智达自动化

ZDAUTO-MIO-Raspberry Pi

树莓派 I/O 扩展板

标配模块组合

产品手册（V1.0）

www.zdauto.com

中山智达自动化科技有限公司

目录

1. 简介

Raspberry Pi(中文名为“树莓派”),它是一个只有信用卡大小的开发板，又称卡片式电脑，上面可运行 LINUX 系统，能替代日常桌面计算机的多种用途，包括文字处理、电子表格、媒体中心甚至是游戏。您可以将树莓派连接电视、显示器、键盘鼠标等设备使用。用户可以用它开发自己想要的设备。

ZDAUTO-MIO-Raspberry Pi 是一款兼容树莓派开发板的 I/O 扩展板，将主板 I/O 口引至扩展板重新统一编排，并加入 Smart Relay 模块，可选择开关量，脉冲量，模拟量等输入输出模块，供用户接入/驱动外围设备。

2. 产品特点

-扩展板有八个 M5S 模块基座，客户可根据功能需求选择模块，从而实现设计的功能；

-外部电路接入和切换的电压可以比树莓派上的额定电压更高；

-选择隔离模块，实现电路隔离，提高模块安全性；

3. 扩展板标准组合配置

-ZDAUTO-MIO-Raspberry Pi 扩展板 *1
（内含 8 个 M5S 基座，可接入 8 个 M5S 模块）

-M5S-BID0324B1 *1

-M5S-BOT03750D1b *2

-M5S-AOA03020D3Ab *1

-原装镊子（用于拔插 M5S 零件） *1

-原装跳线 *8

（如需空白扩展板，请与客服联系）

4. 引脚说明

4.1 扩展板

①：内部/外部提供 M5S 模块 7 号引脚电源（V+）

②：外部设备信号输入/输出口（每个 M5S 的引脚 8）

③：内部/外部提供 M5S 模块 6 号引脚电源（V-）

④：连接外部电路与内部电源(0V 公用地)

⑤：对应 8 个 M5S 模块的 1 号引脚（接入控制信号）

⑥：针脚连接部分

G：内部电源（公共地 V-）

V：内部电源（V+）

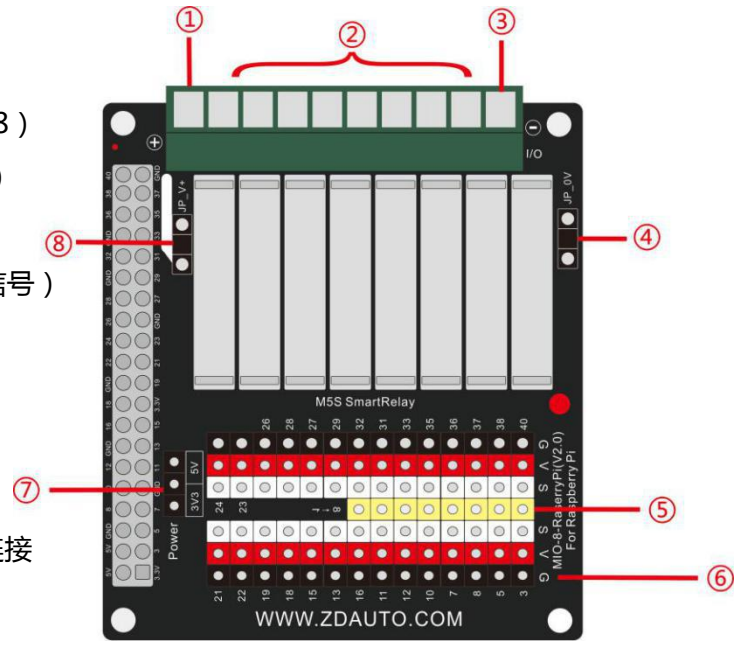
S：对应的树莓派 GPIO 编号信号脚

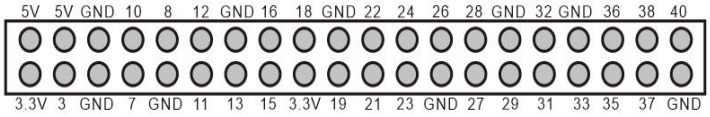
⑦：将树莓派的电压与所有 M5S 模块的 3 号引脚连接（M5S 供电）：

-引脚 1-2:3.3 V

-引脚 2-3:5v(不应使用!)

