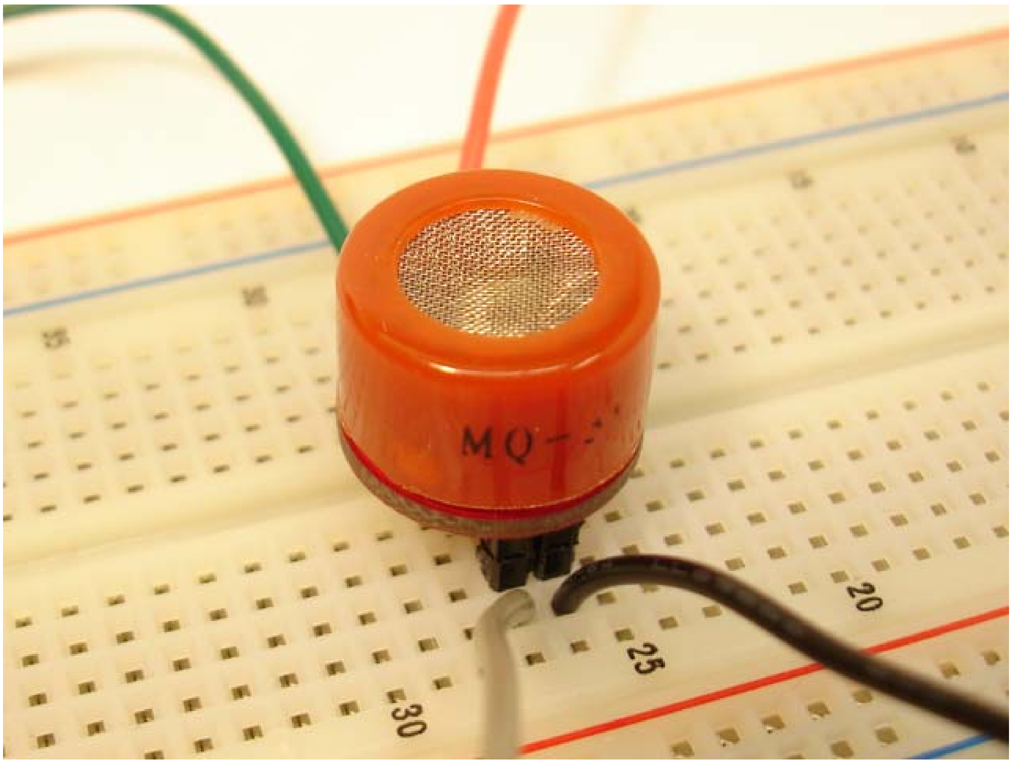
project lab
nootropic design

Arduino Breathalyzer: Calibrating the MQ-3 Alcohol Sensor

***Difficulty Level* = 3** [What's this?]

The MQ-3 is an alcohol gas sensor that is available for about $5 from Sparkfun, Seeed Studio, and others. It's easy to use and has sparked the imagination of anyone who has dreamed of building their own breathalyzer device for measuring the amount of alcohol in the human body. I got an MQ-3 sensor a couple of months ago and have spent a lot of time trying to figure out how to do this. After lots of "data gathering", I found that this task is not as easy as it sounds.



MQ-3 Alcohol Sensor

First of all, don't try to use the MQ-3 as a way to determine if you are sober enough to drive. If you've been drinking, just don't drive! And if you've had a few drinks, don't do any soldering either.

The Circuit

The MQ-3 has a 6 contacts which is confusing, but not as confusing as the MQ-3 datasheet. I also got the Sparkfun breakout PCB for the MQ-3 so it makes the pinout easier to understand. Basically, there is a heater which you power by connecting to VCC and GND. The heater draws about 150mA of current, so I would not power the heater directly from an Arduino board (and *certainly* not from an output pin). The Arduino voltage regulator can handle 800mA, but you need to consider the power dissipation capability of that voltage regulator, too. I think it's best to power the heater from an external source, like a regulated 5V power supply or a 7805 voltage regulator.

