**Application Note**

**User Guide for**

**FS1012 Flow Sensor with Arduino Uno**

**Issue Date: 01/26/2017**

This application note is a guide for how to use the FS1012 an analog flow sensor with Arduino Uno.

Table of Contents

[1 Introduction 3](#_Toc38444817)

[2 Getting Started 3](#_Toc38444818)

[3 Materials 3](#_Toc38444819)

[4 Hardware Setup 4](#_Toc38444820)

[5 Software 5](#_Toc38444821)

[6 Scripting 5](#_Toc38444822)

[7 How to Run the Application 7](#_Toc38444823)

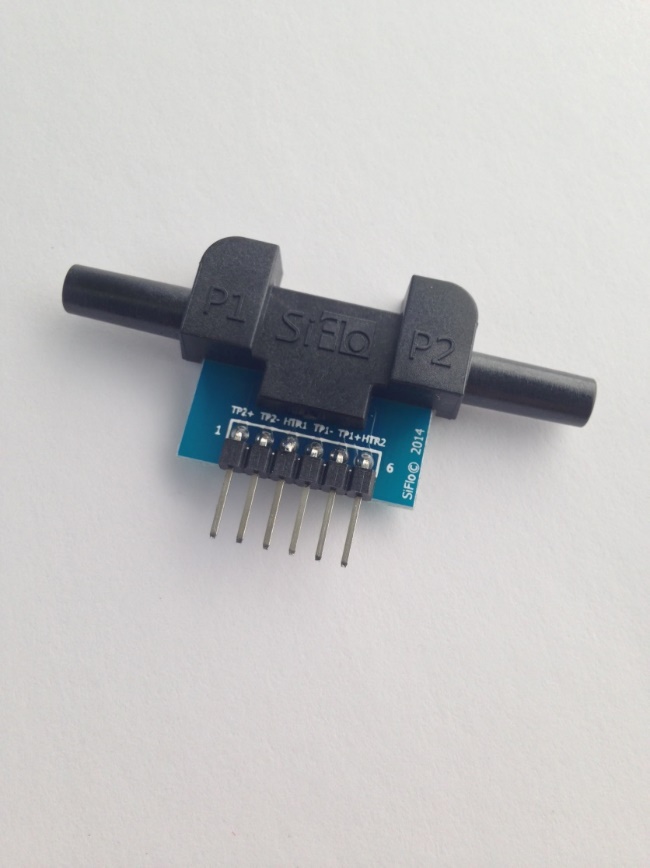
[8 Arduino Uno Script 8](#_Toc38444824)

[9 Disclaimer 9](#_Toc38444825)

# Introduction

The FS1012 analog flow sensor is a ready to use flow sensor for gases and liquids[[1]](#footnote-1). The FS1012 analog flow sensor includes a MEMS flow sensor die mounted on a printed circuit board (PCB), a molded flow housing and a standard 6-pin male connector. Getting started is easy. Simply connect a tube to the flow housing with a gas or fluid and supply power to the sensor to start measuring flow rates. A step-by-step procedure to measure flow using Arduino Uno has been included in section 4.

The FS1012 analog flow sensor is targeted for applications with gas flow rates up to 10 liters per minute. A unique feature of this model is the ability to measure flow in two directions, from P1 to P2 and from P2 to P1. (Please refer to Figure 1 below).

****

***Figure 1:*** *FS1012 Analog Flow Sensor*

# Getting Started

The aim of this user guide is to show how easy it is to get started with measuring flow using the Arduino Uno kit. A flow sensor is a device for measuring the rate of fluid or gas. The FS1012 is an analog sensor and the measurement setup described in this document performs analog measurements. This is meant as starting point to develop applications to measure flow in various applications. Note: Absolute accuracy of a measurement will require flow sensor calibration.

