

# RPi Zero Relay

---

Industrial 6-Channel Relay Module For Raspberry Pi Zero, RS485/CAN Bus, Power Supply Isolation, Photocoupler Isolation

## Features

---

- RS485 half-duplex communication: using SP3485, UART control, auto RX/TX switch
- CAN half-duplex communication: using MCP2515 + SN65HVD230 solution, SPI control
- Onboard unibody power supply isolation, provides stable isolated voltage, needs no extra power supply for the isolated terminal
- Onboard photocoupler isolation, prevent interference from external high-voltage circuit connected to the relay
- Onboard TVS (Transient Voltage Suppressor), effectively suppress surge voltage and transient spike voltage in the circuit, lightningproof & anti-electrostatic
- Onboard resettable fuses and protection diodes, ensuring current/voltage stable output, preventing over current/voltage, better shock-resistance performance
- High quality relay, contact rating:  $\leq 10A$  250V AC or  $\leq 10A$  30V DC
- ABS protection enclosure with rail-mount support, easy to install, safe to use

## Specifications

---

- Operating voltage: 7V~36V (industrial input voltage compatible)
- Relay channel: 6ch
- Communication Protocol: RS485,CAN
- Contact form: 1NO 1NC

## Pinouts

---

- CAN bus

PIN	Raspberry Pi (BCM)	Description
GND	GND	Ground
SCK	SCK	SPI clock input
MOSI	MOSI	SPI data input

MISO	MISP	SPI Data output
CS	CE0	Data/Command Selection
INT	25	Interrupt output

- RS485 bus

<b>PIN</b>	<b>Raspberry Pi (BCM)</b>	<b>Description</b>
GND	GND	Ground
RXD	RXD	UART receive
TXD	TXD	UART transmit

- Relay interfaces

<b>PIN</b>	<b>Raspberry Pi (BCM)</b>	<b>Description</b>	GND!	GND	Ground
IN_CH1	GPIO5	Relay Channel 1			
IN_CH2	GPIO6	Relay Channel 2			
IN_CH3	GPIO13	Relay Channel 3			
IN_CH4	GPIO16	Relay Channel 4			
IN_CH5	GPIO19	Relay Channel 5			
IN_CH6	GPIO20	Relay Channel 6			

# Install libraries

---

- BCM2835 libraries

```
wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.68.tar.gz
tar zxvf bcm2835-1.68.tar.gz
cd bcm2835-1.68/
sudo ./configure && sudo make && sudo make check && sudo make install
```

- Install wiringPi

```
sudo apt-get install wiringpi
wget https://project-downloads.drogon.net/wiringpi-latest.deb
sudo dpkg -i wiringpi-latest.deb
gpio -v
```

- Python

```
sudo apt-get update
sudo apt-get install python-serial
sudo pip install python-can
```

## Download Examples

---

Open a terminal and run the following commands to download the demo codes

```
sudo apt-get install p7zip-full
wget https://www.waveshare.com/w/upload/2/2f/RPi_Zero_Relay_Code.7z
7z x RPi_Zero_Relay_Code.7z -r -o./RPi_Zero_Relay_Code
sudo chmod 777 -R RPi_Zero_Relay_Code/
```

## CAN Bus

---

### Configuration

- Modify the config.txt file

```
sudo nano /boot/config.txt
```

Add the following lines to the config.txt file and save

```
dtoverlay=spi=on
```

```
dtoverlay=mcp2515-can0,oscillator=16000000,interrupt=25,spimaxfrequency=3000000
```

- Reboot

```
sudo reboot
```

- Check if the driver is initialized normally

```
dmesg | grep -i '\(can\|spi\|)'
```

The result should like the picture:

```
pi@raspberrypi:~$ dmesg | grep -i '\(can\|spi\|)'
[ 16.369968] systemd[1]: Cannot add dependency job for unit regenerate_ssh_host_keys.service, ignoring: Unit regenerate_ssh_host_keys.service failed to load: No such file or directory.
[ 16.568756] systemd[1]: Cannot add dependency job for unit display-manager.service, ignoring: Unit display-manager.service failed to load: No such file or directory.
[ 20.892310] CAN device driver interface
[ 20.915484] mcp251x spi0.0 can0: MCP2515 successfully initialized.
```

If the MCP2515 driver is not initialized normally you need to reboot and check if the devices are connected properly (H to H and L to L).

```
pi@raspberrypi:~$ dmesg | grep -i '\(can\|spi\|)'
[ 16.300731] systemd[1]: Cannot add dependency job for unit regenerate_ssh_host_keys.service, ignoring: Unit regenerate_ssh_host_keys.service failed to load: No such file or directory.
[ 16.499602] systemd[1]: Cannot add dependency job for unit display-manager.service, ignoring: Unit display-manager.service failed to load: No such file or directory.
[ 20.661718] CAN device driver interface
[ 20.680261] mcp251x spi0.0: Cannot initialize MCP2515. Wrong wiring?
[ 20.680293] mcp251x spi0.0: Probe failed. err=19
```

