FINAL REPORT

Downhole Electro-Hydraulic Control System

Schlumberger 5

Swapnil Dubey, Aboud Jamal, Jong Woo Ha,

Joshua Harden, Taylor Wegelin, Scott Hodes, Kexin You

4 May 2020

Introduction

Our goal for this project was to build and design electro-hydraulic control systems. With this control system, a user can actuate a hydraulic tool and receive feedback from pressure sensors. The control system will contain a user interface, control electronics, a brushless DC motor, hydraulic pump and a hydraulic reservoir. The hardware, electronics, and user interface will be contained in a compact package for ease of transportation. Raspberry Pi 3B+ was used to build our user interface upon, due to its fast response time. Brushless DC motor was used for its long lifespan and high torque at low speeds. Two Ashcroft GV pressure transducers due to its capability to read up to 5000 psi and also being cost efficient. The design will be used by Schlumberger's manufacturing and new product development teams to assess the reliability of their motor, pressure sensors, and downhole hydraulic tools. Our design for this project is not only cheaper than others available on the market, but also more efficient, easier to use and collect data. This project was a perfect fit for the team, since the focus in selected majors were applied directly to build and design the project.

Design Approach

Our approach was to divide the project into three subgroups: mechanical, electrical, and user interface. This allowed us to delegate different responsibilities within those groups to further divide up the tasks. The main idea for the mechanical group was to design physical parts for the pump manifold and housing for the hydraulic system. The electrical group divided their tasks into three subsystems that all connected to the microcontroller as shown in Figure 1.

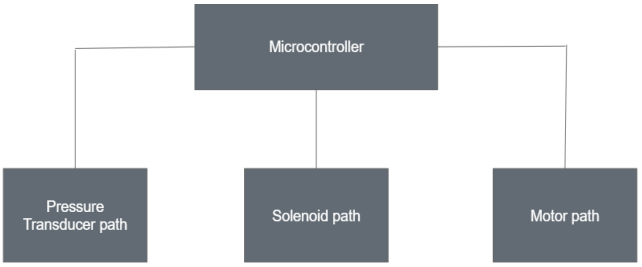

Figure 1. Electrical Block Diagram

The electrical team would construct the control electronics on a breadboard to be able to debug and work on the design, then implement a PCB for the final product. Finally, the user interface team will make a platform capable of being integrated so the user can adjust, motor rotation, motor speed, motor count, motor power, pressure rating, and solenoid power.

Implementation

Hydraulic System

The sponsor provided a basic hydraulic diagram for actuating a hydraulic cylinder. This basic diagram, shown in Figure 2, served as the guide for designing the hydraulic system.

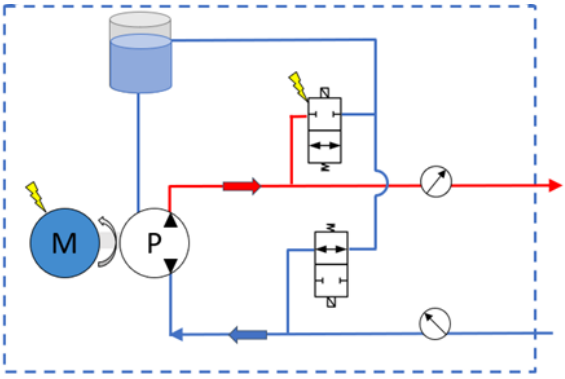

Figure 2. Basic Hydraulic Diagram Provided by the Sponsor

Since the hydraulic system is meant to perform a simple physical function, the majority of the system could be constructed with commercial off-the-shelf (COTS) parts as long as they meet the proper specifications. These include a minimum 5000 psi rating, resistance to corrosion, removal filters with a maximum pore size of 10 microns, 1% accuracy pressure transducers and the provided motor and pump from the sponsor.

