Valve Control Group Write Up

Patrick Borgman, Jaimie Thomas, Austin McCright, Harun Resulovic

Surf Rover Project Spring 2020

**Description**

The valve control group was tasked with developing a hard wire, and eventually RC, control system that would be used to turn the surf rover valve tracks that are actuated by 24 VDC solenoids. The hard-wired control system was powered by a 24 VDC source, had an Arduino Mega board as the main processor, and a series of relays and amplifiers that would be responsible for sending the actuating signals to the rover relays. Power electronics components would have to ensure that components such as the Arduino and relays receive their rated voltages, and amplifiers would have to ensure that the actuating signals from the Arduino were received by the correct relay/solenoid pair.

**Background**

No previous work was done for this task prior to our group on the big surf rover. However, a quarter scale surf rover was designed and fabricated for a senior design project. The quarter scale model had a full RC control circuit and written code for operation of the steering mechanisms. The same code is believed to work just the same for the big surf rover once the circuit is installed.

**Thought Process**

After going over the information available for the quarter-scale surf rover, the first thing the group did was make a power distribution flow chart in order to help organize what exactly needed to be accomplished for the project. A 12V battery was on hand at the beginning of the project but this was changed to a 24V battery due to the requirements of the valves. The flow chart is shown in Figure 1.

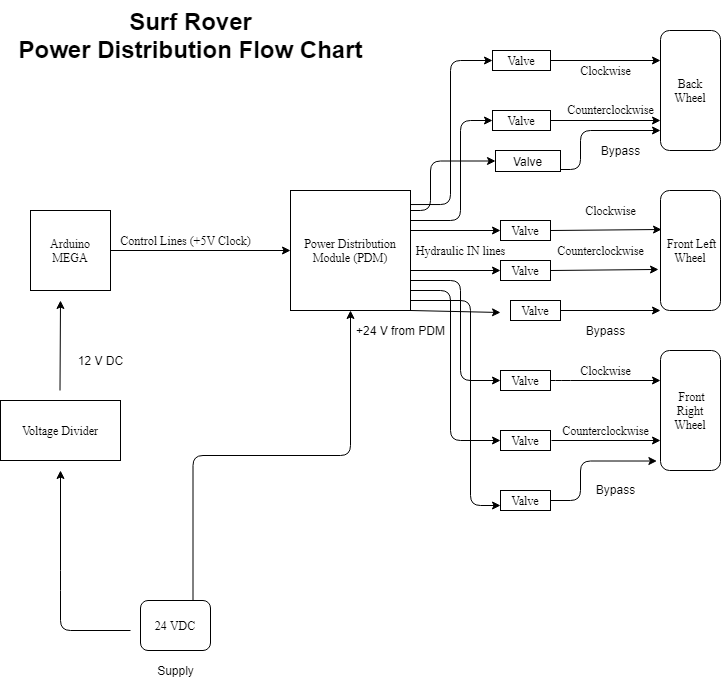


Figure *1. Overall power distribution flow chart.*

After knowing where the power and control signals needed to be distributed, brainstorming could begin on how to accomplish that. Originally, a power distribution module (PDM) was considered for this task. Since these are designed for use on vehicles, adapting them for the surf rover was considered. However, upon further research, it was found that most either use manual switches, which would not work for obvious reasons, or electromechanical relays. This brought up some concern because of how electromechanical works, using a mechanical arm to open and close the output circuit. This behavior is shown in more detail in Figure 2.

