

# TR or TR 兔安装教程 V3 版本

## 目录

<b>TR or TR 兔安装教程 V3 版本</b>	<b>1</b>
<b>一、配件购买及 TR 文件打印</b>	<b>3</b>
1、配件购买参考	3
2、打印件打印根据赠送文件中文介绍完成相关打印件打印	3
3、文件夹清单及打印件名称（建议屏蔽掉）	3
1) 工具文件清单：工具浪费时间打着	3
2) 舵机选择清单：舵机改版	3
3) 送料零件清单：送料电机固定-下压力臂打印件-如果打了改件的部分零件无需打	4
4) 选择器清单：9mm 还是 6mm 自己选择	4
5) 通道及选择电机	4
4、TR 兔问题 FAQ ——QQ：865233470（雪妹儿 ~喵）	5
1) 什么是 tr 兔	5
2) 为什么选择 tr 兔	5
3) 什么是快乐兔	6
4) 建议	6
5) 注意事项	6
6) 如何装机	6
7) 兔子板并不是必须的，只要下位机有两个多余的步进驱动口，一个输出型 IO 接口（控制舵机），两个输入型 IO 接口（接归零和编码器）即可	7
8) 关于接线	7
9) 其他注意事项	7
10) 避坑指南：	7
5、打印件打印指南（仅是推荐个别截图存在重复建议看原版）	8
1、大齿轮避坑（台阶问题）	8
2、90 舵机避坑（117M 无视）	8
3、选择电机文件夹（改 TR 兔文件，不打固定电机座）	8
4、送料电机固定-下压力臂打印件	9
5、选择器-加通道	10
6、TR 兔文件（多个版本自行选择-以冷冷哒为例部分零件必须使用黑色料打印）	11
<b>二、总体装配-仅进行机械装配不涉及固定-理线太麻烦不想介绍</b>	<b>12</b>
1、整体打印件一览个别会有差别（冷冷哒为例）	12
2、选择电机装配（优先安装这个是为了后面装配型材及线轨）	12
3、送料电机安装（由于没有挤出机齿轮临时使用 bmg 送料齿轮也是可以使用的）	13
4、兔子传感器模块组装-冷冷哒（红色文字为螺丝长度）	16
5、兔子传感器模块-切刀组装-大 C	18
6、安装线轨及线轨限位不想写了抽时间继续	22

7、组合车头，传感器，皮带模块	23
8、组合通道（完成硬件安装，不含理线及限位安装）	23
<b>三、 兔子主板刷机及调试教程-群友贡献为主不做主动更新</b>	<b>23</b>
1、 兔子主板刷机	23
1) 确认主控类型为 xiao	23
2) 按照要求下载 xiao 编译文件（兔子文档内部），并进行初步编译	23
3) 进入 klipper 文件夹进行固件编译	24
4) 刷机教程	24
2、 配置文件更新-xiao 或是 RP2040	25
3、 TR 或是 TR 免调试（群友更新不做具体介绍-可参考兔子调试教程）	26
1) 舵机调试	26
2) 选择电机及送料电机调试	26
3) TR 送料调试	26
4) TR 编码器调试	26
5) 完整调试	26
6) 开始浪	27
<b>四、快乐兔安装教程——QQ：865233470（雪妹儿 ~喵 ）</b>	<b>27</b>
1、 快乐兔安装	27
2、可能出现的问题：	27
3、装完软件后的一些调试语句	28
4、软件部分：快乐兔屏幕	29
1、出现如图警告	30
2、出现如图错误	30
<b>五、 KlipperScreen 安装教程群友贡献-QQ：1546685250</b>	<b>30</b>
1. 关闭系统图形界面	31
2. 检查 xserver-xorg-legacy 是否正确安装	31
3. 查看日志	31
4、安装驱动：（以下为 78 平板使用不适用其它文件）	32
<b>六、Klipper 配置 wifi 和下位机-唔，待定(484785558);</b>	<b>34</b>
1、设置 wifi	34
step1: 设置会话并登录	34
step2: nmtui 设置 wifi	35
2、klipper 和主板常见连接错误	40
3、编译主板固件	40

## 一、配件购买及 TR 文件打印

### 1、配件购买参考

《TR\_材料清单\_带淘宝衔接-梦梦专享-Vn》根据表格内内容完成配件购买

### 2、打印件打印根据赠送文件中文介绍完成相关打印件打印

《TradRack\_梦梦整理 MOD-Vn》

### 3、文件夹清单及打印件名称（建议屏蔽掉）

#### 1) 工具文件清单：工具浪费时间打着

- | 导轨安装定位件，如果装过 2.4 就去找找没有看你心情打两个.stl
- | 说真的舵机臂调试件我都没打过.STL
- |
- | —建议直接删掉的固定件-当然土豪随意
- | | —没多余型材的需要打孔固定自己选
- | | 右侧固定.STL
- | | 左侧固定.STL
- |

#### 2) 舵机选择清单：舵机改版

- | | —9g servo parts-S90 舵机
- | | 90S 束线器随意打.STL
- | | 90S 舵机固定件.STL
- | | README.md
- | | 舵机臂尺寸-4.7\_x1\_rev2.STL
- | | 舵机臂尺寸-4.8\_x1\_rev2.STL
- | | 舵机臂尺寸-4.9\_x1\_rev2.STL
- | | 舵机臂尺寸-5.0\_x1\_rev2.STL
- | | 舵机臂尺寸-5.1\_x1\_rev2.STL
- | |
- | | —RIDGA parts-BMG 齿轮替代

- | README.md
- | 配合改版件的-左侧固定.STL
- | 配合有台阶的 BMG 使用-电机固定.STL
- |

### 3) 送料零件清单：送料电机固定-下压力臂打印件-如果打了改件的部分零件无需打

- | 力臂必须打.STL
- | 右力臂.STL
- | 左力臂.STL
- | 弹簧螺丝支撑.STL
- | 电机固定 1.STL
- | 电机固定 2.STL
- |

### 4) 选择器清单：9mm 还是 6mm 自己选择

- | 117M 舵机固定件必打如果使用 90S 不打.STL
- | 117M 舵机臂必打如果使用 90S 不打.STL
- | C 型导轨 6MM 车头-TR 兔或是传感器有专门的车头改件.STL
- | C 型导轨 9MM 车头-TR 兔或是传感器有专门的车头改件.STL
- | h 型导轨 6MM 车头-TR 兔或是传感器有专门的车头改件.STL
- | h 型导轨 9MM 车头-TR 兔或是传感器有专门的车头改件.STL
- | 左侧固定件.STL
- | 改 TR 兔不动车头前面板-类型 1.STL
- | 改 TR 兔不动车头前面板-类型 2.STL
- | 束线器.STL
- | 束线器打不打随意.STL
- |

### 5) 通道及选择电机

- | 6MM 皮带用的一个件主要是弥补惰轮长度使用的，在选择电机的对端固定惰轮处使用.STL
- | C 型导轨选择电机底部固定座原版必打，改传感器方案（TR 兔）无需打因为在改造方案里面有.STL
- | H 型导轨选择电机底部固定座原版必打，改传感器方案（TR 兔）无需打因为在改造方案里面有.STL
- | 底部束线的一个模型打一下吧.STL
- | 归零的微动盖子看你心情打不打建议打一个.STL

- | 惰轮固定座-6MM 惰轮需配合弥补件. STL
- | 线轨固定件件避免型材过长，导轨过短造成的滑块滑落的看你需要. STL
- | 选择电机固定不打没法玩. STL
- | 选择电机固定必打. STL
- | 通道-你有几色就打几个. STL

└—就是一个通道口鲍登头的固定件区分通道根据需求打印

```

number_11_collet_clip_x1_rev1.STL
number_12_collet_clip_x1_rev1.STL
number_13_collet_clip_x1_rev1.STL
number_14_collet_clip_x1_rev1.STL
number_16_collet_clip_x1_rev1.STL
number_17_collet_clip_x1_rev1.STL
number_18_collet_clip_x1_rev1.STL
number_19_collet_clip_x1_rev1.STL
number_1_collet_clip_x1_rev1.STL
number_2_collet_clip_x1_rev1.STL
number_3_collet_clip_x1_rev1.STL
number_4_collet_clip_x1_rev1.STL
number_6_collet_clip_x1_rev1.STL
number_7_collet_clip_x1_rev1.STL
number_8_collet_clip_x1_rev1.STL
number_9_collet_clip_x1_rev1.STL
[a]_number_0_collet_clip_x1_rev1.STL
[a]_number_10_collet_clip_x1_rev1.STL
[a]_number_15_collet_clip_x1_rev1.STL
[a]_number_5_collet_clip_x1_rev1.STL

```

## 4、TR 兔问题 FAQ ——QQ: 865233470 (雪妹儿 ~喵)

### 1) 什么是 tr 兔

tr 兔是指 tr 部分加装兔子多色的编码器，以使用兔子多色或者快乐兔软件进行控制的一种叫法

### 2) 为什么选择 tr 兔

tr 兔只需要两套 BUG 齿轮，其中第二套不需要大齿轮，大大节省成本，而且

装机难度更低

### 3) 什么是快乐兔

快乐兔是指兔子多色的升级版，更加智能优秀，而且可以配合屏幕进行控制调试

### 4) 建议

建议组装至少 12 色，可以不用，但用是不能没有哈哈哈哈哈哈

tr 加兔的 mod（编码器部分《TR 加装编码器全套改装 MOD\_R1.4(完结)》由冷冷哒开发 QQ: 244253339）（编码器有个编码器支架测试打印单独的《5000 中.stl》更加合适）（建议先按照原版的 tr 三维安装在安装 mod，否则你可能变得不幸）

### 5) 注意事项

编码器齿轮不需要涂色，控制编码器上的挡片几乎接近齿尖即可，但是打印件如果不是黑色的需要吧编码器光线可能照到的地方涂色

测试：给编码器供电，抽动耗材灯光会闪烁即成功，不亮或者常量都是不行的

### 6) 如何装机

Tr 三维图-对照装机.exe，是个三位图，打开对照三位进行装机，光标移动到零件上会显示零件的规格型号

7) 兔子板并不是必须的，只要下位机有两个多余的步进驱动口，一个输出型 IO 接口（控制舵机），两个输入型 IO 接口（接归零和编码器）即可

## 8) 关于接线

步进电机 2\*4=8

舵机 3 +5 gnd s

编码器 1 供电与舵机共线 s

滑车归零 1 gnd 与舵机共线 s

使用网线仅需两根

## 9) 其他注意事项

1. 线轨购买 H(加长) 的
2. 舵机购买 1171MG(贵但是劲大)
3. 原版需要完整一套 BMG 齿轮
4. TR 兔多一个滚针轴承挤出轮用于编码器识别
5. 下位机有多余两个步进驱动口的无需购买兔子版
6. 步进电机多长的无所谓
7. 线轨 MGN9H 227 长做防锈
8. 型材欧标 2020 257 长
9. 线轨与型材可以长一些，长度没影响
10. 群里所有资料均为 12 色，想做其他数量的自行解决

## 10) 避坑指南：

- 1、如果使用 TR 原版请按照原板文件进行打印即可，如果使用 TR 兔请按照要求打印推荐打印件 MOD《TR 加装兔子相关全套改装 MOD-V1-冷冷哒》、《TR 编码器-大 C》

