KY-053 Analog digital converter

Contents

Picture

Technical data / Short description

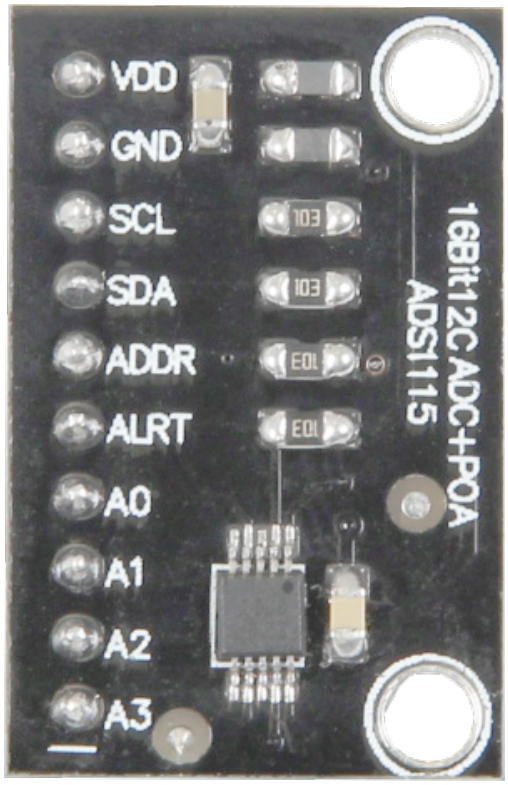
With the right command on the I2C-bus, you are able to measure voltage values with a precision of 16-bit at 4 different input pins.

The result of the measurement will be send encoded via I2C-bus.

For this mode you will need a special software.

Pinout

The Pin connections are printed on the module board



Code example Arduino

The Arduino board includes a 10-bit-ADC with 6 channels by default. If you need more channels or a higher precision you can use the KY053 analog digital converter module to expand the channel by 4 with a precision of 12-bit which will connected with the Arduino via I2C.

You have a few options to control the module - the best choice is to use the ADS1X15 libraries from the company Adafruit which you can get here: [https://github.com/adafruit/Adafruit_ADS1X15] which were published under the open source license: [BSD-Lizenz]

The example below uses these libraries - we advise to download the libraries from Github, to unzip it and to copy it into the Arduino-library-folder, which you can find here : (C:\user\[username] \documents\Arduino\libraries) by default. To use it in the following code example and for following projects, you have to download and copy it. Alternatively, you can get it with the download package below.

```
#include <Adafruit_ADS1015.h>

// ADS1115 module will be initialised
Adafruit_ADS1115 ads;

void setup(void)
{
  Serial.begin(9600);

  Serial.println("Analog input values of the ADS1115 will be read and outputted");
  Serial.println("ADC Range: +/- 6.144V (1 bit = 0.1875mV)");
```

```
  Serial.println("ADC Range: +/- 6.144V (1 bit = 0.1875mV)");

  // The module includes signal amplifier at the inputs
  //                                                                ADS1115
  //                                                                -------
  ads.setGain(GAIN_TWOTHIRDS);  // 2/3x gain +/- 6.144V  1 bit = 0.1875mV
  // ads.setGain(GAIN_ONE);        // 1x gain   +/- 4.096V  1 bit = 0.125mV
  // ads.setGain(GAIN_TWO);        // 2x gain   +/- 2.048V  1 bit = 0.0625mV
  // ads.setGain(GAIN_FOUR);       // 4x gain   +/- 1.024V  1 bit = 0.03125mV
  // ads.setGain(GAIN_EIGHT);      // 8x gain   +/- 0.512V  1 bit = 0.015625mV
  // ads.setGain(GAIN_SIXTEEN);    // 16x gain  +/- 0.256V  1 bit = 0.0078125mV

  ads.begin();
}

void loop(void)
{
  uint16_t adc0, adc1, adc2, adc3;
  float voltage0, voltage1, voltage2, voltage3;
  float gain_conversion_factor;

  // The command "ads.readADC_SignalEnded(0)" is the operation, which starts the mesurement
  adc0 = ads.readADC_SingleEnded(0);
  adc1 = ads.readADC_SingleEnded(1);
  adc2 = ads.readADC_SingleEnded(2);
  adc3 = ads.readADC_SingleEnded(3);

  // You need this value to calculate the voltage - it depends on the configured amplification
  gain_conversion_factor= 0.1875;

  // Calculating of the voltage values from the measured values.
  voltage0 = (adc0 * gain_conversion_factor);
  voltage1 = (adc1 * gain_conversion_factor);
  voltage2 = (adc2 * gain_conversion_factor);
  voltage3 = (adc3 * gain_conversion_factor);

  // The values will be outputted to the serial interface
  Serial.print("Analog input 0: "); Serial.print(voltage0);Serial.println("mV");
  Serial.print("Analog input 1: "); Serial.print(voltage1);Serial.println("mV");
  Serial.print("Analog input 2: "); Serial.print(voltage2);Serial.println("mV");
  Serial.print("Analog input 3: "); Serial.print(voltage3);Serial.println("mV");
  Serial.println("-----------------------");

  delay(1000);
}
```

