ESP32-M1 Reach Out Development Board Technical Manual

Perth, Western Australia

Document history

Version	Reason for Revision	Date
1.0	New document	30-3-2021

Contents

1 Introduction 1
1.1 Objective 1
1.2 Background 1
2 ESP32-M1 Technical Details 2
2.1 Power Management 2
2.2 Microcontroller 3
2.3 RF Section 5
2.4 Serial Communication and FW Flashing 6
2.5 Accessible Pins 7
3 PCB 9
3.1 PCB Top Section 9
3.2 Layers 10
4 Code Explanation 11
4.1 Antenna TX/RX Selection 11
4.2 Wi-Fi and Bluetooth Path Selection 11
4.3 Wi-Fi RFFE Bypass Selection 12
5 Reference and Datasheet 13

Acronyms

AP – Access Point

BPF – Band Pass Filter

BT – Bluetooth

ESD - Electrostatic Discharge

GPIO – General Purpose Input Output

IC – Integrated Circuit

LNA – Low Noise Amplifier

LDO – Low Dropout Regulator

PA – Power Amplifier

PCB – Printed Circuit Board

RF – Radio Frequency

RFFE – Radio Frequency Front End

RX - Receiver

STA – Station

TX – Transmitter

ULDO - Ultra Low Dropout Regulator

USB – Universal Serial Bus

1 Introduction

1.1 Objective

This document serves as a technical manual for the ESP32-M1 Reach Out development board. Discussion in this document is solely for ESP32-M1 board and the functionality, including schematic and PCB, Wi-Fi and BT path selection code, ESP32 antenna switch operational code and RFFE bypass operation, serial communication, and board firmware flashing.

1.2 Background

The ESP32-M1 board is designed to help makers, engineers, researchers, and hobbyist to develop robust communication systems using Wi-Fi and Bluetooth protocols for a distance. The hardware design and development can be tedious, with lots of design process, many design reviews, time, and knowledge consuming development works, but final product built need to target for simplicity and usefulness rather complex and difficult to use. The reason for this project is, the system made simple, seamless communication, reuse most of the existing code and reach further.

The effort started to answer a simple question but an effective one, can the residential Wi-Fi access point be a replacement for mobile data. Technically a complex question in every angle in a communication system. Well, the answer is, it is possible not largely but locally for small coverage, and the ESP32-M1 is the first step to achieve it, there is more work need to be done for this question but if you were able to transmit and receive within 1KM range, theoretically 10 units of ESP32-M1 can cover up to 10KM radius. It is not a perfect system but useful in lots of ways.

2 ESP32-M1 Technical Details

The ESP32-M1 board has three major sections, power management, microcontroller, and RF section as shown in Figure 2.1. This chapter summarises these sections and the operation of the board.

