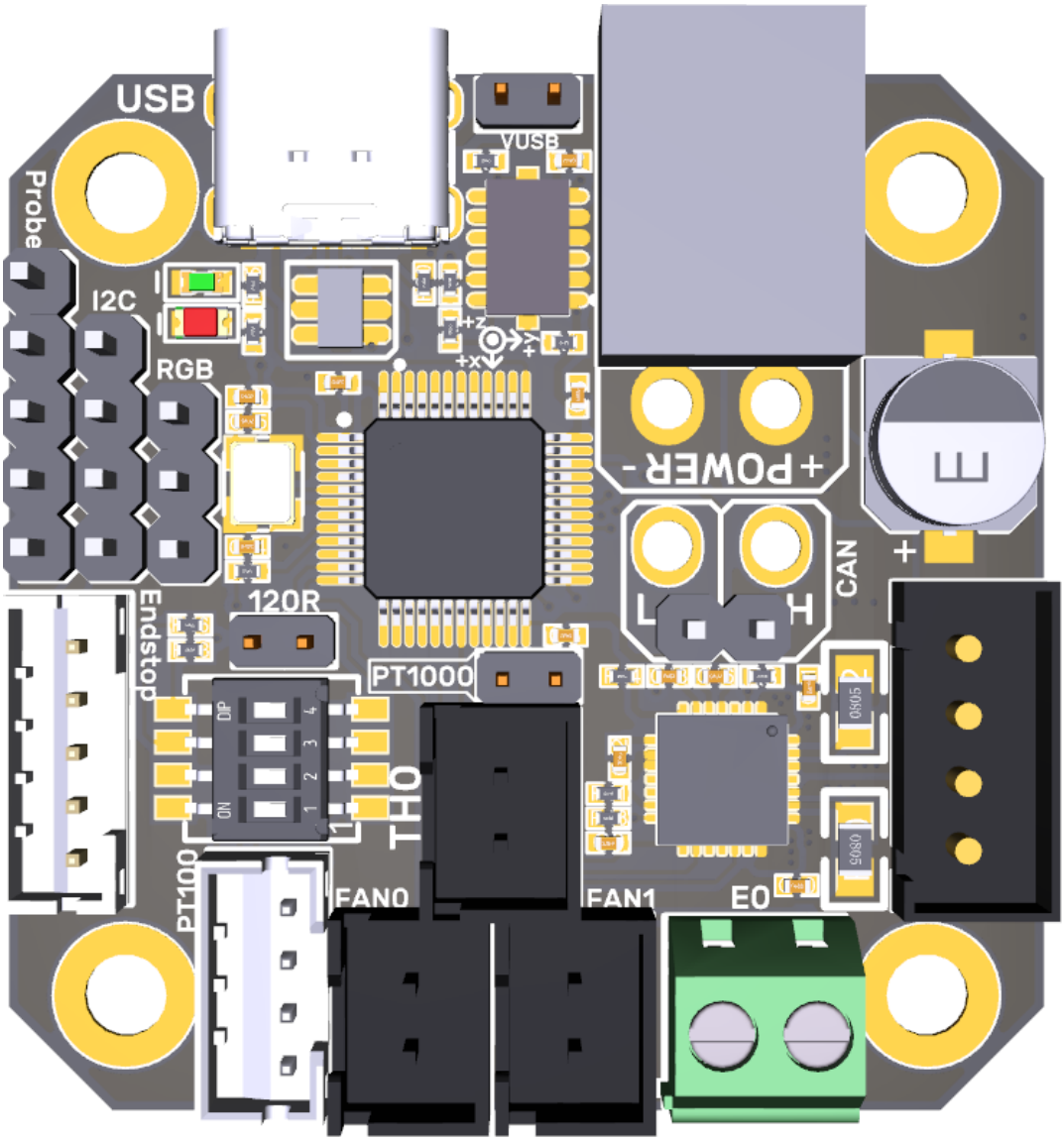
BIGTREETECH EBB42 CAN V1.0

User Manual



Contents

Revisions

Version	Statements	Date
01.00	First Draft	2022/04/18

Ⅰ. Brief Instruction

BIGTREETECH EBB42 CAN V1.0 is a nozzle adapter board specially designed for the 42 extruder stepper motor, which is launched by the 3D printing team Shenzhen Big Tree Technology Co., Ltd. It can communicate via USB port, or CAN BUS, greatly simplifying wirings.

1.1 Features

1. With BOOT and RESET buttons reserved, users can update the firmware via DFU mode by USB.

2. Added protection circuit on the thermistor avoids burning the main control chip caused by leakage current from the heating rod.

3. User can select the thermistor's pull-up resistors values through jumper wire, so as to support PT1000 (2.2K pull-up resistors), which makes it convenient for DIY.

4. Connect the USB with a jumper cap to get it power on, which effectively isolates the main control board DC-DC from USB 5V.

5. Reserved I2C interface allows for filament broke and clogged detection, and supports other DIY functions.

6. Added anti-flyback diodes on the heating rod and fans' ports effectively protect the MOS tube from being burned due to reverse voltage.

7. Anti-reverse connection protection on the power interface prevents users from burning the motherboard when mistakenly connecting the reverse power line during DIY.

8. Onboard MAX31865 supports selecting 2-wire or 4-wire of PT100/ PT1000.

9. Supports communication via CAN or USB. The terminal resistor 120R of CAN can be selected through the jumper cap, and it reserves CAN expansion interfaces.

10. Equipped ESD protection chip on the USB port prevents the main control board from being broken down by the static electricity of the USB port.

11. The adaptor board is equipped with terminals, female reeds, double-way studs and screws, which are required accessories for DIY, greatly meeting the DIY needs of customers.