⑧：连接外部电路与内部电源（5V）

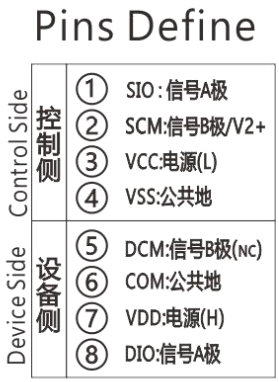


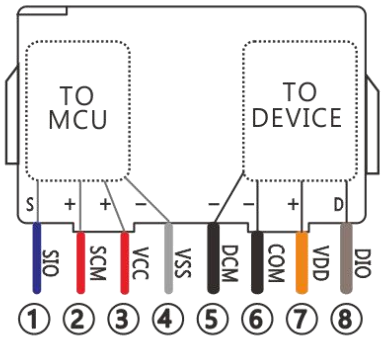


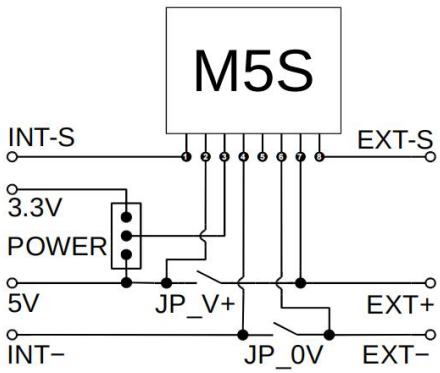
*注意：

④⑧：不能并联

4.2 M5S 模块引脚说明

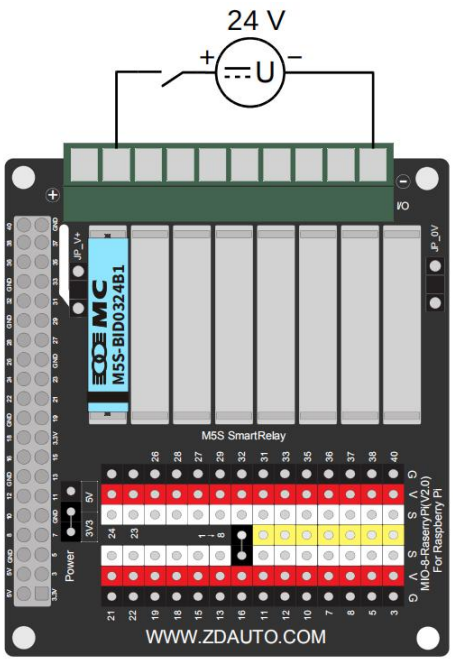
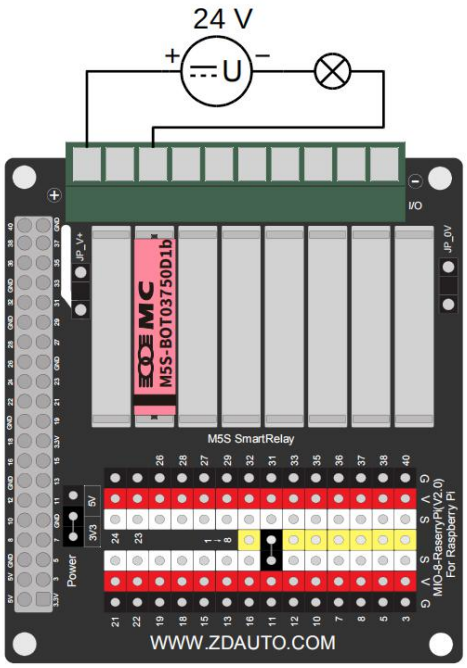
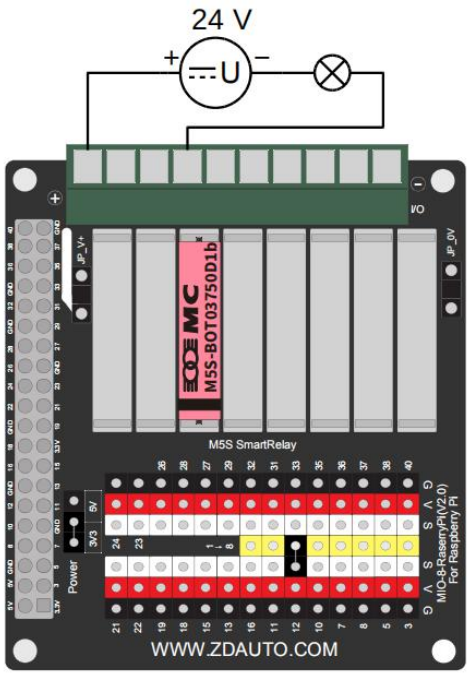
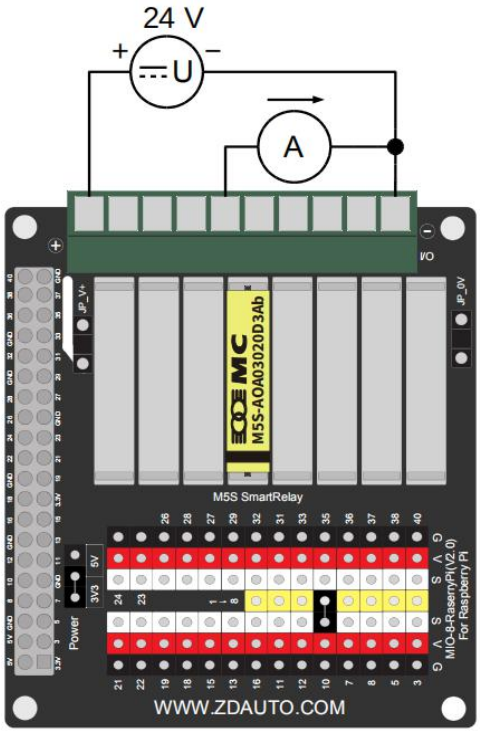






