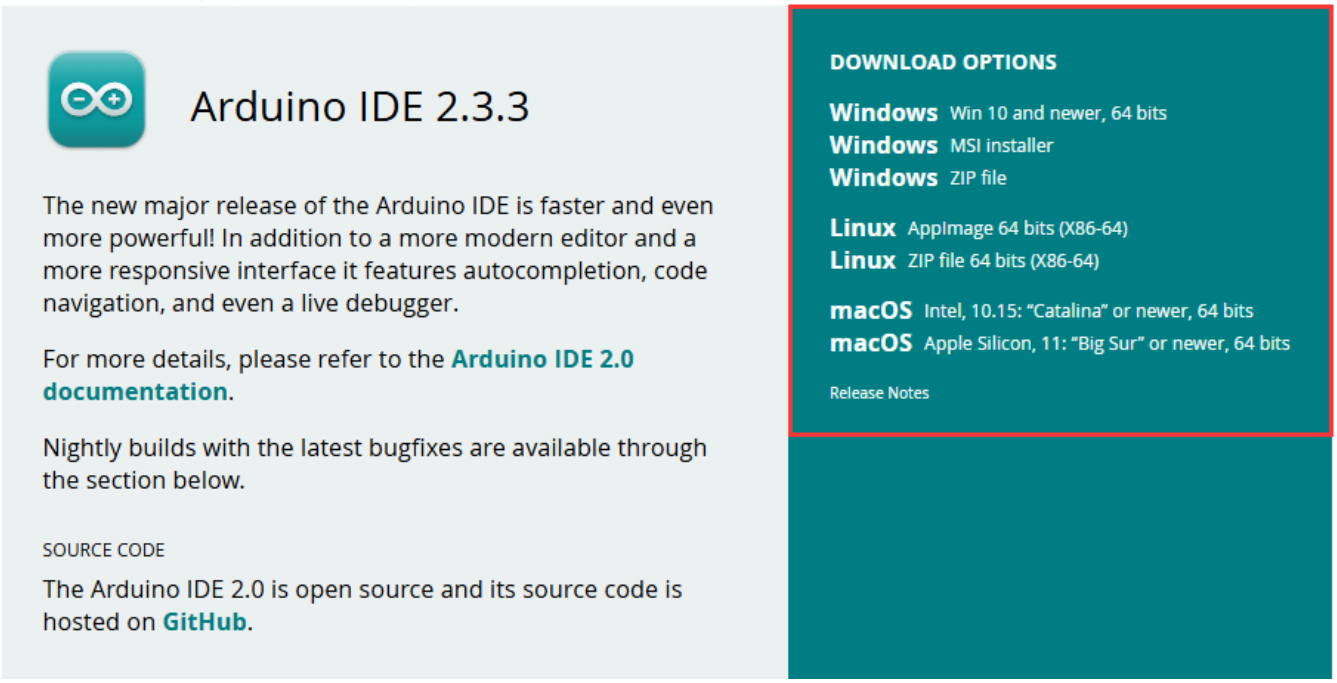


(/wiki/File:Arduino-flow-04.png)

Environment Setup

Download and Install Arduino IDE

- Click to visit the Arduino official website (https://www.arduino.cc/en/software), select the corresponding system and system bit to download



(/wiki/File:ESP32-S3-AMOLED-1.91-Ar-software-01.png)

- Run the installer and install all by default

Install ESP32 Development Board

- Regarding ESP32-related motherboards used with the Arduino IDE, the **esp32 by Espressif Systems** library must be installed first.
- According to **Board installation requirement**, it is generally recommended to use **Install Online**. If online installation fails, use **Install Offline**
- For the installation tutorial, please refer to Arduino board manager tutorial (https://www.waveshare.com/wiki/Arduino_Board_Managers_Tutorial)

- ESP32-S3-ETH-8DI-8RO-C required development board installation description

Board name	Board installation requirement	Version number requirement
esp32 by Espressif Systems	"Install Offline" / "Install Online"	3.0.0 and above

