Circuits Final Project: nMOS Half Bridge

Erin Pierce and Evan Dorsky

May 2014

1 Introduction

H-bridges are circuits that can be used to control DC motors, even fairly powerful ones, with just a digital logic signal. They can control both the motors' speed and direction. H-bridges also have applications in switched mode power supplies. An H-bridge is composed of two half bridges, one of which we built for this project. This allowed us to run various motors in one direction. By varying the duty cycle, we could indeed control the speed of the motor, though we chose to focus on the affects of varying the supply voltage instead for the writeup. Since the half bridge has a high-side and a low-side switch, it is simplest to build them out of one nMOS and one pMOS transistor. We wanted to explore using two nMOS transistors instead, using a bootstrapped capacitor to drive the upper nMOS. One might make a design decision like this because power pMOS transistors are less efficient and more expensive than their nMOS counterparts.

2 The Circuit

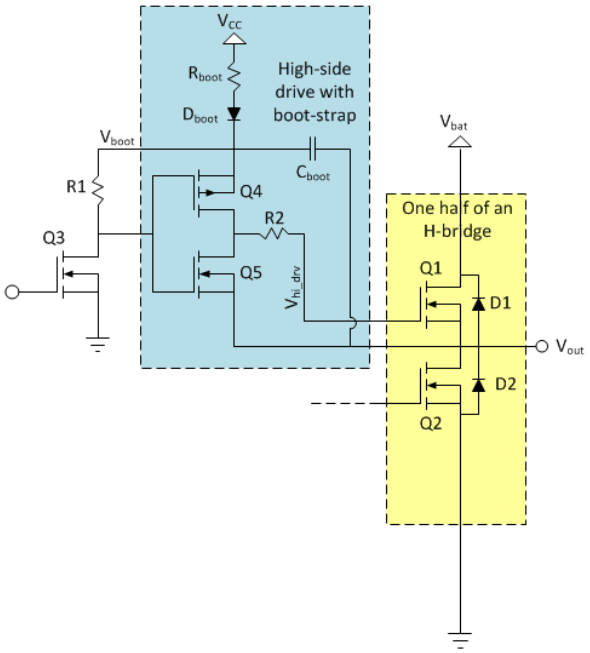

Figure 1: The circuit we used, taken from `http://modularcircuits.tantosonline.com/blog/articles/h-bridge-secrets/h-bridge_drivers`

The circuit we decided to use is shown in figure 1. We used the IRF610 for the power nMOS, which has a continuous drain current of around 3A and a threshold voltage of around 3V. We used the 2N7000 for the little nMOS and the ZVP2106 for the little pMOS because they were what we had on hand. Though it is not shown in this schematic, we used another standard gate driver for the low side power nMOS.

To control the circuit, we used an Arduino, and there are four basic steps. Let us assume that the high-side transistor is off and the low-side one is on. The means that V_{out} is initially at ground.

1. First we would send a high control to the low side. Since the signal is inverted in the gate driver, this results in the low side being turned off.

2. The Arduino would then send the high signal to the high side. This signal is inverted by Q3, bringing it to ground. It is again inverted by the gate driver, and V_g is pulled up to V_{boot}, which is at V_{dd}. Since V_{out} is still floating around ground, C_{boot} is charged to have a voltage drop of V_{dd}. Additionally, we have enough V_{gs} to switch the high side nMOS on. As soon as the high side transistor turns on, it pulls V_{out} toward V_{dd}. As V_{out} moves up, so does V_{boot} and V_g. The diode prevents the capacitor from discharging back into V_{dd} and V_{boot} reaches about $2V_{dd}$, keeping the high-side transistor on.

3. Now we bring the control signal for the high side back to low. This gets inverted twice, bringing V_g to V_{out} and shutting off the high-side transistor.

4. We then bring the control signal for the high side back to low as well. This gets inverted, turning the low-side transistor on and bringing V_{out} back to ground. During this phase, C_{boot} can top off any charge it lost. This cycle then repeats. It is important that there is some time where both transistors are off. If they are on at the same time, they effectively short V_{dd} to ground, and this current surge could damage components.