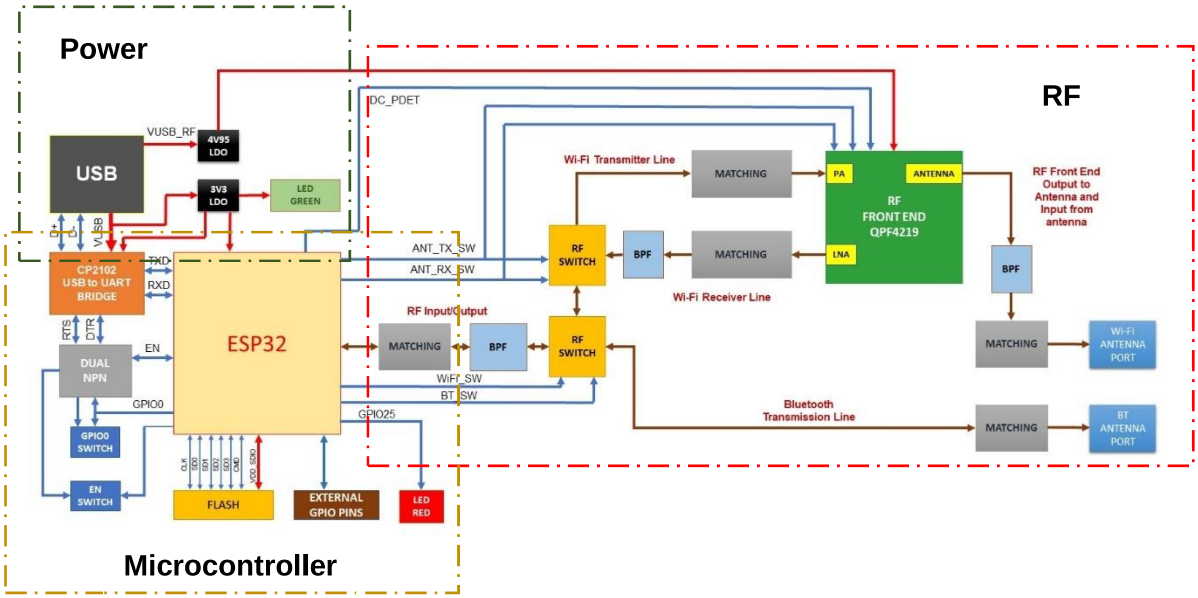
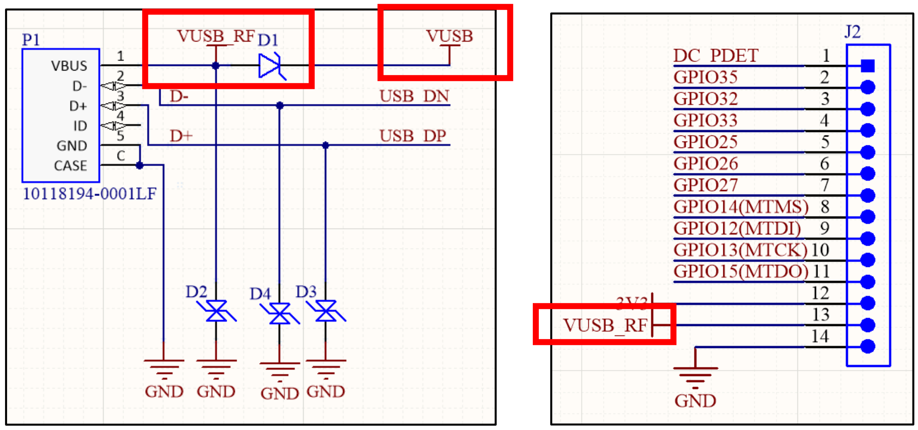


Fig 2.1: ESP32-M1 block diagram showing three major sections, power, microcontroller, and RF.

2.1 Power Management

The ESP32-M1 needs 5V to operate. The board can be powered from Micro-USB (al the lines connected to ESD diodes) or a 5V power source using external pin *5VUSB (on PCB)* connected to net VUSB_RF. Figure 2.2 illustrate the nets.



a. b.

Fig 2.2: Red box showing a. VUSB_RF and VUSB from USB connector P1, b. VUSB_RF pin for 5V on PCB.

The 5V supply on net VUSB_RF is supplied directly to U4 via Schottky diode D1, the supply net is VUSB. The U4 is a 3V3 LDO to supply U1 (ESP32 microcontroller), U2 (CP2102 USB Bridge, also supplied by VUSB), and the voltage is accessible from *3V3* pin *(on PCB)*. LED D5 (Green) is connected to 3V3 to indicate the components around the microcontroller section is powered. U3 (Flash) is powered by VDD_SDIO from the ESP32 microcontroller. The 5V from the USB port is accessible from pin *5VUSB (on PCB)*.

The 5V VUSB_RF is supplied to the U5 an ULDO. The output of the U5 is 4.9V approximately and power the RFFE.

2.2 Microcontroller

The schematic of the ESP32 microcontroller section is shown in Figure 2.3. The ESP32 design is close to an Espressif Dev Kit C.

The GPIO21 and 22 is WiFi_SW and BT_SW net. These GPIOs net connected to the RF Switch U6, to select between the Wi-Fi or Bluetooth protocol. The high on WiFi_SW and low on BT_SW enables the Wi-Fi communications and fed the RF signal to RFFE from ESP32. With high on the BT_SW and low on the WiFi_SW enables the BT communications. The WiFi_SW and BT_SW have an independent LED connected to indicate active protocol. The D8 orange LED connected to WiFi_SW indicates Wi-Fi is active and the D10 blue LED connected to BT_SW indicates BT is active.