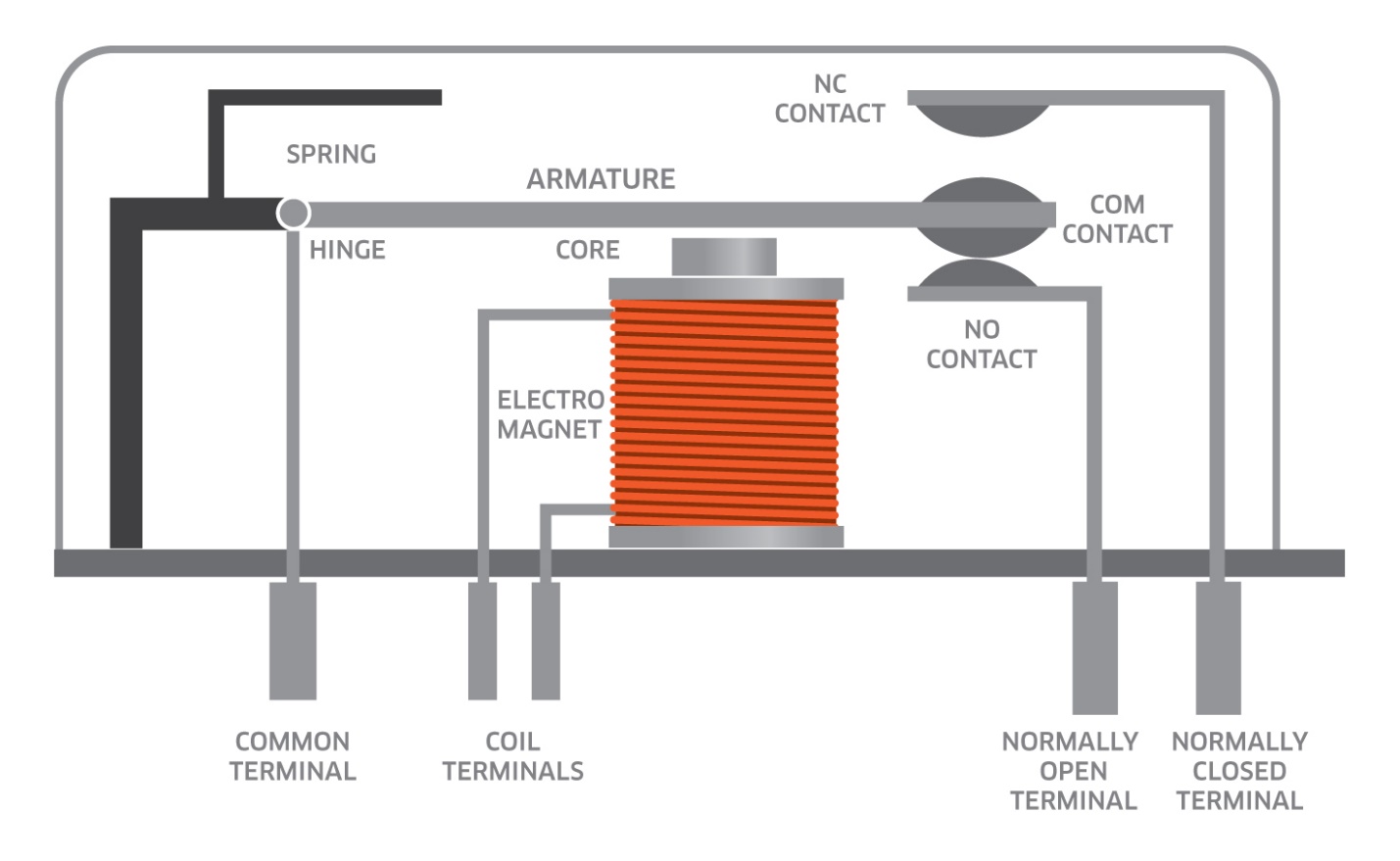


Figure 2. Internal diagram of an electromechanical relay.

Due to the aquatic nature of the surf rover as well as the electrical components being housed next to the engine, concern was raised about vibrations moving the arm and closing the circuit at unintended times. This would be a problem as this would cause the voltage control signal to be sent to the valves erratically, and as these control the steering of the rover, it could make the rover uncontrollable. Because of this, it was decided that the PDM would not be used and would be replaced with a bank of solid-state relays (SSRs). These also allow for a small signal input circuit to act as a switch for a much higher power output circuit, but their internal workings involve no moving parts. A diagram of an SSR is shown in Figure 3.