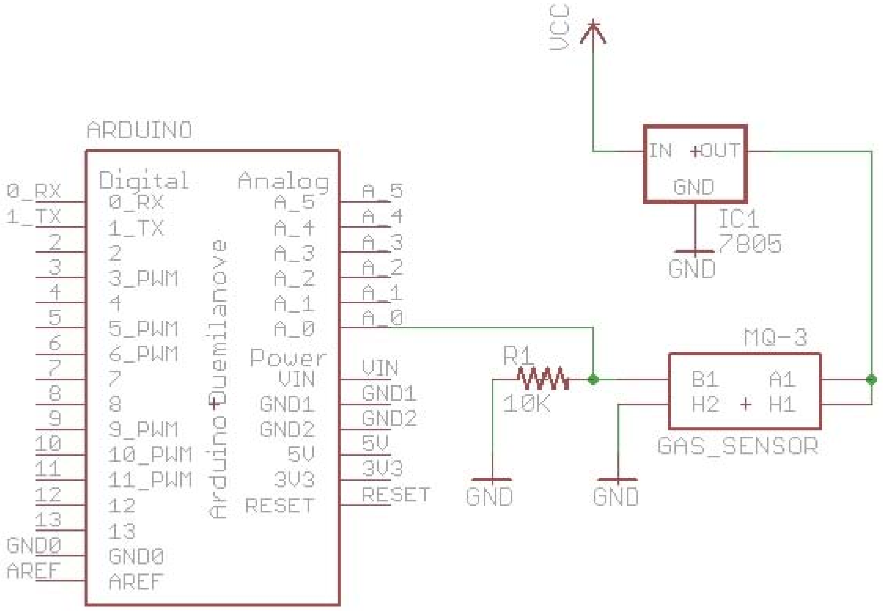
The other part of the circuit is essentially a variable resistor inside the sensor. The resistance across an A pin and a B pin varies depending on how much alcohol is in the air in the sensor. The more alcohol, the lower the resistance. We measure alcohol in the breath by measuring this resistance. Instead of measuring the resistance directly, we measure the voltage level at the point between the sensor and a load resistor. The sensor and load resistor form a voltage divider, and the lower the sensor resistance, the higher the voltage reading will be. An Arduino analog input pin is a good way to measure the voltage and gives us a reading between 0 and 1023. Higher values mean more alcohol.

One more caveat that every MQ-3 owner needs to know: the sensor has a 24-48 hour "break in" period. That is, the sensor needs to be turned on for 24-48 hours before the readings become stable.

Archives

- May 2016
- September 2015
- July 2015
- February 2015
- December 2014
- November 2014
- October 2014
- October 2013
- July 2013
- July 2012
- June 2012
- February 2012
- January 2012
- December 2011

Here’s a circuit that shows how I connected the MQ-3. I used a 10K ohm load resistor in series with the sensor resistor. The MQ-3 datasheet says to use a 200K ohm resistor, but anyone who’s used this sensor knows that this constrains the measured voltage to a tight, high range. I recommend a 10K ohm resistor. Make sure you connect all the grounds together.



Arduino Breathalyzer Schematic

It’s easy to read the analog value from the analog input pin with a very simple Arduino program. I chose to run the Standard Firmata firmware on the Arduino and run a Processing program on my computer to display the values. More on that later.

And Now, Some Math

The good news is that if you know how much alcohol is in your breath, there is a straightforward formula used to calculate your blood alcohol content (BAC). Alcohol content in a volume of breath or blood is expressed as mg/L (milligrams per liter). A 1% blood alcohol content is 10g/L or 10,000mg/L. Since this level would almost certainly be fatal, let’s scale this down from “dead” to “drunk”. A 0.1% BAC is 1000mg/L.

How does this relate to the breath? It turns out that there is a standard conversion from breath alcohol content to BAC that is employed by commercial breathalyzers. Breath and blood alcohol content differ by a factor of 2100; that is, for every mg of alcohol in the breath, there are 2100mg of alcohol in the blood. So, a person with BAC of 0.1% has 1000mg/L of alcohol in their blood and 1000/2100 = 0.4762mg alcohol in their breath. So, our final formula for calculating BAC from the alcohol measured in the breath is:

% BAC = breath mg/L * 0.21

Sensor Calibration Challenges

So, I simply need to use the sensor to determine how many mg/L of alcohol there is in my breath, right? The bad news is that it is very difficult to know how much alcohol is in your breath in terms of mg/L! The poorly written MQ-3 datasheet says to calibrate the sensor by exposing it to a known alcohol gas of 0.4mg/L. From there, you can determine gas alcohol content in terms of mg/L.