Install Library

- When installing Arduino libraries, there are usually two ways to choose from: **Install online** and **Install offline**. **If the library installation requires offline installation, you must use the provided library file**
  For most libraries, users can easily search and install them through the online library manager of the Arduino software. However, some open-source libraries or custom libraries are not synchronized to the Arduino Library Manager, so they cannot be acquired through online searches. In this case, users can only manually install these libraries offline.
- For library installation tutorial, please refer to Arduino library manager tutorial (https://www.waveshare.com/wiki/Arduino_Library_Manager_Tutorial)
- ESP32-S3-ETH-8DI-8RO-C library file is stored in the demo, click here to jump: ESP32-S3-ETH-8DI-8RO-C Demo

- ESP32-S3-ETH-8DI-8RO-C library file installation description

Library Name	Description	Version	Library Installation Requirement
ArduinoJson	Lightweight JSON library	v6.21.4	"Install Online" or "Install Offline"
PubSubClient	MQTT message subscription publishing library	v2.8.0	"Install Online" or "Install Offline"
NTPClient	Network time synchronization client library	v3.2.1	"Install Online" or "Install Offline"

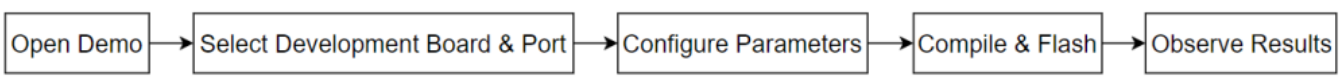
> For more learning and use of LVGL, please refer to LVGL official documentation (https://docs.lvgl.io/master/intro/introduction/index.html)

Run the First Arduino Demo

> If you are just getting started with ESP32 and Arduino, and you don't know how to create, compile, flash, and run Arduino ESP32 programs, then please expand and take a look. Hope it can help you!

[Expand]

