Hacking a Bathroom Scale to Work with an Arduino

Glen Meyerowitz

I. Things you will need:

- Walgreens Digital Glass Scale
- Arduino
- Multimeter
- Soldering iron and solder
- Headers
- 28 gauge wire
- NPN 2N3904 transistor
- Two 200Ω resistors
- Printed circuit board
- Drill with small bit

II. Safety

This project is very safe to undertake, but please keep in mind that there are a few areas of potential danger. We are working with electronics, and there will be current flowing through the scale when you have it open. This current is very small, but still exists, so you should be aware of what you are working with! Additionally, this project will require you to use a soldering iron, and that can also be dangerous. If you have never used a soldering iron, please get help from a friend with experience! Whenever you solder, you should take the batteries out of the scale! This project should be able to be completed safely and easily. If you chose to follow these directions, you are doing so at your own risk.

III. Some theory

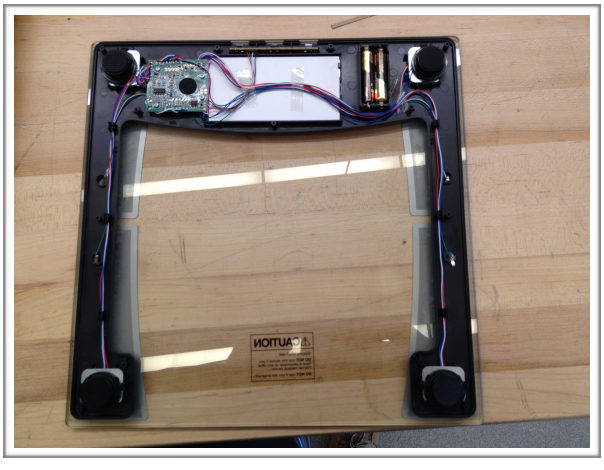
A standard digital bathroom scales uses strain gauges placed at various locations under the scale to measure the force applied to the scale. A strain gauge can be thought of as a piece of wire that is mechanically attached to a piece of metal, like a beam. When a force is applied to the beam it will bend a certain amount. The wire bends as the beam bends. When the wire bends, the resistance of the wire changes. If a current is sourced through the wire, and voltage is measured, this will result in a change in voltage that is proportional to the change in resistance. The change in resistance is related to the force applied, thus the weight of the object on the scale can be determined by measuring the change in voltage across the strain gauge.

IV. How to wire the scale

This project is going to work with the Walgreens Digital Glass Scale. I chose this scale simply because it was the cheapest digital scale I could find and there is nothing special about it. The following details in this document cannot be used for other digital scales, but the ideas presented here are certainly applicable to other types of scales.

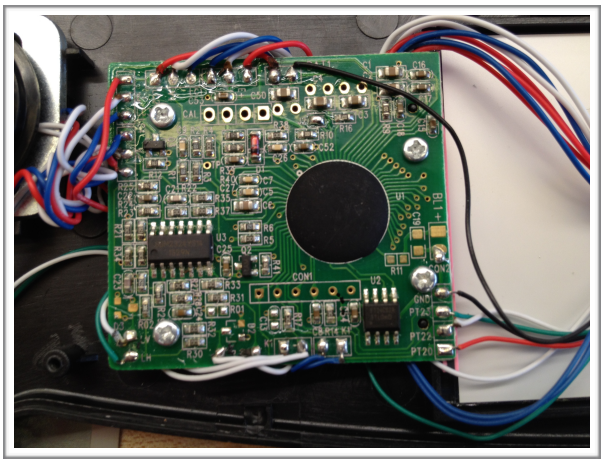
This scale is priced between $20-$25 depending on where you buy it, and it has all the functionality that we need in order to connect it to an Arduino and read the output signal. Before we get started dissecting the scale, I recommend you put in the batteries and take a few minutes to turn it on and play around with it. There are several things you will notice. First, the scale does not turn on until weight is applied to the top of the scale. Additionally, the scale turns itself off after not being used for 10 seconds. The weight the scale reads also varies depending on where you are standing on the scale, and may also vary (often by more than a pound!) if you simply get off and get back on. While this inaccuracy is certainly not desirable, it is hard to avoid when dealing with cheap scales such as this.

After you have spent a few minutes familiarizing yourself with the scale, it is time to start having fun! The first step is to remove the back of the scale by unscrewing the various screws that hold the back plastic piece on. Once you remove that piece of plastic, you will see something like the image below.