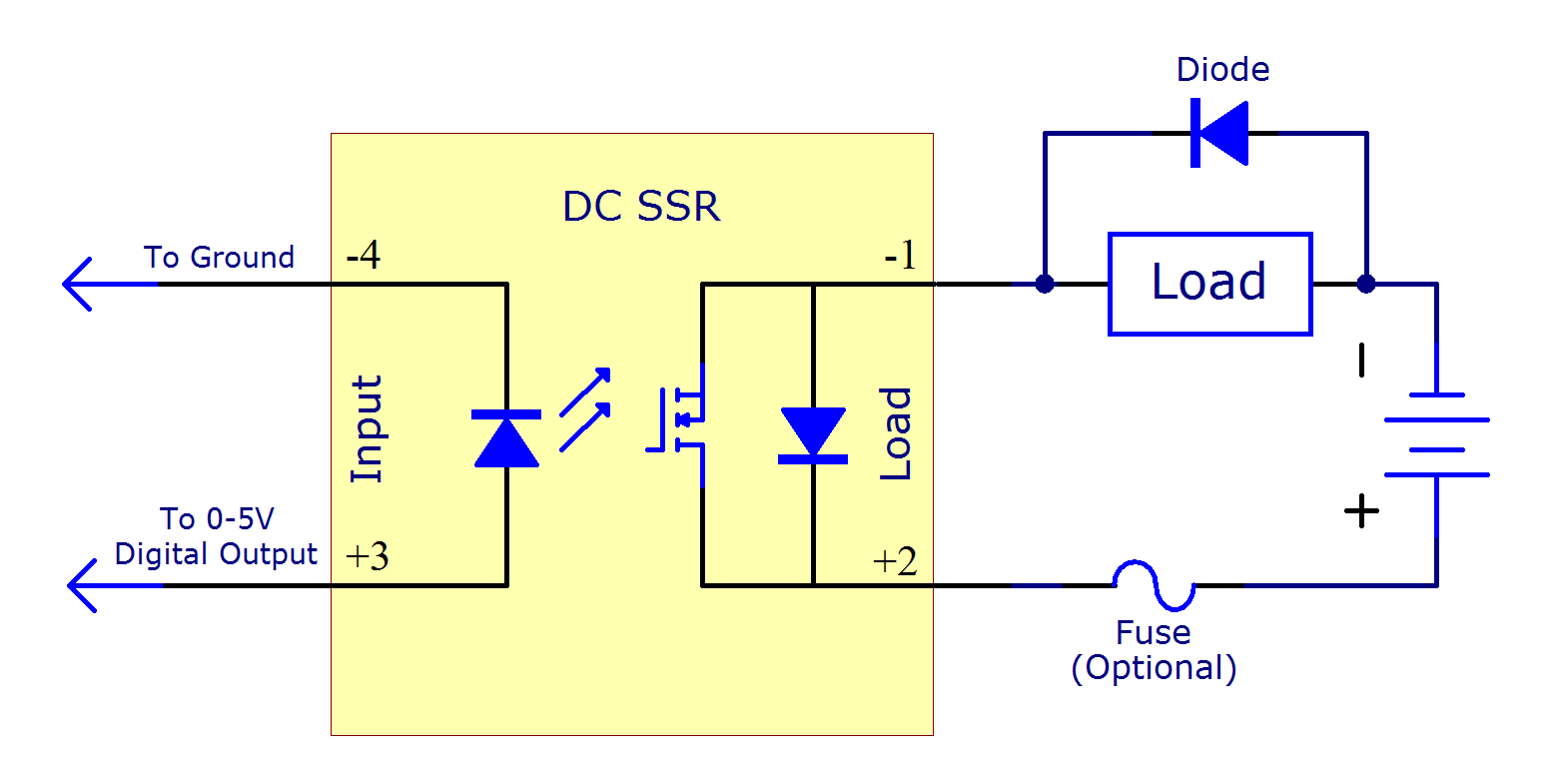


Figure 3. Internal schematic of an SSR.

The way that an SSR works is the input voltage turns on a small LED housed inside the integrated circuit (IC). This in turn activates a phototransistor on the output circuit, which closes the circuit and allows current to flow. This is a fairly common configuration called “opto-coupling”. It has the additional advantage of electrically isolating the input and output circuits, which protects the small signal circuit and, in this case, the Arduino that is controlling all of the circuits.