The output should be a square wave between ground and V_{dd}, and changing the ratio between the on time and off time of the transistors controls the duty cycle of the output, and thus the speed of the motor.

3 Comparison to SPICE

We simulated the half bridge in LTspice to decide on component values and how to switch the FETs. We also used the simulation to validate the behavior of our circuit once we had built it.

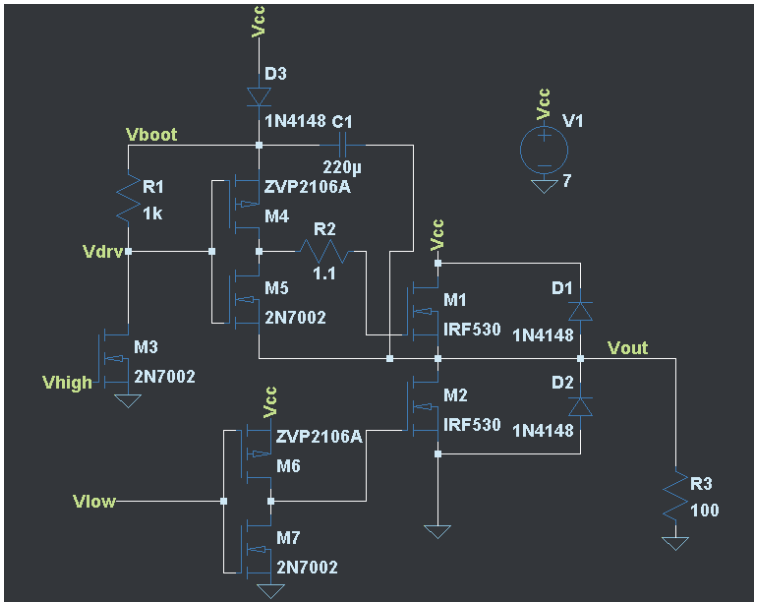

Figure 2: The LTspice schematic of the half bridge we built

We loaded both the real and theoretical circuits with 100Ω resistors and compared important voltages to see if our circuit's behavior matched the expected behavior. The data comparison can be seen in figure 3.

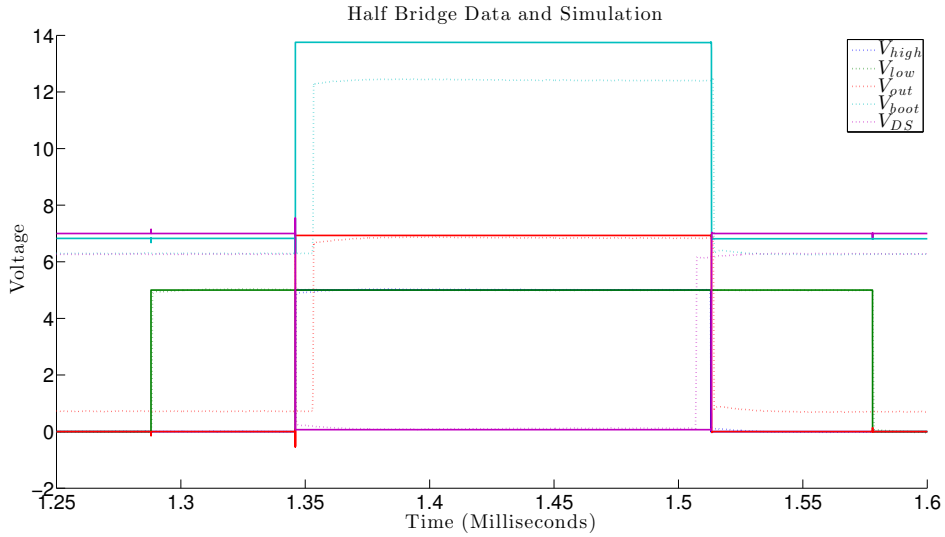

Figure 3: Dotted lines are gathered data and solid lines are theoretical data. V_{DS} is the drain-source voltage of M_1.