But how do I create an air environment with 0.4mg/L alcohol? I could evaporate 0.4mg of ethanol (that’s the type of alcohol we’re talking about here) in a liter of air, but how do I measure out exactly 0.4mg of pure ethanol? I don’t have a source of pure ethanol. I could buy 151 proof Everclear grain alcohol but then would need to account for the different density when measuring this tiny quantity. And there’s the additional complexity in that evaporating aqueous ethanol will alter the humidity, and the sensor readings are sensitive to humidity! This is getting complicated. If were really motivated, I would drive to Wisconsin where I could buy 190 proof (95%) Everclear. Last time I drove to Wisconsin was to buy illegal fireworks, and it seems that my only reason for visiting Wisconsin is to purchase contraband and smuggle it back into Minnesota. But I digress…the bottom line is that calibration according to the MQ-3 datasheet is hard.

Another calibration approach would be to use my breath as a “known” concentration of ethanol after drinking a known quantity of alcoholic beverages. I could drink the right amount of alcohol to give a 0.4mg/L concentraion in my breath and measure the resistance to get my calibration baseline. But this presents problems, too! The

- November 2011
- September 2011
- July 2011
- May 2011
- March 2011
- February 2011
- January 2011
- December 2010
- October 2010
- September 2010
- July 2010
- April 2010
- March 2010
- January 2010
- November 2009
- October 2009

Categories

- Android
- Arduino
- Art
- Audio
- Automotive
- AVR
- Difficulty Levels
    - Level 1
    - Level 10
    - Level 2
    - Level 3
    - Level 4
    - Level 5
    - Level 6
    - Level 7
    - Level 8
    - Level 9
- GPS
- Hackvision
- IOIO
- Processing
- RF Wireless
- Robotics
- Uncategorized
- Video
- XBee

Meta

- Site Feed
- Comments Feed

sensor calibration depends on 0.4mg/L concentration in “clean air”. What is “clean air”? I think it means air that is devoid of other contaminating gases that affect the sensor, and assumes 21% oxygen concentration and 65% humidity at room temperature. Those conditions are not going to be true in my breath!

My head is spinning, and I haven’t even started drinking yet. In short, I don’t know how to create the right conditions for accurate calibration without a laboratory. Any ideas welcome.

Forget Calibration — Use Correlation

Ultimately, I decided to simply correlate sensor readings with “computed” BAC based on body weight, amount of alcohol consumed, and elapsed time. There are plenty of online charts and calculators that tell you your approximate BAC based on these inputs. *Unfortunately, they don’t give the same values for the same input!* I used the first link below from Viginia Tech because it was the source of the values cited in the Wikipedia BAC page and everything in Wikipedia is correct, right?
http://www.alcohol.vt.edu/Students/alcoholEffects/estimatingBAC/index.htm
http://www.dot.wisconsin.gov/safety/motorist/drunkdriving/calculator.htm
http://www.bloodalcoholcalculator.org/
http://www.healthstatus.com/cgi-bin/calc/calculator.cgi
http://www.intox.com/wheel/drinkwheel.asp
http://www.1800duilaws.com/dui-resources/BAC-calculator.asp
http://health.discovery.com/tools/calculators/alcohol/alcohol.html

