AZ-Delivery

*Ihr Experte für Mikroelektronik!*

Reed Sensor Modul
Datenblatt
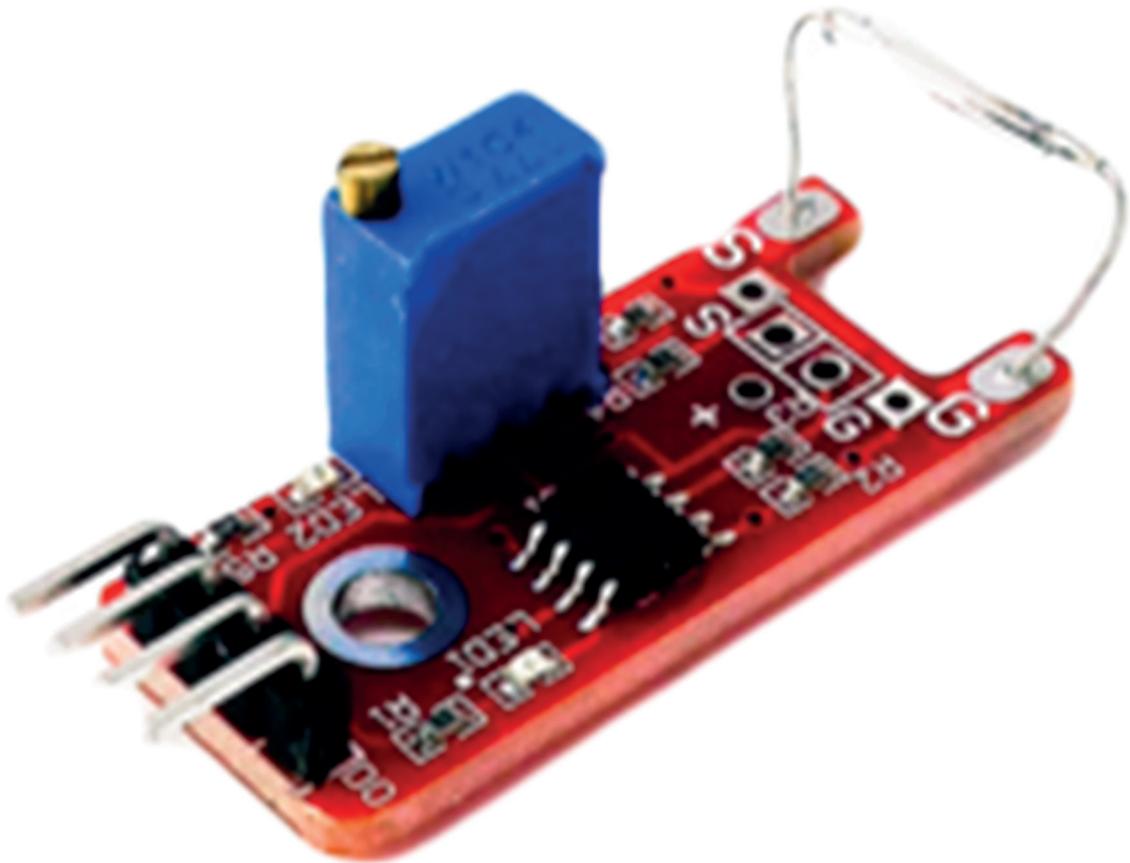
Contents:

1. Short description

If a magnetic field was detected, it will be shown at the digital output.
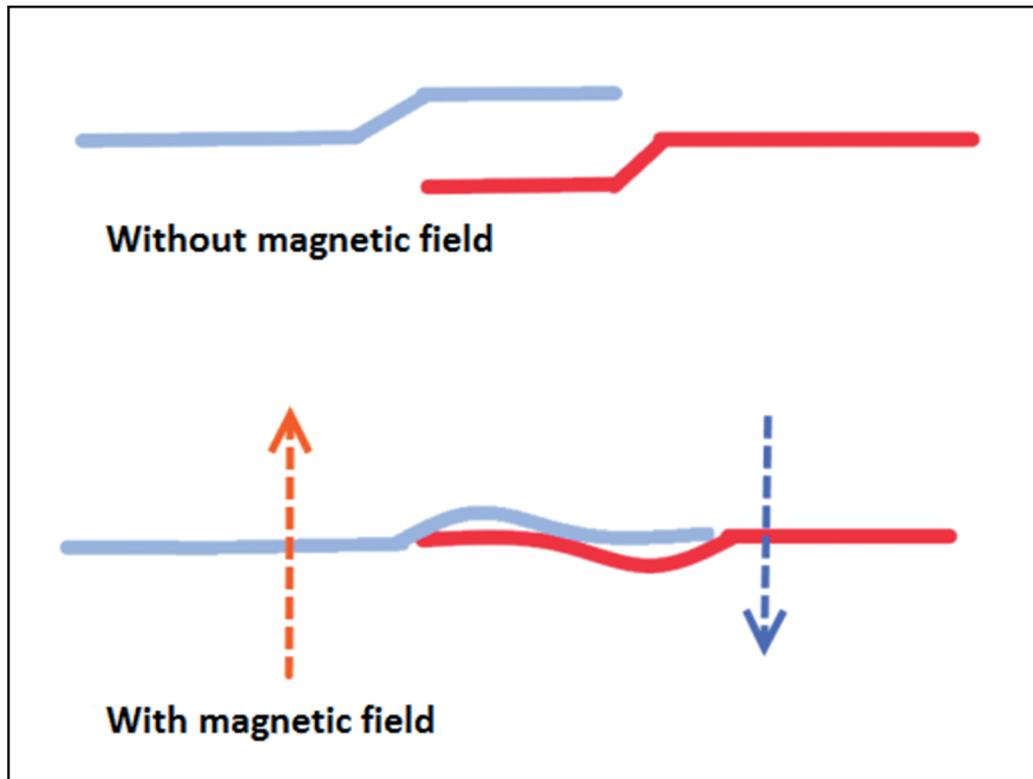
Because of that we will get an electrical connection and send the signal.

Digital Out: After detecting a magnetic field, a signal will be outputted

Analog Out: Direct measurement of the sensor unit

LED1: Shows that the sensor is connected with power

LED2: Shows that the sensor detects a magnetic field

2. Pinout
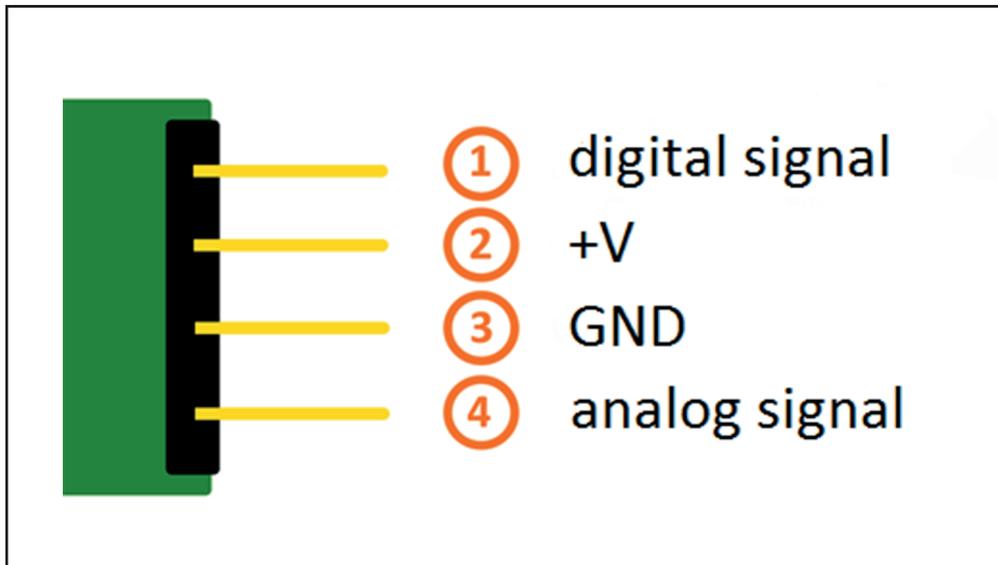
3. Functionality of the sensor

The sensor has 3 main components on its circuit board. First, the sensor unit at the front of the module which measures the area physically and sends an analog signal to the second unit, the amplifier. The amplifier amplifies the signal, according to the resistant value of the potentiometer, and sends the signal to the analog output of the module. The third component is a comparator which switches the digital out and the LED if the signal falls under a specific value.
You can control the sensitivity by adjusting the potentiometer.

