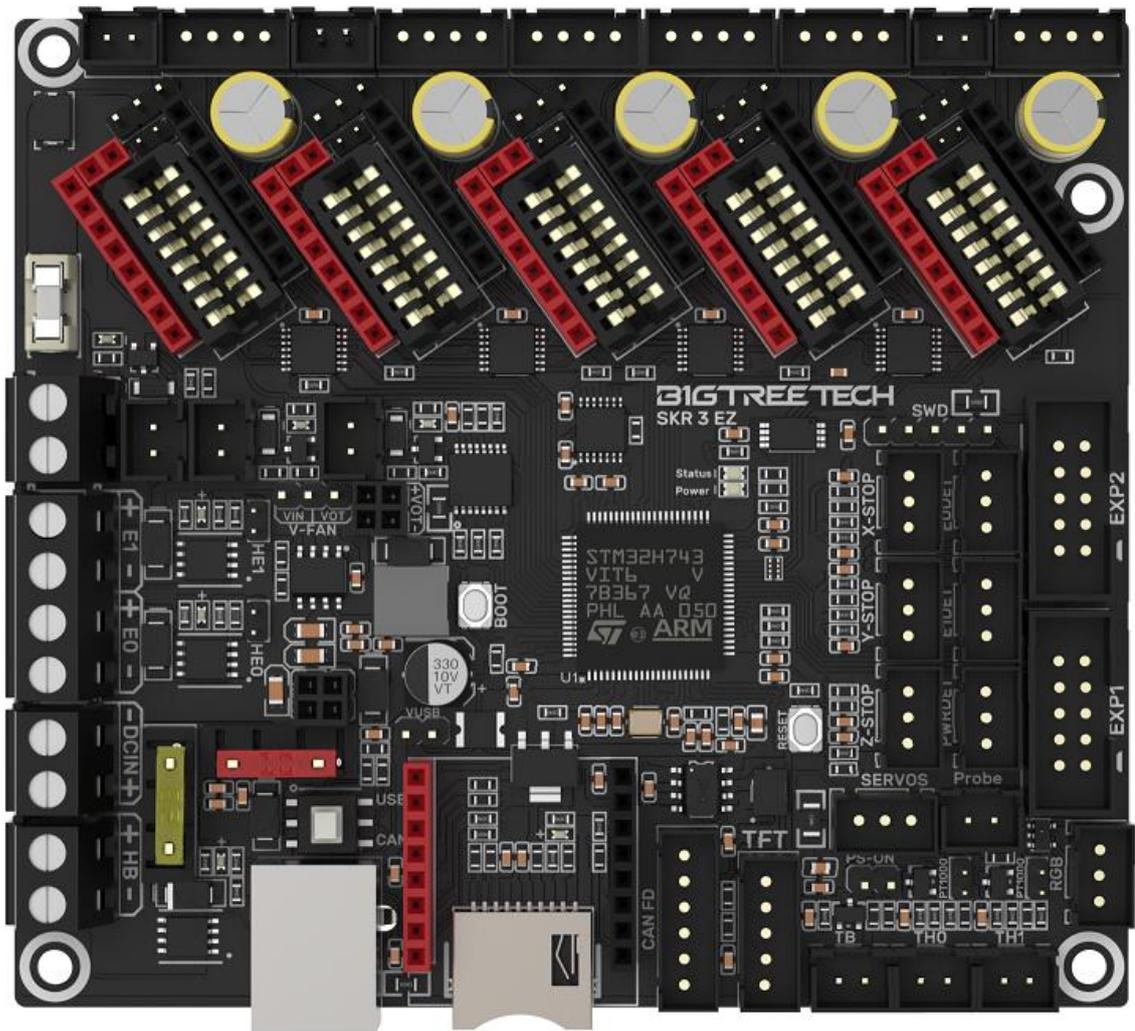


BIGTREE TECH

# BIGTREE TECH

## SKR 3 EZ

### User Manual



## Content

Content.....	2
Revised History .....	5
<b>1. Product Introduction .....</b>	<b>6</b>
<b>1.1 Product Features.....</b>	<b>6</b>
<b>1.2 Product Parameters .....</b>	<b>7</b>
<b>1.3 Firmware Support.....</b>	<b>8</b>
<b>1.4 Product Size.....</b>	<b>9</b>
<b>2. Peripheral Interface .....</b>	<b>10</b>
<b>2.1 Interface Diagram .....</b>	<b>10</b>
<b>2.2 Pins Description .....</b>	<b>11</b>
<b>3. Interface Introduction.....</b>	<b>12</b>
<b>3.1 USB Powered.....</b>	<b>12</b>
<b>3.2 Motor Voltage Selection .....</b>	<b>13</b>
<b>3.2.1 Motherboard Power Voltage for Motor Voltage Selection.....</b>	<b>13</b>
<b>3.2.2 Motor Supply Voltage for Motor Voltage Selection.....</b>	<b>13</b>
<b>3.3 Step Motor Drivers .....</b>	<b>13</b>
<b>3.3.1 TMC-driven Mode.....</b>	<b>13</b>
<b>3.3.2 TMC/EZ-driven UART/SPI Mode.....</b>	<b>13</b>
<b>3.3.3 TMC-driven DIAG mode(Sensorless Homing) .....</b>	<b>14</b>
<b>3.4 USB and CAN Mode .....</b>	<b>14</b>
<b>3.5 Voltage selection for NC fans .....</b>	<b>15</b>
<b>3.6 100K NTC or PT1000 Setup .....</b>	<b>16</b>
<b>3.7 BLTouch Connection .....</b>	<b>16</b>
<b>3.8 Completed Shut-down Module(Relay V1.2) Connection .....</b>	<b>17</b>
<b>3.9 Resume Printing(UPS 24V V1.0) Connection .....</b>	<b>17</b>
<b>3.10 RGB Connection.....</b>	<b>18</b>
<b>3.11 Break Detection Connection .....</b>	<b>18</b>

3.12 Touch Screen Connection .....	19
3.13 Heater cartridge IO .....	19
<b>4. Marlin .....</b>	<b>20</b>
<b>4.1 Compiler Environment Installation .....</b>	<b>20</b>
<b>4.2 Download of Marlin Firmware .....</b>	<b>20</b>
<b>4.3 Firmware Configuration .....</b>	<b>20</b>
4.3.1 Open the Marlin Project .....	20
4.3.2 Compiler Environment Configuration .....	21
4.3.3 Motherboard type and Serial port number Configuration .....	21
4.3.4 Motor Driver Configuration .....	23
4.3.5 Sensorless Homing .....	24
4.3.6 100K NTC or PT1000 .....	25
4.3.7 BL Touch .....	25
4.3.8 Completed Shutdown Module (Relay V1.2) .....	28
4.3.9 Resume Printing .....	28
4.3.10 RGB Light .....	29
4.3.11 Filament Break Detection .....	30
4.3.12 Smart Filament Sensor(SFS V1.0) .....	32
4.3.13 ESP3D .....	33
<b>4.4 Compile the Firmware .....</b>	<b>34</b>
<b>5. Klipper .....</b>	<b>35</b>
<b>5.1 Preparation .....</b>	<b>35</b>
5.1.1 Download System Image .....	35
5.1.2 Download and Install Raspberry Pi Imager .....	36
<b>5.2 Burn Image .....</b>	<b>36</b>
<b>5.3 Set up WIFI .....</b>	<b>38</b>
<b>5.4 Connection of ssh software with Raspberry Pi .....</b>	<b>39</b>
<b>5.5 Compile the Firmware .....</b>	<b>40</b>

**5.6 Configure Klipper** ..... 42

**6. Firmware Update** ..... 43

**7. Cautions** ..... 43

**8. FAQ** ..... 44

## Revised History

Version	Note	Date
01.00	1 <sup>st</sup> Draft	2022/04/15
01.01	Add support for RRF	2022/05/21
01.02	Add description of IO of Heater cartridge	2022/08/27

## 1. Product Introduction

BIGTREETECH SKR 3 EZ V1.0 motherboard is a 32-bit 3D printer motherboard updated by the team of Shenzhen Biqu Technology Co., Ltd. for our EZ series drivers on the basis of SKR 3. It is compatible with both the EZ series driver and the series of TMC drivers.

### 1.1 Product Features

1. Using 32-bit ARM Cortex-M7 series STM32H743VI main control chip with a main frequency of 480MHz, the performance has greatly improved.
2. The power chip adopts TPS5450-5A, which supports DC12/24V power input. The output current of the chip is up to 5A, and the peak value can reach 6A, which perfectly supports the power supply of Raspberry Pi.
3. The motherboard reserves the BOOT button, users can update the motherboard boot program through DFU.
4. Increase the protection circuit of the thermistor part to avoid the burning of the main control chip due to leakage of the heated bed or heater cartridge.
5. The numerical control fan realizes 24V, 12V, 5V voltage selection through the external power supply module, eliminating the need for the operation of the customer's external transformer module, thereby reducing the probability of damage to the motherboard.
6. The thermistor can select the pull-up resistance value through the jumper, and support PT1000 in this way without the need for external modules, which is convenient for customers to use DIY.
7. Support all versions of our company's serial screen, SPI screen and LCD screen.
8. Upgrade the configuration firmware through an SD card, the operation is simple, convenient and efficient.
9. On-board DIAG function pins can be used by simply plugging and unplugging the jumper cap.
10. Supports functions such as resume printing, Filament Runout Detection, Completed Shutdown, BLTouch, RGB Lights, etc.
11. High-performance MOSFETs are used to reduce heat generation.
12. Adopt a replaceable fuse for easy replacement.
13. WIFI module (ESP-12S, ESP-07, ESP32) general interface.

14. The on-board non-self-elastic Micro SD card slot, and is SDIO working mode, which greatly speeds up the transfer rate.
15. Onboard EEPROM, which is convenient for users to save parameter information.
16. Two types of CAN interfaces are reserved, USB port and XH2.54 6Pin terminal interface. The USB port is used to select CAN and USB through the double-pole double-throw switch, which is convenient for customers to use other accessories of the CAN interface.
17. The temperature sensor interface adopts a high-precision pull-up resistor.
18. Two types of drive sockets are used, which are compatible with our EZ series drive modules and TMC series drive modules.
19. Each motor drive module can select the corresponding motor voltage through the jumper cap.
20. The motor power supply supports up to 48V, and for the larger voltage when using TMC5160 and EZ5160, an isolation chip is used to protect the mainboard from burning IO.

### 1.2 Product Parameters

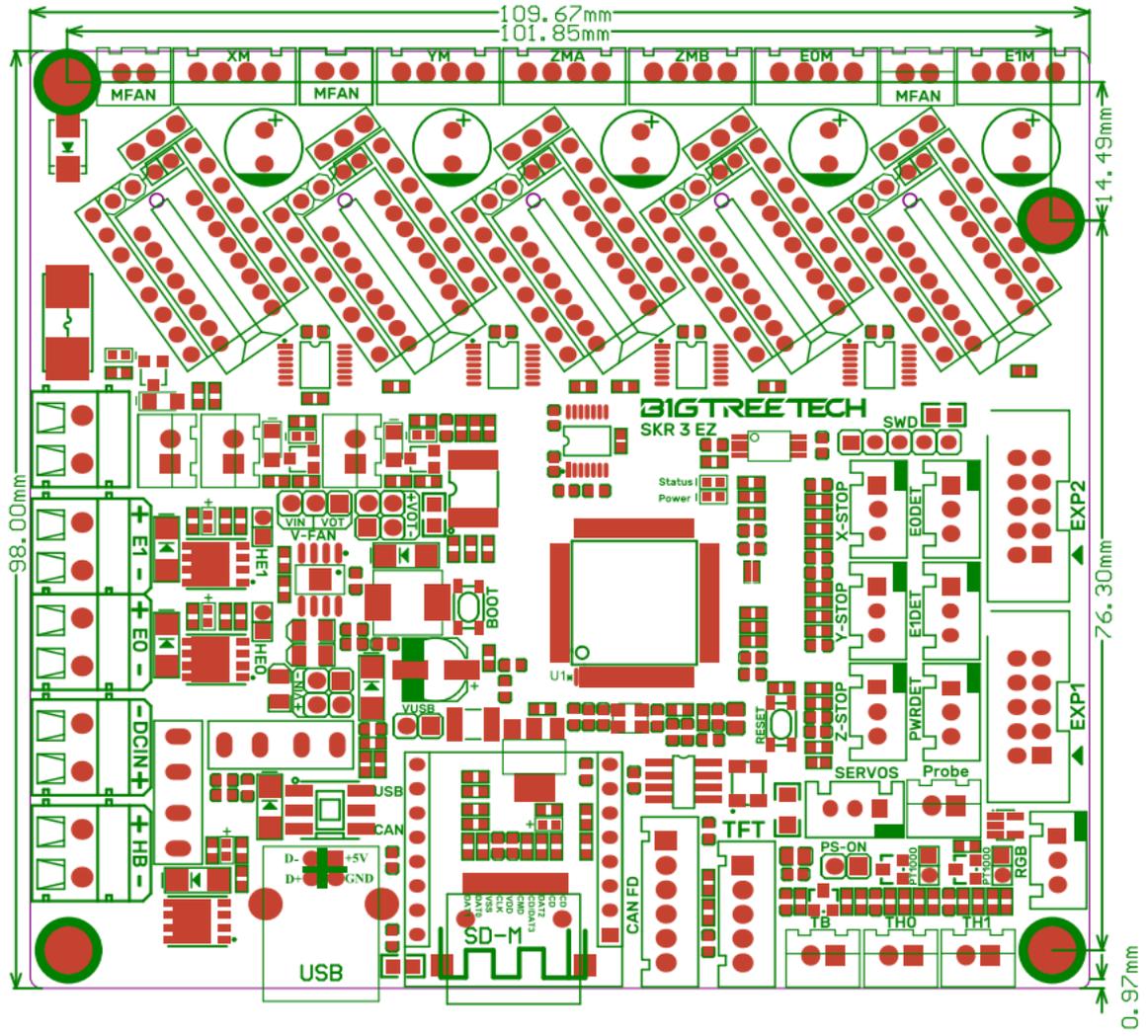
1. Product Size: 109.7 x 98mm, for details, please refer to **BIGTREETECH SKR 3 EZ V1.0-SIZE.pdf**
2. Installation Size: 102 x 76mm
3. Microprocessor: ARM Cortex-M7 STM32H743VI
4. EEPROM: 24C32 32Kbit
5. Input Voltage: DC12V-DC24V
6. Motor Voltage: DC12V-DC48V
7. Logic Voltage: DC 3.3V
8. Heating Interface: Heated bed (HB), Heater cartridge (E0, E1)
9. Maximum Output Current of Heated Bed Port: 10A, Peak Current 11A
10. Maximum Output Current of Heater cartridge Port: 5.5A, Peak Current 6A
11. Fan Interface: Three CNC fans, three normally open fans, the voltage of the CNC fans is optional.
12. Maximum Output Current of Fan Interface: 1A, Peak Current 1.5A

13. The Total Current of Heater cartridge + Driver + Fan: less than 10A
14. WIFI Interface: ESP-12S, ESP-07S, ESP32
15. Expansion Interface: BLTouch (Servos, Probe), PS-ON, PWR-DET, Fil-DET, RGB, CAN FD
16. Motor Drive: Support EZ5160, EZ2209, EZ2208, EZ2225, EZ2226, EZ2130, EZ6609, TMC5160, TMC2209, TMC2225, TMC2226, TMC2208, TMC2130, etc.
17. Driver Working Mode Support: SPI, UART, STEP/DIR
18. Motor Drive Interface: X, Y, Z (dual Z-axis), E0, E1 Five Channels
19. Temperature Sensor Interface: 1 100K NTC, 2 100K NTC and PT1000 optional
20. Display: Serial Touch Screen, SPI Touch Screen, LCD
21. PC Communication Interface: Square USB A, easy to plug and unplug.
22. Supported File Format: G-code
23. Support Machine Structure: Cartesian, Delta, Kossel, Ultimaker, CoreXY
24. Recommended software: Cura, Simplify3D, Pronterface, Repetier-host, Makerware.

### **1.3 Firmware Support**

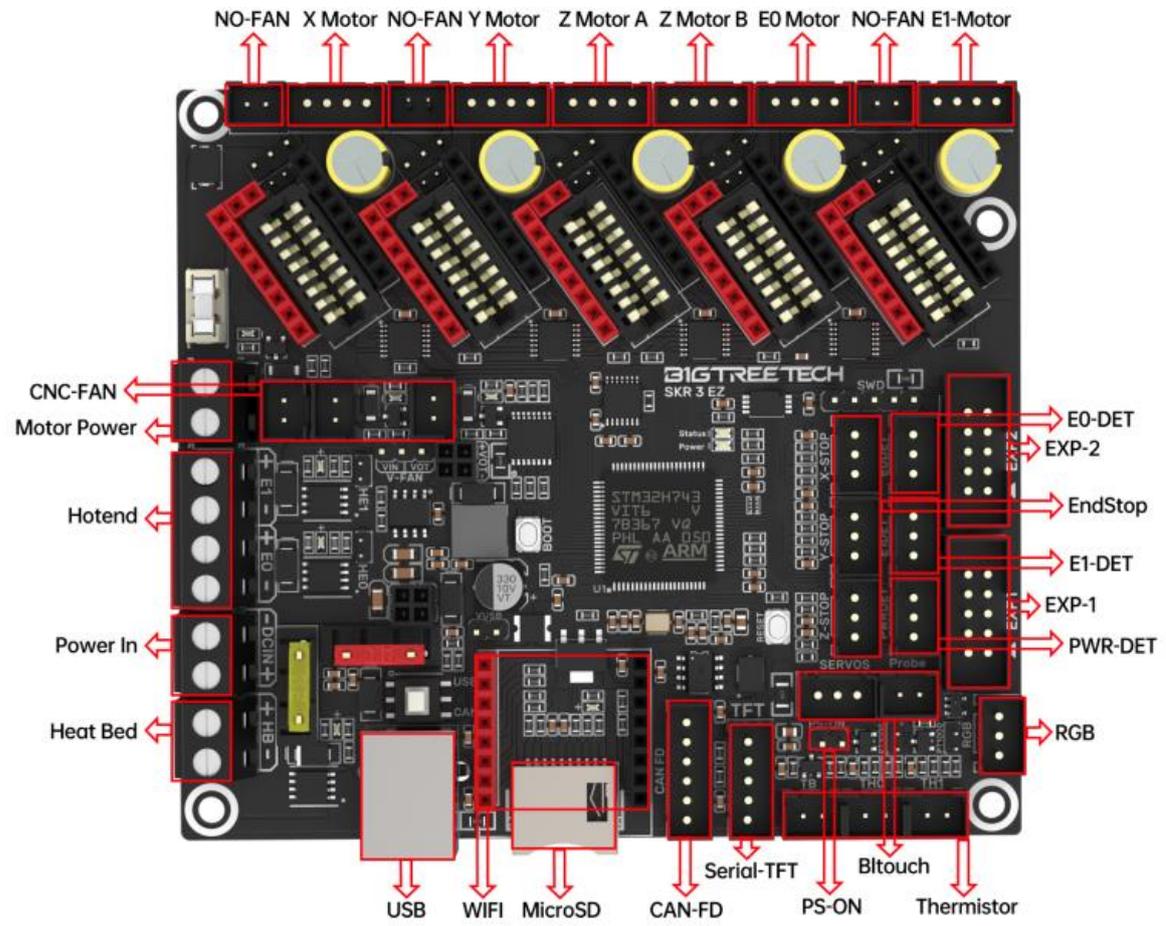
Supported Firmware: Marlin, Klipper, RRF.

