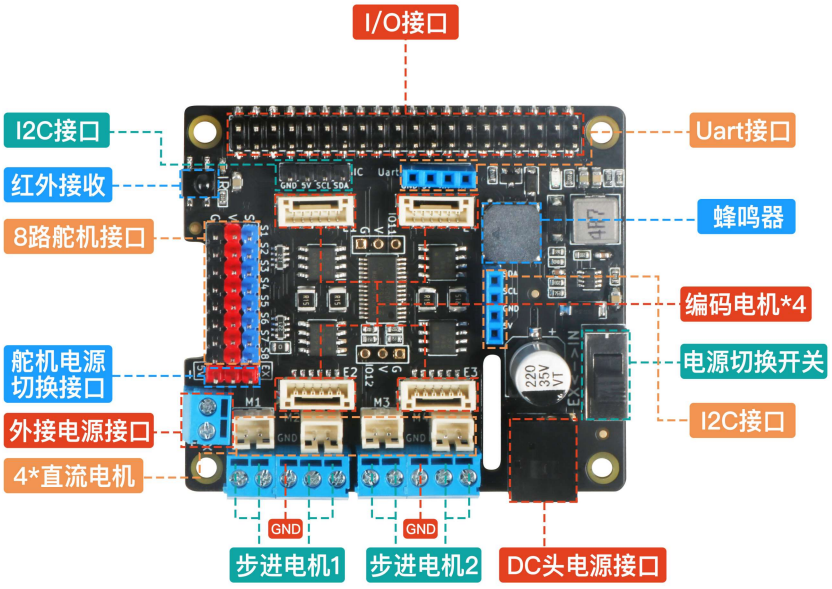
树莓派驱动板

RaspberryPi 多功能电机驱动扩展板由**深圳市易创空间科技有限公司**出品的一款全功能的机器人电机驱动扩展版，目前已经升级到**V4.0**（**V3.0老版本资料查看**） 本电机驱动板适用于Raspberry Pi Zero/Zero W/Zero WH/A+/B+/2B/3B/3B+/4B。能够同时支持多路电机/步进电机/舵机/编码电机(Stepper/Motor/Servo/Encoder)，空出摄像头和DIP显示屏排线接口，并且可以多板层叠使用扩展出更多的控制接口，特别适合玩家DIY机器人,智能小车,机械手臂,智能云台等各种应用。



原理图

由于我们驱动板是使用I2C控制PCA9685芯片输出16路PWM，所有驱动直流电机或者舵机，不存在所谓的树莓派IO口和控制电机对应关系

详情可以看 **树莓派驱动板电路原理图**

还可以查看驱动板正反面的丝印标注。

特点

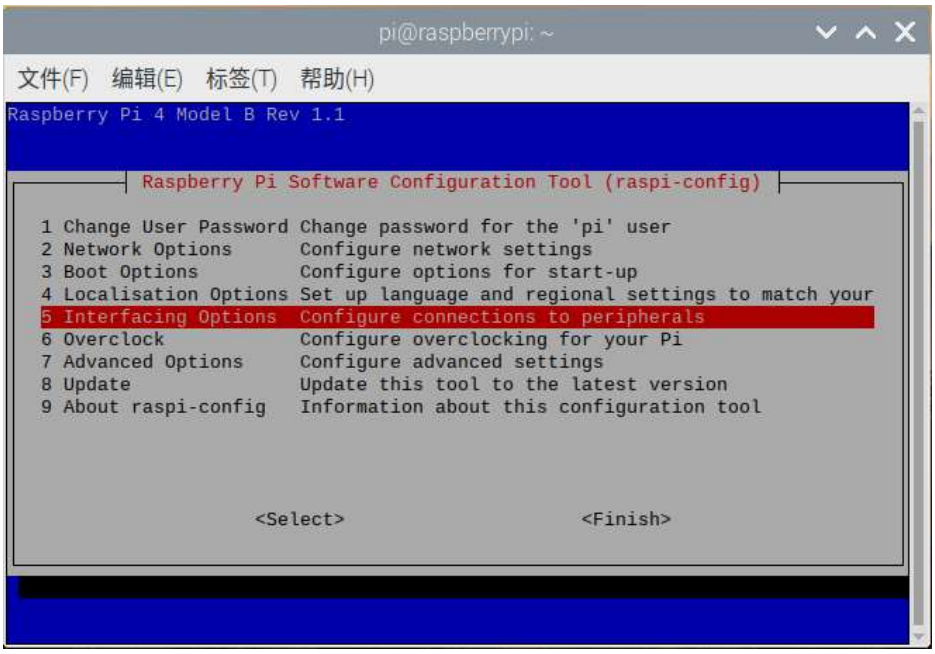
- 5.5 ~ 2.1mmDC头 供电电压6 ~ 25V，内置DC-DC稳压电路，为Raspberry Pi供电3A以上 (建议使用7.4V航模电池)

