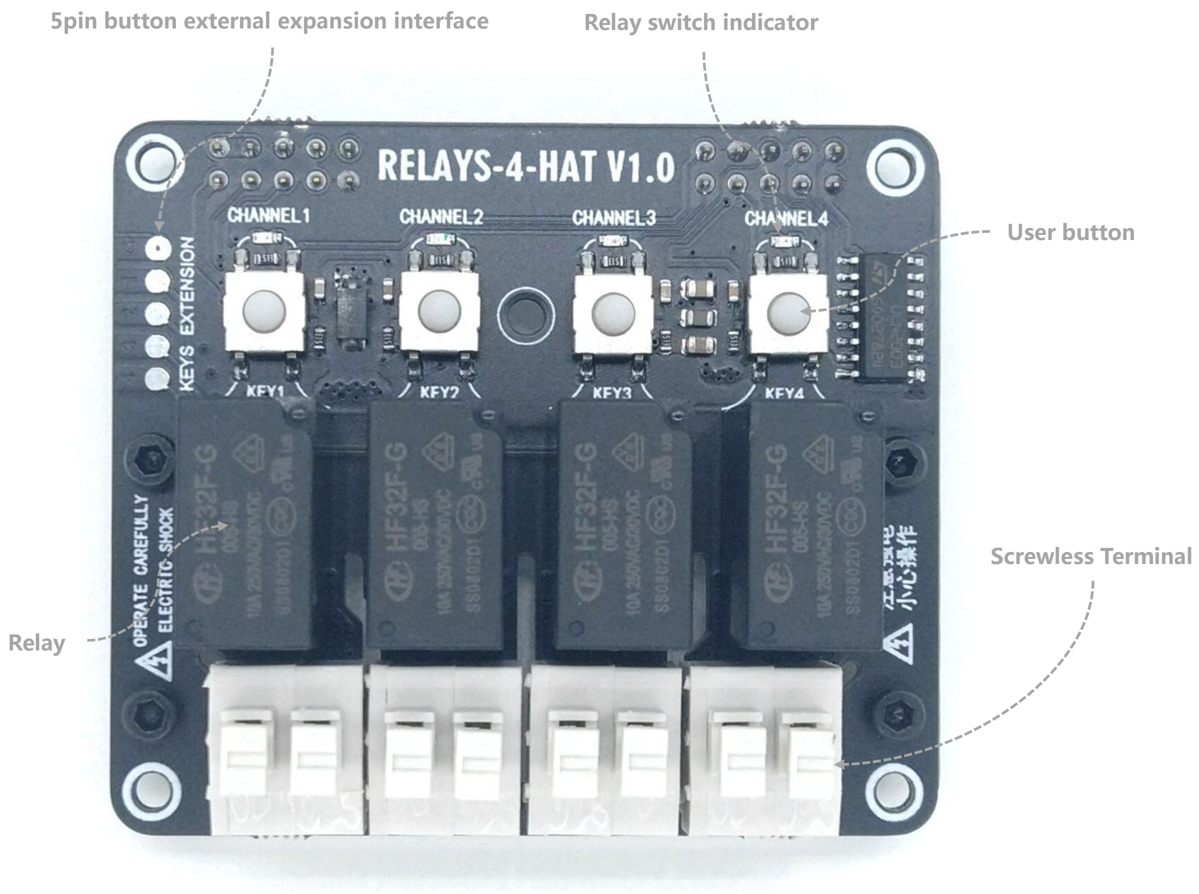
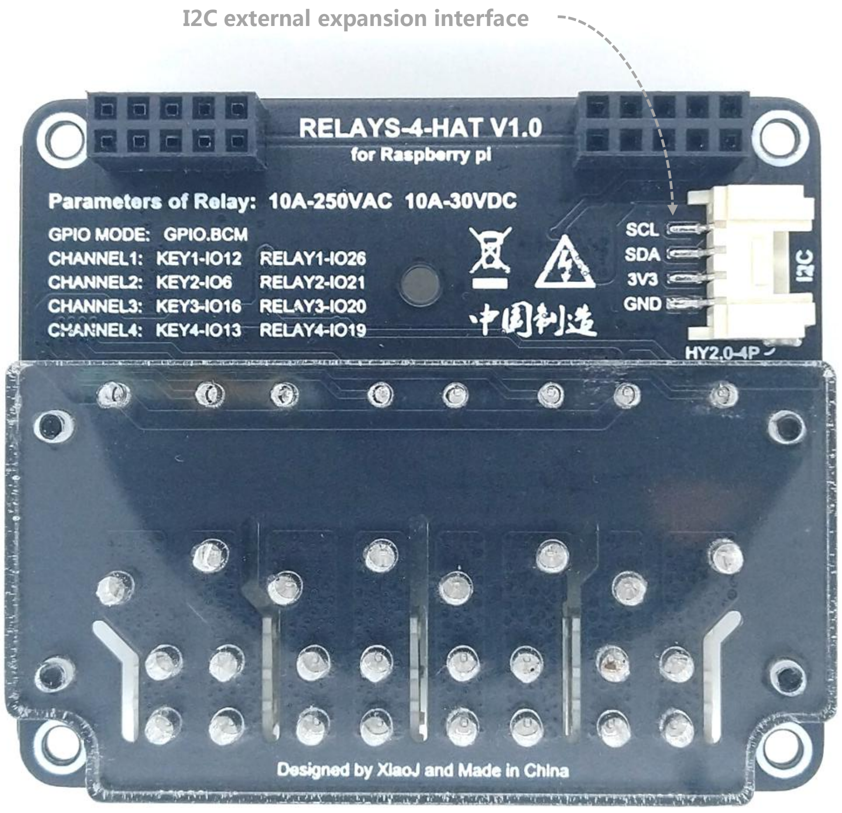
Instructions for RELAYS-4-HAT

