**Lou Auto Robotics Research Institute:**

**Tactile Robot Skin**

Daniel Nguyen

MRI Project

April 2021

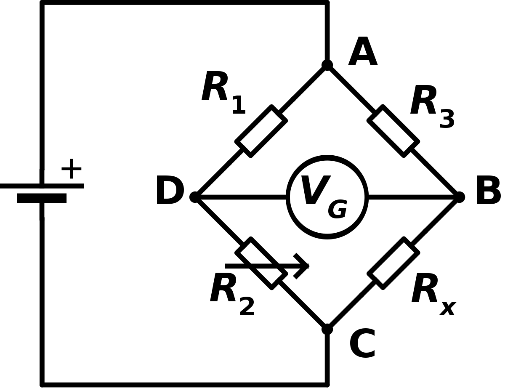
# Appendix

1. Introduction & Objectives
2. Major Activities
   1. Hardware
   2. Software
3. Conclusion

# Introduction & Objectives

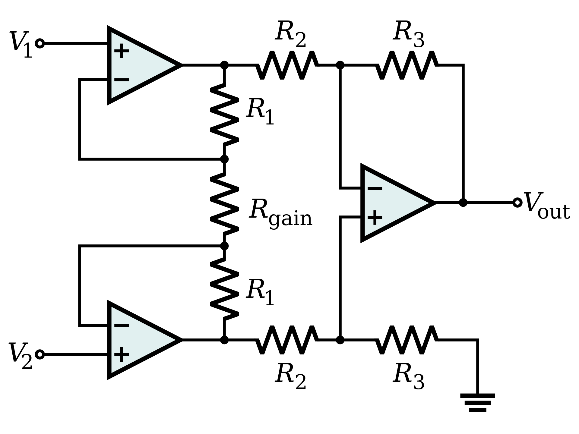
The development of tactile robot skin is crucial for the next generation of robots to interact among human-centered environments. At the Louisville Automation & Robotics Research Institute (LARRI) under the MRI Project, researched have developed electronic skin using multiple sensors and components. The skin can sense changes in pressure using a lever hinge system. The main disadvantage of this system is the narrow output range; thus, the output must be amplified adding overall noise.

Measuring the resistive sensor uses a system including the Wheatstone bridge, an instrument amplifier (in-amp), and an analog-digital-convertor (ADC). Figure 1 illustrates an example of the Wheatstone bridge.



**Figure 1: Wheatstone Bridge**

According to figure 1, with 3 known resistor values, the fourth resistor value can be solved. A similar system is implemented for the skin sensor demo. The skin sensor project uses a DAC and an instrument amplifier. Figure 2 is a reference of the instrument amplifier.



**Figure 2: Instrument Amplifier**

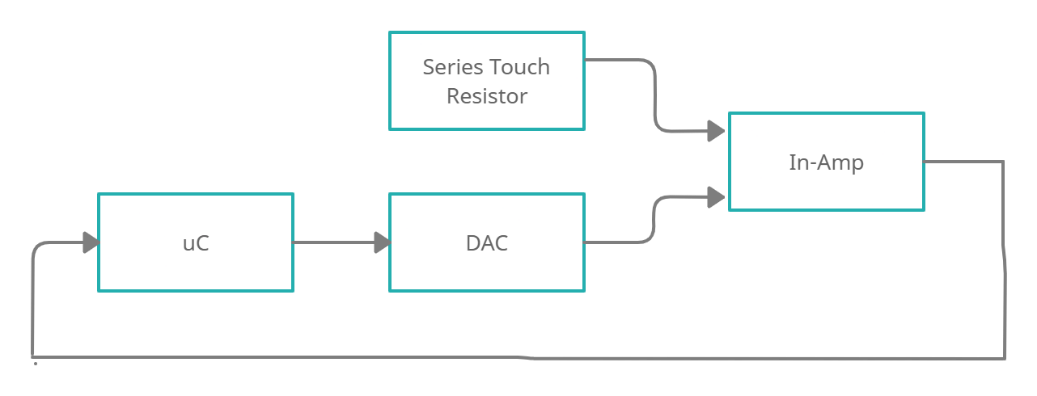
The base equation for an instrument amplifier is:

Through calibration in the software and the DAC, is set to a constant value within a small range very close to . is connected to the touch resistor. Thus, any changes in the resistor results in a change in voltage in being amplified to . Gain is set to a high value because the resistance change of the sensor is small. Under brief testing, a gain of 200 is found to work the best.

# Major Activities

### Hardware

Figure 3 shows a block diagram of the circuit.

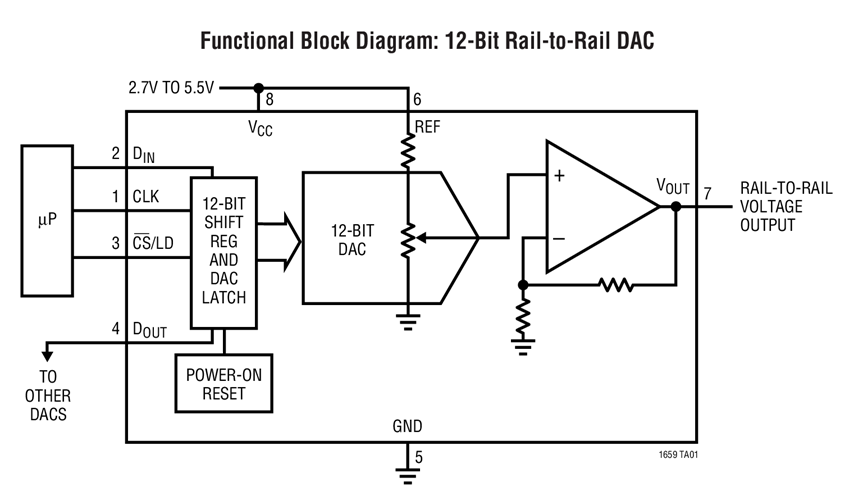


**Figure 3: Block Diagram**

As shown in figure 3, the voltage difference measured in the in-amp is the key in measuring the resistance change. The analog-to-digital converter (ADC) of the micro controller and the DAC are 12 bits. The two components are also referenced to 3 volts - GND to limit the amount of current needed to be drawn from the micro controller.

