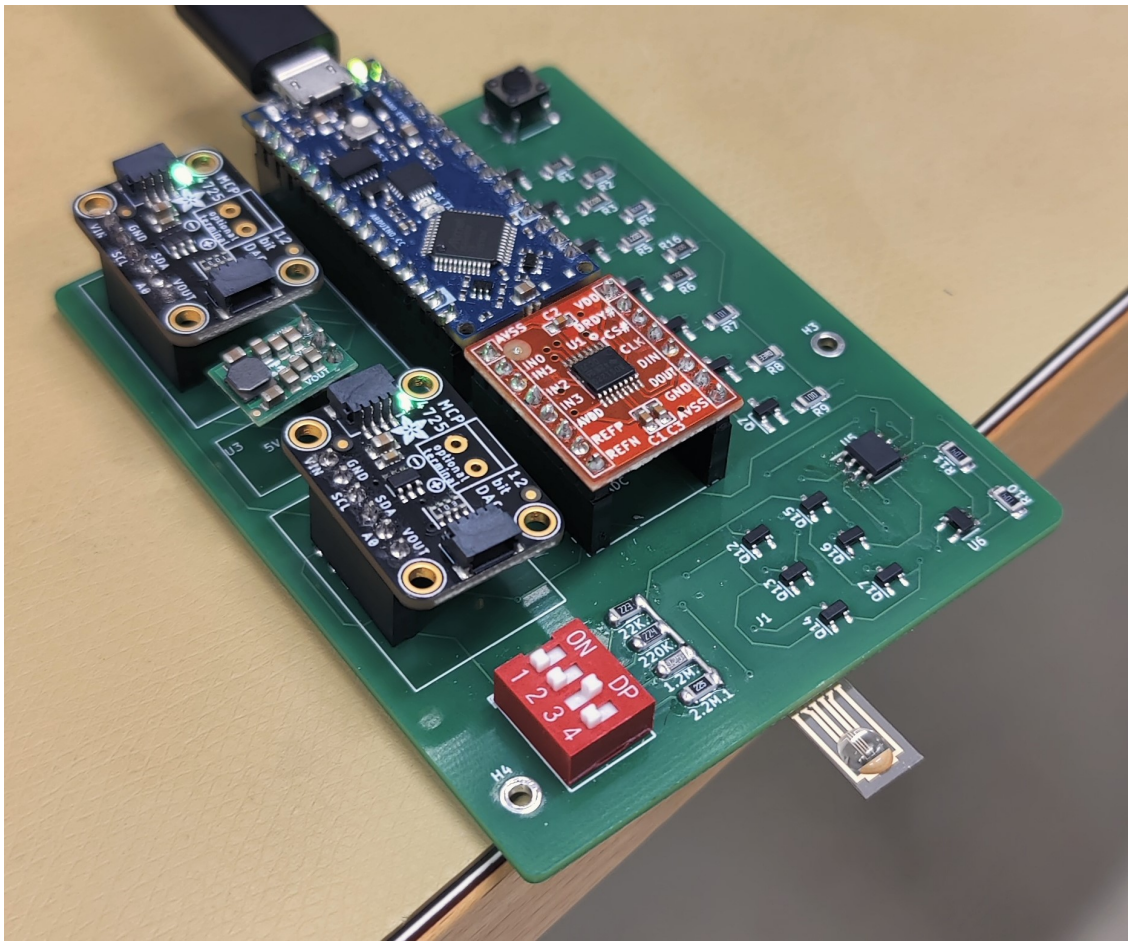
CHALMERS



Portable Electronics For Graphene Biosensor

Degree Project Report in Electrical Engineering

JIMMY BREVITZ
VINCENT OLOFSSON

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

Degree Project Report 2024

Portable Electronics For Graphene Biosensor

JIMMY BREVITZ
VINCENT OLOFSSON

CHALMERS

Department of Electrical Engineering
Division of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2024

Controlling circuits via an microcontroller
Measuring resistance over a transistor
Signal conversions

Supervisor: Avgust Yurgens, Department of Microtechnology and Nanoscience
Examiner: Rob Maaskant, Department of Electrical Engineering

Degree Project Report 2024
Department of Electrical Engineering
Division of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Complete system while measuring a G-FET with a drop of sterile water on it. Photo taken in lab at Chalmers campus Johanneberg, Gothenburg.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2024

Abstract

This report details the planning, design, testing, and execution of an electrical system capable of detecting the presence of Listeria bacteria on a Graphene Field Effect Transistor (G-FET). The system includes an Arduino as the micro controller, components for signal conversion, amplifiers, transistors, resistors, and a micro SD card connector for connecting the G-FET. This system is designed in KiCad and manufactured into a PCB for ease of use and potential commercial distribution. To perform a test, the system conducts a gate-voltage sweep, handles signal conversions, controls various circuit components, and measures the resistance of the G-FETs. The results are then transmitted via a USB cable to a PC and uploaded to an external website for data processing.

