



Servo Driver HAT

用户手册

前言

本产品是基于树莓派而设计的 PWM/舵机扩展板，通过 PCA9685 芯片扩展 16 路舵机控制或者 PWM 输出，每个通道 12 位分辨率。通过 I2C 接口控制，无需占用额外的引脚。板载 5V 稳压芯片，可接电池供电，最大输出 3A 电流。适用于控制机械手臂，以及各种舵机机器人。

产品特性

输入电压 VIN：6V~12V

舵机电压：5V

逻辑电压：3.3V

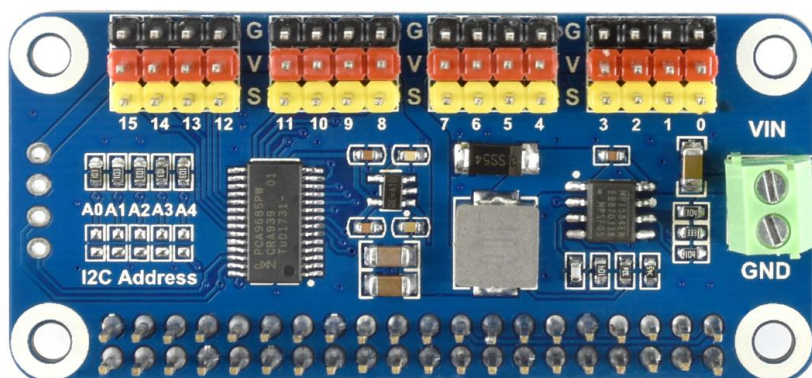
驱动芯片：PCA9685

控制接口：I2C

产品尺寸：65mm x 30mm

固定孔通径：3.0mm

硬件说明



板子可以从树莓派上取电不需要额外供电。

也可以通过右边绿色端子 VIN 接电池供电，输入电压范围 6V~12V。经过板载的 5V 稳压芯片输出 5V 电源给舵机和树莓派供电，最大输出电流 3A。

A0~A4 可以设置 PCA9685 芯片的 I2C 设备地址，可以同时接多个 Servo Driver HAT

最上面的排针是舵机接口，黑色排针是 GND(大部分舵机对应的是褐色线)，红色排针是 5V 电源，黄色排针是 PWM 信号线，有 0~15 个通道，可以同时接 16 个舵机。注意舵机线不要接反，否则舵机不会转动。

注意:

如果接大功率的舵机可能会出现供电不足的现象，需要接更多的电源

使用指南

开启 I2C 接口

执行如下命令进行树莓派配置：

```
sudo raspi-config
```

选择 Interfacing Options -> I2C ->yes 启动 i2C 内核驱动

```
Raspberry Pi Software Configuration Tool (raspi-config)

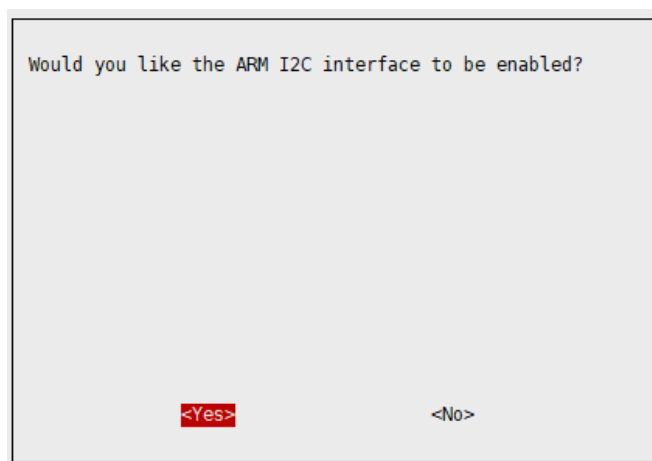
1 Change User Password Change password for the current user
2 Network Options       Configure network settings
3 Boot Options          Configure options for start-up
4 Localisation Options  Set up language and regional settings to match your location
5 Interfacing Options   Configure connections to peripherals
6 Overclock             Configure overclocking for your Pi
7 Advanced Options      Configure advanced settings
8 Update               Update this tool to the latest version
9 About raspi-config    Information about this configuration tool

<Select>                                <Finish>
```

```
Raspberry Pi Software Configuration Tool (raspi-config)

P1 Camera      Enable/Disable connection to the Raspberry Pi Camera
P2 SSH         Enable/Disable remote command line access to your Pi using SSH
P3 VNC         Enable/Disable graphical remote access to your Pi using RealVNC
P4 SPI         Enable/Disable automatic loading of SPI kernel module
P5 I2C         Enable/Disable automatic loading of I2C kernel module
P6 Serial      Enable/Disable shell and kernel messages on the serial connection
P7 1-Wire      Enable/Disable one-wire interface
P8 Remote GPIO Enable/Disable remote access to GPIO pins

<Select>                                <Back>
```