4 Channel Switch Relay Expansion Board

1. Introduction

RELAYS-4-HAT is a relay switch control expansion board specially designed for Raspberry Pi A/B motherboards. The expansion board provides 4 middle-power switch relays, 4 user buttons, 5pin button external expansion interface, 1 I2C external expansion interface (compatible with GROVE interface), and use Screwless Terminal Block which is convenient and easy in operation. Among them, 4 switch relays can be used for switch control of various small and medium power electrical equipment. When the relay is closed, the corresponding red light on the board will light up, indicating that the relay has been turned on. The 4 user buttons can be set as the relay switch control buttons by writing the corresponding program to realize the manual control function. The 5pin button external expansion interface can extend 4 user buttons through the connection. I2C external expansion interface adopts 4pin HY2.0 terminal block, which can be connected with various I2C interface sensors.




I2C external expansion interface
RELAYS-4-HAT V1.0
for Raspberry pi
Parameters of Relay: 10A-250VAC 10A-30VDC
GPIO MODE: GPIO.BCM
CHANNEL1: KEY1-IO12 RELAY1-IO26
CHANNEL2: KEY2-IO6 RELAY2-IO21
CHANNEL3: KEY3-IO16 RELAY3-IO20
CHANNEL4: KEY4-IO13 RELAY4-IO19
中国制造
SCL
SDA
3V3
GND
I2C
HY2.0-4P
Designed by XiaoJ and Made in China

Electrical parameters:

Total power: 1.5W (4 relays are fully open, 300mA @ 5V)