As figure 3 shows, our circuit behaved as expected. The small deviations of V_{out}, V_{boot}, and V_{DS} from

ideal can all be explained by considering the effect of V_{out} never dropping all the way to ground. As a result of this, V_{DS} $(V_{cc} - V_{out})$ never reaches V_{cc} and V_{boot} does not charge all the way to V_{cc}.

If V_{out} was never pulled to ground, then M_2 must not have been turned all the way on by the V_{low} pulse. This is actually reasonable, as the low side driver is weaker than the high side driver. The Arduino V_{low} pin needs to drive two FETs, whereas V_{high} drives only one. Furthermore, M_3 will be a stronger driver than the Arduino on its own. We didn't focus much on the low side driver in this lab, but we could have fixed this problem by adding an M_3-like level shift nMOS on the low side driver and flipping V_{low} in the Arduino code.

4 Driving Motors

Once we had confirmed that the circuit behaved as expected, we selected three motors to drive with the half bridge, as shown below.

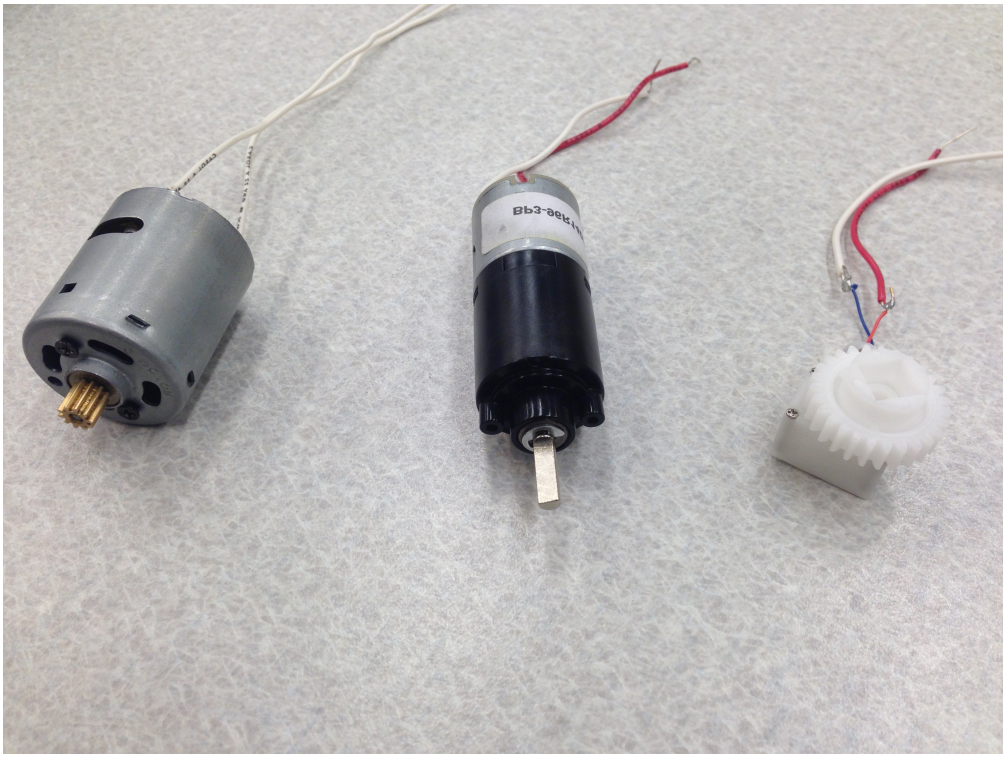

Figure 4: From left to right: M1, DC motor; M2, large gear motor; M3, small gear motor

We then did some rudimentary characterizations of the motors.

Motor	Resistance (Ω)	Current Draw at 6V (mA)
M1	10.7	54.7
M2	1.5	152
M3	25.2	14.2

We found that for each motor, there was a threshold voltage below which the half bridge could no longer drive the motor well. We believe that this has to do with R_{dson} of the power transistors. We found the following helpful graph on the IRF610 datasheet.