### 1.4 Product Size

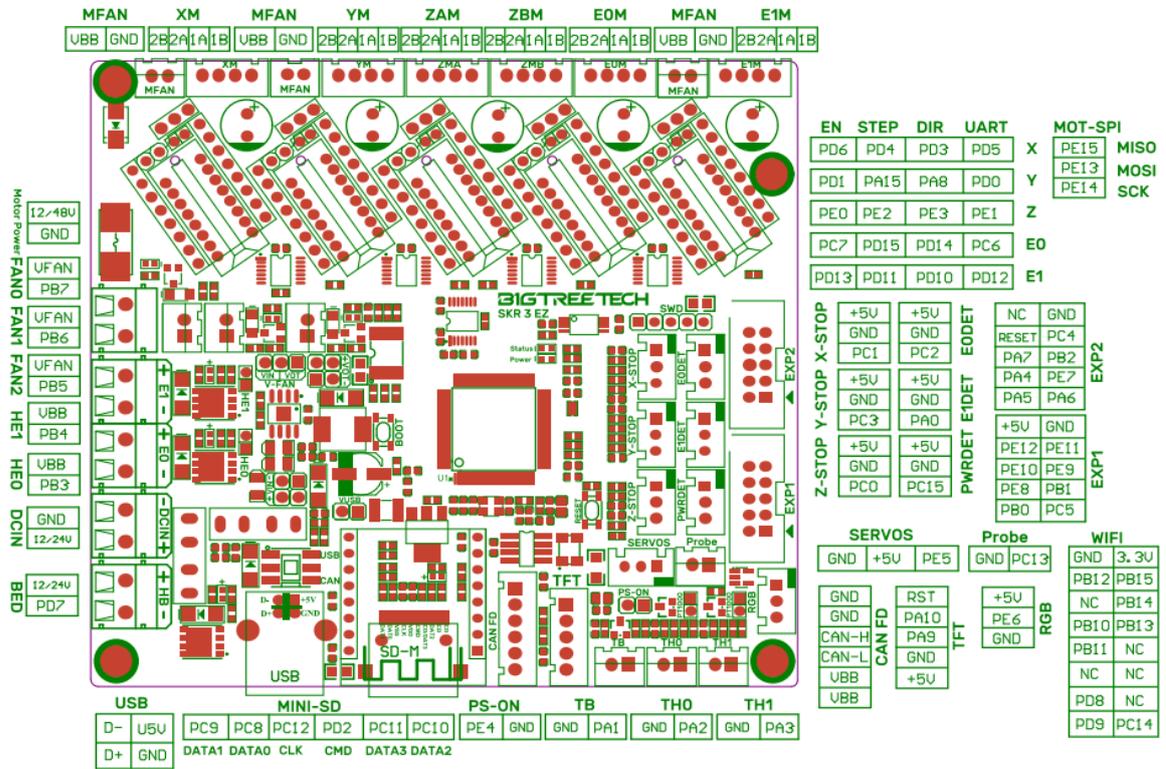


## 2. Peripheral Interface

### 2.1 Interface Diagram



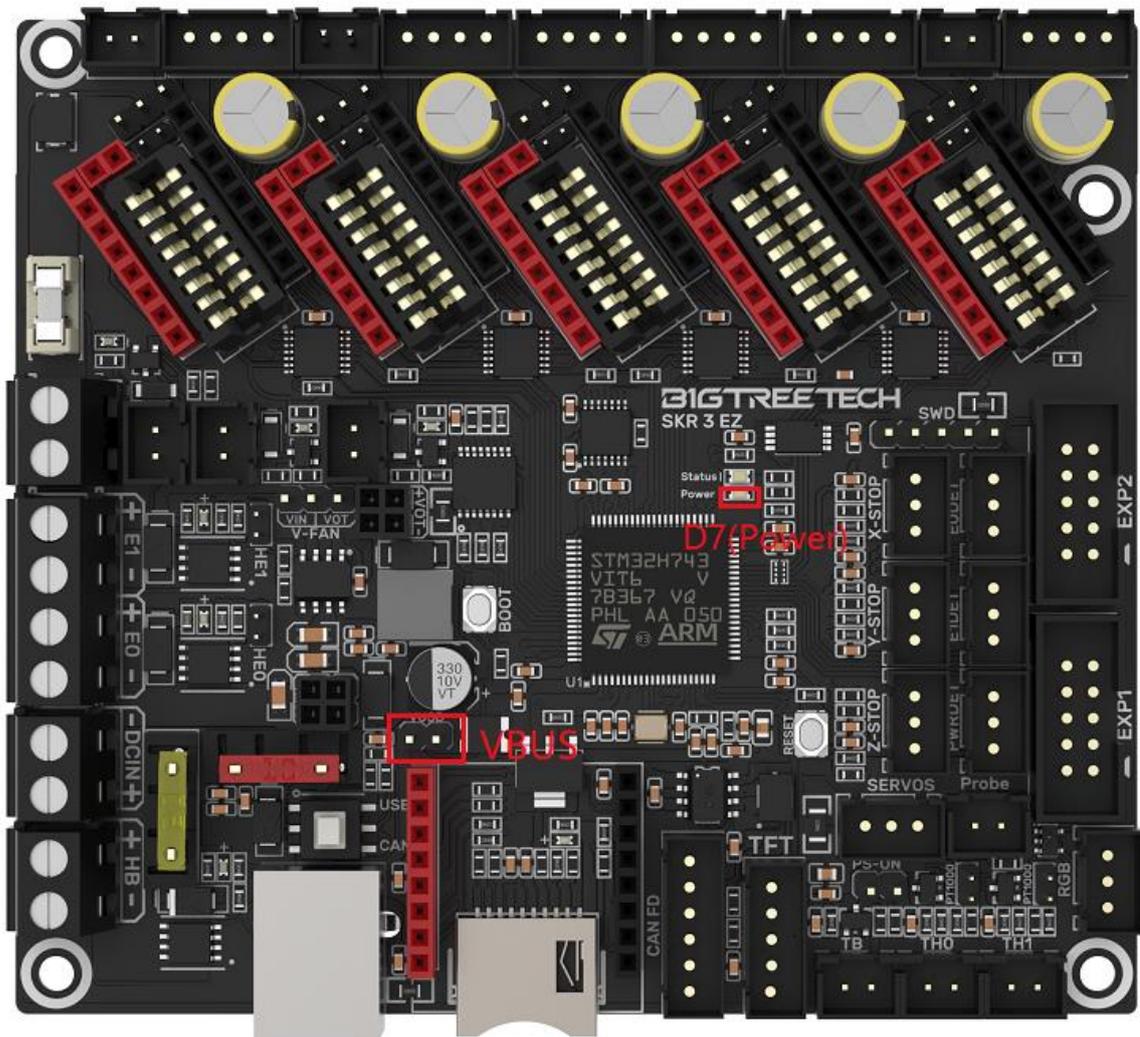
## 2.2 Pins Description



### 3. Interface Introduction

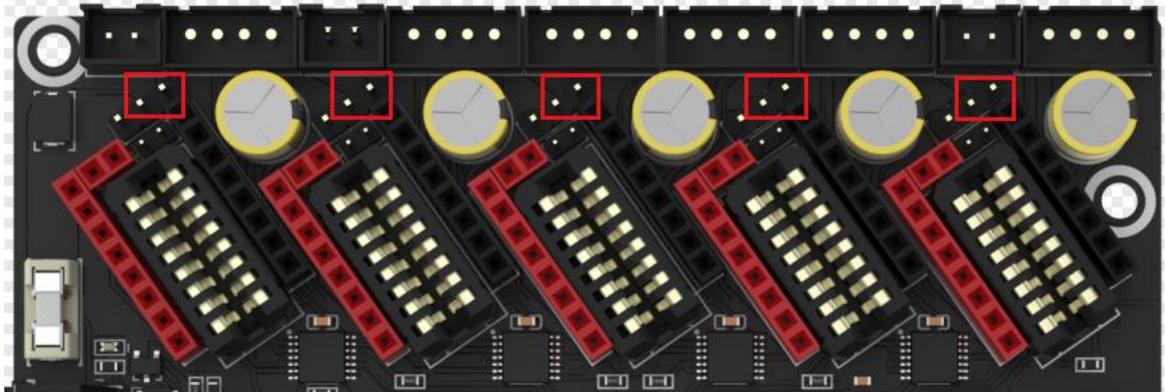
#### 3.1 USB Powered

After the SKR 3 EZ V1.0 motherboard is powered on, the red light of D7 (Power) in the upper right corner of the MCU will light up, indicating that the power supply is normal. The VUSB in the middle of the board is the power selection terminal. Only when using USB to supply power to the motherboard or need to supply power through USB, you need to use the jumper to make the VUSB short circuit.

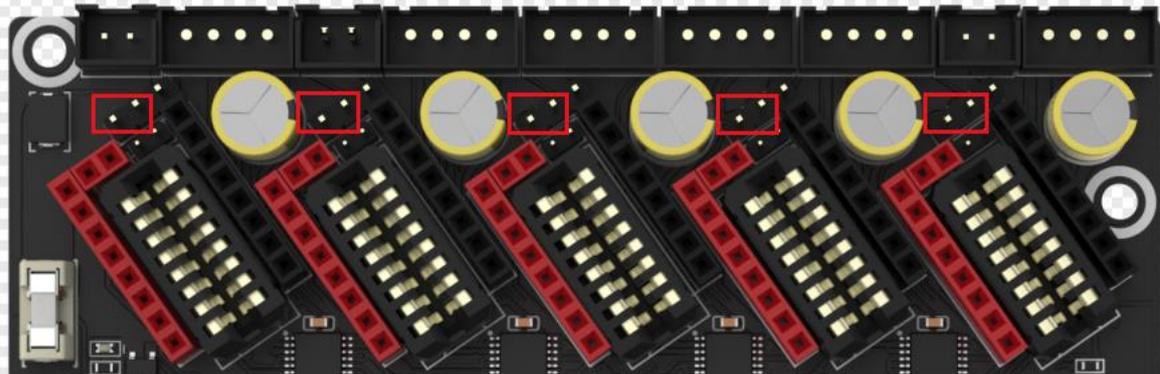


## 3.2 Motor Voltage Selection

### 3.2.1 Motherboard Power Voltage for Motor Voltage Selection



### 3.2.2 Motor Supply Voltage for Motor Voltage Selection



## 3.3 Step Motor Drivers

### 3.3.1 TMC-driven Mode

The number of subdivisions needs to be set high or low by firmware to the corresponding subdivision configuration pins.

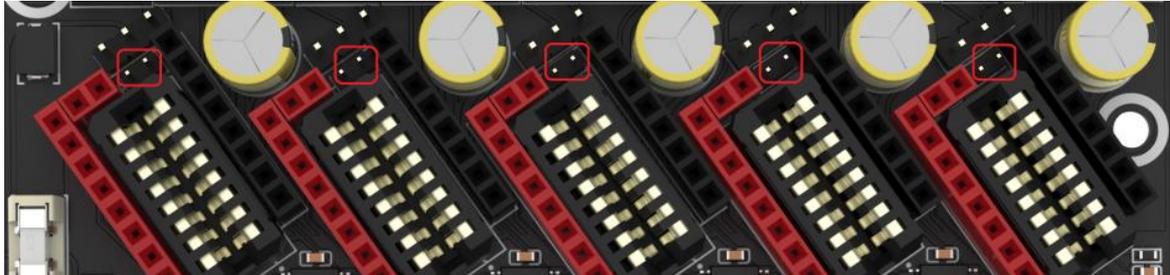
### 3.3.2 TMC/EZ-driven UART/SPI Mode

TMC series drivers do not support the use of both UART and SPI drivers at the same time, for example: X, Y-axis use TMC/EZ2209 (UART), Z, E0 axis use TMC/EZ5160 (SPI).

The EZ series drivers support the simultaneous use of both UART and SPI drivers.

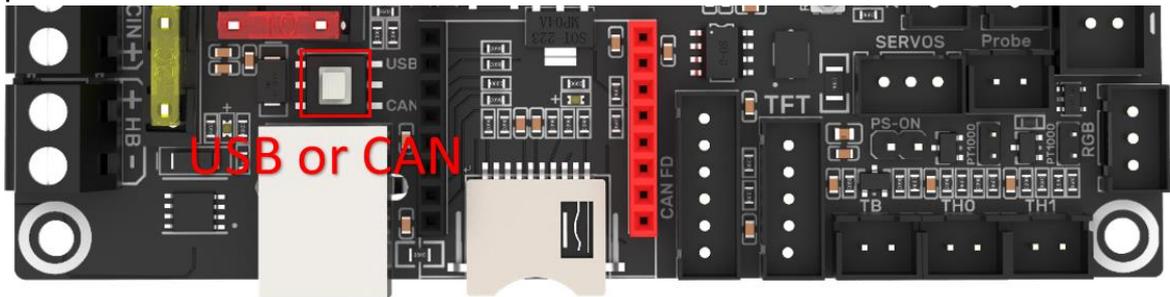
### 3.3.3 TMC-driven DIAG mode(Sensorless Homing)

As shown in the pictures, plug the jumper cap when using the Sensorless Homing function, and leave it unplugged when not in use. There is no need to cut the DIAG pin of the driver.



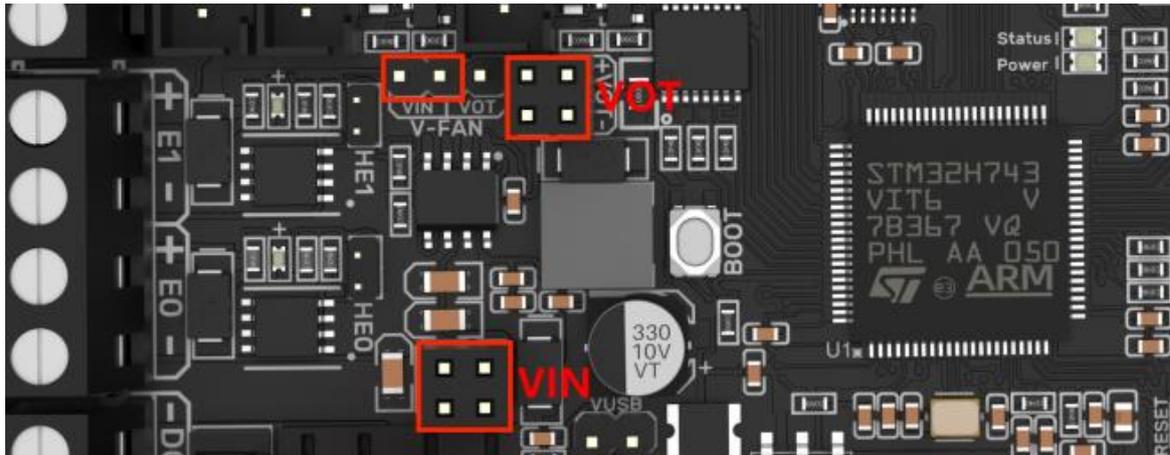
### 3.4 USB and CAN Mode

As shown in the figure below, the double-pole double-throw switch is in USB mode when it is in the pop-up state, and in CAN FD mode when it is in the pressed state.

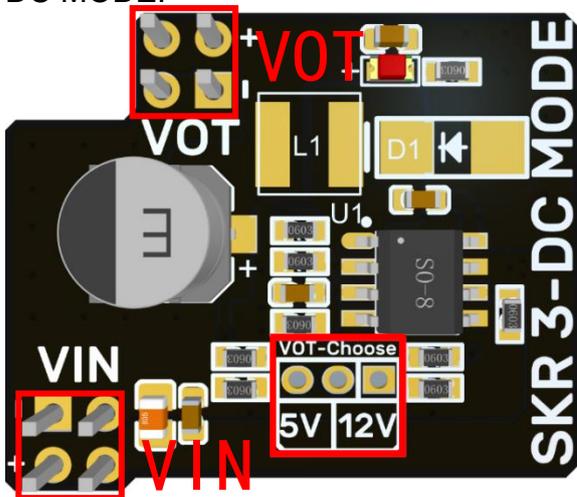


### 3.5 Voltage selection for NC fans

If DCIN is used as the power supply of the numerical control fan, a jumper cap should be used to short-circuit the two pins within the VIN range. If you want to use 12V or 5V as the NC fan power supply, you need to make a jumper cap short-circuit two pins within the VOT range, and insert the SKR 3-DC MODE into the 2\*4Pin VOT and VIN headers.

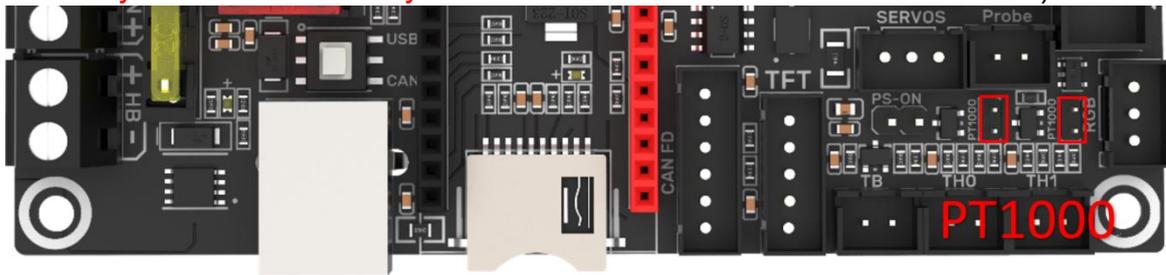


Set the VOT output voltage to 5V or 12V by setting the jumper cap on the SKR 3-DC MODE.

