Button Switch

Overview

This course uses the Raspberry Pi to capture button signal, which control the LED light on and off.

Experimental Materials

RaspberryPi \*1

Breadboard \*1

Button switch \*1

Led \*1

Dupont Line

Ready to work

1. Install python interpreter in your Raspberry Pi system

2. Install the RPi.GPIO library in your Raspberry Pi system

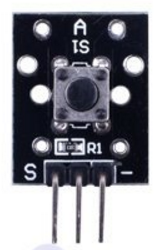
3. Install the wiringPi library in your Raspberry Pi system

Refer to the attached "Installing a Python Interpreter and Corresponding Libraries in a Raspberry Pi System" for details.

product description

1. Introduction:

When button module is not pressed, the pin S of the button is connected to VCC through the pull-up resistor. At this time, the electric level of the pin S is high; when the button is pressed, pin S and pin GND are connected, the level of the pin S becomes low. And then we can make the corresponding action according to this electric level change. In this experiment, the button was used to control the LED light on and off. Buttons are widely used in audio and video products, communications products, medical equipment, security products, toys, digital products, fitness equipment and other fields.



1. characteristic parameters

◆Rated Range: 50mA 12VDC

◆Contact resistance: 50mΩ max(initial)

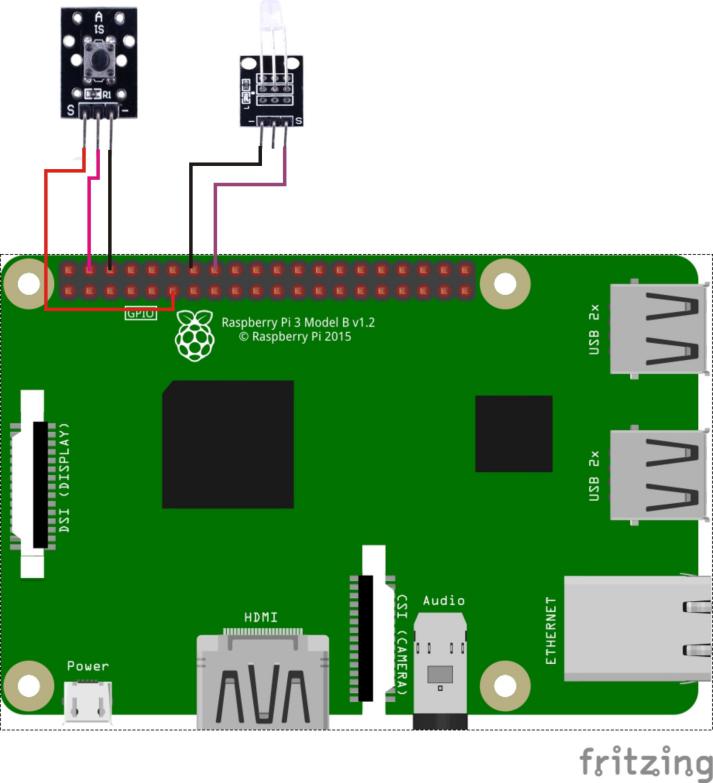
◆Insulation resistance: 100MΩ (DC250V)

◆[withstand](D:/%E7%BD%91%E6%98%93%E6%9C%89%E9%81%93%E8%AF%8D%E5%85%B8/Dict/8.0.0.0/resultui/html/index.html" \l "/javascript:;) [voltage](D:/%E7%BD%91%E6%98%93%E6%9C%89%E9%81%93%E8%AF%8D%E5%85%B8/Dict/8.0.0.0/resultui/html/index.html" \l "/javascript:;) [strength](D:/%E7%BD%91%E6%98%93%E6%9C%89%E9%81%93%E8%AF%8D%E5%85%B8/Dict/8.0.0.0/resultui/html/index.html" \l "/javascript:;): AC 250V(50/60Hz for 1 minute)

◆Environmental temperature: -25°C~+105°C

◆ Heat distortion temperature: 250°C~280°C

Wiring diagram



Sample code

1、python code

#!/usr/bin/env python

import RPi.GPIO as GPIO

BtnPin = 11

LedPin = 16

Led\_status = 0

def setup():

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

GPIO.setup(LedPin, GPIO.OUT) # Set LedPin's mode is output

GPIO.setup(BtnPin, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

GPIO.output(LedPin, GPIO.LOW) # Set LedPin low to off led

def swLed(ev=None):

global Led\_status

Led\_status = not Led\_status

GPIO.output(LedPin, Led\_status)

print "LED: on " if Led\_status else "LED: off"

def loop():

GPIO.add\_event\_detect(BtnPin, GPIO.FALLING, callback=swLed, bouncetime=200) # wait for falling

while True:

pass # Don't do anything

def destroy():

GPIO.output(LedPin, GPIO.LOW) # led off

GPIO.cleanup() # Release resource

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

setup()

try:

loop()

except KeyboardInterrupt:

destroy()

1. C code

#include <wiringPi.h>

#include <stdio.h>

#define BtnPin 0

#define LedPin 4

void myBtnISR(void)

{

digitalWrite(LedPin, !digitalRead(LedPin));

printf("Button is pressed\n");

}

int main(void)

{

if(wiringPiSetup() == -1){ //when initialize wiring failed,print messageto screen

printf("setup wiringPi failed !");

return 1;

}

if(wiringPiISR(BtnPin, INT\_EDGE\_FALLING, myBtnISR)){

printf("setup ISR failed !");

return 1;

}

pinMode(LedPin, OUTPUT);

while(1);

return 0;

}

Experimental phenomena

Whenever the button is pressed, the status of the LED lamp will change.