注意：

运行程序后有提示 I2C 错误，可以运行如下命令打开配置文件

```
sudo nano /etc/modules
```

如果没有这两行语句则添加上去，并保存退出。

```
i2c-dev
```

```
i2c-bcm2708
```

运行程序

我们提供 python2 和 python3 的示例程序，功能包括简单的 PCA9685 库测试程序，wifi 遥控程序以及蓝牙遥控程序。

其中 wifi 和蓝牙遥控发送端是通过手机 APP 发送指令,需要安装对应的 APP(只支持安卓)手机下载相应的 APP 并安装。

将下载程序复制到树莓派目录下并解压。

1. 库函数测试程序

运行如下命令运行程序

```
cd Servo_Driver_HAT/python
```

```
sudo python PCA9685.py
```

实验现象：将舵机接到 0 号通道，0 号通道的舵机从 0 度转到 180 度，然后又从 180 度转动到 0 度，不断重复循环。

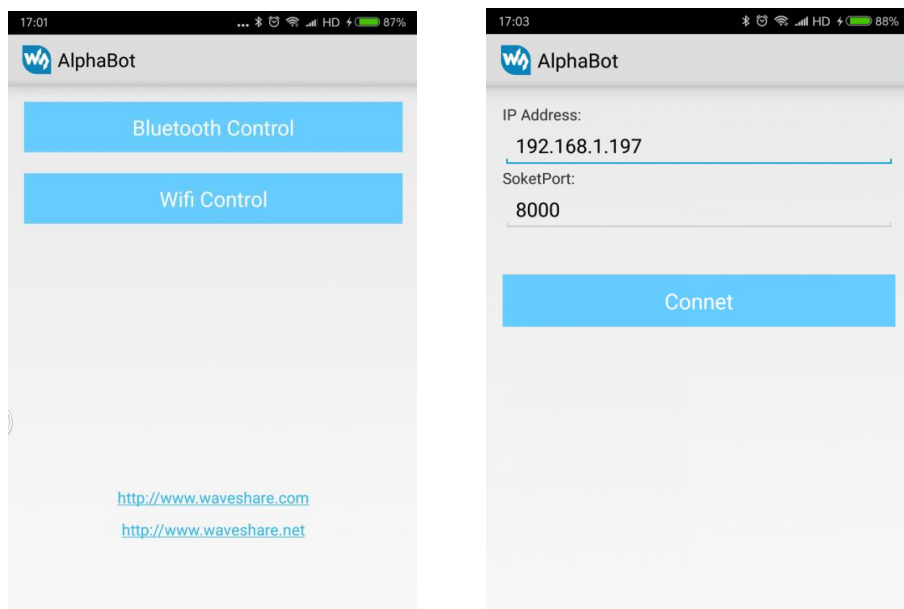
2. wifi 遥控程序

运行如下命令运行程序