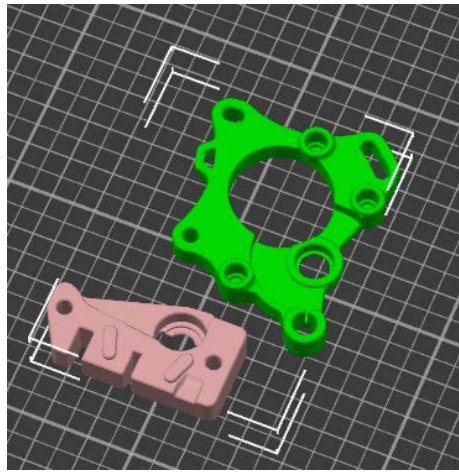
MOD 安装教程参考兔子安装教程 49-53 页不做具体介绍

2、编码选择按照当前更新进度无论编码器孔在中间还是边上均不影响使用。

## 5、打印件打印指南（仅是推荐个别截图存在重复建议看原版）

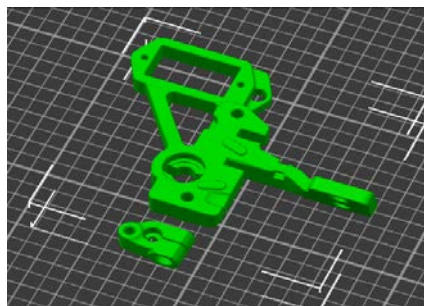
### 1、大齿轮避坑（台阶问题）

舵机改版\RIDGA parts-BMG 齿轮替代-配合有台阶的 BMG 使用-电机固定.stl-配合改版件的-左侧固定.STL



### 2、90 舵机避坑（117M 无视）

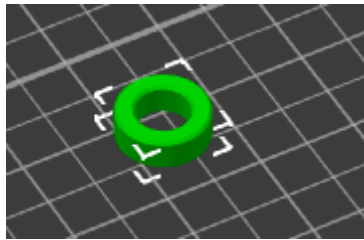
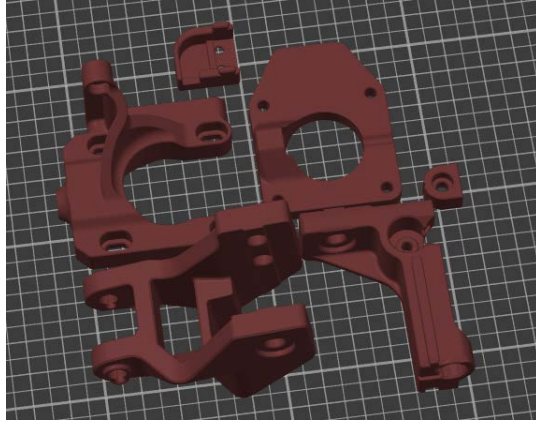
舵机改版\9g servo parts-S90 舵机-90S 舵机固定件.STL-舵机臂尺寸-4.7\_x1\_rev2.STL



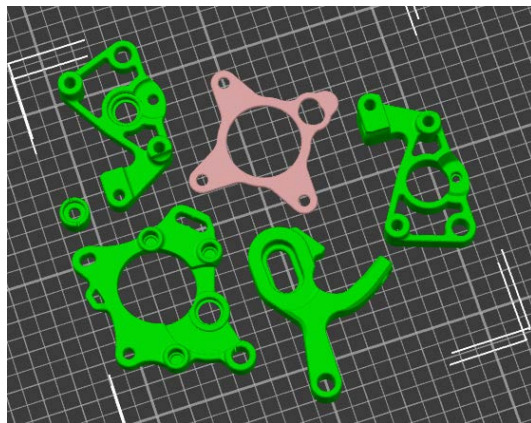
### 3、选择电机文件夹（改 TR 兔文件，不打固定电机座）

通道及选择电机-全文件其实没多少根据中文介绍打印，看不懂的别怨我

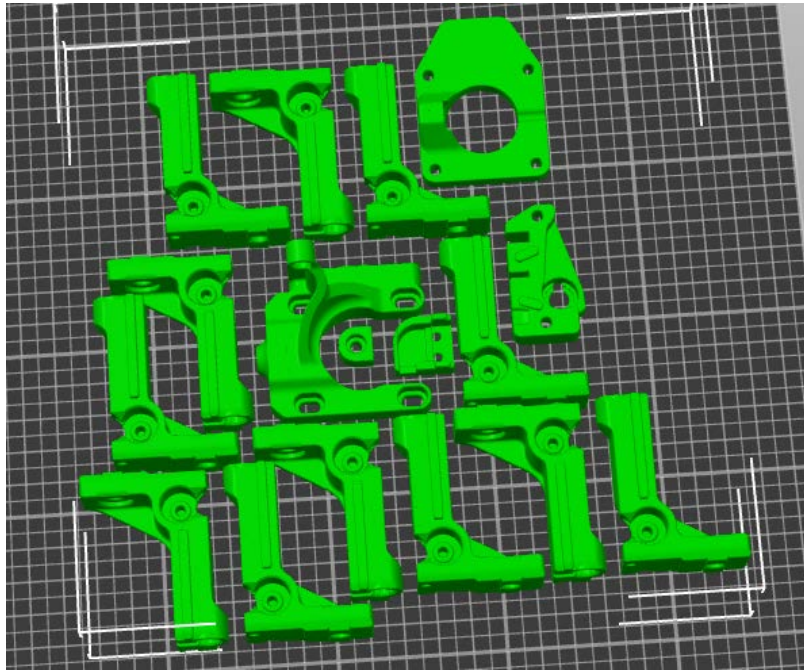




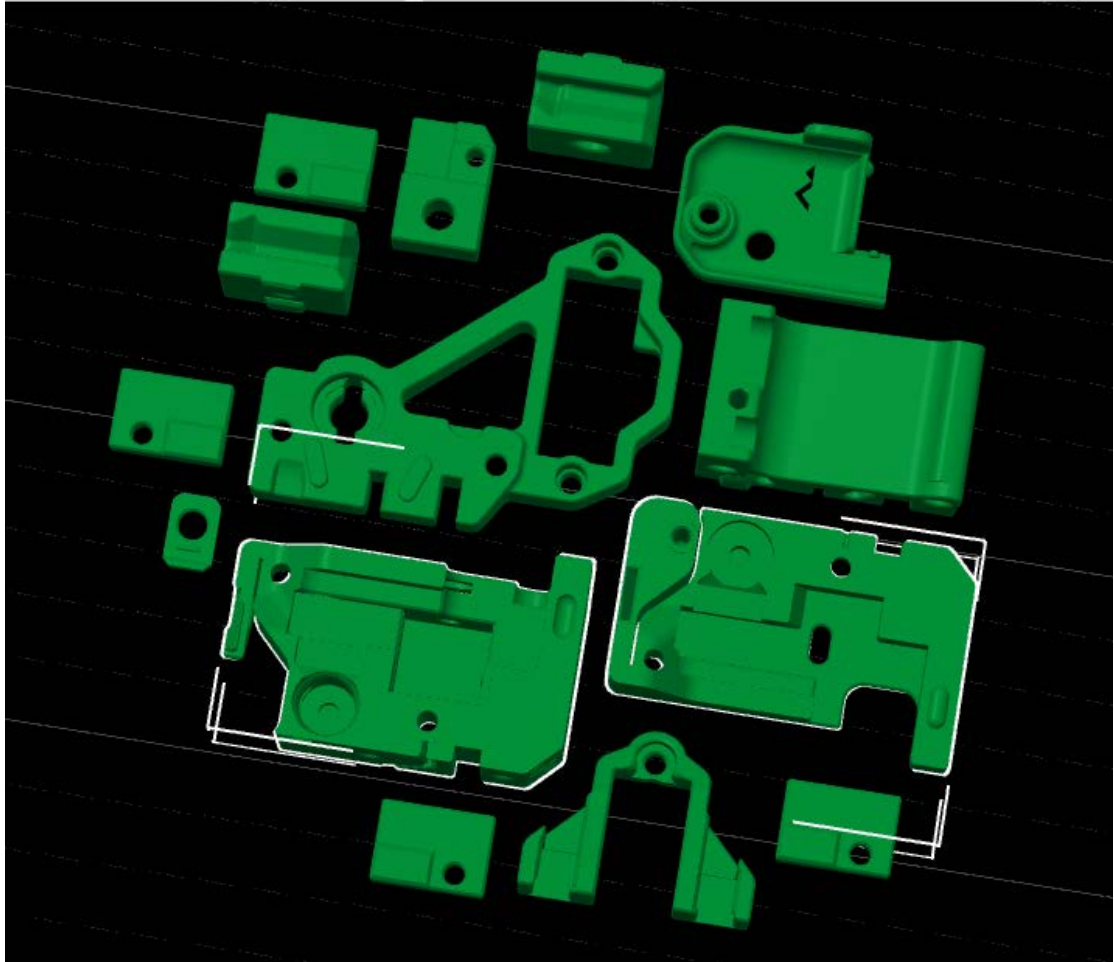
#### 4、送料电机固定-下压力臂打印件



## 5、选择器-加通道



6、TR 兔文件（多个版本自行选择-以冷冷哒为例部分零件必须使用黑色料打印）



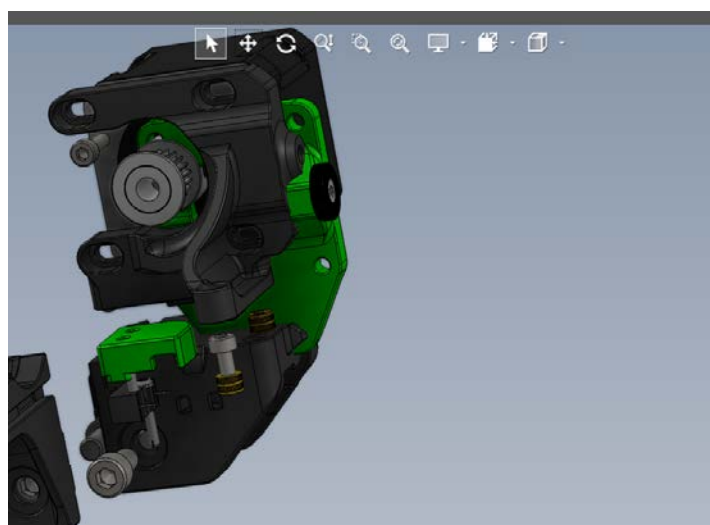
## 二、 总体装配-仅进行机械装配不涉及固定-理线太麻烦不想介绍

### 1、 整体打印件一览个别会有差别（冷冷哒为例）



### 2、选择电机装配（优先安装这个是为了后面装配型材及线轨）

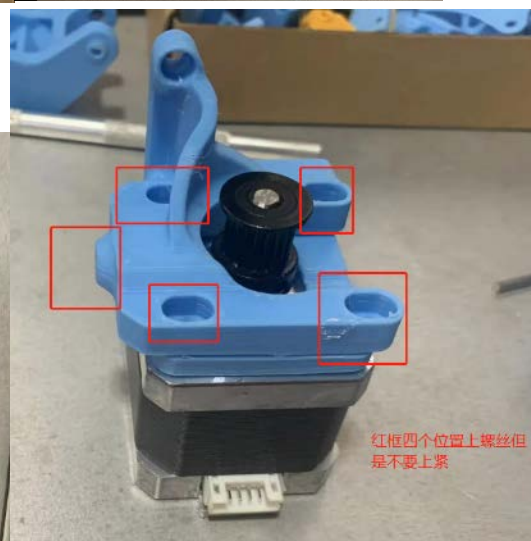
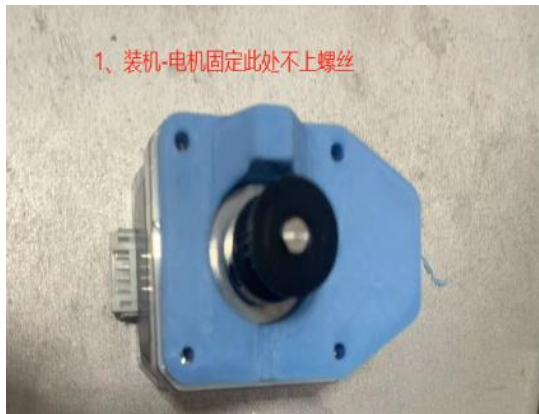
1)、安装送料电机-自己可以选其它的也行使用配件及螺丝如下