Please notice: The signal will be inverted; that means that if you measure a high value, it is shown as a low voltage value at the analog output.
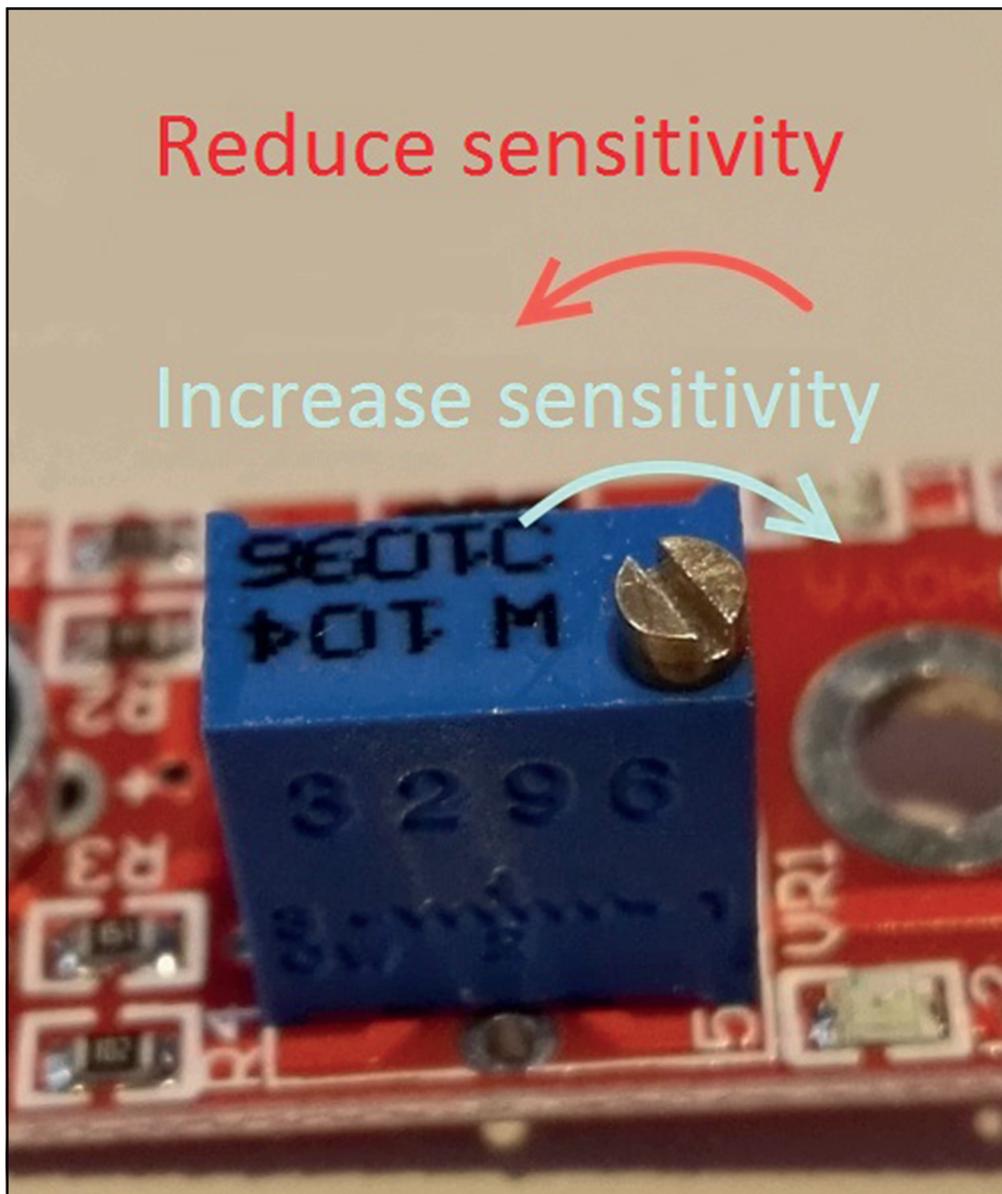
This sensor doesn't show absolute values (like exact temperature in °C or magneticfield strenght in mT).
It is a relative measurement: you define an extreme value to a given normal environment situation and a signal will be send if the measurement exceeds the extreme value.

4. Code example Arduino

The program reads the current voltage value which will be measured at the output pin and shows it via serial interface.

Additionally, the status of the digital pin will be shown at the terminal.

```
1   // Declaration and initialization of the input pin
2   int Analog_Eingang = A0; // X-axis-signal
3   int Digital_Eingang = 3; // Button
4
5   void setup ()
6   {
7     pinMode (Analog_Eingang, INPUT);
8     pinMode (Digital_Eingang, INPUT);
9
10    Serial.begin (9600); // Serial output with 9600 bps
11  }
12
13  // The program reads the current value of the input pins
14  // and outputs it via serial out
15  void loop ()
16  {
17    float Analog;
18    int Digital;
19
20    // Current value will be read and converted to the voltage
21    Analog = analogRead (Analog_Eingang) * (5.0 / 1023.0);
22    Digital = digitalRead (Digital_Eingang);
23
24    // and outputted here
25    Serial.print ("Analog voltage value:"); Serial.print (Analog, 4);  Serial.print ("V, ");
26    Serial.print ("Extreme value:");
27
28    if(Digital==1)
29    {
30        Serial.println (" reached");
31    }
32    else
33    {
34        Serial.println (" not reached yet");
35    }
36    Serial.println ("-----------------------------------------------------------------");
37    delay (200);
38  }
```

