Low Cost Spirometer

by MariaL7 on April 18, 2015

Table of Contents

Intro: Low Cost Spirometer

This is a project I completed for a biomedical instrumentation class at Vanderbilt University. This spirometer uses a pressure transducer and an Arduino Uno to compute the volume of air blown through the plastic tube. The circuitry is very basic- the majority of this project relies on the coding and an understanding of fluid dynamics. I'll go through the set up of the physical circuitry and then delve in deeper to the code and mathematics.

Spirometers are typically used in a clinical setting to diagnose pulmonary disease. These tools measure the amount of air expired, and in some cases, can show the change in flow rate over time. This simple, low cost spirometer allows for a rough calculation of the volume of air expired from the lungs over a period of time. A button on the Arduino LCD screen is used to control the period of time over which this calculation takes place. The results are clearly displayed on the LCD screen, making this a fun, easy to use device.

Although this device is not intended to be used in a clinical setting, it was interesting to learn about pressure sensors and Arduino applications while working on this project. I hope you enjoy!

Materials:

-PVC pipe

-Plastic tubing

-Glue

-Pressure sensor (see Step 2)

-Arduino Uno

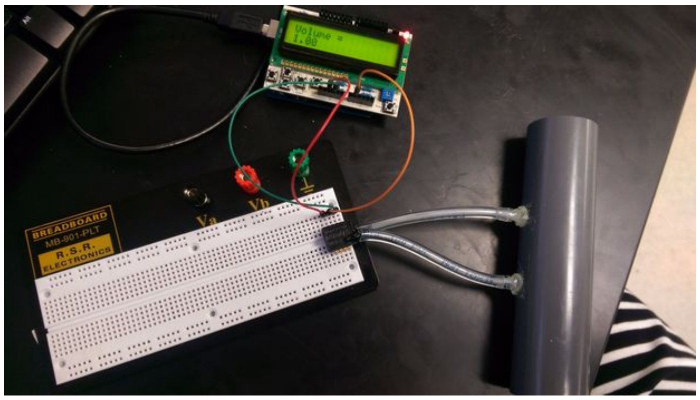
-Arduino compatible LCD screen

-Computer

-Micro USB cable

-Breadboard

-Wires



Step 1: Construct a Spirometer Tube

I was able to salvage my tube from a previous year's project, but I would recommend building your own spirometer tube in order to better customize the diameter size.

The tube should have two sections, one with a large diameter and one with a significantly smaller diameter. In my project I used D1 = 2.3 cm and D2 = 0.6 cm. PVC piping might be a good alternative to using a plastic tube. If I were to do this project again, I would definitely use tubes with a larger diameter difference. I ended up using duct tape in order to create a smaller diameter within the second half of the tube.

On each side of the diameter change, drill a small hole and connect plastic tubing. These two pieces of tubing will later be attached to the pressure sensor in order to calculate the pressure drop across this section of the tube. Make sure the seal is tight by using enough glue.

Step 2: Select a Pressure Sensor

The only circuit component used for this project is a pressure sensor. I used a differential Honeywell ASDX Series Silicon Pressure Sensor. I liked using this sensor and I've attached the data sheet to this page.

Differential pressure sensors are able to convert a difference in two pressures to a voltage. A diaphragm is placed between two compartments of the sensor. Each compartment has a port where a pressure can be applied. When different pressures are applied to the two compartments, the diaphragm moves. The diaphragm is attached to a series of piezo-resistive strain gauges that are connected in a Wheatstone bridge configuration. As the resistances change, the output of the bridge circuit changes and can be used to calculate the pressure difference.

The sensor requires connections to a supply voltage (typically 5 V) and ground. There is also an output voltage pin, which is where the signal we are using will be produced. For this particular sensor, the output voltage when the pressure difference is zero is equal to 1/2 of the supply voltage. I used a 5 V supply, so my output was 2.5 V when there was no pressure differential applied. There is an equation included in the data sheet than can be used to calculate the pressure based on the output voltage.

File Downloads

honeywell-sensing-asdx-series-analog-pressure-sensors-product sheet-008090-12-EN.pdf (571 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'honeywell-sensing-asdx-series-analog-pressure-sensors-product sheet-008090-12-EN.pdf']

Step 3: Attach Arduino Microcontroller

Another major component of this device is an Arduino Uno microcontroller and a compatible LCD screen with buttons. They can be purchased here: http://www.arduino.cc/en/main/arduinoBoardUno. Connecting the Arduino to the pressure sensor is very simple. Since the Arduino is able to supply 5 V and ground, I was able to power the pressure sensor with the Arduino alone.

