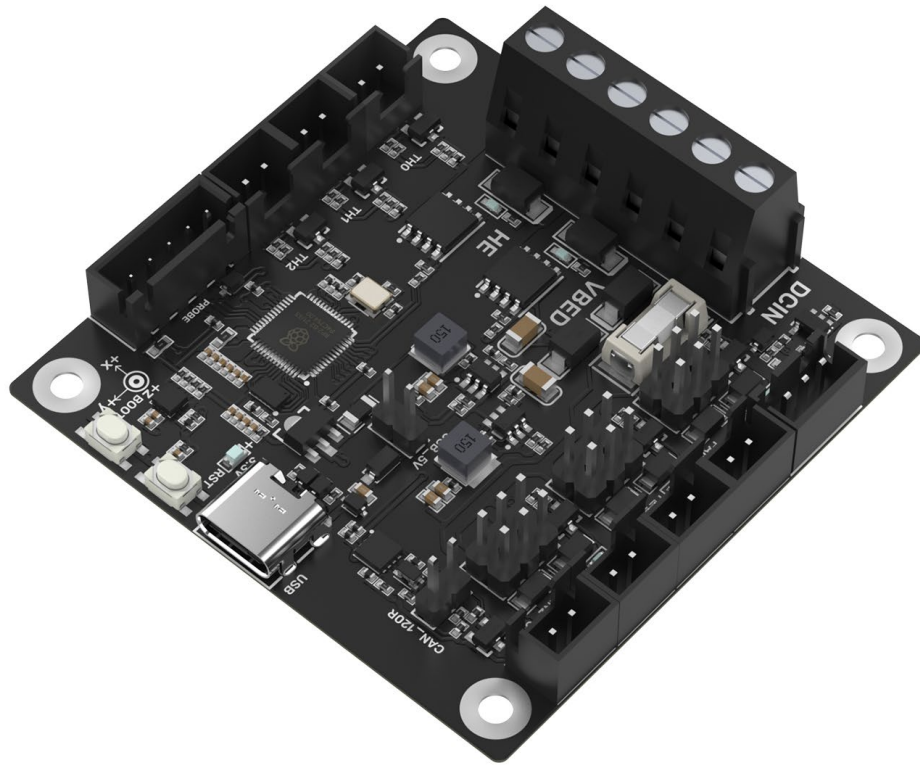


BIGTREE TECH

MMB Cubic V1.0

用户手册



修订历史

版本	日期	修改说明
v1.00	2024/7/18	初稿
v1.01	2024/9/09	更正风扇电压选择指示图

目录

修订历史	2
一、产品简介	4
1.1 产品特点	4
1.2 产品参数	4
1.3 产品尺寸	5
二、外设接口	6
2.1 接口示意图.....	6
2.2 Pin 脚说明.....	7
三、接口介绍	8
3.1 USB 供电.....	8
3.2 数控风扇的电压选择	8
3.3 测温接口	9
3.4 RGB 接口.....	9
3.5 CAN 接口.....	9
3.6 PROBE 接口.....	10
3.7 加热棒接口	10
3.8 热床接口	11
四、Klipper 固件.....	12
4.1 烧录 Katapult (Canboot)	12
4.2 编译 Klipper 固件	13
4.3 通过 katapult 进行固件更新	14
4.4 通过 DFU 进行固件更新	15
4.5 CAN bus 配置	16
4.6 配置 Klipper.....	17
五、注意事项	19

一、产品简介

BIGTREETECH MMB Cubic 是针对 MMB 板子做的拓展板，上面集成了一些 MMB 板子上没有的功能，可以和 MMB 板子通过 CAN 和 USB 通讯，增加了客户的选择。