Demo

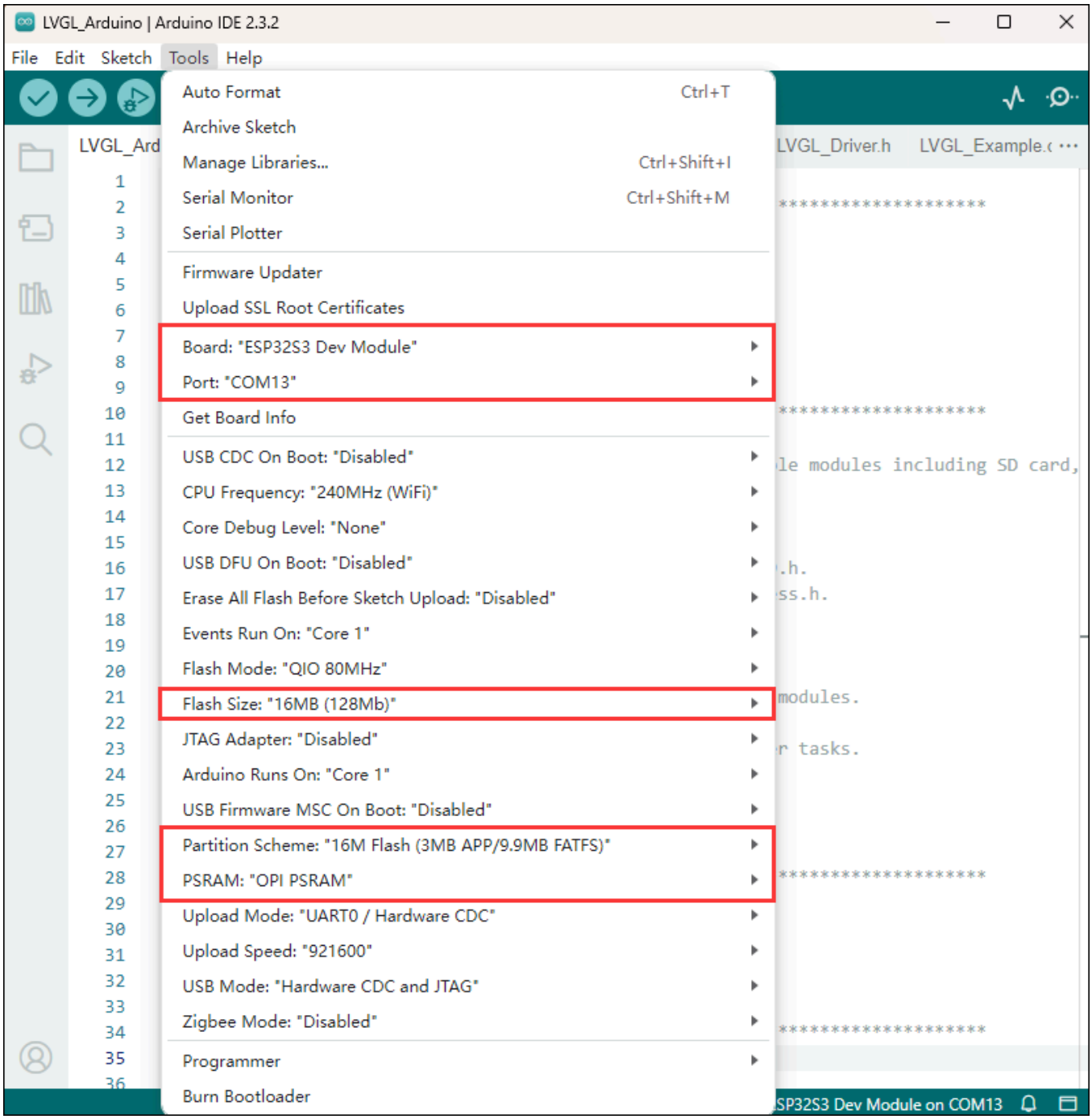


(/wiki/File:Demo-flow-01.png)

- ESP32-S3-ETH-8DI-8RO-C demos

Demo	Basic Description	Dependency Library
01_MAIN_WIFI_AP	Bluetooth control, Bluetooth send IP, Web page control (short distance)	Can be flashed directly Web pages are only available if they are connected to device WIFI
02_MAIN_WIFI_STA	Bluetooth control, Bluetooth send IP, Web page control (short distance)	Need to **modify** the WIFI to be connected Web pages can only be used within the intranet
03_MAIN_WIFI_MQTT	Bluetooth control, Bluetooth transmit IP, Waveshare cloud control (long distance)	Need to **modify** the WIFI to be connected The device must be **created** in the Waveshare cloud (**Note: Waveshare Cloud is no longer maintained. Please choose another cloud service.**)
04_MAIN_ALL	Bluetooth control, Bluetooth send IP, Web page control (short distance), Waveshare cloud control (long distance)	Need to **modify** the WIFI to be connected The device must be **created** in the Waveshare cloud Web pages can only be used within the intranet (**Note: Waveshare Cloud is no longer maintained. Please choose another cloud service.**)

Arduino Project Parameter Setting



(/wiki/File:ESP32-S3-16MB-OPI-Parameters.png)

01_MAIN_WIFI_AP

Demo description

- This example implements the control of 8 relays on and off through WiFi, Bluetooth, and network port. The AP mode of WiFi is enabled in this example

Precautions

- Can be flashed directly

- The web page can only be accessed by connecting to the device's WIFI, or if connected to a network cable, by accessing the network port IP within the internal network