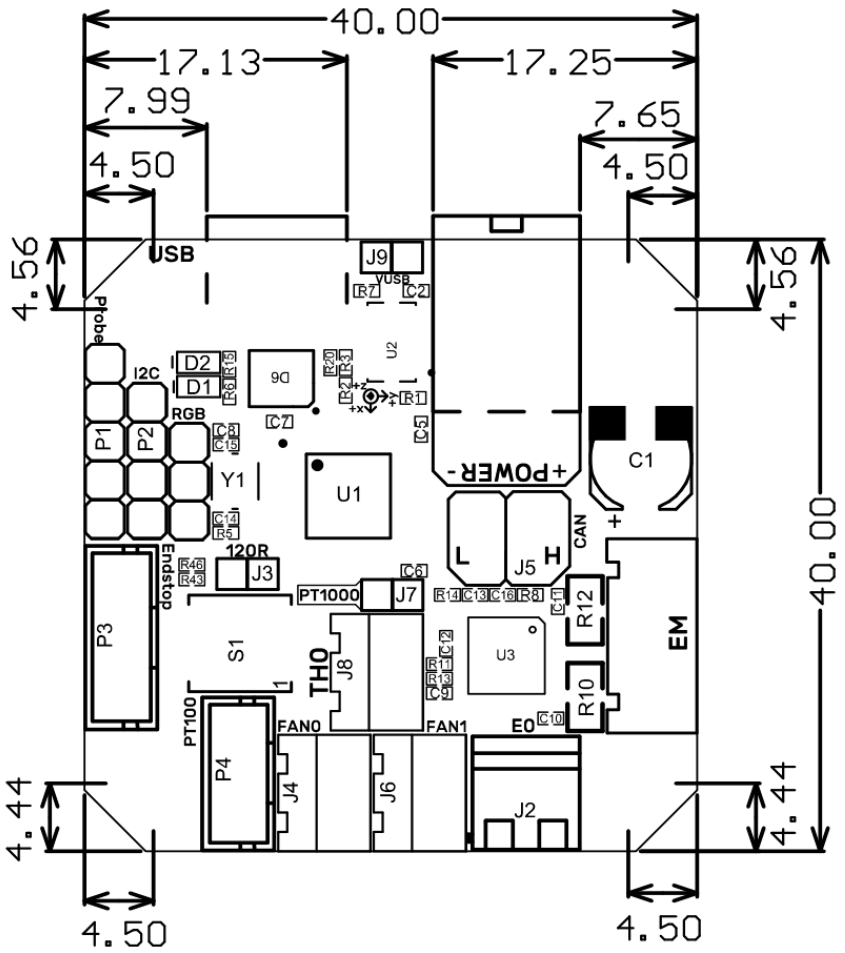
1.2 Parameters

1. External Dimensions: 40mm x 40mm. For further details please read: **BIGTREETECH EBB42 CAN V1.0-SIZE.pdf**
2. Installation Dimensions: Hole spacing 31mm x 31mm, M3 screw hole x 4
3. Microprocessor: ARM Cortex-M0 STM32F072C8T6 48MHz
4. Input Voltage: DC12V-DC24V 9A
5. Logic Voltage: DC 3.3V
6. Heating Interface: Heating Rod (E0), maximum output current: 5A
7. Onboard Sensor: ADXL345
8. Fan Interfaces: Two CNC Fans (FAN0, FAN1)
9. Maximum Output Current of Fan Interface: 1A, peak value 1.5A
10. Expansion Interfaces: EndStop, I2C, Probe, RGB, PT100/PT1000, USB Interface, CAN Interface
11. Motor Drive: Onboard TMC2209
12. Driver Working Mode: UART
13. Stepper Motor Interface: EM
14. Temperature Sensor Interface(Optional): 1 channel 100K NTC or PT1000 (TH0), 1 channel PT100/PT1000
15. USB Communication Interface: USB-Type-C
16. DCDC 5V Maximum Output Current: 1A

1.3 Firmware

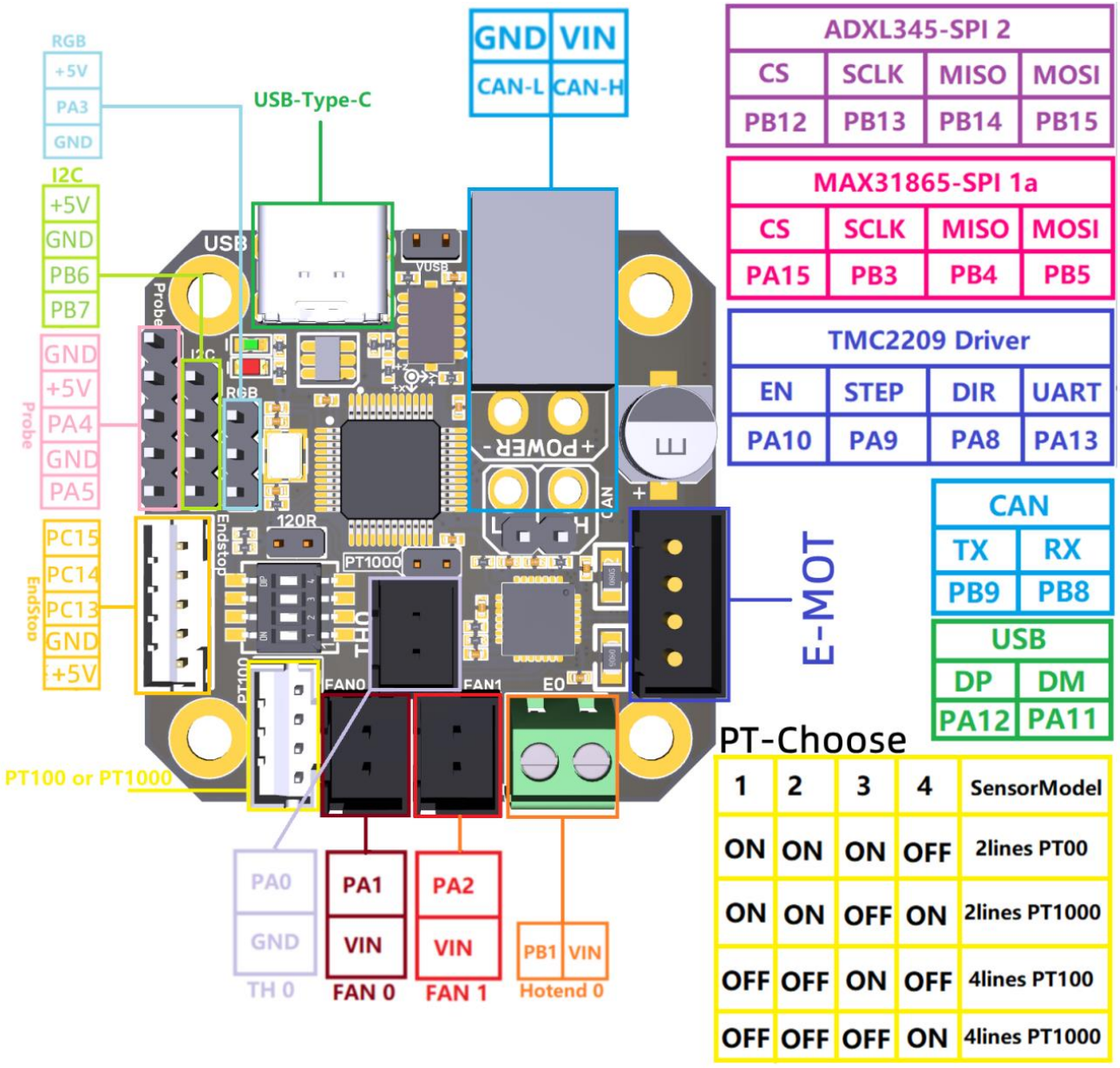
BTT EEB42 CAN V1.0 only supports Klipper at the present.