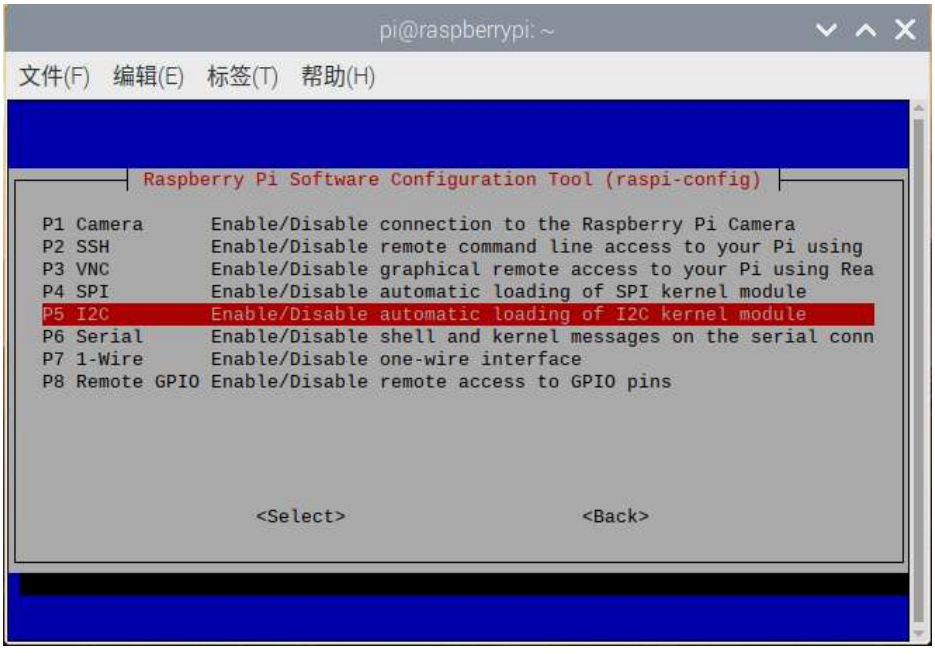
- 驱动板IIC地址为0x60，地址可以由背面6个电阻配置地址A0~A5
- 12位分辨率，可调PWM频率高达1.6KHz，可配置的推挽或开漏输出
- 支持同时驱动8路舵，3Pin(黑红蓝GVS)标准接口接线，方便连接舵机，舵机电源可通过跳线帽切换到外部独立供电
- 支持4路6~24V直流电机，PH2.0接口或者3.5mm接线柱，电机单路输出高达电流3A
- 支持同时驱动2路4线步进电机
- 板载无源蜂鸣器
- 主板预留2个IIC扩展接口，1个串口接口