Code analysis

- **Relay_Analysis()**: This function is mainly used to analyze received data, perform corresponding relay control operations based on the data content, and output corresponding status prompt information.
  - **Parameter analysis**
    - `uint8_t *buf` : A pointer to an unsigned 8-bit integer array, which is supposed to store the received data, and the function determines the content of the command based on the value of the first element (`buf[0]`) of the array.
    - `uint8_t Mode_Flag` : The mode flag is used to represent the data source, and the corresponding data source prompt information (such as Bluetooth data, Wi-Fi data) is output by judging this flag, and different relay operations are performed according to different commands.
  - **Logical flow**
    - First output the corresponding data source prompt message according to the value of `Mode_Flag` .
    - Then use the `switch` statement to perform different operations based on the value of `buf[0]` :
      - For the cases from `CH1` to `CH8` , the `digitalToggle` function is used to switch the level states of the corresponding GPIO pins (such as `GPIO_PIN_CH1` , etc.), and the corresponding elements of the `Relay_Flag` array are updated to record the change in relay status. The `Buzzer_PWM` function is called to control the buzzer, and the corresponding on or off prompt information is output according to the final state of the relay.
      - For the `ALL_ON` command, set all GPIO pins (corresponding to 8 channel relays) to high level (on state), use the `memset` function to set all elements of the `Relay_Flag` array to 1, indicating that all relays are turned on, output all relay ON prompt information, and control the buzzer.
      - For the `ALL_OFF` command, set all relevant GPIO pins to low level (off state) similarly, update the `Relay_Flag` array elements to 0, output a message that all relays are turned off and control the buzzer, and perform two additional buzzer control operations (with a delay in between).
      - If the value of `buf[0]` does not belong to the above command cases, a prompt message for non-instruction data is output.
- **WIFI_Init ()**: Configure the device as a Wi-Fi access point (AP), set up a Web server, and set up request processing functions corresponding to different paths to implement related network functions and device control functions, and give corresponding status prompts.
  - AP creation
    - First set the Wi-Fi mode to `WIFI_AP` , and then attempt to create a soft AP through `WiFi.softAP(ssid, password)` . If it fails, it will loop through prompt

and retry until it succeeds
- Parameter configuration and prompts
  - After successfully creating the AP, give a prompt with RGB light (green light on for 1 second)
- IP address display
  - Obtain and format the IP address of the soft AP, store it in `ipStr` and output it for easy network address recognition
- Web server setting
  - Set up corresponding request handling functions (such as `handleRoot`, `handleGetData`, etc.) for multiple paths (such as "/", "/getData", etc.) through `server.on`, each function should be defined elsewhere for different functional operations like returning pages, getting data, controlling switches etc.
  - Finally call `server.begin()` to start the Web server and output a startup prompt message.

CAN interface

- When receiving CAN data, the received CAN data will be printed

Bluetooth Control

- Jump to Bluetooth Control Tutorial (https://www.waveshare.com/wiki/ESP32-S3-POE-ETH-8DI-8RO-Bluetooth)

Web Page Control

- Jump to Web Control Tutorial (https://www.waveshare.com/wiki/ESP32-S3-POE-ETH-8DI-8RO-Web), this example is in AP mode

02_MAIN_WIFI_STA

Demo description

- This example implements the control of 8 relays on and off through WiFi and Bluetooth. The STA mode of WiFi is enabled in this example

Precautions

- Need to modify the WIFI to be connected
- The Web page only supports controlling the device and this product to be used under the same network, if you use the mobile phone control, you need to turn off the mobile

network

Code analysis

- **WIFI_Init ()**: Connect the device to a specified Wi-Fi network in station (STA) mode. If the connection is successful, it will perform subsequent network-related configurations (such as obtaining an IP address, starting a Web server, and registering callback functions, etc.). If the connection fails, it will give a corresponding prompt and set the connection status flag. The process also provides visual connection status indicators through an RGB light
  - Attempt to connect to Wi-Fi network:
    - First set Wi-Fi to `WIFI_STA` mode and enable sleep mode, then start connecting to the specified network. In the cyclic wait of unsuccessful connection, a point is output as a prompt every half second, and the RGB light briefly turns red on every even number of attempts (except the first attempt). If every 10 attempts fail, the connection is disconnected and reconnected. If the number of attempts exceeds 22, the connection is considered a failure and the loop is exited.
  - Operations after successful connection:
    - If the number of connection attempts is less than 23 (i.e. successful connection), set `WIFI_Connection` to 1, light up the green light for 1 second to indicate success, and then obtain and display the local IP address. Next, register callback functions corresponding to multiple paths for the web server (such as root path, data retrieval, control of different switches, etc.), and finally start the server and output startup prompt, so that the device can receive corresponding control through the web page.
  - Operations after connection failure:
    - If the number of attempts is greater than or equal to 23 (connection failure), set `WIFI_Connection` to 0, output a prompt to inform that the device can be controlled through Bluetooth debugging assistant, and light a red light to indicate a connection failure status

CAN interface

- When receiving CAN data, the received CAN data will be printed

Bluetooth Control

- Jump to Bluetooth Control Tutorial (https://www.waveshare.com/wiki/ESP32-S3-POE-ETH-8DI-8RO-Bluetooth)

Web Page Control

- Jump to Web Control Tutorial (https://www.waveshare.com/wiki/ESP32-S3-POE-ETH-8DI-8RO-Web), this example is in STA mode

03_MAIN_WIFI_MQTT

Note: Waveshare Cloud is no longer maintained. Please choose another cloud service.

