Blood Pressure Monitor

by LunesCuatro on April 20, 2015

Table of Contents

Intro: Blood Pressure Monitor

For our Biomedical Instrumentation course, we created a blood pressure monitor. This blood pressure monitor measures the mean arterial pressure (MAP) and approximates the systolic and diastolic pressures. It requires the use of a pressure transducer, an Arduino Uno, and coding to control the valve and air pump. The circuit design is composed of three basic stages: a low pass filter, a high pass filter, and a noninverting amplifier. In this Instructable, we will explain the set up of the circuity and then explore the coding behind the monitor.

Blood pressure monitors are primarily used in clinical settings to analyze patient's blood pressure and in order to prescribe the best treatment. This low cost monitor inflates and deflates the blood pressure cuff in order to determine the mean arterial pressure. It then roughly approximates the diastolic and systolic pressures based on the mathematically relationship between mean arterial pressure, diastolic, and systolic pressures. The results are then displayed on the LCD screen.

It should be noted that this device is not designed for clinical applications, however, it is a fun way to learn about pressure sensors and Arduino while creating a functional device applicable to the real world!

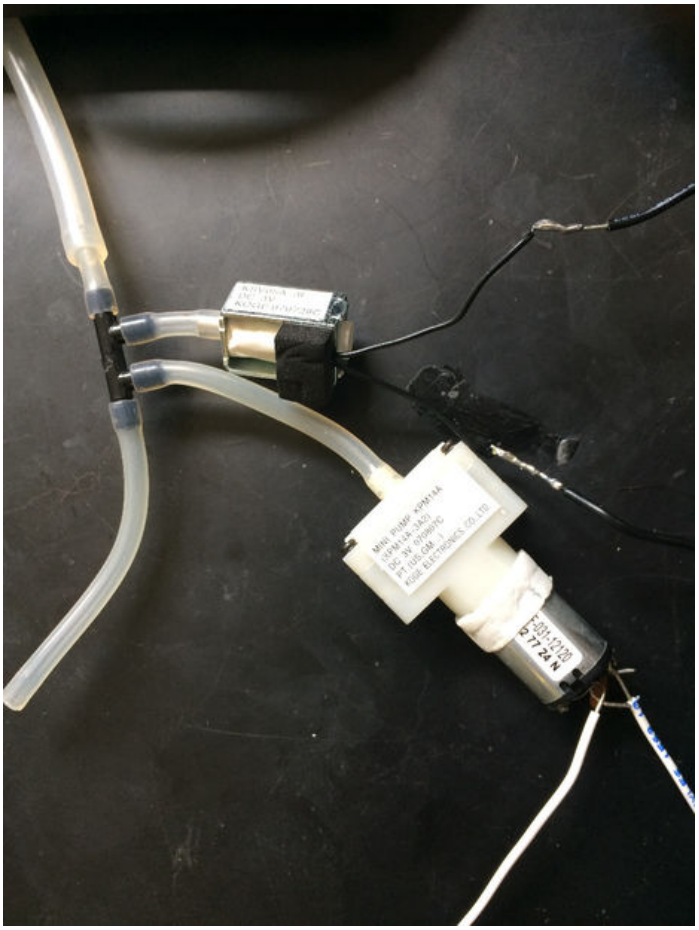
Step 1: Materials

- Arduino UNO board w/USB cable
- Arduino compatible LCD
- Pressure Transducer (We used the Honeywell Differential Transducer 015PDAA5)
- Voltage-controlled valve
- Air pump
- Wires
- Power Supply (±15V)
- Resistors (Four 100k?, One 1 k?)
- Capacitors (Two 1µF)
- Breadboard
- Three-way splitter
- Plastic Tubing
- Blood Pressure Cuff
- TL072 Op Amp

Our design consists of a passive bandpass filter composed of a low pass filter in series with a high pass filter. The low pass filter eliminates any high frequency noise (cutoff = 10Hz) and the high pass filter (cutoff = 2Hz) allows us to determine the MAP based pressure fluctuations in the cuff.

Step 2: Set up the Cuff, Pump and Valve Configuation

Use the three-way splitter and plastic tubing to join the cuff, air pump, and valve. You will want tubing that fits snugly around the pump and valve so that no air leaks out, or else your measurements will be less accurate. Cut a fourth piece of tubing and connect it to the splitter. The pressure transducer will be connected to this tubing later.