While most of the system can be constructed with COTS parts, it was identified that a custom manifold for the pump to connect to the rest of the system would need to be designed since no commercial version existed. This manifold design would have to connect to the pump, keep the pump secure, and have some way of accepting COTS fittings for connecting to the rest of the hydraulic system. The sponsor provided a concept for the pump manifold and is shown in Figure 3.

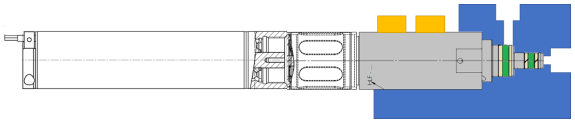

Figure 3. Pump Manifold Concept Provided by the Sponsor

The pump manifold and hydraulic system parts selection and schematic were developed in parallel. This was made possible by dividing the hydraulic system into various subsystems and deciding on a tube size to use as a standard. Figure 4 shows a basic hydraulic diagram broken up into its subsystems. Since the pump and motor are designed to be submerged in hydraulic fluid, the reservoir was incorporated into the container holding the motor and pump rather than be a separate component. The pressure subsystem consists of both the pressure relief valves and pressure transducers while the other subsystems consist of the parts in their title.

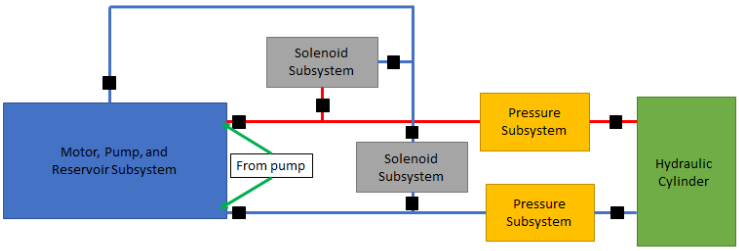

Figure 4. Updated Hydraulic Diagram Showing Each Subsystem and Filter Locations

The pump manifold model was made using SolidWorks and contained the same features as the concept provided by the sponsor. The outlet ports were dimensioned to accept SAE 5/16”-24 threads to accept SAE fittings for ⅛” tubing. The port for the pump to go into was dimensioned to have a 0.002” gap between the manifold walls and the pump’s max material condition with a tolerance of 0.002” to ensure no material overlap between the manifold and pump. Figure 5 shows a couple views of the pump manifold while Figure 6 shows a

couple assembled views of the pump manifold with the pump and shaft collars. A simple set of calculations was then run to determine if the friction force from the shaft collars would be enough to hold the pump in.

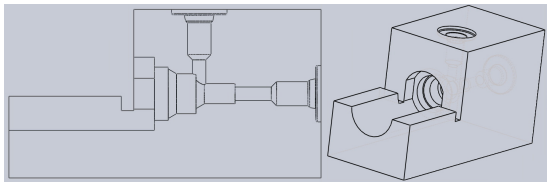

Figure 5. Pump Manifold

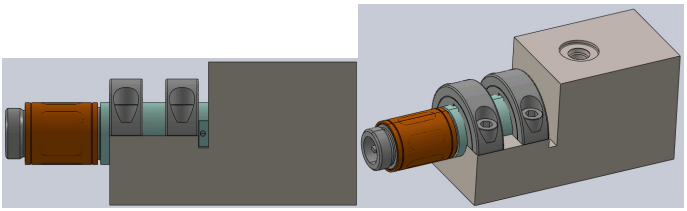

Figure 6. Assembled Pump Manifold

Once the pump manifold was designed, a detailed hydraulic schematic was created with part, part source, and part number. The detailed schematic is shown in Figure 7.