Because of the 24V battery being too high for the Arduino to run on, a step-down circuit was needed. This was achieved using a voltage regulator circuit built around an LM317H. Finally, there was an amplifier circuit built on the output of the SSR banks in order to make sure the voltage was brought back up to 24V to control the valves. These circuits could be replicated as many times as needed for however many valves are in the final build on the surf rover.

**Thorough Description**

After going through previous designs which used 12 VDC and electromechanical relays, it was determined that a much high powered, non-contact design would have to be used to adapt to a hydraulic vehicle with heavy vibrations. The same valve would be used, but however, solid state relays would be used to pass the actuating signals to the valves, due to their optical coupling rather than electromechanical coupling, which could be thrown accidentally due to the high vibrations on the vehicle. After consulting Dr. Dally and Mr. Eric Rutherford who helped us tremendously with this project, our design input voltage was changed to 24 VDC and stepped down to 12 VDC for sensitive components such as the Arduino Mega; the purpose of going with this design was to avoid excess routing onboard the vehicle, which could obstruct other components such as the diesel engine. To complete such a task, a linear voltage regulator IC was selected over a simple voltage divider due to less power dissipation. The voltage pulses, or clock signals, that were sent to the Arduino to energize the solenoid valves were at 5 V peak, well below the turn on voltage for the solenoid valves. To ensure that the solenoid valves were receiving the correct voltage signals, a series of amplifiers had to be constructed before the solenoid valve to step up the input voltage to 24 VDC, and before the input to the SSR to step up the voltage to 12 VDC. To step up the input to the SSR, a MOSFET was selected, since the voltage source being amplified was periodic and small signal. At the output, an operational amplifier was designed to smooth out the square wave signal which characterizes the input clock signal into a 24 VDC pulse. The semiconductor device needed to be ‘told’ what voltage to amplify up to, which is why the rail, or supply voltages, of the op-amp were biased at +24 VDC.

**Computations explained**

The first step in the calculations process of the control system design was determining the appropriate power distribution to each component in the control system. Because the Arduino Mega Board could only take a maximum input voltage of 12 VDC, an electronic regulator was designed using an LM317 linear regulator IC to step down the voltage to the Arduino’s appropriate operating level. Equation (1) best describes the LM317 output voltage as a function of the control resistances of the regulator, provided by the manufacturer datasheet.

(1)

In Equation 1, control resistances *R1* and *R2* had to be selected such that the ratio of *R1* to *R2* yielded an output voltage of 12 VDC. Manipulating Equation (1) resulted in a resistance ratio of 8.14. To accommodate this computation, resistances *R1* and *R2* were selected as 2.2 kΩ and 270 Ω, respectively.

The next calculation involved the amplifiers found in the control circuit for the Arduino output signals. To actuate the solenoid valves, a 24 VDC pulse is required. However, the onboard regulator on the Arduino Mega takes in 12 VDC from the supply Vin and outputs a 5V peak max clock signal. One classic solution to sourcing a higher peak clock signal is to use a small signal transistor amplifier. A 2N7000 small signal metal oxide semiconductor field effect transistor (MOSFET) was used in common source amplification configuration. In the common source amplifier topology, the output clock voltage had to be amplified to 12 V peak to feed to the onboard solid-state relay (SSR). In the common source topology, the output voltage is given as a function of the input voltage by Equation (2), neglecting transistor non-ideal effects.

(2)

In the above equation, the small signal parameter *gm* is the forward transconductance of the 2N7000 transistor, which can be found in its respective datasheet. *RD* is the resistance connected to the drain pin of the 2N7000 transistor and had to be carefully selected to obtain the correct output voltage. Knowing that the desired output voltage had to be 12 V peak, solving for *RD* resulted in a drain resistance of 51 kΩ.

The next amplification stage was at the output of the solid-state relay (SSR), which was a 741 operational amplifier in non-inverting configuration; the purpose of using non-inverting configuration was to amplify the clock signal while avoiding an unnecessary phase shift in the output voltage. Assuming an ideal op amp in non-inverting configuration, the output voltage of the 741 op amp as a function of the input voltage is given by Equation (3).