Step 3: Add the Transudcer to the Breadboard

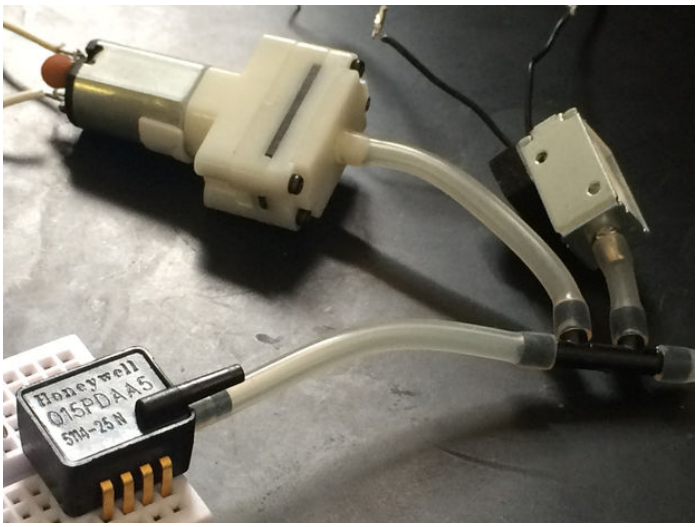
Place the transducer on the breadboard and attach the ground and power supply according to the data sheet:

Pin1= supply

Pin2= output voltage

Pin3= ground

For this device, 5 V was applied to pin 1, the output voltage was connected to the circuitry (built in the next step), and lastly, pin 3 was connected to the ground from the Arduino Uno.



Step 4: Build the Low Pass and High Filter and Amplifier

On a breadboard you will create the follow circuitry as illustrated in the diagram below. Corresponding to the diagram labels:

Vaa= voltage source

R1=100k?

R2=200k? (Two 100k? in series)

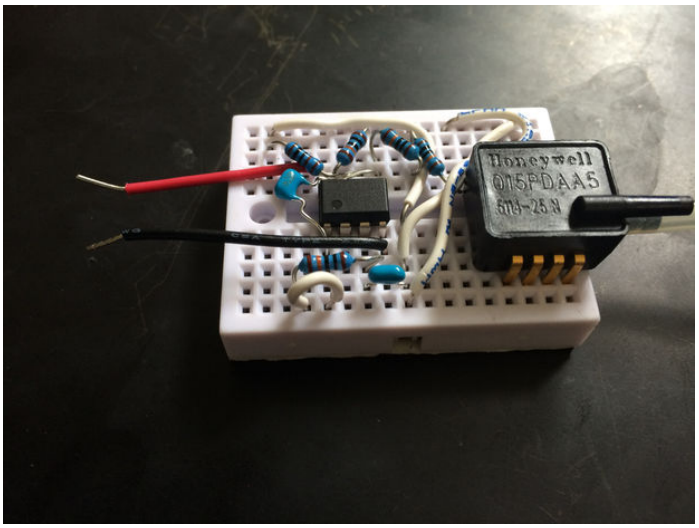
R3=100k?

R4=1k?

C1=1µF

C2=1µF

Don't forget to orient the transducer and TL072 so that the 1 IN terminal receives the transducer output. Also, we used color-coded wires to distinguish between the negative (black) and positive(red) supply terminals of the op amp terminals.



Step 5: Code

The first picture shows the code used to initialize the LCD library ("LiquidCrystal"), the LCD screen dimensions, and all necessary global variables. The screen dimensions are initialized in the setup and specify the number of columns and rows on the screen, 16 and 2 for our code. The valve doesn't release air when a voltage is applied across it, so Pin 3 (to which the valve is connected) is initialized in Output Mode and HIGH.

The second picture contains the code that actually calculates the pressures. We created an array of the output voltages using a for loop. The loop is set to run 50 times with a delay of .25s. This number of values corresponds to about 12.5s. Cuff inflation after this amount of time was too tight for our "patient." You can adjust these values as you see fit. Since Arduino is a 10bit system, the analogRead function returns an integer within the rang [0,1023]. **volt** is calculated by converting this integer value into its corresponding voltage. The for loop also stores the maximum voltage as data is collected by the Arduino. We subtracted 2.5V from this **volt** due to the 2.5V offset that the transducer has when both ports are exposed to atmospheric pressure.