Relay load power: 10A-250VAC 10A-30VDC

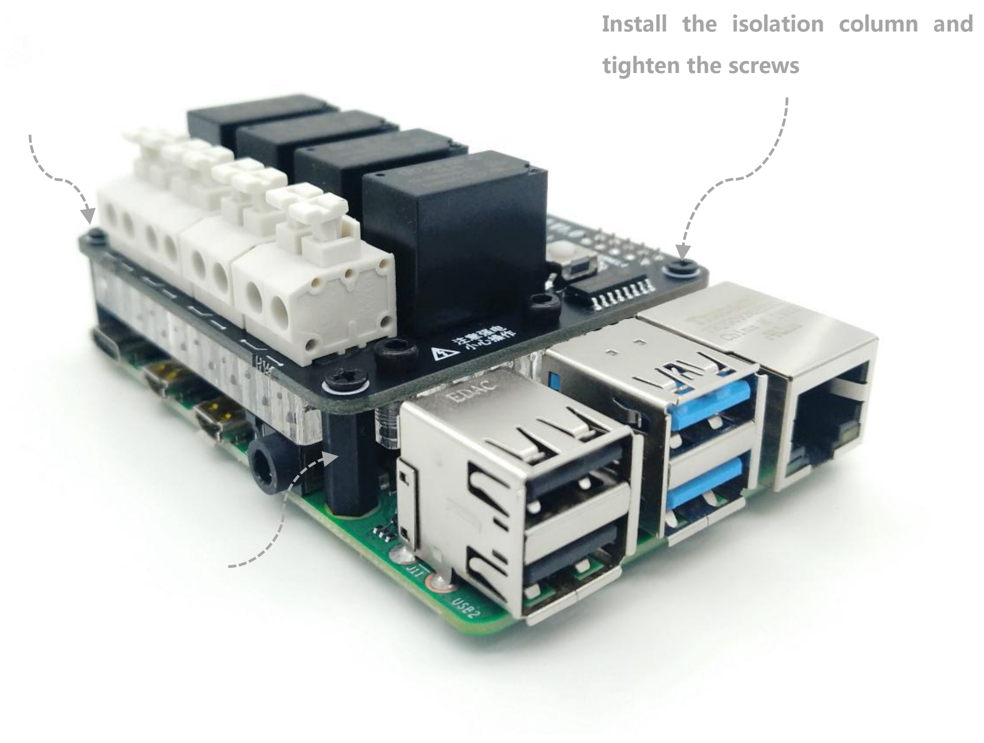
Wire diameter: 22 ~ 14 AWG

GPIO definition :

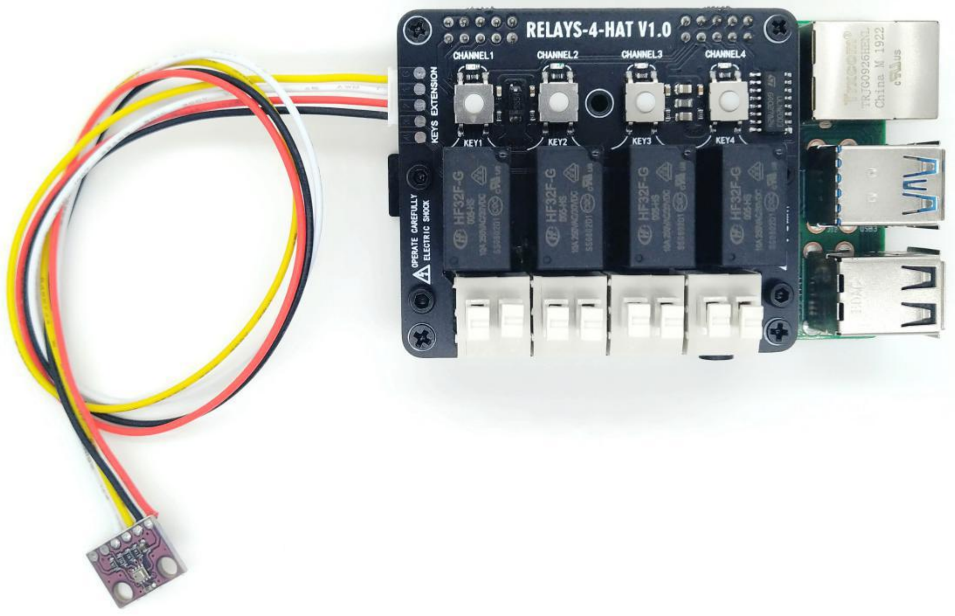
Switch channel 1	Relay 1--IO26	Button1--IO12
Switch channel 2	Relay 2--IO21	Button 2--IO6
Switch channel 3	Relay 3--IO20	Button 3--IO16
Switch channel 4	Relay 4--IO19	Button 4--IO13
Note: GPIO mode is BCM. The relay is turned on at a high level. A pull-up resistor is connected to the key, and the low level is pressed.		