1、M3\*14, M3\*10 螺丝上线固定使用螺丝固定电机到打印件使用应该是这个

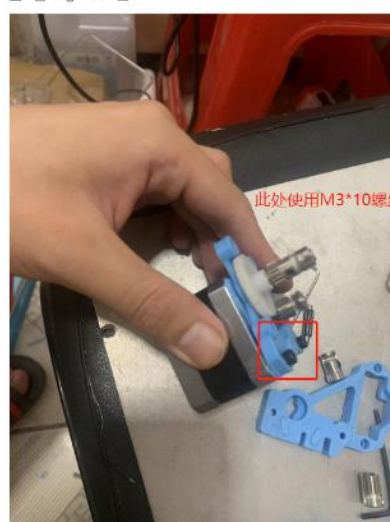


- 2、3\*4\*5 土八螺母 4 个用来内嵌打印件
- 3、M5\*10 螺丝，船型，T 型螺母 3 对 固定打印件到型材



### 3、送料电机安装（由于没有挤出机齿轮临时使用 bmg 送料齿轮也是可以使用的）

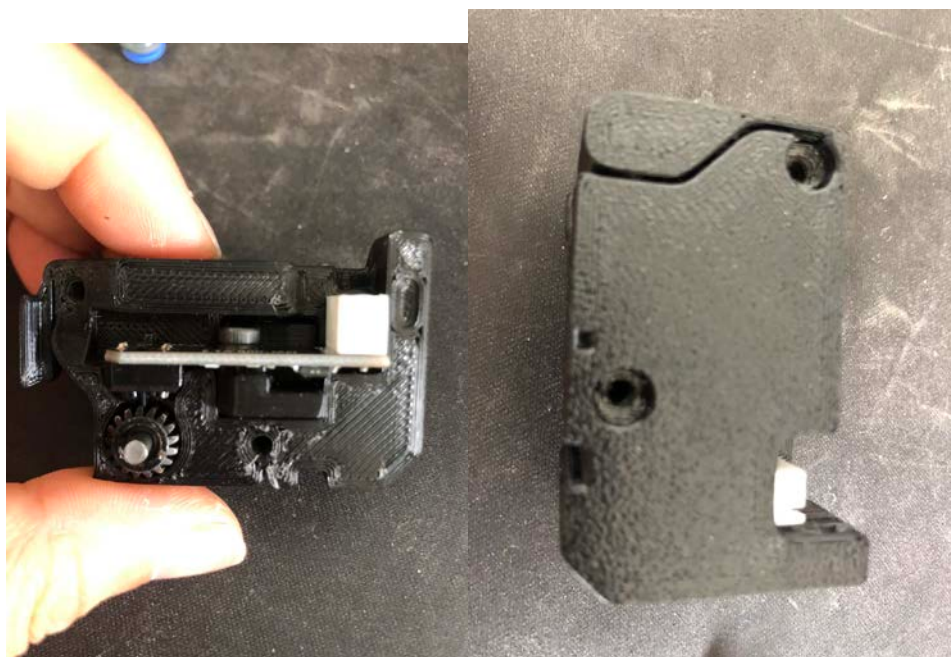
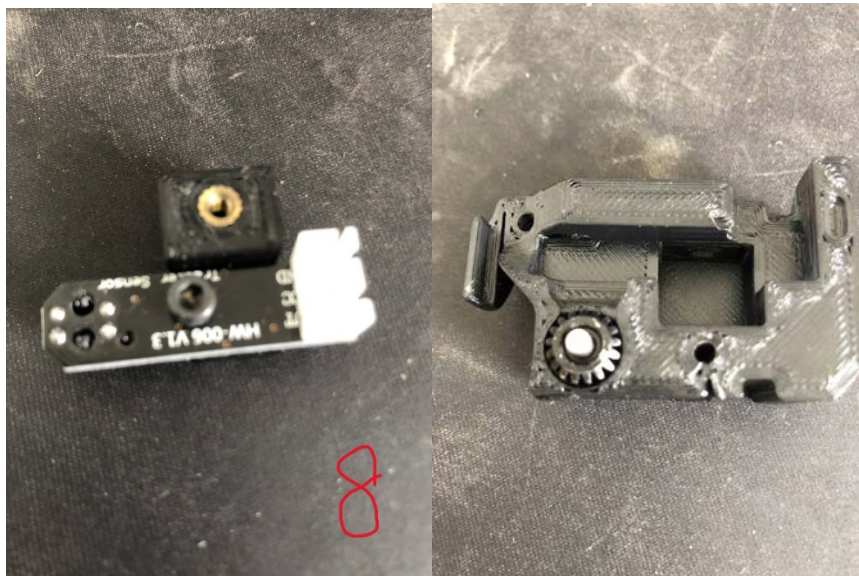
- 1) 这里使用的螺丝主要是 M3\*10，M3\*25
- 2) 螺母为 3\*4\*5 土八螺母等
- 3) 同时此处需要用到 5\*40 梢钉（打磨后的）
- 4) 散装 BMG 拆了准备着要用到。



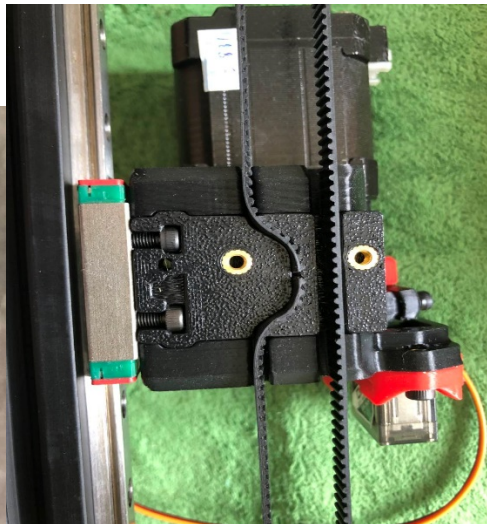
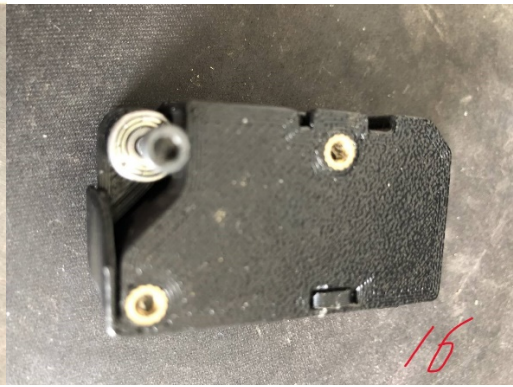
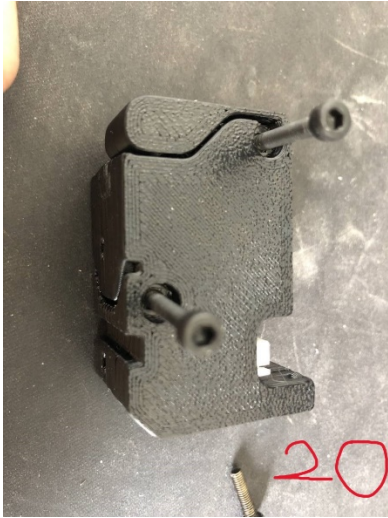


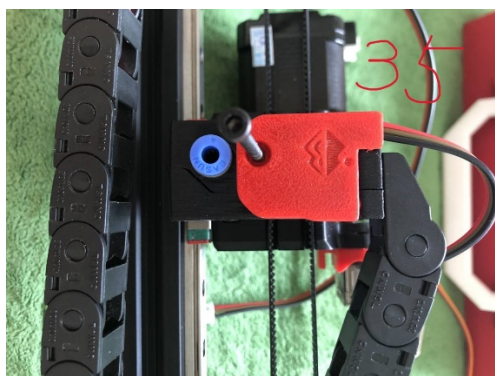
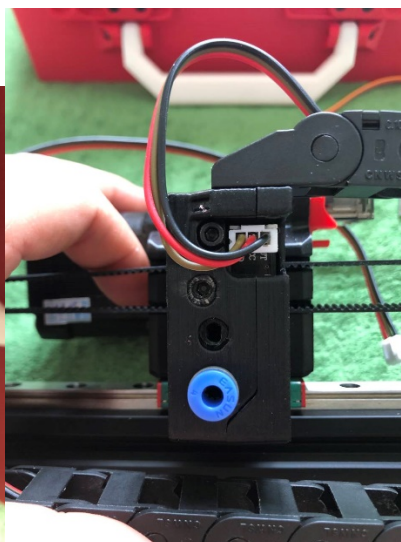
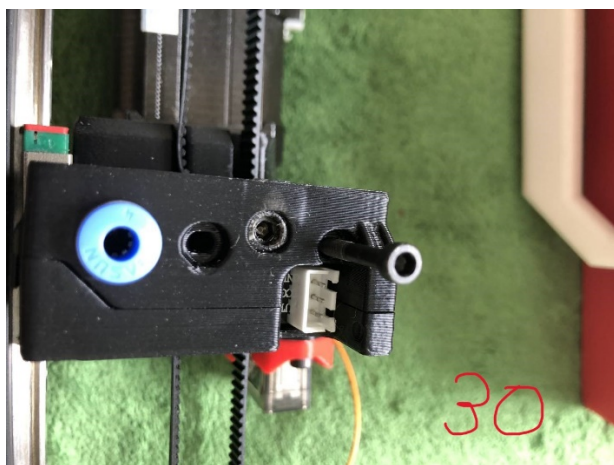