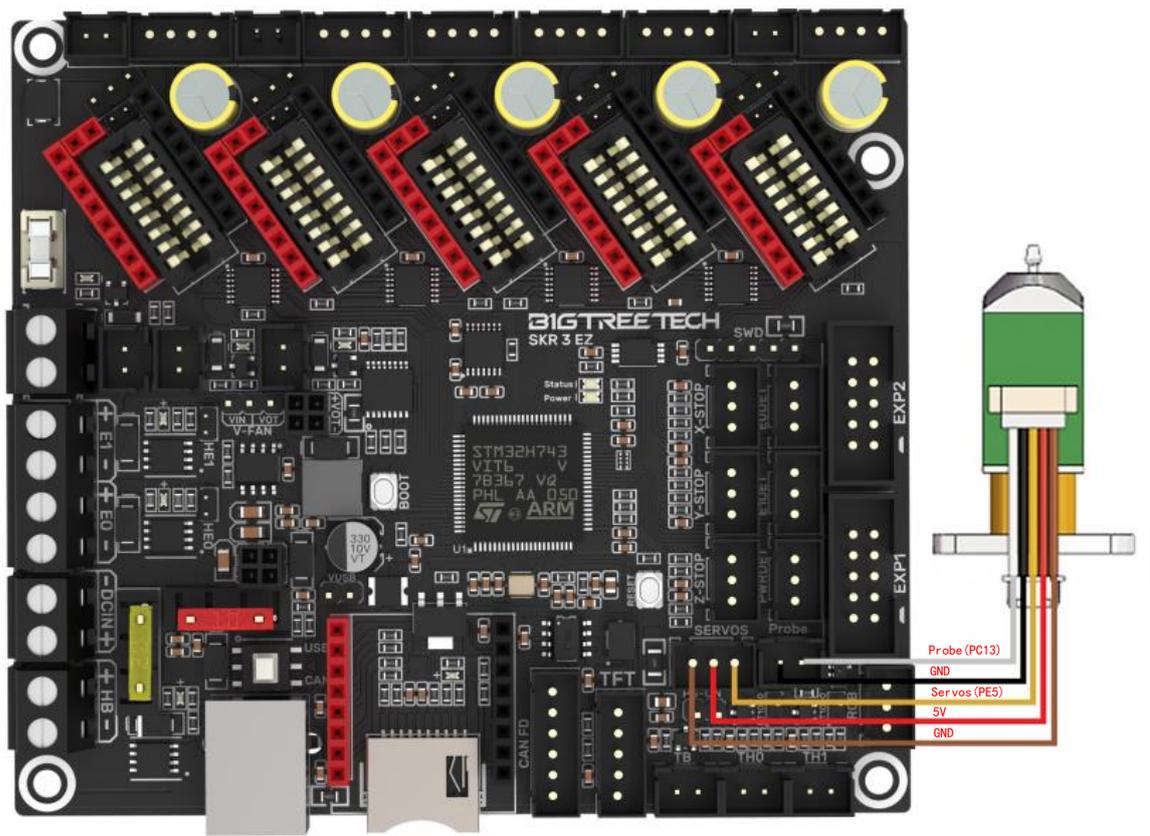


## 3.6 100K NTC or PT1000 Setup

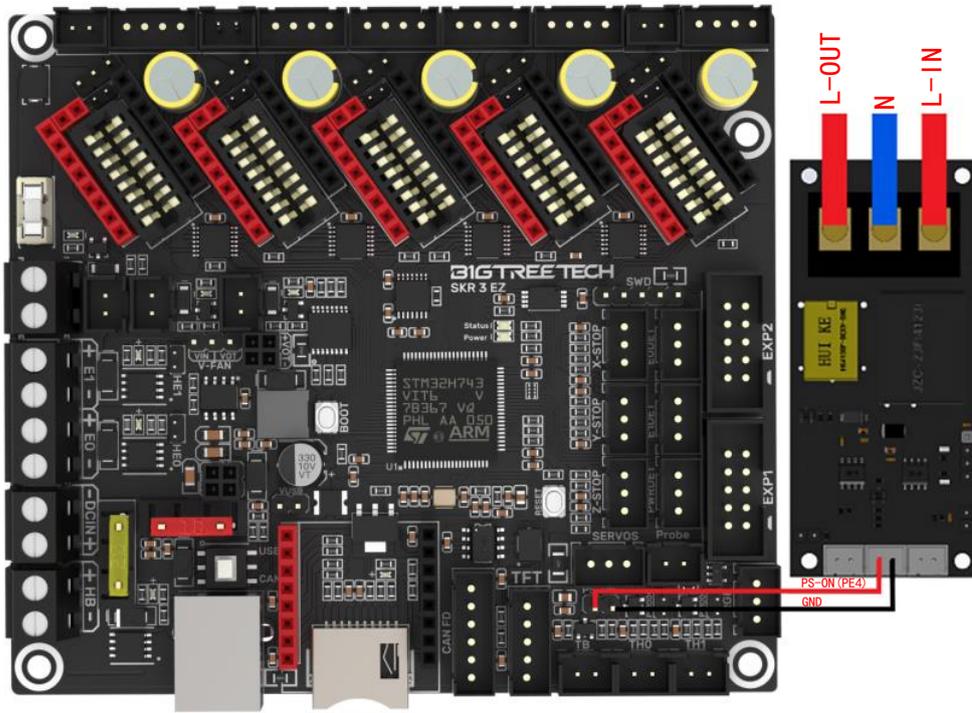
When using a 100K NTC thermistor, no need to insert a jumper cap. At this time, the pull-up resistors of TH0 and TH1 are 4.7K. When using PT1000, you need to use jump caps to short-circuit the two pins in the red box in the picture below. At this time, the pull-up resistors of TH0 and TH1 are 2.2K (Note: the temperature accuracy read out in this way will be much worse than that of MAX31865).



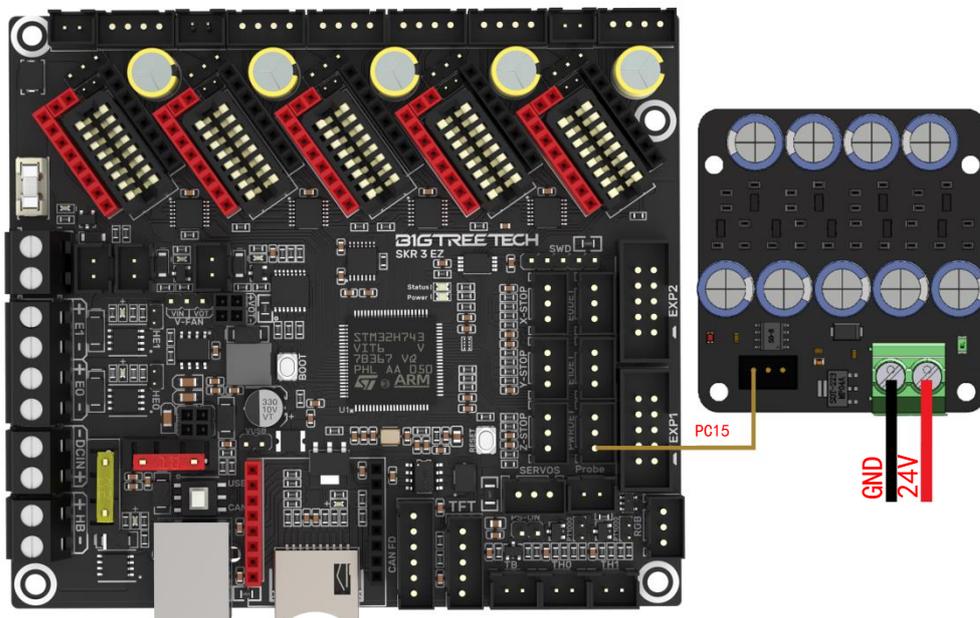
## 3.7 BLTouch Connection



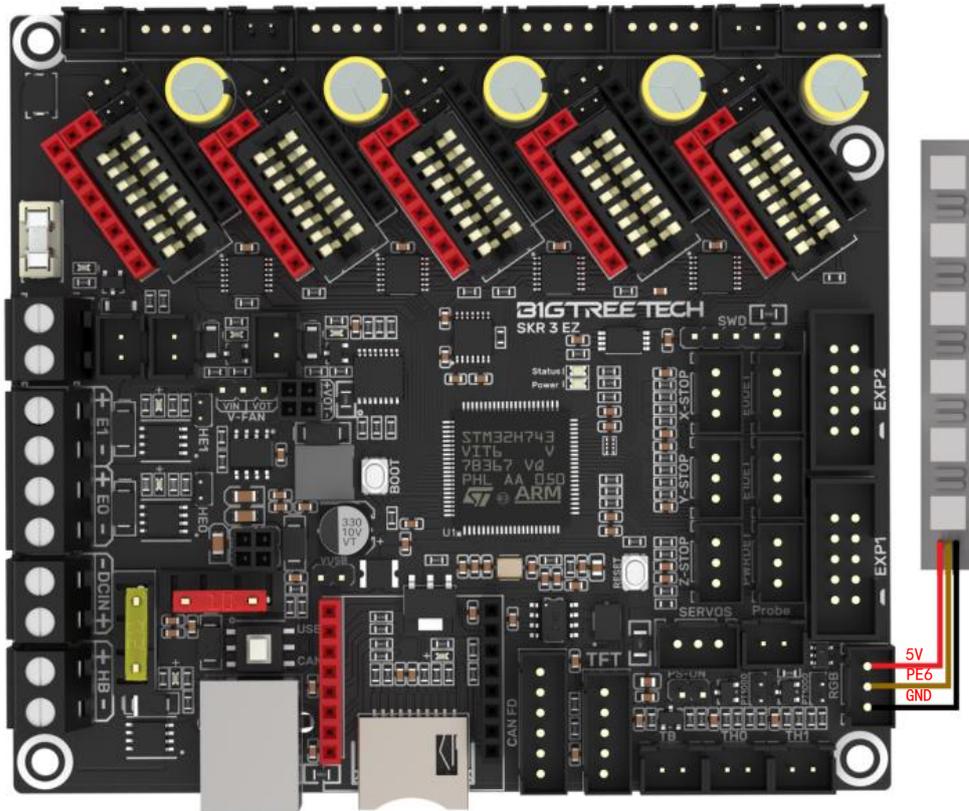
### 3.8 Completed Shut-down Module(Relay V1.2) Connection



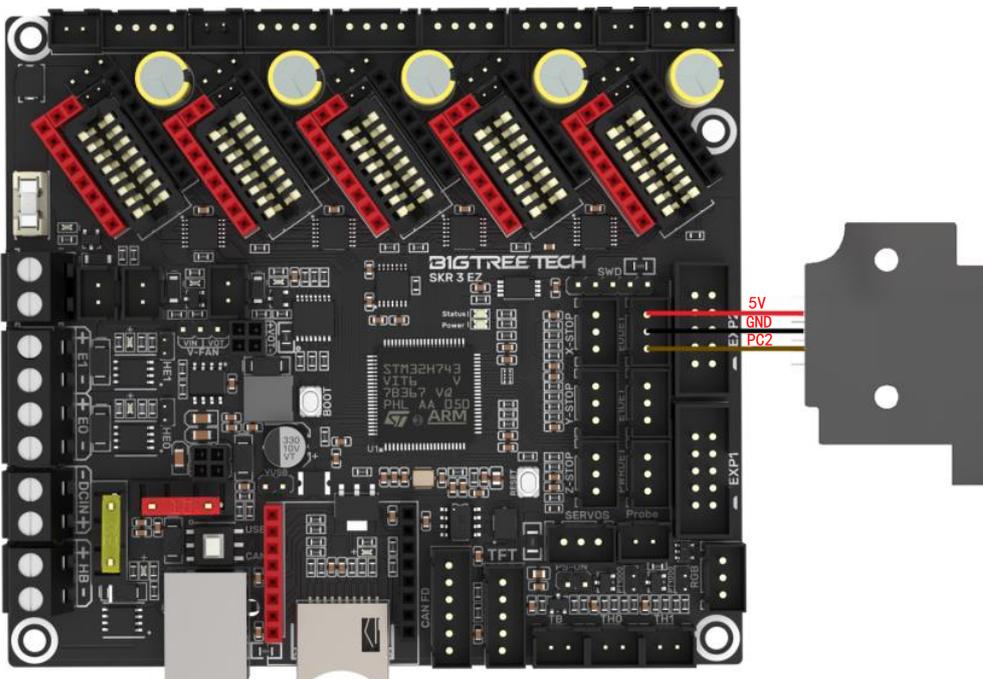
### 3.9 Resume Printing(UPS 24V V1.0) Connection



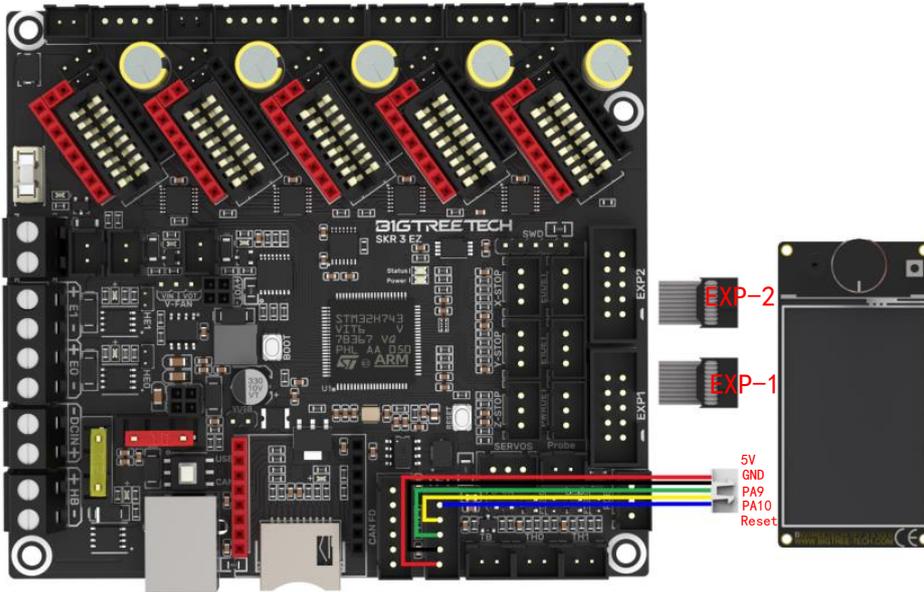
### 3.10 RGB Connection



### 3.11 Break Detection Connection

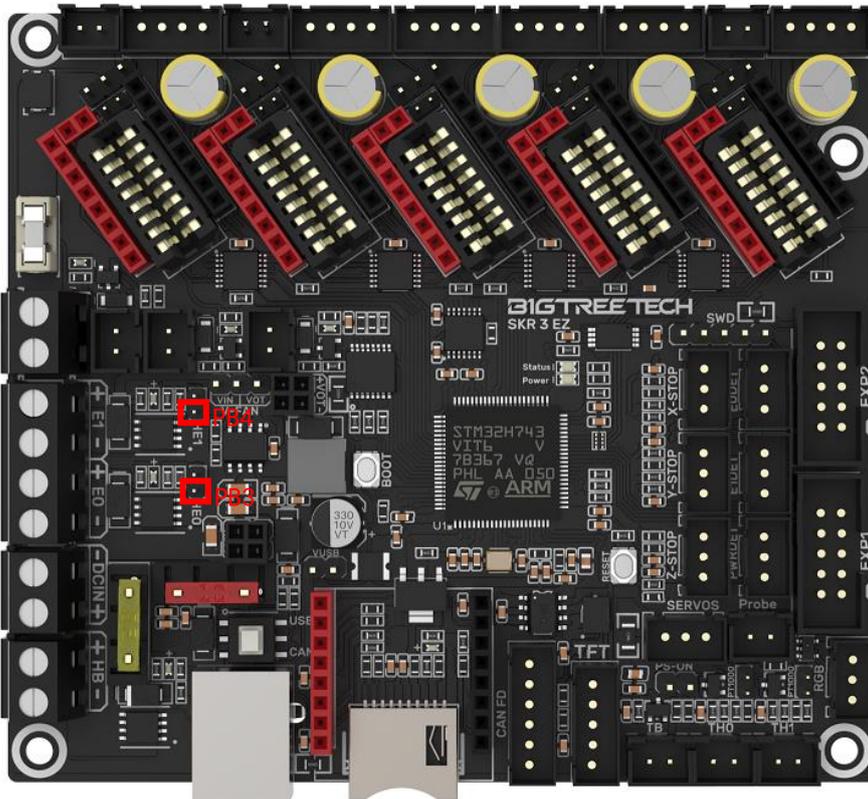


### 3.12 Touch Screen Connection



### 3.13 Heater cartridge IO

The IO of the SKR 3 heater cartridge is wired to the MOS by the jumper. You can remove the jumper and connect the IO to device directly if you need to use the laser or other device that need PWM. (Note: The IO passes through logic conversion chip, the output high level is 5V, and cannot be used as an input)



## 4. Marlin

### 4.1 Compiler Environment Installation

<https://github.com/bigtreetech/Document/blob/master/How%20to%20install%20VSCode%2BPlatformio.md>

[https://marlinfw.org/docs/basics/install\\_platformio\\_vscode.html](https://marlinfw.org/docs/basics/install_platformio_vscode.html)

Refer to the instructions in these two links to install VSCode and PlatformIO plugins (domestic users may be slow to install PlatformIO plugins online).

