Dribbler Motor Writeup
Lisa Liu
January 10th, 2013

Purpose: In the Robocup competition, many teams employ a dribbler in order to put backspin on the ball so that the robot controls the ball while moving. Although RFC-Cambridge installed a prototype of the dribbler, it has not yet been used in gameplay. For the Fall '12 semester of Power Electronics (6.131), my final project was to get the dribbler running in response to signals from the microcontroller, so it could put backspin on the game ball.

Background: The dribbler mechanism is composed of a DC dribbler motor and a rubber tube. The tube is placed onto an spinning axis, which is connected to the motor with an elastic band. Therefore, when the motor spins, so does the tube, which spins the ball backward.

The DC motor spins at a rate proportional to the applied voltage. That is, w = k*V, where w is the rotational velocity; k is the motor constant; and V is the voltage applied. We can vary the voltage using a PWM signal.

The PWM signal comes from a microcontroller on the auxilliary kicker board. However, the microcontroller only outputs 0 and 5 V, so there is some additional circuitry on the auxilliary kicker board which buffers the 5 V up to 12 V.

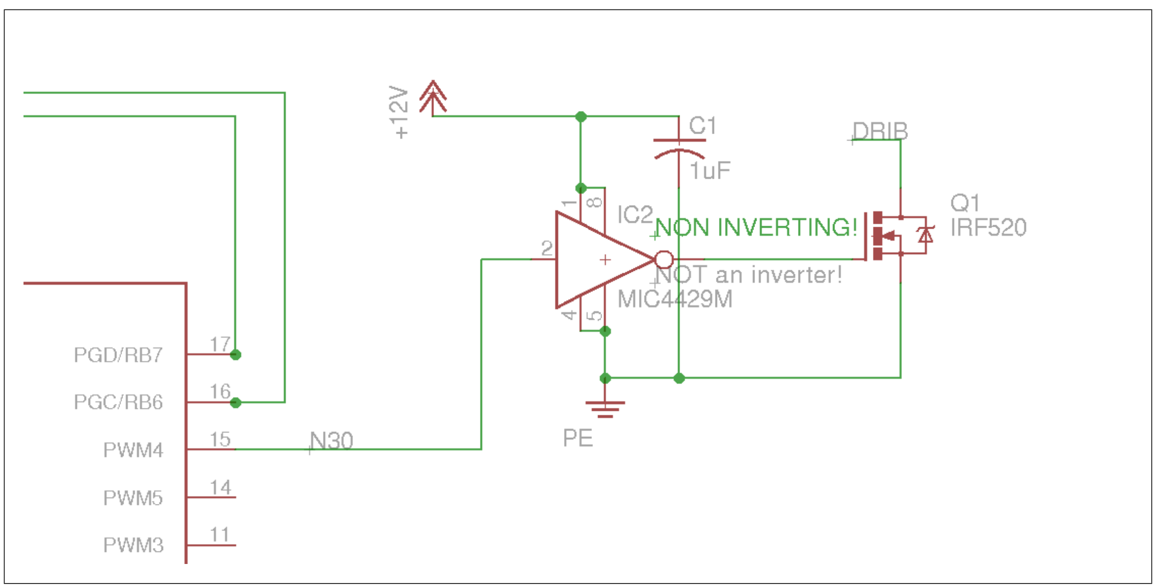

Figure 1. The microcontroller on the auxillery kicker board outputs a PWM signal from pin 15 (`PWM4`) which goes to the `MIC4429M` MOSfet driver. The `MIC4429M` boosts the 0-5 V signal from the microcontroller up to 12 volts.

The `DRIB` signal is connected to the motherboard, where the motor terminals are. One terminal is connected to the `DRIB` (which is relabeled as “`DRIB-`” on the motherboard schematic). The other terminal is connected to `V+`, which is the positive terminal of the battery.

In gameplay, we would like the microcontroller to process radio commands. When it receives a command to run the dribbler motor, the dribbler motor will ideally put backspin on the ball.

Problem: There are some mechanical problems with the design. That is, the tube doesn't turn so easily, which may make the load too much for the motor. There are varying sizes of elastics, and not all (*cough* very few) of them are conducive to making the tube spin.

A minor electrical issue was that the motor was not properly speced to the robot. The motor could take voltages up to 24 volts, but the robot provided only 12 volts. Even though the motor could backspin the ball at 12 volts, there was concern that the full potential of the motor was not utilized.

The main problem encountered at the EE submeetings is getting the microcontroller to output a PWM signal at all. We suspected that the microcontroller was incorrectly programmed.

Project:

Boost Converter

In speaking to `baker1` from ME, I found out the ME subteam is updating the dribbler design and getting a new motor, which is more properly speced to the robot. This is perhaps the better solution, as building the boost converter requires extra PCB space and relatively large capacitors, which might not fit in the robot and would certainly weigh the robot down. Therefore, this portion of my project is no longer relevant, but may be useful for learning or for future reference.

I built a a boost converter which boosted the voltage to a higher voltage, which depended on the duty cycle according to `V_out = V_in /(1-D)`.

However, the PWM signal only controlled the amount of boosting, and otherwise required the motor to always be connected to the battery. This is unideal because unneeded dribbling will waste battery energy. Therefore, I added a `LM555` timer in one-shot monostable mode to act as a switch; when there was no PWM, the motor was disconnected from battery, else it let the PWM signal power the the robot.