(3)

In Equation (3), because the input voltage was 12 VDC, it had to be doubled to meet the required output voltage of 24 VDC. Based on the formula in Equation (3), the resistance ratio of the feedback resistance (*Rf*) to the input resistance (*Ri*) must be one for the input voltage to be doubled. For practical reasons, two 10 kΩ were selected as the output and input resistances. However, to command the op amp to amplify to only 24 VDC, the positive and negative rails of the op amp had to be connected to 24 VDC from main supply and ground, or zero volts, respectively.

To test the circuit, an arbitrary clock signal was programmed on the Arduino using Appendix 1. The frequency of the clock signal could be adjusted by the formula in Equation (4).

(4)

In Equation (4), *f* is the frequency of the clock signal from the Arduino digital GPIO, and *T* is the period, or clock delay in milliseconds, that can be manually programmed into the Arduino IDE.

**Datasheets explained**

Once a hard-wired control circuit was devised and tested, a datasheet was made of the circuit by the team. The datasheet was made for the case of only one unit of the control circuit wired in series with a solenoid valve. The datasheet of the control circuit was devised for easy and fast assembly. In developing the circuit however, datasheets for the 2N7000 transistor, 741 operational amplifier, and solid state relay had to be used. The datasheets were used for the pin-out of each component, as well as intrinsic properties such as rated operating voltage, gain, and transconductance/input impedance. The datasheets for the control circuit’s constituent components are found in the datasheets section of this report.

**Summary of how far we got**

During the semester, the primary focus was designing a hard-wired control system that could switch on a hydraulic valve. Once the circuit was tested for the correct output voltage to the valve, it would then be used in conjunction with a 24 VDC power supply to turn the solenoid valves. The surf rover electrical team managed to prototype a hard-wired control system used for one valve on a breadboard and test its output voltages. Each component was tested for their correct output voltages. First, the linear regulator and SSR were tested to ensure that they were outputting the correct voltages of 12 VDC. Next, an oscilloscope was used to observe the input/output signals of each amplification stage in the circuit; the transistor amplifier was tested see if its output clock signal was 12 V peak and the operational amplifier IC was tested to see if its output signal was a 24 VDC pulse. Though this was unnecessary, a digital multimeter was used to ensure that the output voltages were at appropriate root mean square (RMS) values. Due to university closures from unexpected circumstances, there was no prototyping and testing of the surf rover wireless communications system. However, to emulate bits that would be received from the wireless communications system, a 5 V peak clock signal was programmed into the GPIO of the Arduino Mega board and the amplification of the clock pulse to the output was studied.

**Parts List and Vendor Contact Information**

Although the solenoid valves and 24 VDC supply were in house in the surf rover shop, the Arduino Mega, the brain of the surf rover, had to be purchased from Arduino company on Amazon due to a defect in the Arduino Mega that was previously in house. Solid state relays also had to be purchased, because only 5 VDC rated electromechanical relays were in house. Minor components, such as LM317 regulators, small signal MOSFETS, prototyping breadboard, wire, and resistors were kept in house. Table 1 shows the list of billable parts for the Spring 2020 semester

Table 1: Parts list for Spring 2020 Semester.

|  |  |  |
| --- | --- | --- |
| **Part Name** | **Vendor** | **Vendor Contact Info** |
| Arduino Mega AtMega 2560 Board | Arduino | Email: support@arduino.cc |
| Arduino USB A to B Programming Cable | Arduino | Email: support@arduino.cc |
| PVT 312 Solid State Relays | Infineon | Phone: +49 89 234 65555Support Page: <https://www.infineon.com/cms/en/about-infineon/company/contacts/support/> |
| RF Tx/Rx Pairs ASK 433 MHz | HiLetGo | Email: [sales@hiletgo.com](mailto:sales@hiletgo.com)  Phone:+86-0755-36625387 |