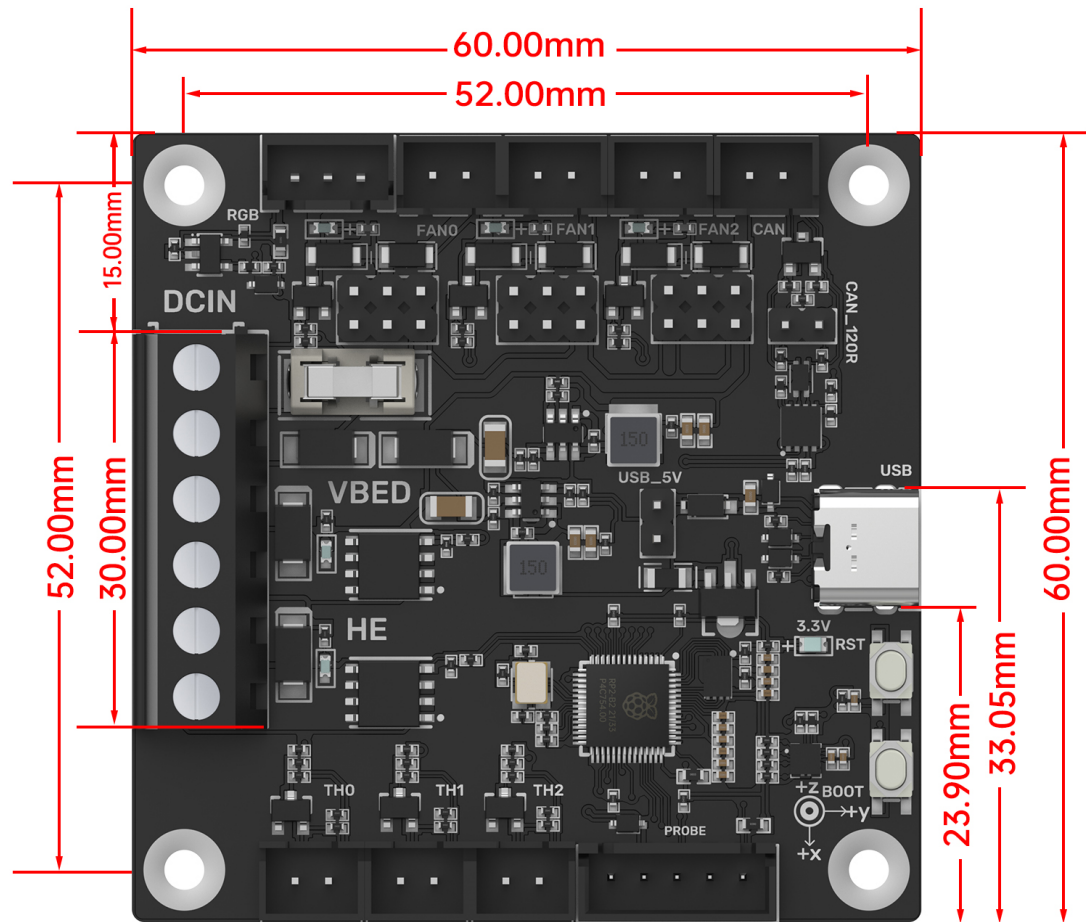
1.1 产品特点

1. 主控芯片：采用 32 位主频 133 MHz 的 ARM Cortex-M0+ RP2040 主控芯片
2. 电源芯片：采用 LN5016-1.5A，支持 24V (MAX:36V) 电源输入, 该芯片输出电流最大可达 1.5A。
3. 主板预留 BOOT 按键，用户可以通过 DFU 方式更新主板引导程序；
4. 增加热敏电阻部分的保护电路，避免因热床或者加热棒漏电导致主控芯片烧毁；
5. 数控风扇 DCIN、12V、5V 电压可选，省去客户外接变压模块的操作，从而减少主板损坏几率；
6. 可通过 USB 接电脑进 DFU，也可通过 Klipper 的 make flash 命令通过 DFU 更新 MCU 固件；
7. 采用高性能 MOSFET 管，减少发热量；
8. 采用可更换的保险丝，方便更换；

1.2 产品参数

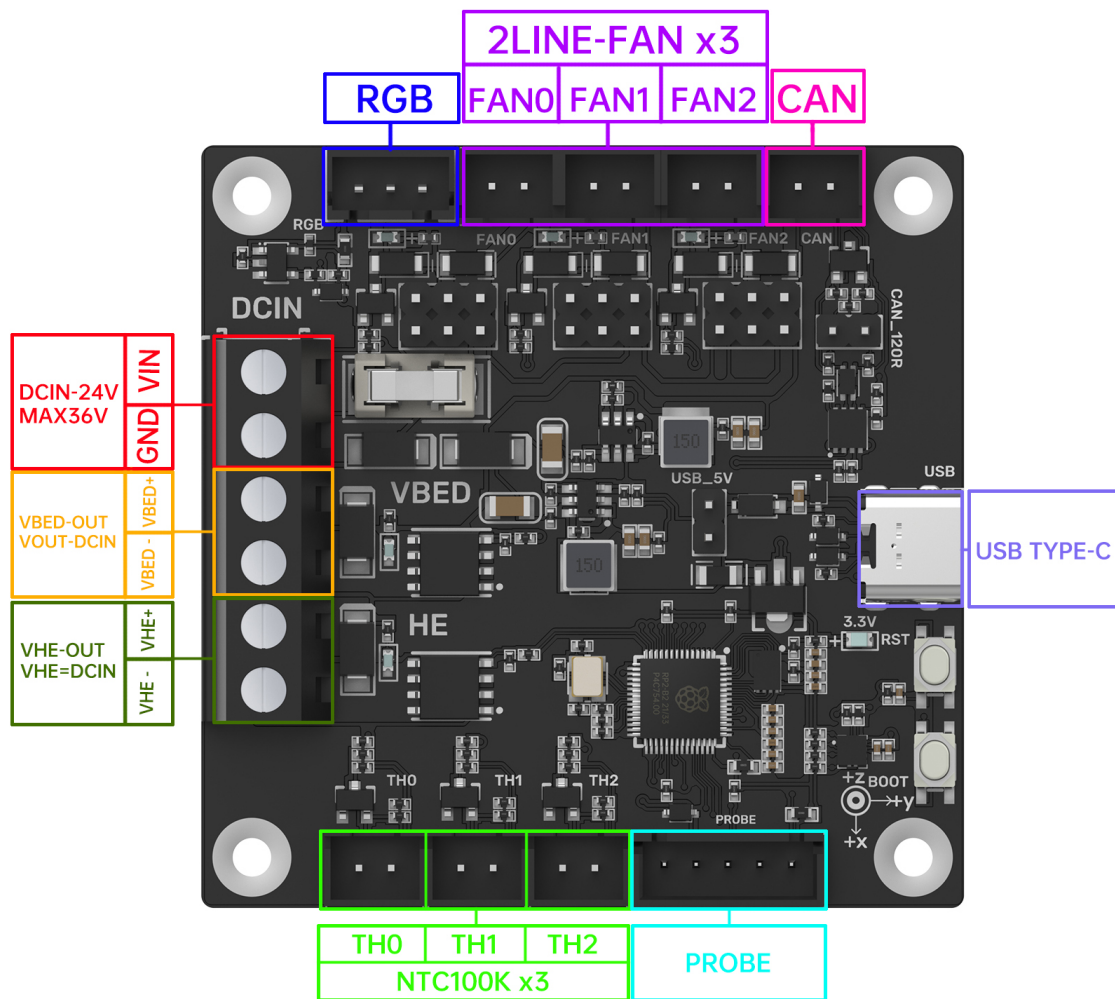
外观尺寸	60mm x 60mm
微处理器	ARM Cortex-M0+ RP2040 133 MHz
主板输入电压	DC24V-36V
热床输入电压	VBED DCIN
逻辑电压	DC3.3V
加热接口	热床 (VBED)、加热棒 (HE)
热床端口最大输出电流	10A
加热棒端口功率	120W (24V 5A)
风扇接口	3 路两线数控风扇 (FAN0, FAN1, FAN2)
风扇接口最大输出电流	总和：1A，峰值 1.5A
加热棒 + 风扇的总电流	最大 10A
主板 5V 输出最大电流	峰值 1.5A
主板 12V 输出最大电流	峰值 1.5A
拓展接口	Probe (Servos、Probe)、RGB、CAN、THx3 (NTC100K)、5V-TYPE-C 供电接口等