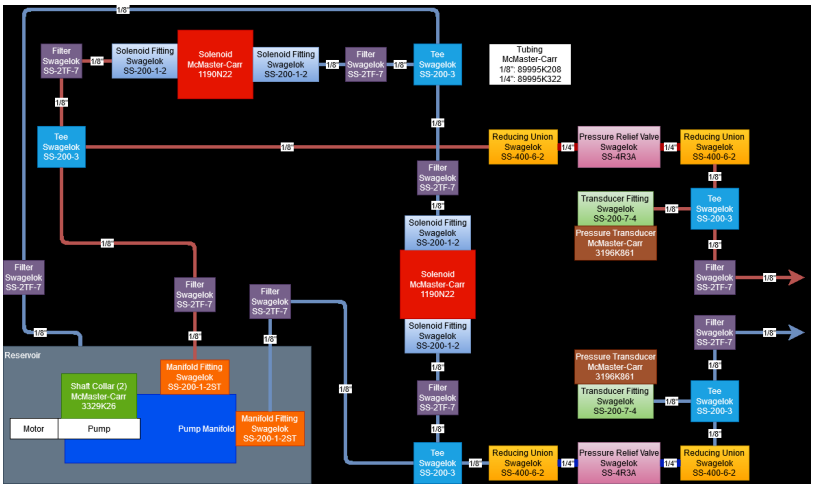

Figure 7. Detailed Hydraulic Schematic Showing Each Part by Name, Manufacturer, and Part Number

Pressure Transducer Subsystem

A design requirement given for this project was after providing output pressure, to be able to measure the pressure feedback from whatever the design was attached to. In order to measure the pressure in both hydraulic lines two Ashcroft GV pressure transducers were chosen. These transducers were chosen due to their capability to read up to 5000 psi (a design requirement) and due to their relatively low cost of $128.85 per unit. Additionally, the pressure transducers had a range of input voltages of 14-36V DC and thus could be powered safely by the 24 V power supply used for the solenoids. In order for the Raspberry Pi to be able to read the analog output of the transducer, an ADS1015 analog to digital converter was purchased. This ADC communicates with the Raspberry Pi through I2C with Pi's SDA1 (Serial Data) pin, and this data transfer is

synchronized via the Pi's SCL1 (serial clock) pin. The output would be on the level of 4-20 milliamps however the ADC takes readings via voltage. In order to turn this current into a voltage reading, it was run across a 250Ω resistor, which would vary the voltage across the resistor 1-5V based on the pressure in the transducer. Two OP-27 op amps were wired in voltage follower configuration and used as buffers that would allow the ADC to read the voltage without affecting the flow of current. The electrical connection for the pressure transducer is shown in Figure 8.

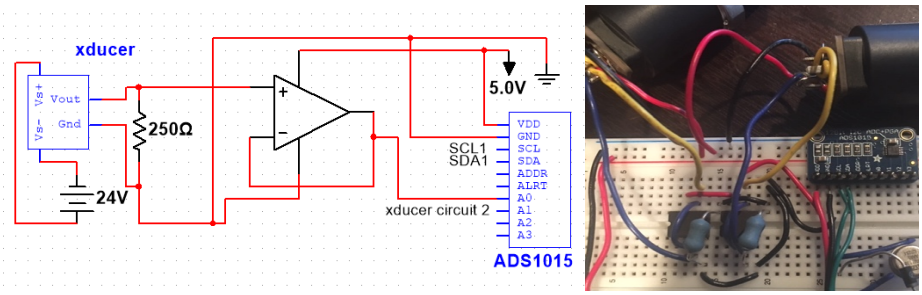

Figure 8. Schematic and Physical Realization of Pressure Transducer

Motor Subsystem

One of the major differences between a brushed and brushless DC motor is the axel's being in contact with our circuit which is a possible point of failure as is shown in Figure 9. A brushless motor solves that problem but unfortunately are more expensive and have a more complex controlling method.

