Raging Bits MQTT

Magic Lights Device

v2.1

Top level specs

1x UART for ESP32
1x UART for Addressable LED driver
1x Addressable LED output

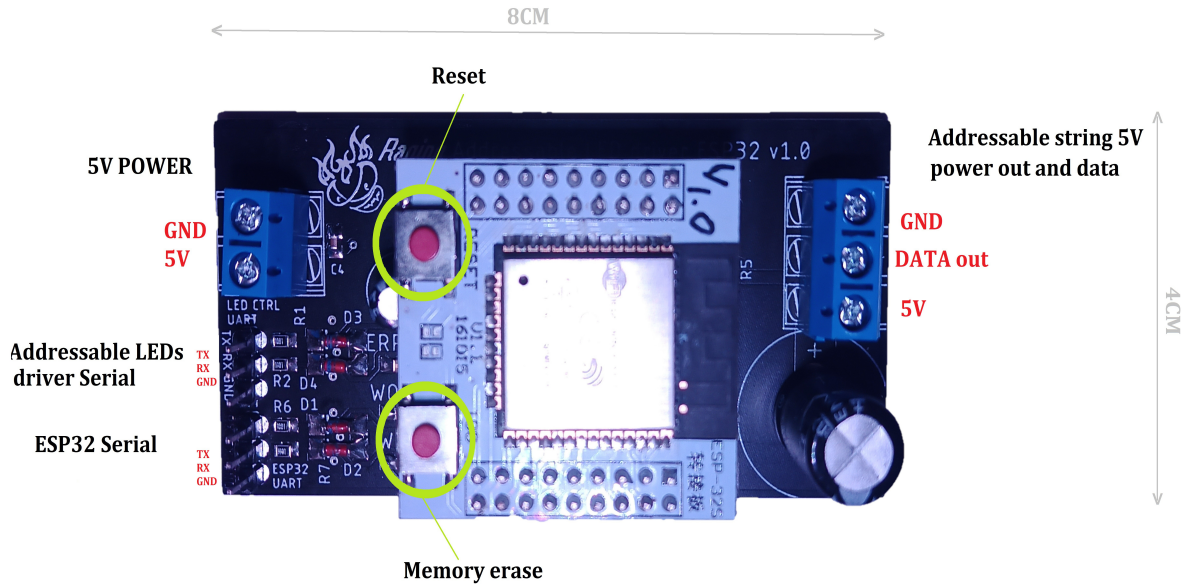
5V Power Supply Input
32Mbit flash dedicated to effects data
Dual control of LEDs – Solid color or Effects run

This device uses an ESP32 Wroom, such that it may be used as a development platform for Arduino.

Default Software (Software on which the datasheet is based)

Full MQTT ESP32 based arduino software
WiFi Hotspot for simple setup
Automatic Hotstart after first time setup

Device hardware interfaces



MQTT Magic Lights Device detail pinout

Hardware IOs pinout from ESP to board functions.

BOARD Function	**ESP32 PIN**
WIFI_LED	IO22
ERROR_LED	IO21
WORK_LED	IO23
ADDRESSABLE_LEDS_DRIVER_RESET	IO05
MEMORY_ERASE_REQUEST/PROGRAM	IO00
ESP32_UART_TX	IO01
ESP32_UART_RX	IO03
ADDRESSABLE_LEDS_DRIVER_UART_TX	IO17
ADDRESSABLE_LEDS_DRIVER_UART_RX	IO16
FLASH_SPI_MISO	IO15
FLASH_SPI_MOSI	IO02
FLASH_SPI_CS	IO04
FLASH_SPI_CLK	IO18

UART

The Uart interface can be used for:

ESP32 Programming

MQTT Generic Device Debug output

It has a maximum of 3.3v line interface level with protection.

The default Uart debug speed is 115200 8N0

Power

The system processors and logical control is powered by the 5V@0.5A input.

Consumptions:

150mA when hotspot active and 80mA with no hotspot active. (This is the unit only.)

The Power for addressable LEDs bus will be powered by the same 5V.

Although the board can support over 5A supply to the LEDs from the input, it's advised to power the LEDs string using a power connector such that the string will feed the device and not the device feeding the string.

Buttons

Reset button, resets the unit.

IO0, is used for both programming the ESP32 and request the main application to erase any settings saved.

Addressable LEDs Bus

The addressable LEDs bus will support mostly of the market regular brands.
SK and WS are currently garanteed to work.
The data format is of RGB (3 components) formats ONLY. Can be set as RGB, GRB, BRG, BGR, etc... for strips of up to 300LEDs.

MQTT Magic Lights Device Default Software

Device Serial Setup

At power up, the device will have a 5second period where waits for a serial number to be given through the UART.

This number will:

Ovewrite the default one permanently

BE **10 chars long**. This is imperative.

Be used for the WIFI hotspot identification.

Be used for MQTT topics interface.