1.3 产品尺寸

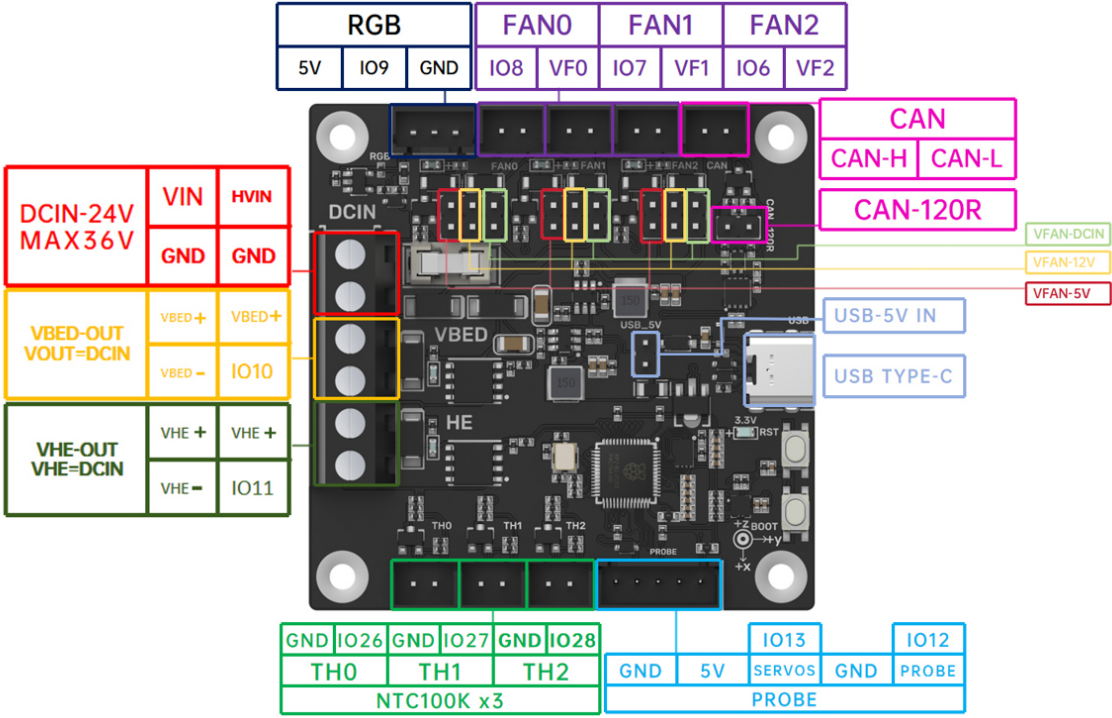


二、外设接口

2.1 接口示意图



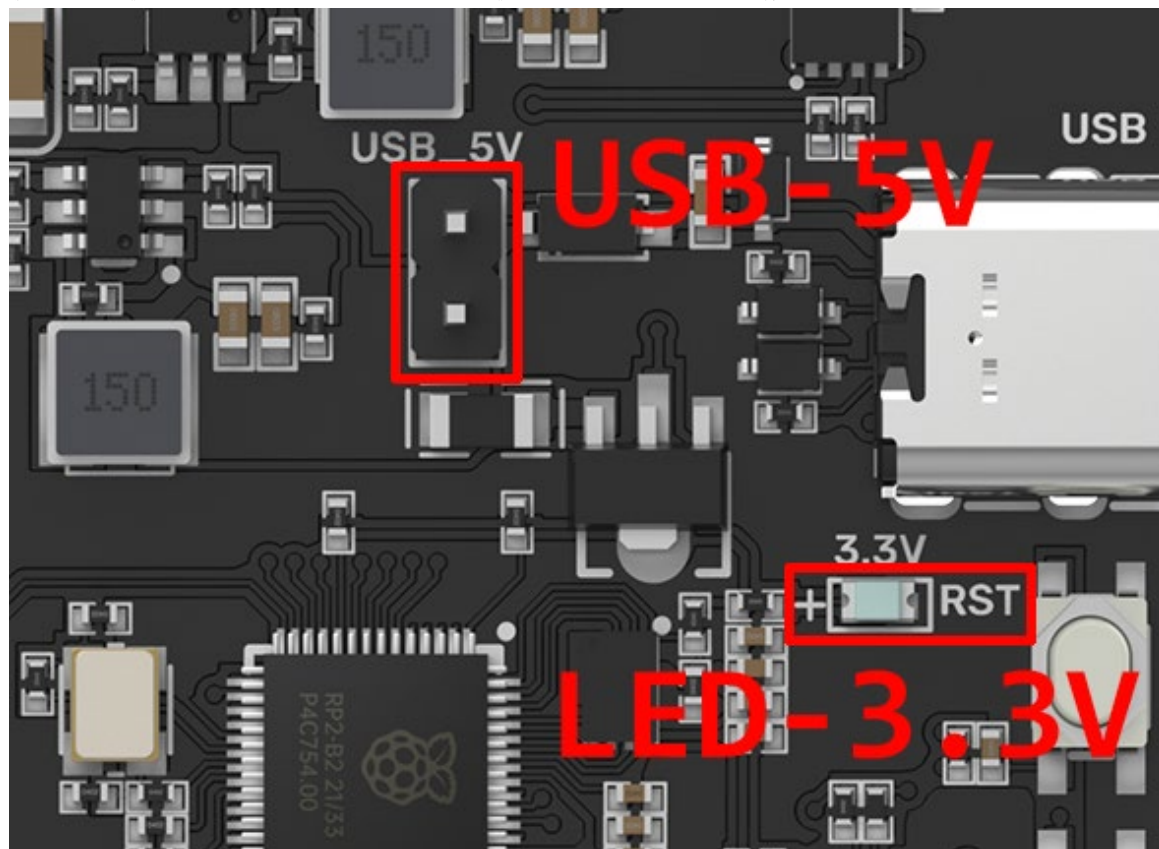
2.2 Pin 脚说明



三、接口介绍

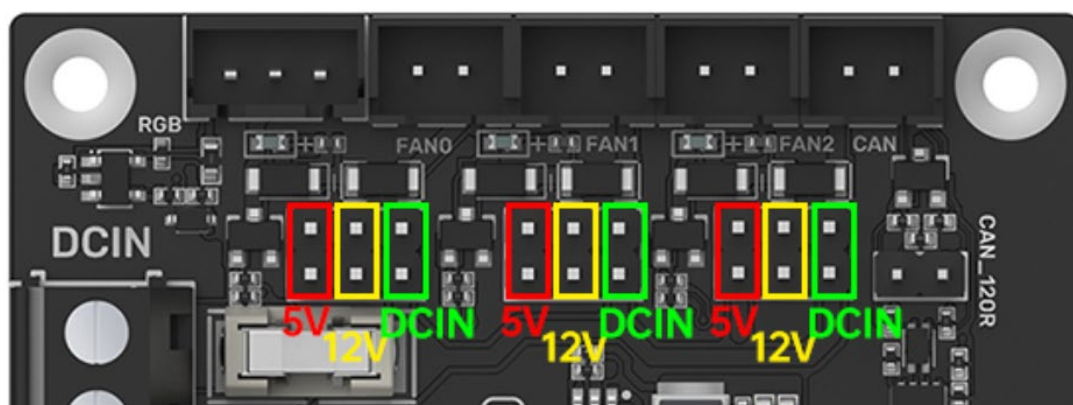
3.1 USB 供电

使用 USB 供电需要短接 USB-5V 排针，供电正常 3.3V 的灯会亮。



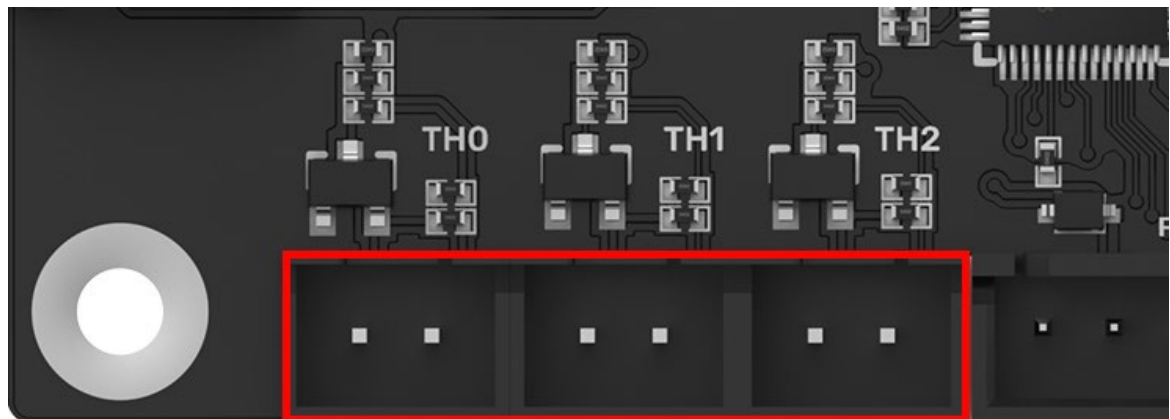
3.2 数控风扇的电压选择

通过跳帽来设置输出电压为 5V，12V 或是 24V。注意：选择电压前请确认清楚风扇支持的电压为哪一档，因为选错导致的风扇烧毁，我司不与承责。