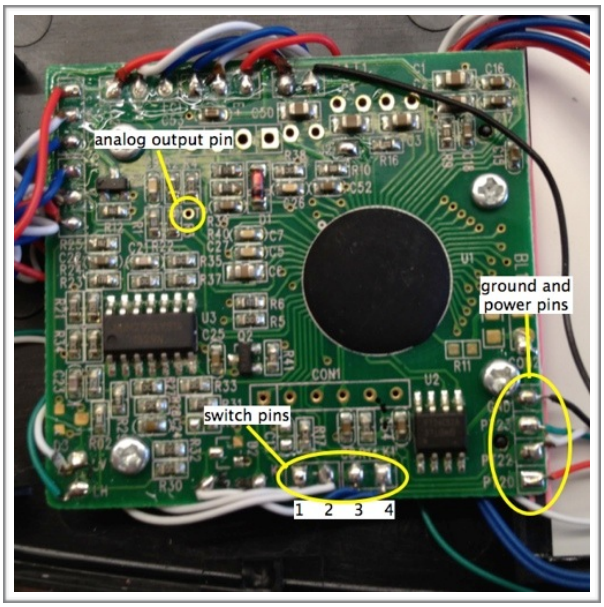


In the four corners, you can see that there are four legs that allow the scale to make contact with the ground. There are also four strain gauges each attached to a piece of metal. In addition, there is a circuit board in the upper left, and a screen in the upper middle. The top two legs on the scale also serve as on/off switches, while the bottom two are simply points of contact with the ground.

The most important part of this scale that we will be dealing with is the circuit board. A picture of the circuit board is shown below. Here you can see a few key elements. There are two integrated circuit chips, one microcontroller, and various wires coming into the board.



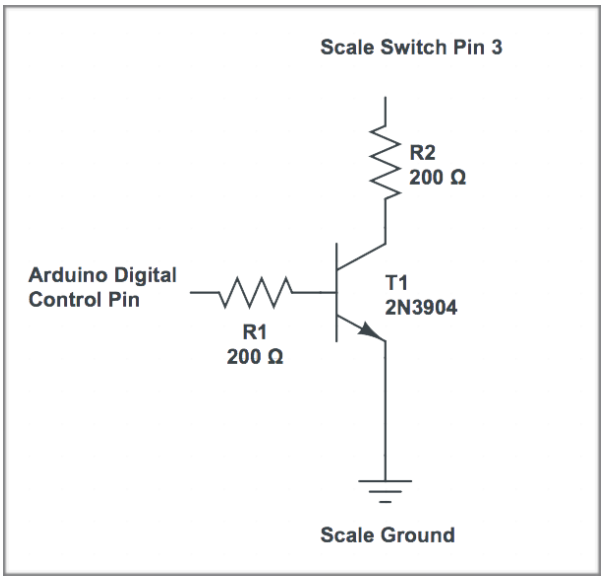
At this point you should break out your multimeter and start playing around. Have the multimeter set to measure voltage and then you can start probing various parts of the board and see what happens when you do different things. The scale runs on two AAA batteries at around 3.1 volts (V). You can try turning the scale on and off, pressing on the various strain gauges and legs, pushing the buttons on the top of the scale, or doing whatever else you want to try. After spending some time on this, you will find that there are a few places on the board more interesting than the rest, and I have highlighted these places in the image below.



The most important part is close to the middle of the board, labeled 'analog output pin,' above. The signal coming out of this pin is the amplified signal from the four strain gauges and this is exactly what we need to feed into the Arduino. When the scale is off, you will get small voltages coming from this pin that do not mean very much (about 0.3V) . However, when the scale is on, the voltage will increase to about 1.7V with no weight applied to the scale. As the weight increases, the voltage also increases. We are going to solder one wire from this pin and read it into the Arduino analog pin.

The second point of interest is labeled 'switch pins.' There are four wires here, two white and two blue, labeled 1-4 from left to right. The two white wires (1 and 2) come from the upper left switch and the two blue wires (3 and 4) from the upper right. By probing these contacts with a multimeter, you will find that the voltage difference between the two blue wires is 3.1V (the same voltage as across the batteries), and the voltage difference between the two white wires is also 3.1V. When you press either of the switches, however, the voltage drop between the two wires will decrease to 0 volts. This means that when both of the wires are at ground voltage, the scale thinks someone is standing on it and that it should read the weight. We can take advantage of this with an Arduino and a simple electronic circuit.

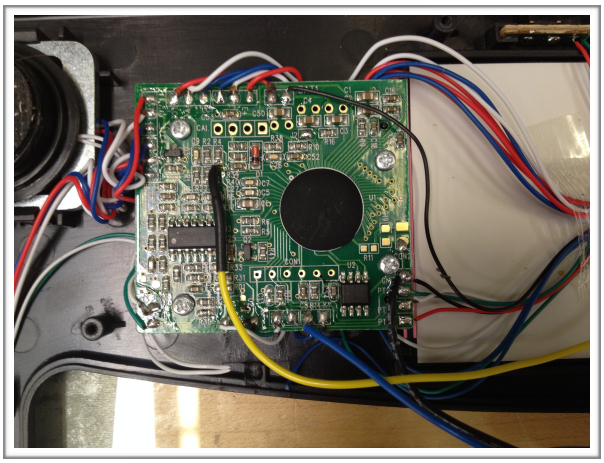
The Arduino has many digital pins, in addition to analog pins. We will use one digital pin to control when we want the scale on and off. If the pin is HIGH, the pin will be at 5V relative to ground. If the pin is LOW, the pin will be at 0V relative to ground. The only difficulty is that the scale runs on 3.1V and not 5V, so we cannot control the scale directly from an Arduino. We will need a circuit in order to control the scale. Using an NPN transistor, we can control when pin 3 on the scale receives either 3.1V or 0V by giving either a HIGH or LOW signal to the Arduino pin. The circuit is below.