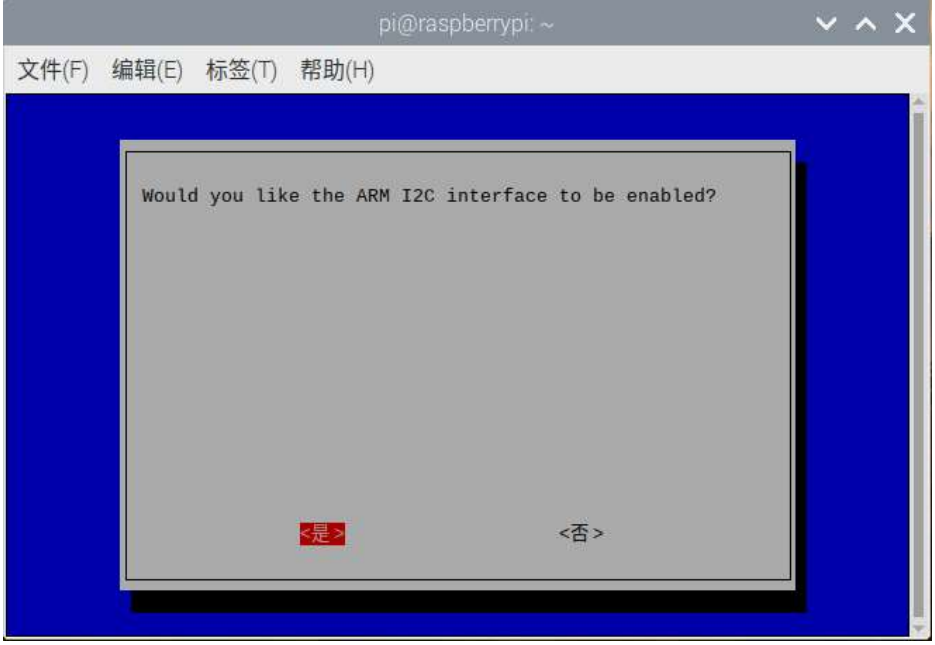
安装I2C库并使能

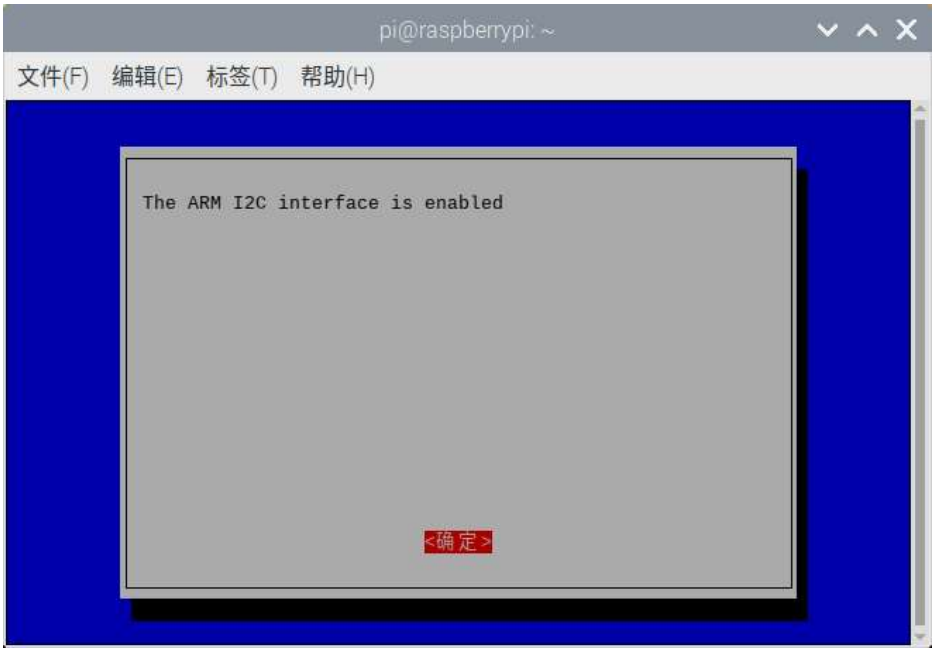
在使用驱动板之前，必须要先安装I2C库并使能。

打开树莓派终端输入 `sudo raspi-config` 命令，然后按照下图顺序依次操作即可。

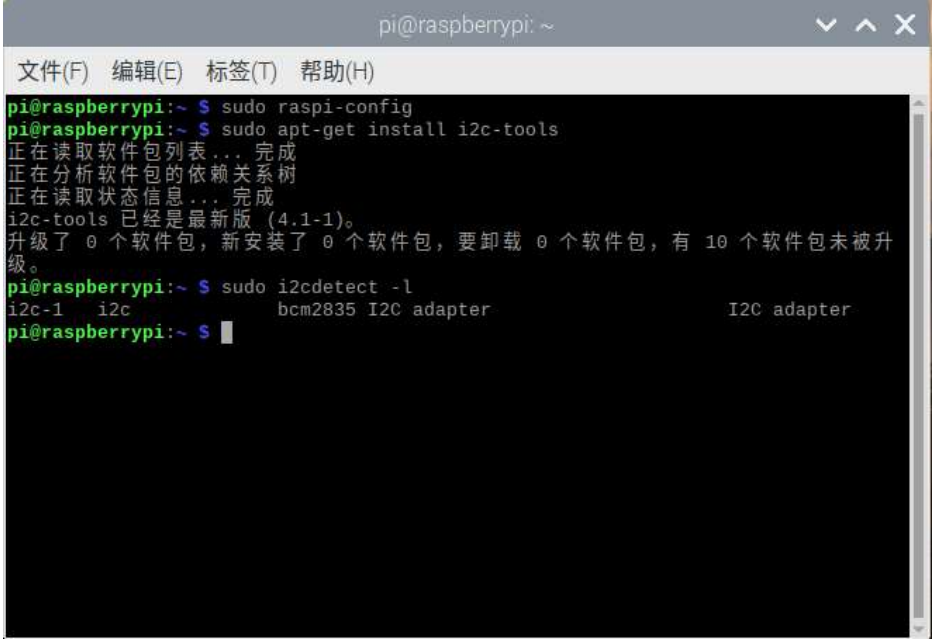






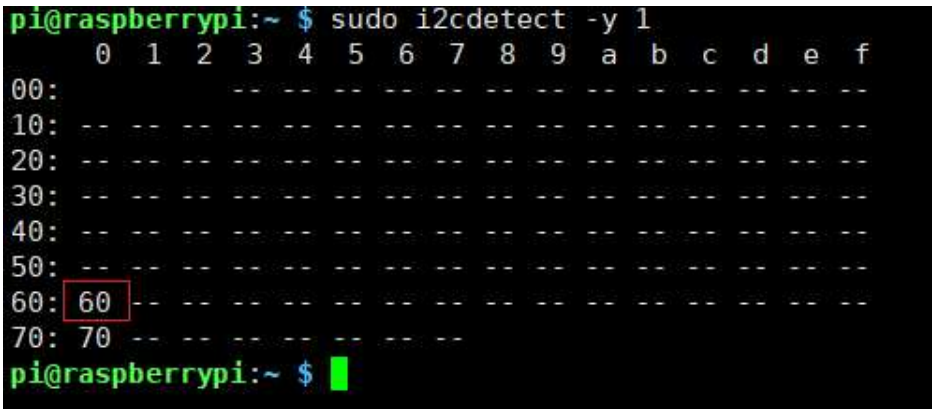


以上就是开启树莓派I2C，接下来我们安装树莓I2C库在终端入 `sudo apt-get install i2c-tools` ，输入完成后就可以看到正在下载I2C库，安装完成之后可以在终端输入 `sudo i2cdetect -l` 检测是否安装正确，若出现类似于下面的信息就说明安装正常。



在终端输入 `sudo i2cdetect -y 1` 命令即可扫描接在I2C总线上的所有I2C设备，并打印出该设备的I2C总线地址，且我们的扩展板的I2C地址为0x60，如下图。

另外用i2cdetect检测出还有一个0x70地址一直存在，这是一个通用地址，可以给所有从机下达指令



重新启动树莓派，使新的设置生效:

```
sudo reboot
```

功能介绍

驱动舵机

C++代码

C++

```
#include "Emakefun_MotorShield.h"

int main() {
  Emakefun_MotorShield pwm = Emakefun_MotorShield();
  pwm.begin(50);

  // demo这里只操作舵机1, 其他舵机操作相同
  Emakefun_Servo *my_servo_1 = pwm.getServo(1);

  // 速度值是 1 ~ 10 的正整数, 数值越大速度越快
  int speed = 9;
  while (true) {
    // demo这里只操作舵机1, 其他舵机操作相同
    my_servo_1->writeServo(0, speed);
    delay(2000);
    my_servo_1->writeServo(90, speed);
    delay(2000);
    my_servo_1->writeServo(180, speed);
    delay(2000);
  }
}
```

Python代码

Python

```
#!/usr/bin/python

from Emakefun_MotorHAT import Emakefun_MotorHAT, Emak
import time

mh = Emakefun_MotorHAT(addr=0x60)
my_servo = mh.getServo(1)

# 速度值是 1 ~ 10 的正整数, 数值越大速度越快
speed = 9

while (True):
    # demo这里只操作舵机1, 其他舵机操作相同
    my_servo.writeServoWithSpeed(0, speed)
    time.sleep(1)

    my_servo.writeServoWithSpeed(90, speed)
    time.sleep(1)
```

```
    my_servo.writeServoWithSpeed(180, speed)
    time.sleep(1)
```