***M5S 模块器件有上百款型号，详细资料请联系客服。**

5. 扩展板接线说明图

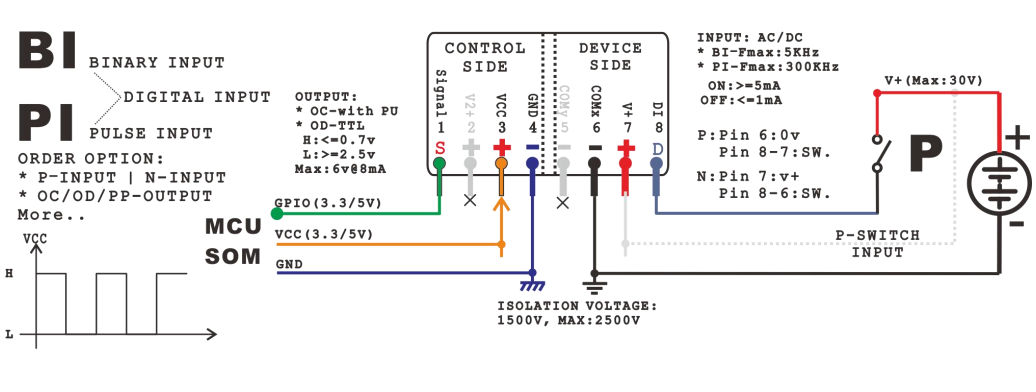
<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>检测 24v 直流电压
输入模块 M5S-BID0324B1 接线图
M5S 模块 1 连接树莓派引脚 16(GPIO23)</td>
<td>最大电压为 DC 24V，最大电流为 750mA
输出模块 M5S-BOT03750D1b 接线图
M5S 模块 2 连接树莓派引脚 11(GPIO17)</td>
</tr>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>最大电压为 24VDC ，最大电流为 750mA
输出模块 M5S-BOT03750D1b 接线图
M5S 模块 3 连接树莓派引脚 12(GPIO18)</td>
<td>输出模拟电流 0-20mA，电压为 24VDC
输出模块 M5S-AOA03020D3Ab 接线图
M5S 模块 4 连接树莓派引脚 10(GPIO15)</td>
</tr>
</table>