For these curves, R_{dson} is the inverse of the slope of the lines. So if the motor draws I_{out} at V_{dd}, and V_{gs} is about $V_{dd}$for both transistors, if we increase V_{dd} we increase V_{gs}, and we also increase I_{out} (because

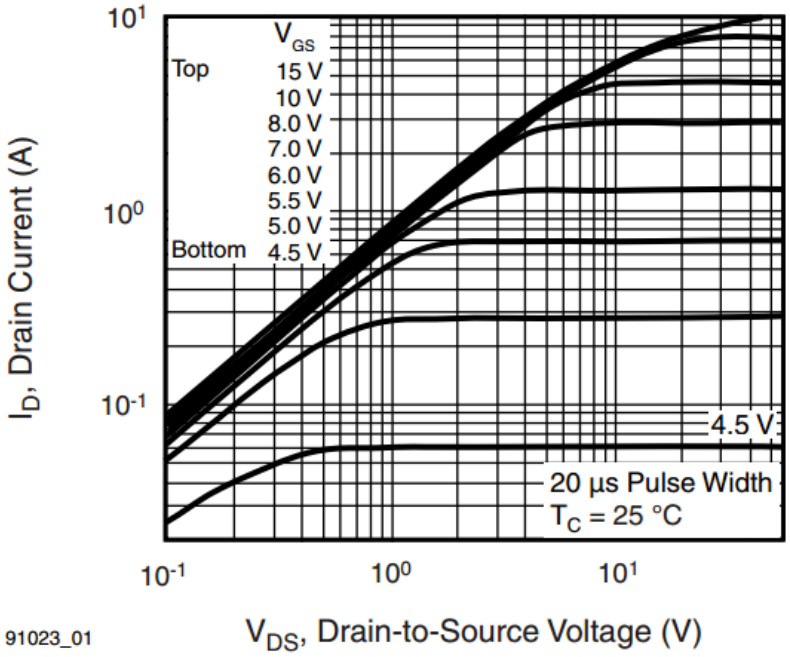

Fig. 1 - Typical Output Characteristics, T_C = 25 °C

Figure 5: Graph from the datasheet showing the factors that affect R_{dson}

V_{out} increases). Increasing I_{ch} and V_{ds} would seem to push us into the high resistance region, but, especially at the more modest V_{gs} values we used, jumping up a line to a higher V_{gs} was plenty to offset this. As we increased V_{dd}, we were less likely to be in the flat, high resistance area. When we were in this region, the increased resistance prevented the half bridge from producing a clean square wave output. The figures after the conclusion show the half bridge working, barely working, and not working for all three motors.

As we would expect, since motor 2 requires much more current than motor 1, it takes a higher voltage to successfully operate motor 2. We would expect motor 3 to require the least voltage to operate well, but this was not the case. Even at higher voltages, our half bridge was unable to drive it cleanly (though it did turn). We have no explanation for why this might be, other than that there is something internal to the motor that we don't know about, like a protection diode.

5 Conclusion

As mentioned in the introduction, driving a pMOS on the high side is simpler than driving an nMOS, because the pMOS is already referenced to V_{cc}. To turn on a high side nMOS, we must bring its gate voltage above V_{cc}. But in high power circuits, the higher efficiency and lower price of nMOS transistors outweigh the difficulties of high side switching. We wanted to learn more about the nMOS half bridge because it is a common circuit with many applications. A power half bridge is also a circuit that is often customized to meet requirements, rather than simply bought, so this project gave us some experience that may serve us well in the future.