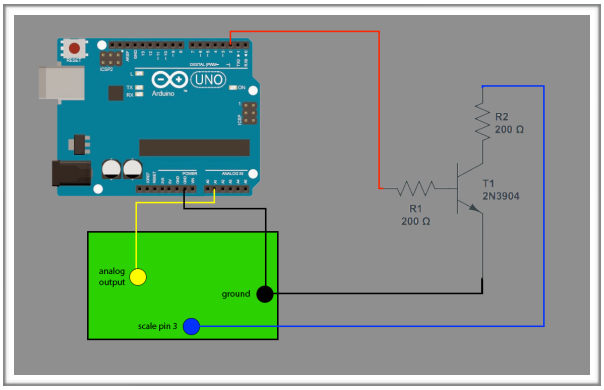
The NPN transistor has three leads. The collector is connected to pin 3 on the scale circuit board. The base is connected to the digital pin on the Arduino. The emitter is connected to the scale ground.

It is now necessary to cut the wires going from the buttons on the scale to the circuit board. Doing this will make the scale unusable as a standard bathroom scale, so there is no going back after this step! We will be cutting three of the four wires. I cut both of the white wires, and the left blue wire (wires 1-3). This leaves the right blue wire (wire 4) still intact, which carries ground voltage. Once the wires are cut we will need to solder another wire to the contact where wire 3 used to be. This wire will be our control wire.

There is one final step before we can close the scale back up. On the lower right side of the board there are four wires (black, green, white, and red in descending order). The black wire is our ground, and we will need to tap into this to make sure that the ground of the scale and our Arduino are all common. This ground wire also connects to the emitter on the NPN transistor. Solder one wire onto the black contact and you are done! An image of the finished circuit board, with soldered wires, is shown below.



In order to connect the wires into Arduino pins, I recommend that you also solder a header onto the end of each of the three wires. This will allow for easy connections to be made between the Arduino and the scale. Also, I recommend drilling a hole through the plastic back of the scale that we unscrewed at the very beginning so the four wires can easily exit. The a complete diagram for this project is below.



V. Calibrating the scale

Now that your scale is connected to an Arduino, you can begin the process of calibrating it! You will first need to connect the 3.3V pin on the Arduino to the AREF pin on the Arduino. Make sure to use the analogReference(EXTERNAL); command in your code. This sets the analog reference to 3.3V. Thus, the Arduino will divide the range of 0-3.3V into 1024 equal steps and compare the analog input signal to this. When no weight is applied, the scale gives an output voltage of about 1.7V, which gives a value of about 527 on the Arduino. We first need to correctly correlate our zero on the Arduino with zero weight, and this can be done by averaging over a large number of values. Our code is set up to do that as well.

```
double scaleValue = 0;
int iter = 75;

for(int i=1; i<=iter; i++){
    scaleValue += analogRead(scalePin);
    delay(15);
}

scaleValue = scaleValue / iter;
Serial.println(scaleValue);
```

Every 15ms, this code will take a reading from the scale and average 75 of them together before outputting them to the serial monitor. This allows for increased accuracy and will give you a correlation value close to 527. You can then add a line of code into the for loop to subtract this off from each measurement. I found this number to be 532.8.

Once you have properly calibrated the zero, you can now find the equivalent weights to analog values. For this, I used a variety of objects (or people) with weights between 0 pounds and 200 pounds. For each of these objects, I placed them on the scale and recorded the weight reading from the scale and the analog signal read in by the Arduino. Then using Excel I was able to determine a quadratic correlation between the data points. There is clearly some error in using this method, and a quadratic is not accurate enough to determine exact weights, but it gave me reasonable and consistent values. This quadratic equation can be put into the Arduino code and run to determine the weight from the analog signal.

```
double scaleValue = 0;
double weight = 0;
int iter = 75;

for(int i=1; i<=iter; i++){
    scaleValue += analogRead(scalePin);
    scaleValue -= 532.8;
    delay(15);
}

scaleValue = scaleValue / iter;
weight = 0.001*scaleValue*scaleValue + 4.5899*scaleValue + 0.0007;
Serial.println(weight);
```

The scale should be re-calibrated relatively frequently to avoid drift errors or other issues.

VI. Programming the Arduino

The entire code for this project can be found online at http://github.com/glenmeyerowitz/readScale.