### 4.2 Download of Marlin Firmware

1. Download the latest version of the bugfix firmware from the Marlin official website:

<https://github.com/MarlinFirmware/Marlin/tree/bugfix-2.0.x>

2. Download pre-configured firmware of Compiler Environment and board type from our GitHub:

<https://github.com/bigtreetech/SKR-3>

### 4.3 Firmware Configuration

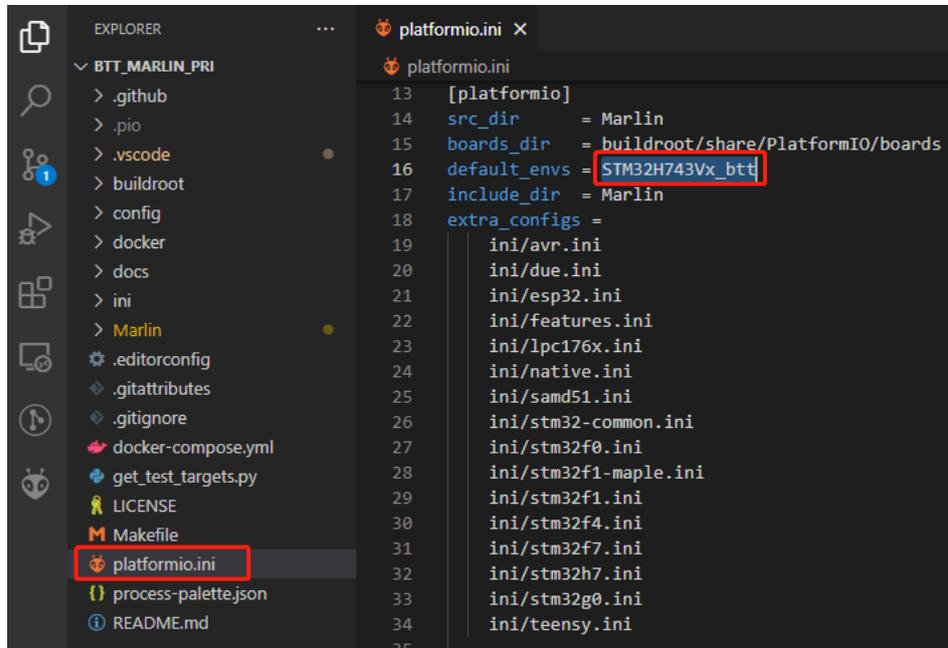
#### 4.3.1 Open the Marlin Project

You can open Marlin in VSCode in one of the following ways:

- Drag the downloaded Marlin Firmware folder onto the VSCode application icon.
- Use the **Open...** command from the VSCode **File** menu.
- Open the PIO Home tab and click the **"Open Project"** button.

### 4.3.2 Compiler Environment Configuration

Open `platformio.ini` file and modify `default_envs` to `STM32H743Vx_btt`.



### 4.3.3 Motherboard type and Serial port number Configuration

Set Motherboard type `MOTHERBOARD` to `BOARD_BTT_SKR_3`

```
#define MOTHERBOARD BOARD_BTT_SKR_3
```

```
#define SERIAL_PORT 1 (Enable TFT serial port)
```

```
#define BAUDRATE 115200 (Set the baud rate, pay attention to the same as the communication device)
```

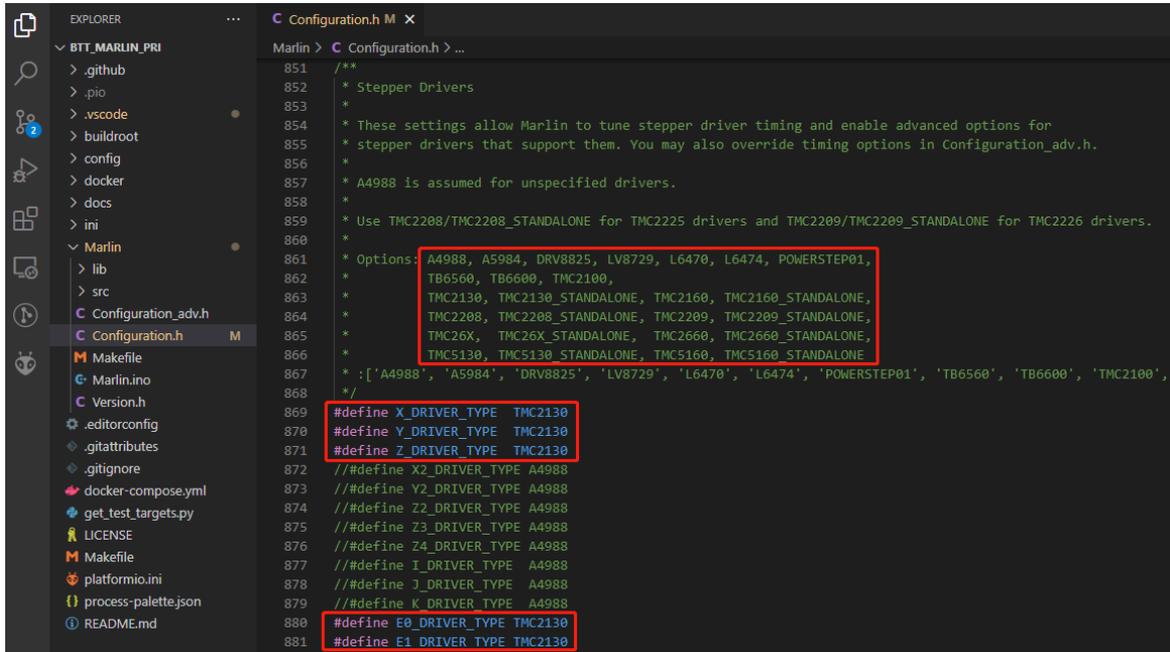
```
#define SERIAL_PORT_2 -1 (Enable USB emulated serial port)
```

```
#define SERIAL_PORT_3 3 (Enable WIFI serial port)
```

The above settings can be enabled according to the needs.

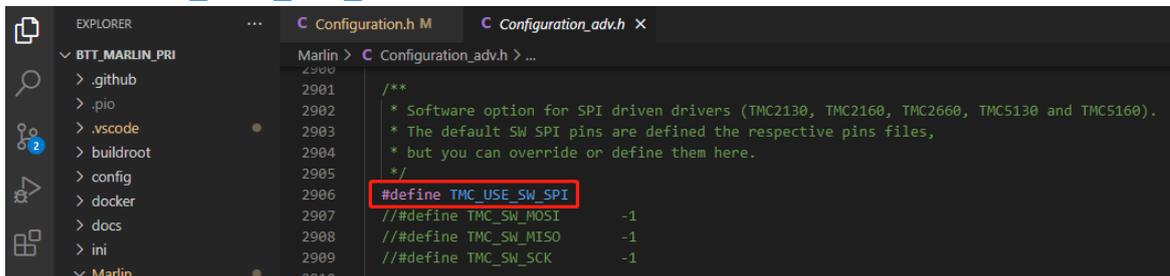
```
97 // Choose the name from boards.h that matches your setup
98 #ifndef MOTHERBOARD
99 #define MOTHERBOARD BOARD_BTT_SKR
100 #endif
101
102 /**
103  * Select the serial port on the board to use for communication with the host.
104  * This allows the connection of wireless adapters (for instance) to non-default port pins.
105  * Serial port -1 is the USB emulated serial port, if available.
106  * Note: The first serial port (-1 or 0) will always be used by the Arduino bootloader.
107  *
108  * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]
109  */
110 #define SERIAL_PORT 1
111
112 /**
113  * Serial Port Baud Rate
114  * This is the default communication speed for all serial ports.
115  * Set the baud rate defaults for additional serial ports below.
116  *
117  * 250000 works in most cases, but you might try a lower speed if
118  * you commonly experience drop-outs during host printing.
119  * You may try up to 1000000 to speed up SD file transfer.
120  *
121  * :[2400, 9600, 19200, 38400, 57600, 115200, 250000, 500000, 1000000]
122  */
123 #define BAUDRATE 115200
124 // #define BAUD_RATE_GCODE // Enable G-code M575 to set the baud rate
125
126 /**
127  * Select a secondary serial port on the board to use for communication with the host.
128  * Currently Ethernet (-2) is only supported on Teensy 4.1 boards.
129  * :[-2, -1, 0, 1, 2, 3, 4, 5, 6, 7]
130  */
131 #define SERIAL_PORT_2 -1
132 // #define BAUDRATE_2 250000 // Enable to override BAUDRATE
133
134 /**
135  * Select a third serial port on the board to use for communication with the host.
136  * Currently only supported for AVR, DUE, LPC1768/9 and STM32/STM32F1
137  * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]
138  */
139 #define SERIAL_PORT_3 3
140 // #define BAUDRATE_3 250000 // Enable to override BAUDRATE
141
```

### 4.3.4 Motor Driver Configuration



```
851 /**
852  * Stepper Drivers
853  *
854  * These settings allow Marlin to tune stepper driver timing and enable advanced options for
855  * stepper drivers that support them. You may also override timing options in Configuration_adv.h.
856  *
857  * A4988 is assumed for unspecified drivers.
858  *
859  * Use TMC2208/TMC2208_STANDALONE for TMC2225 drivers and TMC2209/TMC2209_STANDALONE for TMC2226 drivers.
860  *
861  * Options: A4988, A5984, DRV8825, LV8729, L6470, L6474, POWERSTEP01,
862  *          TB6560, TB6600, TMC2100,
863  *          TMC2130, TMC2130_STANDALONE, TMC2160, TMC2160_STANDALONE,
864  *          TMC2208, TMC2208_STANDALONE, TMC2209, TMC2209_STANDALONE,
865  *          TMC26X, TMC26X_STANDALONE, TMC2660, TMC2660_STANDALONE,
866  *          TMC5130, TMC5130_STANDALONE, TMC5160, TMC5160_STANDALONE
867  */
868 #define X_DRIVER_TYPE  TMC2130
869 #define Y_DRIVER_TYPE  TMC2130
870 #define Z_DRIVER_TYPE  TMC2130
871 // #define X2_DRIVER_TYPE  A4988
872 // #define Y2_DRIVER_TYPE  A4988
873 // #define Z2_DRIVER_TYPE  A4988
874 // #define Z3_DRIVER_TYPE  A4988
875 // #define Z4_DRIVER_TYPE  A4988
876 // #define I_DRIVER_TYPE  A4988
877 // #define J_DRIVER_TYPE  A4988
878 // #define K_DRIVER_TYPE  A4988
879 #define E0_DRIVER_TYPE TMC2130
880 #define E1_DRIVER_TYPE TMC2130
```

If the driver used is SPI mode, you also need to enable `TMC_USE_SW_SPI`  
`#define TMC_USE_SW_SPI`



```
2900
2901 /**
2902  * Software option for SPI driven drivers (TMC2130, TMC2160, TMC2660, TMC5130 and TMC5160).
2903  * The default SW SPI pins are defined the respective pins files,
2904  * but you can override or define them here.
2905  */
2906 #define TMC_USE_SW_SPI
2907 // #define TMC_SW_MOSI    -1
2908 // #define TMC_SW_MISO    -1
2909 // #define TMC_SW_SCK     -1
```

### 4.3.5 Sensorless Homing

```

3047 /**
3048  * Use StallGuard to home / probe X, Y, Z.
3049  *
3050  * TMC2130, TMC2160, TMC2209, TMC2660, TMC5130, and TMC5160 only
3051  * Connect the stepper driver's DIAG1 pin to the X/Y endstop pin.
3052  * X, Y, and Z homing will always be done in spreadCycle mode.
3053  *
3054  * X/Y/Z_STALL_SENSITIVITY is the default stall threshold.
3055  * Use M914 X Y Z to set the stall threshold at runtime:
3056  *
3057  * Sensitivity  TMC2209  Others
3058  * HIGHEST     255     -64   (Too sensitive => False positive)
3059  * LOWEST      0       63   (Too insensitive => No trigger)
3060  *
3061  * It is recommended to set HOMING_BUMP_MM to { 0, 0, 0 }.
3062  *
3063  * SPI_ENDSTOPS *** Beta feature! *** TMC2130/TMC5160 Only ***
3064  * Poll the driver through SPI to determine load when homing.
3065  * Removes the need for a wire from DIAG1 to an endstop pin.
3066  *
3067  * IMPROVE_HOMING_RELIABILITY tunes acceleration and jerk when
3068  * homing and adds a guard period for endstop triggering.
3069  *
3070  * Comment *_STALL_SENSITIVITY to disable sensorless homing for that axis.
3071  */
3072 #define SENSORLESS_HOMING // StallGuard capable drivers only
3073
3074 #if EITHER(SENSORLESS_HOMING, SENSORLESS_PROBING)
3075 // TMC2209: 0..255. TMC2130: -64..63
3076 #define X_STALL_SENSITIVITY 8
3077 #define X2_STALL_SENSITIVITY X_STALL_SENSITIVITY
3078 #define Y_STALL_SENSITIVITY 8
3079 #define Y2_STALL_SENSITIVITY Y_STALL_SENSITIVITY
3080 //#define Z_STALL_SENSITIVITY 8
3081 //#define Z2_STALL_SENSITIVITY Z_STALL_SENSITIVITY
3082 //#define Z3_STALL_SENSITIVITY Z_STALL_SENSITIVITY
3083 //#define Z4_STALL_SENSITIVITY Z_STALL_SENSITIVITY
3084 //#define I_STALL_SENSITIVITY 8
3085 //#define J_STALL_SENSITIVITY 8
3086 //#define K_STALL_SENSITIVITY 8
3087 //#define SPI_ENDSTOPS // TMC2130 only
3088 #define IMPROVE_HOMING_RELIABILITY
3089 #endif

```

`#define SENSORLESS_HOMING` //Turn on drive stall detection as a function of the Home limit switch.

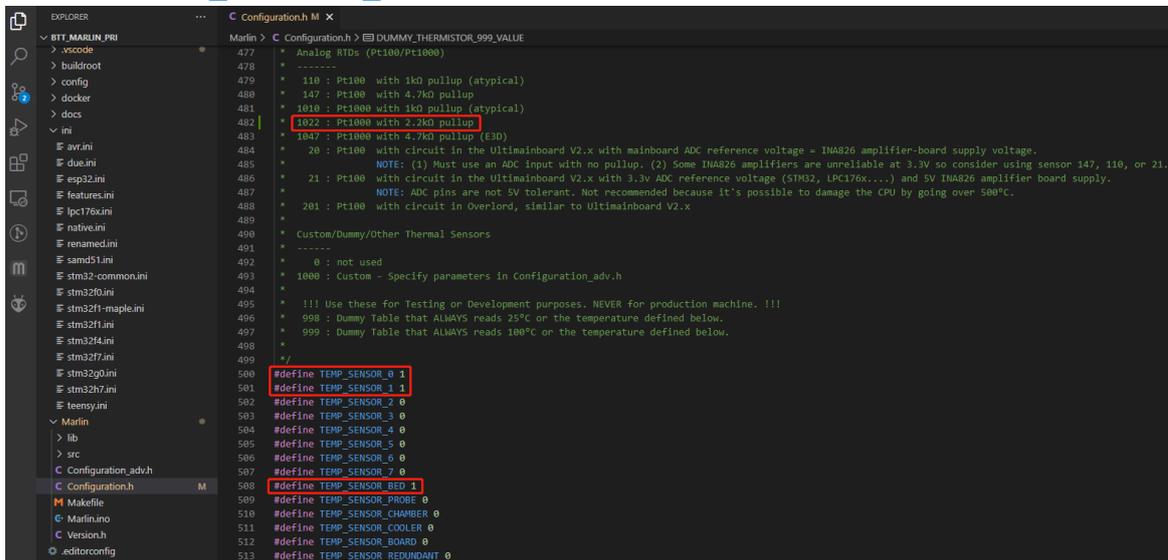
`#define xx_STALL_SENSITIVITY 8` // Set the sensitivity of stall detection. The range of TMC2209 is 0~255. The larger the value, the more sensitive it is, and it is easy to trigger falsely. When the phenomenon is Home, the axis stops before returning to the origin. The smaller the value, the less sensitive it is, and the easier it is not to trigger. Make a "Deng Deng Deng" sound. Other driving ranges are 63~-64, the smaller the value, the more sensitive. `#define IMPROVE_HOMING_RELIABILITY` // Set the current parameter above(`X_CURRENT_HOME`) when returning to zero separately , so as to get the best zeroing effect

`#define IMPROVE_HOMING_RELIABILITY` // The current parameter (xx\_CURRENT\_HOME) during zeroing can be set separately above to get the best zeroing effect

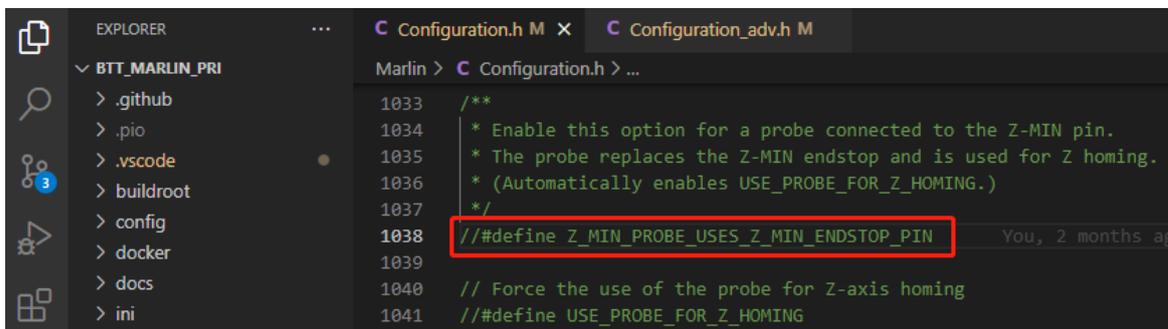
### 4.3.6 100K NTC or PT1000

Set the pull-up resistor of the thermistor to 4.7K (with 100K NTC) or 2.2K (with PT1000) through the jumper cap, 1 in Marlin firmware means 100K NTC + 4.7K pull-up resistor, 1022 means PT1000 + 2.2K pull-up Resistance (**Note**: The temperature accuracy read in this way will be much worse than the MAX31865).

```
#define TEMP_SENSOR_0 1
#define TEMP_SENSOR_1 1
#define TEMP_SENSOR_BED 1
```



### 4.3.7 BL Touch



`//#define Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN` // Do not remap Z\_PROBE\_PIN to Z\_MIN port