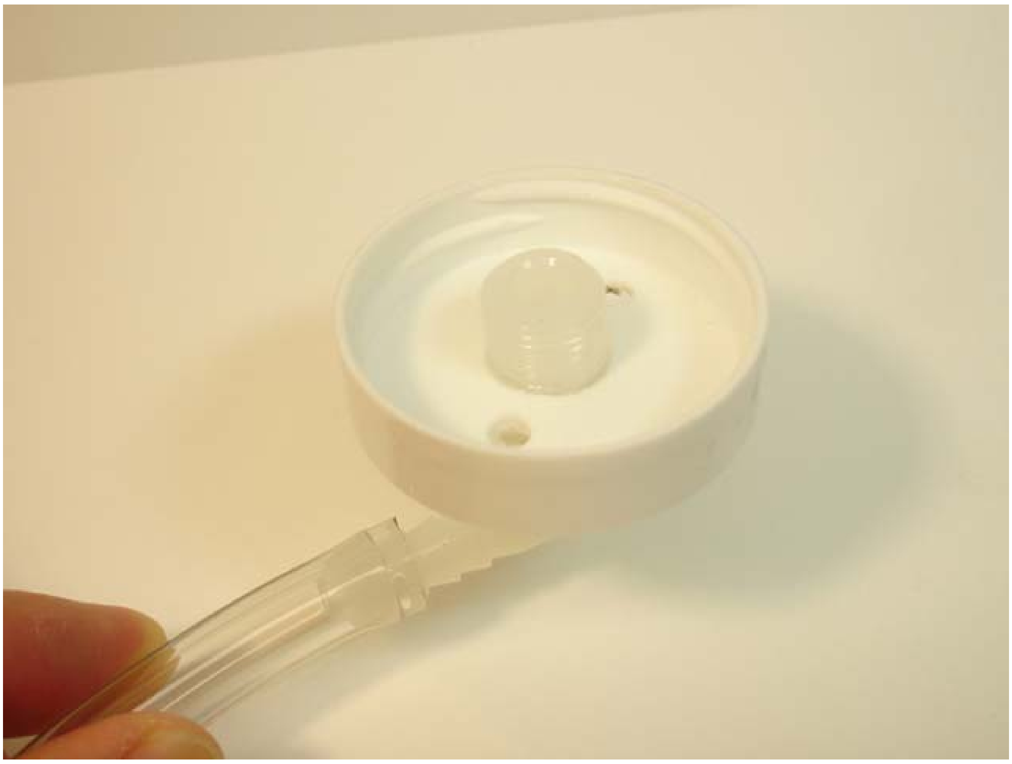
My experimental approach is simply to drink alcohol, compute my BAC according to a chart, and correlate that value to the voltage reading from the Arduino analong input pin. If the sensor is accurate, and if I gather enough data points, I should be able to correlate measured voltage to BAC. This will involve a lot of drinking, night after night. I’m doing this for science. Science!

Sensor Hardware

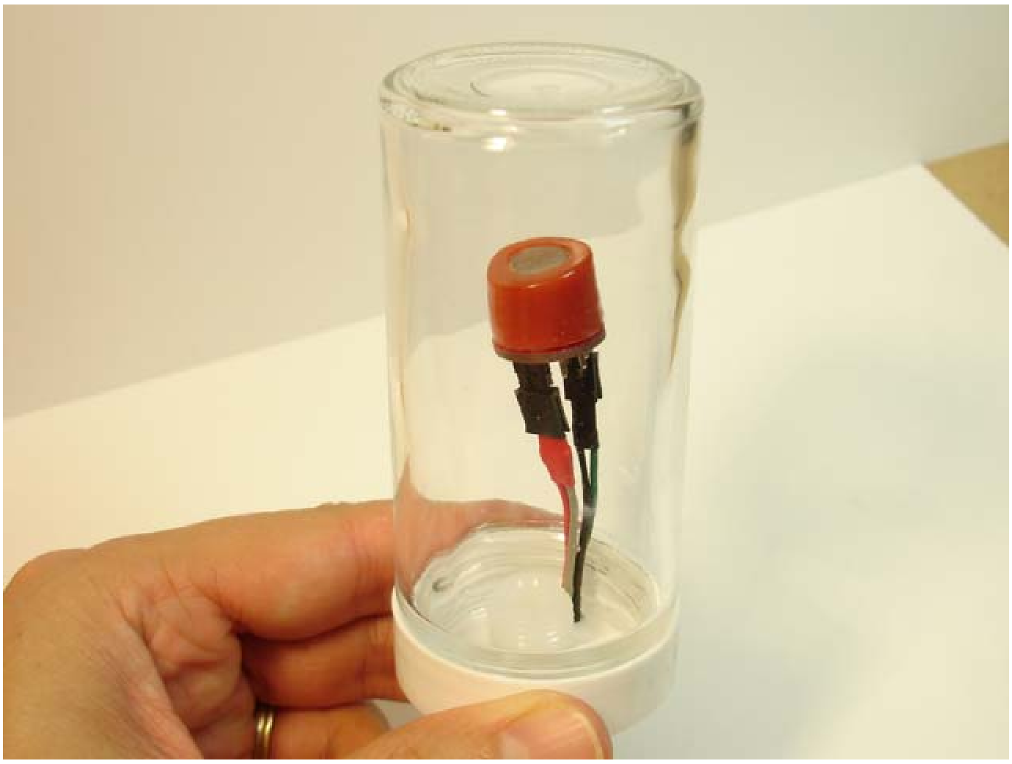
In order to ensure that I exposed the sensor to a uniform breath sample, I decided to enclose the sensor in a glass chamber. I used a simple glass jar with a plastic lid (it used to hold spices).