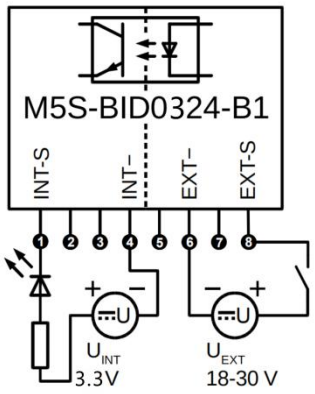
6. M5S 模块功能介绍

6.1 M5S-BID0324B1

光隔直流开关量源型输入(输入：DC,PNP 型,6-8 脚)，不带上拉电阻 NPN,OC 输出,1-4 脚)

型号	Control Side（控制侧，输出）				频率	隔离	Device Side（设备侧，输入）				电路索引
	电压(1)	电流(1)	电源(3)	极性			电压(8)	电流(8)	电源(7)	极性	
M5S-BID0324B1	H:3.3V L:0V	Max 8mA	x	N OC	0～ 5KH	●	24VDC	L:7mA H:1mA	x	P	B1

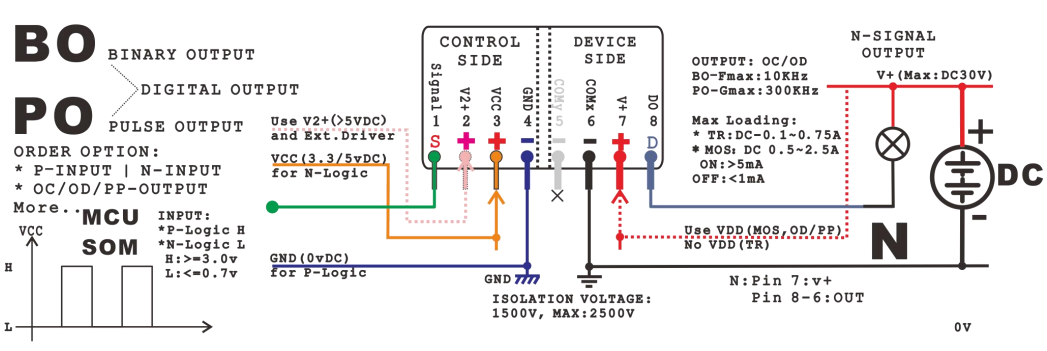


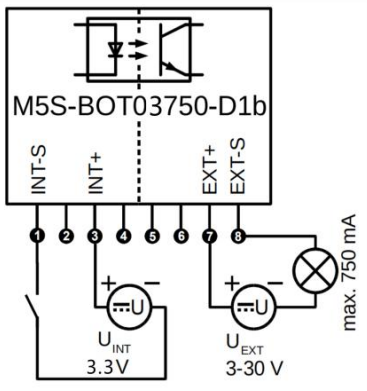


6.2 M5S-BOT03750D1b

光隔晶体管开关量源型输出(输入：1，3 脚 N 信号，7-8 脚 晶体管-OC-P，晶体集管电极开路输出)

型号	Control Side（控制侧，输入）				频率	隔离	Device Side（设备侧，输出）				电路索引
	电压(1)	电流(1)	电源	极性			电压(8)	电流(8)	电源	极性	
M5S-BOT03750D1b	ON:0V OFF:3V	Max 7mA	3.3V	N	0～ 20KHz	●	DC 24V	Max 750mA	x	P TOC	D1

