RS-485 Shield for Raspberry Pi

Release date : 01/09/2019
Version : 1.0

Bazaar: https://www.seeedstudio.com/RS-485-Shield-for-Raspberry-Pi.html
RS-485 is a cost-effective solution in serial communication networks. It can be used up to data rates at 10 Mbit/s or distances up to 1200m at lower speeds. This RS-485 Shield is a standard add-on board for Raspberry Pi. It is integrated with simple screw terminal as well as DB9 interface.

**Feature**

- One driver and one receiver per part
- EMI noise minimization
- Transmission rate up to 2.5Mbps
- No driver slew rate limitation
- Short-circuit current limited
- Fail-Safe Applications
## Specification

<table>
<thead>
<tr>
<th>Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Supply Voltage</td>
<td>3.3V</td>
</tr>
<tr>
<td>Interface</td>
<td></td>
</tr>
<tr>
<td>RS-485 DB9 Interface X1</td>
<td></td>
</tr>
<tr>
<td>RS-485 Screw Interface X1</td>
<td></td>
</tr>
<tr>
<td>2×13 Female Header To Raspberry X1</td>
<td></td>
</tr>
<tr>
<td>2×13 Expansion Female Header X1</td>
<td></td>
</tr>
<tr>
<td>Grove I2C Interface X1</td>
<td></td>
</tr>
<tr>
<td>Data Rate</td>
<td>2.5Mbps</td>
</tr>
<tr>
<td>Number of Receivers on</td>
<td>32</td>
</tr>
<tr>
<td>Storage Temperature Range</td>
<td>-65～160℃</td>
</tr>
<tr>
<td>Channel Number</td>
<td>8</td>
</tr>
<tr>
<td>Resolution</td>
<td>12-Bit</td>
</tr>
<tr>
<td>Power Consumption</td>
<td>Different power consumption depending on the transmission rate</td>
</tr>
<tr>
<td>Weight</td>
<td>42g</td>
</tr>
<tr>
<td>Length</td>
<td>62.0mm</td>
</tr>
<tr>
<td>Width</td>
<td>39.2mm</td>
</tr>
<tr>
<td>Height</td>
<td>21.8mm</td>
</tr>
</tbody>
</table>
RS-485 Shield for Raspberry Pi

Typical Applications

- Low-Power RS-485 Transceivers
- Level Translators
- Transceivers for EMI-Sensitive Applications Industrial-Control Local Area Networks
- Half-duplex applications

Hardware Overview

Pin Out

Overview
RS-485 Interface & RS-485 Screw Interface

The 485 interface uses differential signal transmission. Please make sure the port A connect to the port A of the 485 device, and port B connect to the port B of the 485 device.

485-A: The end A of RS485 data transmit wire, connect to the pin A of the MAX485 chip.
485-B: The end B of RS485 data transmit wire, connect to the pin B of the MAX485 chip.
GND: Connect to the Raspberry Pi GND.

You can click the picture to view the original file

As you can see, GPIO14 and GPIO15 are applied for data transmission, and we use GPIO18 as the enable signal.

For the definition of logic signals, please refer to the table below.
We use a 2X13 Female Header to plug this module into the Raspberry Pi, please make sure the pins are aligned.
Extension Female Headers

This RS-485 Shield occupies 26 Raspberry Pi pins, and only 5 GPIO pins are actually used. We take these 26 pins out in case you need those pins for other purposes.
## GPIO Occupied List

<table>
<thead>
<tr>
<th>GPIO Number</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPIO02</td>
<td>SDA for Grove I2C Port</td>
</tr>
<tr>
<td>GPIO03</td>
<td>SCL for Grove I2C Port</td>
</tr>
<tr>
<td>GPIO14</td>
<td>Connect to the pin DI of Max485 chip, for data transmission.</td>
</tr>
<tr>
<td>GPIO15</td>
<td>Connect to the pin RO of Max485 chip, for data transmission.</td>
</tr>
<tr>
<td>GPIO18</td>
<td>Connect to the pin RE &amp; DE of Max485 chip, work as the enable signal.</td>
</tr>
</tbody>
</table>

### Grove I2C Port

We have reserved the I2C interface so that you can easily use it with I2C devices. It should be noted that the VCC of this port is 5V, you need to confirm that whether the module is compatible with 5V voltage.