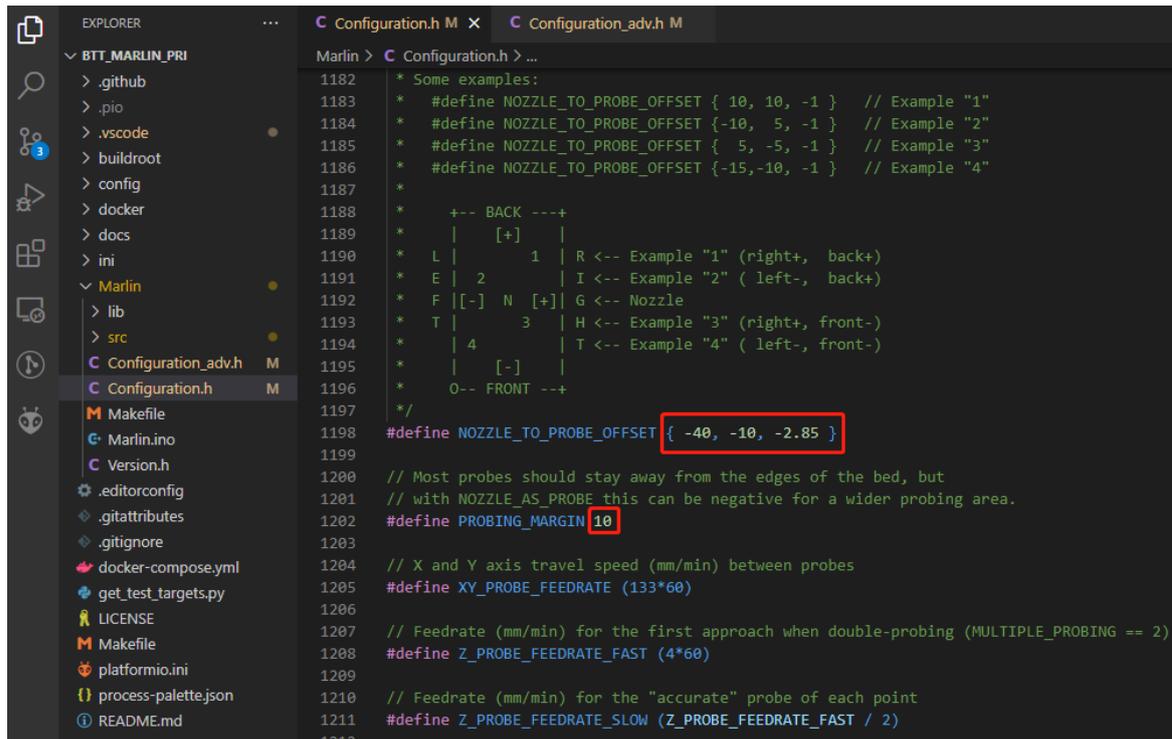
# BIGTREETECH



```
EXPLORER
  BTT_MARLIN_PRI
    .github
    .pio
    .vscode
    buildroot
  config
  docker
  docs
  ini
  Marlin
    lib
    src
  Configuration_adv.h M
  Configuration.h M
  Makefile
  Marlin.ino
  Version.h
  .editorconfig
  .gitattributes
  .gitignore
  docker-compose.yml
  get_test_targets.py
  LICENSE
  Makefile
  platformio.ini
  process-palette.json
  README.md

Marlin > C Configuration.h > ...
1092 /**
1093  * The BLTouch probe uses a Hall effect sensor and emulates a servo.
1094  */
1095 #define BLTOUCH
1096
```

#define BLTOUCH // Enable BL Touch function

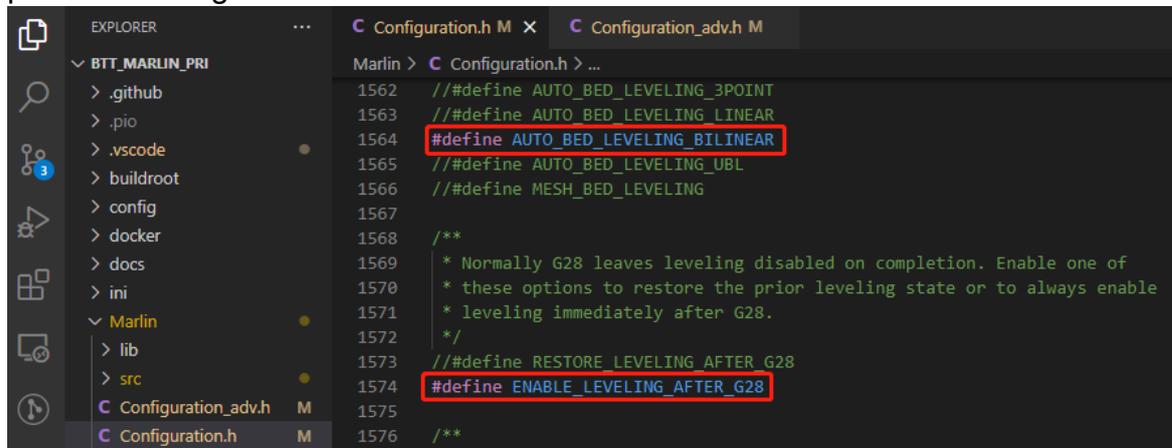


```
EXPLORER
  BTT_MARLIN_PRI
    .github
    .pio
    .vscode
    buildroot
  config
  docker
  docs
  ini
  Marlin
    lib
    src
  Configuration_adv.h M
  Configuration.h M
  Makefile
  Marlin.ino
  Version.h
  .editorconfig
  .gitattributes
  .gitignore
  docker-compose.yml
  get_test_targets.py
  LICENSE
  Makefile
  platformio.ini
  process-palette.json
  README.md

Marlin > C Configuration.h > ...
1182 * Some examples:
1183 * #define NOZZLE_TO_PROBE_OFFSET { 10, 10, -1 } // Example "1"
1184 * #define NOZZLE_TO_PROBE_OFFSET {-10, 5, -1 } // Example "2"
1185 * #define NOZZLE_TO_PROBE_OFFSET { 5, -5, -1 } // Example "3"
1186 * #define NOZZLE_TO_PROBE_OFFSET {-15,-10, -1 } // Example "4"
1187 *
1188 * +-+ BACK +-+
1189 * | | [+ ] |
1190 * L | 1 | R <-- Example "1" (right+, back+)
1191 * E | 2 | I <-- Example "2" ( left-, back+)
1192 * F |[-] N [+ ] G <-- Nozzle
1193 * T | 3 | H <-- Example "3" (right+, front-)
1194 * | 4 | T <-- Example "4" ( left-, front-)
1195 * | | [- ] |
1196 * 0-- FRONT --+
1197 */
1198 #define NOZZLE_TO_PROBE_OFFSET { -40, -10, -2.85 }
1199
1200 // Most probes should stay away from the edges of the bed, but
1201 // with NOZZLE_AS_PROBE this can be negative for a wider probing area.
1202 #define PROBING_MARGIN 10
1203
1204 // X and Y axis travel speed (mm/min) between probes
1205 #define XY_PROBE_FEEDRATE (133*60)
1206
1207 // Feedrate (mm/min) for the first approach when double-probing (MULTIPLE_PROBING == 2)
1208 #define Z_PROBE_FEEDRATE_FAST (4*60)
1209
1210 // Feedrate (mm/min) for the "accurate" probe of each point
1211 #define Z_PROBE_FEEDRATE_SLOW (Z_PROBE_FEEDRATE_FAST / 2)
1212
```

#define NOZZLE\_TO\_PROBE\_OFFSET { -40, -10, -2.85 } // Set up the offset of the BL Touch probe relative to the nozzle

#define PROBING\_MARGIN 10 // Set up the distance from the leveling detection point to the edge



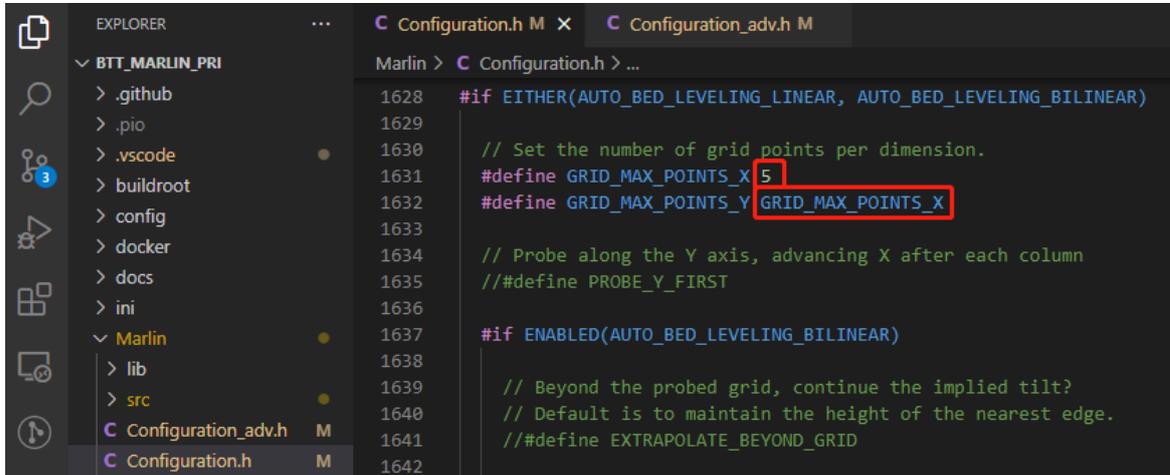
```
EXPLORER
  BTT_MARLIN_PRI
    .github
    .pio
    .vscode
    buildroot
  config
  docker
  docs
  ini
  Marlin
    lib
    src
  Configuration_adv.h M
  Configuration.h M
  Makefile
  Marlin.ino
  Version.h
  .editorconfig
  .gitattributes
  .gitignore
  docker-compose.yml
  get_test_targets.py
  LICENSE
  Makefile
  platformio.ini
  process-palette.json
  README.md

Marlin > C Configuration.h > ...
1562 //#define AUTO_BED_LEVELING_3POINT
1563 //#define AUTO_BED_LEVELING_LINEAR
1564 #define AUTO_BED_LEVELING_BILINEAR
1565 //#define AUTO_BED_LEVELING_UBL
1566 //#define MESH_BED_LEVELING
1567
1568 /**
1569  * Normally G28 leaves leveling disabled on completion. Enable one of
1570  * these options to restore the prior leveling state or to always enable
1571  * leveling immediately after G28.
1572  */
1573 //#define RESTORE_LEVELING_AFTER_G28
1574 #define ENABLE_LEVELING_AFTER_G28
1575
1576 /**
```

#define AUTO\_BED\_LEVELING\_BILINEAR // Set up leveling strategy

## BIGTREETECH

`#define RESTORE_LEVELING_AFTER_G28` // Auto reload level compensation after Home

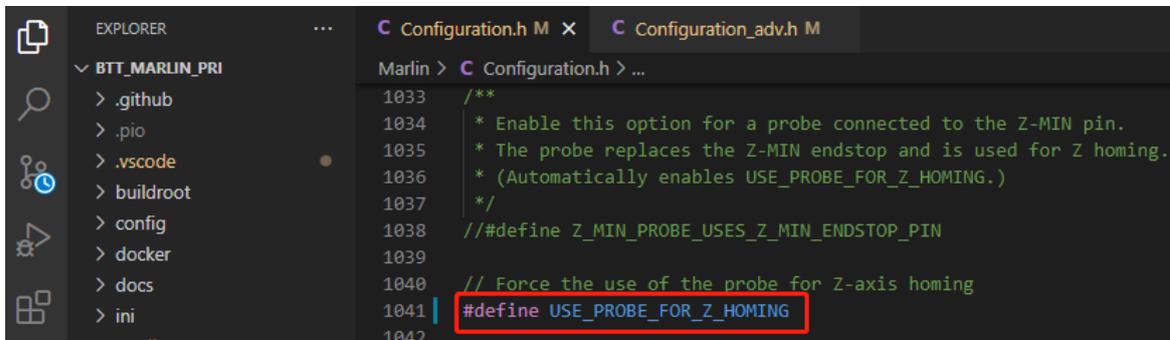


```
1628 #if EITHER(AUTO_BED_LEVELING_LINEAR, AUTO_BED_LEVELING_BILINEAR)
1629
1630 // Set the number of grid points per dimension.
1631 #define GRID_MAX_POINTS_X 5
1632 #define GRID_MAX_POINTS_Y GRID_MAX_POINTS_X
1633
1634 // Probe along the Y axis, advancing X after each column
1635 //#define PROBE_Y_FIRST
1636
1637 #if ENABLED(AUTO_BED_LEVELING_BILINEAR)
1638
1639 // Beyond the probed grid, continue the implied tilt?
1640 // Default is to maintain the height of the nearest edge.
1641 //#define EXTRAPOLATE_BEYOND_GRID
1642
```

`#define GRID_MAX_POINTS_X 5` // Set up the number of points for leveling detection, 5 points for X-axis detection

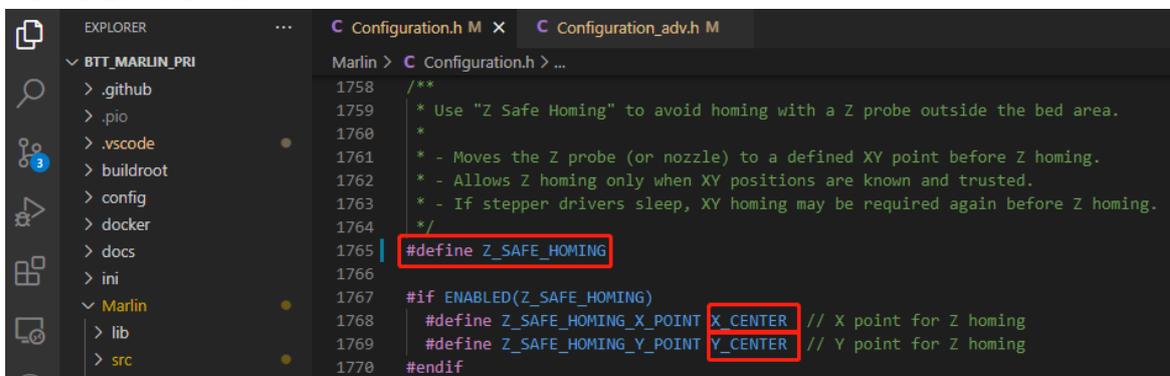
`#define GRID_MAX_POINTS_Y GRID_MAX_POINTS_X` // Y-axis probes 5 points

If you want to use BL Touch as the Z-axis limit switch, you don't need to change the connection just need to modify the firmware settings.



```
1033 /**
1034  * Enable this option for a probe connected to the Z-MIN pin.
1035  * The probe replaces the Z-MIN endstop and is used for Z homing.
1036  * (Automatically enables USE_PROBE_FOR_Z_HOMING.)
1037  */
1038 //#define Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN
1039
1040 // Force the use of the probe for Z-axis homing
1041 #define USE_PROBE_FOR_Z_HOMING
1042
```

`#define USE_PROBE_FOR_Z_HOMING` // Use Z Probe(BL Touch) as Z-axis Home Limit Switch



```
1758 /**
1759  * Use "Z Safe Homing" to avoid homing with a Z probe outside the bed area.
1760  *
1761  * - Moves the Z probe (or nozzle) to a defined XY point before Z homing.
1762  * - Allows Z homing only when XY positions are known and trusted.
1763  * - If stepper drivers sleep, XY homing may be required again before Z homing.
1764  */
1765 #define Z_SAFE_HOMING
1766
1767 #if ENABLED(Z_SAFE_HOMING)
1768 #define Z_SAFE_HOMING_X_POINT X_CENTER // X point for Z homing
1769 #define Z_SAFE_HOMING_Y_POINT Y_CENTER // Y point for Z homing
1770 #endif
```

`#define Z_SAFE_HOMING` //When the Z-axis is Home, move X and Y to the specified coordinates (usually the center of the platform) to ensure that when the

Z-axis is Home, the probe of the Z Probe (BL Touch) is within the scope of the platform.

### 4.3.8 Completed Shutdown Module (Relay V1.2)

```

359  /**
360  * Power Supply Control
361  *
362  * Enable and connect the power supply to the PS_ON_PIN.
363  * Specify whether the power supply is active HIGH or active LOW.
364  */
365  #define PSU_CONTROL
366  #define PSU_NAME "Power Supply"
367
368  #if ENABLED(PSU_CONTROL)
369  // #define MKS_PWC // Using the MKS PWC add-on
370  // #define PS_OFF_CONFIRM // Confirm dialog when power off
371  // #define PS_OFF_SOUND // Beep 1s when power off
372  #define PSU_ACTIVE_STATE HIGH // Set 'LOW' for ATX, 'HIGH' for X-Box
373
374  // #define PSU_DEFAULT_OFF // Keep power off until enabled directly with M80
375  // #define PSU_POWERUP_DELAY 250 // (ms) Delay for the PSU to warm up to full power
376
377  // #define POWER_OFF_TIMER // Enable M81 D<seconds> to power off after a delay
378  // #define POWER_OFF_WAIT_FOR_COOLDOWN // Enable M81 S to power off only after cooldown

```

`#define PSU_CONTROL` // Turn on the control power function, you can turn on through the M80 and turn off through the M81

`#define PSU_ACTIVE_STATE HIGH` // Set up the power-on level. The Relay V1.2 module is powered on at a high level and powered off at a low level, so it needs to be set to HIGH

### 4.3.9 Resume Printing

There are currently two ways to realize the resume printing:

1.No external module is required, the firmware regularly saves the printing status to the SD card, and continues to print from the point saved in the SD card after a power failure and restart. The disadvantage of this method is that data is frequently written to the SD card, which greatly affects the SD card lifespan.

```

1459  * Store the current state to the SD Card at the start of each layer
1460  * during SD printing. If the recovery file is found at boot time, present
1461  * an option on the LCD screen to continue the print from the last-known
1462  * point in the file.
1463  */
1464  #define POWER_LOSS_RECOVERY
1465  #if ENABLED(POWER_LOSS_RECOVERY)
1466  #define PLR_ENABLED_DEFAULT true // Power Loss Recovery enabled by default. (Set with 'M413 Sn' & M500)
1467  #define BACKUP_POWER_SUPPLY 10 // Backup power / UPS to move the steppers on power loss
1468  #define POWER_LOSS_Z_RAISE 10 // (mm) Z axis raise on resume (on power loss with UPS)
1469  // #define POWER_LOSS_PIN 44 // Pin to detect power loss. Set to -1 to disable default pin on boards without module.
1470  // #define POWER_LOSS_STATE HIGH // State of pin indicating power loss
1471  // #define POWER_LOSS_PULLUP // Set pullup / pulldown as appropriate for your sensor
1472  // #define POWER_LOSS_PULLDOWN
1473  #define POWER_LOSS_PURGE_LEN 20 // (mm) Length of filament to purge on resume
1474  #define POWER_LOSS_RETRACT_LEN 10 // (mm) Length of filament to retract on fall. Requires backup power.
1475
1476  // Without a POWER_LOSS_PIN the following option helps reduce wear on the SD card,
1477  // especially with "vase mode" printing. Set too high and vases cannot be continued.
1478  #define POWER_LOSS_MIN_Z_CHANGE 0.05 // (mm) Minimum Z change before saving power-loss data
1479
1480  // Enable if Z homing is needed for proper recovery. 99.9% of the time this should be disabled!
1481  // #define POWER_LOSS_RECOVER_ZHOME
1482  #if ENABLED(POWER_LOSS_RECOVER_ZHOME)
1483  // #define POWER_LOSS_ZHOME_POS { 0, 0 } // Safe XY position to home Z while avoiding objects on the bed
1484  #endif
1485  #endif