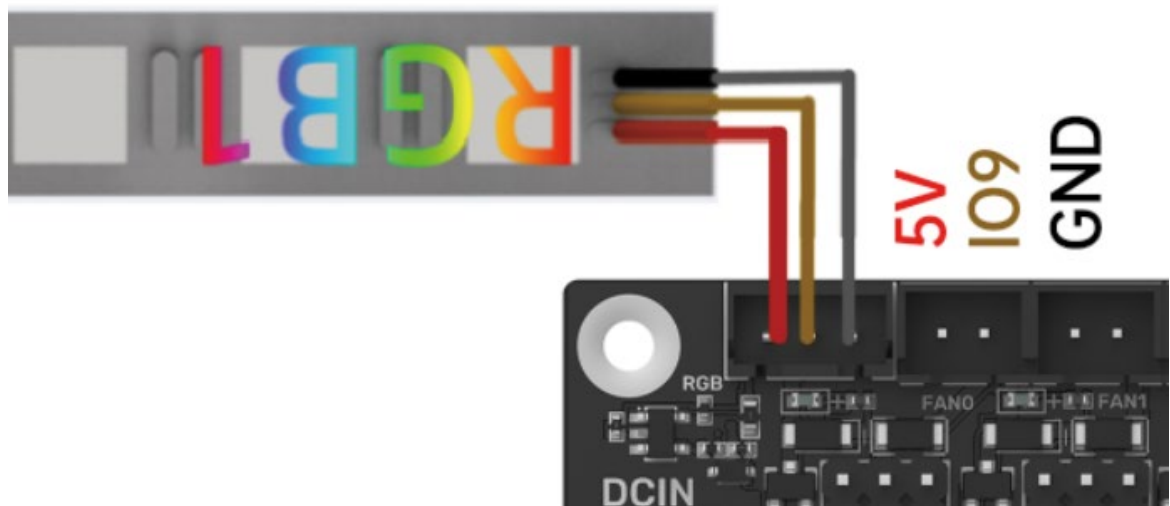


3.3 测温接口

这 3 个接口默认接 NTC100K。

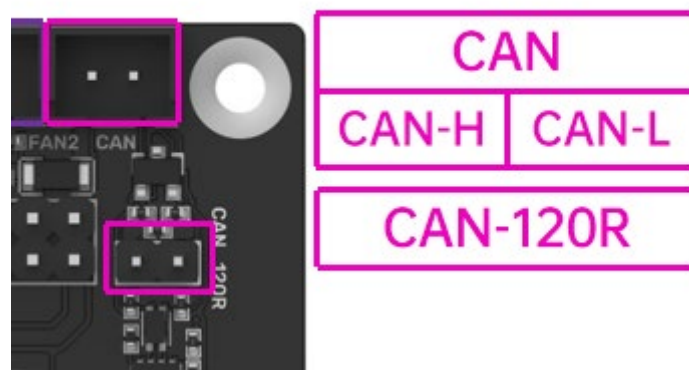


3.4 RGB 接口

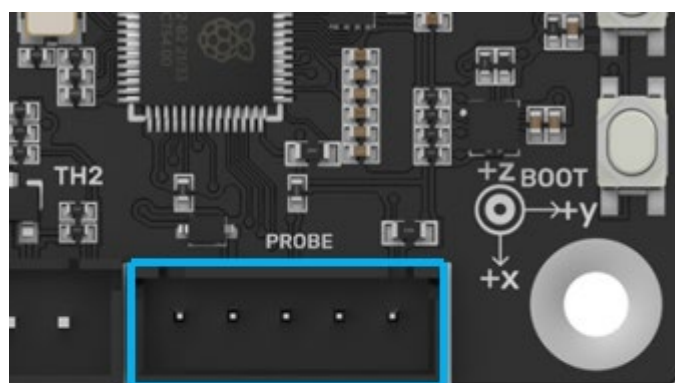


3.5 CAN 接口

如果板子作为 CAN 链的末端，应该接上 CAN-120R。

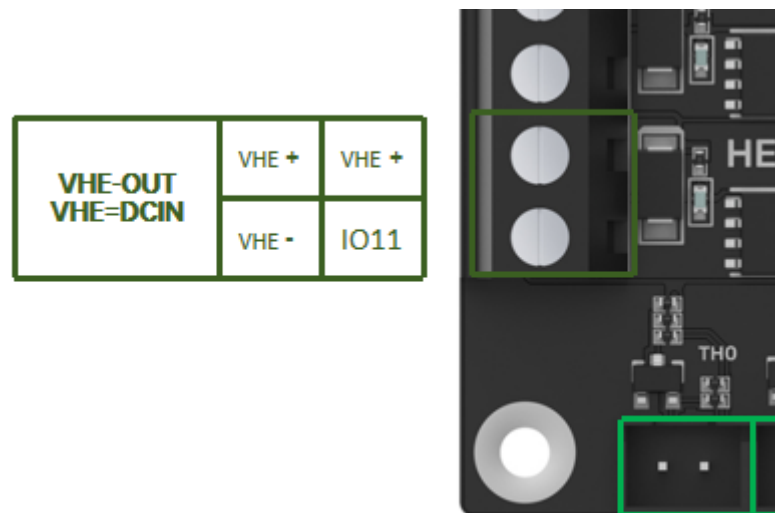


3.6 PROBE 接口



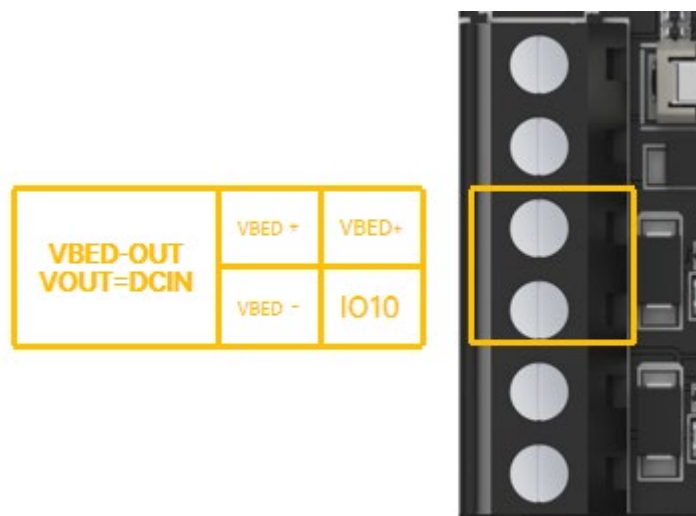
3.7 加热棒接口

注意：加热棒接口的电压和输入电压是一样的，输入如果接 36V，那就需要看加热棒是否支持 36V 输入。