Before attaching wires from the pressure sensor to the Arduino, attach the LCD screen to the microcontroller. It should easily fit into pins of the Arduino. The pressure sensor can then be connected using the pins on the LCD screen.

My configuration was as follows:

Pressure Sensor Arduino

Pin 1 (supply) 5 V

Pin 2 (output) A1

Pin 3 (ground) GND

My pressure sensor was attached to a basic breadboard and the Arduino microcontroller was attached to a PC via a micro USB cable.

Step 4: Creating the Code

This step was by far the most time intensive portion of this project. Luckily for you, I've done most of the work already! In this step, I will attempt to explain the fluid mechanics and the path I took to get from voltage to air volume using some relevant equations. I've also attached my Arduino code as a word document, so feel free to take a look at that and change parameters as necessary for your design.

1. Converting between digital and analog signals:

When an analog signal (like the voltage produced by our pressure sensor) is processed by the Arduino, it is converted to a digital value between 0 and 1023. Our first step in the code after reading in this value is to convert it back to an analog value. Here is the bit of code doing just that:

```
inputVolt = analogRead(analogInPin); // Voltage read in (0 to 1023)

volt = inputVolt*(vs/1023.0);
```

2. Converting from a voltage to pressure:

The purpose of the pressure transducer is to turn a pressure difference into a voltage, but now we want to determine the pressure difference based on a given voltage. The datasheet provides an equation to do just that, and with some rearranging, the pressure can be calculated as follows:

```
pressure_psi = (15/2)*(volt-2.492669); // Pressure in psi
```

You may notice that a value of 2.492669 is used instead of the 2.5 that I expected the sensor to produce at equilibrium. I determined this more precise value after a number of calibrations showed that my equilibrium value was not at exactly 2.5 V. You may need to adjust this number based on your own sensor's tendencies.

3. Psi to Pa

The equation given in the data sheet gives us the pressure in psi. In order to make further calculations easier, we will convert this to Pascals, which is the SI unit for pressure.

```
pressure_pa = pressure_psi*6894.75729; // Pressure in Pa
```

4. Calculating mass flow from pressure

This next step involves some fluid mechanics knowledge and creative algebra, but ultimately allows you to convert your pressure difference into a mass flow rate. The follow equation can be rearranged to solve for W, the mass flow rate in kg/s:

```
dP=((W^2)/2rho)*(1/A2^2?1/A1^2)
```

Where dP is the change in pressure across the tube in Pa, W is the mass flow rate in kg/s, rho is the density of air in kg/m^3, and A1 and A2 are the cross section areas of the two different sections of your tube in m^2. After rearranging and including values for rho, and the A1 and A2 for my specific tube design, I was able to compute W with the following code:

```
massFlow = 1000*sqrt((abs(pressure_pa)*2*rho)/((1/(pow(area_2,2)))-(1/(pow(area_1,2))))); // Mass flow of air
```

The syntax in Arduino makes this a bit messy, so be sure to check your parentheses. I also included a factor of 1000 in order to have the mass flow be in kg/L, which makes the volume calculations easier.

5. Mass flow to volumetric flow

This step is relatively easy- we can convert mass flow into volumetric flow by dividing by the density.

```
volFlow = massFlow/rho; // Volumetric flow of air
```

6. Computing volume

Finally, we have reached a point where volume can be computed. Since Arduino does not have the capability to perform integrals, we have to manually add up our volumetric flow rate over time. Since volumetric flow rate is simply volume over time, we can sum up the volumetric flow rate over small bits of time to compute total volume. This can be done using a delay in Arduino and multiplying each volumetric flow rate value by a small dt value.