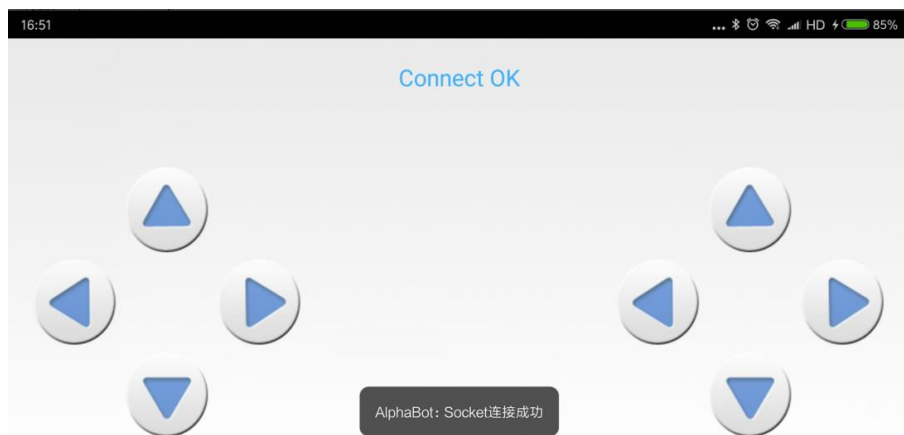
```
cd Servo_Driver_HAT/python/Wifi-Control  
sudo python main.py
```

操作与现象:发送端(手机或者电脑)和接收端(树莓派), 需要连接到同一个局域网。程序是通过 TCP 协议传输数据, 程序运行后会显示树莓派 IP,服务器端口号是 8000.

手机打开 APP,选择 Wifi 遥控, 输入对应的 IP 地址和端口号, 点击连接

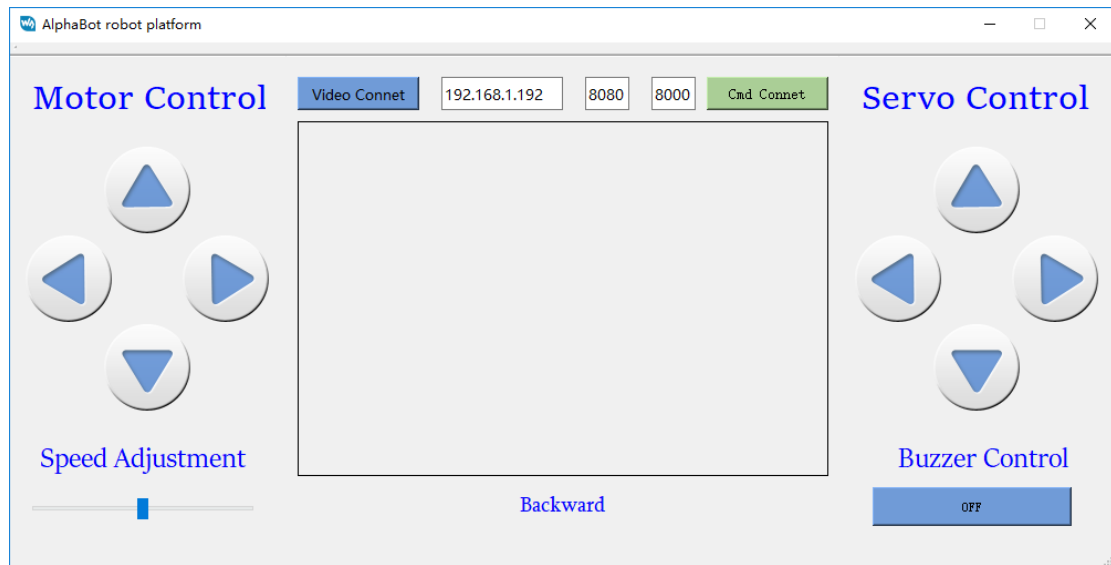


连接成功进入控制界面, 点击按键可以控制 0~4 号通道的舵机正反转。



```
pi@raspberrypi:~/Servo_Driver_HAT/python/Wifi-Control $ sudo python main.py  
start  
192.168.1.192  
  
server is running...  
( 'got connection from ', ('192.168.1.184', 59089))  
Forward  
Stop  
Backward  
Stop
```