Device Connectivity Setup

The device will have a Hotspot named
"rb_wifi_mqtt_lights_xxxxxxxxxx" where xxxxxxxxxx is the device Unit ID.
Usually this ID is unique and pre configured from factory. It may be modified under specific conditions.

Once powered, the device will load the setup information and try to reconnect using it.
At this point the green Work LED will start blinking.
For the first time, there will be no data so the user will need to connect a wifi and browser capable device to the hotspot.
Once connected, open a browser and in the address bar set the address to
" 1.2.3.4 " and Enter/Proceed.
A webpage will open requesting the following information:

- WiFi details, SSID and paraphrase, where to connect to.
- Network Broker IP and Port.
- RGB data format.

As the device attempts to connect to Wifi, the Error red LED and Work green LED will blink alternatively.

Once the data is filled in, press connect.
At this point the device will save the data and try to connect to the WiFi.
Once connected to the wifi, the red Error LED stops blinking and the blue connectivity LED will blink, indicating the wifi is now connected and it's attempting to connect to the MQTT broker.

Once the device is connected and subscribed to the MQTT broker, the blue connectivity LED stays solid on, while the green work LED blinks to indicate the device work heartbeat.

Once connected the Hotspot will turn off after 10 seconds.

Erase device saved configuration data

If the setup has not been correctly set or needs to be modified, press the ESP32 button labelled IO0 (Program/Erase button), until the Error LED lights up (about 5seconds), indicating the data has been erased.

Once the button is released, the unit will reset and be ready at default state.
At this point follow **Device Setup** instructions again.

*Effects file *.leds.esp32*

The file is pure binary with the following structure:

32Bit - Total effects length in bytes.

16Bit - Strip size in bytes. (3bytes per led).

Leds data in 24bit RGB format.

Generating *.leds.esp32 file from LEDMatrixStudio output *.leds file.

Open a command line and run:

python AddressableLEDsConverterForESP32.py -f "<file_path>\<file_name>.leds"

The output will be:

<file_path>\<file_name>.leds.esp32

Note:

This command needs python instaled to run.

Updating the device effects

By default, the device will bring a basic set of demo 50LED strip examples.
More, or less effects can be added, to a maximum of 32Mbit of data.
The device is update through MQTT messaging file transfer, on which the updater is based on.

Edit the file list ***file_list.txt*** and add the path to the desired *.leds.esp32 file to be updated on the device.

Open a command line and run the updater as:

python file_sender.py -f file_list.txt -s <10 digit serial number> -d <MQTT broker ip>

For example:

python file_sender.py -f file_list.txt -s 0000000005 -d 192.168.1.6

All the paths in the list file must be sequential. The updater will stop as soon as one of the lines in the list does not contain a valid path or file.

MQTT Services

The MQTT services provided allow to control the addressable LEDs colour.
The Inputs and Uart input, if enabled for user work, will automatically generate MQTT messages with the updated data.

Addressable LEDs – Solid Color

The PWMs/LEDs can be controlled using the MQTT Topic "**rb_wifi_mqtt_lights_***XXXXXXXXXX***_rgb_set**" where *XXXXXXXXXX* is the device Unit ID.

All the Addressable LEDs will have the same data set.
The Addressable LEDs cannot be individually addressed, these will all have the same colour set by the RGB command. The data sent to the Addressable LEDs will follow the RGB format set at setup.

The message supported data formats are:

"#RRGGBB" or "r g b" where RR, GG and BB are hexadecimal format of colour component Red, Green and Blue. r, g and b are n either decimal or hexadecimal format.

For example:

Set pure white: "#FFFFFF" or "255 255 255" or "255 255 0xff" or "255 0xFF 0xff"

Set pure Blue: "#0000FF" or "0 0 255" or "0 0 0xff" or "0 0x00 0xff"

Addressable LEDs - Effects

The effects to run on the LEDs can be controlled using the MQTT Topic "**rb_wifi_mqtt_lights_***XXXXXXXXXX***_effect_selection**" where *XXXXXXXXXX* is the device Unit ID.

The Addressable LEDs will be run with the selected effect. If the effect has a different length of strip, the led driver is re-initialised accorddingly.

The message supported data formats are:

“1” , “2”... “n”

For example:

Set effect 3: "3"

If the selected effect does not exist, the unit will revert to the last one of the list.

Reprogramming the ESP32

This procedure will erase the device main application, and shall be **done at user own risk**.

1) Power the device.
2) Press and hold both Reset and IO0
3) Release Reset
4) Release IO0
5) Use Arduino IDE, espressif tool or simple programming too to reprogram the device using ISP through the uart.

References

https://github.com/RagingBits/MQTT_GenDevice

https://randomnerdtutorials.com/esp32-pinout-reference-gpios

https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf

https://www.espressif.com/en/products/modules/esp32

https://github.com/RagingBits/ESP32_Wroom_Tools