- **SCL**: I2C serial clock, connect to the GPIO03 of Raspberry Pi.
- **SDA**: I2C serial data, connect to the GPIO02 of Raspberry Pi.
- **VCC**: Connect to the Raspberry Pi 5V pin.
- **GND**: Connect to the Raspberry Pi GND pin.
Max485 Chip

We use the MAX485ESA IC for this shield, for more detail about this IC, please check the MAX485 Datasheet.

Platforms Supported

<table>
<thead>
<tr>
<th>Arduino</th>
<th>Raspberry Pi</th>
<th>BeagleBone</th>
<th>Wio</th>
<th>LinkIt ONE</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Arduino" /></td>
<td><img src="image2" alt="Raspberry Pi" /></td>
<td><img src="image3" alt="BeagleBone" /></td>
<td><img src="image4" alt="Wio" /></td>
<td><img src="image5" alt="LinkIt ONE" /></td>
</tr>
</tbody>
</table>
Getting Started

Hardware

Materials required

<table>
<thead>
<tr>
<th>Raspberry pi</th>
<th>RS-485 Shield for Raspberry Pi</th>
</tr>
</thead>
</table>

- Step 1. Plug the RS-485 Shield for Raspberry Pi into the Raspberry Pi.

- Step 2. Connect the 485-A to 485 wire A, connect the 485-B to 485 wire B.

!!!Note If the wire is reversed, communication will not be possible.

- Step 3. Power the Raspberry Pi with micro-usb cable.

Software

Communication test code

You can create a new python file and copy the following code into the new file, or you can download the source file in the resource download area. Then run it in your terminal.
Sending Code.

```python
#!/usr/bin/env python

import time
import serial
import os

send_str = "*******rs485888888--\r\n"

os.system("echo 18 > /sys/class/gpio/export")
os.system("echo out > /sys/class/gpio/gpio18/direction")
ser = serial.Serial(port='/dev/ttyAMA0', baudrate =10000000, bytesize=8, stopbits=1, timeout=1)

last_time = time.time()

now_time = time.time()
os.system("echo 1 > /sys/class/gpio/gpio18/value")
time.sleep(0.01)
n = 800
while n>0:
    ser.write(send_str)
    n=n-1
#    time.sleep(0.001)
os.system("echo 0 > /sys/class/gpio/gpio18/value")
```
#!/usr/bin/env python

import time
import serial
import os

send_str = "********abcdefghijklmnopqrstuvwxyz&"

os.system("echo 18 > /sys/class/gpio/export")
os.system("echo out > /sys/class/gpio/gpio18/direction")

ser = serial.Serial(port='/dev/ttyAMA0',baudrate=115200,bytesize=8,stopbits=1,timeout=1)

last_time = time.time()

while 1:
    now_time = time.time()
    if((now_time-last_time)>=1):
        last_time = now_time
        os.system("echo 1 > /sys/class/gpio/gpio18/value")
        time.sleep(0.01)
        ser.write(send_str)
        time.sleep(0.01)
        os.system("echo 0 > /sys/class/gpio/gpio18/value")
        count = ser.inWaiting()
        if(count != 0):
            x = ser.readline()
            if "********" in x:
                print "str length is: " + str(count)
                print x

You need two shield and two raspberry to test the code above, or you can use the serial tool in the PC to communicate with your raspberry pi.
Resources

- [Zip] RS-485 Shield for Raspberry Pi Eagle Files
- [Zip] Python Test Code
- [PDF] MAX485 Datasheet
- [PDF] PDF Format Wiki

Tech Support

Please do not hesitate to submit the issue into our forum or drop mail to techsupport@seeed.cc.