#### 4、兔子传感器模块组装-冷冷哒（红色文字为螺丝长度）

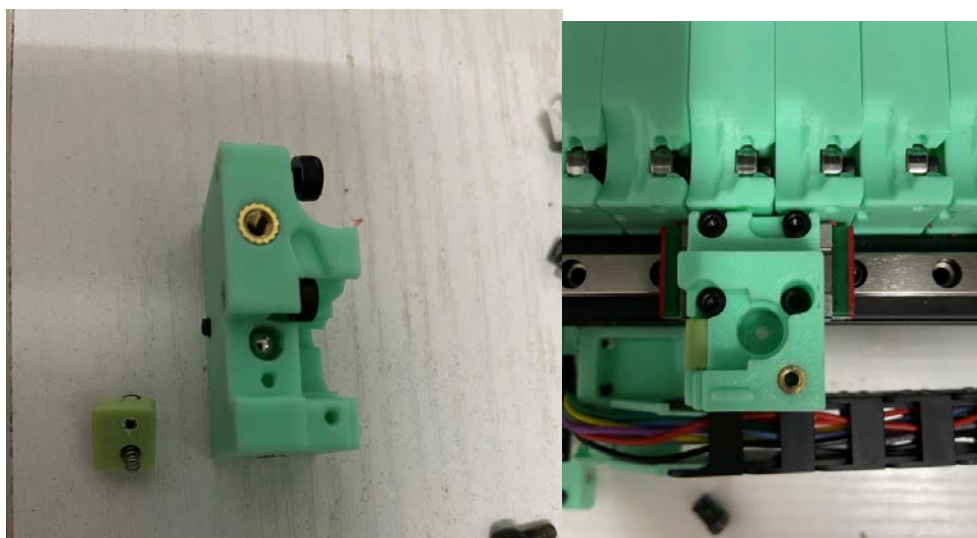




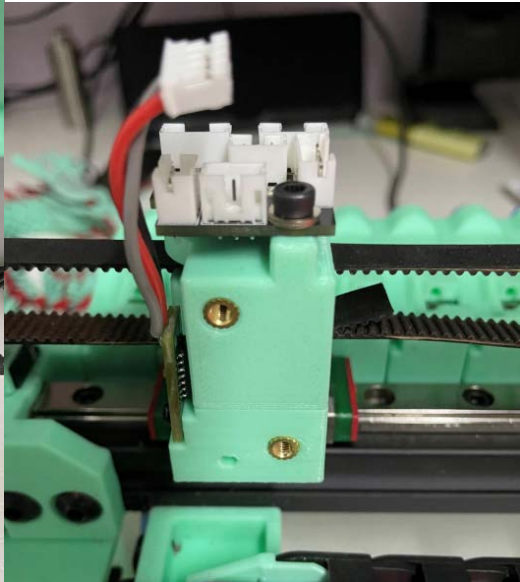
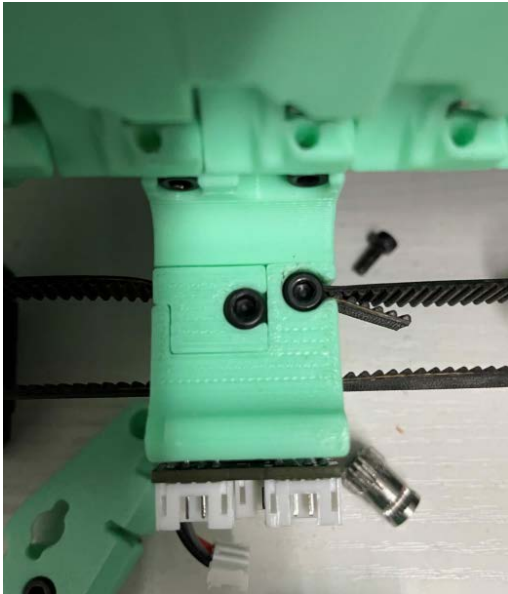
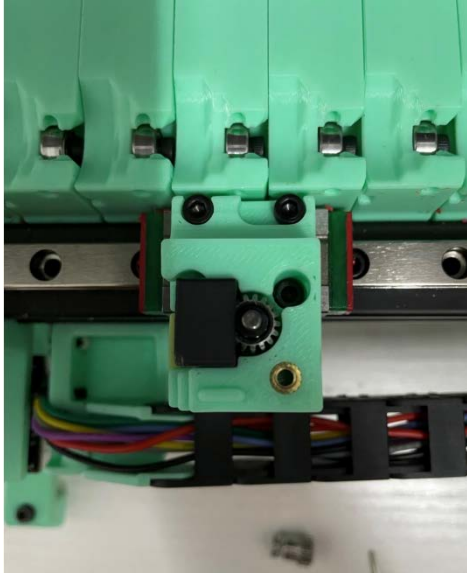


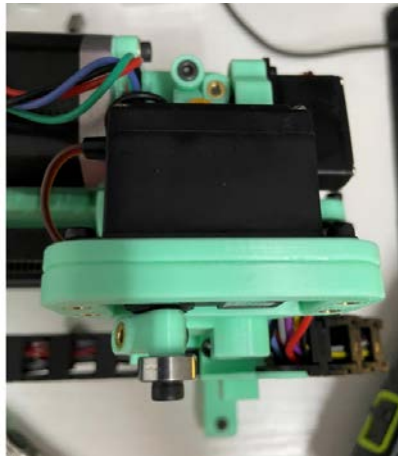
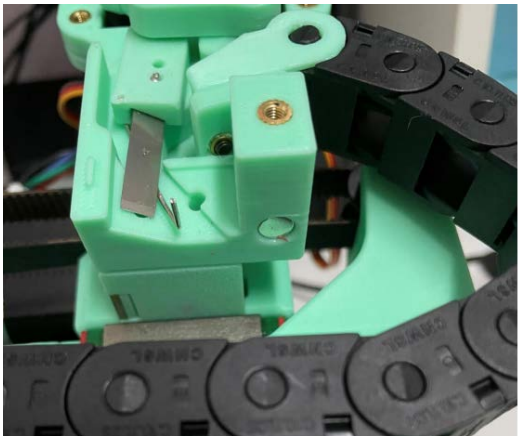
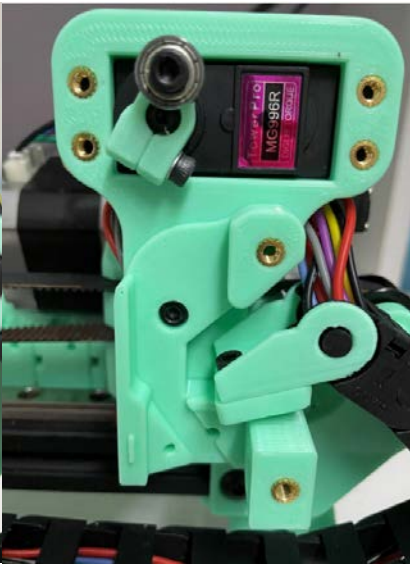
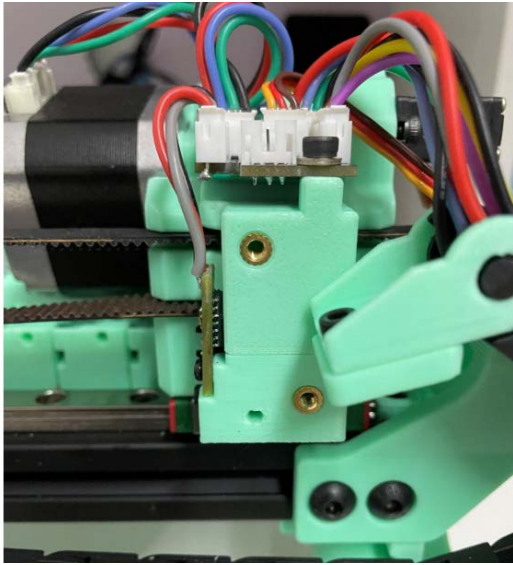


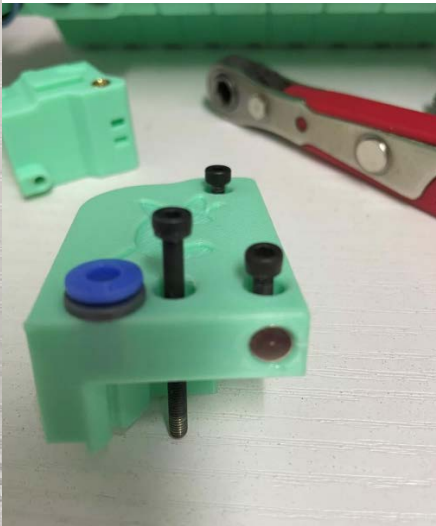
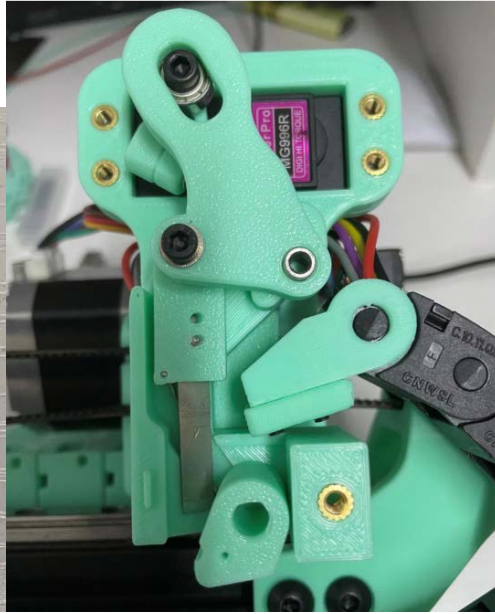
## 5、兔子传感器模块-切刀组装-大 C



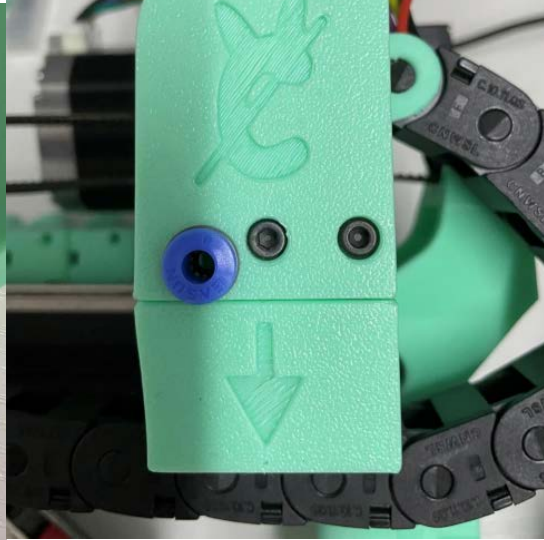
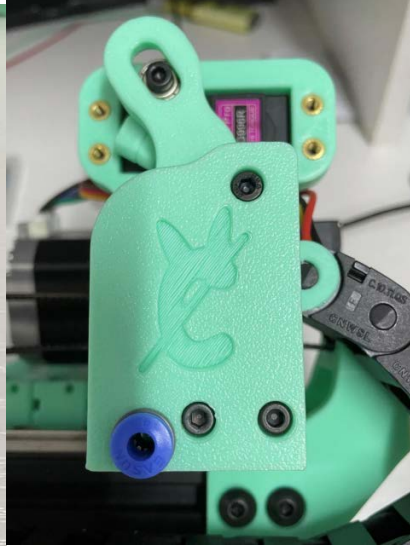












6、安装线轨及线轨限位不想写了抽时间继续

## 7、组合车头，传感器，皮带模块



## 8、组合通道（完成硬件安装，不含理线及限位安装）

### 三、兔子主板刷机及调试教程-群友贡献为主不做主动更新

#### 1、 兔子主板刷机

1) 确认主控类型为 xiao

2) 按照要求下载 xiao 编译文件（兔子文档内部），并进行初步编译

```
sudo apt install libreadline-dev libwxgtk3.0-*  
git clone https://github.com/shumatech/BOSSA.git  
cd BOSSA  
make  
sudo cp bin/bossac /usr/local/bin
```