3.8 热床接口

注意：热床接口的电压和输入电压是一样的，输入如果接 36V，那就需要看热床是否支持 36V 输入。



四、Klipper 固件

4.1 烧录 Katapult (Canboot)

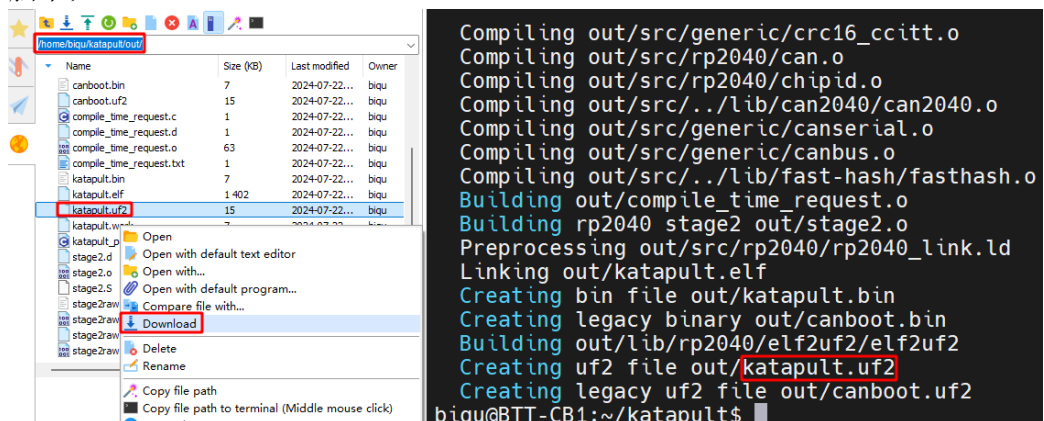
注意：Katapult 旨在通过 CAN bus 接口直接更新 MCU 固件，若您更倾向于使用 DFU 更新方法，请跳过此步骤。

1. 输入
`cd ~`
 跳转到主目录，输入
`git clone https://github.com/Arksine/katapult`
 下载 katapult 工程，然后输入
`cd katapult`
 跳转到 katapult 目录中。