Note: When GPIO21 and 22 are high, this is not a valid condition, and when both GPIO21 and 22 are low, this condition switches OFF U6. The only time off condition is valid when Wi-Fi and BT are not used.

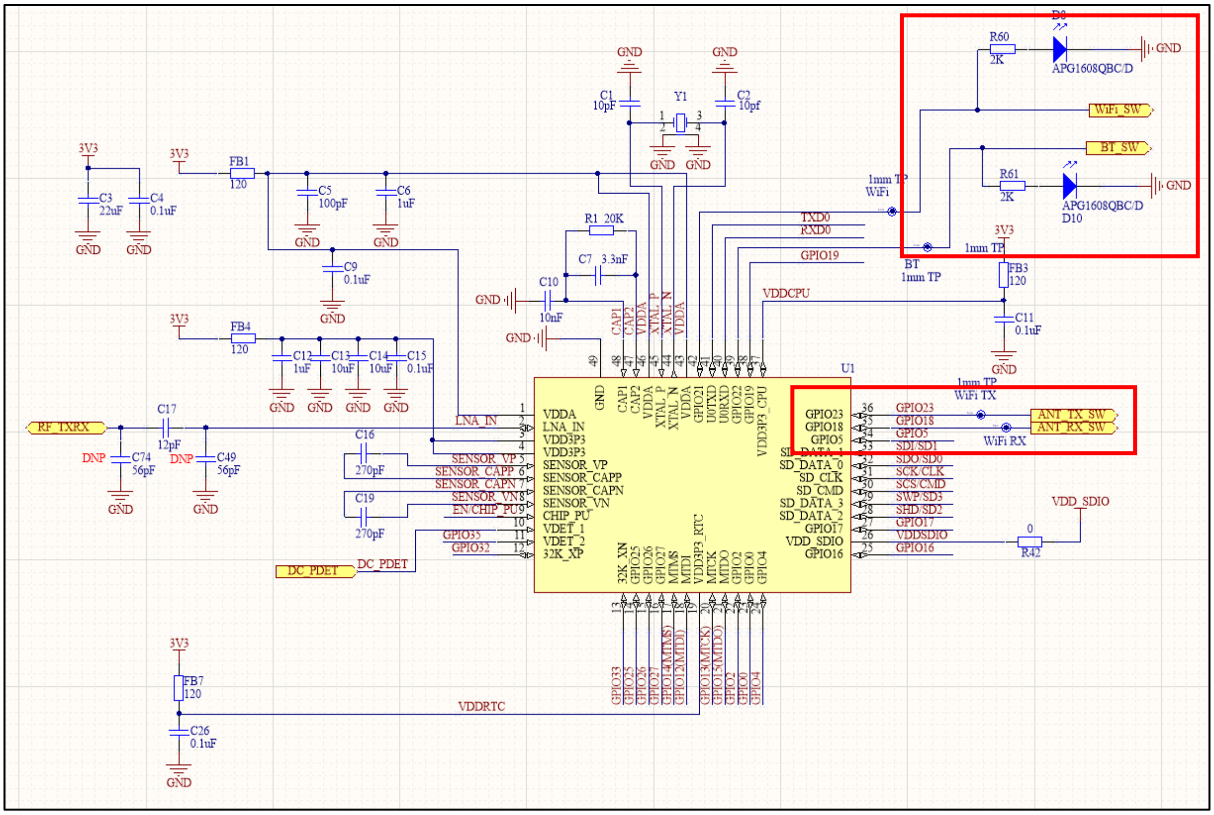


Fig 2.3: ESP32 circuit.

GPIO18 net ANT_RX_SW and GPIO23 net ANT_TX_SW are used as RX and TX switch. The code for this usage is explained in section 4.1. The RX and TX switch function switches between the transmitting and receiving events that direct the U10 to switch between transmitter and receiver line-up section and the RFFE to ON the PA / OFF the LNA during transmitting, and OFF the PA / ON the LNA during receiving. Below Table 2.1 summarise the operation.

Table 2.1: Wi-Fi and BT Selection, and Antenna Switch Summary.

