User Guide

2 Channel 5V Optical Isolated Relay Module

This is a LOW Level 5V 2-channel relay interface board, and each channel needs a 15-20mA driver current. It can be used to control various appliances and equipment with large current. It is equipped with high-current relays that work under AC250V 10A or DC30V 10A. It has a standard interface that can be controlled directly by microcontroller. This module is optically isolated from high voltage side for safety requirement and also prevent ground loop when interface to microcontroller.

Brief Data:

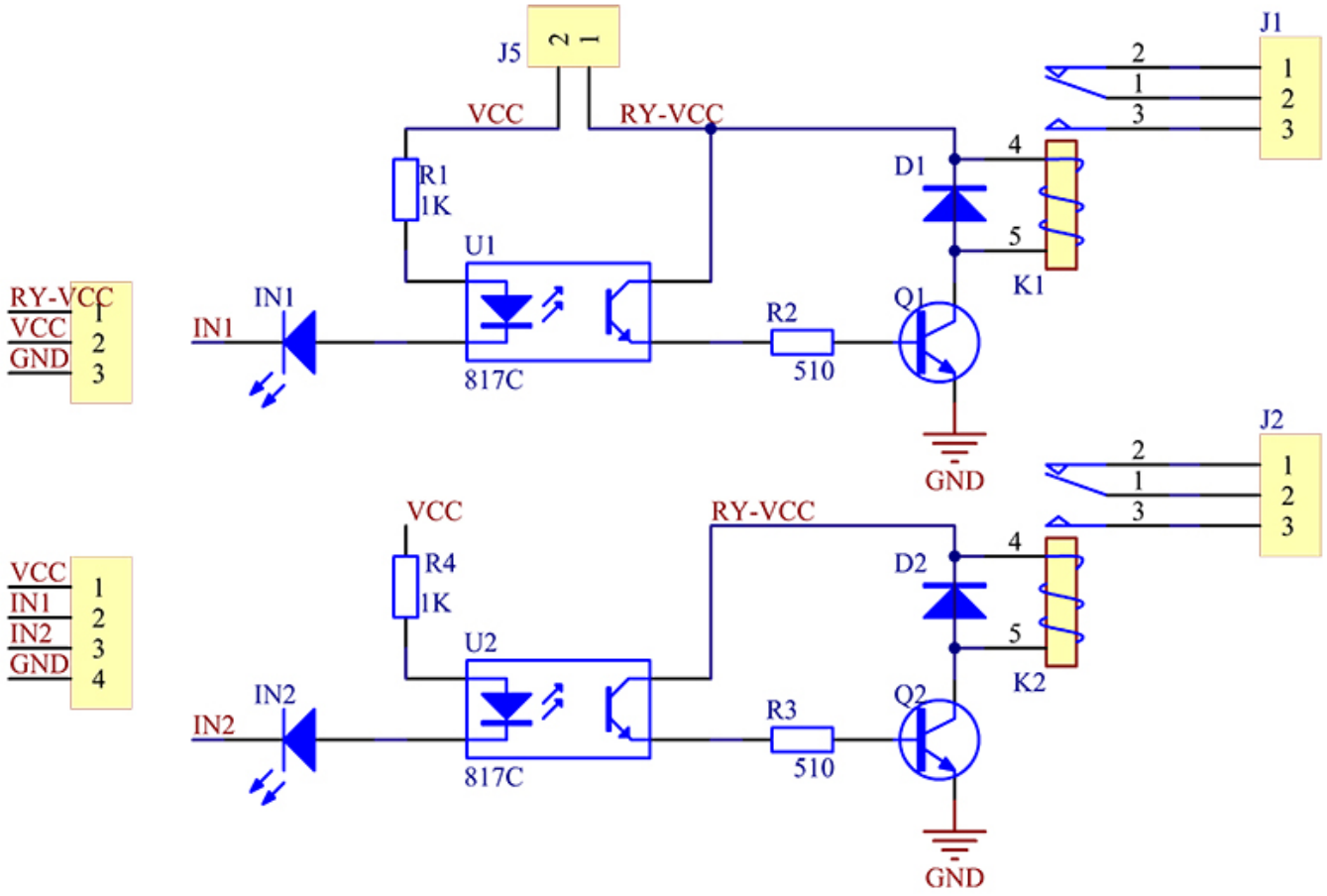
- Relay Maximum output: DC 30V/10A, AC 250V/10A.
- 2 Channel Relay Module with Opto-coupler. LOW Level Trigger expansion board, which is compatible with Arduino control board.
- Standard interface that can be controlled directly by microcontroller (8051, AVR, *PIC, DSP, ARM, ARM, MSP430, TTL logic).
- Relay of high quality low noise relays SPDT. A common terminal, a normally open, one normally closed terminal.
- Opto-Coupler isolation, for high voltage safety and prevent ground loop with microcontroller.