驱动直流电机

C++代码

C++

```
#include "Emakefun_MotorShield.h"

int main() {
  Emakefun_MotorShield pwm = Emakefun_MotorShield();
  pwm.begin(50);
  Emakefun_DCMotor *dc_motor_1 = pwm.getMotor(1);
  Emakefun_DCMotor *dc_motor_2 = pwm.getMotor(2);
  Emakefun_DCMotor *dc_motor_3 = pwm.getMotor(3);
  Emakefun_DCMotor *dc_motor_4 = pwm.getMotor(4);

  dc_motor_1->setSpeed(255);
  dc_motor_2->setSpeed(255);
  dc_motor_3->setSpeed(255);
  dc_motor_4->setSpeed(255);

  while (1) {
    dc_motor_1->run(FORWARD);
    dc_motor_2->run(FORWARD);
    dc_motor_3->run(FORWARD);
    dc_motor_4->run(FORWARD);
    delay(1000);
    dc_motor_1->run(BACKWARD);
    dc_motor_2->run(BACKWARD);
    dc_motor_3->run(BACKWARD);
    dc_motor_4->run(BACKWARD);
    delay(1000);
  }
}
```

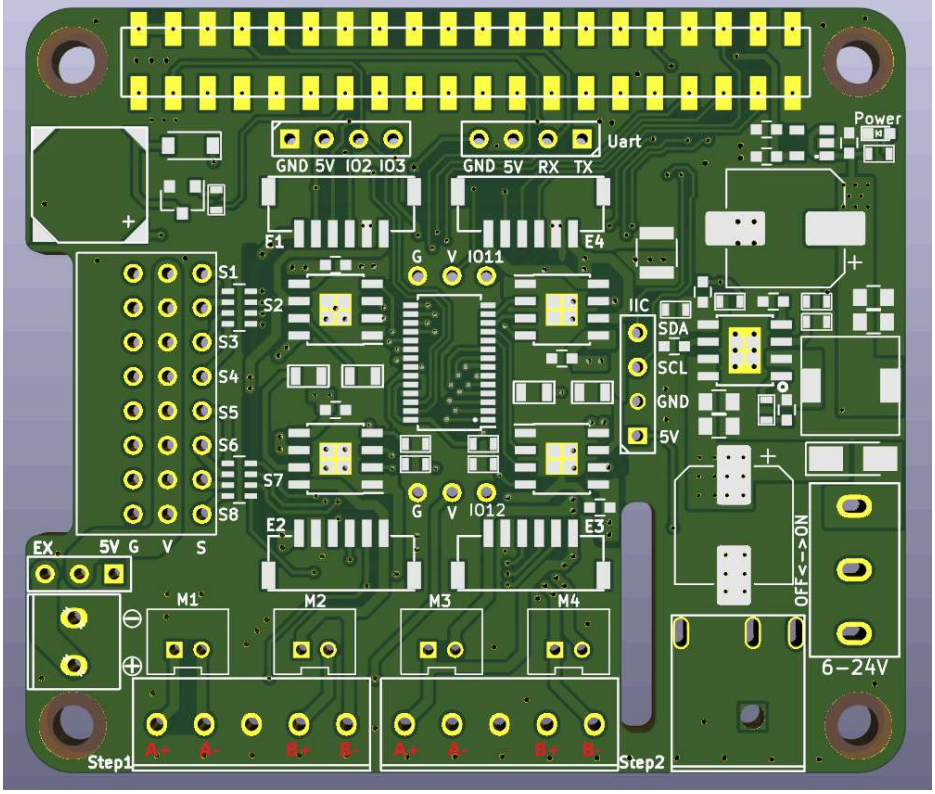
Python代码

```python
#!/usr/bin/python
from Emakefun_MotorHAT import Emakefun_MotorHAT, Emake

import time
import atexit

# create a default object, no changes to I2C address o
mh = Emakefun_MotorHAT(addr=0x60)

# recommended for auto-disabling motors on shutdown!
def turnOffMotors():
    mh.getMotor(1).run(Emakefun_MotorHAT.RELEASE)
    mh.getMotor(2).run(Emakefun_MotorHAT.RELEASE)
    mh.getMotor(3).run(Emakefun_MotorHAT.RELEASE)
    mh.getMotor(4).run(Emakefun_MotorHAT.RELEASE)

atexit.register(turnOffMotors)

################################# DC motor test!
my_motor = mh.getMotor(4)

# set the speed to start, from 0 (off) to 255 (max spe
my_motor.setSpeed(150)
my_motor.run(Emakefun_MotorHAT.FORWARD);
# turn on motor
my_motor.run(Emakefun_MotorHAT.RELEASE);

while (True):
    # demo这里只操作舵机1, 其他舵机操作相同
    my_servo.writeServoWithSpeed(0, speed)
    time.sleep(1)

    my_servo.writeServoWithSpeed(90, speed)
    time.sleep(1)

    my_servo.writeServoWithSpeed(180, speed)
    time.sleep(1)
while (True):
    print ("Forward! ")

    print ("\tSpeed up...")
    for i in range(255):
        my_motor.setSpeed(i)
        my_motor.run(Emakefun_MotorHAT.FORWARD)
        time.sleep(0.01)
```

```
print ("\tSlow down...")
for i in reversed(range(255)):
    my_motor.setSpeed(i)
    my_motor.run(Emakefun_MotorHAT.FORWARD)
    time.sleep(0.01)

print ("Backward! ")

print ("\tSpeed up...")
for i in range(255):
my_motor.setSpeed(i)
my_motor.run(Emakefun_MotorHAT.BACKWARD)
time.sleep(0.01)

print ("\tSlow down...")
for i in reversed(range(255)):
my_motor.setSpeed(i)
my_motor.run(Emakefun_MotorHAT.BACKWARD)
time.sleep(0.01)

print ("Release")
my_motor.run(Emakefun_MotorHAT.RELEASE)
time.sleep(1.0)
```