```

```
#define POWER_LOSS_RECOVERY // Enable resume printing function
#define PLR_ENABLED_DEFAULT true // true default to use open resume printing
```

2. The external module UPS 24V V1.0 provides power and sends a signal to the mainboard when it is power-off, reminding the mainboard to save the printing state. This method only writes data to the SD card when the power is off, and has little effect on the lifespan of the SD card.

```

1459 * Store the current state to the SD Card at the start of each layer
1460 * during SD printing. If the recovery file is found at boot time, present
1461 * an option on the LCD screen to continue the print from the last-known
1462 * point in the file.
1463 */
1464 #define POWER_LOSS_RECOVERY
1465 #if ENABLED(POWER_LOSS_RECOVERY)
1466 #define PLR_ENABLED_DEFAULT true // Power Loss Recovery enabled by default. (Set with 'M413 Sn' & M500)
1467 #define BACKUP_POWER_SUPPLY // Backup power / UPS to move the steppers on power loss
1468 #define POWER_LOSS_ZRAISE 10 // (mm) Z axis raise on resume (on power loss with UPS)
1469 #define POWER_LOSS_PIN 44 // Pin to detect power loss. Set to -1 to disable default pin on boards without module.
1470 #define POWER_LOSS_STATE HIGH // State of pin indicating power loss
1471 #define POWER_LOSS_PULLUP // Set pullup / pulldown as appropriate for your sensor
1472 // #define POWER_LOSS_PULLDOWN
1473 #define POWER_LOSS_PURGE_LEN 20 // (mm) Length of filament to purge on resume
1474 #define POWER_LOSS_RETRACT_LEN 10 // (mm) Length of filament to retract on fail. Requires backup power.
1475 // Without a POWER_LOSS_PIN the following option helps reduce wear on the SD card,
1476 // especially with "vase mode" printing. Set too high and vases cannot be continued.
1477 #define POWER_LOSS_MIN_Z_CHANGE 0.05 // (mm) Minimum Z change before saving power-loss data
1478 // Enable if Z homing is needed for proper recovery. 99.9% of the time this should be disabled!
1479 // #define POWER_LOSS_RECOVER_ZHOME
1480 // #define POWER_LOSS_RECOVER_ZHOME
1481 #if ENABLED(POWER_LOSS_RECOVER_ZHOME)
1482 // #define POWER_LOSS_RECOVER_ZHOME_POS { 0, 0 } // Safe XY position to home Z while avoiding objects on the bed
1483 #endif
1484 #endif
1485 #endif

```

```
#define POWER_LOSS_RECOVERY // Enable resume printing function
```

```
#define PLR_ENABLED_DEFAULT true // true default to use open resume printing
```

```
#define POWER_LOSS_ZRAISE 10 // When the power is off, the nozzle is raised by 10mm to prevent the nozzle from scalding the model
```

```
#define POWER_LOSS_STATE HIGH // When the UPS 24V V1.0 is working normally, the module feedback a low level, and when the power is off, the feedback is a high level, so it is set to HIGH
```

### 4.3.10 RGB Light

```

2926 // Support for Adafruit NeoPixel LED driver
2927 #define NEOPIXEL_LED
2928 #if ENABLED(NEOPIXEL_LED)
2929 #define NEOPIXEL_TYPE NEO_GRB // NEO_GRBW / NEO_GRB - four/three channel driver type (defined in Adafruit_NeoPixel.h)
2930 #define NEOPIXEL_PIN 4 // LED driving pin
2931 #define NEOPIXEL2_TYPE NEOPIXEL_TYPE
2932 // #define NEOPIXEL2_PIN 5
2933 #define NEOPIXEL_PIXELS 30 // Number of LEDs in the strip. (Longest strip when NEOPIXEL2_SEPARATE is disabled.)
2934 #define NEOPIXEL_IS_SEQUENTIAL // Sequential display for temperature change - LED by LED. Disable to change all LEDs at once.
2935 #define NEOPIXEL_BRIGHTNESS 255 // Initial brightness (0-255)
2936 #define NEOPIXEL_STARTUP_TEST // Cycle through colors at startup
2937
2938 // Support for second Adafruit NeoPixel LED driver controlled with M150 S1 ...
2939 // #define NEOPIXEL2_SEPARATE
2940 #if ENABLED(NEOPIXEL2_SEPARATE)
2941 #define NEOPIXEL2_PIXELS 15 // Number of LEDs in the second strip
2942 #define NEOPIXEL2_BRIGHTNESS 127 // Initial brightness (0-255)
2943 #define NEOPIXEL2_STARTUP_TEST // Cycle through colors at startup
2944 #else
2945 // #define NEOPIXEL2_INSERTIES // Default behavior is NeoPixel 2 in parallel
2946 #endif
2947
2948 // Use some of the NeoPixel LEDs for static (background) lighting
2949 // #define NEOPIXEL_BKGD_INDEX_FIRST 0 // Index of the first background LED
2950 // #define NEOPIXEL_BKGD_INDEX_LAST 5 // Index of the last background LED
2951 // #define NEOPIXEL_BKGD_COLOR { 255, 255, 255, 0 } // R, G, B, W
2952 // #define NEOPIXEL_BKGD_ALWAYS_ON // Keep the backlight on when other NeoPixels are off
2953 #endif

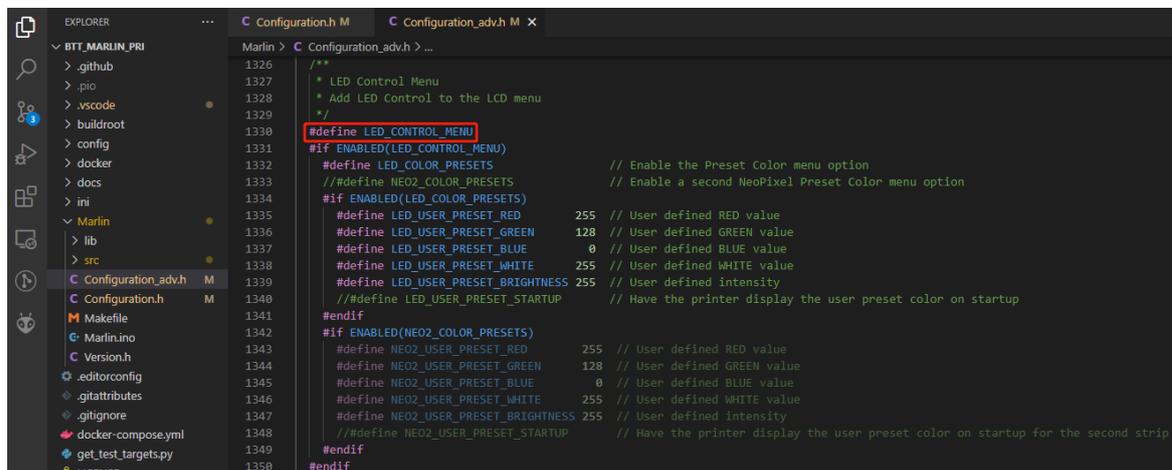
```

```
#define NEOPIXEL_LED // Enable Neopixel function
#define NEOPIXEL_TYPE NEO_GRB // Set up the type of lights
// #define NEOPIXEL_PIN 4 // Mask the PIN setting, use the correct signal line in
the motherboard pin file

#define NEOPIXEL_PIXELS 30 // Quantity of lights

#define NEOPIXEL_STARTUP_TEST // When the machine is turned on, it will
display three colors of red, green and blue in sequence, which is convenient for
testing.
```

If you enable LCD2004, 12864, mini12864 and other monitors, you can also enable the RGB control menu on the interface

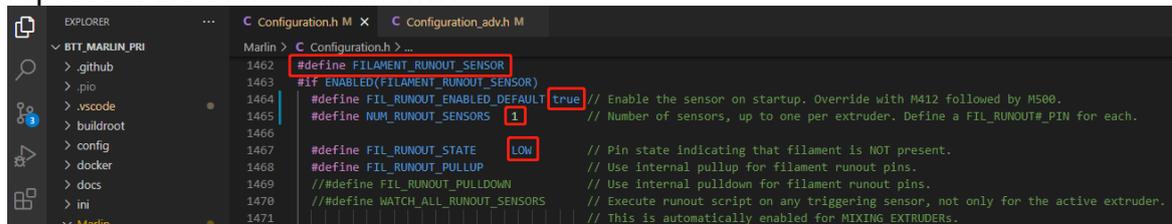


```
1326 /**
1327  * LED Control Menu
1328  * Add LED Control to the LCD menu
1329  */
1330 #define LED_CONTROL_MENU
1331 #if ENABLED(LED_CONTROL_MENU)
1332 #define LED_COLOR_PRESETS // Enable the Preset Color menu option
1333 // #define NEO2_COLOR_PRESETS // Enable a second NeoPixel Preset Color menu option
1334 #if ENABLED(LED_COLOR_PRESETS)
1335 #define LED_USER_PRESET_RED 255 // User defined RED value
1336 #define LED_USER_PRESET_GREEN 128 // User defined GREEN value
1337 #define LED_USER_PRESET_BLUE 0 // User defined BLUE value
1338 #define LED_USER_PRESET_WHITE 255 // User defined WHITE value
1339 #define LED_USER_PRESET_BRIGHTNESS 255 // User defined intensity
1340 // #define LED_USER_PRESET_STARTUP // Have the printer display the user preset color on startup
1341 #endif
1342 #if ENABLED(NEO2_COLOR_PRESETS)
1343 #define NEO2_USER_PRESET_RED 255 // User defined RED value
1344 #define NEO2_USER_PRESET_GREEN 128 // User defined GREEN value
1345 #define NEO2_USER_PRESET_BLUE 0 // User defined BLUE value
1346 #define NEO2_USER_PRESET_WHITE 255 // User defined WHITE value
1347 #define NEO2_USER_PRESET_BRIGHTNESS 255 // User defined intensity
1348 // #define NEO2_USER_PRESET_STARTUP // Have the printer display the user preset color on startup for the second strip
1349 #endif
1350 #endif
```

```
#define LED_CONTROL_MENU // Add a menu to control the LED color on the
screen
```

### 4.3.11 Filament Break Detection

Ordinary material break detection module is generally designed by a mechanical switch, the module gives the motherboard a constant high and low level to represent the state of filaments.



```
1462 #define FILAMENT_RUNOUT_SENSOR
1463 #if ENABLED(FILAMENT_RUNOUT_SENSOR)
1464 #define FIL_RUNOUT_ENABLED_DEFAULT true // Enable the sensor on startup. Override with M412 followed by M500.
1465 #define NUM_RUNOUT_SENSORS 1 // Number of sensors, up to one per extruder. Define a FIL_RUNOUT#_PIN for each.
1466
1467 #define FIL_RUNOUT_STATE LOW // Pin state indicating that filament is NOT present.
1468 #define FIL_RUNOUT_PULLUP // Use internal pullup for filament runout pins.
1469 // #define FIL_RUNOUT_PULLDOWN // Use internal pulldown for filament runout pins.
1470 // #define WATCH_ALL_RUNOUT_SENSORS // Execute runout script on any triggering sensor, not only for the active extruder.
1471 // This is automatically enabled for MIXING_EXTRUDERS.
```

```
#define FILAMENT_RUNOUT_SENSOR // Enable filament detection function
#define FIL_RUNOUT_ENABLED_DEFAULT true // true is on by default
#define NUM_RUNOUT_SENSORS 1 // Quantity of filaments detection sensors
#define FIL_RUNOUT_STATE LOW // The level state when the filaments are
```

## BIGTREETECH

---

abnormal, set up according to the actual situation of the module. If the module sends a low level when the consumables are abnormal, set it to LOW.

## 4.3.12 Smart Filament Sensor(SFS V1.0)

The Smart Filament Sensor will continuously send a jumping level signal when the filaments pass normally. When abnormal conditions such as material blockage/disconnection occur, the filaments cannot pass through the SFS normally, and the module cannot send a jumping signal to the mainboard, which thus knows that the filaments are abnormal.

```

1462 #define FILAMENT_RUNOUT_SENSOR
1463 #if ENABLED(FILAMENT_RUNOUT_SENSOR)
1464 #define FIL_RUNOUT_ENABLED_DEFAULT true // Enable the sensor on startup. Override with M412 followed by M500.
1465 #define NUM_RUNOUT_SENSORS 1 // Number of sensors, up to one per extruder. Define a FIL_RUNOUT#_PIN for each.
1466
1467 #define FIL_RUNOUT_STATE LOW // Pin state indicating that filament is NOT present.
1468 #define FIL_RUNOUT_PULLUP // Use internal pullup for filament runout pins.
1472
1473 // Override individually if the runout sensors vary...
1477
1478 //#define FIL_RUNOUT2_STATE LOW...
1481
1482 //#define FIL_RUNOUT3_STATE LOW...
1485
1486 //#define FIL_RUNOUT4_STATE LOW...
1489
1490 //#define FIL_RUNOUT5_STATE LOW...
1493
1494 //#define FIL_RUNOUT6_STATE LOW...
1497
1498 //#define FIL_RUNOUT7_STATE LOW...
1501
1502 //#define FIL_RUNOUT8_STATE LOW...
1505
1506 // Commands to execute on filament runout.
1507 // With multiple runout sensors use the %c placeholder for the current tool in commands (e.g., "M600 T%c")
1508 // NOTE: After 'M412 H1' the host handles filament runout and this script does not apply.
1509 #define FILAMENT_RUNOUT_SCRIPT "M600"
1510
1511 // After a runout is detected, continue printing this length of filament
1512 // before executing the runout script. Useful for a sensor at the end of
1513 // a feed tube. Requires 4 bytes SRAM per sensor, plus 4 bytes overhead.
1514 #define FILAMENT_RUNOUT_DISTANCE_MM 7
1515
1516 #ifndef FILAMENT_RUNOUT_DISTANCE_MM
1517 // Enable this option to use an encoder disc that toggles the runout pin
1518 // as the filament moves. (Be sure to set FILAMENT_RUNOUT_DISTANCE_MM
1519 // large enough to avoid false positives.)
1520 #define FILAMENT_MOTION_SENSOR
1521 #endif
1522 #endif

```

`#define FILAMENT_MOTION_SENSOR` // Set filament sensor to encoder type  
`#define FILAMENT_RUNOUT_DISTANCE_MM 7` // Set up the detection sensitivity. The recommended setting for SFS V1.0 is 7mm. If there is no level jump within 7mm of the filaments, it means that the filaments are abnormal.

Filaments detection also needs to set up the action after the abnormal suspension of the filaments through the following two places.

```

1907 #define NOZZLE_PARK_FEATURE
1908
1909 #if ENABLED(NOZZLE_PARK_FEATURE)
1910 // Specify a park position as { X, Y, Z raise }
1911 #define NOZZLE_PARK_POINT { (X_MIN_POS + 10), (Y_MAX_POS - 10), 20 }
1912 //#define NOZZLE_PARK_X_ONLY // X move only is required to park
1913 //#define NOZZLE_PARK_Y_ONLY // Y move only is required to park
1914 #define NOZZLE_PARK_Z_RAISE_MIN 2 // (mm) Always raise Z by at least this distance
1915 #define NOZZLE_PARK_XY_FEEDRATE 100 // (mm/s) X and Y axes feedrate (also used for delta Z axis)
1916 #define NOZZLE_PARK_Z_FEEDRATE 5 // (mm/s) Z axis feedrate (not used for delta printers)
1917 #endif

```

`#define NOZZLE_PARK_FEATURE` // Nozzle Pause Function  
`#define NOZZLE_PARK_POINT { (X_MIN_POS + 10), (Y_MAX_POS - 10), 20 }`  
 //Set the X, Y coordinates and the height of the Z axis when the nozzle is paused

```

2488 * Requirements:
2489 * - For Filament Change parking enable and configure NOZZLE_PARK_FEATURE.
2490 * - For user interaction enable an LCD display, HOST_PROMPT_SUPPORT, or EMERGENCY_PARSER.
2491 *
2492 * Enable PARK_HEAD_ON_PAUSE to add the G-code M125 Pause and Park.
2493 */
2494 #define ADVANCED_PAUSE_FEATURE
2495 #if ENABLED(ADVANCED_PAUSE_FEATURE)

```

`#define ADVANCED_PAUSE_FEATURE` // You can set parameters such as the length and speed of filament retraction during pause, and the length and speed of filament extrusion after continuing to print.