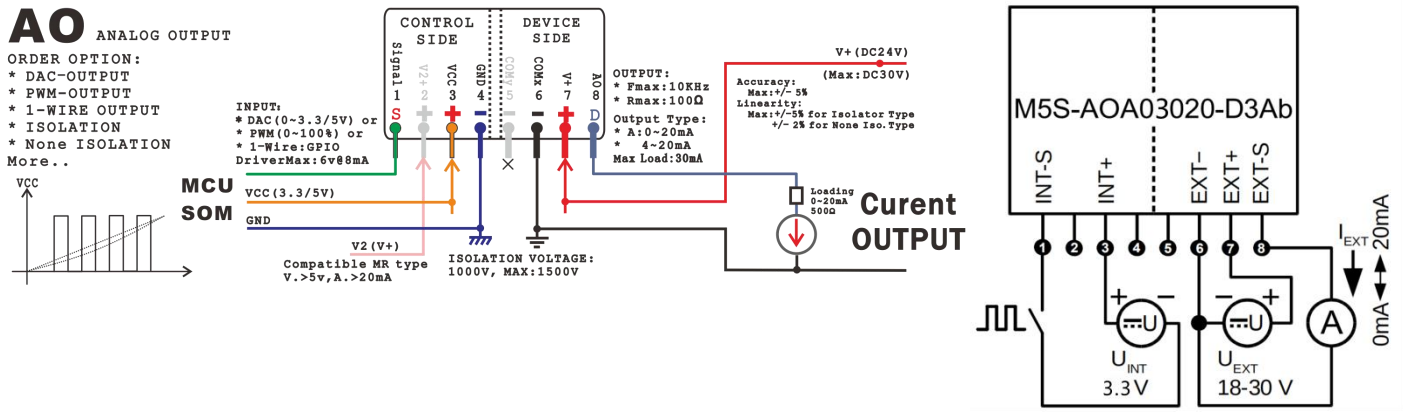




6.3 M5S-AOA03020D3Ab

光隔 PWM 型 电流模拟量输出(输入,1，3TTL 电平 PWM 信号， 输出：电压信号,6-7-8 脚）

型号	Control Side（控制侧，输入）				曲线	隔离	Device Side（设备侧，输出）				电路索引
	电压(1)	电流(1)	电源	精度			电压(8)	电流(8)	电源	频响	
M5S-AOA03020D3Ab	PWM	<5mA	3.3V	3%	线性	●		0~20mA	24V	1KHz	D3A



7. 扩展板的实际应用

此款 I/O 扩展板上带有 1 路 BI 开关量输入，2 路 BO 开关量输入，1 路 AO 模拟量输出模块，另外还可自行选择加入 4 路 M5S 模块。

因此可基于树莓派平台对 GPIO 点进行控制，与对应的 M5S 开关量输入输出模块信号脚连接，从而与外部设备连接，如开关量输入（接入开关、按钮旋钮、极限开关、水位开关、按键信号等），开关量输出（控制继电器、接触器、电磁阀、电热丝、灯、蜂鸣器、电机等）。同样的可以与对应的 M5S 模拟量输入输出模块信号脚连接，从而与外部设备连接，如模拟量输入（接入电位器、温度传感器、压力传感器），模拟量输出（控制马达速度、调节电炉温度、控制拉力等）。

