EECS 2210 Finale Project Report: Solenoid Valve Driver

Arthur Sabadini Nascimento, Student Number: 220007175

April 15, 2024

Contents

All design files and proposal reports can be found in this GitHub directory (https://github.com/ArthurSabadini/eecs2032-braille-writer), and the demonstration video can be found here.

(https://www.youtube.com/watch?v=A-jqoaMqUTU)

Section 1

Abstract

The project consists of a solenoid valve driver that should be controlled with a microcontroller, as an Arduino, for example. Using resistors, a MOSFET, a power supply or battery, diodes and an Arduino microcontroller, the driver should be able to control a solenoid valve safely. Since solenoid valves are inductors, and on the action of switching it on or off, this would generate a large current in the solenoid, potentially damaging the MOSFET and/or microcontroller. This project was inspired by the following website [4].

Roles

- Arthur Sabadini Nascimento
  - Circuitry Design

    Design the circuitry of the system, the series of six solenoids connected to a control unit (transistor and a diode, to control current use) and overall inputs/outputs to the Arduino and other devices. These solenoids should also be controllable from the Actuator control unit.
  - Infrared Communication Development

    Develop a system that can recognize a series of inputs from an infrared device (concerning some protocol), and encode those signals into a data format that the translation software can understand (could be just English text). Then feed the encoded data to the translation software.
  - Testing

    Run vigorous testing on the software and its compatibility with the hardware

Expected Results

I expect to be able to safely control the solenoid valve from an Arduino microcontroller, being able to switch it on and off without hurting either the MOSFET or Arduino. This setup should also be duplicable, to how many other drivers the user needs, perhaps I could add a shift register so it is possible to control many drivers from a single microcontroller.

Section 2

Solenoid Driver

I have designed a basic solenoid driver for the control of the six push-pull solenoid valves. The driver (for one solenoid) requires the following items

Components

The IRF520 ([3]) MOSFET was chosen because it can handle gate-source voltages up to 100V, and a drain current up to 9.2A. It will serve for this application.

The 1N4001 ([1]) diode was selected because it can handle a spike voltage up to 400V, a constant current up to 1A, and a non-repetitive current peak up to 30A. It definitely will protect the MOSFET and Arduino from any damage.

Driver Design

The circuit below was the chosen design for the solenoid driver. The MOSFET is used for switching the solenoid on and off, and the diode is for flyback protection

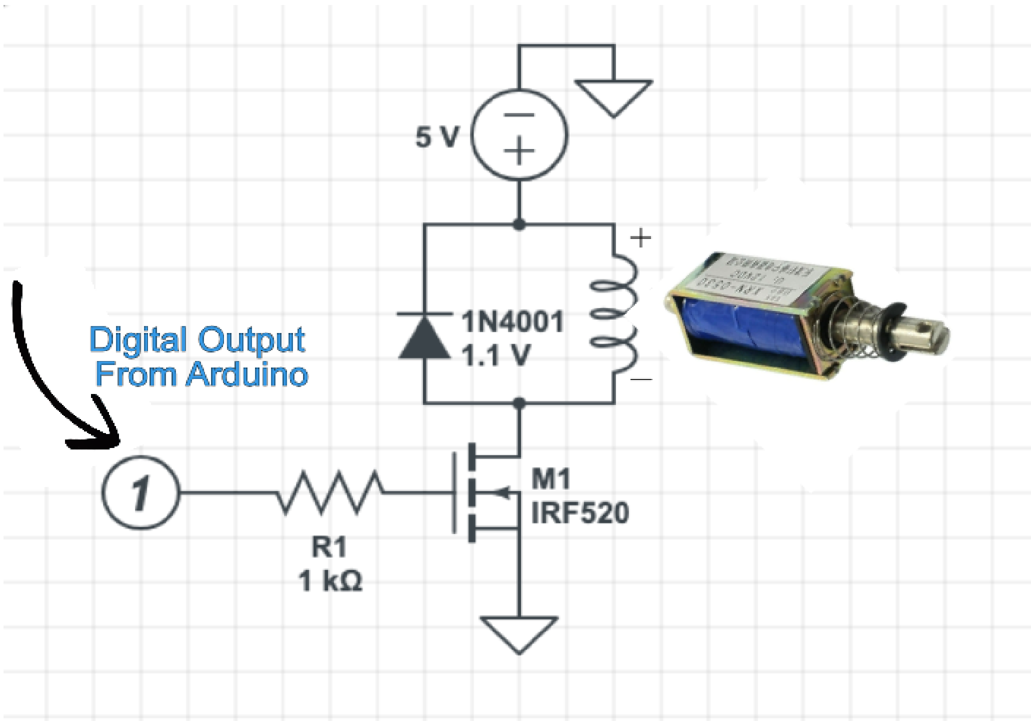

Figure 1: Solenoid Driver Circuit

Implemented Driver

The circuit below is the implementation of the driver described above. This driver circuit was constructed 5 more times (6 in total) to control all 6 solenoids.