Schematic:

VCC and RY-VCC are also the power supply of the relay module. When you need to drive a large power load, you can take the jumper cap off and connect an extra power to RY-VCC to supply the relay; connect VCC to 5V of the MCU board to supply input signals.

NOTES: If you want complete optical isolation, connect "Vcc" to Arduino +5 volts but do NOT connect Arduino Ground. Remove the Vcc to JD-Vcc jumper. Connect a separate +5 supply to "JD-Vcc" and board Gnd. This will supply power to the transistor drivers and relay coils.

If relay isolation is enough for your application, connect Arduino +5 and Gnd, and leave Vcc to JD-Vcc jumper in place.

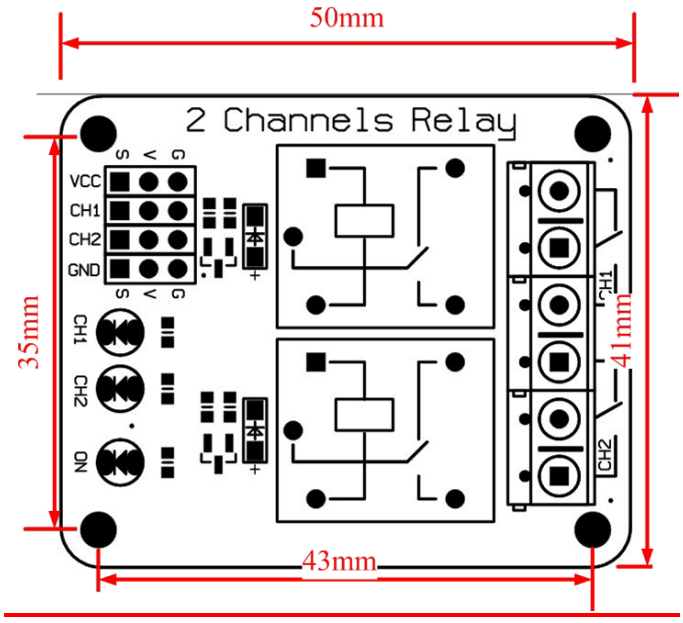


It is sometimes possible to use this relay boards with 3.3V signals, if the JD-VCC (Relay Power) is provided from a +5V supply and the VCC to JD-VCC jumper is removed. That 5V relay supply could be totally isolated from the 3.3V device, or have a common ground if opto-isolation is not needed. If used with isolated 3.3V signals, VCC (To the input of the opto-isolator, next to the IN pins) should be connected to the 3.3V device's +3.3V supply.

NOTE: Some Raspberry-Pi users have found that some relays are reliable and others do not actuate sometimes. It may be necessary to change the value of R1 from 1000 ohms to something like 220 ohms, or supply +5V to the VCC connection.

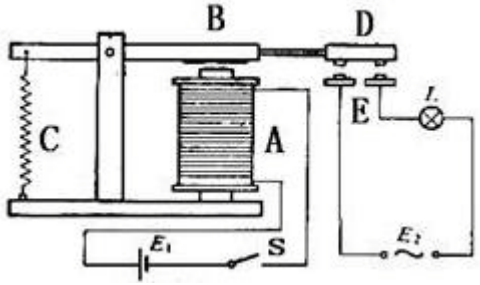
NOTE: The digital inputs from Arduino are Active LOW: The relay actuates and LED lights when the input pin is LOW, and turns off on HIGH.

Module Layout:



Operating Principle:

See the picture below: A is an electromagnet, B armature, C spring, D moving contact, and E fixed contacts. There are two fixed contacts, a normally closed one and a normally open one. When the coil is not energized, the normally open contact is the one that is off, while the normally closed one is the other that is on.