### 4.3.13 ESP3D

Just set the correct "SERIAL\_PORT" and "BAUDRATE" in Marlin. The serial port for communication between ESP8266 and Marlin on the motherboard is UART3, so set SERIAL\_PORT to 3.

```

97 // Choose the name from boards.h that matches your setup
98 #ifndef MOTHERBOARD
99   #define MOTHERBOARD BOARD_BTT_SKR_3
100 #endif
101
102 /**
103  * Select the serial port on the board to use for communication with the host.
104  * This allows the connection of wireless adapters (for instance) to non-default port pins.
105  * Serial port -1 is the USB emulated serial port, if available.
106  * Note: The first serial port (-1 or 0) will always be used by the Arduino bootloader.
107  *
108  * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]
109  */
110 #define SERIAL_PORT 1
111
112 /**
113  * Serial Port Baud Rate
114  * This is the default communication speed for all serial ports.
115  * Set the baud rate defaults for additional serial ports below.
116  *
117  * 250000 works in most cases, but you might try a lower speed if
118  * you commonly experience drop-outs during host printing.
119  * You may try up to 1000000 to speed up SD file transfer.
120  *
121  * :[2400, 9600, 19200, 38400, 57600, 115200, 250000, 500000, 1000000]
122  */
123 #define BAUDRATE 115200
124 // #define BAUD_RATE_GCODE // Enable G-code M575 to set the baud rate
125
126 /**
127  * Select a secondary serial port on the board to use for communication with the host.
128  * Currently Ethernet (-2) is only supported on Teensy 4.1 boards.
129  * :[-2, -1, 0, 1, 2, 3, 4, 5, 6, 7]
130  */
131 #define SERIAL_PORT_2 -1
132 // #define BAUDRATE_2 250000 // Enable to override BAUDRATE
133
134 /**
135  * Select a third serial port on the board to use for communication with the host.
136  * Currently only supported for AVR, DUE, LPC1768/9 and STM32/STM32F1
137  * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]
138  */
139 #define SERIAL_PORT_3 3
140 // #define BAUDRATE_3 250000 // Enable to override BAUDRATE
141

```

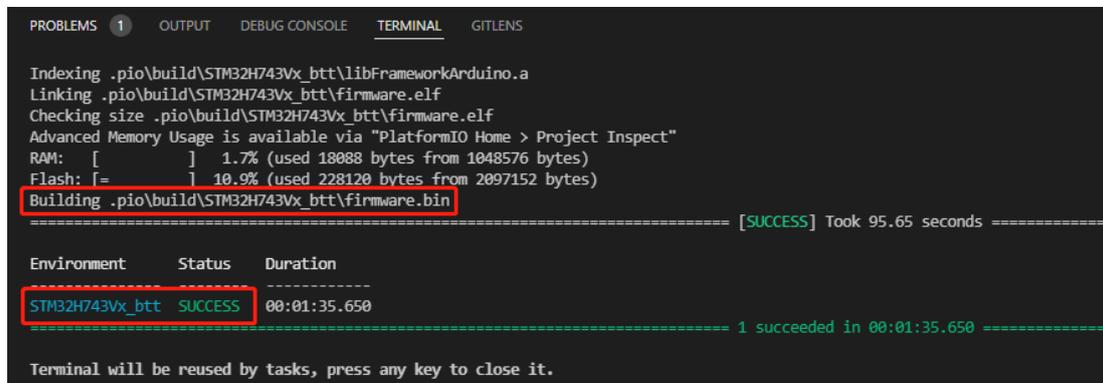
You can get the latest ESP3D firmware at <https://github.com/luc-github/ESP3D>, compile your own binary, rename it to "esp3d.bin" and copy it to the root directory of the SD card, plug it into the motherboard and then Reset, the bootloader in the motherboard will automatically update the esp3d.bin to the ESP8266, and the file will be renamed to "ESP3D.CUR" after the update is completed.

## 4.4 Compile the Firmware

1. Click "✓" in the status bar at the bottom to compile the firmware.



2. After the compilation is completed, the "firmware.bin" file will be generated, copy it to the SD card to update the firmware.



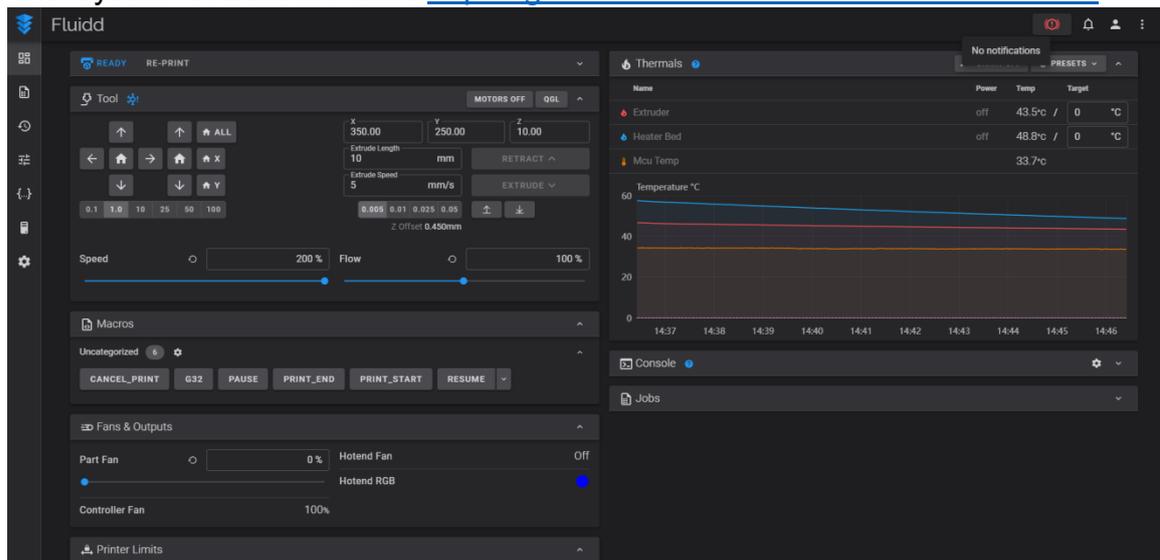
## 5. Klipper

### 5.1 Preparation

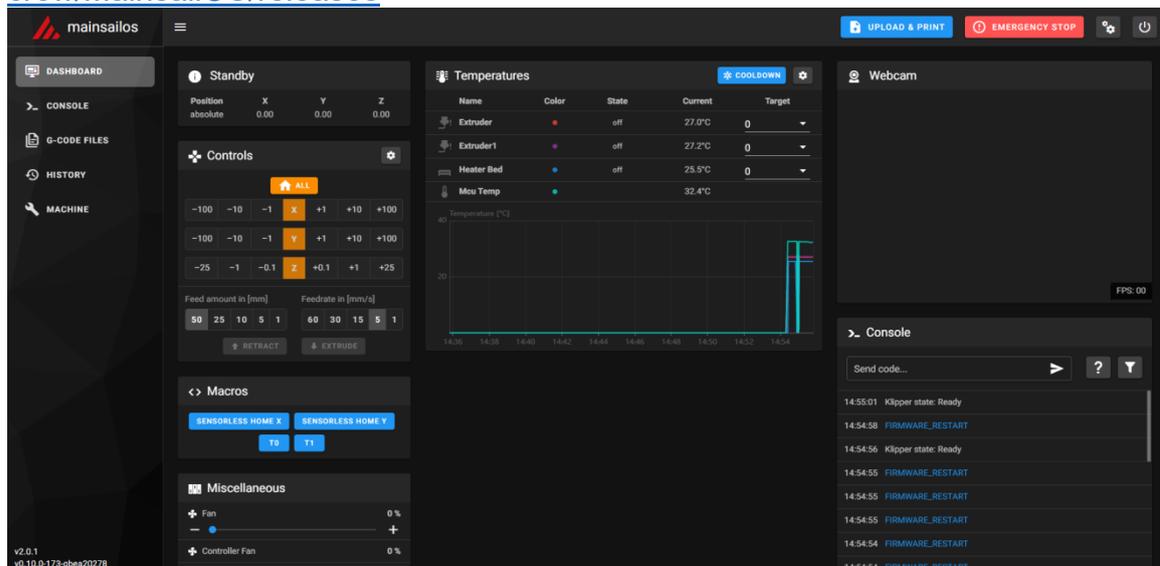
#### 5.1.1 Download System Image

Download the system image with your favorite WebUI built-in, currently, the mainstream ones are Fluidd, Mainsail, etc.

The System of Built-in Fluidd:<https://github.com/fluid-core/FluiddPI/releases>



The System of Built-in Mainsail:<https://github.com/mainsail-crew/MainsailOS/releases>



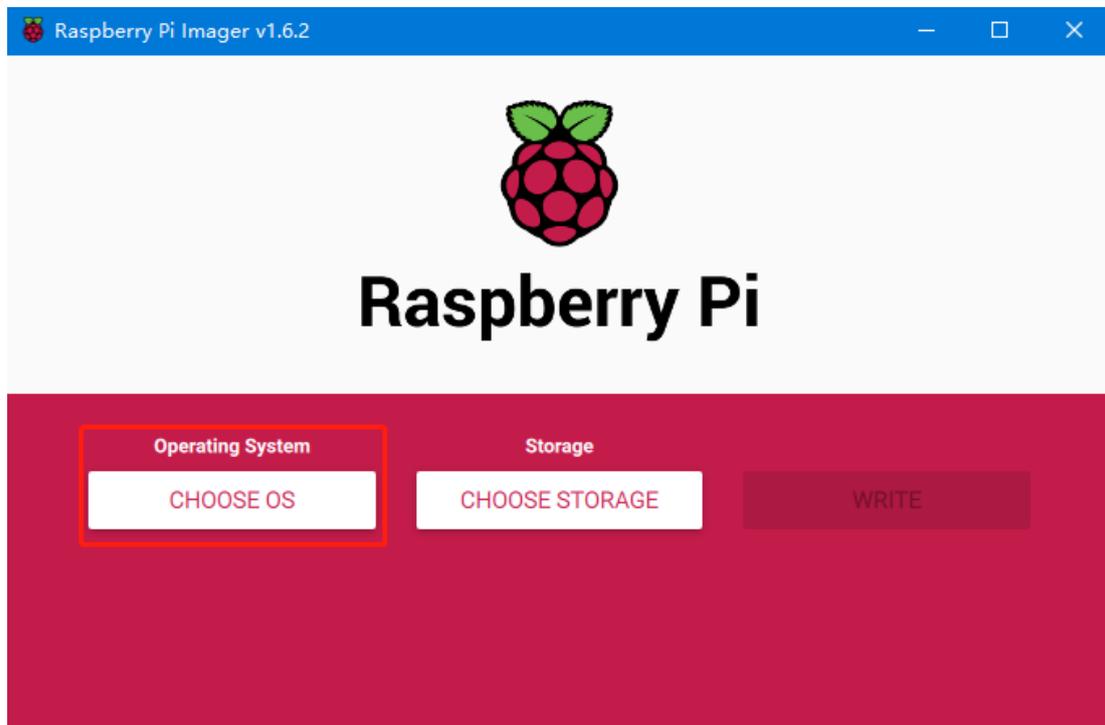
or refer to [Klipper official installation instructions](#) Use Octoprint

### 5.1.2 Download and Install Raspberry Pi Imager

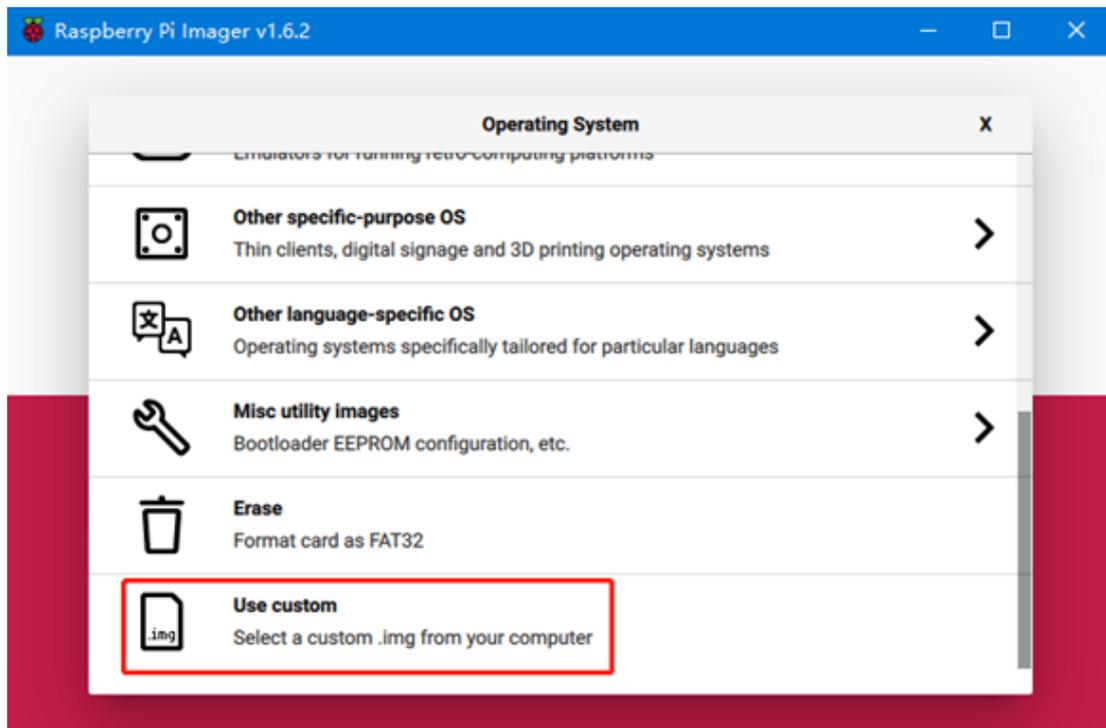
Download and install the official burning software for Raspberry Pi:  
<https://www.raspberrypi.com/software/>

### 5.2 Burn Image

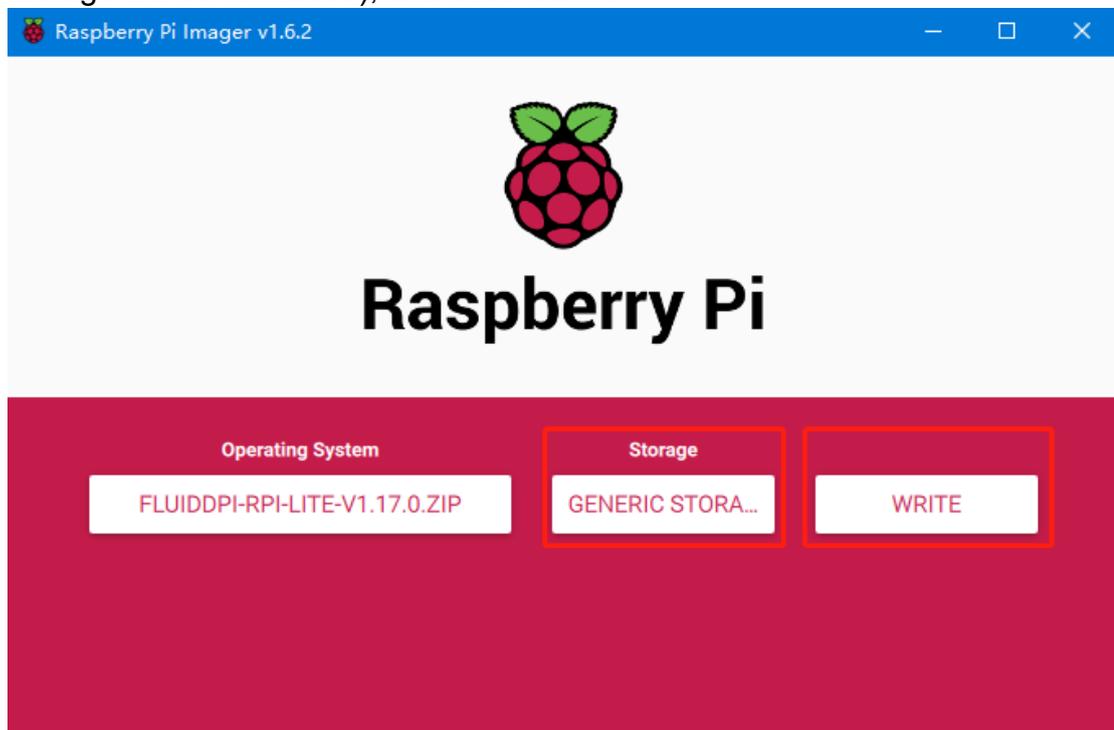
1. Insert the Micro SD card into the computer through the card reader.
2. Select System.



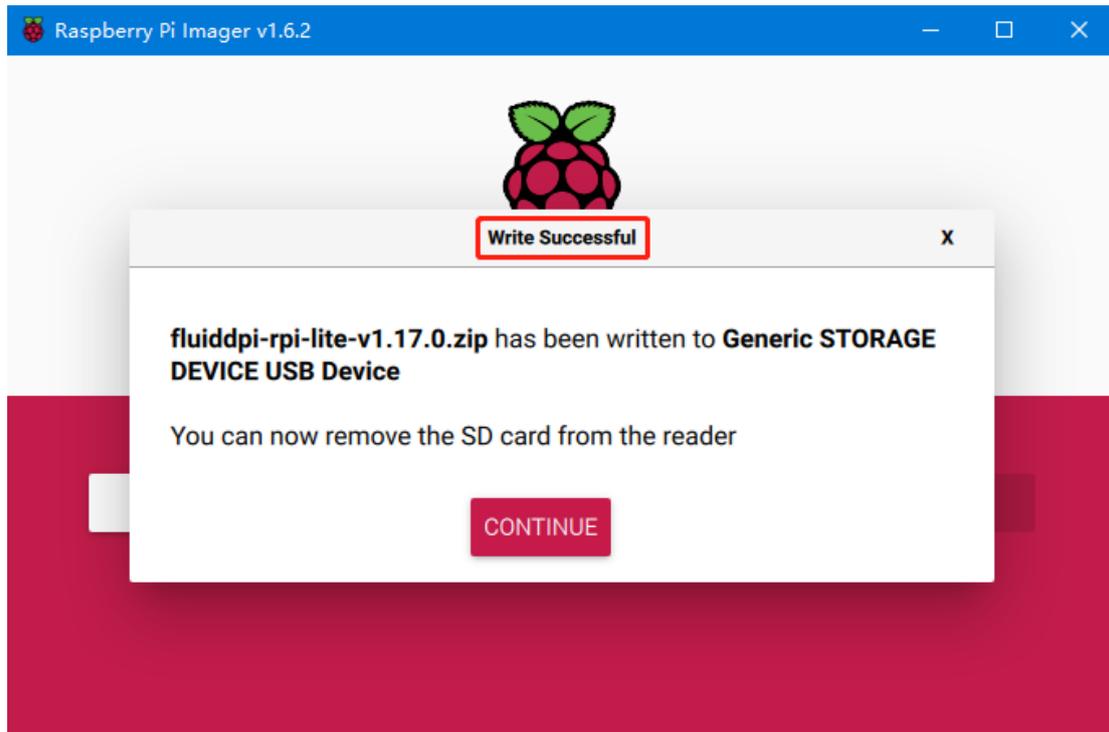
3. Select User Defined, and then select the image downloaded to your computer.



4. Select the SD card to be burned (burning the image will format the SD card, be careful not to select the wrong drive letter, otherwise the data on other storage will be formatted), and click "burn".