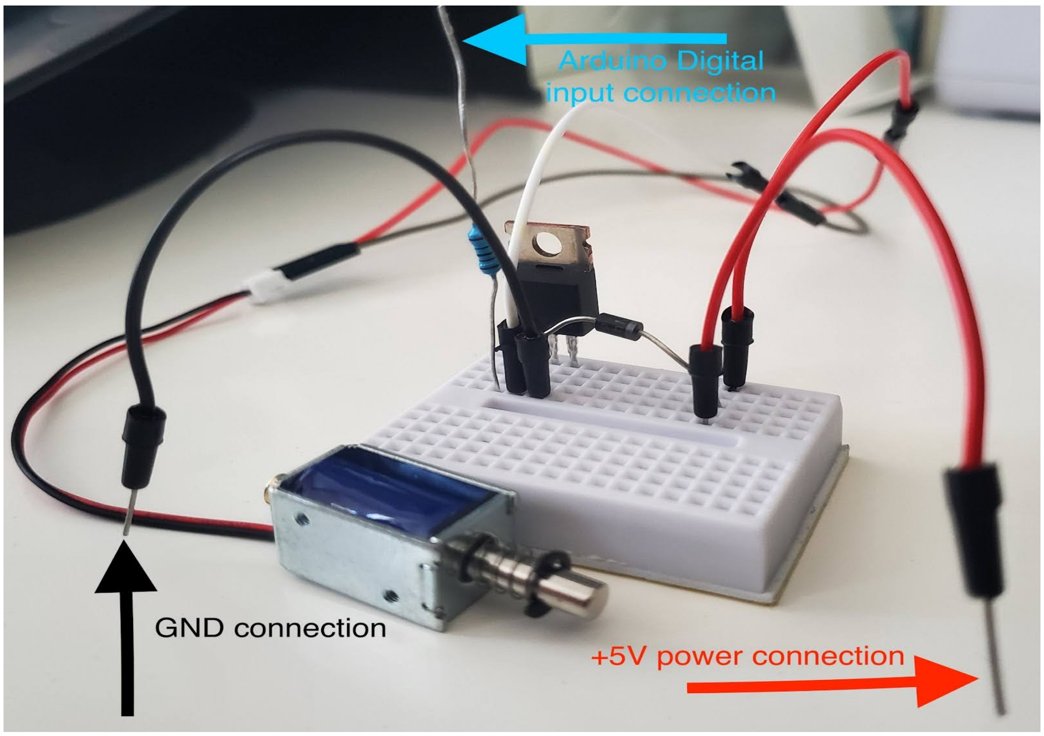

Figure 2: Implemented Driver

Measurements

When making measurements in the lab, I noticed that the solenoids were drawing more current than I had predicted.

The below measurement was taken across one active solenoid. The current of 1.1A was higher than the predicted in the midterm report (about 100mA).

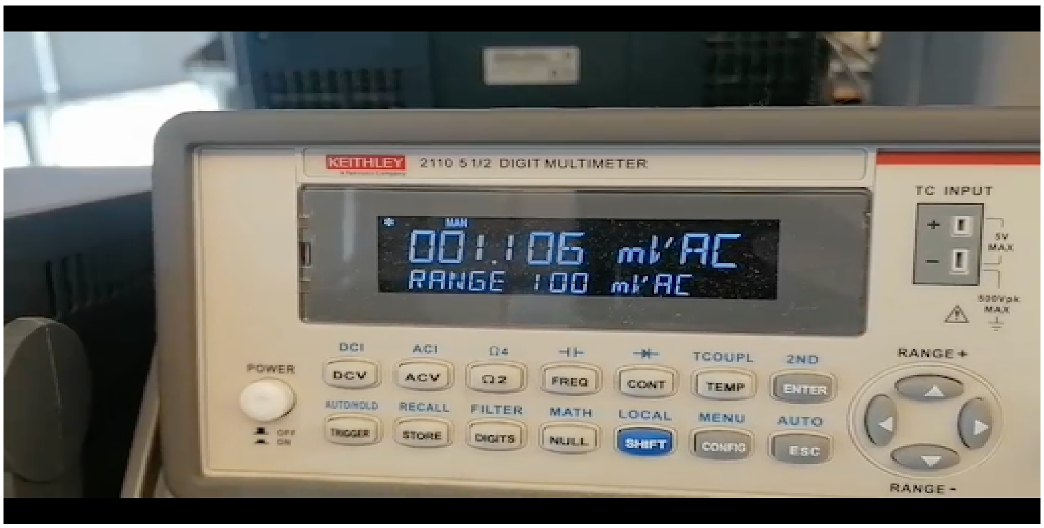

Figure 3: Multimeter Measurement

The problem seemed to be that the IRF520Ns used were too general, which led to miscalculations. To fix this, I replaced the general IRF520s with IRF520NPBFs. IRF520NPBFs have well-documented datasheets, which will make calculations more reliable.