1、插件补全

输入命令：sudo apt install libreadline-dev libwxgtk3.0-\*

2、下载主文件(如果遇到下载错误或是网络错误使用另外一条命令)

输入命令：git clone https://github.com/shumatech/BOSSA.git

如果上面命令网络错误使用下面这条命令

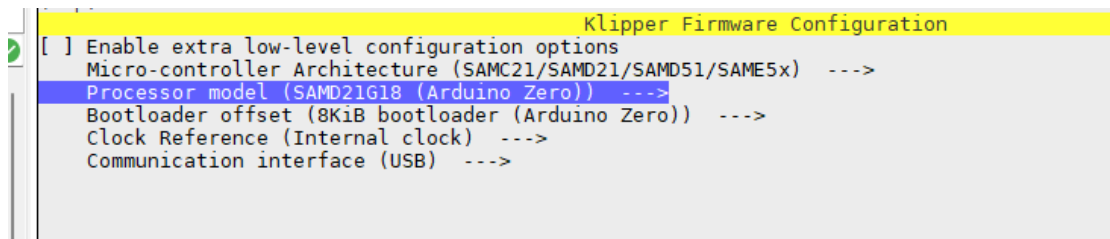
输入命令：git clone https://ghproxy.com/https://github.com/shumatech/BOSSA.git

- 3、进入文件夹内部  
输入命令：cd BOSSA
- 4、编译  
输入命令：make
- 5、将编译使用的命令放入默认路径  
输入命令：sudo cp bin/bossac /usr/local/bin

### 3) 进入 klipper 文件夹进行固件编译

```
cd ~/klipper  
make clean  
make menuconfig
```

- 1、进入 klipper 目录  
输入命令：cd ~/klipper
- 2、清楚原始配置文件  
输入命令：make clean
- 3、开始进入配置  
输入命令：make menuconfig
- 4、配置内容如下

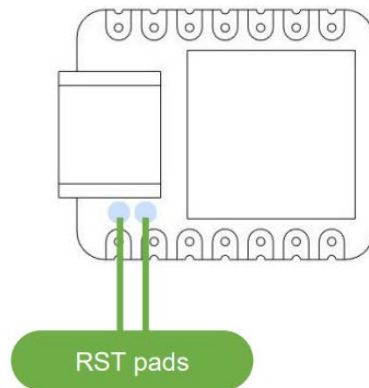


- 5、编译  
输入命令：Make
- 6、编译完成后的文件位置在（确认编译文件是否生成）  
~/klipper/out/klipper.bin

### 4) 刷机教程

- 1、编译完成开始刷机-上电短接 xiao 的两个引脚出现橙色灯闪烁进入刷机模式





- 2、查询 xiao 主板在上位机的位置或是端口, 无 linux 基础建议直接拔掉上位机其他 USB 设备只接 xiao 主控 (兔子板), 此处只需要关注 >>ttyACM\* 处的值即可

输入命令: `ls -l /dev/serial/by-id/`

```
mjf@ubuntu:~/klippers$ ls -l /dev/serial/by-id/
total 0
lrwxrwxrwx 1 root root 13 Jul 15 22:48 usb-Klipper_samd21g18a_D72E79B54134555020312E30422617FF-if00 -> ../../ttyACM0
lrwxrwxrwx 1 root root 13 Jul 15 22:45 usb-Klipper_stm32f446xx_370038000F51303530323539-if00 -> ../../ttyACM2
mjf@ubuntu:~/klippers$
```

如果输入以上命令未查询到相关主板信息使用 `lsusb` 配合插拔主控查询是否有相关信息变化来确认主控状态。

输入命令: `lsusb`

```
mjf@ubuntu:~/klippers$ ls -l /dev/serial/by-id/
total 0
lrwxrwxrwx 1 root root 13 Jul 15 22:48 usb-Klipper_samd21g18a_D72E79B54134555020312E30422617FF-if00 -> ../../ttyACM0
lrwxrwxrwx 1 root root 13 Jul 15 22:45 usb-Klipper_stm32f446xx_370038000F51303530323539-if00 -> ../../ttyACM2
mjf@ubuntu:~/klippers$ lsusb
Bus 001 Device 052: ID 1d50:614e OpenMoko, Inc. stm32f446xx
Bus 001 Device 053: ID 1d50:614e OpenMoko, Inc. samd21g18a
Bus 001 Device 005: ID 046d:0825 Logitech, Inc. Webcam C270
Bus 001 Device 004: ID 1a40:0101 Terminus Technology Inc. Hub
Bus 001 Device 003: ID 0d8c:0014 C-Media Electronics, Inc. Audio Adapter (Unitek Y-247A)
Bus 001 Device 002: ID 8087:07e6 Intel Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
mjf@ubuntu:~/klippers$
```

如果通过以上两条命令都查询不到相关信息, 建议先换跟 USB 线吧。

- 3、刷机

主板已正常接入上位机, 同时已在上一步确认好了 ttyACM 编号

输入命令: `sudo /usr/local/bin/bossac -i -d -p /dev/ttyACM1 -e -w -v -R --offset=0x2000 out/klipper.bin`

- 4、刷机结束后通过 `ls -l /dev/serial/by-id/` 查询是否存在带有 klipper 字眼的 usb 设备  
如果存在证明刷机成功, 否则意味着失败

输入命令: `ls -l /dev/serial/by-id/`

```
mjf@ubuntu:~/klippers$ ls -l /dev/serial/by-id/
total 0
lrwxrwxrwx 1 root root 13 Jul 15 22:48 usb-Klipper_samd21g18a_D72E79B54134555020312E30422617FF-if00 -> ../../ttyACM0
lrwxrwxrwx 1 root root 13 Jul 15 22:45 usb-Klipper_stm32f446xx_370038000F51303530323539-if00 -> ../../ttyACM2
mjf@ubuntu:~/klippers$
```

## 2、 配置文件更新-xiao 或是 RP2040

[https://github.com/1314mjf521/TradRack\\_Beta-main-](https://github.com/1314mjf521/TradRack_Beta-main-)

mod/tree/main/%E5%88%87%E5%88%80-%E5%BF%AB%E4%B9%90%E5%85%94%E9%85%8D%E7%BD%AE%E5%8F%82%E8%80%83

### 3、 TR 或是 TR 兔调试（群友更新不做具体介绍-可参考兔子调试教程）

#### 1) 舵机调试

a)

#### 2) 选择电机及送料电机调试

a)

#### 3) TR 送料调试

a)

#### 4) TR 编码器调试

a)

#### 5) 完整调试

a)

## 6) 开始浪

## 四、快乐兔安装教程——QQ: 865233470 (雪妹儿 ~喵 )

### 1、 快乐兔安装

```
cd -
git clone https://github.com/moggieuk/ERCF-Software-V3.git
cd ERCF-Software-V3
./install.sh -i
```

```
cd ~
git clone https://github.com/moggieuk/ERCF-Software-V3.git
cd ERCF-Software-V3
./install.sh -i
```

### 2、可能出现的问题：

- 1) Are you using the EASY-BRD or Fysetc Burrows ERB controller? (y/n)?n  
是否使用兔子板： 否  
Sensorless selector operation? This allows for additional selector recovery steps  
Enable sensorless selector operation (y/n)?n  
是否使用无传感器模式： 否
- 2) How many gates (selectors) do you have (3, 6, 9, 12)?  
Number of gates? 12  
装几色? :12
- 3) Have you built with a selector bypass? (in one of the selector separators)  
Selector bypass (y/n)?n  
是否启用旁路模式? : 否
- 4) Do you have a toolhead sensor you would like to use? If reliable this provides the smoothest and most reliable loading and unloading operation  
是否使用退料检测： 是  
What is the mcu pin name that your toolhead sensor is connected too?  
If you don't know just hit return, I can enter a default and you can change later  
Toolhead sensor pin name? ABC123  
退料的 pin 编号： 随便填装完后再改
- 5) Clog detection? This uses the ERCF encoder movement to detect clogs and can call your filament runout logic  
Enable clog detection (y/n)?Y

是否启用堵头检测： 是

6) Would you like ERCF to automatically adjust clog detection length (recommended)?  
Automatic (y/n)?Y

是否使用自动检测堵头长度： 是

7) What is the length of your reverse bowden tube in mm?

(This is just to speed up calibration and needs to be approximately right but not longer than the real length)

Reverse bowden length in mm? 456

换色到打印头的特氟龙管长度，随便填比实际值小就行

8) Finally, would you like me to include all the ERCF config files into your printer.cfg file  
Add include? (y/n)?Y

是否自动引用多色配置文件： 是

Would you like to include Mini 12864 menu configuration extension for ERCF

Include menu (y/n)?N

9) 是否使用小屏幕的菜单： 否（若有小屏幕也可以开）

### 3、装完软件后的一些调试语句

ps：直接备份原来的 cfg 把群主的 cfg 文件替换进行可省你一大部分时间，尽管可能引脚与你的不对

装载一段耗材	ERCF_TEST_LOAD	
卸载耗材	ERCF_TEST_UNLOA	
	D	
挤出 200	ERCF_TEST_MOVE_GE AR	（挤出 200 测量实际长度，用下方计算器进行计算实际值,注意黄色的是自动计算的公式不能改，只需要改绿色的地方）
挤出计算	rotation_distance	旧 rotation_distance
	36.83895993	36.75259134
校准编码器精度	ercf_calibrate_encoder	填写到 ercf_hardware.cfg 的 encoder_resolution
参数	文件	说明
parking_distance	ercf_parameters.cfg	从编码其到待机位置的距离
calibration_bowden_length	ercf_parameters.cfg	从编码器大致到打印头的距离（校准时使用）
ercf_calib_clog_length	ercf_vars.cfg	从编码器精确到打印头的距离（校准生成）
ercf_selector_offsets	ercf_vars.cfg	每个色块到零点的距离

enable_clog_detection	ercf_parameters.cfg	开启堵料检测 1 普通 2 自动
sync_to_extruder	ercf_parameters.cfg	同步挤出
servo_up_angle	ercf_parameters.cfg	舵机抬起角度
servo_down_angle	ercf_parameters.cfg	舵机下压角度
colorselector	ercf_parameters.cfg	好像是每个耗材块的位置，但是好像改了也没用
ercf_selector_offsets	ercf_vars.cfg	好像是每个耗材块的位置，但是好像改了有用
宏指令	说明	
ERCF_CALIBRATE（不推荐使用）	完整校准：校准编码器到打印头距离以及每个口的编码器补偿值	
ercf_eject	退出耗材到待机位	
ercf_servo_down	舵机下压	
ercf_servo_up	舵机抬起	
ERCF_MOTOR_S_OFF	关闭多色电机	
ERCF_LOAD	加载耗材	
ERCF_SYNC_GEAR_MOTOR sync=1 servo=1	同步模式开启	
ERCF_SYNC_GEAR_MOTOR sync=0 servo=1	同步模式关闭	