驱动步进电机

本驱动板只支持42步进电机，接线方法如下：



步进电机要驱动顺畅不卡顿，必须将树莓派I2C速度设置为400K，具体操作步骤如下，

- 打开终端并编辑配置文件(需要root权限) **/boot/config.txt**
- 找到以下行：

```
#dtparam=i2c_arm=on
```

- 将其取消注释并将其更改为：

```
dtparam=i2c_arm=on,i2c_arm_baudrate=400000
```

- 保存文件并重新启动树莓派

C++代码

c++

```
#include "Emakefun_MotorShield.h"

int main() {
  Emakefun_MotorShield pwm = Emakefun_MotorShield();
  pwm.begin(50);
  Emakefun_StepperMotor *stepper_motor_1 = pwm.getSte
```

```
  while (1) {
    stepper_motor_1->setSpeed(30);
    stepper_motor_1->step(100, BACKWARD, SINGLE);
  }
}
```

Python代码

python

```python
#!/usr/bin/python
#import Raspi_MotorHAT, Raspi_DCMotor, Raspi_Stepper
from Emakefun_MotorHAT import Emakefun_MotorHAT, Emake

import time
import atexit

# create a default object, no changes to I2C address o
mh = Emakefun_MotorHAT(0x60)

# recommended for auto-disabling motors on shutdown!
def turnOffMotors():
    mh.getMotor(1).run(Emakefun_MotorHAT.RELEASE)
    mh.getMotor(2).run(Emakefun_MotorHAT.RELEASE)
    mh.getMotor(3).run(Emakefun_MotorHAT.RELEASE)
    mh.getMotor(4).run(Emakefun_MotorHAT.RELEASE)

atexit.register(turnOffMotors)

my_stepper = mh.getStepper(200, 1)   # 200 steps/rev,
my_stepper.setSpeed(30)    # 30 RPM

while (True):
    print("Single coil steps")
    my_stepper.step(100, Emakefun_MotorHAT.FORWARD,  E
    my_stepper.step(100, Emakefun_MotorHAT.BACKWARD, E

    print("Double coil steps")
    my_stepper.step(100, Emakefun_MotorHAT.FORWARD,  E
    my_stepper.step(100, Emakefun_MotorHAT.BACKWARD, E

    print("Interleaved coil steps")
    my_stepper.step(100, Emakefun_MotorHAT.FORWARD,  E
    my_stepper.step(100, Emakefun_MotorHAT.BACKWARD, E

    print("Microsteps")
```

```
my_stepper.step(100, Emakefun_MotorHAT.FORWARD,  I
my_stepper.step(100, Emakefun_MotorHAT.BACKWARD, I
```

C#代码

有提供一个C# demo，有兴趣可以研究，不提供技术支持

点击下载上述代码

注意事项

1、连接驱动板后，检查树莓派IIC，看是否检测到驱动板地址。**链接**

2、执行代码时，需将Github上代码下载到本地执行，上面只是部分示例。**Github链接**

3、树莓派用普通锂电池供电可能会出现树莓派重启的问题，建议使用7.4V航模电池。

4、可用该驱动板可以驱动精度不高的42步进电机，驱动高精度步进电机时需接专用步进电机驱动。

5、该驱动板可接编码电机，代码正在更新.....