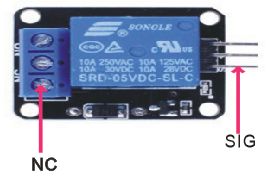
Relay  


Overview

A relay is an electrically-operated switch, often used when a low-power circuit or signal has to control a higher-power circuit. For example, computers like the Raspberry Pi or microcontrollers like found in an Arduino operate in the range of 3.3V/5V DC, where household electrical supply operates at much higher voltages: 110V/220V AC. Using a relay, a lower-powered driver like the Raspberry Pi could control a highered-power circuit like a house light. Originally, relays were used extensively in long-distance telephone systems as amplifiers, with the long-distance system made up of multiple separate circuits arranged end-to-end, each of which “controlled” the next with a relay.

In this experiment, you’ll make your Raspberry operate an electromagnetic relay on one *control circuit* that open and closes a separate *working circuit* on which there is an LED. (In this demonstration, since both circuits are being powered by your Raspberry Pi, they operate at the same power level, but in the more general application of relays, the *working circuit* would operate at a higher power level.)

Materials Needed

Raspberry Pi x1

Breadboard x1

Relay module x1

LED (3-pin) x1

Resistor(330Ω) x1

Dupont jumper wires

Experimental Procedure

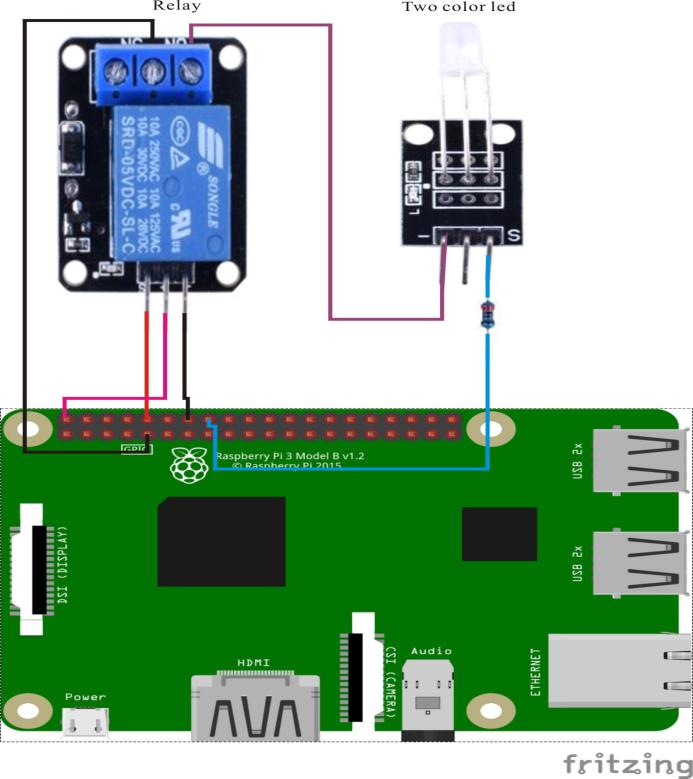
1. If you have not done so already, prepare your development system by installing the Python interpreter, RPi.GIO library, and wiringPi library as described in READ\_ME.TXT.
2. Install the relay and three-pin LED on your breadboard, connecting them with Dupont jumper wires and the resistor to your Raspberry Pi as illustrated in the Wiring Diagram below. Note you will connect only two of the three pins on the LED.
3. Execute the sample stored in this experiment’s subfolder.

If using C, compile and execute the C code:  
cd Code/C  
gcc relay.c -o relay.out –lwiringPi  
./relay.out

If using Python, launch the Python script:  
cd Code/Python  
python relay.py

1. Observe the LED turning on and off on the workingcircuit as the source code signals the relay on the separate control circuit

Wiring Diagram



Relay pin position:

"S" ↔ Raspberry Pi pin 10

"+" ↔ Raspberry Pi +5V

"-" ↔ Raspberry Pi GND

LED pin position:

"S” ↔ Raspberry Pi pin 16 (through resistor)

"-" ↔ Raspberry Pi GND

Technical Background

◆Operation Voltage: 5V

◆Operation Time: 10ms(max)

◆Insulation Resistance: 100MΩ(min)

◆Ambient Temperature: -25°C~+70°C

◆Ambient Humidity: 45%~85%RH

◆Electrical Level Type: Active High

Sample Code

Python code

#!/usr/bin/env python

import RPi.GPIO as GPIO

import time

RelayPin = 10 # pin10

LedPin = 16

def setup():

GPIO.setmode(GPIO.BOARD) # Numbers GPIOs by physical location

GPIO.setup(RelayPin, GPIO.OUT)

GPIO.setup(LedPin, GPIO.OUT)

GPIO.output(LedPin, GPIO.HIGH)

def loop():

while True:

print '...relayd on'

GPIO.output(RelayPin, GPIO.HIGH)

time.sleep(0.5)

print 'relay off...'

GPIO.output(RelayPin, GPIO.LOW)

time.sleep(0.5)

def destroy():

GPIO.output(RelayPin, GPIO.HIGH)

GPIO.cleanup() # Release resource

if \_\_name\_\_ == '\_\_main\_\_': # Program start from here

setup()

try:

loop()

except KeyboardInterrupt:

destroy()

C code

#include <wiringPi.h>

#include <stdio.h>

#define RelayPin 16

#define LedPin 4

int main(void)

{

if(wiringPiSetup() == -1)

{

printf("setup wiringPi failed !");

return 1;

}

pinMode(RelayPin, OUTPUT);

pinMode(LedPin, OUTPUT);

digitalWrite(LedPin, HIGH);

while(1)

{

digitalWrite(RelayPin, HIGH);

delay(500);

digitalWrite(RelayPin, LOW);

delay(500);

}

return 0;

}