## 4、软件部分：快乐兔屏幕

进入 kiauh `./kiauh/kiauh.sh`

注意数字可能因版本的不同而不一致，请以自己的 kiauh 为准

卸载 2  
卸载 KlipperScreen 7  
返回 B  
安装 1  
安装 KlipperScreen 5  
返回 B  
退出 Q  
安装快乐兔屏幕

cd ~

mv KlipperScreen KlipperScreen.orig

git clone <https://github.com/moggieuk/KlipperScreen-Happy-Hare-Edition.git>

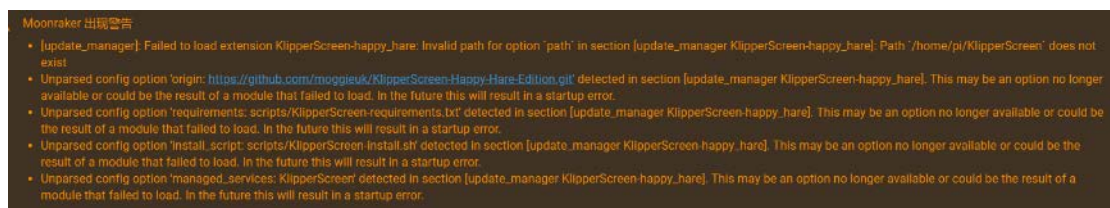
KlipperScreen

cd ~/KlipperScreen/happy\_hare

注意把颜色数量 12 改为实际数量

./install\_ks.sh -g 12

## 1、出现如图警告



moonraker.conf 中找到段: update\_manager KlipperScreen-happy\_hare

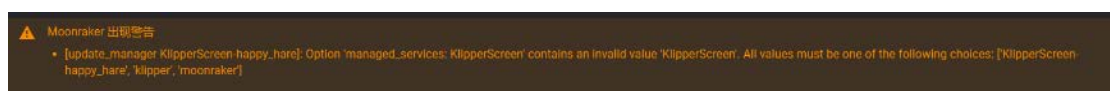
将 path 和 env 中的路径改为实际路径

mks 为 :

path: /home/mks/KlipperScreen

env: /home/mks/.KlipperScreen-env/bin/python

## 2、出现如图错误



moonraker.conf 中找到段: update\_manager KlipperScreen-happy\_hare

整段直接注释掉, 想要更新软件时再取消注释, 或者屏蔽掉永不出现

## 五、 KlipperScreen 安装教程群友贡献-QQ: 1546685250

一、安装脚本: `git clone https://github.com/th33xitus/kiauh`

二、使用脚本进行安装，在这里就不进行赘述了

三、KlipperScreen 常见问题修复：

如果您在安装 KlipperScreen 后没有看到 KlipperScreen 显示，可尝试以下解决方法。

视频版教程：<https://www.bilibili.com/video/BV1Ud4y1d7kR/>

## 1. 关闭系统图形界面

这里以 Ubuntu20.04 为例

Ubuntu20.04 的默认桌面环境管理服务是 gdm3，停止 gdm3 就关闭了图形界面 (GUI)。

在不关机情况下关闭和打开图形界面

```
sudo systemctl stop gdm3          # 在的当前会话中关闭图形界面
```

设置开机默认进入 GUI 或命令行

```
sudo systemctl set-default multi-user.target    # 设置开机默认  
进入命令行  
sudo reboot
```

## 2. 检查 xserver-xorg-legacy 是否正确安装

```
sudo apt-get install xserver-xorg-legacy      # 安装 xserver-xorg-  
legacy  
sudo systemctl restart KlipperScreen.service  # 重启  
KlipperScreen 服务
```

## 3. 查看日志

如果您的屏幕仍未启动，请尝试使用 `journalctl -xe -u KlipperScreen` 以查看问题所在。

如果日志以这样的方式结束：

```
Oct 26 10:55:17 raspberrypi systemd[1]: KlipperScreen.service: Main  
process exited, code=exited, status=1/FAILURE  
Oct 26 10:55:17 raspberrypi systemd[1]: KlipperScreen.service:  
Failed with result 'exit-code'.  
Oct 26 10:55:18 raspberrypi systemd[1]: KlipperScreen.service:  
Service RestartSec=1s expired, scheduling restart.
```

```
Oct 26 10:55:18 raspberrypi systemd[1]: KlipperScreen.service:
Scheduled restart job, restart counter is at 14.
Oct 26 10:55:18 raspberrypi systemd[1]: Stopped KlipperScreen.
Oct 26 10:55:18 raspberrypi systemd[1]: Started KlipperScreen.
Oct 26 10:55:18 raspberrypi xinit[768]: (EE)
Oct 26 10:55:18 raspberrypi xinit[768]: Fatal server error:
Oct 26 10:55:18 raspberrypi xinit[768]: (EE) Server is already
active for display 0
Oct 26 10:55:18 raspberrypi xinit[768]: #011If this server is no
longer running, remove /tmp/.X0-lock
这意味着它缺少权限。
```

使用此命令为其授予权限: `sudo gpasswd -a username tty`  
如果 klipperscreen 仍然不启动, 请继续使用 `journalctl -xe -u KlipperScreen` 查看问题。  
如果日志中有这样的东西

```
Oct 26 15:24:55 raspberrypi xinit[1466]: (EE) xf86OpenConsole:
Cannot open virtual console 2 (Permission denied)
需要修改/etc/X11/Xwrapper.config, 将下面两行代码写入 Xwrapper.config
文件中。
needs_root_rights=yes
allowed_users=anybody
不出意外过一会就能显示
```

## 4、安装驱动：（以下为 78 平板使用不适用其它文件）

```
apt install gcc automake autoconf libtool make
#驱动编译（注意：内核更新后需要重新编译驱动）
cd ~
git clone https://ghproxy.com/https://github.com/onitake/gslx680-acpi.git
cd gslx680-acpi
make
如果出现：/lib/modules/6.1.0-9-amd64/build: 没有那个文件或目录。:
sudo apt-get install build-essential //install build-essential(optional)

sudo apt-get update //install linux-headers
sudo apt-get install linux-headers-$(uname -r)
make install
depmod -a
rmmod silead
modprobe gslx680_ts_acpi
#触摸屏参数
```



```
cd ~
git clone https://ghproxy.com/https://github.com/onitake/gsl-firmware.git
wget -P gsl-firmware/tools -N http://code.imnks.com/z3735/gsl_ts.fw
cd gsl-firmware/tools
./fwtool -c gsl_ts.fw -m 1680 -w 960 -h 600 -t 10 -f xflip,track silead_ts.fw
cp -f silead_ts.fw /lib/firmware
#参数修改后重启才生效
Reboot
```

#### 4、 屏幕翻转：

```
sudo nano /usr/share/X11/xorg.conf.d/90-monitor.conf
```

修改如下内容并保存：

```
Section "Monitor"
    Identifier "DSI-1"
    # This identifier would be the same as the name of the connector printed by
xrandr.
    # it can be "HDMI-0" "DisplayPort-0", "DSI-0", "DVI-0", "DPI-0" etc

    Option "Rotate" "left"
    # Valid rotation options are normal,inverted,left,right

    Option "PreferredMode" "800x1280"
    # May be necessary if you are not getting your preferred resolution.
EndSection
```

Ctrl+x 输入 y， 然后 enter

#### 5、 开机自动登录：

```
sudo vi /lib/systemd/system/getty@.service
修改字段 ExecStart
```

```
ExecStart=-/sbin/agetty -o '-p -- \u' --noclear %l $TERM
如下：
```

```
ExecStart=-/sbin/agetty --autologin [yourusername] --noclear %l $TERM
```

把[yourusername] 替换为你需要自动登录的用户名。

保存退出， 并重启电脑。看看是否生效。

#### 6、 安装中文语言包

1、先查看是否有中文语言环境

```
locale -a
```

结果会显示你的电脑已经安装的语言环境：

```
en_US.ISO-8859-1
```

```
zh_CN.GBK
```

2、安装语言环境（root 权限）：

```
sudo dpkg-reconfigure locales
```

3、安装中文字体

```
sudo apt-get install ttf-wqy-zenhei
```

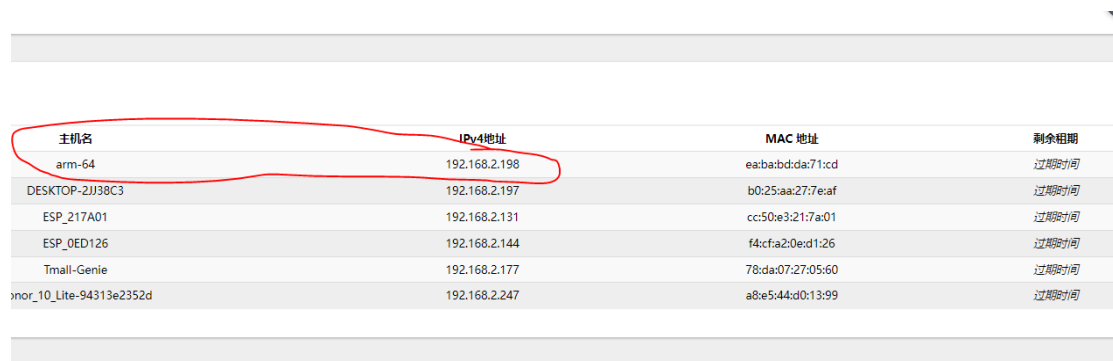
```
sudo apt-get install xfonts-intl-chinese wqy*
```

## 六、Klipper 配置 wifi 和下位机-唔，待定(484785558);

### 1、设置 wifi

#### step1：设置会话并登录

给路由器上电插上网线，路由器管理界面出现一个主机 arm-64，对应 ip 填到 ssh 软件的远程主机上面



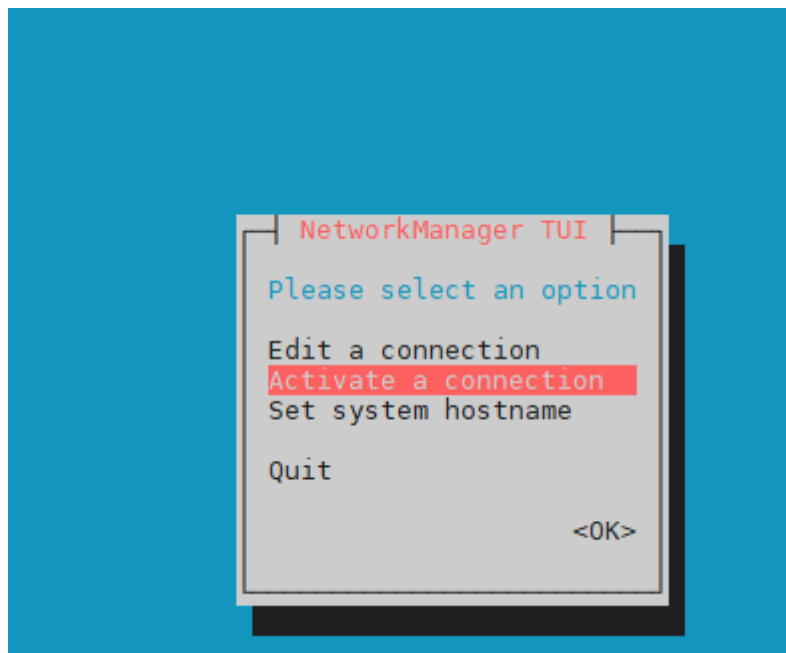
主机名	IPv4地址	MAC 地址	剩余租期
arm-64	192.168.2.198	ea:ba:bd:da:71:cd	过期时间
DESKTOP-2JJ38C3	192.168.2.197	b0:25:aa:27:7e:af	过期时间
ESP_217A01	192.168.2.131	cc:50:e3:21:7a:01	过期时间
ESP_0ED126	192.168.2.144	f4:cf:a2:0e:d1:26	过期时间
Tmall-Genie	192.168.2.177	78:da:07:27:05:60	过期时间
ynor_10_Lite-94313e2352d	192.168.2.247	a8:e5:44:d0:13:99	过期时间