Wi-Fi and BT Event		
	Wi-Fi ON / BT OFF	Wi-Fi OFF / BT ON
GPIO21	High	Low
GPIO22	Low	High
Wi-Fi Transmitter and Receiver Event		
	Transmitter OFF / Receiver ON	Transmitter ON / Receiver OFF
GPIO18	High	Low
GPIO23	Low	High
RFFE OFF		
GPIO18	Low	
GPIO23	Low	
Wi-Fi RFFE Bypass Mode		
GPIO18	Low	
GPIO23	Low	
GPIO21	Low	
GPIO22	High	

2.3 RF Section

The RF section is shown in Figure 2.4. The RF output of the ESP32 is fed to FIL5 BPF. The input signal at U6 either take the BT path or Wi-Fi path depends on the chosen protocol discussed in Table 2.1. If the BT path is chosen, the signal is transmitted out using BT_PORT antenna, and this applies to receiving BT. For the Wi-Fi path, the line-up is now separated into TX and RX section, the RFFE has different TX and RX pins but this not the case for ESP32 both sharing a single pin. To use separate pins for RFFE from a single RF pin from ESP32, the signal from U6 is fed to U10. Table 2.1 shows the switching events set in ESP32 discussed in the previous subtopic.

The TX line comprises of 18dB attenuator, input matching to PA and RFFE. The output of RFFE into FIL6 and to WIFI_PORT. For the RX line, the incoming signal shares the same antenna and FIL6 to RFFE. The RFFE receiver output can be used without the LNA from output pin RX_OUT, in this design the LNA is used, RX_OUT is fed back to RFFE LNA_IN. The output of LNA_OUT into matching to FIL4, now the RF Switch U10 RX_LINE is high, allow the signal to ESP32 via U6 and FIL5.

The RFFE can be switch off by pulling GPIO18 and 23 low. In this scenario, the Wi-Fi signal significantly drops in range. The Wi-Fi protocol can still be used when RFFE is OFF, this scenario called Wi-Fi Front End Bypass mode is recommended for short-range Wi-Fi communication and lower current, please see section 4.3 for more details.

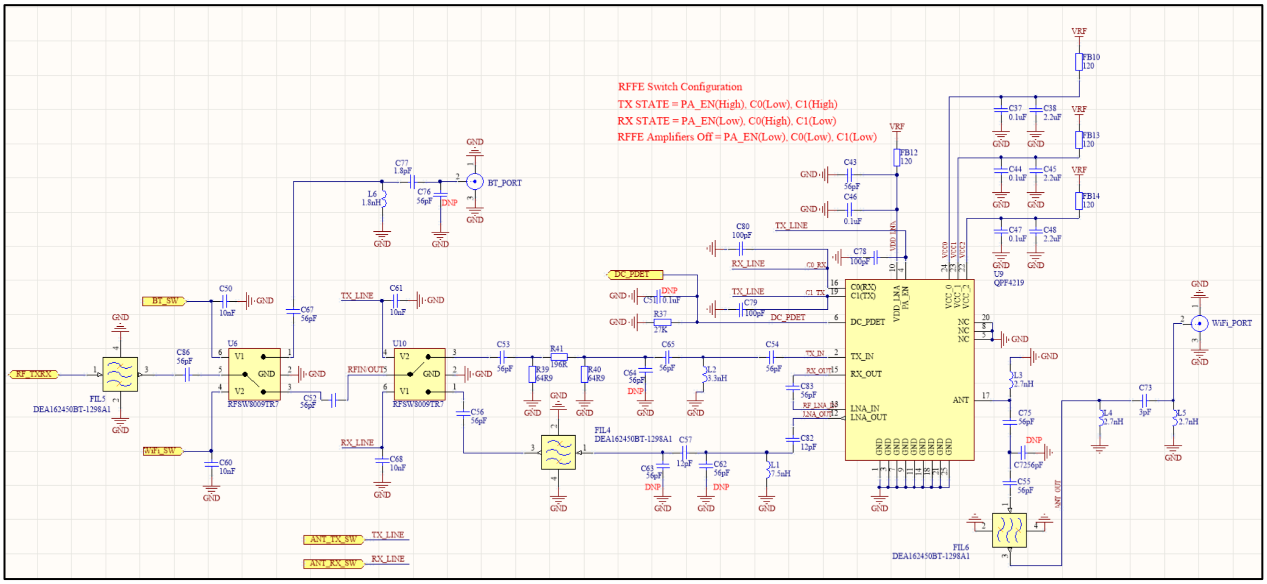


Fig 2.4: ESP32-M1 RF circuit.