2.Install

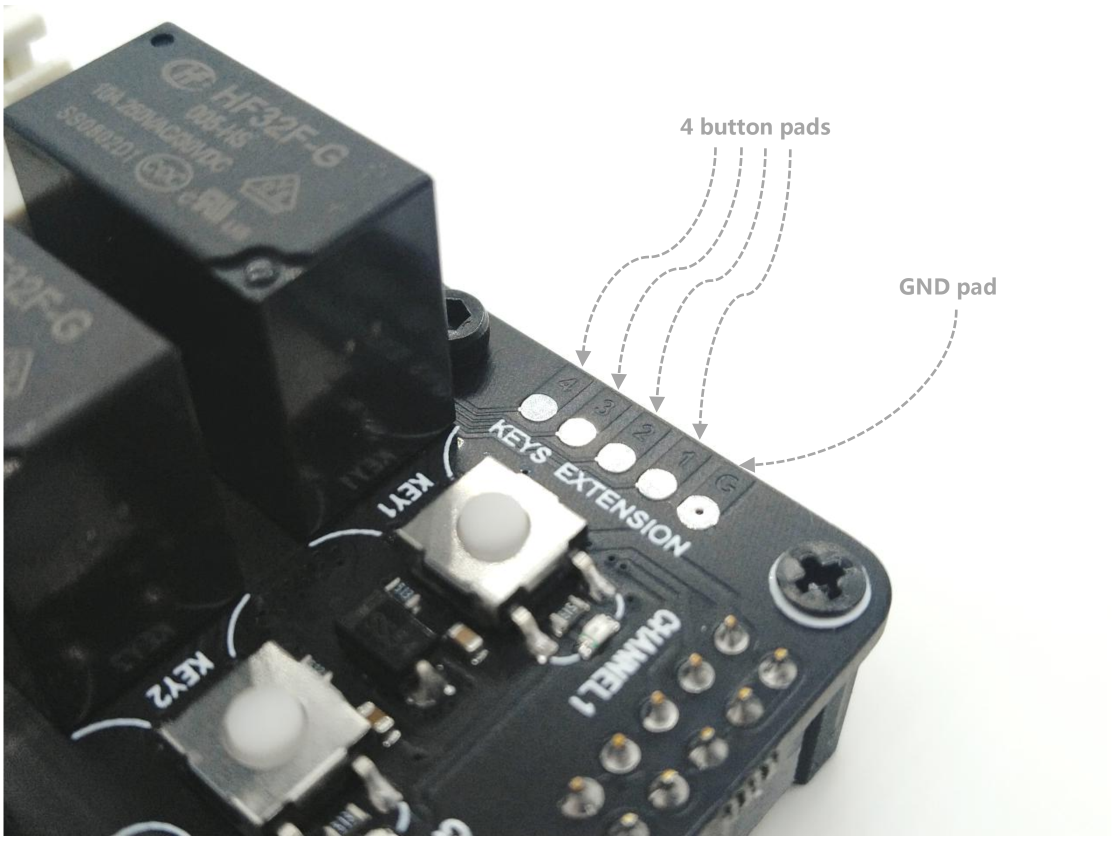
Just assemble the matching screws and spacers as shown in the figure



The I2C external expansion interface can be connected to an I2C interface sensor like BMP280 as shown in the figure.



