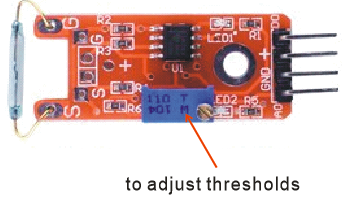
Reed Switch  


Overview

A reed switch is a type of switch in which the open gap between two wires separated in a sealed glass tube can be closed by introducing the presence of a nearby magnet. Compared to more recently developed Hall effect sensors (which also detect magnetic fields), they are electro-mechanical rather than solid-state in operation. However, over a long history of improvement, reed switches’ reliability and low cost have kept them popular in many applications, such as airbag mechanisms in automotive safety systems.

This experiment uses the Raspberry Pi to drive an LED that illuminates when a magnet is positioned near the reed switch.

Experimental Materials

Raspberry Pi x1

Breadboard x1

Reed Switch sensor x1

Resistor (330Ω) x1

LED (3 pin) x1

Dupont jumper wires

Any magnet (you provide)

Experimental Procedure

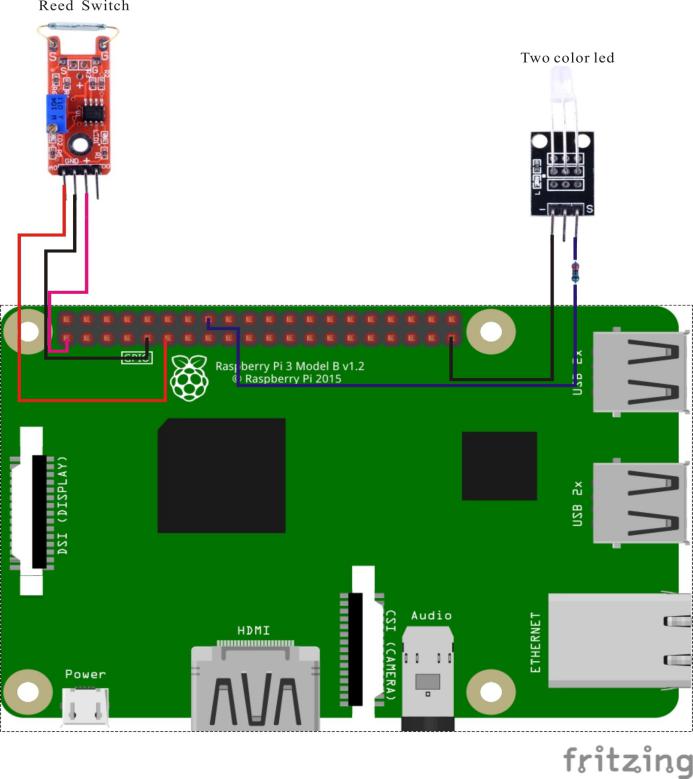
1. If you have not done so already, prepare your development system by installing the Python interpreter, RPi.GPIO library, and wiringPi library as described in READ\_ME.TXT.
2. Install the reed switch sensor, three-pin LED and resistor on your breadboard, and use Dupont jumper wires to connect them to each other and 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 reedSwitch.c -o reedSwitch.out –lwiringPi  
./reedSwitch.out

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

1. Execute the sample stored in this experiment’s subfolder. Make experimental observations. When you hold your magnet close to the sensor, the LED comes on, and goes off when you remove the magnet.(If the LED does not respond, you can adjust the sensitivity by turning the trimmer potentiometer on the sensor board.)