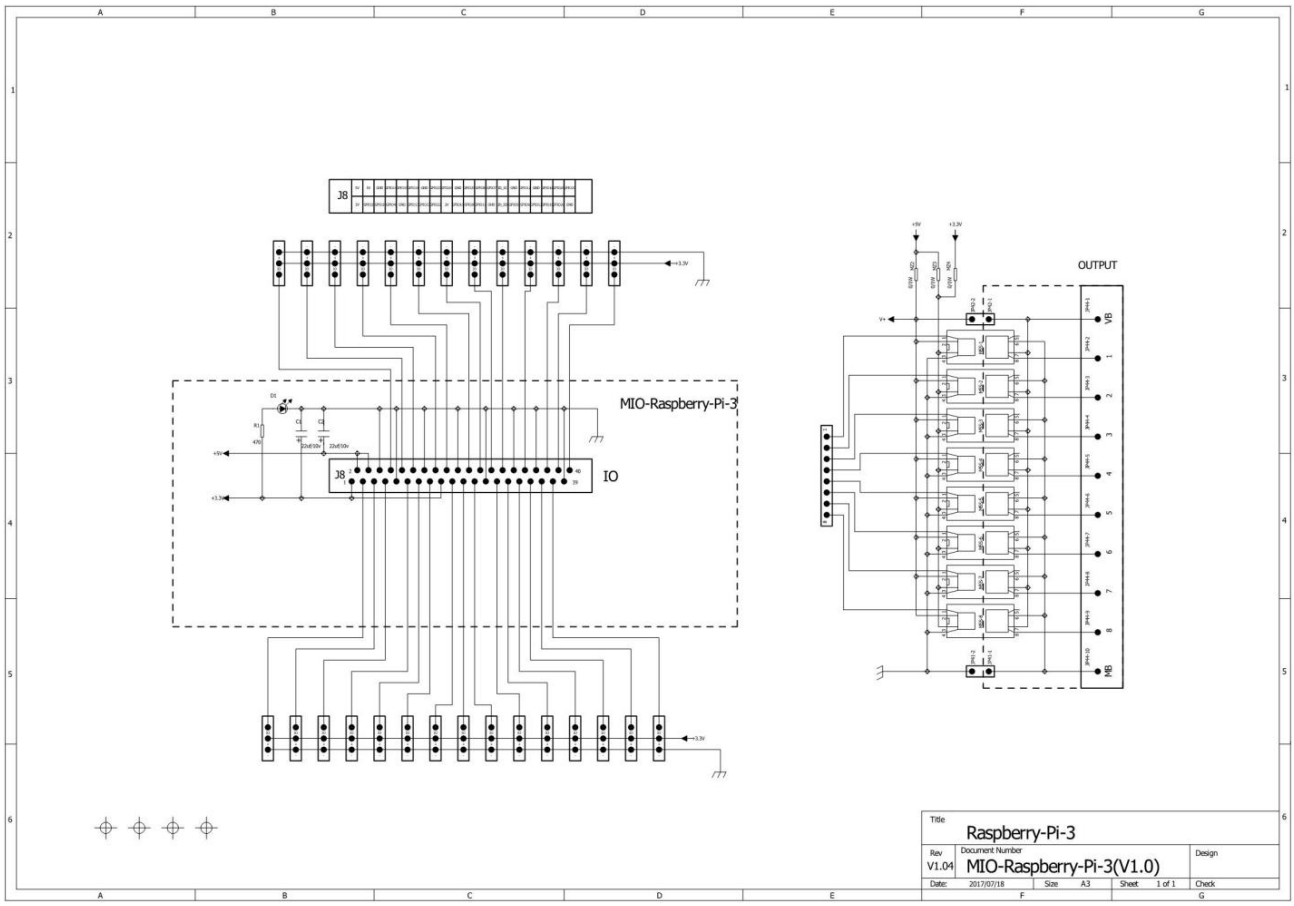
8. 各版本树莓派配置参数

型号	A 型	A+型	B 型	B+型	2 代 B 型	3 代 B 型
SOC	BroadcomBCM2835（CPU，GPU，DSP 和 SDRAM，USB）				Broadcom BCM2836	Broadcom BCM2837
CPU	ARM1176JZF-S 核心（ARM11 系列）700MHz				ARM Cortex-A7 (ARMv7 系列) 900MHz (四核心)	ARM Cortex-A53 1.2GHz 64-bit quad-core ARMv8 CPU
GPU	Broadcom VideoCrore IV，OpenGL ES 2.0,1080p 30 h.264/MPEG-4 AVC 高清解码器					
内存	256 MB（与 GPU 共享，可以理解为集成显卡的显存与内存共享）		512MB		1GB (LPDDR2)	
USB 2.0 接口 个数	1（支持 USB hub 扩展）		2	4		
视频输入	15-针头 MIPI 相机 (CSI) 界面，可被树莓派相机或树莓派相机(无红外线版)使用					
影像输出	RCA 视频接口输出（仅 1 代 B 型有此接口），支持 PAL 和 NTSC 制式，支持 HDMI (1.3 和 1.4)，分辨率为 640 x 350 至 1920 x 1200 支持 PAL 和 NTSC 制式。					
音源输出	3.5mm 插孔，HDMI 电子输出或 I²S					
板载存储	SD/MMC/SDIO 卡插槽	MicroSD 卡插槽	SD/ MMC / SDIO 卡插槽	MicroSD 卡插槽		
网络接口	无				10/100 以太网接口	• 10/100 以太网接口 • 802.11n Wireless LAN • Bluetooth 4.1 • Bluetooth Low Energy (BLE)