Demo description

- This example controls 8 relays on and off through MQTT and Bluetooth. It uses ESP32 as the main control unit, supports Wi-Fi and Bluetooth connections, and provides remote control based on the MQTT protocol

Precautions

- Need to modify the WIFI to be connected
- Must create the device in Waveshare Cloud

Code analysis

- **Relay_Analysis()**: Receive data from different communication sources and perform corresponding relay control operations
  - **Data Delivery**: Commands (such as `CH1`, `ALL_ON` etc.) are sent to the device via Bluetooth or Wi-Fi. The device parses and executes the corresponding operation after receiving a command. For example, when the command `CH1` is received, the function will switch the state of relay 1
  - **Data Feedback**: The execution of control commands is printed to the serial monitor via `printf` (e.g., relay status update: "Relay CH1 on") and fed back via the buzzer `Buzzer_PWM`). This allows users to see real-time status updates
  - **Communication Protocol**
    - Bluetooth: Wireless communication with mobile phones or other devices through Bluetooth modules, and the device controls the relay after receiving the command
    - MQTT: Connect to the MQTT server via Wi-Fi, and the device subscribes to a specific topic (such as relay control commands). When a new command is issued, the device receives the message via MQTT and performs relay control
- **setup ()**
  - **Initialize the various modules required for the system, including serial port, GPIO, RTC, Bluetooth and Wi-Fi (MQTT)**
    - Call `Bluetooth_Init()`, and the device can establish connections with other Bluetooth devices and receive control commands
    - With `MQTT_Init()`, the device will connect to the Wi-Fi network and be able to communicate with remote servers through the MQTT protocol

- **Time Synchronization**
  - If the system is connected to Wi Fi and RTC is enabled, the `Acquisition_time()` function will synchronize the current time to RTC through the network, ensuring that the device has accurate system time
- **Data Upload and Delivery**
  - Upload: When the device successfully connects to Wi Fi or Bluetooth, it can upload device status information or sensor data (such as temperature, humidity) through MQTT. These data will be sent to the MQTT server regularly for other systems or users to view
  - Delivery: The device receives a control command from the MQTT server and performs corresponding actions, such as switching relays on and off or changing configurations

CAN interface

- When receiving CAN data, the received CAN data will be printed

Bluetooth Control

- Jump to Bluetooth Control Tutorial (https://www.waveshare.com/wiki/ESP32-S3-POE-ETH-8DI-8RO-Bluetooth)

Waveshare Cloud Control

- Jump to Waveshare Cloud Control (https://www.waveshare.com/wiki/ESP32-S3-POE-ETH-8DI-8RO-Waveshare_Cloud)

04_MAIN_ALL

Note: Waveshare Cloud is no longer maintained. Please choose another cloud service.

Demo description

- This example is a collection of Bluetooth control, Web page control (near distance), and Waveshare cloud control (long distance)

Precautions

- Need to modify the WIFI to be connected
- The Web page only supports controlling the device and this product to be used under the same network, if you use the mobile phone control, you need to turn off the mobile

network
- Must create the device in Waveshare Cloud

CAN interface

- When receiving CAN data, the received CAN data will be printed

Bluetooth Control

- Jump to Bluetooth Control Tutorial (https://www.waveshare.com/wiki/ESP32-S3-POE-ETH-8DI-8RO-Bluetooth)

Web Page Control

- Jump to Web Control Tutorial (https://www.waveshare.com/wiki/ESP32-S3-POE-ETH-8DI-8RO-Web), this example is in AP mode

Waveshare Cloud Control

- Jump to Waveshare Cloud Control (https://www.waveshare.com/wiki/ESP32-S3-POE-ETH-8DI-8RO-Waveshare_Cloud)