（也可以使用 Windows 自带的 ssh 连接，输入“ssh [klipper@192.168.2.198](mailto:klipper@192.168.2.198)”，输入密码）

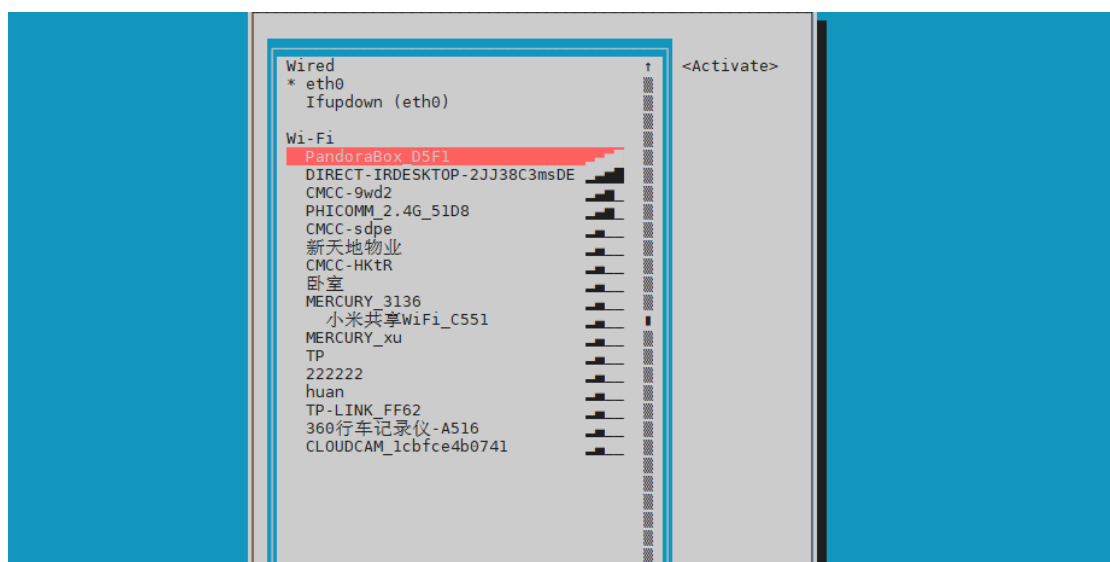
注 1：地址改为自己查询到的地址

注 2：输入密码时，密码不会显示

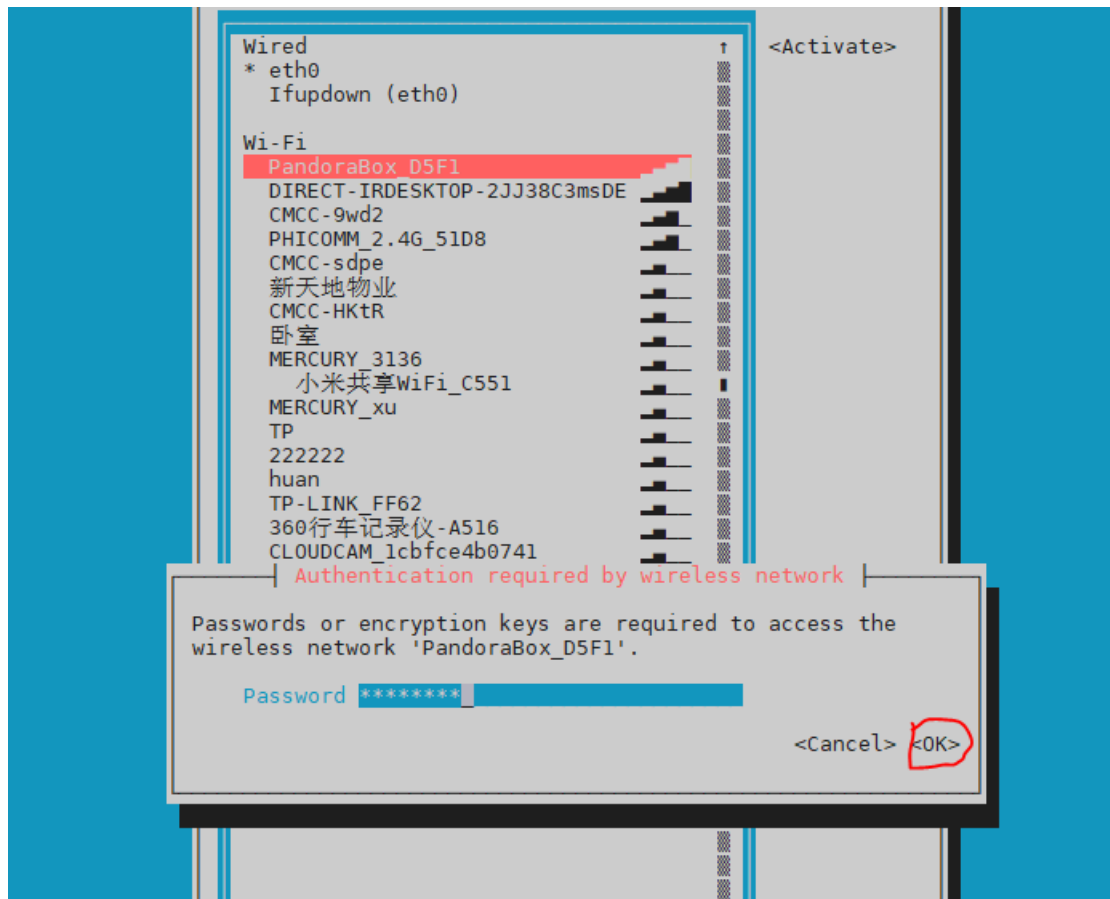




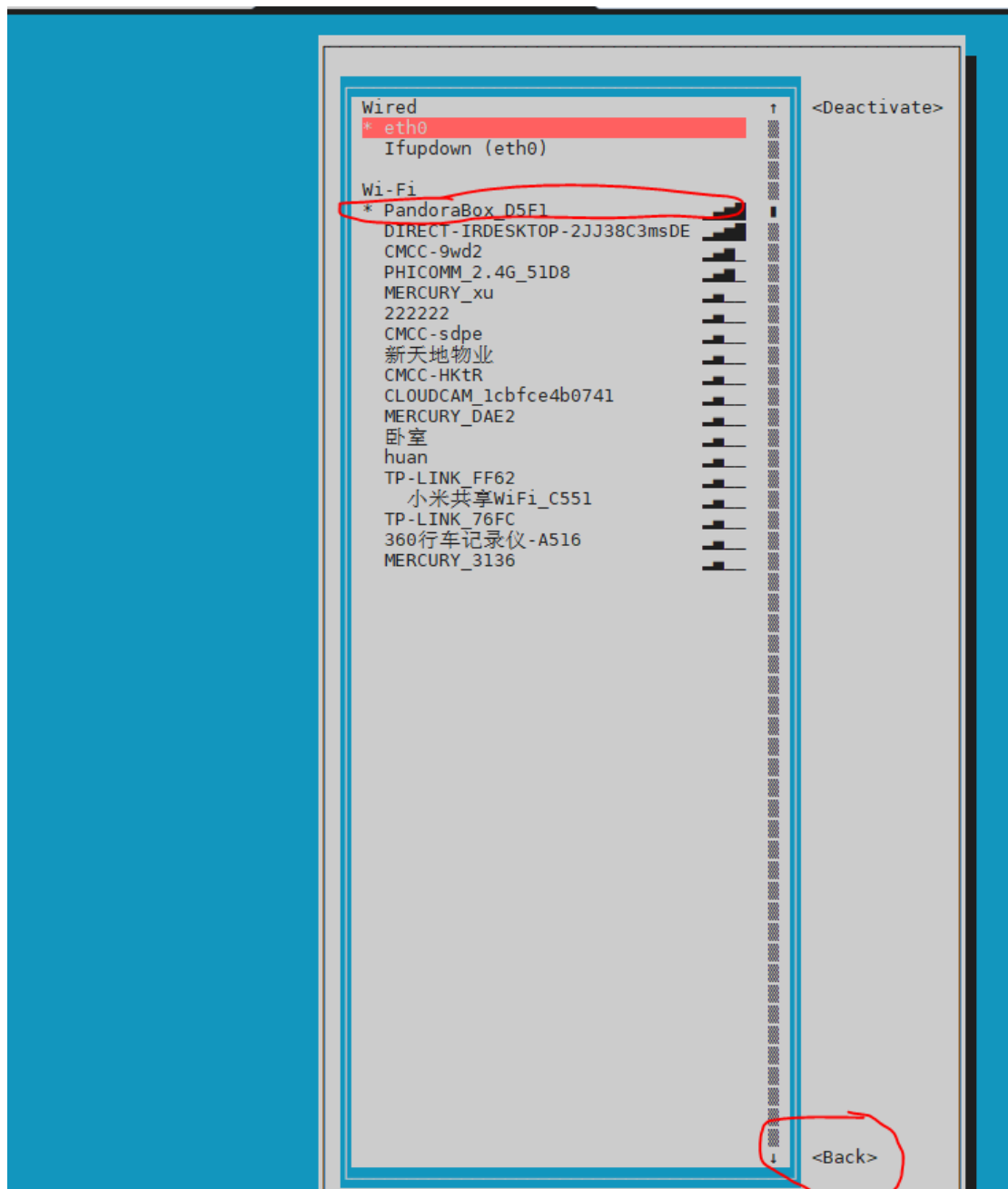
选择 activate a connection 激活一个网络，回车



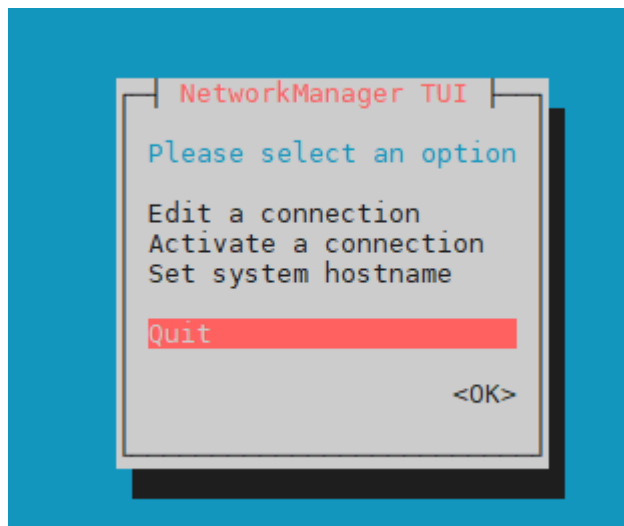
选择对应的 wifi 热点，回车后



输入 wifi 密码，选择下方 ok 按钮，回车



热点名前方出现\*号代表连上，选择 back，回车后如下



选择 quit, 回车退出

```
> For more info, ctrl+click on help or visit our website

ARMbian
Welcome to Armbian 20.10 Focal with Linux 5.9.0-arm-64
No end-user support: built from trunk

System load:  2%           Up time:       19 min
Memory usage: 18% of 924M  IP:           192.168.2.198
CPU temp:     75°C        Usage of /:    42% of 6.7G

[ General system configuration (beta): armbian-config ]

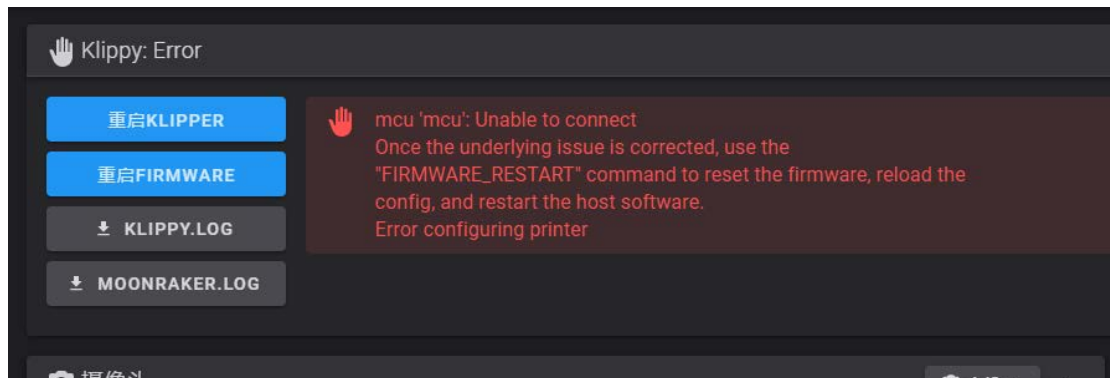
Last login: Fri Jul 29 06:17:33 2022 from 192.168.2.197
klipper@arm-64:~$ nmtui
klipper@arm-64:~$ sudo reboot
```