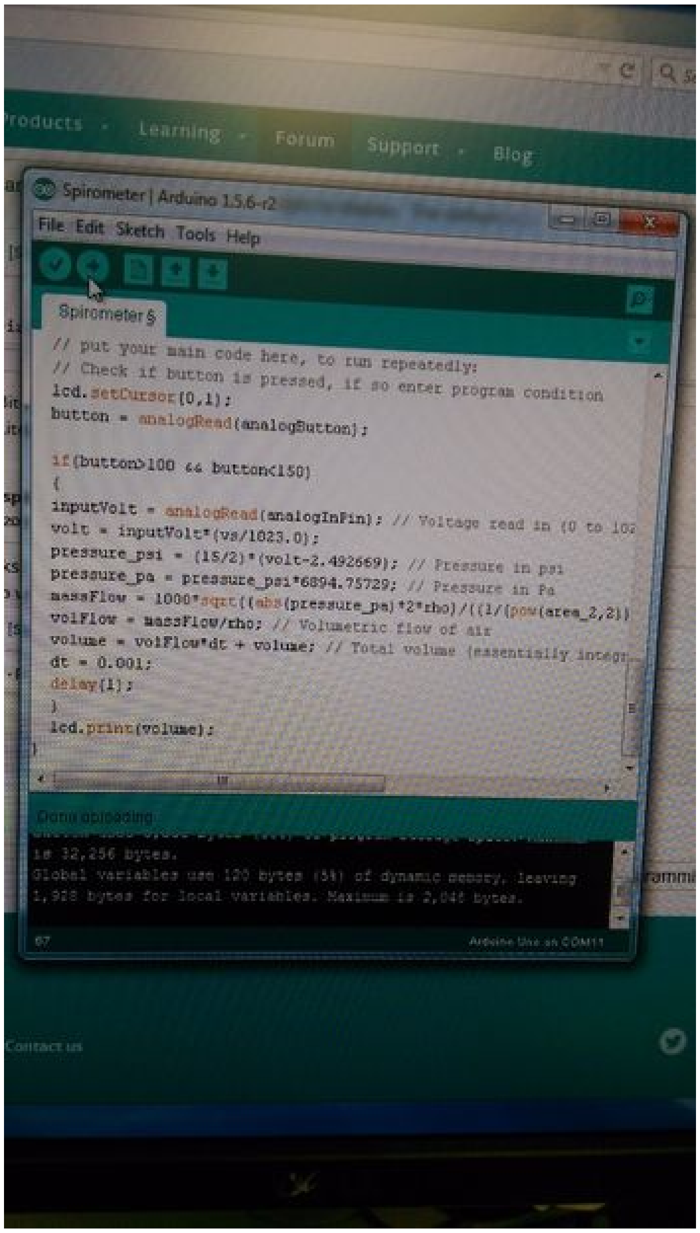
```
volume = volFlow*dt + volume; // Total volume (essentially integrated over time)

dt = 0.001;

delay(1);
```

That's all the math! The rest of the code is just setting up the LCD screen, defining variables, and setting up the button controls.

The if statement in the code causes the volume to be calculated only when the button is being pushed. This feature prevents the calculation from being affected by noise, and also allows the user to indicate when air is actually flowing through the tube.



```
// put your main code here, to run repeatedly:
// Check if button is pressed, if so enter program condition
lcd.setCursor(0,1);
button = analogRead(analogButton);

if(button>100 && button<150)
{
inputVolt = analogRead(analogInPin); // Voltage read in (0 to 102
volt = inputVolt*(vs/1023.0);
pressure_psi = (15/2)*(volt-2.492669); // Pressure in psi
pressure_pa = pressure_psi*6894.75729; // Pressure in Pa
massFlow = 1000*sqrt((abs(pressure_pa)*2*rho)/((1/(pow(area_2,2))
volFlow = massFlow/rho; // Volumetric flow of air
volume = volFlow*dt + volume; // Total volume (essentially integr
dt = 0.001;
delay(1);
}
lcd.print(volume);
}
```

Done uploading.

```
is 32,256 bytes.
Global variables use 120 bytes (5%) of dynamic memory, leaving
1,928 bytes for local variables. Maximum is 2,048 bytes.
```

67 Arduino Uno on COM11

File Downloads

spirometer code.docx (22 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'spirometer code.docx']

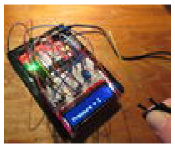
Step 5: Try Out Your Spirometer!

Finally, you can try out your spirometer! In order to use the device, attach the tube created in Step 1 to the pressure sensor using the two pieces of tubing. Bring the tube up to your mouth and blow through it, while simultaneously pressing the up button on the LCD screen. Once you have finished expiring, release the button. The volume you expired will be neatly displayed on the LCD screen.

I hope you enjoyed this instructable! Thanks for reading!

Related Instructables

Hack an ELM327 Cable to make an Arduino OBD2 Scanner by mviljoen2


Arduino pressure sensor (FSR) with LCD display by NickW8

Making Data Loger for Room conditions Record Using Arduino by gigin

The Arduino Weather Station / Thermostat by sspence

Connect A 16x2 LCD Display To An Arduino by shanecoley

Calculando cuantos dias has vivido con Arduino by ElectroCrea

Comments

1 comments **Add Comment**

acheide says: Apr 18, 2015. 6:21 PM **REPLY**

I remember having my lung capacity tested. Thanks for showing how it is done.