5. Wait for the burn to complete.



## 5.3 Set up WIFI

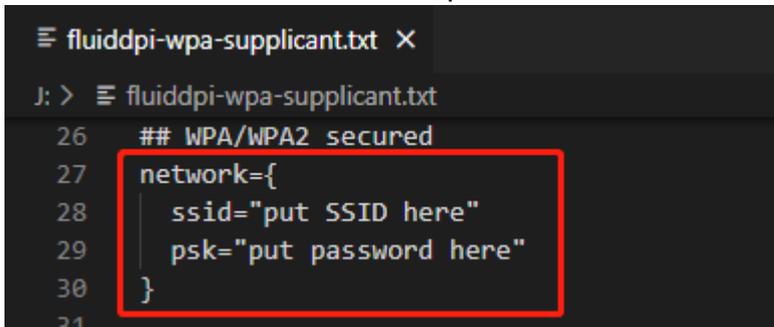
Note: You can skip this step if using a cable port instead of WIFI

1. Re-plug the card reader
2. Find the "fluiddpi-wpa-supPLICANT.txt" or "mainsail-wpa-supPLICANT.txt" file in the boot disk of the SD card and open it with VSCode (do not open it with the Notepad that comes with Windows)

boot (J:)

名称	修改日期	类型	大小
config.txt	2022/2/25 20:55	文本文档	3 KB
fluiddpi-wpa-supPLICANT.txt	2022/2/25 20:55	文本文档	2 KB
ssh	2022/2/25 20:54	文件	0 KB
issue.txt	2022/1/28 1:22	文本文档	1 KB

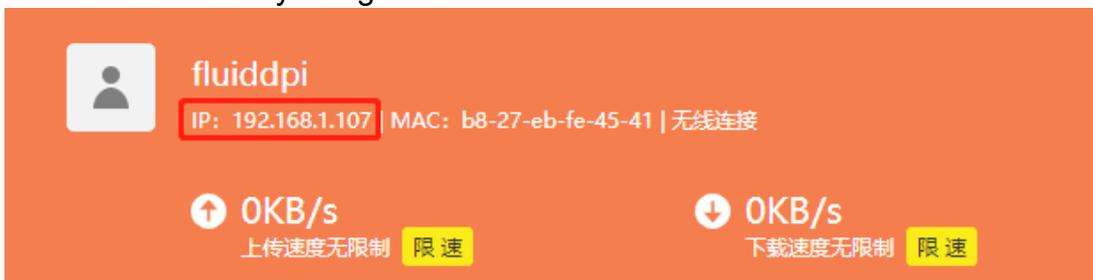
3. Delete the '#' character at the beginning of the four lines in the red box, then set the correct WIFI name and password and save it



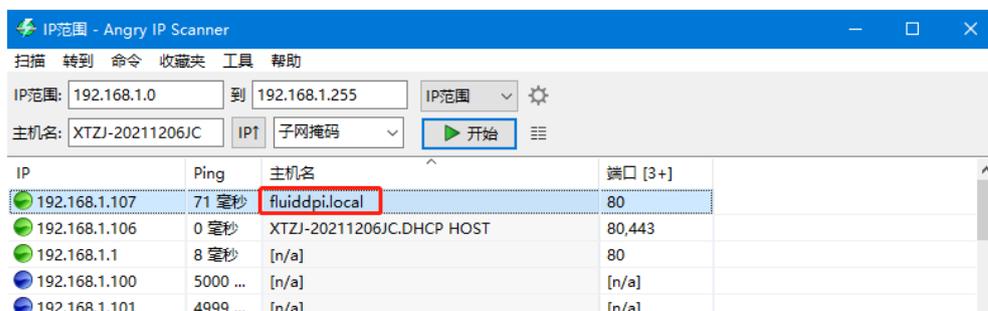
```
fluiddpi-wpa-supPLICANT.txt X
J: > fluiddpi-wpa-supPLICANT.txt
26  ## WPA/WPA2 secured
27  network={
28     ssid="put SSID here"
29     psk="put password here"
30  }
```

### 5.4 Connection of ssh software with Raspberry Pi

1. Install the ssh software Mobaxterm: <https://mobaxterm.mobatek.net/download-home-edition.html>
2. Insert the SD card into the Raspberry Pi, power on and wait for the system to start, about 1~2 minutes
3. After the Raspberry Pi is connected to WIFI or plugged in the Internet cable, it will be automatically assigned an IP
4. After the Raspberry Pi is connected to WIFI or plugged in the Internet cable, it will be automatically assigned an IP



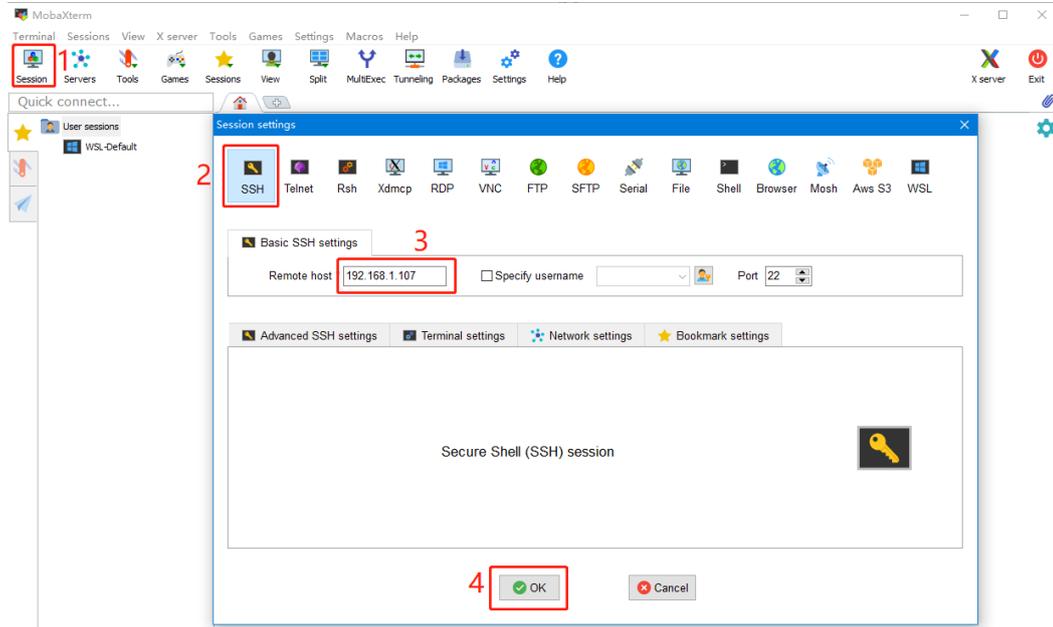
5. Or use [the https://angryip.org/](https://angryip.org/) tool to scan all IP addresses under the current local area network, and use the hostname to reorder to find the device with the hostname Fluidd or Mainsail, as shown in the following figure.



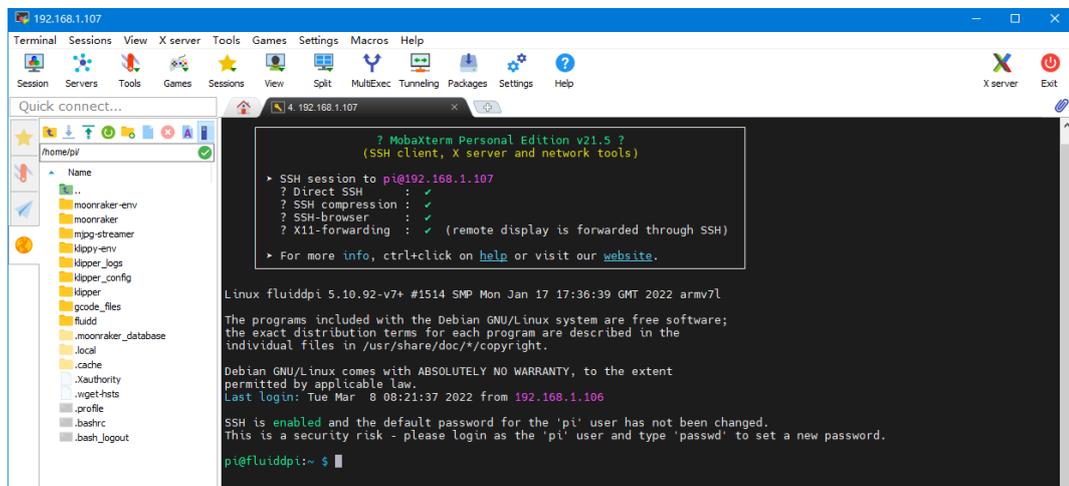
IP	Ping	主机名	端口 [3+]
192.168.1.107	71 毫秒	fluiddpi.local	80
192.168.1.106	0 毫秒	XTZJ-20211206JC.DHCP HOST	80,443
192.168.1.1	8 毫秒	[n/a]	80
192.168.1.100	5000 ...	[n/a]	[n/a]
192.168.1.101	4999 ...	[n/a]	[n/a]

6. Open the installed Mobaxterm software, click "Session", click "SSH" in the pop-up window, enter the IP address of the Raspberry Pi in the Remote host

column, and click "OK" (Note: the computer and the Raspberry Pi must be under the same local area network).



7. Enter the login name login as: pi, login password: raspberry, to enter the SSH terminal interface.



## 5.5 Compile the Firmware

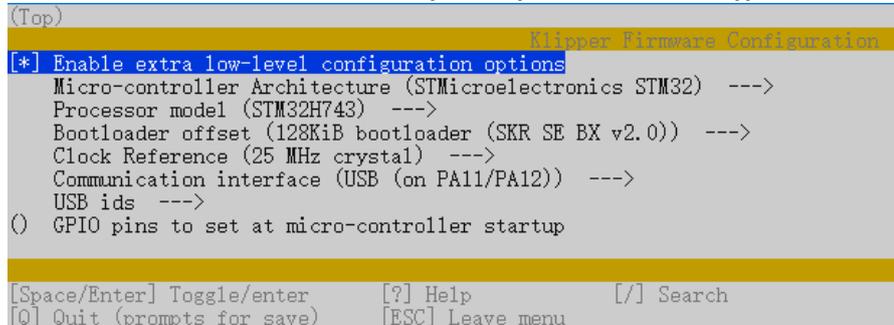
1. After connecting to the Raspberry Pi via ssh, enter at the command line:  
`cd ~/klipper/`

`make menuconfig`

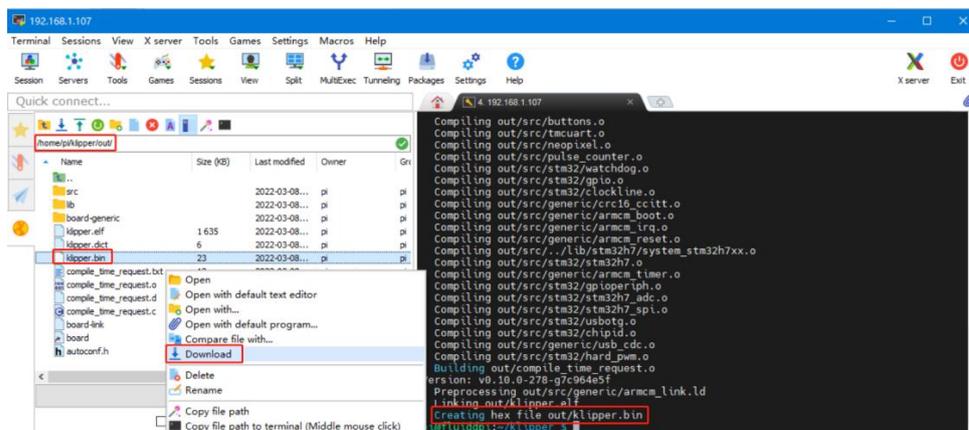
Compile the firmware with the following configuration (if the following options are not available, please update the Klipper firmware source to the latest version).

- \* **[\*] Enable extra low-level configuration options**
- \* **Micro-controller Architecture (STMicroelectronics STM32) --->**
- \* **Processor model (STM32H743) --->**

- \* **Bootloader offset (128KiB bootloader (SKR SE BX v2.0)) --->**
- \* **Clock Reference (25 MHz crystal) --->**
- \* **Communication interface (USB (on PA11/PA12)) --->**



2. After the configuration selection is completed, enter 'q' to exit the configuration interface, when asked whether to save the configuration, select "Yes"
3. Enter make to compile the firmware. When make is completed, the `klipper.bin` firmware we need will be generated in the home/pi/klipper/out folder of the Raspberry Pi, which can be downloaded directly to the computer on the left side of the ssh software.



4. Rename klipper.bin to "firmware.bin" and copy it to the SD card to update the firmware
5. Enter at the command line: `ls /dev/serial/by-id/` to query the ID of the motherboard to confirm whether the firmware is successfully burned. If the burning is successful, it will return a klipper device ID, as shown in the following figure

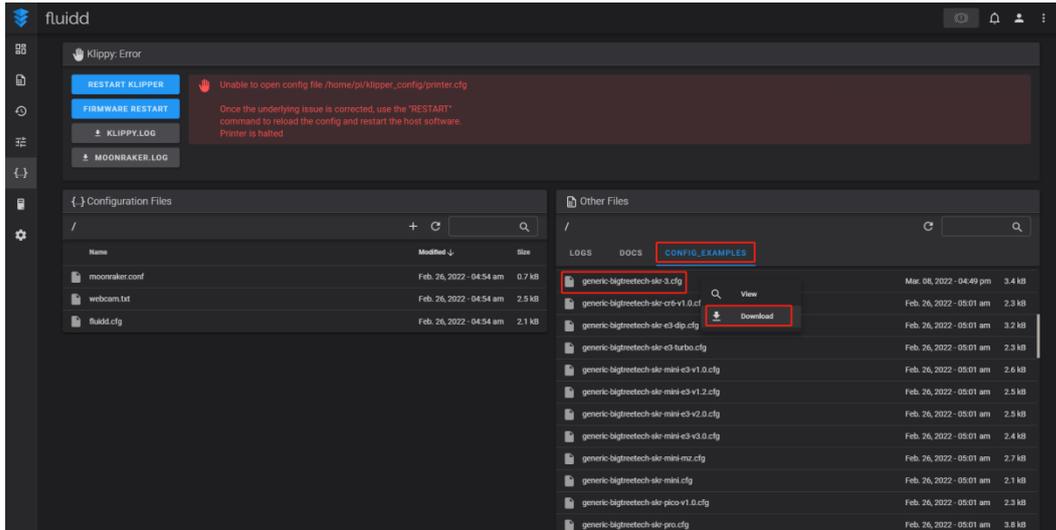


Copy and save this ID, this ID needs to be set in the configuration file.

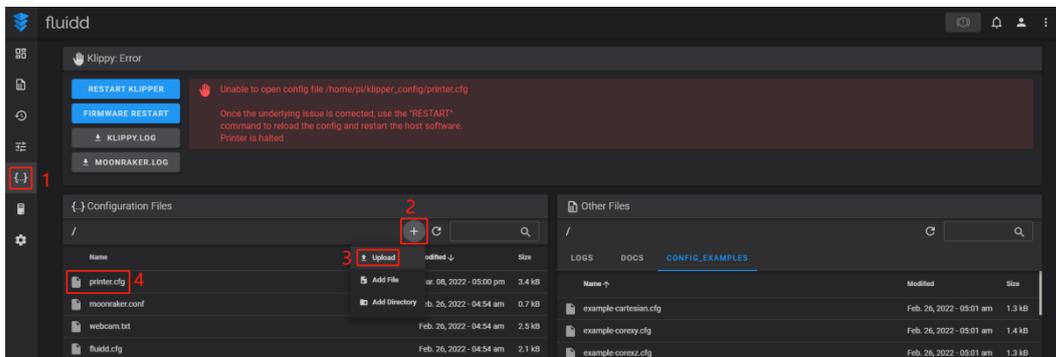
## 5.6 Configure Klipper

1. Enter the IP address of the Raspberry Pi in the computer's browser, and download the reference configuration of the motherboard from the path shown in the figure below. If you cannot find this file, please update the Klipper firmware source code to the latest version, or download it from GitHub

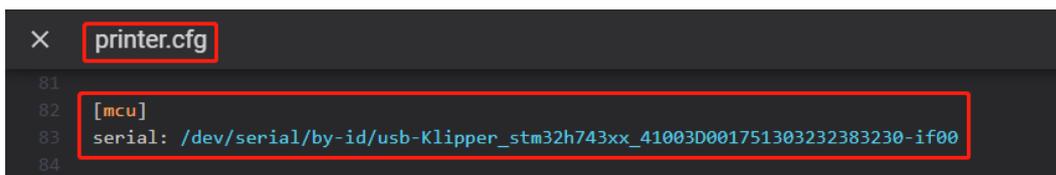
<https://github.com/bigtreetech/SKR-3>



2. Upload the motherboard configuration file to Configuration Files and rename it to "printer.cfg".



3. Modify the ID number in the configuration file to the actual ID of the motherboard



4. Follow the instructions at <https://www.klipper3d.org/Overview.html> to configure the specific features of the machine.

## 6. Firmware Update

### Micro SD card update

1. Make sure the Micro SD card has been formatted as FAT32 file system.
2. Rename the firmware compiled by yourself or downloaded from GitHub to "firmware.bin" (note: clarify the extension settings of the computer system, some users hide the extension, "firmware.bin" actually shows "firmware" )
3. Copy "firmware.bin" to the root directory of the Micro SD card.
4. Insert the Micro SD card into the card slot of the motherboard, power on the motherboard again, and the motherboard's bootloader will automatically update the firmware.
5. During the firmware update process, the status indicator on the upper right corner of the motherboard will start to flash.
6. When the status indicator stops flashing and the file name in the Micro SD card is renamed to "FIRMWARE.CUR", it means the firmware update is successful.

## 7. Cautions

1. When the PT1000 is not used, the jumper cap cannot be inserted on it, otherwise the 100K NTC cannot be used normally.
2. The current of the hotbed connected to the mainboard must be less than or equal to 10A. If you want to use a high-power hotbed, it is recommended to choose a hotbed powered by 24V, and use 24V to power the mainboard.
3. Pay attention to the power supply selection of the NC fan, the jump cap must be set, so that the fan can work normally.
4. Pay attention to the setting of the USB port switch. When there is no response when plugged into the computer, make sure that the double-pole double-throw switch is in the USB mode of the pop-up state.
5. The mainboard adopts a non-elastic card slot, and the stroke is much less than that of the self-elastic card slot. When the user inserts the card, the action must be light and slow. Do not insert the card vigorously, and the damage caused will not be accepted by our company.

## 8. FAQ

**Q: The maximum current of the hotbed, heater cartridge, and fan ports:**

A: The maximum output current of the heated bed port: 10A, the peak value is 11A.

Heater cartridge port maximum output current: 5.5A, peak 6A.

The maximum output current of fan interface: 1A, peak 1.5A.

The total current of the heater cartridge + driver + fan needs to be less than 10A.

**Q: SD card cannot update firmware:**

A: Make sure that the SD card has been formatted as a FAT32 file system, and make sure the firmware name is "firmware.bin". Some users' computers have set "Hide known extensions", and "firmware.bin" is displayed. The file name is actually "firmware.bin.bin".