Wiring Diagram



Reed Switch pin position:

A0 ↔ Raspberry Pi Pin 12

D0 ↔ None

GND ↔ Raspberry Pi GND

"+" ↔ Raspberry Pi +3.3V

LED pin position:

"S" ↔ RaspberryPi Pin 16(through resistor)

"-" ↔ Raspberry Pi GND

Sample Code

Python Code

#!/usr/bin/env python

import RPi.GPIO as GPIO

import ADC0832

import time

Reed\_DO\_PIN = 15

LedPin = 16

thresholdVal = 100

def init():

GPIO.setmode(GPIO.BOARD)

GPIO.setup(Reed\_DO\_PIN, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

GPIO.setup(LedPin, GPIO.OUT)

ADC0832.setup()

def loop():

while True:

global digitalVal, analogVal

digitalVal = GPIO.input(Reed\_DO\_PIN)

if(digitalVal == 1):

print 'DO is %d' % digitalVal

analogVal = ADC0832.getResult(0)

print 'Current analog value is %d'% analogVal

if(analogVal < thresholdVal):

GPIO.output(LedPin, GPIO.HIGH)

time.sleep(0.2)

else:

GPIO.output(LedPin, GPIO.LOW)

if \_\_name\_\_ == '\_\_main\_\_':

init()

try:

loop()

except KeyboardInterrupt:

ADC0832.destroy()

print 'The end !'

C Code

#include <wiringPi.h>

#include <stdio.h>

#include <string.h>

#include <errno.h>

#include <stdlib.h>

#define ADC\_CS 0

#define ADC\_CLK 1

#define ADC\_DIO 2

#define Reed\_DO\_Pin 3

#define LedPin 4

#define thresholdVal 100

typedef unsigned char uchar;

typedef unsigned int uint;

uchar get\_ADC\_Result(void)

{

uchar i;

uchar dat1=0, dat2=0;

digitalWrite(ADC\_CS, 0);

digitalWrite(ADC\_CLK,0);

digitalWrite(ADC\_DIO,1); delayMicroseconds(2);

digitalWrite(ADC\_CLK,1); delayMicroseconds(2);

digitalWrite(ADC\_CLK,0);

digitalWrite(ADC\_DIO,1); delayMicroseconds(2);

digitalWrite(ADC\_CLK,1); delayMicroseconds(2);

digitalWrite(ADC\_CLK,0);

digitalWrite(ADC\_DIO,0); delayMicroseconds(2);

digitalWrite(ADC\_CLK,1);

digitalWrite(ADC\_DIO,1); delayMicroseconds(2);

digitalWrite(ADC\_CLK,0);

digitalWrite(ADC\_DIO,1); delayMicroseconds(2);

for(i=0;i<8;i++)

{

digitalWrite(ADC\_CLK,1); delayMicroseconds(2);

digitalWrite(ADC\_CLK,0); delayMicroseconds(2);

pinMode(ADC\_DIO, INPUT);

dat1=dat1<<1 | digitalRead(ADC\_DIO);

}

for(i=0;i<8;i++)

{

dat2 = dat2 | ((uchar)(digitalRead(ADC\_DIO))<<i);

digitalWrite(ADC\_CLK,1); delayMicroseconds(2);

digitalWrite(ADC\_CLK,0); delayMicroseconds(2);

}

digitalWrite(ADC\_CS,1);

pinMode(ADC\_DIO, OUTPUT);

return(dat1==dat2) ? dat1 : 0;

}

int main(void)

{

uchar digitalVal = 1;

uchar analogVal = 0;

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

printf("setup wiringPi failed !");

return 1;

}

pinMode(ADC\_CS, OUTPUT);

pinMode(ADC\_CLK, OUTPUT);

pinMode(Reed\_DO\_Pin, INPUT);

pullUpDnControl(Reed\_DO\_Pin, PUD\_UP);

pinMode(LedPin, OUTPUT);

while(1){

if((digitalVal = digitalRead(Reed\_DO\_Pin)))

{

printf("Do is %d.\n", digitalVal);

analogVal = get\_ADC\_Result();

printf("Current analog value is %d.\n", analogVal);

if(analogVal < thresholdVal)

{

digitalWrite(LedPin, HIGH);

}

delay(200);

}

else

{

digitalWrite(LedPin, LOW);

}

}

return 0;

}

Characteristic Parameters

◆Using normally open reed switch

◆Comparator output, clean signal, good waveform, strong driving ability, over 15mA

◆Working voltage 3.3V-5V

◆Output form : Digital switch output (0 and 1)

◆with a fixed bolt hole, easy to install  
◆Use Wide Voltage LM393 Comparator