Calculations

Note that the specifications of the solenoid were not provided by the manufacturer, since they are a quite new company. I'll assume the inductance value to be 1H, but according to the simulations and my calculations, this value won't matter for the application.

MOSFET Saturation Mode (Switch ON) Calculation

When the MOSFET is in saturation mode, we'll assume that the solenoid is short and the diode is off to calculate the maximum forward current through the solenoid.

Using the values from the IRF520N PBF datasheet ([3]), $g_m = 2.7S$ when $V_{DS} = 50V$ and $I_D = 5.7A$. Since the MOSFET is in saturation, we assume $\lambda = 0V^{-1}$.

Therefore, we can estimate the constant k as

$$k = \frac{(g_m)^2}{2 \cdot I_D} = \frac{(2.7S)^2}{2 \cdot 5.7A} = \frac{7.29S^2}{11.4A} \approx 640mAV^{-2}$$

Thus, we can calculate the saturation current as

$$I_{Dsat} = \frac{k}{2}(V_{GS} - V_{TH})^2 = \frac{640mAV^{-2}}{2}(5V - 3V)^2 = 320mAV^{-2} \cdot 4V^2 \approx 1.3A$$

We get values that are more accurate than in previous calculations. The error or 200mA can be attributed to deviations of manufacturing parameters, non-ideal components, etc.

Diode ON (Switch OFF) Calculation

When the MOSFET is switched off (still in saturation), we can assume that the diode is on, since then a high current would be detected in the solenoid due to the sudden voltage change.

The MOSFET would then behave as a resistor, with quite a big resistance value. To model the switch off of the MOSFET we assume the Gate-Source channel is open (disconnected), and we would substitute the MOSFET with a resistor, usually labelled as $R_{DS(off)}$. It is common practice to model this resistance as infinite, and we do this in the analysis.

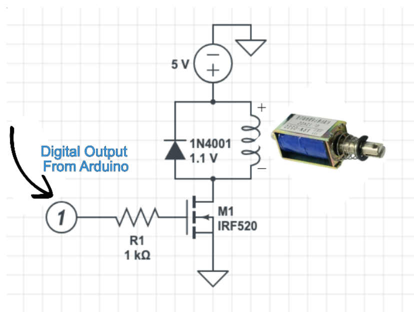

Figure 4: Solenoid Driver Circuit

Performing KVL on the circuit of the above circuit, it yields that the current through the inductive load is given by the equation

$$L\frac{dI}{dt} + V_T \ln\left(1 + \frac{I}{I_s}\right) = 0 \iff \frac{dI}{dt} = -\frac{V_T}{L}\ln\left(1 + \frac{I}{I_s}\right)$$

Using this equation we get

$$\left.\frac{dI}{dt}\right|_{t=0} = -\frac{V_T}{L}\ln\left(1 + \frac{I(0)}{I_s}\right) = -\frac{25mV}{1H}\ln\left(1 + \frac{1.3A}{77.8zA}\right) \approx -1.1As^{-1}$$

Using a linear approximation (similar to midterm report), we get that $I(t) \approx I(0) + t \cdot dI/dt|_{t=0} = 1.3 - 1.1t$. We can approximate the time it takes for the solenoid to discharge as $\hat{t} \approx 1.3A/1.1As^{-1} = 1.2s$.

Swicthin Event Simulation

We made a simulation to give us an idea of how the switching event would happen, and what kind of stresses would be applied to the components.

A PWM input was generated, at 0.67Hz and 17.34% duty cycle. This would keep the MOSFET on for 260ms, within a period of 1.5s. A shunt resistor was added to better visualize the input and output signals.

Below is the simulation of the switch-off event. We see the numbers are quite close to the previously calculated ones.

