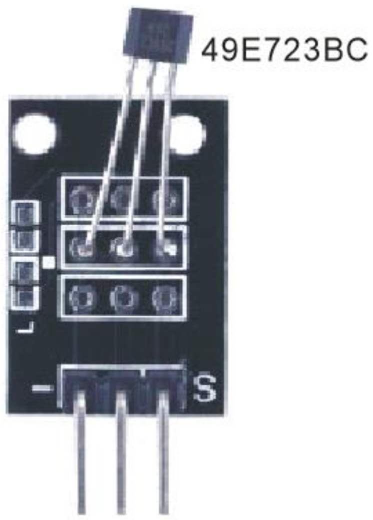
Analog Hall Sensor



Overview

Hall effects are magnetic sensors, and vary their voltage output in relation to a detected magnetic field. They are used to detect proximity, position, speed, and current. Analog Hall sensors report a signal proportional to the magnetic field strength (an analog quantity) rather than act as a discrete (digital) switch indicating a magnet's "presence" or "absence." This experiment uses the Raspberry Pi to measure the signal of a simple analog Hall sensor and drive a blinking LED based on the captured signal.

Experimental Materials

Raspberry Pi	x1
Breadboard	x1
Analog Hall sensor	x1
ADC0832	x1
LED (3-pin)	x1
Resistor (330Ω)	x1
Dupont jumper wires	
any magnet	x1 (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_FIRST.TXT.

2. Install the ADC0832 analog/digital converter IC, analog Hall effect 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 analogHall.c -o analogHall.out –lwiringPi
   ./analogHall.out
   ```

   If using Python, launch the Python script:

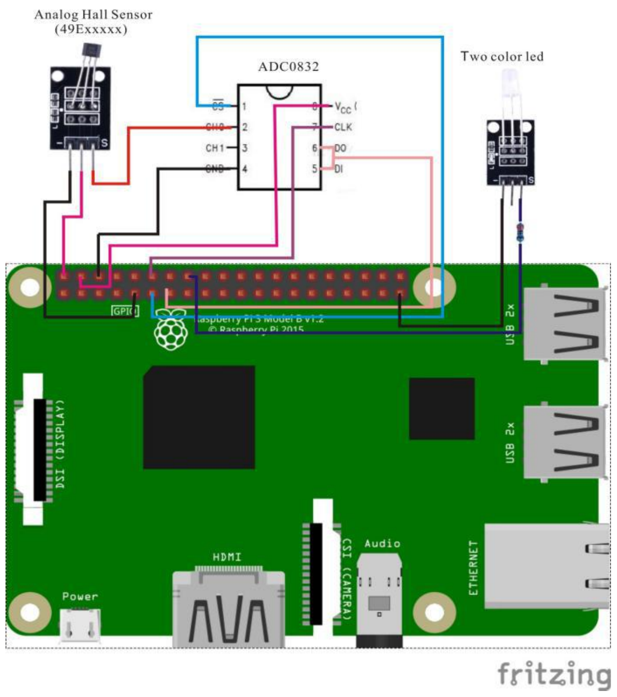
   ```
   cd Code/Python
   python analogHall.py
   ```

4. Make experimental observations. When you hold your magnet vertically close to the sensor, the Hall effect generates an (analog) voltage, which the ADC converts to a (digital) signal readable by the RaspberryPi. The sample code then turns on the LED if that voltage exceeds a certain threshold.

   Depending on the location of magnet and sensor, Hall effects can be useful in many applications. When the magnet is mounted on a door and the sensor on the doorframe, they can be used in burglar alarms to answer "is the door open?" When the magnet is mounted on a rotating wheel and the sensor on a fixed point next to the wheel, they can be used in speedometers and odometers to answer "has the wheel made a complete revolution?"

www.kumantech.com

Wiring Diagram



ADC0382 pin position:

CS	↔	Raspberry Pi pin 11
CLK	↔	Raspberry Pi pin 12
DI	↔	Raspberry Pi pin 13
D0	↔	Raspberry Pi pin 13
CH0	↔	Analog Hall Sensor pin "S"
VCC	↔	Raspberry Pi +5V
GND	↔	Raspberry Pi GND

Analog Hall pin position:

"S"	↔	ADC0382 pin CH0
"+"	↔	Raspberry Pi +5V
"-"	↔	Raspberry Pi GND

LED pin position:

"S"	↔	Raspberry Pi pin 16(through resistor)
"-"	↔	Raspberry Pi GND

Technical Background

When there is current at both ends of a semiconductor sheet and a uniform magnetic field with a magnetic induction strength is applied perpendicular to the sheet, a Hall effect with a potential difference of UH will be generated in the direction perpendicular to the current and the magnetic field. Hall elements— semiconductor components sensitive to the Hall effect—have many advantages as mechanical sensors (sensitivity to magnetic field, simple structure, small volume, wide frequency response, large output voltage change, long service life, etc.) and have been widely used in measurement, automation, computer and information technology. A voltage difference is generated when the Hall element and the magnet meet in the forward direction, and there is no voltage difference when the Hall element and the magnet meet in the reverse direction. You can measure this voltage difference with the Raspberry Pi, and then judge the proximity of the magnet. The sample experiment uses this distance to control the LED light. For more information, see https://en.wikipedia.org/wiki/Hall_effect_sensor.

Sample Code

Python Code

```
#!/usr/bin/env python
import ADC0832
import time
import RPi.GPIO as GPIO

LedPin = 16
thresholdVal = 150
def init():
   ADC0832.setup()
   GPIO.setup(LedPin, GPIO.OUT)

def loop():
   while True:
      analogVal = ADC0832.getResult(0)
      print 'analog value is %d' % analogVal
      if(analogVal < thresholdVal):
         GPIO.output(LedPin, GPIO.HIGH)
      else:
         GPIO.output(LedPin, GPIO.LOW)
        time.sleep(0.2)
```

```
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      LedPin     4
#define      thresholdVal   150

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 analogVal;
   if(wiringPiSetup() == -1)
   {
      printf("setup wiringPi failed !");
      return -1;
   }

   pinMode(ADC_CS,  OUTPUT);
   pinMode(ADC_CLK, OUTPUT);
   pinMode(LedPin, OUTPUT);

   while(1)
   {
      analogVal = get_ADC_Result();
```

```
        printf("Current analog : %d\n", analogVal);
        if(analogVal < thresholdVal)
        {
            digitalWrite(LedPin, HIGH);
        }
        else
        {
            digitalWrite(LedPin, LOW);
        }
        delay(200);
    }

    return 0;
}
```