2.4 Serial Communication and FW Flashing

The CP2102 is the USB to serial bridge chip used in the design as the interface between the PC serial port and ESP32. No configuration needed to flash the code to the board. Plug in Micro-USB and flash the binary.

The EN switch is connected to ESP32 pin 9. Pin 9 is CHIP_PU, high signal on this pin turns ON ESP32 and the low signal OFF the chip. This pin pulled high using 3V3, and accessible at board level from pin *EN.* The EN switch resets ESP32 when pressed by pulling low to the ground.

The IO0 switch is connected to ESP32 pin 23 and accessible at board level from pin *IO0.*

The EN and IO0 line is part of the flashing configuration, do not press these switches when flashing. The IO0 switch can be used as a functional switch.

2.5 Accessible Pins

ESP32-M1 has total of 17 dedicated GPIOs accessible from external connectors/pins (16 Input and Output pins and 1 Input Only pins). There are 3V3, 4V9 and 5V USB output voltages and three Ground from these connectors. The connector also provides DC_PDET, TXD0, RXD0 and RESET pins. Please see Figure 2.5 and 2.6 for pinout and Table 2.2 for connector pin detail description and corresponding to ESP32-M1.

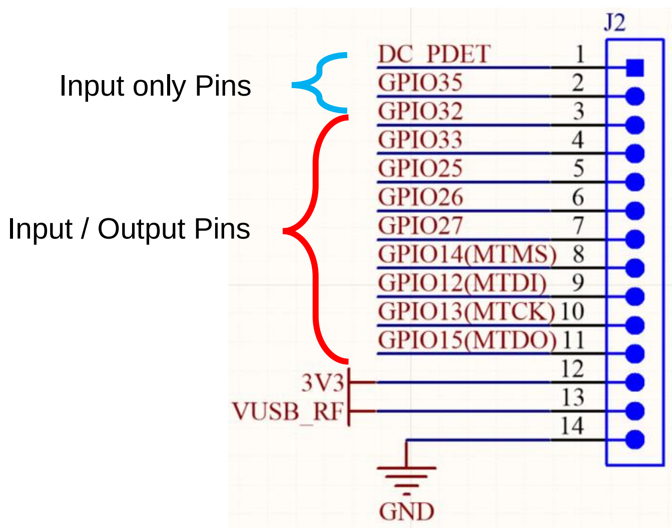


Fig 2.5: Schematic pinout for left side.

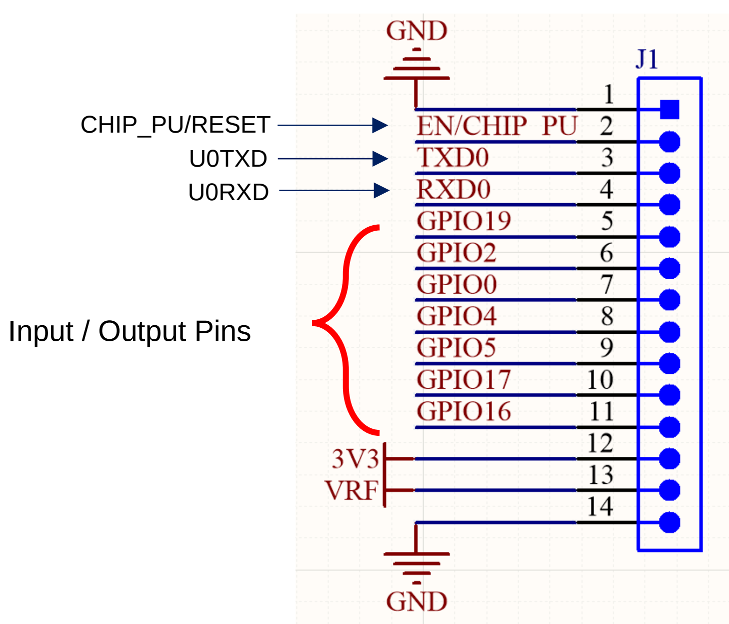


Fig 2.6: Schematic pinout for right side.

Table 2.2: ESP32-M1 PinOut, type and function.