## C examples

- Run the receive example

```
cd ~/RPi_Zero_Relay_Code/CAN/wiringPi/receive/
make clean
make
sudo ./can_receive
```

After running the receive code, the terminal is waiting for data

```
pi@raspberrypi:~/RPi_Zero_Relay_Code/CAN/wiringPi/receive$ sudo ./can_receive
set wiringPi lib success !!!
RTNETLINK answers: Device or resource busy
this is a can receive demo
```

- Run the Send examples:

```
cd RPi_Zero_Relay_Code/CAN/wiringPi/send/
make clean
make
sudo ./can_send
```

```
pi@raspberrypi:~/RPi_Zero_Relay_Code/CAN/wiringPi/send $ sudo ./can_send
RTNETLINK answers: Device or resource busy
this is a can send demo
can_id = 0x123
You can always receive data, press Ctrl + C to exit
You can enter up to 7 characters
Send "abcdef" to open relays 1, 2, 3, 4, 5 and 6 respectively
Send "123456" to close relays 1, 2, 3, 4, 5 and 6 respectively

Please enter the value you want to enter : 
```

Type data to send

```
pi@raspberrypi:~/RPi_Zero_Relay_Code/CAN/wiringPi/send $ sudo ./can_send
RTNETLINK answers: Device or resource busy
this is a can send demo
can_id = 0x123
You can always receive data, press Ctrl + C to exit
You can enter up to 7 characters
Send "abcdef" to open relays 1, 2, 3, 4, 5 and 6 respectively
Send "123456" to close relays 1, 2, 3, 4, 5 and 6 respectively

Please enter the value you want to enter : abcdef
The data received back is : abcdef
Turn on Relay 1
Turn on Relay 2
Turn on Relay 3
Turn on Relay 4
Turn on Relay 5
Turn on Relay 6

Please enter the value you want to enter : 123456
The data received back is : 123456
Turn off Relay 1
Turn off Relay 2
Turn off Relay 3
Turn off Relay 4
Turn off Relay 5
Turn off Relay 6

Please enter the value you want to enter : 
```

In the receive terminal, you will get the packet related to id.

```
pi@raspberrypi:~/RPi_Zero_Relay_Code/CAN/wiringPi/receive $ sudo ./can_receive
set wiringPi lib success !!!
RTNETLINK answers: Device or resource busy
this is a can receive demo

The data received is : abcdef
Turn on Relay 1
Turn on Relay 2
Turn on Relay 3
Turn on Relay 4
Turn on Relay 5
Turn on Relay 6

The data received is : 123456
Turn off Relay 1
Turn off Relay 2
Turn off Relay 3
Turn off Relay 4
Turn off Relay 5
Turn off Relay 6
```

## Python examples

Open a terminal and run the commands

```
cd RPi_Zero_Relay_Code/CAN/python/
#For receiving :
sudo python can_reveive.py
#For sending :
sudo python can_send.py
```

## Connect to other CAN device

If you need to connect the RPi Zero Relay to other CAN devices, please note that:

- Please check the connection, it should be H to H and L to L
- Make sure that the baud rate of both sides are the same, the default baud rate of examples is 100K
- Check if the CAN ID of both sides are the same
- If data is always lost while transmitting, please decrease the baud rate and test it again.

```
import RPi.GPIO as GPIO
import serial
import sys
import os
from can import Message

os.system('sudo ip link set can0 type can bitrate 100000')
os.system('sudo ifconfig can0 up')

can0 = can.interface.Bus(channel = 'can0', bustype = 'socketcan_ctypes')# socketcan_native

print("Wait for the required configuration to complete.")
msg1 = can.Message(arbitration_id=0x123)

if msg1 is None:
    print("Timeout occurred, no message.")

print("You can always send data, press Ctrl + C to exit")
print("Send '123456' to open relays 1, 2, 3, 4, 5 and 6 respectively")
print("Send '123456' to close relays 1, 2, 3, 4, 5 and 6 respectively")
while True:
    msg = can.Message(arbitration_id=0x123, data=[0, 1, 2, 3, 4, 5], extended_id=False)
    can0.send(msg)

    msg1 = can0.recv(10.0)
    if msg1 is None:
        print("Timeout occurred, no message.")
        print("The data received back is: ",msg1.data)

    msg1 = can0.recv(10.0)
    if msg1 is None:
        print("Timeout occurred, no message.")

    for i in range(6):
        if(msg1.data[i] & 0x01 << i):
            print("Relay ",i+1," off")
        if(msg1.data[i] & 0x01 << i):
            print("Relay ",i+1," on")

os.system('sudo ifconfig can0 down')

os.system('sudo ip link set can0 type can bitrate 100000')
os.system('sudo ifconfig can0 up')

can0 = can.interface.Bus(channel = 'can0', bustype = 'socketcan_ctypes')# socketcan_native

# msg = can.Message(arbitration_id=0x123, data=[0, 1, 2, 3, 4, 5, 6, 7], extended_id=False)

i = 0
b = '\n'
fun = 0x0f
fud = 0x00
buff = [0,0]

msg1 = can.Message(arbitration_id=0x123, data=[0], extended_id=False)
can0.send(msg1)

print("You can always receive data, press Ctrl + C to exit.\n")
while True:
    msg = can0.recv(10.0)
    if msg is None:
```