**Conclusions, Recommendations, Next Steps**

The next steps involved in the electronic design process would first be to connect the test circuit to the waterproof connectors of the solenoid valve and energize the valve to click; this indicates that the solenoid valve successfully received the correct actuating signal. After testing the hard-wired prototype circuit, the next step would be to prototype a wireless communications system. A 433 MHz amplitude shift keying (ASK) transmitter and receiver pair were selected and ordered from Amazon due to their compatibility with open source hardware such as Arduino and transmitting distance of 200 meters. However, the transmitter and receiver in each pair will require its own Arduino board. In short, one Arduino board would be required to be a wireless controller; an Arduino nano board could work for such a task. The Aduino Mega, used to provide the actuating signals to the solenoid valves, should be used as the receiver, as the extensive GPIO pins on the Arduino Mega make it a viable option for such a task. Once prototyped, the serial window on the Arduino IDE can be used to ensure that the transmitter and receiver are indeed able to communicate with one another. It is recommended that before testing, communications are set at the standard rate of 9600 baud (serial communication timing for the Arduino). This is done by entering the Arduino command *Serial.begin(9600)* in the setup function of the microcontroller. Once the transmitter and receiver can communicate with one another, the next step would be to send bits from transmitter, to receiver, and then to the hard-wired system to convert bits sent by the transmitter as actuating signals for the surf rover. This process is done to replace the hard-wired clock signal from the Arduino. Once the ASK communications system is developed, another option that should be explored is a backup communications system if the ASK system fails.

**Next Steps (Continued....)**

Once the Arduino wireless system works with the hard-wired control system, the next step would be to integrate steering algorithms referenced by the senior design report (this being provided for the project). The reason for this is to ensure that the valves, mounted on the rover, can be steered by the Arduino RC system (after implementation of a transmitter and receiver, serial communication). Only after this stage should a PCB be designed and milled to accommodate ten hydraulic valves. The PCB design must have the necessary Rx and Tx implementation before milling. Of course, coordination should occur between the diesel engine group, hydraulics group, and valve control group to ensure that the electronic system is properly insulated and mounted in the appropriate area on the surf rover.

One major recommendation that was brought up at the end of the semester was possibly stepping down the supply voltage to 12 VDC and using an onboard boost DC-DC converter to step up 12 VDC voltages to 24 VDC where necessary. It is also possible that the MOSFET is not needed in the design, and a 5 VDC clock signal could be used with only one amplifying stage due to the wide range of input voltages allowed by the SSR.

**Appendix 1.** This appendix contains the source code for generating clock signals from the Arduino MEGA GPIO pins.

/\*Surf Rover Electrical Squad Valve Test Source Code

\* Written by Patrick Borgman, Jaimie Thomas, Harun Resulovic, and Austin McCright

\* University of North Florida School of Engineering

\* Spring 2020 Semester

\*/

//Declare GPIO pin. In this case, Digital pin 5 was selected. Pin is named pump

//Since actuating signals will go the valve

const int pump = 5;

void setup() {

//Enable selected output pin pump. What this means is that this pin will

//output the clock signal.

pinMode(pump,OUTPUT);

}//end setup

void loop() {

/\*In the looping process, the exact same procedure is done for blinking an LED.

\* This function will continuously write to the digital output pin logic HIGH, delay then Logic

\* LOW, delay. The continuation of this process over time is what forms the square wave clock signal.

\* The time delay, by default in milliseconds, is what affects the frequency of the clock signal.

\*/

digitalWrite(pin,HIGH);

delay(200); //delay(200)--> wait for 200 ms then return to zero.

digitalWrite(pin,LOW);

delay(200);

}//end loop

**Datasheets**

[1] Infineon Technologies, “PVT 312 6-Pin DIP Solid State Relays,” *Infineon* PVT 312 SSR Datasheet, Revised May 15, 2015.

[2] Motorola, “2N7000 Enhancement Mode N-Channel MOSFET”, *Motorola, Inc.* 2N7000 Small Signal MOSFET, Revised 1997.

[3] Texas Instruments, “µA741 General Purpose Operational Amplifier,” *Texas Instruments Inc.*, UA741 Operational Amplifier, Revised 2018.

[4] ON Semiconductor, “LM317 Adjustable Output, Positive Voltage Regulator,” *ON Semiconductor,* LM317 Linear Voltage Regulator, Revised 2019.

[5] National Semiconductor “LM1577/LM2577 Series SIMPLE SWITCHER Step-Up Voltage Regulator,” *National Semiconductor,* LM1577/LM2577 DC-DC Boost Converter, Revised 2019.