9. 扩展板引脚定义

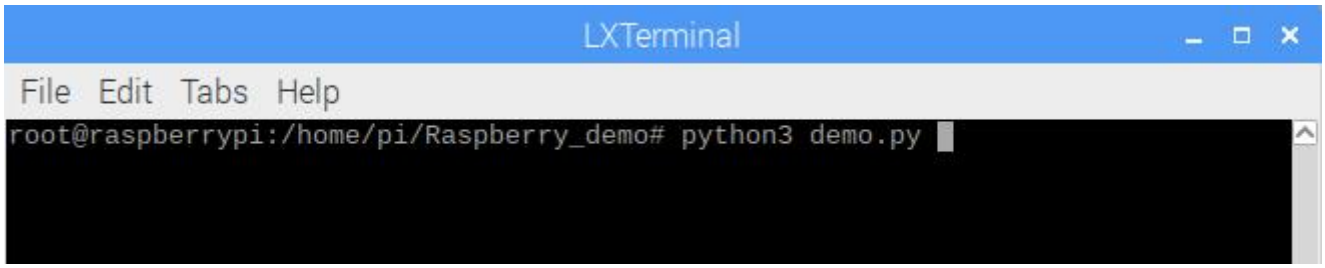
wringPi 编码	BCM 编码	功能名	物理引脚 BOARD 编码		功能名	BCM 编码	wringPi 编码
		3.3V	1	2	5V		
8	2	SDA.1	3	4	5V		
9	3	SCL.1	5	6	GND		
7	4	GPIO.7	7	8	TXD	14	15
		GND	9	10	RXD	15	16
0	17	GPIO.0	11	12	GPIO.1	18	1
2	27	GPIO.2	13	14	GND		
3	22	GPIO.3	15	16	GPIO.4	23	4
		3.3V	17	18	GPIO.5	24	5
12	10	MOSI	19	20	GND		
13	9	MISO	21	22	GPIO.6	25	6
14	11	SCLK	23	24	CE0	8	10
		GND	25	26	CE1	7	11
30	0	SDA.0	27	28	SCL.0	1	31
21	5	GPIO.21	29	30	GND		
22	6	GPIO.22	31	32	GPIO.26	12	26
23	13	GPIO.23	33	34	GND		
24	19	GPIO.24	35	36	GPIO.27	16	27
25	26	GPIO.25	37	38	GPIO.28	20	28
		GND	39	40	GPIO.29	21	29

10. MIO-Raspberry-Pi-3 接线图(清晰版请看附件)



11. 演示运行示例软件

示例程序下载地址：http://pan.baidu.com/s/1dET4CuP　密码：61vz



cd Raspberry_demo/ 进入程序文件夹；

python3 demo.py　开始运行程序。

程序的界面

开关量输入：

操作连接着 BI 模块的开关，程序就会受到信号使灯泡 UI 就会改变状态.