Supply voltage to the coil and some currents will pass through the coil thus generating the electromagnetic effect. So the armature overcomes the tension of the spring and is attracted to the core, thus closing the moving contact of the armature and the normally open (NO) contact or you may say releasing the former and the normally closed (NC) contact. After the coil is de-energized, the electromagnetic force disappears and the armature moves back to the original position, releasing the moving contact and normally closed contact. The closing and releasing of the contacts results in power on and off of the circuit.

Input:

VCC : Connected to positive supply voltage (supply power according to relay voltage)

GND : Connected to supply ground.

IN1: Signal triggering terminal 1 of relay module

IN2: Signal triggering terminal 2 of relay module

Output:

Each module of the relay has one NC (normally close), one NO (normally open) and one COM (Common) terminal. So there are 2 NC, 2 NO and 2 COM of the channel relay in total. NC stands for the normal close port contact and the state without power. NO stands for the normal open port contact and the state with power. COM means the common port. You can choose NC port or NO port according to whether power or not.

Testing Setup:

When a low level is supplied to signal terminal of the 2-channel relay, the LED at the output terminal will light up. Otherwise, it will turn off. If a periodic high and low level is supplied to the signal terminal, you can see the LED will cycle between on and off.

For Arduino：

Step 1：

Connect the signal terminal IN1、IN2 of 2-channel relay to digital pin 4 & 5 of the Arduino Uno or ATMega2560 board, and connect an LED at the output terminal.

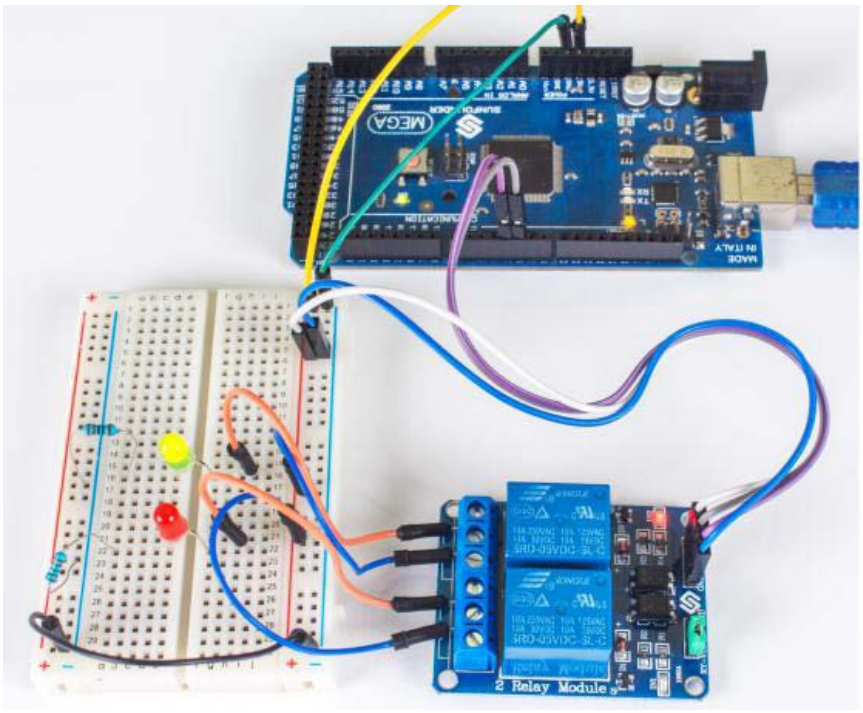
IN1> 4

IN2> 5

Step 2:

Upload the sketch "text_code" to the Arduino Uno or ATMega2560 board.Then you can see the LED cycle between on and off.

The actual figure is shown below:



For raspberry Pi:

Step1:

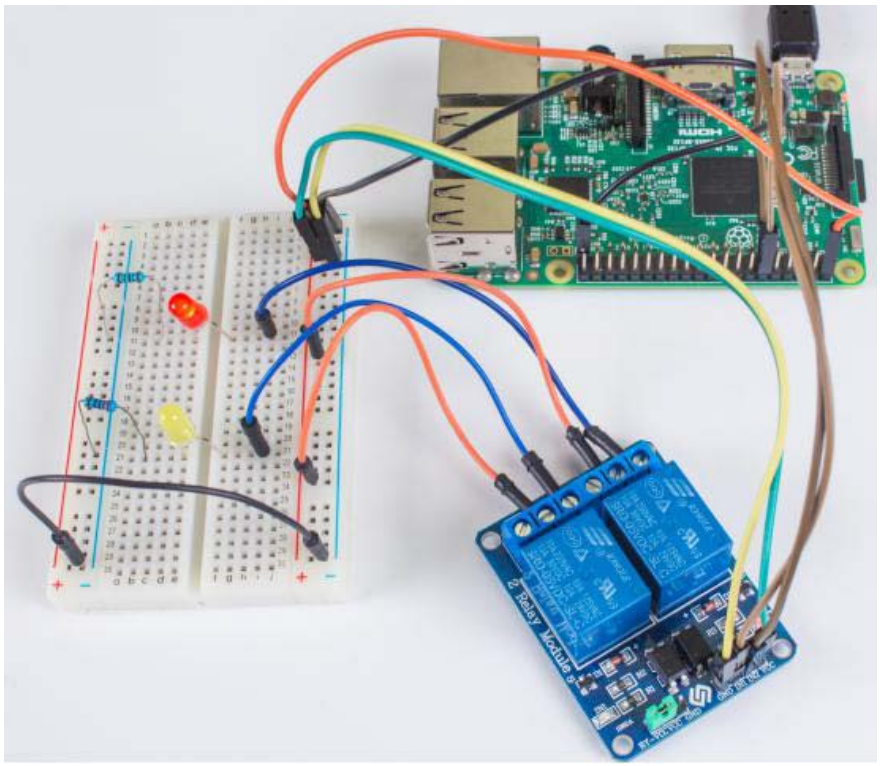
Connect the signal terminal IN2、IN1 of 2-channel relay to port 17、18 of the Raspberry Pi, and connect an LED at the output terminal.

IN2 > 17

IN1 > 18

Step 2:

Run the “test_code”. Then you can see the LED cycle between on and off.



Sketch for Arduino:

```
/*************************************************
   Name：_2_channel_relay
   Description: control the 2 channel relay module to ON or OFF
   Website: www.handsontec.com
   Email: techsupport@handsontec.com
**************************************************/

//the relays connect to
int IN1 = 4;
int IN2 = 5;

#define ON   0
#define OFF  1

void setup()
{
  relay_init();//initialize the relay
}

void loop() {
  relay_SetStatus(ON, OFF);//turn on RELAY_1
```

```
  delay(2000);//delay 2s
  relay_SetStatus(OFF, ON);//turn on RELAY_2
  delay(2000);//delay 2s
}
void relay_init(void)//initialize the relay
{
  //set all the relays OUTPUT
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  relay_SetStatus(OFF, OFF); //turn off all the relay
}
//set the status of relays
void relay_SetStatus( unsigned char status_1,  unsigned char status_2)
{
  digitalWrite(IN1, status_1);
  digitalWrite(IN2, status_2);
}
```

Code for Raspberry Pi:

```
#!/usr/bin/env python
'''
**********************************************************************
* Filename      : 2_channel_relay.py
* Description   : a sample script for 2-Channel High trigger Relay
* E-mail        : techsupport@handsontec.com
* Website       : www.handsontec.com
* Detail        : New file
**********************************************************************
'''
import RPi.GPIO as GPIO
from time import sleep

Relay_channel = [17, 18]

def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(Relay_channel, GPIO.OUT, initial=GPIO.LOW)
    print "|=====================================================|"
    print "|         2-Channel High trigger Relay Sample         |"
    print "|-----------------------------------------------------|"
    print "|                                                     |"
    print "|          Turn 2 channels on off in orders           |"
    print "|                                                     |"
    print "|                    17 ===> IN2                      |"
    print "|                    18 ===> IN1                      |"
    print "|                                                     |"
    print "|                                                     |"
    print "|=====================================================|"

def main():
    while True:
        for i in range(0, len(Relay_channel)):
            print '...Relay channel %d on' % i+1
            GPIO.output(Relay_channel[i], GPIO.HIGH)
            sleep(0.5)
            print '...Relay channel %d off' % i+1
            GPIO.output(Relay_channel[i], GPIO.LOW)
            sleep(0.5)

def destroy():
    GPIO.output(Relay_channel, GPIO.LOW)
    GPIO.cleanup()
```

```
if __name__ == '__main__':
    setup()
    try:
        main()
    except KeyboardInterrupt:
        destroy()
```