1.4 Size Diagram



II. Peripheral interface

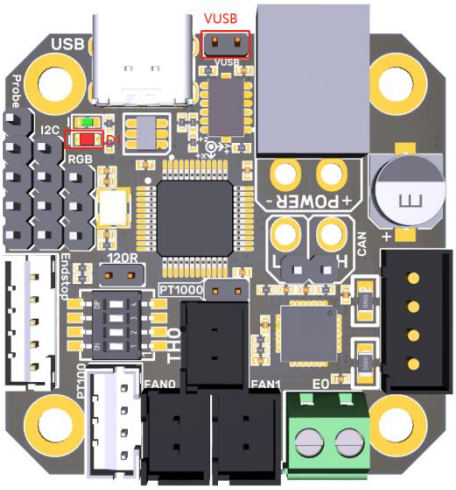
2.1 Pin



III. Introduction of Interface

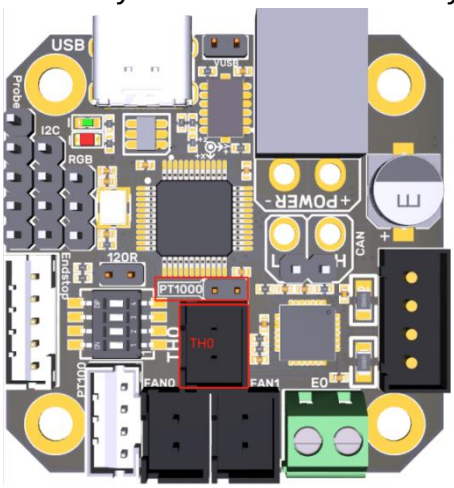
3.1 USB Power Supply

D1 RGB light will be on when the control board is powers on, which shows supplying normal power. VUSB in the middle of the board is the selection terminal for power. Only when USB supplies power to the board, or the board supplies power through USB, users need to short circuit the VUSB by jump cap.