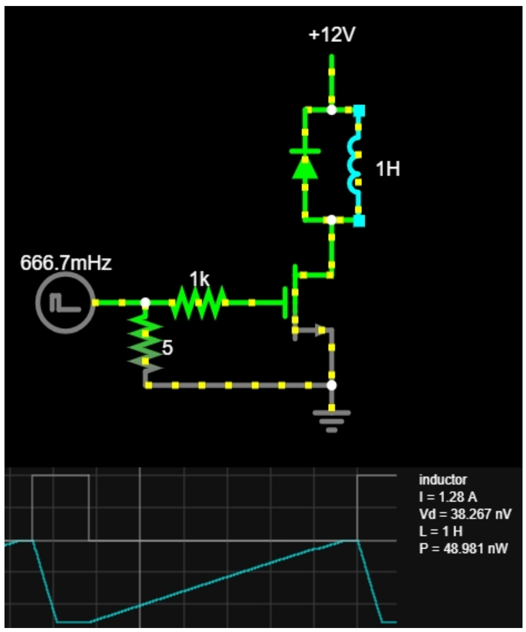

Figure 5: Falstad Simulation [2]

Final Circuitry Design

Full Design Schematic

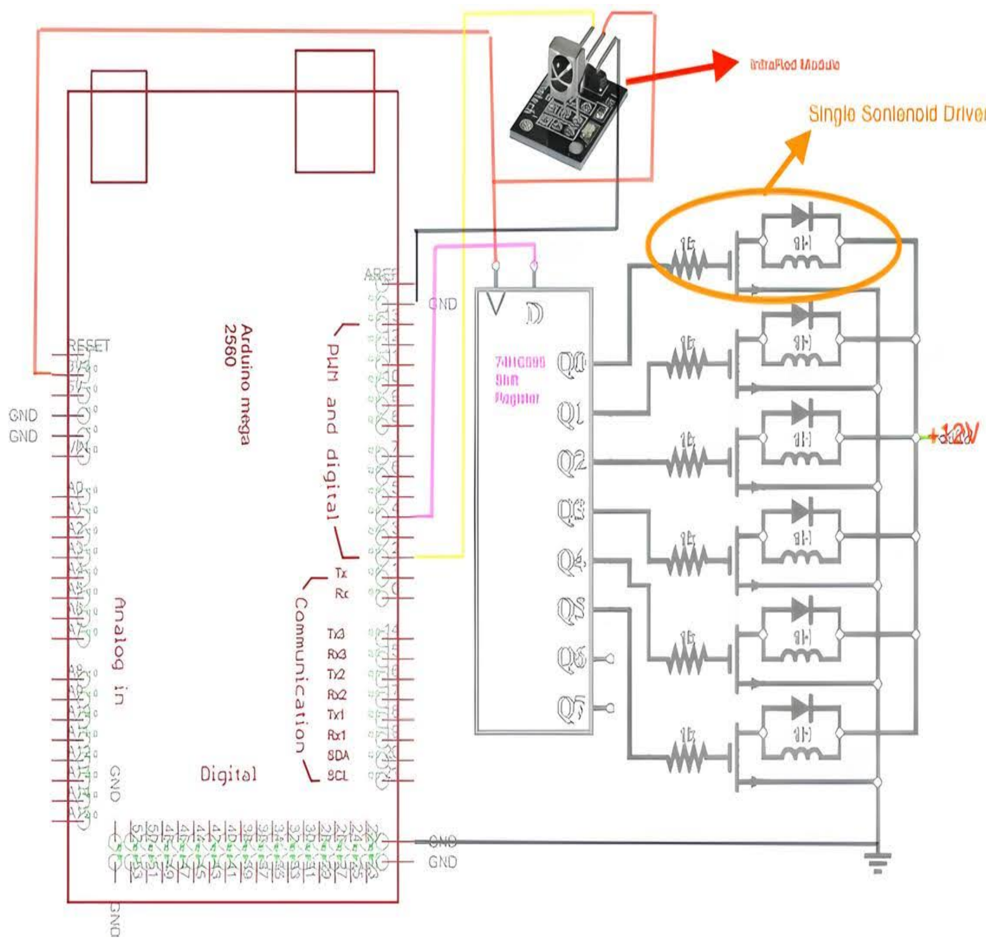

Figure 6: Full Schematic

Implemented Circuitry

The circuit below is the implementation of the full circuitry. In orange, all 6 solenoid drivers. In pink, the shift register used to control all solenoid drivers. In red, the infrared sensor, used to get input from a remote control.