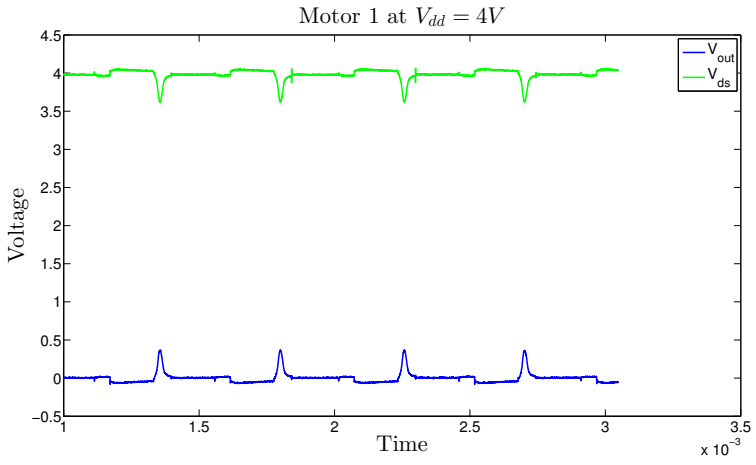

Figure 6: Motor 1 not working

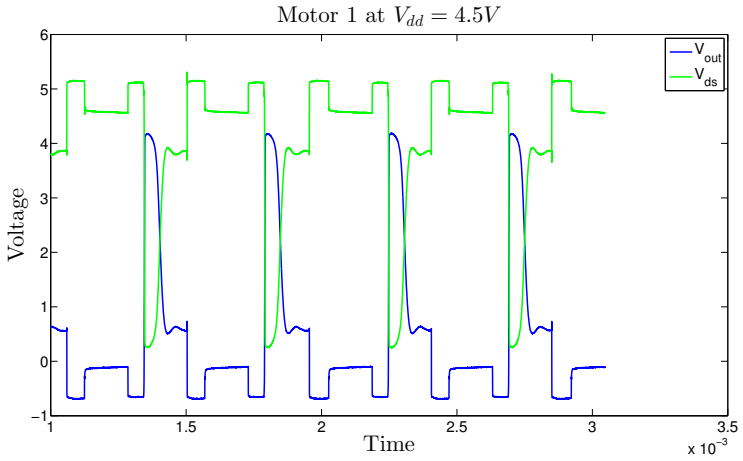

Figure 7: Motor 1 sort of working

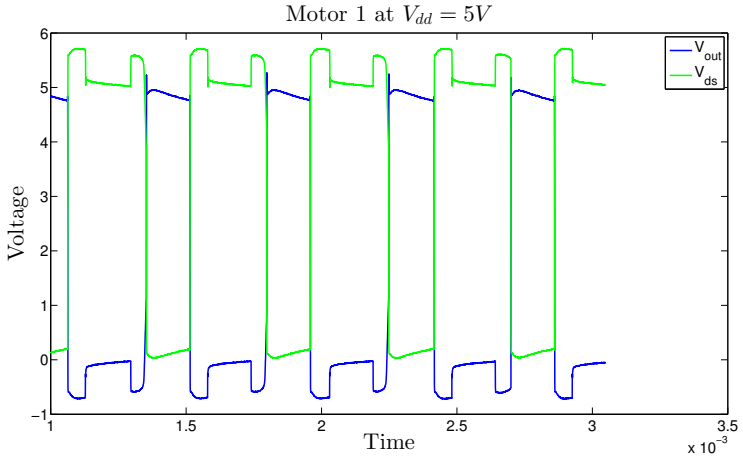

Figure 8: Motor 1 working

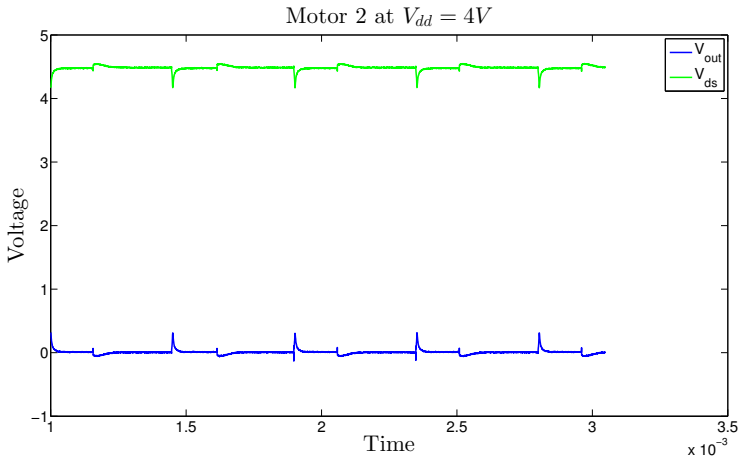

Figure 9: Motor 2 not working