Small jar to serve as breath chamber

I drilled some holes in the lid and used some vinyl tubing as a tube to blow into. One small hole is for the sensor wires, and the other small hole is for breath to escape as I blow more breath into the jar.



Plastic cap for base of breathalyzer chamber



MQ-3 sensor installed in breathalyzer chamber

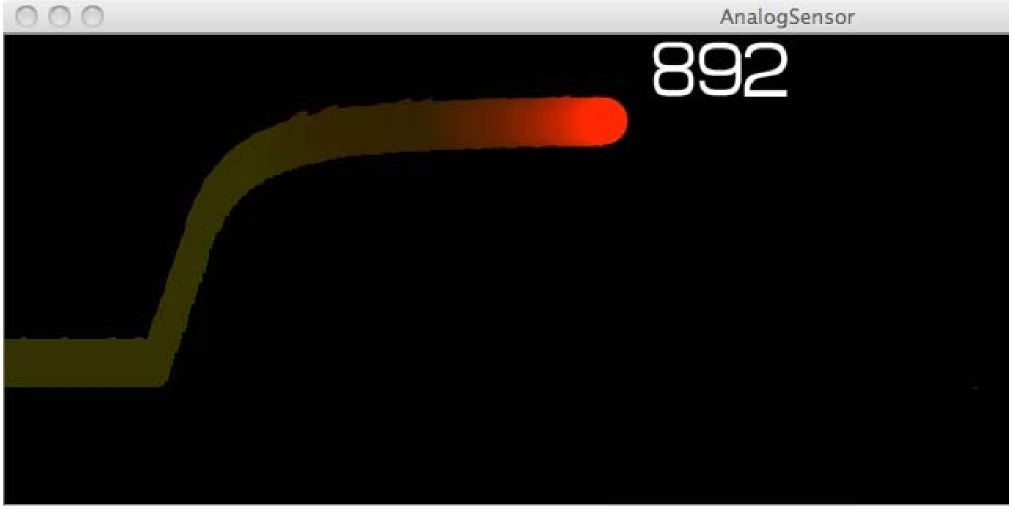
In the blow tube, I added a tee joint so that condensation from my breath could drain out. I know that’s gross, but the condensation builds up. I plugged the tee joint drain when I was blowing a breath sample.

Tee joint for drainage (yuck)

Data Collection (Drinking)