开关量输出：

鼠标或触摸屏点击开关控件便可控制 BO 输出高电平或低电平，从而控制 BO 模连接的灯泡的开关。

脉冲量输入：

扭动旋钮编码器，PI 检测到脉冲输入，旋钮计数便会更新

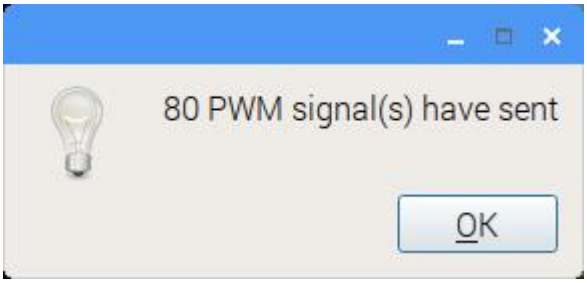
脉冲量输出：

如左图可以修改 PO 模块输出的脉冲量的频率、占空比、脉冲数。点击‘PWM 输出’便会开始输出指定参数的脉冲。

在脉冲输出的过程中可以点击下方两按钮控制电机旋转的方向和电机旋转的暂停/启动。

注意：由于本程序是用 python 语言编写，不能直接操作底层，因此脉冲量的输出频率越大，其误差会越大。

例如输出 80 个脉冲 输出完成后便会弹出输出完成提示的窗口。



12. 程序代码展示

12.1 初始化：

```
#导入所运行程序必须的库文件
import RPi.GPIO as gpio
import time,os
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtCore import Qt,QThread,pyqtSignal

gpio.setwarnings(False)

#-----------------------global_trigger-----------------------#

global pwm_num,pwm_count
pwm_num=0
pwm_count=0

#-------------------------gpio_setup--------------------------#
#根据需要分别将各个连接着 M5S 模块的 gpio 口设定为所需要的模式

gpio.setmode(gpio.BOARD)
gpio.setup(32,gpio.IN)                    #BI
gpio.setup(31,gpio.IN)

gpio.setup(33,gpio.IN)                    #PI

gpio.setup(37,gpio.OUT)                   #BO_lightbulb
gpio.output(37,1)
gpio.setup(36,gpio.OUT)
gpio.output(36,1)

gpio.setup(38,gpio.OUT)                   #BO_DIR
gpio.setup(40,gpio.OUT)                   #BO_ENA
```

12.2 PI、BI 的功能线程：

```
class BI_thread(QThread):
    def run(self):
        while True:
            b = gpio.input(32)
            c = gpio.input(31)
#根据 IO 口的对应状态更新 UI
            if b == 1:
                ui.lightbulb1.setPixmap(QtGui.QPixmap("img/light_off.jpg"))
                time.sleep(0.001)

            if b == 0:
                ui.lightbulb1.setPixmap(QtGui.QPixmap("img/light_on.jpg"))
                time.sleep(0.001)

            if c == 1:
                ui.lightbulb2.setPixmap(QtGui.QPixmap("img/light_off.jpg"))
                time.sleep(0.001)

            if c == 0:
                ui.lightbulb2.setPixmap(QtGui.QPixmap("img/light_on.jpg"))
                time.sleep(0.001)

class PI_thread(QThread):
    def run(self):
        count=0
        while True:
            a = gpio.input(33)
            if a == 0:
                count += 1                              #IO 口状态跳变一次计数值+1
                ui.lineEdit.setText(str(count))
```

12.3 BO 控制灯泡函数：

```
def light_control(self):

        icon = QtGui.QIcon()

        gpio.output(36,not gpio.input(36))
        if gpio.input(36) == 1:
            icon.addPixmap(QtGui.QPixmap("img/off.jpg"), QtGui.QIcon.Normal, QtGui.QIcon.Off)
            self.t1.setIcon(icon)
        else:
            icon.addPixmap(QtGui.QPixmap("img/on.jpg"), QtGui.QIcon.Normal, QtGui.QIcon.Off)
            self.t1.setIcon(icon)

    def light2_control(self):

        icon = QtGui.QIcon()

        gpio.output(37,not gpio.input(37))
        if gpio.input(37) == 1:
            icon.addPixmap(QtGui.QPixmap("img/off.jpg"), QtGui.QIcon.Normal, QtGui.QIcon.Off)
            self.t2.setIcon(icon)
        else:
            icon.addPixmap(QtGui.QPixmap("img/on.jpg"), QtGui.QIcon.Normal, QtGui.QIcon.Off)
            self.t2.setIcon(icon)
```

12.4 PO 控制电机的进程：

```
#----------------------------------------初始化----------------------------------------#
import RPi.GPIO as gpio
from PyQt5 import QtCore, QtGui, QtWidgets
import sys

gpio.setwarnings(False)

pwm_num=int(sys.argv[1])
pwm_count=0
s=0
gpio.setmode(gpio.BOARD)
gpio.setup(35,gpio.OUT)                          #PO
#---------------------------------------开始输出脉冲---------------------------------------#
pwm = gpio.PWM(35,int(sys.argv[2]))
pwm.start(int(sys.argv[3]))

while True:
    if gpio.input(35) == 1:
        pass
    else:
        s=1                                      #starting output pwm

    if gpio.input(35) == 1 and s    == 1: #detect pwm_end
        pwm_count += 1
        s=0
    if pwm_count == pwm_num:
        break

print("stop")
pwm.stop()
```