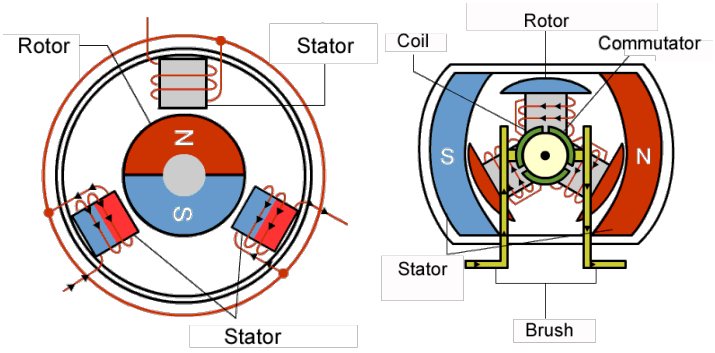

Figure 9. Brushless DC Motor (Left), Brushed DC Motor (Right)

The brushless DC motor for this application came with an Electronic Speed Control (ESC) device. Controlling an ESC will in turn control the motor by providing the required waveforms for the motor. The ESC can be

powered by a 5 to 24 V input with a signal input being a Pulse-Width Modulated (PWM) wave with a particular duty cycle to get a particular response.

Typically, the input duty cycle range is from 10% to 20% for a 50Hz wave but that was not true for the motor for this project. To get an estimate of the duty cycle that works for the particular motor, write an incremental loop with duty cycle starting with 0% up to 100%, while printing the duty cycle value on the screen with a delay of at least 1 second between each new value. The first value when the motor begins to rotate or twitch some is most likely going to be near its central duty cycle value.

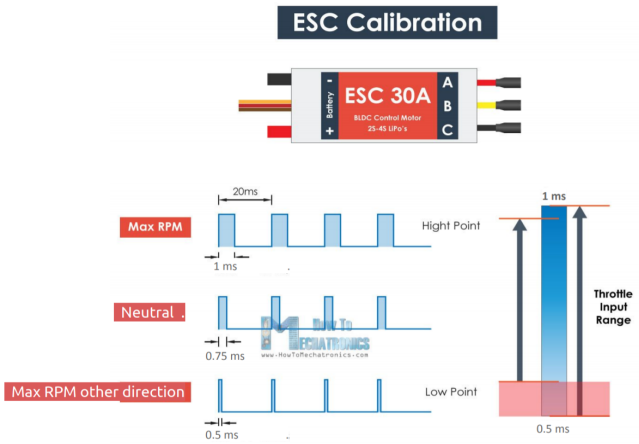

Figure 10. Electronic Speed Control Input Signal's Duty Cycle Range

The ESC had to have a startup "calibration" sequence before it was primed to take in commands consistently. The ESC in this project works within the range 5% to 10% duty cycle as shown in Figure 10. A 5% duty cycle sends a signal to run at maximum RPM in 1 direction and at 10% for the opposite direction and 7.5% is the neutral or stop point. The calibration sequence is to go from maximum RPM, then delay at least 1 second, then minimum RPM, or neutral point. A beep will confirm that the ESC is calibrated and ready to take inputs.

Initially the team had decided to try and use back EMF to measure the motor's RPM since it lacked Hall Effect sensors that are typically used to measure a brushless motor's speed. Due to lack of equipment, it was finally decided to use a tachometer to create a linear relationship with the duty cycle and display the motor's speed to

the user. The motor is able to achieve 4000 RPM as its maximum and 1500 RPM as its minimum rotation speed in either direction.

Solenoid Subsystem

The solenoid valve is used to release pressure from the hydraulic line. A specification of the solenoid was provided by the sponsor. The operating voltage of the solenoid, from the power supply was constantly supplied to the solenoid at 24 VDC. The solenoid had a continuous duty cycle with an operating current of 1.7A. To control the solenoid, the IRF130 NMOS transistor was used as a switch. A drain current of this transistor was 8A, which would be sufficient to turn on the solenoid. In order to turn on the NMOS, voltage above the gate's threshold voltage must be supplied to it, this was done via a 5V step-up controlled by GPIO pins. Then, the drain current would be supplied to the solenoid to operate. To turn off the NMOS switch, 0V was supplied to the gate terminal which would cause the gate to have the same voltage as the source, which was grounded. A diode was added parallel across the solenoid terminals to eliminate flyback.