2. 输入
`make menuconfig`
 并按照下图配置

```
(Top)
Katapult Configuration v0.0.1-64-g3e23332
Micro-controller Architecture (Raspberry Pi RP2040) ---->
Flash chip (W25Q080 with CLKDIV 2) ---->
Build Katapult deployment application (Do not build) ---->
Communication interface (CAN bus) ---->
(4) CAN RX gpio number
(5) CAN TX gpio number
(1000000) CAN bus speed
() GPIO pins to set on bootloader entry
[*] Support bootloader entry on rapid double click of reset button
[ ] Enable bootloader entry on button (or gpio) state
[ ] Enable Status LED
```

3. 输入 `make` 编译固件，当 `make` 执行完成后会在 `home/biqu/katapult/out` 文件夹中生成我们所需要的“`katapult.uf2`”固件，在 SSH 软件左侧可以直接下载到电脑中：



4. 请按住 **Boot** 按钮，然后使用 Type-C 线连接至树莓派/CB1，此时芯片进入 DFU 模式

5. 在 SSH 终端命令行中输入

```
lsusb
```

查询 DFU 设备 ID

```
biqu@BTT-CB1:~/klipper$ lsusb
Bus 008 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 007 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 003: ID 2e8a:0003 Raspberry Pi RP2 Boot
Bus 002 Device 002: ID 1a40:0101 Terminus Technology Inc. Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
biqu@BTT-CB1:~/klipper$
```

6. 请输入以下命令以烧录 katapult

```
make flash FLASH_DEVICE=2e8a:0003
```

其中“2e8a:0003”需替换为上一步中查询到的实际设备 ID

7. 烧录完成后，请拔下 Type-C 数据线。

4.2 编译 Klipper 固件

1. SSH 连接到 CB1/树莓派后，在命令行输入：

```
cd ~/klipper/
```

```
make menuconfig
```

使用下面的配置编译固件(如果没有下列选项，请更新 Klipper 固件源码到最新版本)；

```
(Top)
Klipper Firmware Configuration
[*] Enable extra low-level configuration options
    Micro-controller Architecture (Raspberry Pi RP2040) ---->
    Bootloader offset (No bootloader) ---->
    Flash chip (W25Q080 with CLKDIV 2) ---->
    Communication Interface (CAN bus) ---->
(4) CAN RX gpio number
(5) CAN TX gpio number
(1000000) CAN bus speed
() GPIO pins to set at micro-controller startup
[*] Enable extra low-level configuration options
    Micro-controller Architecture (Raspberry Pi RP2040) ---->
如果不使用 katapult
    Bootloader offset (No bootloader) ---->
如果使用 katapult
```

Bootloader offset (16KiB bootloader) --->

Flash chip (W25Q080 with CLKDIV 2) --->

如果使用 CAN bus 通信

Communication Interface (CAN bus) --->

(4) CAN RX gpio number

(5) CAN TX gpio number

(1000000) CAN bus speed

如果使用 USB 通信

Communication Interface (USB SERIAL) --->

USB ids --->

() GPIO pins to set at micro-controller startup

2. 配置选择完成后, 输入 ‘q’ 退出配置界面, 当询问是否保存配置时选择 “Yes” ;
3. 输入 **make** 编译固件, 当 **make** 执行完成后会在 **home/biqu/klipper/out** 文件夹中生成我们所需要的 “**klipper.bin**” 固件

4.3 通过 katapult 进行固件更新

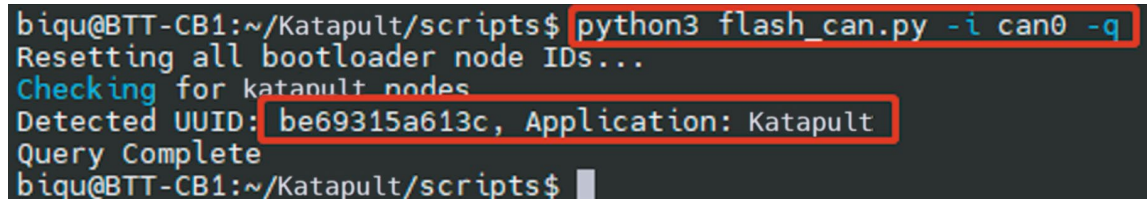
1. 使用 CAN bus 需要接好 CAN bus 线缆以及插上 120R 终端电阻的跳线帽。
2. 输入

```
cd ~/katapult/scripts
```

然后输入

```
python3 flash_can.py -i can0 -q
```

查询 canbus ID (需提前接好 CAN 线并通电), 如下图已找到设备的 UUID



```
biqu@BTT-CB1:~/Katapult/scripts$ python3 flash_can.py -i can0 -q
Resetting all bootloader node IDs...
Checking for katanult nodes
Detected UUID: be69315a613c, Application: Katapult
Query Complete
biqu@BTT-CB1:~/Katapult/scripts$
```

3. 输入

```
python3 flash_can.py -i can0 -f ~/klipper/out/klipper.bin -u be69315a613c
```

替换为实际的 UUID, 注意: **klipper.bin** 需要提前 **make** 生成出来, 并且 **katapult** 的 **Application start offset** 为 **16KiB offset**, 所以 **Klipper** 的 **menuconfig** 中 **Bootloader offset** 也要为 **16KiB bootloader**, 如下图已经烧录

成功。

```
biqu@BTT-CB1:~/Katapult/scripts$ python3 flash_can.py -i can0 -f ~/klipper/out/klipper.bin -u be69315a613c
Sending bootloader jump command...
Resetting all bootloader node IDs...
Checking for katapult nodes...
Detected UUID: be69315a613c, Application: Katapult
Attempting to connect to bootloader
Katapult Connected
Protocol Version: 1.0.0
Block Size: 64 bytes
Application Start: 0x8002000
MCU type: stm32g0b1xx
Verifying canbus connection
Flashing '/home/biqu/klipper/out/klipper.bin'...

[#####]

Write complete: 13 pages
Verifying (block count = 414)...

[#####]

Verification Complete: SHA = C3B1F96A8FCE706587BF4A9119D95D80465875A3
CAN Flash Success
biqu@BTT-CB1:~/Katapult/scripts$
```