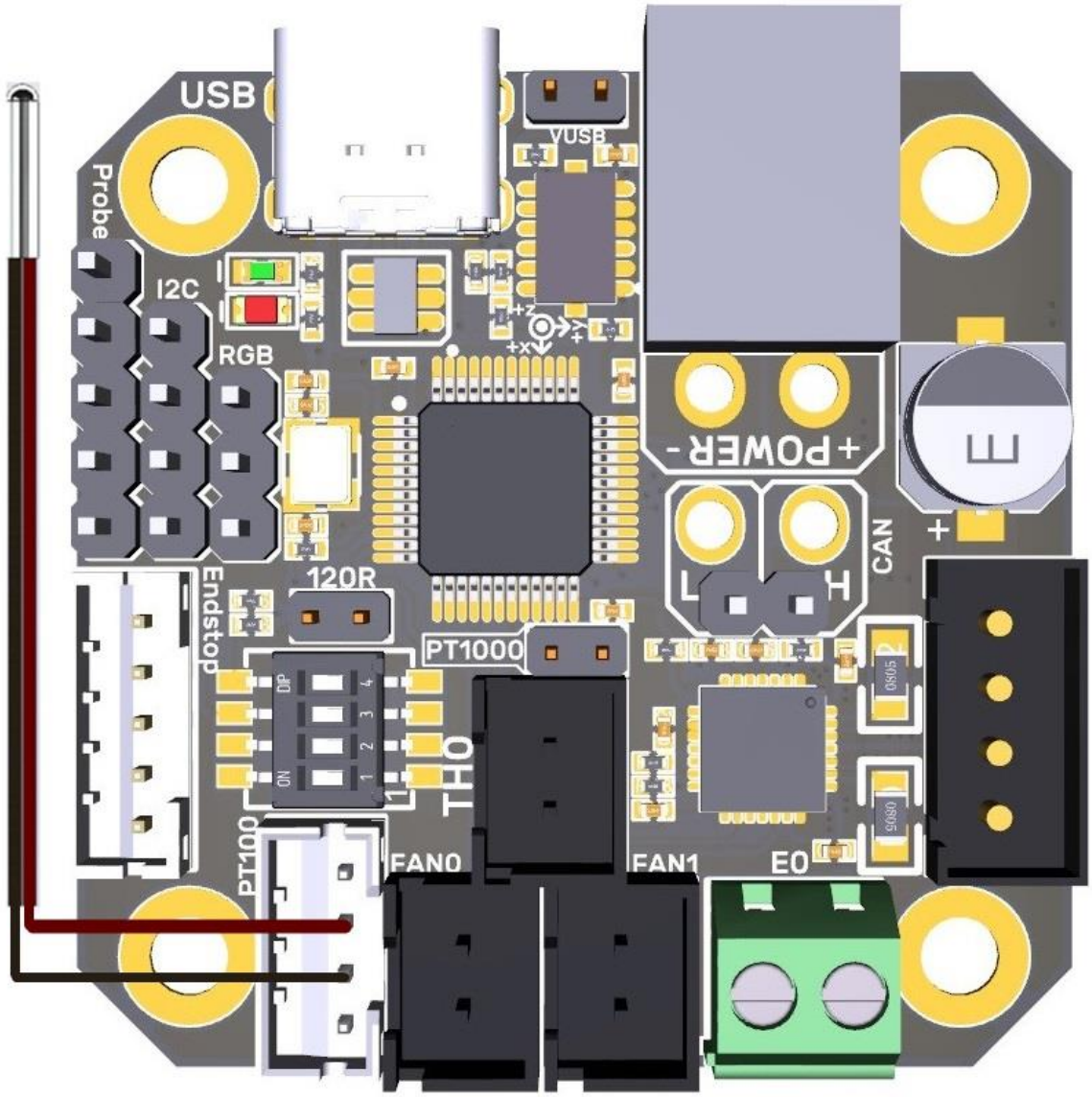
3.2 100K NTC or PT1000 Settings

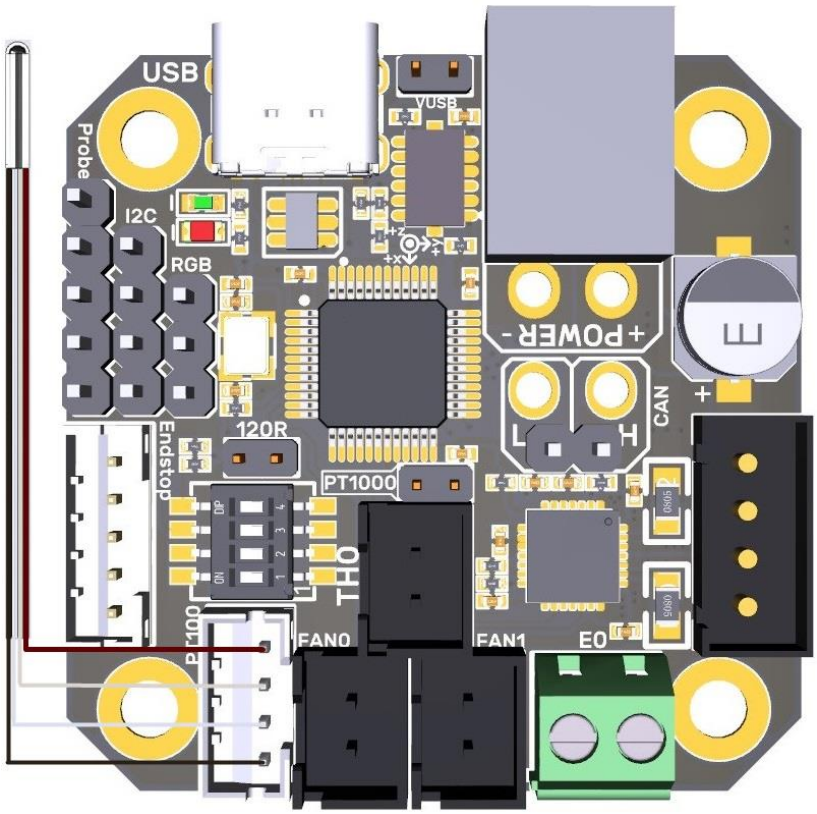
1. Version without 31865: No need to insert a jumper cap when using a 100K NTC thermistor, and TH0's pull-up resistor value is 4.7K. When using PT1000, you need to use the jumper cap to short the two pins circled in red as shown below. At this time, TH0's pull-up resistor value is 2.2K (Notice: the temperature accuracy that's read out this way will be less accurate than that of MAX31865).



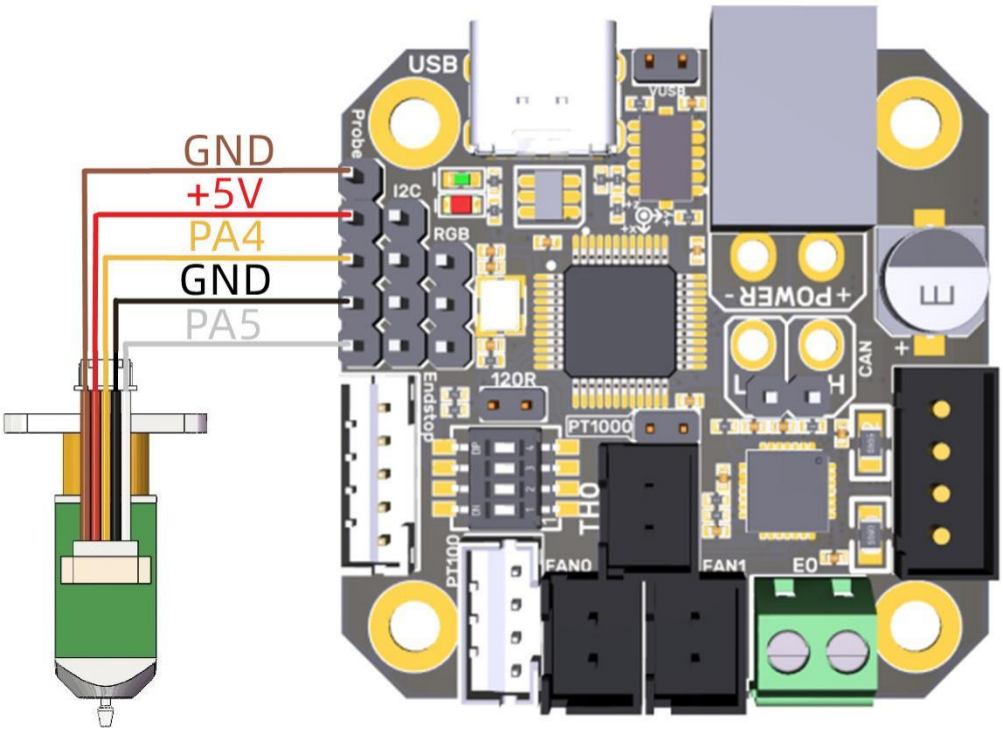
2. Version with 31865: Select PT100/PT1000 by DIP switch, two-line or four-wire.

1	2	3	4	Sensor Model
ON	ON	ON	OFF	Two lines PT100
ON	ON	OFF	ON	Two linesPT1000
OFF	OFF	ON	OFF	Four-wire PT100
OFF	OFF	OFF	ON	Four-wire PT1000