PIN Description on Schematic	PIN Description on PCB	Type	ESP32 Pin Number	Function
DC_PDET	DC PDET	Input	10	RFFE PA Voltage output based on RF power. VDET_1
GPIO35	I_35	Input	11	VDET_2
GPIO32	IO32	Input and Output	12	
GPIO33	IO33	Input and Output	13	
GPIO25	IO25	Input and Output	14	Connected to D9
GPIO26	IO26	Input and Output	15	
GPIO27	IO27	Input and Output	16	
GPIO14	IO14	Input and Output	17	MTMS
GPIO12	IO12	Input and Output	18	MTDI
GPIO13	IO13	Input and Output	20	MTCK
GPIO15	IO15	Input and Output	21	MTDO
3V3	3V3	Output Voltage	-	3.3V output voltage from U4
VUSB_RF	5V USB	Output and Input Voltage	-	+5V USB
GND	GND	Ground	GND	Ground contact to main ground plane
GND	GND	Ground	GND	Ground contact to main ground plane
EN/CHIP_PU	EN/CHIP_PU	Input	9	High enables the chip and Low power down the chip
TXD0	TX0	Input and Output	41	UART0 TX pin connected to USB
RXD0	RX0	Input and Output	40	UART0 RX pin connected to USB
GPIO19	IO19	Input and Output	38	
GPIO2	IO2	Input and Output	22	
GPIO0	IO0	Input and Output	23	Connected to switch IO0
GPIO4	IO4	Input and Output	24	
GPIO5	IO5	Input and Output	34	
GPIO17	IO17	Input and Output	27	
GPIO16	IO16	Input and Output	25	
3V3	3V3	Output Voltage	-	3.3V output voltage from U4
VRF	VRF	Output Voltage	-	4.9V output voltage from U5
GND	GND	Ground	GND	Ground contact to main ground plane

3 PCB

3.1 PCB Top Section

The PCB for ESP32-M1 is kept moderately small with all the external connectors is effortlessly accessible, the USB connector, GPIOs and supply pins, and RF connectors. The board test points for Wi-Fi TX and RX switching connected to GPIO23 and 18, these test points show the high-speed switching from ESP32 between the transmit and receive events.

Test points for Wi-Fi and BT connected to GPIO21 and 22, these test points indicate the high or low of the protocols that used on the board, for example, if Wi-Fi is the current protocols, the Wi-Fi test point will be high, and BT test point will be low, the code related to enable Wi-Fi high and BT low is shown in section 4.2. The EN and GPIO0 switch is placed at bottom of the board. Figure 3.1 shows the PCB image of ESP32-M1.