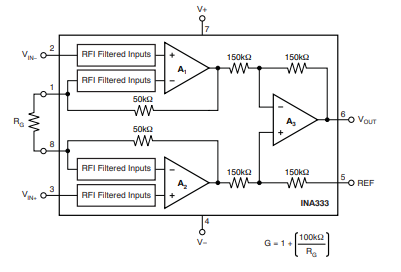
The current design uses a dsPIC33EP32GP502 micro controller, the LTC1659 DAC, and the INA333 instrument amplifier. These components were picked based on how they complement each other as well as the accuracy. The current Micro-Controller Unit (MCU) is used because of its many features such as serial communication busses, flash memory size, and speed.

The LTC1659 DAC is used specifically because it has the same resolution as the micro controller. Because of this, no conversion is needed when sending adjusting the DAC after reading values from the 12-bit ADC of the micro controller during calibration. Figure 4 illustrates the internal structure of the DAC.



**Figure 4: LTC1659 Internal Structure**

During the first testing phase, the instrument amplifier used was the INAx126. Because the device was not rail-to-rail, it was not accurate enough. Because the resistance change of the touch resistor has a very small range, this paired with a less accurate in-amp did not work. For the first prototype, the In-Amp picked is the INA333. It is a rail-to-rail instrument amplifier and is rather accurate. Figure 5 illustrates the internal structure of the INA333.



**Figure 5: INA333 Internal Structure**

Another small board is designed for the MCU to communicate with the computer. This includes the FT231X Breakout UART to USB. The programming board also allows for the PICkit 3 to connect to the main board.

### Software

Two communication busses are used in the MCU. Those are SPI and UART. The MCU communicates with the DAC through the SPI Bus, and with a computer through the UART communication bus. At boot-up of the PCB, there is a delay for the voltages to settle. After this, the software calibrates for the first time.

The software constantly checks the UART Receive register. If it receives a certain keystroke, the software will either recalibrate, stop reading the sensor value, or continue reading the sensor value. A snippet of the code is shown below:

cmd = UART1\_RX\_NB();

switch(cmd) {

case '1': **//Calibrates & reads sensor**

runCalibrate = true;

calibrate();

dacIncrement = 200;

runSensor = true;

break;

case '2': **//Continue sensor reading**

runSensor = true;

break;

case '3': **//Stop sensor reading**

runSensor = false;

break;

}

The calibration algorithm went through multiple test phases. During the first prototype, a step algorithm was put in place. This will increase or decrease the value the DAC outputs by a small margin to match the idle voltage of the touch resistor. Because the resolution of the ADC and DAC are quite large, the step algorithm was considered too slow and the implementation of a PID controller was tested.

The PID controller was not as fast in practice as in theory because the referenced voltage of the ADC is too small. As mentioned earlier, all the components are referenced from 3 Volts to Ground. If the DAC is too large or small, the ADC reading will read the output as either 0 or the max resolution, 4095 for a 12-bit ADC.

The final algorithm put in place is like a Bang controller. Initially, the DAC will swing a large value until it crosses the setpoint range. Once it crosses the setpoint range, the constant value that the DAC swings is decreased by 50% until it crosses the setpoint again. Through testing, this algorithm has been proven too be significantly faster. A snippet of the code structure is shown below:

while (runCalibrate) {

ADC1Val(); **//Updates sensVal (ADC reading)**

**//if the sensor reading is above/below setpoint, change DAC output**

if(sensVal <= lowBound) {

dacVal -= dacIncrement;

flag = 0; **//DAC IS BELOW ADC READING**

} else if (sensVal >= upperBound) {

dacVal += dacIncrement;

flag = 1; **//DAC IS ABOVE ADC READING**

}

SPI\_transfer16(dacVal);

ADC1Val();

**//Checks if ADC reading is within range**

if((sensVal >= lowBound) && (sensVal <= upperBound)) {

runCalibrate = false; // BREAK WHILE LOOP

} else if (dacIncrement == 0) {

dacIncrement += 1;

}

**//Checks if DAC crossed the setpoint range. If so, decrease the increment.**

if((sensVal <= lowBound) && (flag == 1)) {

dacIncrement = (uint16\_t)(dacIncrement / 2);

} else if ((sensVal >= upperBound) && (flag == 0)) {

dacIncrement = (uint16\_t)(dacIncrement / 2);

}

ADC1Val();

printf("Calibrating... %u\n\r", sensVal);

}

}

# Conclusion

This project serves multiple purposes such as a tactile sensor that can be used as electronic robotic skin and a board used to test the Pick-And-Place machine at the LARRI lab. The PCB is designed with the components spaced out to test the accuracy of the new machine. This Sensor Board Demo also contributes to other areas in the MRI Project such as the Octo-Can through the software such as the calibration algorithm.