也可在电脑上运行 Qt 软件控制，如下图



2. 蓝牙遥控程序

1) 升级安装蓝牙相关软件包，安装蓝牙依赖库

```
sudo apt-get update
sudo apt-get upgrade -y
sudo apt-get dist-upgrade -y
sudo apt-get install pi-bluetooth bluez bluez-firmware blueman
```

2) 添加 pi 用户到蓝牙组

```
sudo usermod -G bluetooth -a pi
```

3) 重启树莓派

```
sudo reboot
```

4) 启动/增加 SPP，开启蓝牙设备

```
sudo vi /etc/systemd/system/dbus-org.bluez.service
```

修改文件中这两个语句。

```
1 [Unit]
2 Description=Bluetooth service
3 Documentation=man:bluetoothd(8)
4
5 [Service]
6 Type=dbus
7 BusName=org.bluez
8 ExecStart=/usr/lib/bluetooth/bluetoothd -C
9 ExecStartPost=/usr/bin/sdptool add SP
10 NotifyAccess=main
11 #WatchdogSec=10
12 #Restart=on-failure
13 CapabilityBoundingSet=CAP_NET_ADMIN CAP_NET_BIND_SERVICE
14 LimitNPROC=1
15
16 [Install]
17 WantedBy=bluetooth.target
18 Alias=dbus-org.bluez.service
```

重启树莓派后，输入 hciconfig 命令(类似 ifconfig 命令)查看蓝牙服务

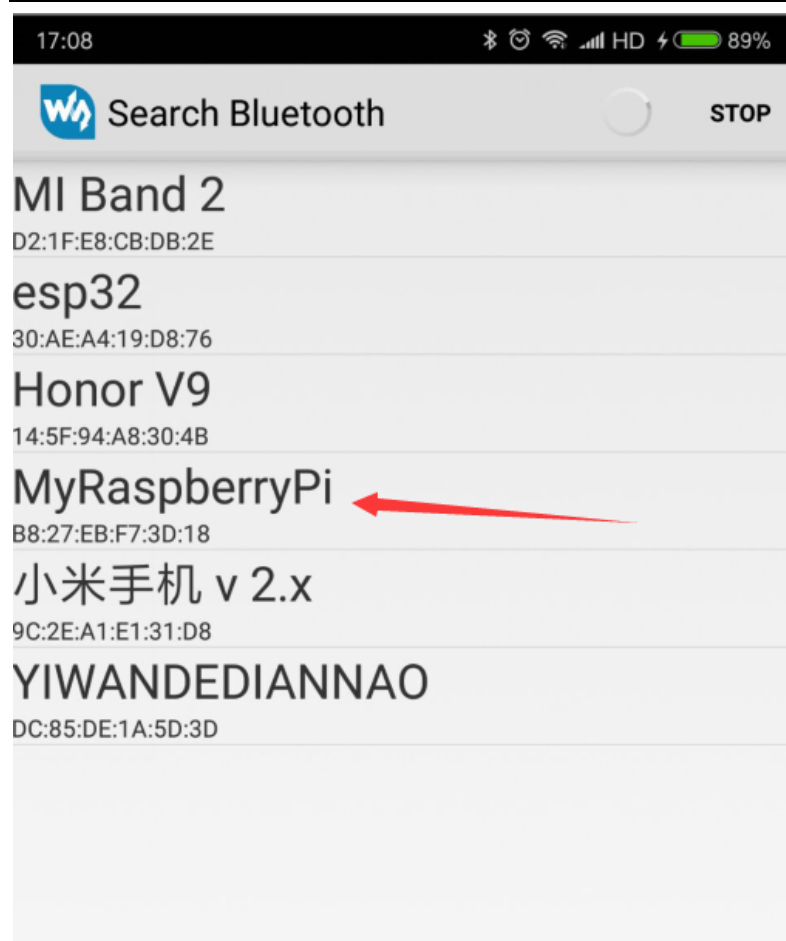
```
pi@raspberrypi:~ $ sudo hciconfig
hci0: Type: BR/EDR Bus: UART
      BD Address: B8:27:EB:2D:00:87 ACL MTU: 1021:8 SCO MTU: 64:1
      UP RUNNING
      RX bytes:717 acl:0 sco:0 events:42 errors:0
      TX bytes:1532 acl:0 sco:0 commands:42 errors:0
```