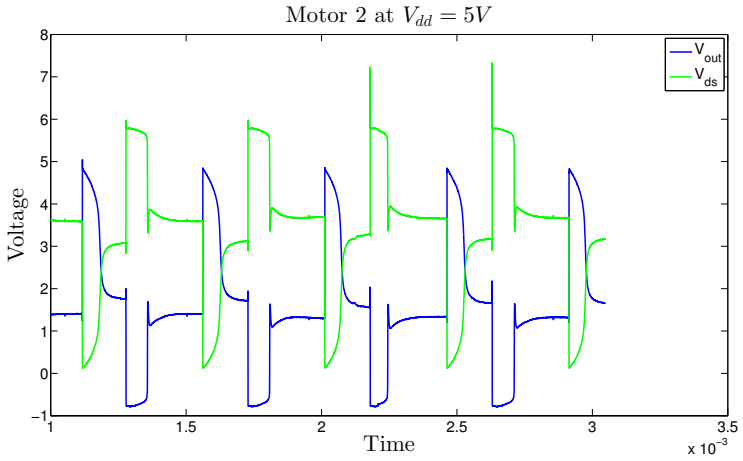

Figure 10: Motor 2 sort of working

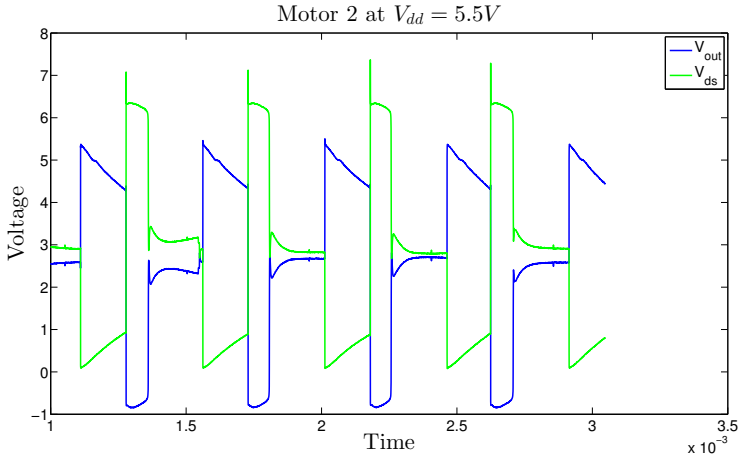

Figure 11: Motor 2 mostly working

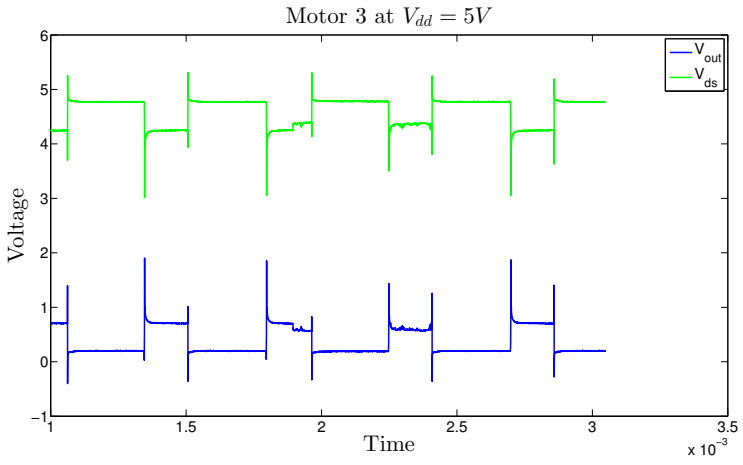

Figure 12: Motor 3 not working

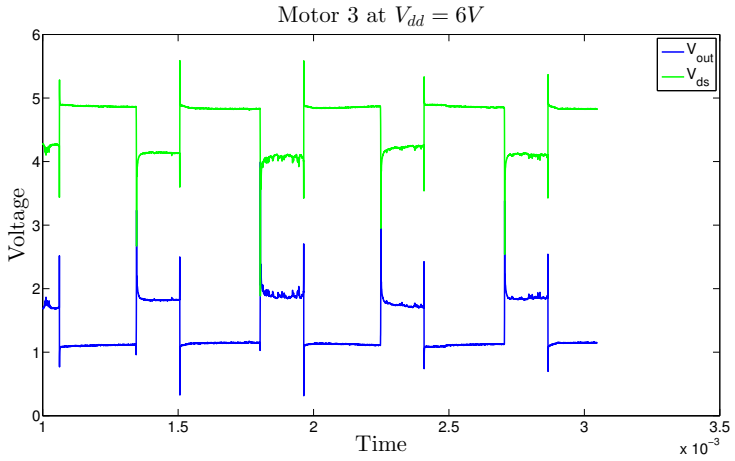