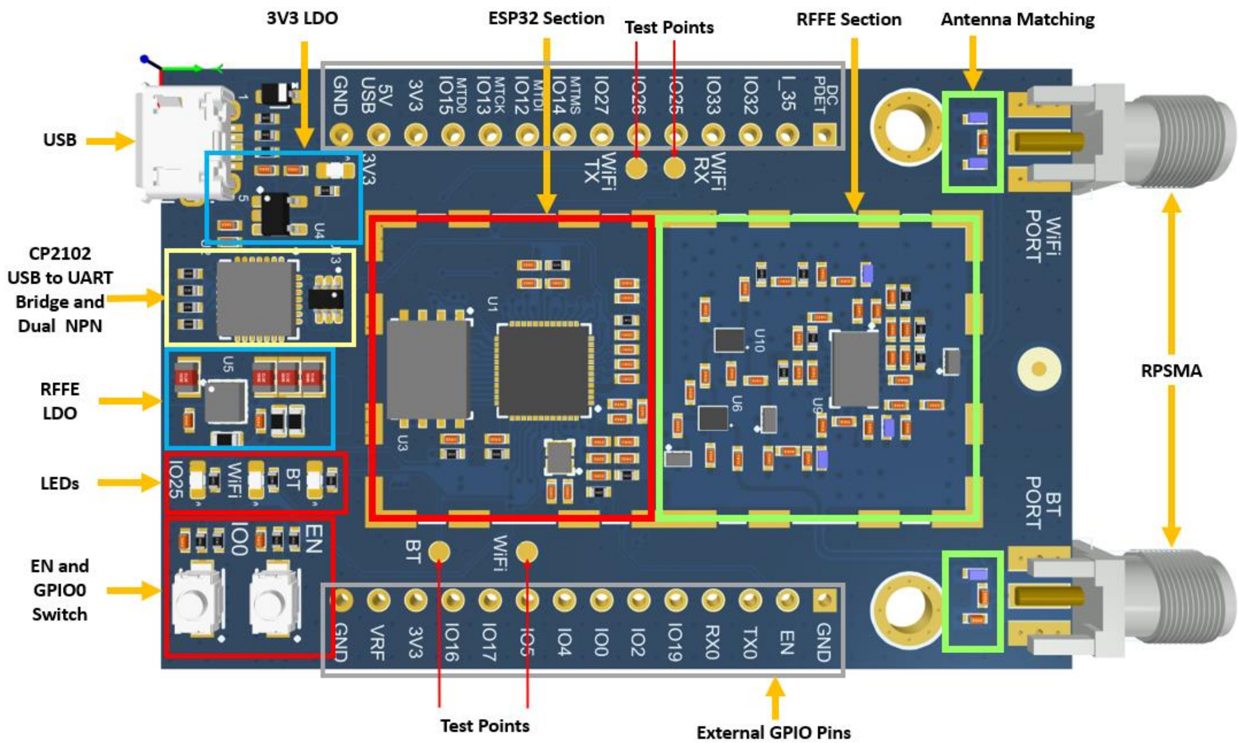


Fig 3.1: PCB component placement and external connectors on top layer, and at the bottom layer is outline for heat shield.

3.2 Layers

The ESP32-M1 PCB (not including RPSMA) is a 62mm x 40mm 4 layers board. The dielectric constant of the board is 4.2. The top layer of the board is the signal plane and component placement, this layer contains all the RF traces, signal traces from ESP32, and serial communications. The second layer is the ground plane and reference ground for RF. The third layer is the supply layer, and the bottom layer is a mix of ground plane, supply trace and signal traces. The image of the layers is shown in below Figure 3.2. a to d.

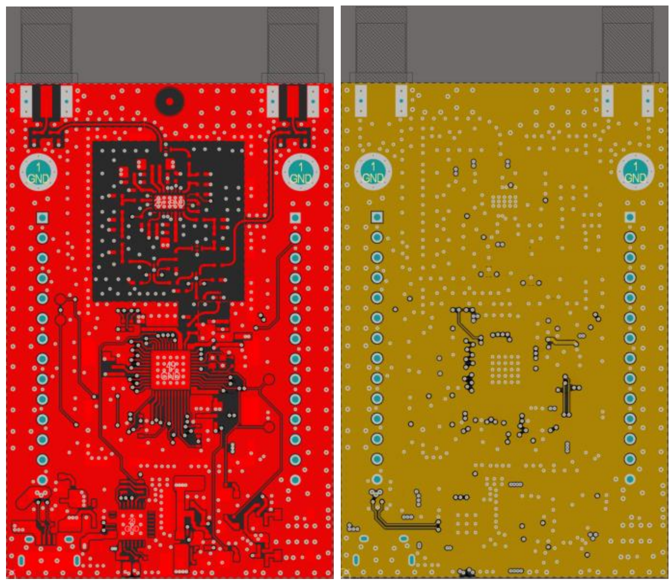


Fig 3.2 a. Top layer in red and b. Second layer in brown

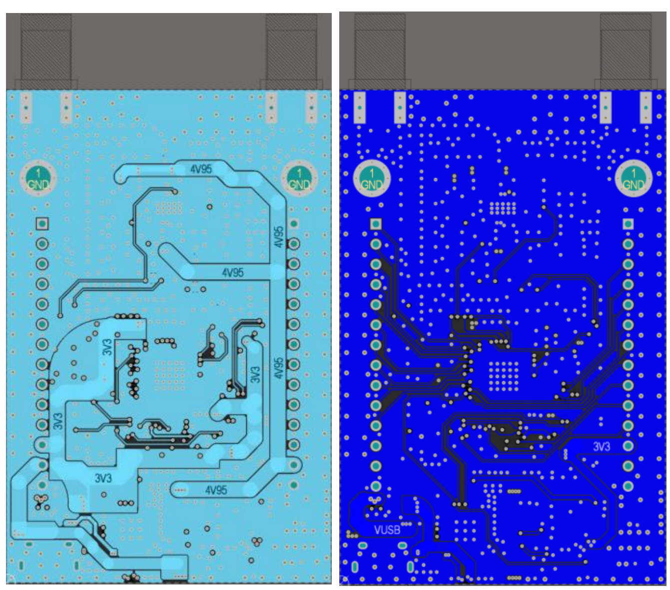


Fig 3.2 c. Third layer in light blue and d. Botom layer in dark blue.