The applied pressure (**pressure**) is calculated using the equation on the transducer data sheet. However, we used a differential transducer which mean the **pressure** we calculated is actually the difference between Ports 1 and 2. **MAP** is the pressure at Port 2 and is calculated by subtracting the **pressure** from atmospheric pressure which is 14.7 psi. This value is multiplied by 51.7 to give **MAP** in units of mmHg. There is an additional term in the equation for **MAP**. After taking several measurements, we noticed a pressure offset that decreased as the voltage increased. We compensated for this by subtracting " 3.16/maxvolt " from the pressure. We obtained this value by averaging the pressure offset and relating it to the measured voltage. Once the for loop is exited and **MAP** calculated, Pin 3 is written to LOW and the valve releases the air from the cuff.

The final piece of code is an empty while loop. This was added so that the Arduino didn't continuously calculate **MAP**.

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(8,9,4,5,6,7);
float PressureMin =-15; //psi
float PressureMax= 15; //psi
float Vsupply=5;//voltage supply

int analogInPin = A0; // Analog Input
float volta = 0;
int i;
float maxvolt = 0;
float volt=0;
float pressure = 0;
float MAP = 0;
float maxv = 0;

void setup() {
  lcd.begin(16, 2);
  pinMode(3,OUTPUT);
 }
```

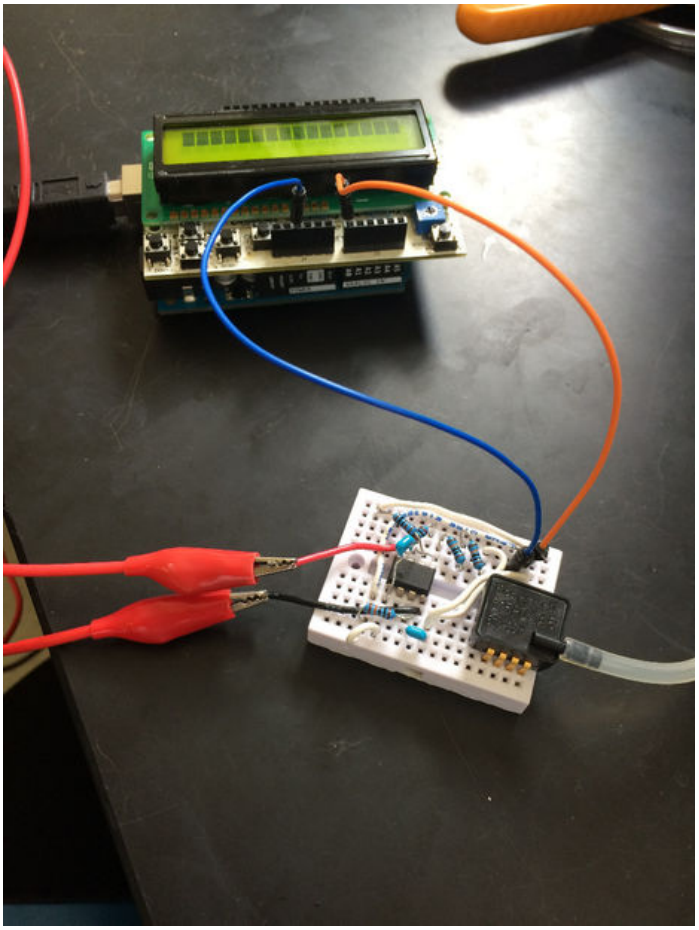
```
void loop() {
  digitalWrite(3,HIGH);

  for (i = 0; i <40; i = i +1){
 volta = analogRead(analogInPin);
 volt = (volta*Vsupply)/(pow(2,10)-1);
 maxv = max(abs(volt-2.5), maxvolt);
 maxvolt = abs(maxv-2.5);
 delay(250);
  }

 pressure = (((maxvolt)-.1*Vsupply)/((.8*Vsupply)/(PressureMax-PressureMin)))+PressureMin; //psi
 MAP= -1*(14.7-pressure*-1)*51.7 - 3.16/maxvolt; //mmHg
 digitalWrite(3,LOW);

 lcd.setCursor(0, 0);
 lcd.print(" MAP = ");
 lcd.print(MAP);
 lcd.setCursor(0, 1);
 Serial.print(" ");
 Serial.print( MAP*1.1);
 lcd.print(" / ");
 lcd.print(MAP*0.8);

