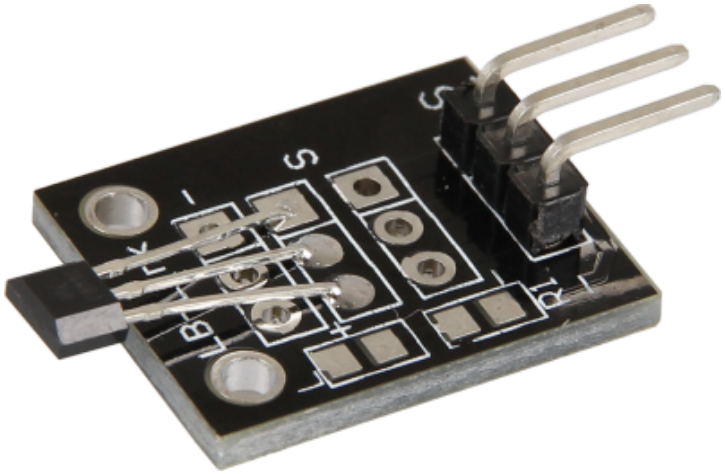
KY-035 Bihor magnetic sensor module

Contents

1 Pictures 1
2 Technical data / Short description 1
3 Pinout 2
4 Code example Arduino 2
5 Code example Raspberry Pi 3

Pictures

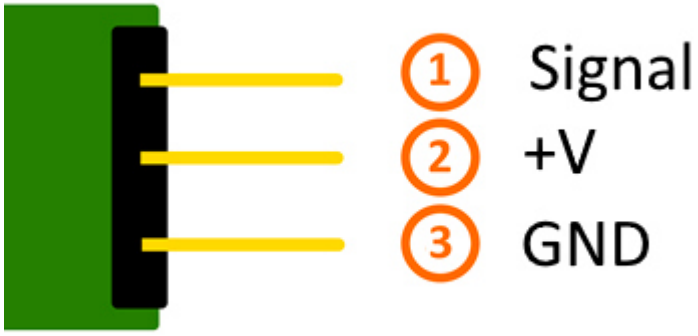


Technical data / Short description

Chipset: AH49E

The sensor measures the current magnetic field, near to the sensor.

Pinout



Code example Arduino

The program measures the current voltage value at the sensor, calculates with it and a known resistor the resistance from the sensor and shows the results via serial output.

```
int sensorPin = A5; // Declaration of the input pin

// Serial OUT in 9600 baud
void setup()
{
        Serial.begin(9600);
}

// The program measures the current voltage at the sensor,
// calculates the resistance with it and a known resistor
// and outputs it via serial OUT

void loop()
{
        // Measuring of the current voltage...
        int rawValue = analogRead(sensorPin);
        float voltage = rawValue * (5.0/1023) * 1000;

        float resitance = 10000 * ( voltage / ( 5000.0 - voltage) );

        // ... output via serial interface
        Serial.print("Voltage:");     Serial.print(voltage); Serial.print("mV");
        Serial.print(", Resistance:"); Serial.print(resitance); Serial.println("Ohm");
        Serial.println("----------------------------------------");

        delay(500);
}
```

Connections Arduino:

Sensor GND = [Pin GND]

Sensor +V = [Pin 5V]

Sensor Signal = [Pin A5]

Example program download

KY-035_Bihor-magnetic-sensor-module

Code example Raspberry Pi

!! Attention !! Analog Sensor !! Attention !!

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.

To evade this, use our *Sensorkit X40* with the *KY-053* module, which provides a 16 Bit ADC, which can be used with the Raspberry Pi, to upgrade it with 4 additional analog input pins. This module is connected via I2C to the Raspberry Pi.
It measures the analog data and converts it into a digital signal which is suitable for the Raspberry Pi.

So we recommend to use the KY-053 ADC if you want to use analog sensors along with the Raspberry Pi.

For more information please look at the infosite: KY-053 Analog Digital Converter

!! Attention !! Analog Sensor !! Attention !!

The program uses the specific ADS1x15 and I2C python-libraries from the company Adafruit to control the ADS1115 ADC. You can find these here: [https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code] published under the BSD-License [Link]. You can find the needed libraries in the lower download package.

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

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

```
##########################################################################################

### Copyright by Joy-IT
### Published under Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License
### Commercial use only after permission is requested and granted
###
### KY-053 Analog Digital Converter - Raspberry Pi Python Code Example
###
##########################################################################################

# This code is using the ADS1115 and the I2C Python Library for Raspberry Pi
# This was published on the following link under the BSD license
# [https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code]
from Adafruit_ADS1x15 import ADS1x15
from time import sleep

# import needed modules
import time, signal, sys, os
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

# initialise variables
delayTime = 0.5 # in Sekunden

# assigning the ADS1x15 ADC
```



```
ADS1015 = 0x00  # 12-bit ADC
ADS1115 = 0x01  # 16-bit

# choosing the amplifing gain
gain = 4096  # +/- 4.096V
# gain = 2048  # +/- 2.048V
# gain = 1024  # +/- 1.024V
# gain = 512   # +/- 0.512V
# gain = 256   # +/- 0.256V

# choosing the sampling rate
# sps = 8    # 8 Samples per second
# sps = 16   # 16 Samples per second
# sps = 32   # 32 Samples per second
sps = 64   # 64 Samples per second
# sps = 128  # 128 Samples per second
# sps = 250  # 250 Samples per second
# sps = 475  # 475 Samples per second
# sps = 860  # 860 Samples per second

# assigning the ADC-Channel (1-4)
adc_channel_0 = 0    # Channel 0
adc_channel_1 = 1    # Channel 1
adc_channel_2 = 2    # Channel 2
adc_channel_3 = 3    # Channel 3

# initialise ADC (ADS1115)
adc = ADS1x15(ic=ADS1115)

##########################################################################################


# ########
# Main Loop
# ########
# Reading the values from the input pins and print to console

try:
        while True:
                #read values
                adc0 = adc.readADCSingleEnded(adc_channel_0, gain, sps)
                adc1 = adc.readADCSingleEnded(adc_channel_1, gain, sps)
                adc2 = adc.readADCSingleEnded(adc_channel_2, gain, sps)
                adc3 = adc.readADCSingleEnded(adc_channel_3, gain, sps)

                # print to console
                print "Channel 0:", adc0, "mV "
                print "Channel 1:", adc1, "mV "
                print "Channel 2:", adc2, "mV "
                print "Channel 3:", adc3, "mV "
                print "---------------------------------------"

                time.sleep(delayTime)


except KeyboardInterrupt:
        GPIO.cleanup()
```

Connections Raspberry Pi:

Sensor

GND	= GND	[Pin 06 (RPi)]
+V	= 3,3V	[Pin 01 (RPi)]

analog Signal = Analog 0 [Pin A0 (ADS1115 - KY-053)]

ADS1115 - KY-053:

VDD	=	3,3V	[Pin 17]
GND	=	GND	[Pin 09]
SCL	=	GPIO03 / SCL	[Pin 05]
SDA	=	GPIO02 / SDA	[Pin 03]
A0	=	look above	[Sensor: analog Signal]

Example program download

KY-053_RPi-AnalogDigitalConverter

To start, enter the command:

```
sudo python KY-053_RPi-AnalogDigitalConverter.py
```