# Materials

* Arduino UNO
* FS1012 analog flow sensor
* Hookup wires
* Optional: external power supply (Recommended)

# Hardware Setup

Here follows a step-by-step guide for how to perform flow measurements using the Arduino Uno kit with the flow direction from P1 to P2 (Please refer to Figure 2 for a graphical depiction):

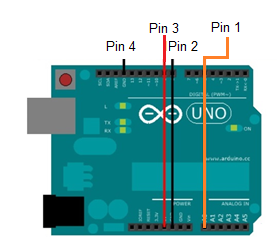
1. Connect a wire from 5V on the Arduino Uno board to pin 3 (HTR1) on FS1012.
2. Connect a wire from GND on the Arduino Uno board to pin 4 (HTR2) on FS1012.
3. Connect a wire from A0 on the Arduino Uno board to pin 1 (TP1+) on FS1012.
4. Connect a wire from GND on the Arduino Uno board to pin 2 (TP1-) on FS1012.

***Tip:*** *It is recommended to use a separate power supply. The 5V from the Arduino Uno board might not be very stable.*

1. Connect a tube with 5 mm internal diameter to the flow housing port P1.

***Tip:*** *It is recommended to use a flexible plastic tube that is made from a tacky material, for example, Tygon. This helps seal the interface between the flow housing and tubing.*

1. The hardware setup is now ready to measure flow.



***Figure 2:*** *Flow Measurement, P1 -> P2*

# Software

[**Latest Arduino software**](http://arduino.cc/en/Main/Software#toc2)

Download and install all libraries into your usual Arduino Library Folder.

The default is usually at /User/Documents/Arduino/Libraries.

# Scripting

Open up FS1012\_flow\_sensor.ino or create a new project. First we need to define a few constant, such as the analog input pin for the flow measurements. (The full source code is included in Section 8).

// Constants

const int analogInPin0 = A0; // analog input pin that the flow sensor is attached to

const int sampleAverage = 20; // number of sensor samples to average

const int serialRateOutput = 100; // sample output rate in ms

const int analogSampleDelay = 2; // sample delay for analog sampling in ms

Then we define a couple of variables, one for receiving flow measurements and one for calculating the average flow rate.

// Variables

int sensorValue = 0; // value read from the flow sensor

float sensorAverage = 0.0; // average sensor value

In the setup() function, we initialize the serial port that we will use to send flow measurement data to the PC. Then we set the internal reference voltage to 1.1V. (The output signal from the flow sensor is in the millivolt range so we want to configure the reference voltage as low as possible. This will result in better measurement resolution).

void setup()

{

// initialize serial communications at 9600 bps:

Serial.begin(9600);

// set internal analog reference to 1.1V

analogReference(INTERNAL);

}

The loop() function is the main part of the program. This function will make flow measurements and output them to the serial port. We start by clearing the average sensor measurement. Then we read n number of flow measurement on analog input channel A0, where n is the number of average samples defined in the constants section. Once the measurements have been completed, we calculate the average and send the average flow measurement to the serial port.

void loop()

{

// initialize average sensor value

sensorAverage = 0.0;

// read and accumulate sensor measurements

for(int i = 0; i < sampleAverage; i++)

{

// read the analog in value:

sensorValue = analogRead(analogInPin0);

// wait for the analog-to-digital converter

// to settle after the last reading:

delay(analogSampleDelay);

// accumulate sensor data

sensorAverage = sensorAverage + sensorValue;

}

// calculate sensor average

sensorAverage = sensorAverage / sampleAverage;

// print the results to the serial monitor:

Serial.print("FS1012 flow sensor = " );

Serial.println(sensorAverage);

// wait to ensure at least 100ms between each serial print

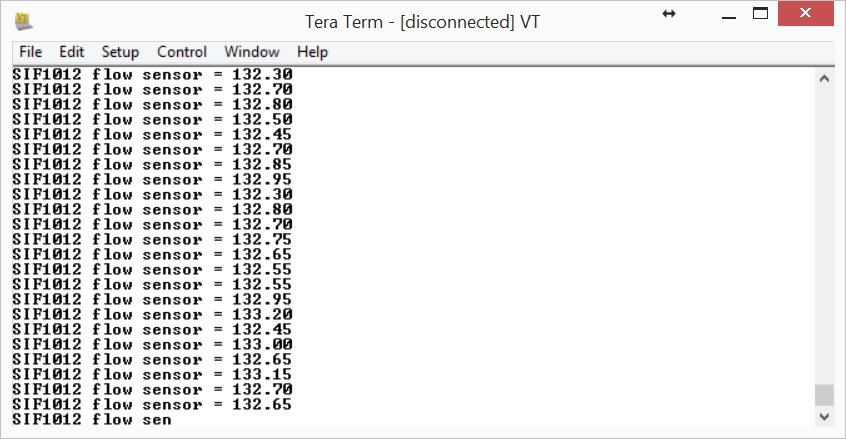
delay(serialRateOutput - (sampleAverage \* analogSampleDelay));

}

We are now ready to run the test setup.