 while(1){}

}
```

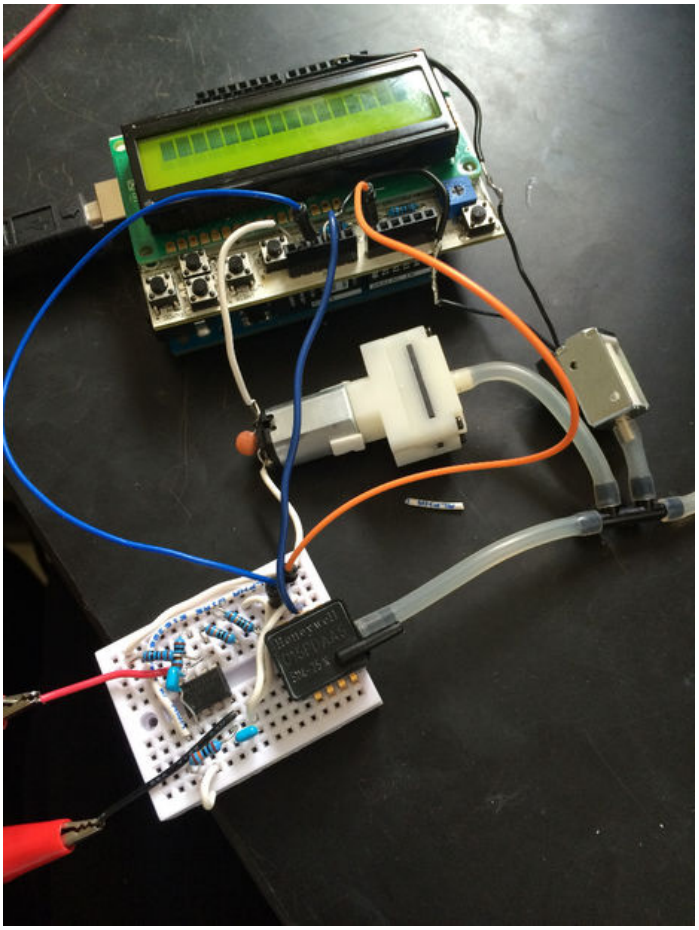
Step 6: Connect All Circuit Components

Connect the following components:

1. LCD to Arduino (the LCD pins should fit into the Arduino pins)
2. Power Supply Terminals to TL072
3. Arduino Ground and Power Supply Ground (All supply grounds must be connected)
4. The 2 OUT terminal of the TL072 should be connected to the A0 Pin.

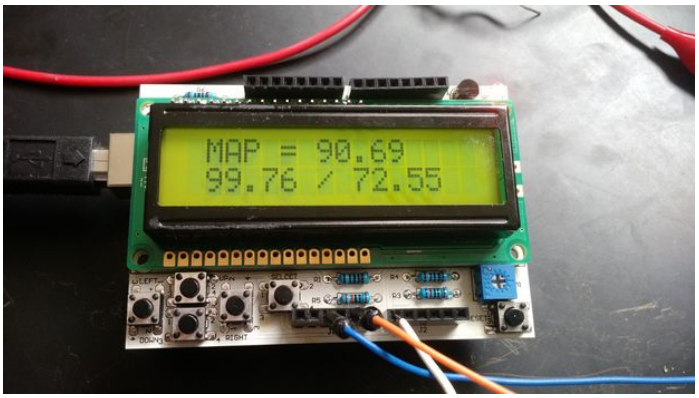
The valve wires should be connected to Digital Pin 3 and ground. For our purposes, the orientation of the wires did not matter. One of the air pump wires will be connected to ground. When the program is run, the other air pump will need to be manually inserted into the 3.3V power supply on the Arduino. If you discover a way an automatic way to control the air pump, please respond in the comments.





Step 7: Take Your Measurements!

Connect your air pump to the 3.3V Pin on the Arduino as soon as the code has been uploaded. Try to sit as still and quiet as possible while the code runs. Once the MAP is displayed you can disconnect the air pump and remove the cuff from your arm. The valve will release any air remaining in the pump. You should see something similar to the above picture.



Related Instructables

LED Pulse Sensor (PPG) for Arduino by Marc_Escobaem

How to Manually Take Blood Pressure by jennifer.teague.90

Pulse sensing faceted heart lamp by Raitis

nitro coffee by mikeasaurus

How to use Arduino Mega 2560 as Arduino isp by tsillen

How doctors treat a hypoplastic heart by supersoftdrink

Comments