## RS485 Interface Configuration

Open a terminal and run the command

```
sudo raspi-config
```

Choose Interfacing Options -> Serial -> No -> Yes

Disable the login shell function and enable the hardware serial port.

## Reboot

```
sudo reboot
```

- Connect the device, A to A and B to B

## C examples

- Run the receive example

```
cd RPi_Zero_Relay_Code/485/WiringPi/send
make clean
make
sudo ./485_receive
```

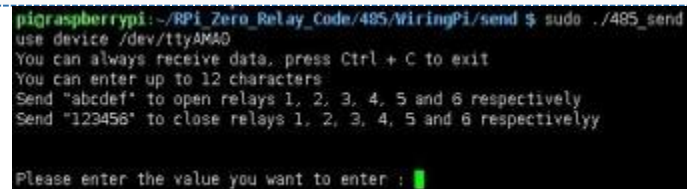
After running the receive code, the terminal is waiting for data



```
pi@raspberrypi:~/RPi_Zero_Relay_Code/485/WiringPi/receive $ sudo ./485_receive
set wiringPi lib success !!!
use device /dev/ttyAMA0
You can always receive data, press Ctrl + C to exit
█
```

## Run the Send example

```
cd RPi_Zero_Relay_Code/485/WiringPi/send
make clean
make
sudo ./485_send
```



```
pi@raspberrypi:~/RPi_Zero_Relay_Code/485/WiringPi/send $ sudo ./485_send
use device /dev/ttyAMA0
You can always receive data, press Ctrl + C to exit
You can enter up to 12 characters
Send 'abcdef' to open relays 1, 2, 3, 4, 5 and 6 respectively
Send '123456' to close relays 1, 2, 3, 4, 5 and 6 respectively

Please enter the value you want to enter : █
```

- Input data and send:

```
pi@raspberrypi:~/RPi_Zero_Relay_Code/485/MiringPi/send $ sudo ./485_send
use device /dev/ttyAMA0
You can always receive data, press Ctrl + C to exit
You can enter up to 12 characters
Send "abcdef" to open relays 1, 2, 3, 4, 5 and 6 respectively
Send "123456" to close relays 1, 2, 3, 4, 5 and 6 respectively

Please enter the value you want to enter : abcdef
The data received back is : abcdef
Turn on Relay 1
Turn on Relay 2
Turn on Relay 3
Turn on Relay 4
Turn on Relay 5
Turn on Relay 6

Please enter the value you want to enter : 123456
The data received back is : 123456
Turn off Relay 1
Turn off Relay 2
Turn off Relay 3
Turn off Relay 4
Turn off Relay 5
Turn off Relay 6

Please enter the value you want to enter : █
```

- The receive terminal receive and print:

```
pi@raspberrypi:~/RPi_Zero_Relay_Code/485/MiringPi/receive $ sudo ./485_receive
set wiringPi lib success: !!!
use device /dev/ttyAMA0
You can always receive data, press Ctrl + C to exit

The data received is : abcdef
Turn on Relay 1
Turn on Relay 2
Turn on Relay 3
Turn on Relay 4
Turn on Relay 5
Turn on Relay 6

The data received is : 123456
Turn off Relay 1
Turn off Relay 2
Turn off Relay 3
Turn off Relay 4
Turn off Relay 5
Turn off Relay 6
█
```

## Python examples

Open a terminal and run the commands:

```
cd RPi_Zero_Relay_Code/485/python/
#Receive example :
sudo python receive.py
#Send example :
sudo python send.py
```

## Troubleshooting

If the RS485 cannot work normally, please check:

Please check if you have disabled the login shell function

Please check the connection, it should be A to A and B to B

Please test with USB to RS485, make sure that all the settings are correct.



- [Schematic](#)
- [Demo codes](#)
- [SN65HVD230 Datasheet](#)
- [MCP2515 Datasheet](#)
- [SP348 Datasheet](#)