When testing the solenoid, some voltage was measured at the gate terminal of the NMOS transistor even when the digital output of the Pi was 0V. In order to have the gate to source voltage to be zero, a 2.5kΩ pull-down resistor was added between the gate terminal and the ground where the source gate was connected to. The electrical connection of the solenoid is shown in Figure 11.

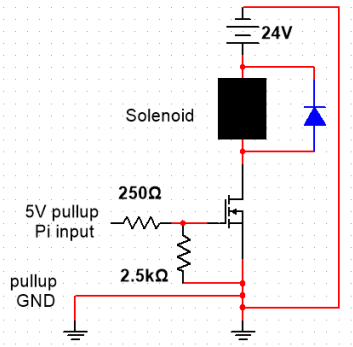

Figure 11. Solenoid Circuit Schematic

User Interface

According to the requirements initially provided by Schlumberger, the user interface should contain 7 user inputs and 6 outputs. There should be two switches to open and to close the solenoid toggle, two switches to control the motor and its direction, two for desired pressure and motor speed, and one button to reset the motor rotation count. Users will be able to type in the exact number they want for pressure and speed. The UI will send signals to the electrical parts and alter the pressure or speed, then the parts will give feedback to the UI and display. The 6 outputs are mainly designed for data display purposes. To find the most compatible environment, three packages were compared and two of them were used in the design phase. Kivy, Tkinter and PySimpleGui were considered and compared. Kivy is better suited for touch interfaces and games and it's much more complicated to implement. Tkinter is desktop focused and easy to design. This project requires a simple, user-friendly interface and packaging and debugging need to be completed in a short time. While PySimpleGui has variables and functions separated, adding GPIO inside the loop would be complicated since the variables will be increased or decreased by more than one function. So Tkinter was finalized. To breakdown the code and to analyze the approach, a logic flow chart was first completed as shown in Figure 12.

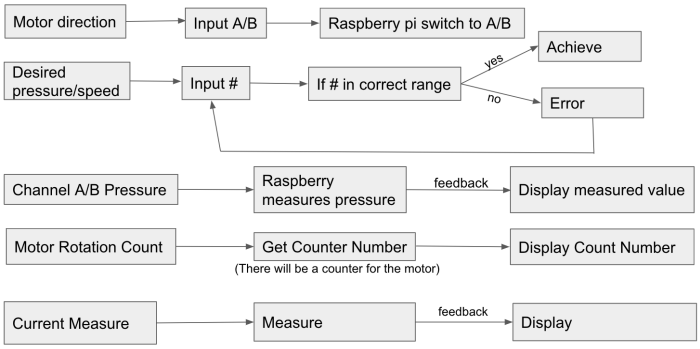

Figure 12. Logic Flow Chart

The first draft of Tkinter was completed and they fulfilled most of the requirements. Figure 13 shows the codes for one of the buttons. First is to create a label which displays the status of the solenoid. This tells users if the

solenoid is open. The text on this label is set to a variable which varies depending on the hit command of the button. Figure 14 shows the final version of the code.

```
sa = tk.Label(window,
    textvariable = var,
    bg = "LightPink", font = ("Arial, 14"), width = 20, height = 3, fg = "White")
sa.grid(row = 0, padx = 3, pady = 0.5)
on_hit = False
def hit():
  global on_hit
  if on_hit == False:
    on_hit = True
    var.set('Solenoid A is Open')
    GPIO.output(26, 0)
  else:
    on_hit = False
    var.set('Solenoid A is Close')
    GPIO.output(26, 1)
b1 = tk.Button(window,
    text = "Solenoid A Open/Close",
    width = 20, height = 2,
    command = hit
    )
b1. grid(row = 0, column = 1, padx = 3, pady =2 )
```