Inteface Description

GPIO	Relay	DIN	RTC	CAN	Network	EXIO	SD Card	Other
GPIO0								GPIO0
GPIO1								ADC_IO1
GPIO2								ADC_IO2
GPIO3								ADC_IO3
GPIO4		IN1						
GPIO5		IN2						
GPIO6		IN3						
GPIO7		IN4						
GPIO8		IN5						
GPIO9		IN6						
GPIO10		IN7						
GPIO11		IN8						
GPIO12					Net_INTn			
GPIO13					Net_MOSI			
GPIO14					Net_MISO			
GPIO15					Net_SCLK			
GPIO16					Net_SCSn			
GPIO17				TXD1				
GPIO18				RXD1				
GPIO19								D_N
GPIO20								D_P
GPIO21							SD_CS	GPIO21
GPIO33								Internal occupancy
GPIO34								Internal occupancy
GPIO35								Internal occupancy
GPIO36								Internal occupancy
GPIO37								Internal occupancy
GPIO38								RGB_CTRL
GPIO39					Net_RSTn			
GPIO40			RTC_INT					
GPIO41			RTC_SCL			EXIO_SCL		SCL
GPIO42			RTC_SDA			EXIO_SDA		SDA
GPIO43								TXD0
GPIO44								RXD0
GPIO45							SD_MISO	GPIO45
GPIO46								Buzzer
GPIO47							SD_MOSI	GPIO47
GPIO48							SD_SCLK	GPIO48
Extend_IO1	CH1							
Extend_IO2	CH2							
Extend_IO3	CH3							
Extend_IO4	CH4							
Extend_IO5	CH5							
Extend_IO6	CH6							
Extend_IO7	CH7							
Extend_IO8	CH8							

(/wiki/File:ESP32-S3-ETH-8DI-8RO-C-logic-1.png)

Flash Firmware Flashing and Erasing

- **The current demo provides test firmware, which can be used to test whether the onboard device functions properly by directly flashing the test firmware**
- **bin file path:**

```
...\ESP32-S3-POE-ETH-8DI-8RO-C-Demo\Firmware\Factory bin
```

Flash firmware flashing and erasing (https://www.waveshare.com/wiki/Flash_Firmware_Flashing_and_Erasing) for reference

Working with Homeassistant

The product can be controlled online through the Homeassistant built on the Raspberry Pi, and the relevant operations can be found in Reference link (https://www.waveshare.com/wiki/Using_Modbus_RTU_Relay_with_Homeassistant)

Resources

Demo

- ESP32-S3-POE-ETH-8DI-8RO-C Demo (https://files.waveshare.com/wiki/ESP32-S3-ETH-8DI-8RO-C/ESP32-S3-POE-ETH-8DI-8RO-C-Demo.zip)

Datasheets

ESP32-S3

- ESP32-S3 Datasheet (https://files.waveshare.com/wiki/common/Esp32-s3_datasheet_en.pdf)
- ESP32-S3 Technical reference manual (https://files.waveshare.com/wiki/common/Esp32-s3_technical_reference_manual_en.pdf)
- ESP32-S3-WROOM-1 Datasheet (https://files.waveshare.com/wiki/common/Esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf)

Software Tools

Arduino

- Arduino IDE Official download link (https://www.arduino.cc/en/software/)

- ESP32_Arduino offline package (https://drive.google.com/drive/folders/1Pcs_A4FKWvdSHnz9lEBYqOpr-noTMbIv?usp=sharing)

VScode

- VScode official website (https://code.visualstudio.com/download)

Thonny

- Thonny official download link (https://thonny.org/)

Debugging tool

- Flash_download_tool (https://dl.espressif.com/public/flash_download_tool.zip)
- SSCOM (https://files.waveshare.com/wiki/ESP32-S3-Relay-6CH/SSCOM5.13.1_For_ESP32_S3_Relay_6CH.zip)
- Bluetooth debugging tool (nRF Connect) (https://www.nordicsemi.com/Products/Development-tools/nRF-Connect-for-mobile)
- USB-CAN-A_TOOL (https://files.waveshare.com/wiki/common/USBCANV2.10.zip)

Other Resource Link

- ESP32-Arduino official documentation (https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html)

Project Resources

This section features third - party project resources. We merely provide links and bear no responsibility for content updates or maintenance. Thank you for your understanding.

trailcurrentopensource - A Compact CAN-Bus Relay Board for Vehicle Builds

- Youtube: https://www.youtube.com/watch?v=SJMmEwOoy_s (https://www.youtube.com/watch?v=SJMmEwOoy_s)

- Github: https://github.com/trailcurrentoss (https://github.com/trailcurrentoss)

FAQ

Question: After the module downloads the demo and re-downloads it, why sometimes it can't connect to the serial port or the flashing fails?