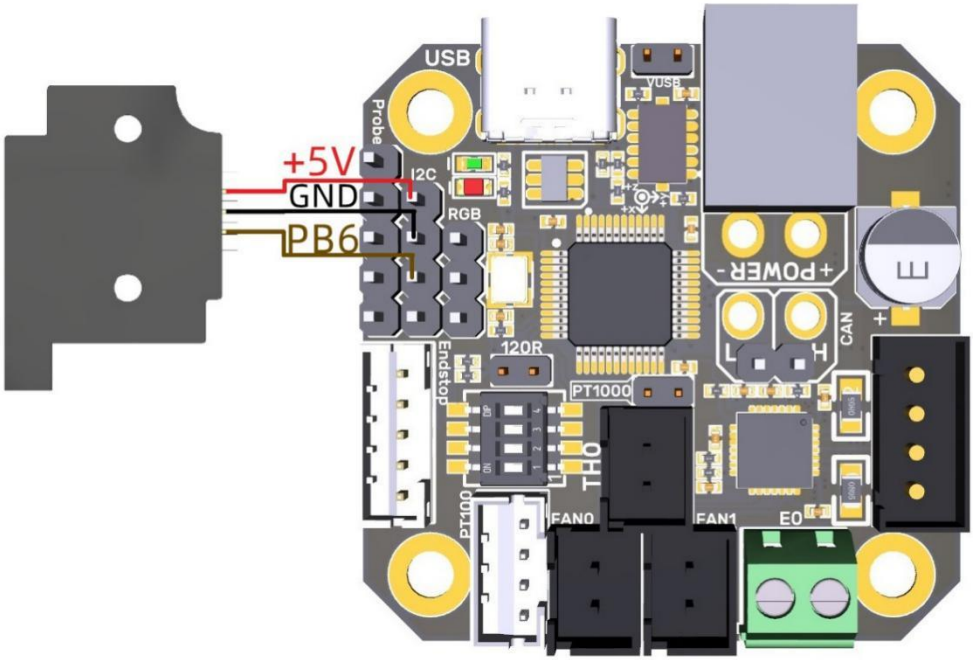




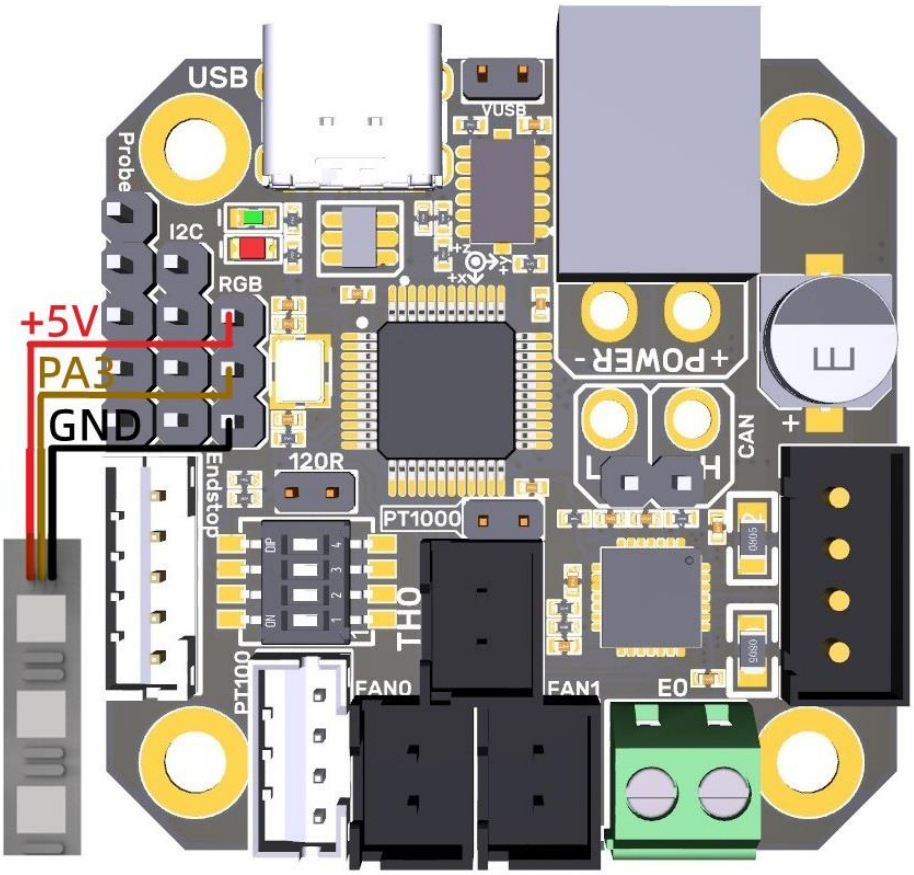
3.3 Connection with BL-Touch



3.4 Connection with Filament Broke Detection



3.5 Connection with RGB

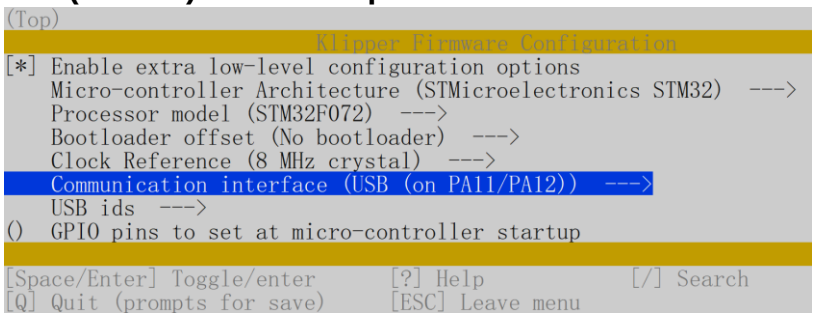


IV. Klipper

4.1 Compile Firmware

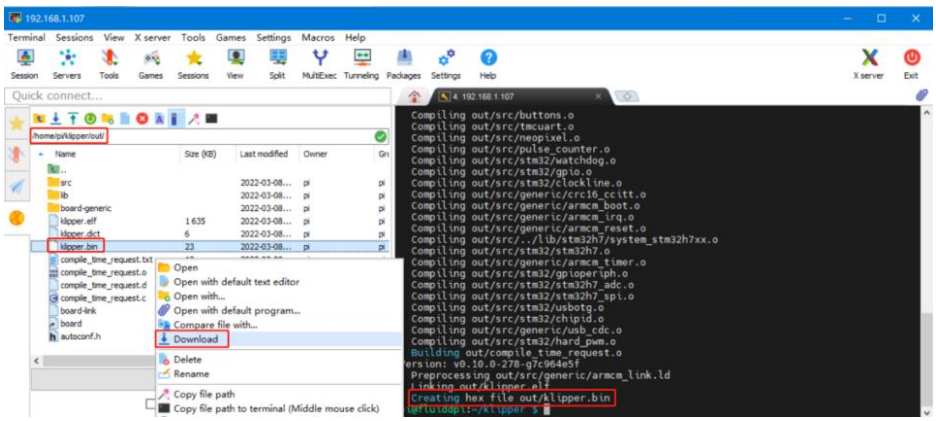
1. After connecting to the Raspberry Pi via ssh, type the below sentences at the command line:
   cd ~/klipper/
   make menuconfig
   Compile the firmware with the following configuration (if the following options are not available, please update the Klipper firmware source to the latest version)

[*] Enable extra low-level configuration options
Micro-controller Architecture (STMicroelectronics STM32) --->
Processor model (STM32F072) --->
Bootloader offset (No bootloader) --->
Clock Reference (8 MHz crystal) --->
If using USB communication over Type-C
Communication interface (USB (on PA11/PA12)) --->
If using CANBus communication
Communication interface (CAN bus (on PB8/PB9)) --->
(250000) CAN bus speed



2. When completed the configuration selection, type 'q' to exit the configuration interface. Select "Yes" when asked whether to save the configuration.