4. 再次输入

```
python3 flash_can.py -i can0 -q
```

查询，此时 Application 由之前的 Katapult 变为 Klipper，代表 Klipper 已经正常运行

```
biqu@BTT-CB1:~/Katapult/scripts$ python3 flash_can.py -i can0 -q
Resetting all bootloader node IDs...
Checking for katapult nodes...
Detected UUID: be69315a613c, Application: Klipper
Query Complete
biqu@BTT-CB1:~/Katapult/scripts$
```

4.4 通过 DFU 进行固件更新

树莓派或 CB1 通过 DFU 更新

1. 请按住 **Boot** 按钮，然后使用 Type-C 线连接至树莓派/CB1，此时芯片进入 DFU 模式

2. 在 SSH 终端命令行中输入

```
lsusb
```

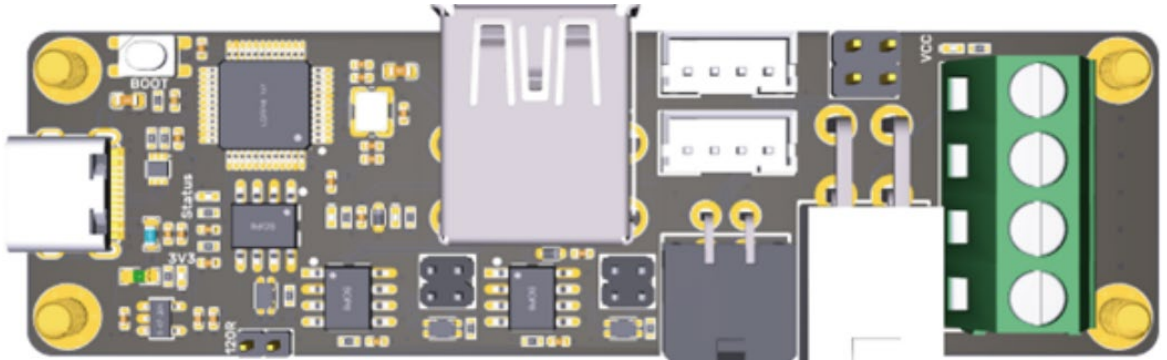
查询 DFU 设备 ID

```
biqu@BTT-CB1:~/klipper$ lsusb
Bus 008 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 007 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 003: ID 2e8a:0003 Raspberry Pi RP2 Boot
Bus 002 Device 002: ID 1a40:0101 Terminus Technology Inc. Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
biqu@BTT-CB1:~/klipper$
```

3. 输入
cd klipper
跳转到 klipper 目录下，输入
make flash FLASH_DEVICE=2e8a:0003
开始烧录固件（注意：将 2e8a:0003 更换为上一步中查询到的实际的设备 ID）
4. 固件烧录完成后，输入
ls /dev/serial/by-id/
查询设备的 Serial ID（只有通过 USB 通信的方式才会有此 ID，CANBus 方式忽略此步骤）。
5. 如果使用 USB 通信，第一次烧录完成之后，再次更新时无需手动按 Boot 按钮进入 DFU 模式，可以直接输入
make flash FLASH_DEVICE=/dev/serial/by-id/usb-Klipper_rp2040_mmb_cubic-if00
烧录固件（注意：将 /dev/serial/by-id/xxx 更换为上一步中查询到的实际的 ID）。
6. 如果使用 CAN bus 通信，烧录完成后，S 请拔下 Type-C 数据线。

4.5 CAN bus 配置

搭配 BIGTREETECH U2C 模块使用



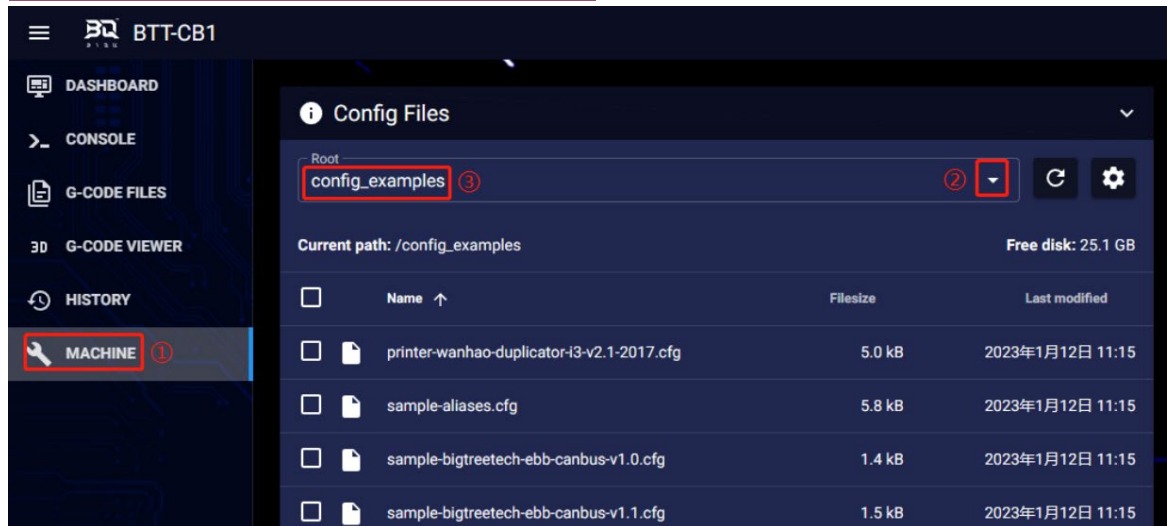
1. 在 SSH 终端中输入
sudo nano /etc/network/interfaces.d/can0
命令, 新增以下内容
allow-hotplug can0
iface can0 can static
bitrate 1000000
up ifconfig \$IFACE txqueuelen 1024
将 CAN bus 速度设置为 **1M**（必须与固件中设置的速度一致(**1000000**) CAN bus speed），修改后保存（Ctrl + S）并退出（Ctrl + X），输入
sudo reboot
重启树莓派。