Figure 13: Motor 3 sort of working

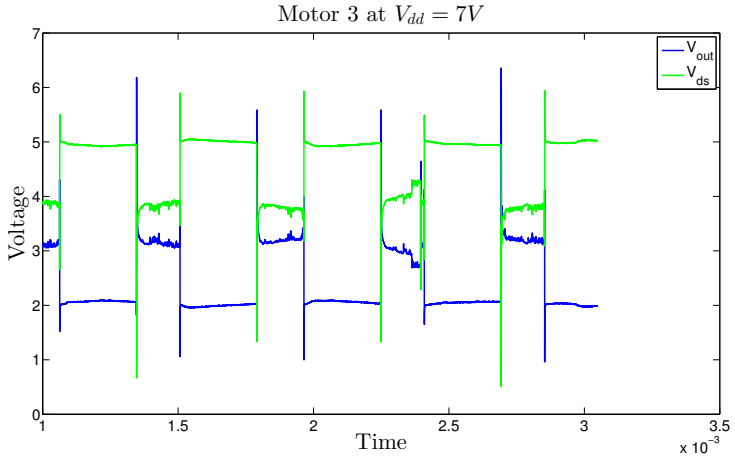

Figure 14: Motor 1 almost working

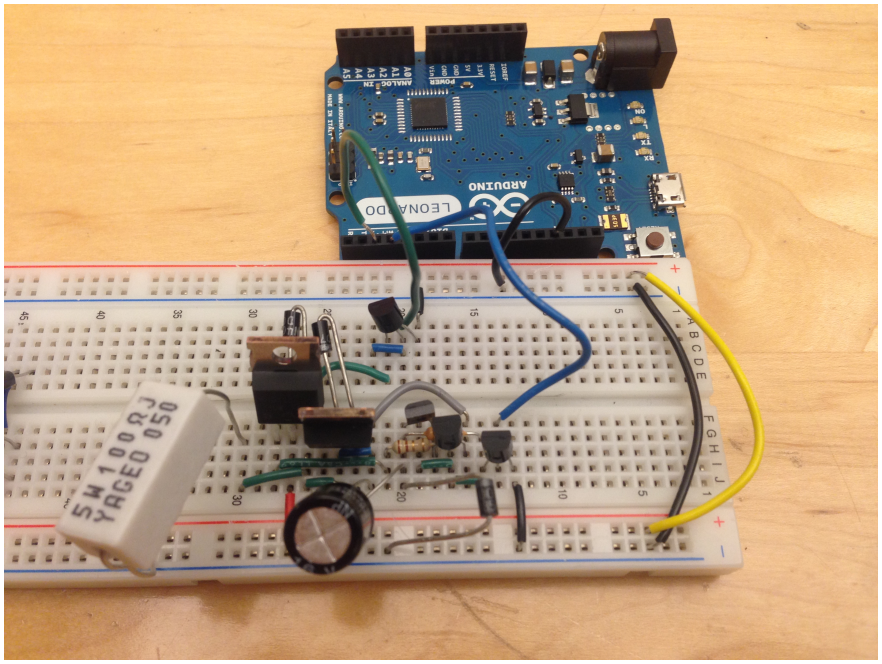

Figure 15: Our project setup. Not pictured: power supply, computer to power Arduino, Analog Discovery for data collection.