# How to Run the Application

Start your favorite serial terminal program. We are using Tera Term in this example. Make sure the terminal program is connected to the Arduino Uno serial port and the baud rate set to 9600. Once the program runs on the Arduino Uno, you will be able to observe average flow measurements as depicted below (Figure 3). The sample rate is set to 100ms, so you should see about 10 averaged measurements per second.



***Figure 3:*** *Sensor output, using external power supply*

# Arduino Uno Script

/\* Analog input, serial output

Reads an analog input pin, performs a simple averaging routine and prints the result to the serial monitor.

The circuit:

FS1012 Analog Flow Sensor:

- pin 1: connected to analog pin 0.

- pin 2: connected to ground.

- pin 4: connected to +5V.

- pin 6: connected to ground.

created 3 Jan. 2015

by MV

\*/

// Constants

const int analogInPin0 = A0; // analog input pin that the flow sensor is attached to

const int sampleAverage = 20; // number of sensor samples to average

const int serialRateOutput = 100; // sample output rate in ms

const int analogSampleDelay = 2; // sample delay for analog sampling in ms

// Variables

int sensorValue = 0; // value read from the flow sensor

float sensorAverage = 0.0; // average sensor value

void setup()

{

// initialize serial communications at 9600 bps:

Serial.begin(9600);

// set internal analog reference to 1.1V

analogReference(INTERNAL);

}

void loop()

{

// initialize average sensor value

sensorAverage = 0.0;

// read and accumulate sensor measurements

for(int i = 0; i < sampleAverage; i++)

{

// read the analog in value:

sensorValue = analogRead(analogInPin0);

// wait for the analog-to-digital converter

// to settle after the last reading:

delay(analogSampleDelay);

// accumulate sensor data

sensorAverage = sensorAverage + sensorValue;

}

// calculate sensor average

sensorAverage = sensorAverage / sampleAverage;

// print the results to the serial monitor:

Serial.print("FS1012 flow sensor = " );

Serial.println(sensorAverage);

// wait to ensure at least 100ms between each serial print

delay(serialRateOutput - (sampleAverage \* analogSampleDelay));

}

# Disclaimer

Information in this document is provided solely to enable implementation and use of Renesas products. The specifications and characteristics are subject to change without notice. Renesas reserves the right to make changes without further notice to any products herein.

“Typical” parameters provided in this document can and do vary in different applications and actual performance may vary over time. Customers must validate all operating parameters for their application.

Renesas is not responsible for direct, indirect, incidental or consequential damages resulting from any defect, error or failure to perform. In no event, regardless of cause, shall Renesas incur any liability for any damages of any kind, including, but not limited to, any direct, indirect, special, incidental, punitive or consequential damages arising out of, resulting from, or any way connected to the use of the product, whether arising under breach of contract, tort (including negligence), strict liability or otherwise, and whether based on this agreement or otherwise, even if advised of the possibility of such damages, and whether or not injury was sustained by persons or property or otherwise; and whether or not loss was sustained from, or arose out of, the results of, the product, or any services that may be provided by Renesas.

The customer assumes all responsibility and liability for proper and safe handling of the goods. It is the customer’s responsibility to take any and all appropriate precautions with regard to electrostatic discharge. Also be aware that the products herein may not be regulatory compliant or agency certified (FCC, UL, CE, etc.).

No License is granted under any patent right or other intellectual property whatsoever. Renesas assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or any other Intellectual property rights of any kind.

1. Hazardous gases or liquids should not be used with FS1012. This includes, but is not limited to, gases or liquids that are flammable, combustible, toxic or otherwise harmful. Gases and liquids that cannot be clearly identified should never be used with FS1012. [↑](#footnote-ref-1)