2. CANBus 上的每个设备都会根据 MCU 的 UID 生成一个 `canbus_uuid`，要查找每个微控制器设备 ID，请确保硬件已通电并正确接线，然后运行：
`~/klippy-env/bin/python ~/klipper/scripts/canbus_query.py can0`
3. 如果检测到未初始化的 CAN 设备，上述命令将报告设备的 `canbus_uuid`
`Found canbus_uuid=0e0d81e4210c`
4. 如果 Klipper 已经正常运行并且连接到此设备，那么 `canbus_uuid` 将不会被上报，此为正常现象。

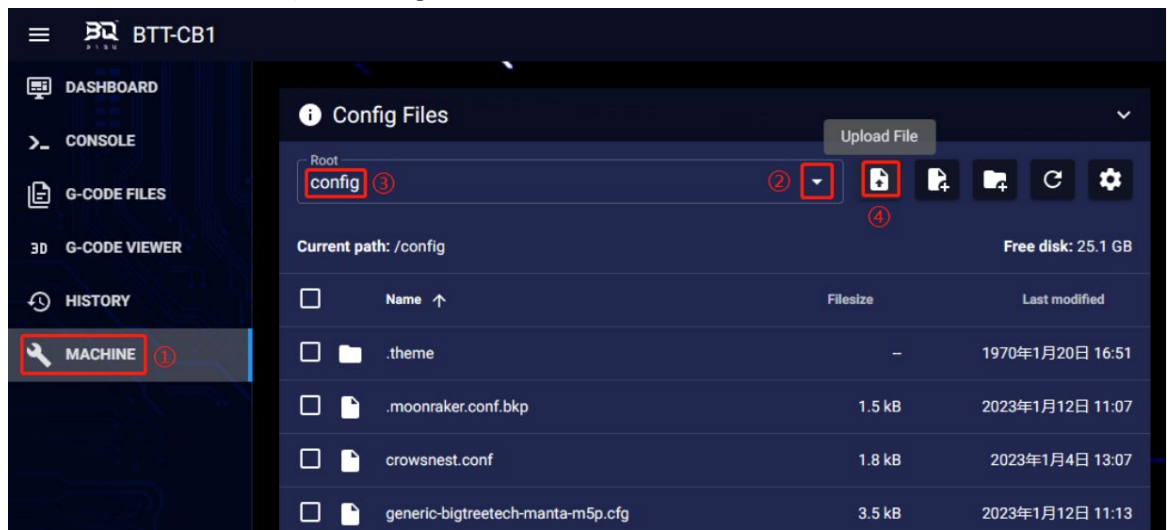
4.6 配置 Klipper

1. 在电脑的浏览器中输入树莓派的 IP 访问，如下图所示的路径中下载名为“`sample-bigtreetech-mm-b-cubic.cfg`”的参考配置，如果找不到此文件，请更新 Klipper 固件源码到最新版本，或者到 GitHub 下载：

<https://github.com/bigtreetech/MMB-Cubic>



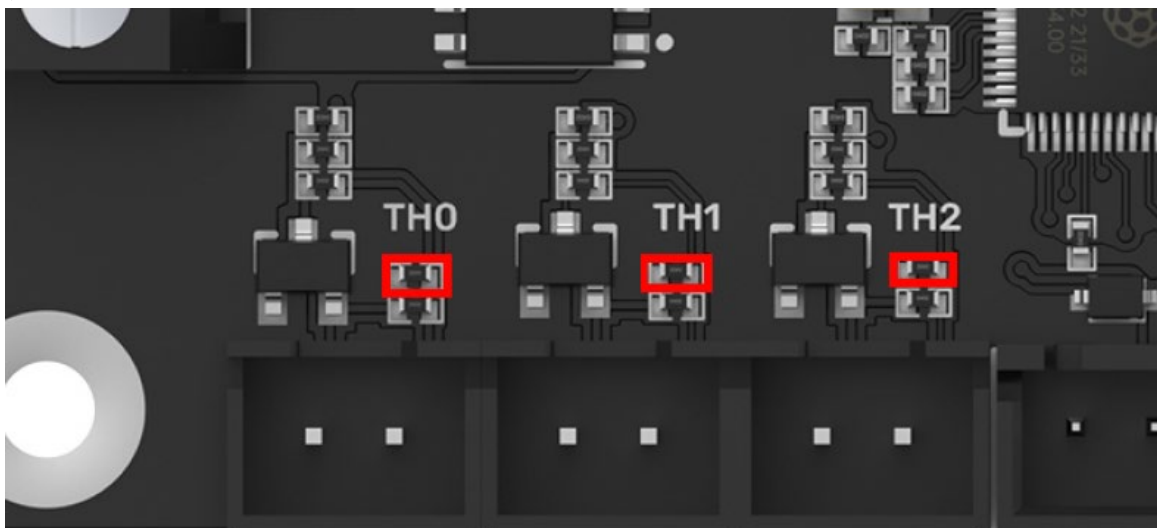
2. 将主板的配置文件上传到 Configuration Files 中;



3. 并在“printer.cfg”文件中添加此主板的配置
[include sample-bigtreetech-mm-b-cubic.cfg]
4. 将配置文件中的 ID 号修改为主板实际的 ID (USB serial 或者 canbus)
5. 按照下方链接的说明配置模块的具体功能:
<https://www.klipper3d.org/Overview.html>

五、注意事项

1. 3 个风扇接口在接 24V 供电电压的时候单个可以达到 1A；如果接 12V 或 5V 的供电电压的时候，3 个加一起最大只能 1.5A。
2. 测温接口默认只能接 NTC100K，如果想要接 PT1000 的话需要加 4.12K 电阻。（下面红框位置）



如果您还需要此产品的其他资源，可以到 <https://github.com/bigtreotech/> 上自行查找，如果无法找到您所需的资源，可以联系我们的售后支持。

若您使用中还遇到别的问题，欢迎您联系我们，我们定会细心为您解答；若您对我们的产品有什么好的意见或建议，也欢迎您回馈给我们，我们也会仔细斟酌您的意见或建议，感谢您选择 BIGTREETECH 制品，谢谢！