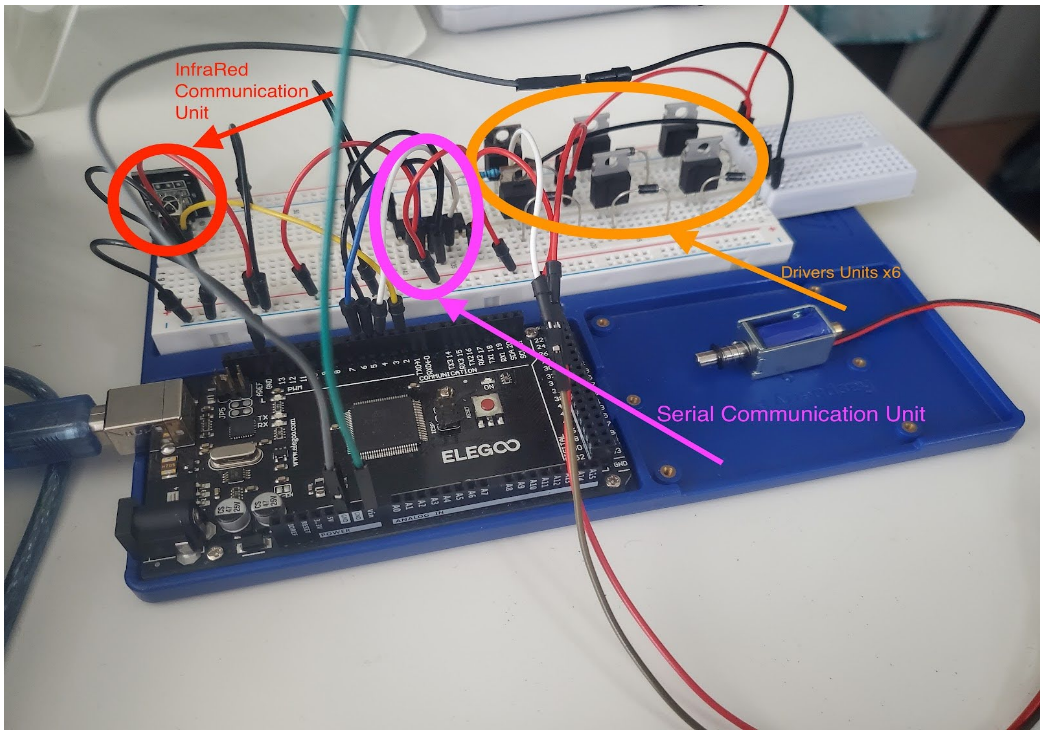

Figure 7: Implemented Circuit

Section 3

How has my Role benefited from the course 2210?

My role in this project, as the hardware designer, has benefited quite a lot from the course 2210.

The designed driver uses components such as MOSFETs and diodes, components that we learned in depth about in the course. Without such components, I wouldn't be able to design the driver, since MOSFETs are usually used for rapid switching with microcontroller applications.

What problems were solved?

We could not just control the solenoids directly from the Arduino pins; the Arduino can't provide enough current to actuate the solenoids. To fix this, I designed a solenoid driver that can be used to safely control each solenoid from the Arduino.

Initially, I thought I could implement the design using BJTs. However, after some calculations, I noticed that the solenoids required more current than a common BJT could provide, MOSFETs were adopted instead.

Another problem was that if the solenoids were kept on for too long, that could cause the MOSFETs to heat up too much. Testing showed that an "on" period of 500ms was enough to be detected, and to minimize stresses, we turned the solenoids off for 250 ms after it was actuated.

What has been learned?

In the process, I learned how to design motor drivers, which was quite exciting! And I also learned how to best select a microcontroller for a given project, depending on the hardware needs.

I also learned how to read datasheets, since I had to do that a lot to check each component's parameters and limits.

Can this project be continued and result in a marketable product?

Yes! There is a great need in the space of accessibility in technology. In the market, there are tons of projects that convert braille to English, but none that do the opposite, as our project does

In conclusion, our project would improve accessibility for people with vision impairments, and there's also a need for it in the market. So indeed it could result in a marketable product!

How successful was the project in achieving its goals?

The project was very successful, all expected results were achieved. I was able to design a driver to easily and safely control all 6 solenoids from a microcontroller, and this design was successfully implemented in my project for 2032; it was used to translate English text to the braille language for people with vision impairments.

References

[1] Fairchild Semiconductor. "1N4001-1N4007 - General-Purpose Rectifiers". In: (2003). URL: https://www.mouser.com/datasheet/2/149/1N4001-81693.pdf.

[2] *Falstad Simulation*. URL: https://tinyurl.com/27pxqw32.

[3] *IRF520NPbF - Power MOSFET NPN Silicon*. URL: https://www.infineon.com/dgdl/irf520npbf.pdf?fileId=5546d462533600a4015355e340711985.

[4] *Solenoid Driver Circuit Diagram*. URL: https://circuitdigest.com/electronic-circuits/solenoid-driver-circuit-diagram.