Connections Arduino:

digital signal = [Pin 3]
+V= [Pin 5V]
GND = [Pin GND]
analog signal = [Pin 0]

5. Code example Raspberry Pi

Unlike the Arduino, the Raspberry Pi doesn't provide an ADC (Analog Digital Converter) on its Chip. This limits the Raspbery Pi if you want to use a non digital Sensor.

The program reads the current values of the input pins and output it at the terminal in [mV].

Additionally, the status of the digital pin will be shown at the terminal to show if the extreme value was exceeded or not.

```
1   ################################################################################
2   ### Copyright by Joy-IT
3   ### Published under Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License
4   ### Commercial use only after permission is requested and granted
5   ###
6   ### KY-053 Analog Digital Converter - Raspberry Pi Python Code Example
7   ###
8   ################################################################################
9
10
11  # This code is using the ADS1115 and the I2C Python Library for Raspberry Pi
12  # This was published on the following link under the BSD license
13  # [https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code]
14  from Adafruit_ADS1x15 import ADS1x15
15  from time import sleep
16
17  # import needed modules
18  import math, signal, sys, os
19  import RPi.GPIO as GPIO
20  GPIO.setmode(GPIO.BCM)
21  GPIO.setwarnings(False)
22
23  # initialise variables
24  delayTime = 0.5 # in Sekunden
25
26  # assigning the ADS1x15 ADC
27
28  ADS1015 = 0x00  # 12-bit ADC
29  ADS1115 = 0x01  # 16-bit
30
31  # choosing the amplifing gain
32  gain = 4096  # +/- 4.096V
33  # gain = 2048  # +/- 2.048V
34  # gain = 1024  # +/- 1.024V
35  # gain = 512   # +/- 0.512V
36  # gain = 256   # +/- 0.256V
37
38  # choosing the sampling rate
39  # sps = 8     # 8 Samples per second
40  # sps = 16    # 16 Samples per second
41  # sps = 32    # 32 Samples per second
42  sps = 64    # 64 Samples per second
43  # sps = 128   # 128 Samples per second
44  # sps = 250   # 250 Samples per second
45  # sps = 475   # 475 Samples per second
46  # sps = 860   # 860 Samples per second
47
48  # assigning the ADC-Channel (1-4)
49  adc_channel_0 = 0     # Channel 0
50  adc_channel_1 = 1     # Channel 1
51  adc_channel_2 = 2     # Channel 2
52  adc_channel_3 = 3     # Channel 3
53
54  # initialise ADC (ADS1115)
55  adc = ADS1x15(ic=ADS1115)
56
57  # Input pin for the digital signal will be picked here
58  Digital_PIN = 24
59  GPIO.setup(Digital_PIN, GPIO.IN, pull_up_down = GPIO.PUD_OFF)
60
61  ################################################################################
62
63  # ########
64  # main program loop
65  # ########
66  # The program reads the current value of the input pin
67  # and shows it at the terminal
68
69  try:
70          while True:
71                  #Current values will be recorded
72                  analog = adc.readADCSingleEnded(adc_channel_0, gain, sps)
73
74                  # Output at the terminal
75                  if GPIO.input(Digital_PIN) == False:
76                          print "Analog voltage value:", analog,"mV, ","extreme value: not reached"
77                  else:
78                          print "Analog voltage value:", analog, "mV, ", "extreme value: reached"
79                  print "-------------------------------------"
80
81                  sleep(delayTime)
82
83
84
85  except KeyboardInterrupt:
86          GPIO.cleanup()
```

Connections Raspberry Pi:

Sensor

digital signal = GPIO 24 [Pin 18 (RPi)]
+V= 3,3V [Pin 1 (RPi)]
GND = GND [Pin 06 (RPi)]
analog signal = Analog 0 [Pin A0 (ADS1115 - KY-053)]
ADS1115 - KY-053:

VDD = 3,3V [Pin 01]
GND = GND [Pin 09]
SCL = GPIO03 / SCL [Pin 05]
SDA = GPIO02 / SDA [Pin 03]
A0 = look above[Sensor: analog signal]