Keywords: Sensor, Detection, Graphene Field Effect Transistor, Microcontroller, Signal conversion, Printed Circuit Board, Data processing

Acknowledgements

We thank Professor Yurgens for supervising the project. Professor Yurgens provided a well equipped lab space and knowledge during the work. We were also lent a book on small signal measurement which increased our understanding of the problems and solutions related to sensors. Our sincere appreciation goes to Mr Khan who produced the G-FET's that was tested in the system, and explained how they and graphene in general worked.

We thank our examiner Professor Maaskant for the input on our report and for the course on sensors he taught in our second year. It set the foundation for the knowledge used in this project. We want also to show appreciation for Professor Hawke that quickly replied to questions related to report structure, as well as teaching courses in writing and presentation.

We wish to show our gratitude towards the leaders of LayerLogic: Sebastian Samuelsson, Ebba Sandbecker, and André Persson. Thank you for trusting us to do this project and for your professional way of team leading.

Last but not least, thank you to Senior Lecture Heikkilä for the course on transistors and being a big part of the structure of the 180hp Electrical Engineering program at Chalmers. We deeply feel that the courses taught throughout the three years at the program could not prepare us better for the field.

Jimmy Brevitz and Vincent Olofsson, Gothenburg, June 2024

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

Vgs	Voltage between the gate and source pin of a transistor
Vds	Voltage between the drain and source pin of a transistor
G-FET	Field Effect Transistor with Graphene
PCB	Printed Circuit Board

Contents

List of Figures

List of Tables

1
Introduction

Sensors are all around us in today's society. They make up for an important part in technology, from a door that opens automatically when a person is approaching, to a machine that stops operating due to safety concerns. Various sensors utilise different methods to activate. In this report, we'll detail the design of the electronics connected to a resistive sensor. This type of sensor relies on changes in resistance within the hardware to initiate activation.

1.1 Purpose

The fish industry is currently battling a problem with slow response time in regard to bacteria testing. The testing itself is very trustworthy, but the procedure includes sending away the test to an off-site lab and waiting a couple of days to get back the results. In case of a positive test response, the fish that might be contaminated has already been sent out to shops and restaurants. The company will call back the product but the potential health danger for consumers is already apparent. Other then bad PR and potential fines for the company, it also results in a negative impact on the environment since unnecessary deliveries has to be made.

LayerLogic aims to develop a sensor that can perform the same bacteria test on-site and give a result within minutes. This solution would make it possibly for the fish producers to avoid sending away possible infected fish while waiting for a test result. The test will be dependent on the relatively new material graphene. The company is cooperating with Professor Avgust Yurgens and PhD student Munis Khan at the Department of Microtechnology and Nanoscience of Chalmers University of Technology, who are researchers in the field of graphene. The sensor will also serve a purpose for this research, and thus will have a couple of functions dedicated to research purposes and not for testing in the fish industry.

1.2 Objectives

The sensor utilizes a transistor with graphene, causing variations in its electrical conductivity when the gate voltage changes. Professor Avgust Yurgens and PhD student Munis Khan fabricated two Field-Effect transistors with graphene, fitted into an SD-card. These transistors form the active part of the sensor. By designing an electrical circuit with suitable software, it is possible to determine whether bacteria are in contact with the graphene.

The electronics will be made up of an electrical circuit which is programmed and controlled with the micro controller Arduino. The circuit measures the resistance of one or two G-FETs while changing the voltage between the G-FET's gate and source, Vgs, from -0.5 to +0.5 V, and plots the resistance versus Vgs. A bacteria test sample will be put on one of the two G-FET's and then the test process begins. The G-FET's resistance as a function of Vgs will be plotted. Any deviation of the sensor resistance from the baseline curve indicates a positive test result.

The G-FET themselves can have an internal resistance of 200-200k ohms, depending on the length and width of the graphene used in the transistor [1]. To ensure the correct functionality of the circuit without damaging the graphene due to large currents, the current needs to be adjustable. This is done by having 4 resistors of different values connected via a switch. This way, it is possible to change the current to fit the characteristics of the G-FET. Lastly, the measurements should be sent from the sensor to a PC with an USB cable, and the system should be small enough to carry in your hand.

1.3 Delimitations

The initial delimitation posed two significant challenges. Firstly, due to the Arduino's 5 V supply limitation, the range of compatible components for the circuit was severely restricted. Additionally, the inability to supply negative voltage further narrowed down the options for components.