Example program download:

KY-053_analog-digital-converter_ARD

Connections Arduino:

VDD = [Pin 5V]
GND = [Pin GND]
SCL = [Pin SCL]
SDA = [Pin SDA]
ADDR = [N.C.]
ALRT = [N.C.]
A0 = [peak Analog 0]
A1 = [peak Analog 1]
A2 = [peak Analog 2]

A3 = [peak Analog 3]

Code example Raspberry Pi

The Raspberry Pi has no ADC (Analog Digital Converter) included on its chip. It is a disadvantage if you want to use sensors which doesn't send digital signals like : Voltage range exceeded -> digital ON | Voltage range deceeded -> digital OFF | Example: Button pushed [ON], Button released [OFF], instead of that you get a continuously changing value (example: potentiometer -> different position = different voltage value).

To evade this problem , our sensorkit X40 comes with a 16 bit precise ADC (KY-053), which can be used with the Raspberry Pi, to expand it for 4 analog Input pins. It is connected with the Raspberry Pi via I2C, takes the analog measurement and sends a digital signal to the Raspberry Pi.

The program uses the ADS1x15 and I2C-python-libraries from the company Adafruit to control the ADC. You can get it here: [https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code] it is published under the open source license MIT OpenSource-Lizenz. You can find the needed libraries in the download package below.

The program reads the current voltage value which is at the 4 channels of the ADS1115, and shows it at the terminal. You can configure the break between the measurements with the variable "delayTime".

In order that the Raspberry Pi can communicate with the sensor via I2C-bus , you have to activate the I2C funktion from the Raspberry Pi first. To do this you have to add the following line at the and of the file "/boot /config.txt" :

```
dtparam=i2c_arm=on
```

You can edit the file with the command:

```
sudo nano /boot/config.txt
```

You can save and close the file, after adding of the line, with the key sequence : [Ctrl + X -> Y -> enter]. You will need additional libraries to use I2C in python. To install them you have to use the following command at the console:

```
sudo apt-get install python-smbus i2c-tools -y
```

After that you can use the following code example:

```

```

Connections Raspberry Pi:

VDD	= 3,3V	[Pin 01]
GND	= GND	[Pin 06]
SCL	= GPIO03 / SCL	[Pin 05]
SDA	= GPIO02 / SDA	[Pin 03]
ADDR	= N.C.	[-]

ALRT	= N.C.	[-]
A0	= Measurement peak Analog 0	[Voltage which is to measure | for example sensor output]
A1	= Measurement peak Analog 1	[Voltage which is to measure | for example sensor output]
A2	= Measurement peak Analog 2	[Voltage which is to measure | for example sensor output]
A3	= Measurement peak Analog 3	[Voltage which is to measure | for example sensor output]

Example program download

KY-053_RPi_AnalogDigitalConverter

To start, enter the command:

```
sudo python KY-053_RPi_AnalogDigitalConverter.py
```

Extended function of the ADS1115 ADC

The function of the ADS1115 which is used above is called "Single Ended Conversion" and says that a measurement from a single picked channel will go against GND.

Additional to this kind of measurement, the ADS1115 ADC provides another measurement function, which can measure difference-voltages between two input pins (Example: Voltage between A0 and A1). Additional to the single-ended measurement, you can activate the Comparator function, which only gives you the results if an extreme value of the voltage was exceeded.

These functions how the changing of the sample rate for example, are included in the Adafruit libraries - for more information look into the documentation of the Adafruit libraries.