Answer:

- Long press the BOOT button, press RESET at the same time, then release RESET, then release the BOOT button, at this time the module can enter the download mode, which can solve most of the problems that can not be downloaded.

Question: Why does the module keep resetting and flicker when viewed the recognition status from the device manager?

Answer:

- It may be due to Flash blank and the USB port is not stable, you can long-press the BOOT button, press RESET at the same time, and then release RESET, and then release the BOOT button, at this time the module can enter the download mode to flash the firmware (demo) to solve the situation.

Question: How to deal with the first compilation of the program being extremely slow?

Answer:

- It's normal for the first compilation to be slow, just be patient

Question: How to handle the display "waiting for download..." on the serial port after successfully ESP-IDF flashing?

Answer:

- If there is a reset button on the development board, press the reset button; if there is no reset button, please power it on again

Question: What should I do if I can't find the AppData folder?

Answer:

- Some AppData folders are hidden by default and can be set to show.
- English system: Explorer->View->Check "Hidden items"
- Chinese system: File Explorer -> View -> Display -> Check "Hidden Items"

Question: What is the I2C device address for Extended IO (TCA9554PWR)?

Answer:

- The device address is 0x20.

Question: How do I check the COM port I use?

Answer:

- Windows system:

①View through Device Manager: Press the Windows + R keys to open the "Run" dialog box; input devmgmt.msc and press Enter to open the Device Manager; expand the "Ports (COM and LPT)" section, where all COM ports and their current statuses will be listed.
②Use the command prompt to view: Open the Command Prompt (CMD), enter the "mode" command, which will display status information for all COM ports.
③Check hardware connections: If you have already connected external devices to the COM port, the device usually occupies a port number, which can be determined by checking the connected hardware.

- Linux system:

①Use the dmesg command to view: Open the terminal.
①Use the ls command to view: Enter ls /dev/ttyS* or ls /dev/ttyUSB* to list all serial port devices.
③Use the setserial command to view: Enter setserial -g /dev/ttyS* to view the configuration information of all serial port devices.

Question: Why does the program flashing fail when using a MAC device?

Answer:

- Install MAC Driver (https://files.waveshare.com/wiki/common/CH34XSER_MAC.7z) and flash again.

Question: Why is there no output after successfully flashing the code with no issues?

Answer:

- Check the schematic diagram for different development boards with Type-C interfaces, and handle the output accordingly:
  - For development boards with direct USB output, printf function is supported for printing output. If you want to support output via the Serial function, you will need to enable the USB CDC On Boot feature or declare HWCDC.
  - For development boards with UART to USB conversion, both printf and Serial functions are supported for printing output, and there is no need to enable USB CDC On Boot.

Question: Please note!

Answer:

- The factory demo is for learning only, if it is used for practical application, please optimize the demo logic by yourself.

Question: When controlling other devices using CAN, it is not sensitive or communication fails?

Answer:

- Please move the jumper cap to 120R and try again. Some CAN devices require a 120R resistor to be connected in series

Question: Is it normal for the blue and green lights to flash together when the device sends data?

Answer:

- Note: CAN needs to loop back to itself to end a frame of data when sending data, so it is normal for the TX (green) and RX (blue) lights to flash at the same time when the data is sent.

Support

Technical Support

If you need technical support or have any feedback/review, please click the **Submit Now** button to submit a ticket, Our support team will check and reply to you within 1 to 2 working days. Please be patient as we make every effort to help you to resolve the issue.
Working Time: 9 AM - 6 PM GMT+8 (Monday to Friday)

Submit Now (https://service.waveshare.com/)