Figure 13. Tkinter Code

The final version of the user interface is shown in Figure 15. The pink label parts are buttons. The text on the label will change if buttons are clicked. The two buttons on the bottom right change the motor rotation count and the count number will display on the blue label area. While the purple labels are safety warnings. The user can type in the desired pressure and current value and hit the ok button. If the inputs are larger than 5000, there will be a popup window and remind users to reenter.

```
def press():
  val = float(e1.get())
  if val < 5000 :
    tk.messagebox.showinfo(title ='success', message = 'Maximum is successfuly set')
  else:
    tk.messagebox.showerror(title = 'warning', message = 'Invalid, Please reenter')
max = tk.Label(window, text ='Maximum Pressure Setting', bg = 'Thistle',
  width = 30, height = 3, fg = 'White')
max.grid(row = 0, column = 3, padx = 5, pady = 2, sticky = tk.E)
e1 = tk.Entry(window, width = 10)
e1.grid(row = 1, column = 3, padx = 1, pady = 2, sticky = tk.E)
b5 = tk.Button(window, text= 'OK', command = press)
```

Figure 14. Popup Logic

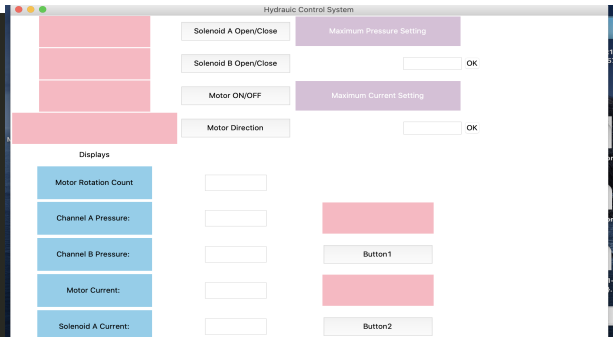

Figure 15. UI Final Version

Results

Mechanical

Since the hydraulic system was never constructed physically, it is unknown how well this system would perform or how much troubleshooting would be required to get the system working. No detailed analysis was conducted on the pump manifold model to determine if it meets the minimum pressure specification of 5000 psi. The calculations for the shaft collar friction force on the pump come out to 1,209.6 lb. while the max

pressure force comes out to 870.8 lb. These calculations suggest the shaft collars are more than sufficient even at the highest pressures, but having no physical prototypes means these calculations can't be verified by experimentation.

Electrical and UI

The Pressure transducer circuit and the UI's ability to display PSI was unable to be tested due to the inability to reach facilities at ARL. The motor was able to be calibrated, change its rotation of spin, increase its speed, decrease its speed, and turn off via inputs given to the UI. However, this is not entirely consistent. The ESC has a problem during certain test runs where it does not calibrate correctly and rings its calibration tone during the direction of spin input (not during the calibration phase). During other runs the ESC has a problem controlling its speed in the counterclockwise direction. While the current actual RPM of the motor was not able to be displayed on the UI, the RPM instruction given to the ESC is able to be displayed. The problems with the ESC could be rectified with more research on ESCs created for RC car use, however these calibration and speed control problem are not expected to be an issue when this system is implemented by the project's sponsor as they will be using a non-RC car intended motor and ESC that will include a detailed data sheet that will allow them to avoid such issues. Each solenoid is able to be closed and opened independently. However, when the solenoids are left powered for long amounts of time, they become hot to the touch and begin to lose effectiveness. This was not rectified due to the late point in the timeline of this project this portion of the design was wired. To fix this issue or design around it, more research, testing, and troubleshooting should be done with the solenoids.

Summary

In conclusion, given the unforeseen challenges, our team developed the foundation for a downhole electro-hydraulic control system. The mechanical team laid down the framework to implement a physical pump manifold. This will allow Schlumberger to directly take their research, drawings, and part selections to develop

a physical part. The electrical team was able to design and construct the control electronics for the system using similar “copycat” components, to allow an easier integration for Schlumberger with the pre-selected parts. Due to not receiving the actual parts from Schlumberger, we were not able to order a PCB, but the sponsor will be able to easily design one with the research and wiring design we have done. Lastly, the user interface was able to meet all the requirements from the sponsor, making this a very user-friendly system for Schlumberger’s manufacturing team.