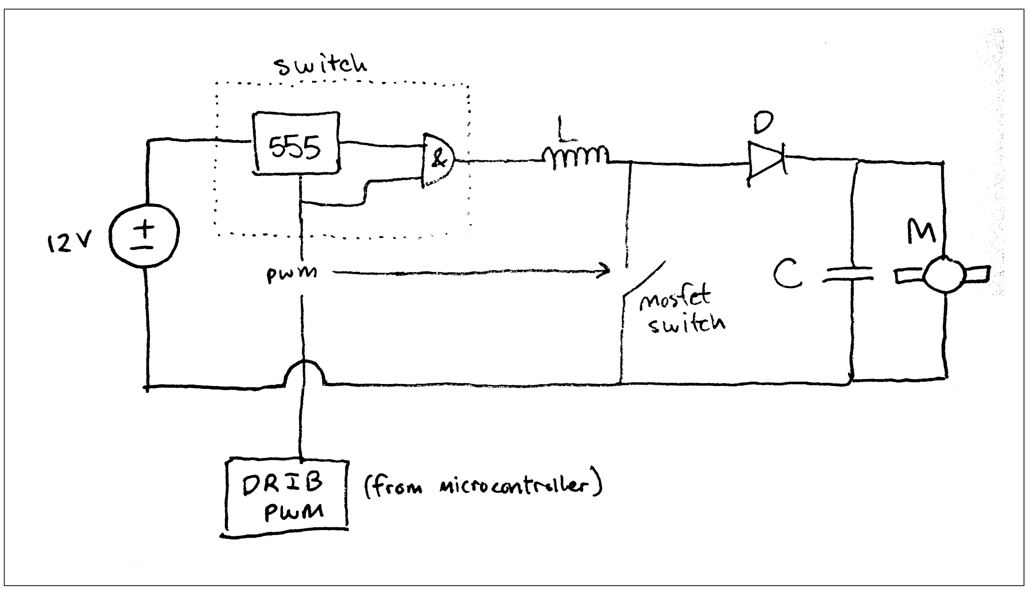

Figure 2. When the 555 is set up in one-shot monostable mode, it will output a `HIGH` signal when it detects a falling edge on the input pin. The duration of the `HIGH` signal depends on an external capacitor and resistor.

Programming the microcontroller:

During subteam meetings, we were unable to successfully program the microcontroller to PWM. The key register that needed to be adjusted was **PWMCON0**. This register needed to be adjusted so that PDC2 would be in complementary mode. Previously, it was in independent mode, which means that PWM1 would output a signal, but PWM4 would be left hanging. PTPERH, PTCON1, and PDC2L (and perhaps a few other registers) control the frequency and duty cycle of the signal.

In a future implementation of the auxilliary kicker board, it may be simpler to use the pins for PWM1, PWM2, or PWM3, which do not require the complementary mode bit to be set.

Measurement:

During subteam meetings, we tried to measure the output of the microcontroller by placing the oscilloscope probes directly on the V+ and DRIB− terminals. While we noticed V+ always as 12 volts, as expected, however, DRIB− was less consistent.

The problem in our method of measurement is that DRIB− is left hanging. That is, when the PWM signal is HIGH, the MOSfet connects DRIB to ground, which is what we expect. However, when the PWM signal is LOW, the MOSfet leaves DRIB hanging, giving output a signal which is neither HIGH nor LOW. Therefore, we need a pullup resistor in order to measure any sort of meaningful signal.

A naïve thought is to use the resistance of the motor as a pull up resistor. However, as we explore in the next section, directly plugging the motor into the motor terminals doesn't work.

Motor:

When I used a pull up resistor on the DRIB terminal, the output of the microcontroller was a PWM signal exactly as we want it to be. However, when we use a motor instead of a resistor, the output looks extremely irregular.

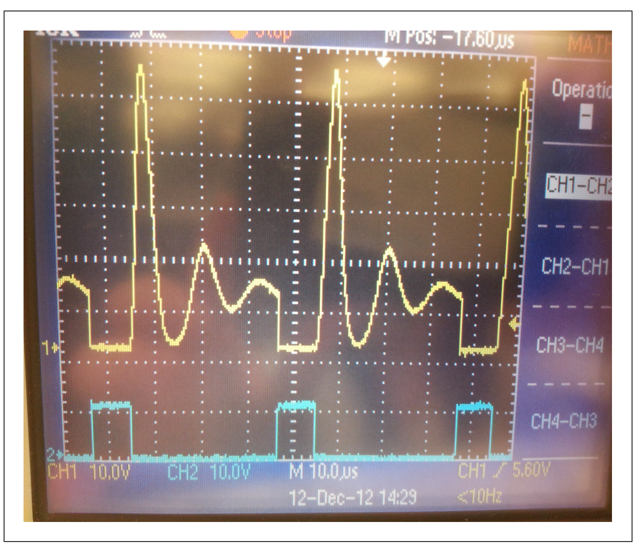

Figure 3. The blue signal is the signal between the MOSfet driver and the MOSfet. The yellow signal is the voltage across the motor; it reaches voltages up to 50 volts and displays significant oscillations.

The oscillations are due the th inductance of the motor, which creates an LRC circuit. This adds reasonance and oscillation to the circuit, explaining the 50 V peak and the inability of the DC motor to operate properly.

Therefore, a diode should be placed across the motor, between the `DRIB-` and `V+` terminals, facing the `DRIB-` terminal, in order to stop the capacitor from charging and creating the reasonant circuit.

Conclusion:

A boost converter is not needed. But perhaps in future implementations of the boards, the following changes should be made:

- Auxillery board should be use PWM1, PWM2, or PWM3, before using PWM4.
- Motherboard should put a diode across the dribbler terminals.

A boost converter is not necessary, since it seems 12 V is enough to provide backspin on the ball. We will leave it up to the ME subteam to get the ball rolling.