4 Code Explanation

This section discusses some of the code used to configure the ESP32-M1 board for the above Table 2.1 setting. The codes below are written in C language for ESP-IDF.

4.1 Antenna TX/RX Selection

The code below configures GPIO18 and 23 to switch between TX and RX events. The header file associate with this code is *exp_wifi.h.*

For further information on this operation please refer to

https://docs.espressif.com/projects/esp-idf/en/latest/esp32/apiguides/wifi.html?highlight=antenna#wi-fi-multiple-antennas

```c
void gpio_out_tx_status()
{
	wifi_ant_gpio_config_t config_gpio;
	config_gpio.gpio_cfg[0].gpio_select = 1;
	config_gpio.gpio_cfg[0].gpio_num = 23;
	config_gpio.gpio_cfg[1].gpio_select = 1;
	config_gpio.gpio_cfg[1].gpio_num = 18;

	wifi_ant_config_t m_ant;
	m_ant.rx_ant_mode = WIFI_ANT_MODE_ANT1;
	m_ant.rx_ant_default = WIFI_ANT_MODE_ANT1;
	m_ant.tx_ant_mode = WIFI_ANT_MODE_ANT0;
	m_ant.enabled_ant0 = 1;
	m_ant.enabled_ant1 = 2;

	esp_wifi_set_ant_gpio(&config_gpio);
	esp_wifi_set_ant(&m_ant);
}
```

4.2 Wi-Fi and Bluetooth Path Selection

The path selection for Wi-Fi and BT essentially turning ON and OFF GPIO21 and 22. The below code shows the Wi-Fi is enabled the GPIO21 is set to **1** and BT is disabled by setting GPIO22 to **0**. And vice versa enable BT path and off Wi-Fi.

```
void enable_wifi(void)
{
    //Enable WiFi
      gpio_pad_select_gpio(21);
      gpio_set_direction(21, GPIO_MODE_OUTPUT);
      gpio_set_level(21, 1);

    //Disable BT
      gpio_pad_select_gpio(22);
      gpio_set_direction(22, GPIO_MODE_OUTPUT);
      gpio_set_level(22, 0);
}
```

4.3 Wi-Fi RFFE Bypass Selection

The Wi-Fi protocol can still be used when RFFE is OFF. To configure this setup GPIO22 is pulled high, and the rest are pulled low (GPIO23, 18 and 21), same as configuring Bluetooth instead the Wi-Fi firmware is running now, and the Blue LED will ON, this scenario called Wi-Fi Front End Bypass mode is recommended for short-range Wi-Fi communication and lower current. This setup is shown in Table 2.1.

```
void rffe_off(void)
{
    //Disable RFFE TX_LINE
      gpio_pad_select_gpio(23);
      gpio_set_direction(23, GPIO_MODE_OUTPUT);
      gpio_set_level(23, 0);

    // Disable RFFE RX_LINE
      gpio_pad_select_gpio(18);
      gpio_set_direction(18, GPIO_MODE_OUTPUT);
      gpio_set_level(18, 0);


    //Disable WiFi Path
      gpio_pad_select_gpio(21);
      gpio_set_direction(21, GPIO_MODE_OUTPUT);
      gpio_set_level(21, 0);

    //Enable BT Path
      gpio_pad_select_gpio(22);
      gpio_set_direction(22, GPIO_MODE_OUTPUT);
      gpio_set_level(22, 1);
}
```

5 Reference and Datasheet

This section listed reference and link for ESP32-M1.

1. ESP32 IC Documents – Design and datasheet

   https://docs.espressif.com/projects/esp-idf/en/stable/esp32/hw-reference/index.html

2. ESP-IDF for Wi-Fi – Code example

   https://github.com/espressif/esp-idf/tree/master/examples/wifi

3. Bison Science Documents and Datasheets – Schematic and PCB

   https://github.com/bisonscience/ESP32-M1-Reach-Out