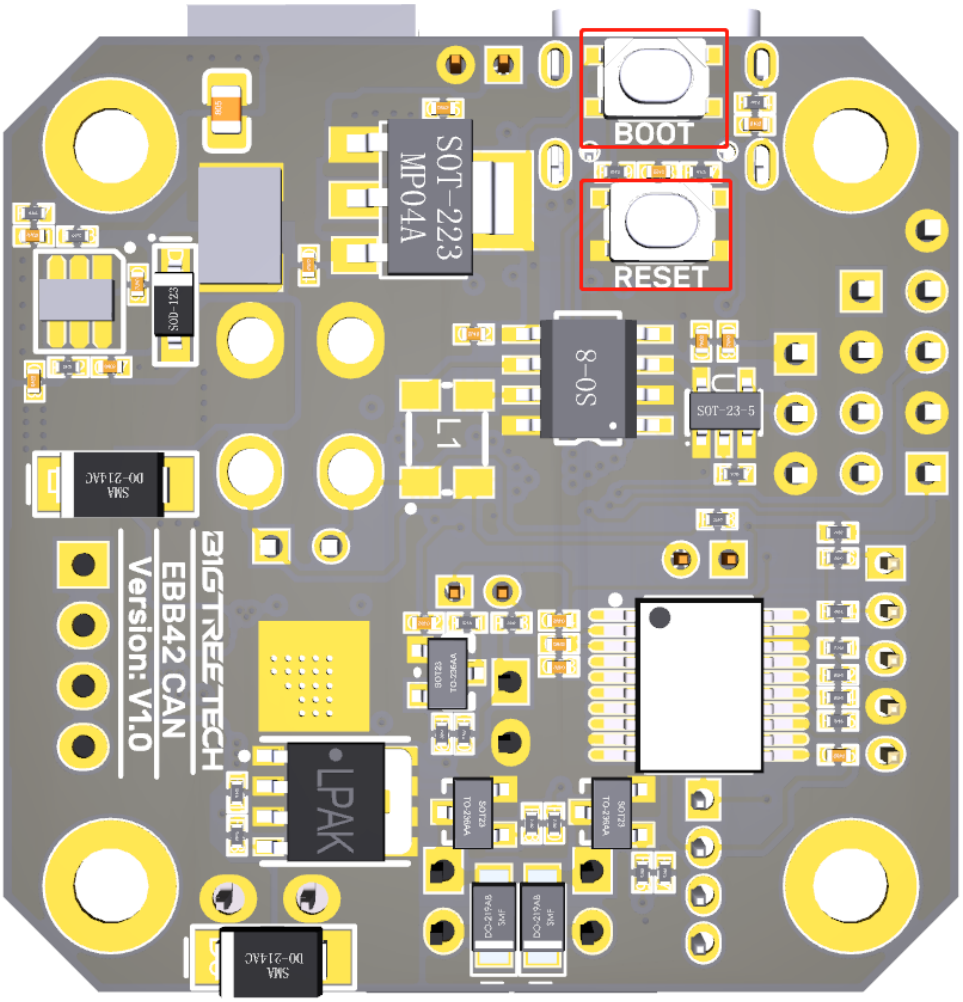
3. Enter **make** to compile firmware. The 'klipper.bin' firmware that we need will be generated on the **home/pi/kliiper/out** folder of the Raspberry Pi only once completed **make**. On the left side of ssh software, user can download the 'klipper.bin' firmware directly to the computer.



4.2 Update Firmware

Update firmware via Raspberry Pi (You can also update via STM32CubeProgrammer software after the board is plugged into the computer and entering DFU mode).

1. Press on the Boot button, then click on the Reset button to enter DFU mode.



2. Type lsusb in the ssh terminal command line to query the ID of the DFU device.

```
pi@fluiddpi:~ $ lsusb
Bus 001 Device 005: ID 0483:df11 STMicroelectronics STM Device in DFU Mode
Bus 001 Device 004: ID 1d50:606f OpenMoko, Inc. Geschwister Schneider CAN adapter
Bus 001 Device 003: ID 0424:ec00 Microchip Technology, Inc. (formerly SMSC) SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Microchip Technology, Inc. (formerly SMSC) SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
pi@fluiddpi:~ $
```

3. Type make flash FLASH_DEVICE=0483:df11 to download the firmware (Notice: Replace 0483:df11 with the actual ID of the DFU device queried in the previous step).

```
pi@fluiddpi:~/klipper $ make flash FLASH_DEVICE=0483:df11
  Building hid-flash
    hid-flash requires libusb-1.0, please install with:
    sudo apt-get install libusb-1.0
  Flashing out/klipper.bin to 0483:df11
sudo dfu-util -d ,0483:df11 -R -a 0 -s 0x8000000:leave -D out/klipper.bin

[sudo] password for pi:
dfu-util 0.9

Copyright 2005-2009 Weston Schmidt, Harald Welte and OpenMoko Inc.
Copyright 2010-2016 Tormod Volden and Stefan Schmidt
This program is Free Software and has ABSOLUTELY NO WARRANTY
Please report bugs to http://sourceforge.net/p/dfu-util/tickets/

dfu-util: Invalid DFU suffix signature
dfu-util: A valid DFU suffix will be required in a future dfu-util release!!!
Opening DFU capable USB device...
ID 0483:df11
Run-time device DFU version 011a
Claiming USB DFU Interface...
Setting Alternate Setting #0 ...
Determining device status: state = dfuERROR, status = 10
dfuERROR, clearing status
Determining device status: state = dfuIDLE, status = 0
dfuIDLE, continuing
DFU mode device DFU version 011a
Device returned transfer size 2048
DfuSe interface name: "Internal Flash  "
Downloading to address = 0x08000000, size = 21728
Download        [=========================] 100%        21728 bytes
Download done.
File downloaded successfully
Transitioning to dfuMANIFEST state
dfu-util: can't detach
Resetting USB to switch back to runtime mode
pi@fluiddpi:~/klipper $
```

4. Type ls /dev/serial/by-id/ to query about the serial ID of the device when finished downloading the firmware (The serial ID will only exist when communicating via USB. Ignore this step when communicating via CAN Bus).

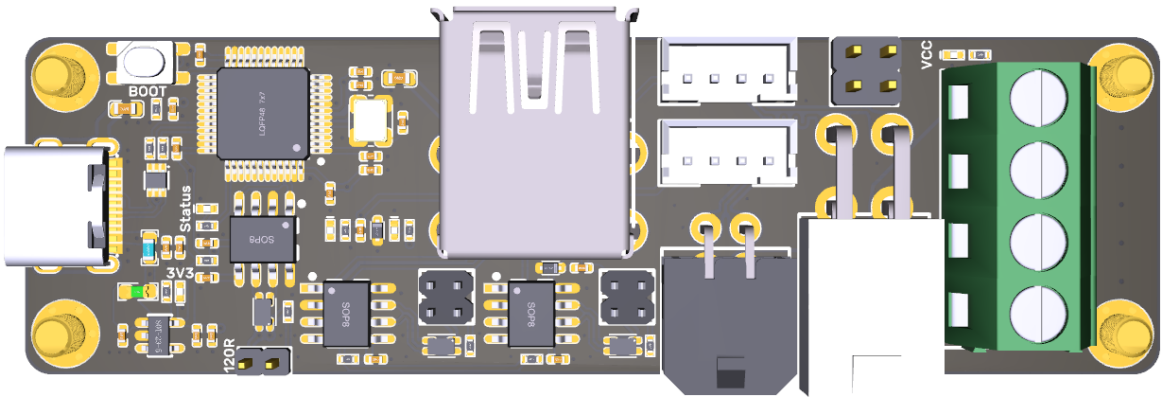
```
pi@fluiddpi:~/klipper $ ls /dev/serial/by-id/
usb-Klipper_stm32f072xb_28002D001557434338313020-if00
pi@fluiddpi:~/klipper $
```