退出后输入命令 `sudo reboot` 回车重启, 立即拔掉网线 可自动连接到 wifi

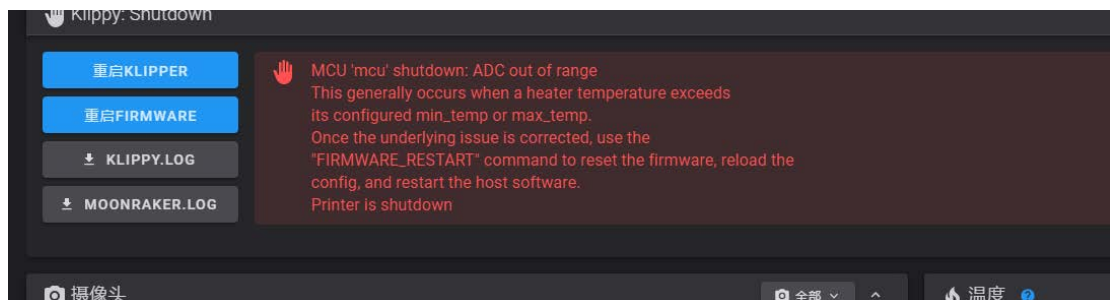
发现没有权限执行该代码 `permission denied` 拒绝访问  
我们回到 sftp 会话

## 2、klipper 和主板常见连接错误

连接主板后点击 重启 firmware 如下图错误



表示打印机主板没有刷入固件或者 usb 端口设置有问题，简而言之串口连不是错误



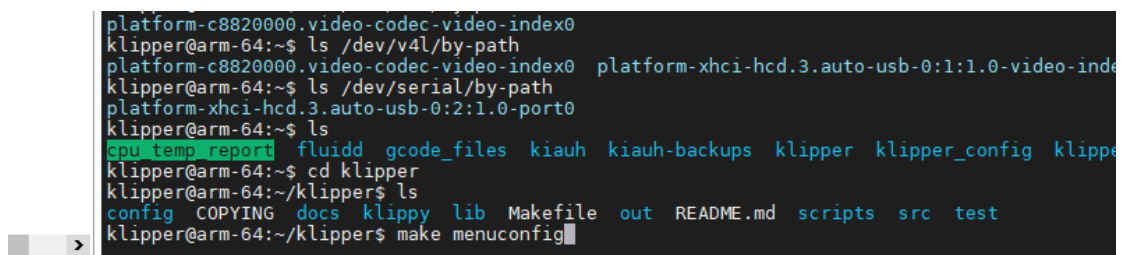
超出 adc 范围表示打印机串口没有问题，打印机温度传感器常见就是 ntc，超出检测范围，一般就是打印机正确端口上没有插温度传感器 或者温度传感器设置错误

## 3、编译主板固件

也可以自己编译固件

ssh 登陆后

- 1、输入 cd klipper
- 2、再输入 make menuconfig



回车后出现下图（如果显示不全 在第一行回车一下就可以展开了）



```
(Top)
Klipper Firmware Configuration
[*] Enable extra low-level configuration options
    Micro-controller Architecture (LPC176x (Smoothieboard)) ---->
    Processor model (lpc1768 (100 MHz)) --->
[*] Target board uses Smoothieware bootloader (NEW)
    Communication interface (USB) --->
    Specify a custom step pulse duration
    () GPIO pins to set at micro-controller startup

[Space/Enter] Toggle/enter    [?] Help    [/] Search
[Q] Quit (prompts for save)  [ESC] Leave menu
```

```
mks@mkspi: ~/klipper
(Top)
Klipper Firmware Configuration
[*] Enable extra low-level configuration options
    Micro-controller Architecture (Atmega AVR) ---->
    Processor model (atmega2560) --->
    Processor speed (16Mhz) --->
    Communication interface (UART0) --->
(250000) Baud rate for serial port (NEW)
() GPIO pins to set at micro-controller startup

[Space/Enter] Toggle/enter    [?] Help    [/] Search
[Q] Quit (prompts for save)  [ESC] Leave menu
```

以上为 mks gen v 2.1 的下位机写配置  
如上设置 q 退出并保存  
会弹出一个对话框直接按键盘 y 代表 yes

这样配置好了编译目标

3、输入 make 如下

```
cpu temp report fluidd gcode_files kiauh kiauh-backups klipper
klipper@arm-64:~$ cd klipper
klipper@arm-64:~/klippers$ ls
config COPYING docs klippy lib Makefile out README.md scri
klipper@arm-64:~/klippers$ make menuconfig
  Creating symbolic link out/board
Loaded configuration '/home/klipper/klipper/.config'
Configuration saved to '/home/klipper/klipper/.config'
klipper@arm-64:~/klippers$ make
```

回车后会进行编译

```
klipper@arm-64:~/klippers$ make menuconfig
  Creating symbolic link out/board
Loaded configuration '/home/klipper/klipper/.config'
Configuration saved to '/home/klipper/klipper/.config'
klipper@arm-64:~/klippers$ make
  Creating symbolic link out/board
Building out/autoconf.h
Compiling out/src/sched.o
Compiling out/src/command.o
Compiling out/src/basecmd.o
Compiling out/src/debugcmds.o
Compiling out/src/initial_pins.o
Compiling out/src/gpiocmds.o
Compiling out/src/stepper.o
Compiling out/src/endstop.o
Compiling out/src/trsync.o
Compiling out/src/adccmds.o
Compiling out/src/spicmds.o
Compiling out/src/thermocouple.o
Compiling out/src/i2ccmds.o
Compiling out/src/pwmcmds.o
Compiling out/src/spi_software.o
Compiling out/src/sensor_adxl345.o
Compiling out/src/lcd_st7920.o
Compiling out/src/lcd_hd44780.o
Compiling out/src/buttons.o
Compiling out/src/tmcuart.o
Compiling out/src/neopixel.o
Compiling out/src/pulse_counter.o
Compiling out/src/lpc176x/main.o
Compiling out/src/lpc176x/gpio.o
Compiling out/src/generic/armcm_boot.o
Compiling out/src/generic/armcm_irq.o
Compiling out/src/generic/armcm_timer.o
Compiling out/src/generic/armcm_reset.o
Compiling out/src/generic/crc16_ccitt.o
Compiling out/src/./lib/lpc176x/device/system_LPC17xx.o
Compiling out/src/lpc176x/adc.o
Compiling out/src/lpc176x/i2c.o
Compiling out/src/lpc176x/spi.o
Compiling out/src/lpc176x/usbserial.o
Compiling out/src/lpc176x/chipid.o
Compiling out/src/generic/usb_cdc.o
Compiling out/src/lpc176x/hard_pwm.o
Building out/compile_time_request.o
Version: v0.9.1-784-g0b918b35
Preprocessing out/src/generic/armcm_link.ld
Linking out/klipper.elf
Creating bin file out/klipper.bin
klipper@arm-64:~/klippers$
```

注：弹出 meke done 表面编译完成 与上图不一致 上图是另一种方案

4、编译结束后输入“`ls /dev/serial/by-id/*`”查询下位机端口，这一步一定要先接好下位机，并将查询结果复制出来记录（接下来会用到）

5、关闭 klipper 服务“`sudo service klipper stop`”

6、输入“`make flash FLASH_DEVICE=`”，等号后面为步骤 4 的查询结果。

回车 刷下位机结束

7、输入“`sudo shutdown -h now`”关闭上位机。关机结束后断电断线再开机。

之后再网页输入你的上位机的 ip 地址即可访问

注：电脑和上位机需要再同一个局域网

## 方案二 对于不能用上位机直接刷入的主板

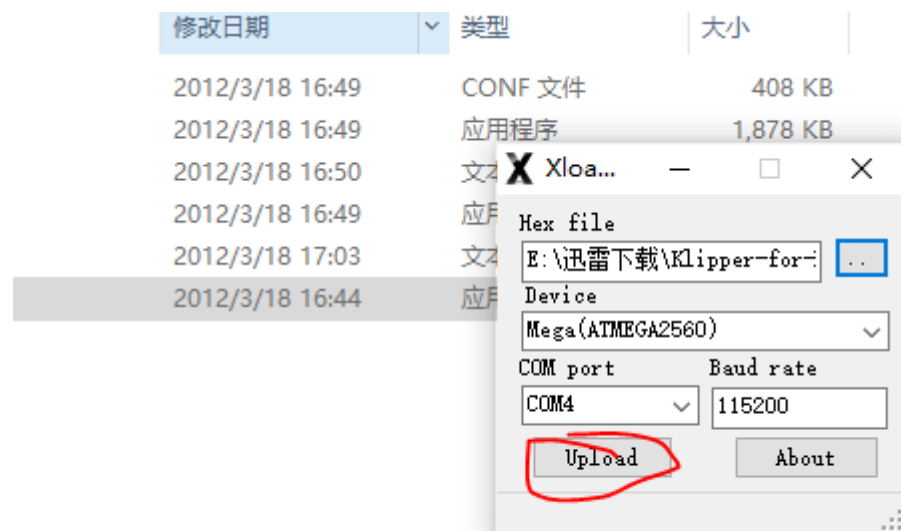
bin 文件为 klipper.bin, 目录在当前目录下的 out 目录, 前面 sftp 可以直接把文件下载下来, 一般名称改为对应名称 sgen 应该改为 firmware.bin ,mks robin nano 改 Robin\_Nano3.5.bin, 这就和 bootloader 能识别的 bin 文件名称有关

hex 文件直接用 xloader 软件下载到主板,

过程如下, 查看主板 usb 在电脑端口号, 设备管理器端口查看找



可以看出端口 com4



配置如上点 upload 下载 hex 文件到 avr 系列主板