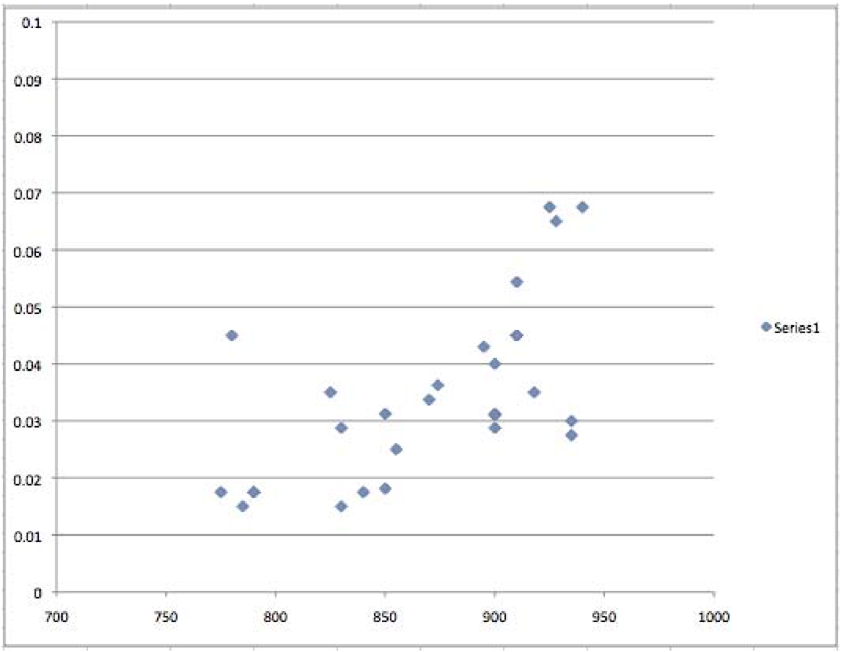
One thing I have learned is that I can't drink nearly as much as I could in college (20 years ago). Your BAC drops 0.01% for every 40 minutes of drinking, so I had to drink faster than I wanted to in order to get my BAC up. I also learned that after about 4 drinks, I kept forgetting to write down the sensor readings.

Here's a screenshot of the Processing program reading the value from the Arduino over USB. Download the source code here. As you can see, the baseline analog reading is about 260 on a scale of 0-1023. When I blow into the sensor chamber, the resistance of the sensor drops dramatically, causing output voltage of the voltage divider to be higher.



Sensor reading display

Here's a scatterplot of the sensor readings (x axis) plotted against the "computed" BAC values (y axis):



Excel spreadsheet scatterplot of BAC vs. analog sensor reading

This doesn’t exactly form a nice straight line, does it? If I look at the data points for a particular drinking session, sometimes there was a nice straight line of points. But the sensor behaved differently on different nights, and I drank different types of alcohol, too, even though I tried to stick to the standard definitions of “serving of alcohol”. Overall, I don’t think I was very successful finding an accurate correlation between sensor reading and BAC. Rough approximation, but not an accurate reading.

Conclusions

It’s really hard to calibrate this sensor for even an approximate BAC reading. It’s even difficult to correlate readings to looked-up BAC values. There are many environmental factors that affect the resistance within the sensor (humidity, temperature, oxygen concentration), and this is only a $5 device anyway. And as evidenced by the lack of consistency between online BAC calculators, there’s not even concensus about how to compute BAC. Law enforcement agencies have much more sophisticated breathalyzers, and often rely on actual blood tests or urinalysis for evidence.

Despite these disappointments, the MQ-3 is still fun to play with, and maybe it’s good enough to just make some LEDs light up when you blow on the sensor. So just enjoy your beverages and have fun. Cheers!

Tweet

Published by Michael, on September 17th, 2010 at 6:43 pm. Filed under: Arduino,Level 3,Processing. | 60 Comments |

---

60 Responses to “Arduino Breathalyzer: Calibrating the MQ-3 Alcohol Sensor”

1. While this sensor doesn’t seem very useful for discovering a fixed BAC because of its calibration issues, couldn’t it still be useful for relative BAC? If two people blow into it one right after the other, environmental factors would seem to then be controlled. So do you think the sensor would be good for relative intoxication between two people? Or even relative intoxication per drink, say every 10 minutes?

   Comment by nebulous on September 23, 2010 at 9:50 AM

2. Yes, nebulous, you can still get a sense for the relative amount of alcohol in your breath. The sensor is well-behaved in that regard. One problem, though, is that once you blow on the sensor and get a reading, it takes several minutes for the resistance of the sensor to rise again (causing the measured voltage to drop back down). This takes about 5 minutes. So, you always have to wait a while between readings.