5. After downloading the firmware for the first time, there's no need to press on Boot and Reset button again to enter DFU mode when updating the firmware again. User can type `make flash FLASH_DEVICE= /dev/serial/by-id/usb-Klipper_stm32f072xb_28002D001557434338313020-if00` to download the firmware(Notice: Replace `/dev/serial/by-id/xxx` with actual ID queried in the previous step)

```
pi@fluiddpi:~/klipper $ ls /dev/serial/by-id/
usb-Klipper_stm32f072xb_28002D001557434338313020-if00
pi@fluiddpi:~/klipper $ make flash FLASH_DEVICE=/dev/serial/by-id/usb-Klipper_stm32f072xb_28002D001557434338313020-if00
  Building hid-flash
    hid-flash requires libusb-1.0, please install with:
    sudo apt-get install libusb-1.0
  Flashing out/klipper.bin to /dev/serial/by-id/usb-Klipper_stm32f072xb_28002D001557434338313020-if00
Entering bootloader on /dev/serial/by-id/usb-Klipper_stm32f072xb_28002D001557434338313020-if00
Device reconnect on /sys/devices/platform/soc/3f980000.usb/usb1/1-1/1-1.4/1-1.4:1.0
sudo dfu-util -p 1-1.4 -R -a 0 -s 0x8000000:leave -D out/klipper.bin

dfu-util 0.9

Copyright 2005-2009 Weston Schmidt, Harald Welte and OpenMoko Inc.
Copyright 2010-2016 Tormod Volden and Stefan Schmidt
This program is Free Software and has ABSOLUTELY NO WARRANTY
Please report bugs to http://sourceforge.net/p/dfu-util/tickets/

dfu-util: Invalid DFU suffix signature
dfu-util: A valid DFU suffix will be required in a future dfu-util release!!!
Opening DFU capable USB device...
ID 0483:df11
Run-time device DFU version 011a
Claiming USB DFU Interface...
Setting Alternate Setting #0 ...
Determining device status: state = dfuERROR, status = 10
dfuERROR, clearing status
Determining device status: state = dfuIDLE, status = 0
dfuIDLE, continuing
DFU mode device DFU version 011a
Device returned transfer size 2048
DfuSe interface name: "Internal Flash  "
Downloading to address = 0x08000000, size = 21728
Download        [=========================] 100%        21728 bytes
Download done.
File downloaded successfully
Transitioning to dfuMANIFEST state
dfu-util: can't detach
Resetting USB to switch back to runtime mode
pi@fluiddpi:~/klipper $
```

4.3 CANBus Configuration

4.3.1 Use with BIGTREETECH U2C Module



1. Type the following command `sudo nano /etc/network/interfaces.d/can0` in the

ssh terminal and execute
`auto can0`
`iface can0 can static`
`    bitrate 250000`
`    up ifconfig $IFACE txqueuelen 1024`
Set the speed for CANBus at 250K(which must be the same as the speed set in the firmware **(250000) CAN bus speed**), Save (Ctrl + S) after modification and exit (Ctrl + X).Type command `sudo reboot` to reboot Raspberry Pi.

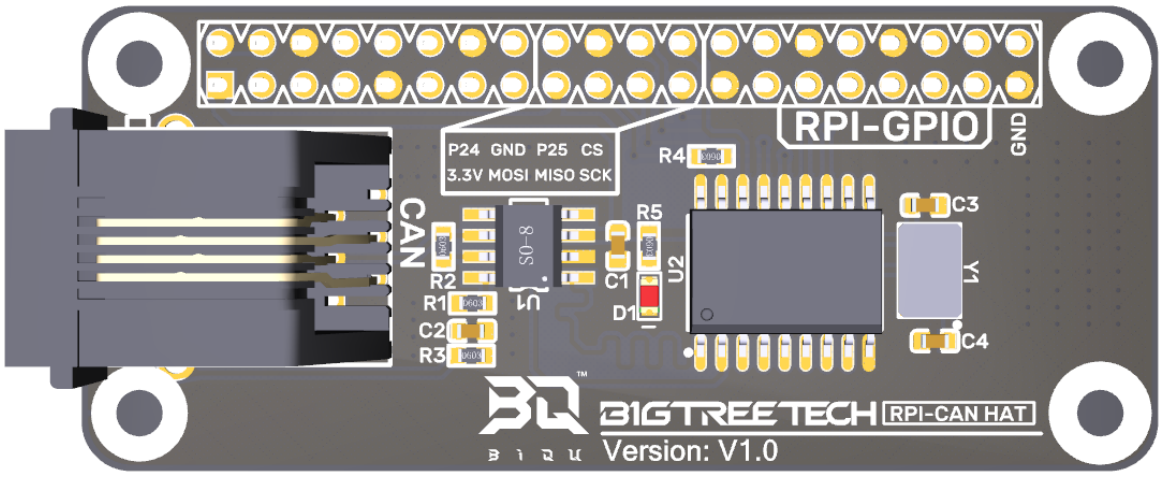
2. Every device on CANBus will generate a canbus_uuid based on MCU's ID. If users want to find the ID for every microcontroller, please make sure the hardware is powered on and wired correctly, then run the following command:

   `~/klippy-env/bin/python ~/klipper/scripts/canbus_query.py can0`

3. If detected an uninitialized CAN device, the above command will report the device's canbus_uuid:
   `Found canbus_uuid=0e0d81e4210c`

4. If Klipper operates normally and is connected to the device, then it won't report canbus_uuid, which is normal.

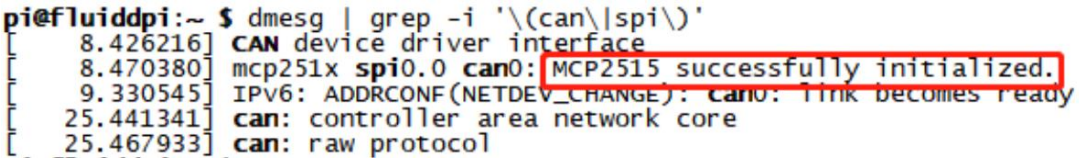
4.3.2 Use with BIGTREETECH RPI-CAN HAT Module



1. Type and run the following command `sudo nano /boot/config.txt`, and added below contents on file config.txt.
   `dtparam=spi=on`
   `dtoverlay=mcp2515-can0,oscillator=12000000,interrupt=25,spimaxfrequency=1000000`
   After modification, save(Ctrl + S)and exist(Ctrl + X), type `sudo reboot` to reboot Raspberry Pi.