5pin button external expansion interface, pads 1~4 correspond to buttons 1~4, pad G corresponds to GND, when 1 and G are short-circuited, it is equivalent to pressing button 1, and the IO port detects a low level. The other keys can be deduced by analogy.



appendix :

Script code of RELAYS-4-HAT.py

```
#!/usr/bin/python
# -*- coding:utf-8 -*-
import RPi.GPIO as GPIO
import time


Relay_Ch1 = 26
Relay_Ch2 = 21
Relay_Ch3 = 20
Relay_Ch4 = 19

Relay_Key1 = 12
Relay_Key2 = 6
Relay_Key3 = 16
Relay_Key4 = 13


Key1_state=0
Key2_state=0
Key3_state=0
Key4_state=0


def Key1_Interrupt(Relay_Key1):
    global Key1_state
    Key1_state=not Key1_state
    #print(Key1_state)
    time.sleep(0.1)
    if (Key1_state == 0):
        GPIO.output(Relay_Ch1,GPIO.LOW)
        print("Channel 1 : OFF")
    elif(Key1_state == 1):
        GPIO.output(Relay_Ch1,GPIO.HIGH)
        print("Channel 1 : ON")


def Key2_Interrupt(Relay_Key2):
    global Key2_state
    Key2_state=not Key2_state
    #print(Key2_state)
    time.sleep(0.1)
    if (Key2_state == 0):
        GPIO.output(Relay_Ch2,GPIO.LOW)
        print("Channel 2 : OFF")
    elif(Key2_state == 1):
        GPIO.output(Relay_Ch2,GPIO.HIGH)
        print("Channel 2 : ON")


def Key3_Interrupt(Relay_Key3):
    global Key3_state
    Key3_state=not Key3_state
    #print(Key3_state)
    time.sleep(0.1)
    if (Key3_state == 0):
        GPIO.output(Relay_Ch3,GPIO.LOW)
        print("Channel 3 : OFF")
    elif(Key3_state == 1):
        GPIO.output(Relay_Ch3,GPIO.HIGH)
        print("Channel 3 : ON")


def Key4_Interrupt(Relay_Key4):
    global Key4_state
    Key4_state=not Key4_state
    #print(Key4_state)
    time.sleep(0.1)
    if (Key4_state == 0):
        GPIO.output(Relay_Ch4,GPIO.LOW)
        print("Channel 4 : OFF")
    elif(Key4_state == 1):
        GPIO.output(Relay_Ch4,GPIO.HIGH)
        print("Channel 4 : ON")




GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

GPIO.setup(Relay_Ch1,GPIO.OUT)
GPIO.setup(Relay_Ch2,GPIO.OUT)
GPIO.setup(Relay_Ch3,GPIO.OUT)
GPIO.setup(Relay_Ch4,GPIO.OUT)
GPIO.setup(Relay_Key1,GPIO.IN)
GPIO.setup(Relay_Key2,GPIO.IN)
GPIO.setup(Relay_Key3,GPIO.IN)
GPIO.setup(Relay_Key4,GPIO.IN)


GPIO.add_event_detect(Relay_Key1,GPIO.FALLING,Key1_Interrupt,200)
GPIO.add_event_detect(Relay_Key2,GPIO.FALLING,Key2_Interrupt,200)
GPIO.add_event_detect(Relay_Key3,GPIO.FALLING,Key3_Interrupt,200)
GPIO.add_event_detect(Relay_Key4,GPIO.FALLING,Key4_Interrupt,200)




GPIO.output(Relay_Ch1,GPIO.HIGH)
time.sleep(0.4)
GPIO.output(Relay_Ch2,GPIO.HIGH)
time.sleep(0.4)
GPIO.output(Relay_Ch3,GPIO.HIGH)
time.sleep(0.4)
GPIO.output(Relay_Ch4,GPIO.HIGH)
time.sleep(0.4)
GPIO.output(Relay_Ch4,GPIO.LOW)
time.sleep(0.4)
GPIO.output(Relay_Ch3,GPIO.LOW)
time.sleep(0.4)
GPIO.output(Relay_Ch2,GPIO.LOW)
time.sleep(0.4)
GPIO.output(Relay_Ch1,GPIO.LOW)
time.sleep(0.4)




while True:

    print("Channel1: %d     Channel2: %d     Channel3: %d
Channel4: %d" %(Key1_state,Key2_state,Key3_state,Key4_state))
    time.sleep(1.5)

GPIO.cleanup()
```

References：

https://github.com/linshuqin329/RELAYS_4_HAT

If you have other questions, please contact me: 416386001@qq.com