如果有看到 hci0 设备则蓝牙已经开启工作。如果没有则没有识别到蓝牙设备

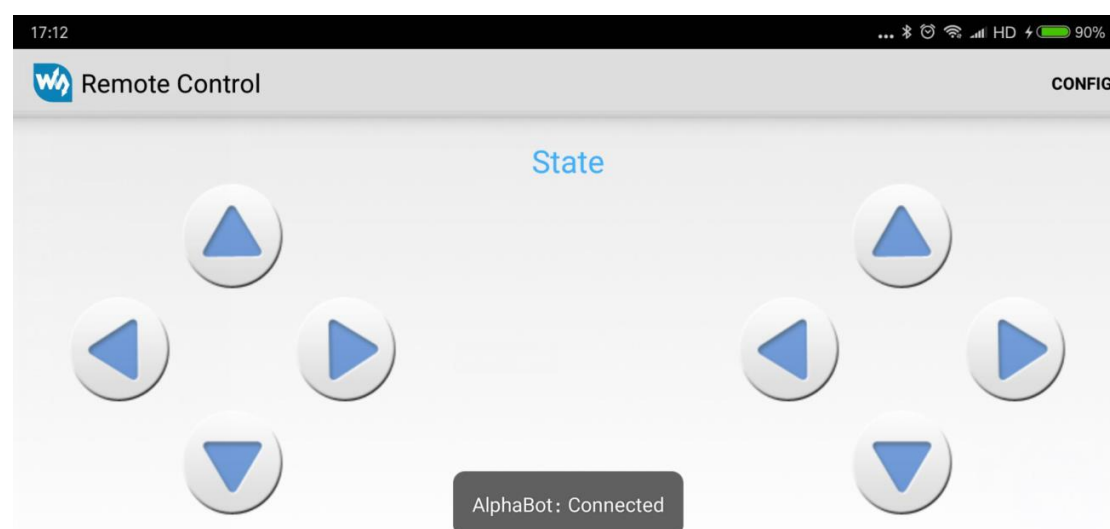
运行如下命令启动程序

```
cd Servo_Driver_HAT/python/Bluetooth-Control
sudo ./Bluetooth.sh
```

操作与现象:程序运行后会提示等待蓝牙连接，手机 App 选择蓝牙遥控，点击扫描，发现 raspberrypi 设备并连接。



连接程序后进入控制界面,点击按键可以遥控 0~4 通道的舵机正反转了。



注意:如果接收到的命令不对,可能需要在 APP 中 config 一下对应按键按下和松开时发送的命令。

```

pi@raspberrypi:~/Servo_Driver_HAT/python/Bluetooth-Control $ sudo ./bluetooth.sh
[NEW] Controller B8:27:EB:F7:3D:18 raspberrypi [default]
[NEW] Device 94:87:E0:DA:12:1D 小米手机
[bluetooth]# discoverable on
[DEL] Controller B8:27:EB:F7:3D:18 raspberrypi [default]
[bluetooth]#
(bluetoothctl:32748): GLib-CRITICAL **: Source ID 23 was not found when attempting to remove it
Waiting for connection on channel 1
Connection from 94:87:E0:DA:12:1D to /dev/rfcomm0
Press CTRL-C for hangup
serial test start ...
Forward
Stop
Backward
Stop