2. Type and run commands `dmesg | grep -i '\(can\|spi\)'` to test if RPI-CAN HAT

module is normally connected. The normal response should be as below:
[8.680446] CAN device driver interface
[8.697558] mcp251x spi0.0 can0: MCP2515 successfully initialized.
[9.482332] IPv6: ADDRCONF(NETDEV_CHANGE): can0: link becomes ready



3. Type sudo nano /etc/network/interfaces.d/can0 on the ssh terminal and run the command.
   auto can0
   iface can0 can static
   bitrate 250000
   up ifconfig $IFACE txqueuelen 1024
   Set the speed for CANBus at 250K(which must be the same as the speed set in the firmware **(250000) CAN bus speed**). Save (Ctrl + S) after modification and exit (Ctrl + X). Type command sudo reboot to reboot Raspberry Pi.

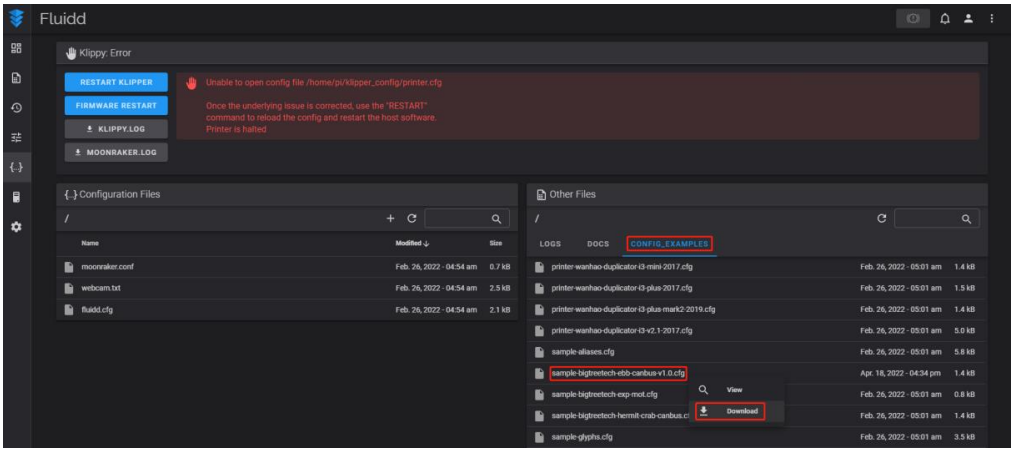
4. Every device on CANBus will generate a canbus_uuid base on MCU's ID. If users want to find ID for every microcontroller, please make sure the hardware is powered on and wired correctly, then run the following command:
   ~/klippy-env/bin/python ~/klipper/scripts/canbus_query.py can0

5. If detected an uninitialized CAN device, the above command will report the device's canbus_uuid:
   Found canbus_uuid=0e0d81e4210c

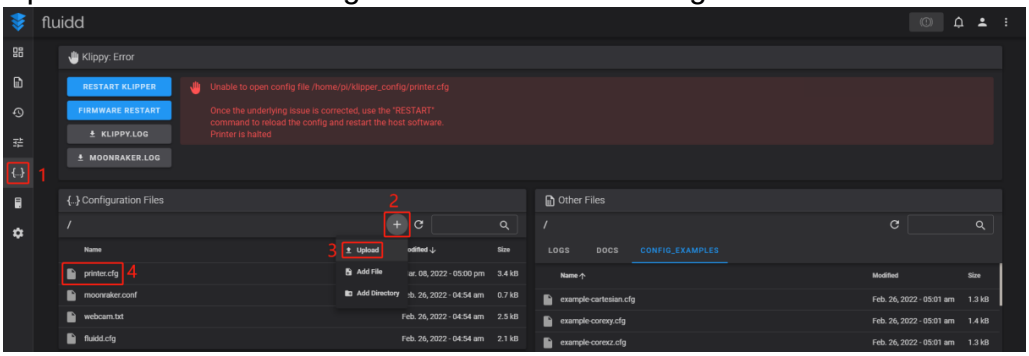
6. If Klipper operates normally and is connected to the device, then it won't report the canbus_uuidm, which is normal.

4.4 Klipper Configuration

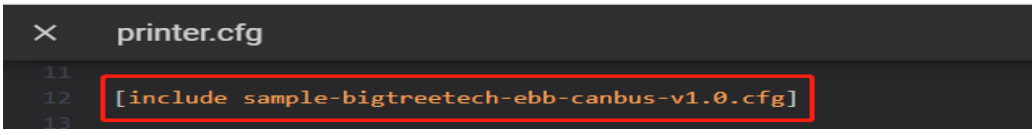
1. Enter Raspberry Pi's IP address in a browser to access the page as shown below picture, and download the reference configuration of the motherboard. If you cannot find this file, please update the Klipper firmware source code to the latest version, or download it from github https://github.com/bigtreetech/EBB



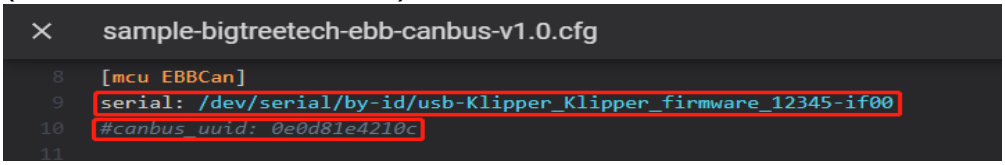
2. Upload the board's configuration files to the Configuration files as shown below.



3. Add the board's configuration to the file“printer.cfg".
[include sample-bigtreetech-ebb-canbus-v1.0.cfg]



4. Revise the ID number of the configuration files as the actual ID of the board (Via USB serial or CANBus).



5. Configure the specific functions of modules as instructed below:

https://www.klipper3d.org/Overview.html

V. Cautions

1. When the TH0 interface doesn't use PT1000, you can't insert the jump cap on it, otherwise 100K NTC won't be used as normal.
2. When using CAN communication, you need to see whether it is used as a terminal. If it is, you must insert the jumper cap on the 120R position.
3. When DIY crimping, you need to pay attention to the line sequences, and do DIY according to the Pin diagram so as to avoid the power line from being reversely connected or connected to the CAN signal, which will get the module burned.
4. If there's no external power supply during writing firmware via USB port, you need to short the VUSB by jumper cap so as to supply the module with working voltage.
5. The load current of the heating rod and the fan interfaces shall not exceed the maximum withstand current to prevent the MOS tube from being burned out.

VI. FAQ

Q: What's the maximum current for the heating rod and fan interface?
A: The maximum output current of the heating rod port: 5A.
The maximum output current of the fan interface: 1A, peak value 1.5A.
The total current for the heating rod, stepper driver and fans needs to be less than 9A.

Q: Cannot update firmware via USB port?
A: You need to make sure that the jump cap is inserted on VUSB, and the indicator light on the board is on.