Tilt Switch  


Overview

A tilt switch detects changes in orientation or inclination through a simple mechanical principle. They consist of an internal cavity or tube partly occupied by a rolling conductive material. Historically, this was a drop of liquid mercury, but to avoid eliminate toxic heavy metals, today they contain metal balls. When the tilt switch is tilted in one direction, the ball rolls onto two internal leads, forming a circuit between them and closing the switch. Tilted the other way, the ball rolls off the contacts, breaking the circuit. Inexpensive (especially compared to accelerometers) and durable, tilt switches are found in toys, safety applications, and other many other consumer devices.

The Tilt Switch reports the module’s current orientation through a single digital output pin, which is LOW when the switch is open and HIGH when the switch is closed. In this experiment, you’ll use a Raspberry Pi to read this pin, and turn on an LED when the switch is tilted.

Experimental Materials

Raspberry Pi x1

Breadboard x1

Tilt Switch x1

LED (3-pin) x1

Resistor(330Ω) x1

Dupont jumper wires

Experimental Procedure

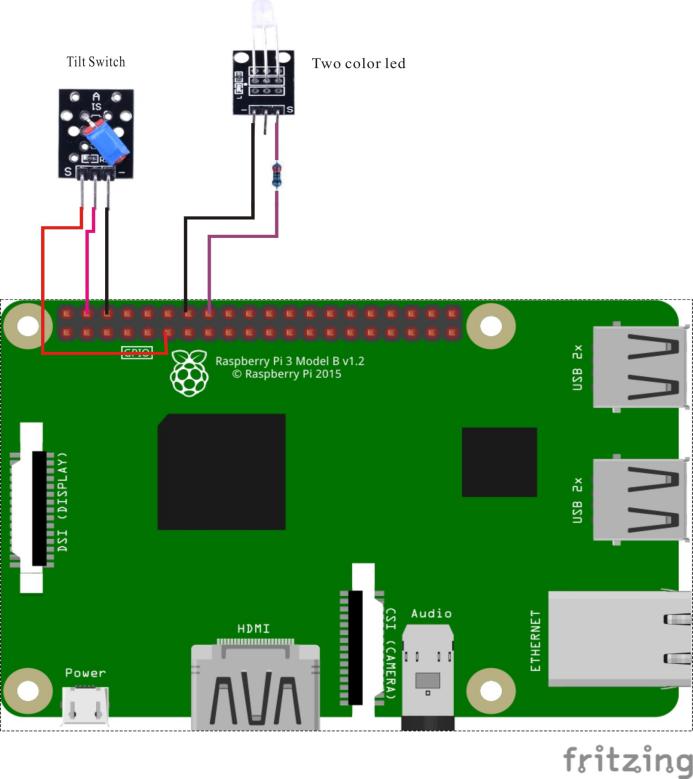
1. If you have not done so already, prepare your development system by installing the Python interpreter, RPi.GIO library, and wiringPi library as described in READ\_ME.TXT.
2. Install the tilt switch and three-pin LED on your breadboard, and use the resistor and Dupont jumper wires as illustrated in 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 tiltSwitch.c -o tiltSwitch.out –lwiringPi  
./tiltSwitch.out

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

1. Make experimental observations. As you move the tilt switch from horizontal to inclined, and back the LED turns on and off as the internal rolling ball makes, and breaks, contact with the internal switch.

Wiring Diagram



Tilt Switch pin position:

"S" ↔ Raspberry Pi pin 11

"+" ↔ Raspberry Pi +5V

"-" ↔ Raspberry Pi GND

LED pin position:

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

"-" ↔ Raspberry Pi GND

Sample Code

Python code

#!/usr/bin/env python

import RPi.GPIO as GPIO

TiltPin = 11

LedPin = 16

Led\_status = 1

def setup():

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

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

GPIO.setup(TiltPin, 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) # switch led status(on-->off; off-->on)

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

def loop():

GPIO.add\_event\_detect(TiltPin, GPIO.FALLING, callback=swLed, bouncetime=100) # 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()

C code

#include <stdio.h>

#include <string.h>

#include <errno.h>

#include <stdlib.h>

#include <wiringPi.h>

#define TiltPin 0

#define LedPin 4

int main(void)

{

if(wiringPiSetup() < 0)

{

printf( " setup wiringPi failed!\n");

return -1;

}

pinMode(TiltPin, INPUT);

pinMode(LedPin, OUTPUT);

while(1)

{

if(0 == digitalRead(TiltPin))

{

digitalWrite(LedPin, LOW);

}

else

{

digitalWrite(LedPin, HIGH);

}

}

return 0;

}