```

注意:树莓派默认是只有 180s 的发现时间, 如果想一直可以发现和配对, 如果通过修改下面的文件配置。

```
sudo vi /etc/bluetooth/main.conf
```

找到下面两个语句并去掉注释。

```

# Defaults to 'BlueZ X.YZ'
#Name = BlueZ

# Default device class. Only the major and minor device class bits are
# considered. Defaults to '0x000000'.
#Class = 0x000100

# How long to stay in discoverable mode before going back to non-discoverable
# The value is in seconds. Default is 180, i.e. 3 minutes.
# 0 = disable timer, i.e. stay discoverable forever
DiscoverableTimeout = 0

# How long to stay in pairable mode before going back to non-discoverable
# The value is in seconds. Default is 0.
# 0 = disable timer, i.e. stay pairable forever
PairableTimeout = 0

# Automatic connection for bonded devices driven by platform/user events.
# If a platform plugin uses this mechanism, automatic connections will be
# enabled during the interval defined below. Initially, this feature
# intends to be used to establish connections to ATT channels. Default is 60.
#AutoConnectTimeout = 60

```

程序兼容 python3 ,运行的时候将 python 替换成 python3 即可。

更多关于树莓派 wifi 和蓝牙遥控可以参考 AlphaBot2:

<http://www.waveshare.net/wiki/AlphaBot2>

<https://www.waveshare.com/wiki/AlphaBot2>

PCA9685 库使用说明

PCA9685.py 驱动库包含以下函数，您可以在 Python 中编写自己的应用程序时使用这些函数来驱动底层硬件：

初始化对象

您可以为每个 HAT 创建一个新对象

```
pwm = PWM(0x40)
```

其中 pwm（小写）是创建的对象，PWM(0x40)是创建调用。默认情况下，模块的 I2C 地址是 0x40，如果通过焊接电阻改变 I2C 地址，您可以创建使用其他地址的对象。

设置 PWM 频率

```
setPWMFreq(self,freq)
```

用于调整 PWM 频率，该频率确定每个脉冲从开始到结束的持续时间长度，即周期。

参数：

- **freq**：表示以 Hz 为单位的频率的数字，介于 40 和 1000 之间

如果 PWM 用于控制舵机，频率必须设置为 50HZ,如下示例是 PWM 频率设置为 50HZ

示例:

```
pwm.setPWMFreq(50)
```

设置 PWM 脉宽

```
setPWM(self,channel,on,off)
```

此函数设置特定通道上 PWM 脉冲的高电平的开始(on)和结束(off)。可以设置高电平在什么时候开启和什么时候结束，on,off 为在 0~4095 之间的值。通道表示 0~15 个 PWM 中的哪一个通道作为 PWM 输出。

参数：

- **channel**: pwm 更新的通道(0~15)
- **on**: 当信号从低电平变为高电平时的位置(在 0~4095 之间)
- **off**: 当信号从高电平变为低电平时的位置(在 0~4095 之间)

示例：

以下示例表示通道 15 从 0 开始输出高电平，高电平在 1024 结束。即(0~1024)高电平占 25%；(1024~4095)为低电平,占 75%。即 PWM 占空比为 25%。周期为(1/freq)。

```
pwm.setPWM(15 ,0 ,1024)
```

设置舵机脉宽

```
setServoPulse (self,channel, pulse)
```

此函数设置特定通道上输出可以控制舵机角度的脉宽。

注意：使用此函数前，必须将 PWM 频率设置为 50HZ,否则舵机转动不正常。

参数：

- **channel**: pwm 更新的通道(0~15)
- **pulse**: 表示输出的脉宽，单位是 us (范围在 500~2500 之间)

示例：

以下示例表示通道 0 输出脉宽为 1500us 的 PWM 信号，即舵机转到到 90 度角。

```
pwm.setServoPulse (0,1500)
```

舵机转动角度和脉宽有以下对应关系

500	-----	0 度
1000	-----	45 度
1500	-----	90 度
2000	-----	135 度
2500	-----	180 度