Furthermore, time constraints presented another notable limitation. The project timeline was relatively short, necessitating the purchase of pre-designed circuits for certain functions instead of developing them in-house. While this approach expedited the process, it also contributed to increased expenses in the overall system implementation.

2
Theoretical Framework

A general overview of the system can be seen in Figure 2.1. The purpose of the system is to measure the voltage drop across the transistor, the G-FET, while the micro controller supplies two different variable voltages to the drain and gate pins. The voltage drop is then amplified, with the gain dependent on the variable resistance R2, which is also controlled by the micro controller. R1 is a variable resistance used to adjust the current as needed.

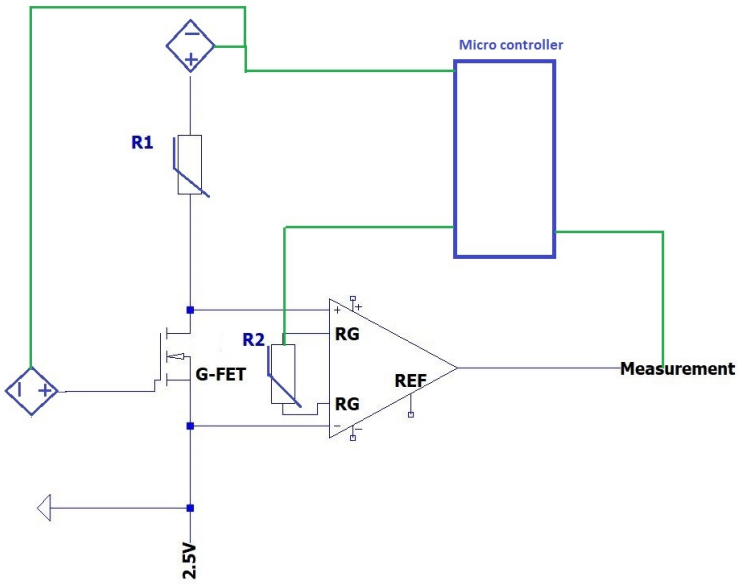


Figure 2.1: *General overview of the system. The green wires are control signals.*

2.1 Graphene and G-FET

Graphene is a single atom thick sheet of carbon atoms aligned in a honeycomb lattice [2]. Graphene is considered to be a 2-D material [1] and has a lot of interesting properties. Most of them are not to be mentioned here hence they are not used in the report, but one of them is its very high conductivity [3]. The conductivity changes when graphene come into contact with certain bacteria, such as Listeria [2] and that is what makes graphene a suitable material for this project. The G-FET is produced in a way to fit an SD-card connector, and as seen in Figure 2.2, each

pin of the G-FET is connected to a leg on the SD-card. The design varies a bit depending on whether one or two G-FETs are fitted into the SD-card.

Figure 2.2: *Two G-FET's fitted on a SD-card carrier*

The pins on the SD-card carrier in Fig 2.2, from left to right, are:

1. Drain pin of G-FET number one.
2. Measurement point + of G-FET number one.
3. Measurement point - of G-FET number one.
4. Source pin, shared by G-FET number one and two.
5. Measurement point + of G-FET number one.
6. Measurement point - of G-FET number two.
7. Drain pin of G-FET number two.
8. Gate pin, shared by G-FET number one and two.

Unlike other Field Effect Transistors, the drain and source pins of a G-FET always allow electron flow, ensuring that current passes regardless of the Vgs. However, the internal resistance of the G-FET changes significantly with different Vgs values. How this is used is described later in the report.

2.2 Arduino micro-controller

An Arduino is a micro-controller designed to control other units and circuits. It consists of programmable input and output peripherals and one or more processor cores. The micro controller in this project is an "Arduino nano every" and will hereafter be referred to only as "Arduino". It generally controls one or more processes with code written in C++. In this project, the standard "high/low" input and output pins are used. This means that the process is controlled only with the micro controller sending or receiving 5 or 0 volts. What makes Arduino special is the vast sea of functions with code that can be found in the Arduino libraries [4]. This makes it very efficient to implement different functions.

2.3 Description of Electrical Components

To overcome the Arduino's voltage limitation of only supplying 0 or 5 V, a Digital to Analog Converter (DAC) is incorporated. This device enables the supply of voltage ranging from 0 to 5 V with a precision of 1,22 mV. Operating with a 5 V supply, the DAC interfaces with the Arduino via the I2C protocol, receiving a 12-bit binary string. Subsequently, the DAC converts this binary string into a numerical value within the 0-5 V range and delivers this voltage to the output pin. Conversely, for analog-to-digital conversion, an Analog to Digital Converter (ADC) is employed. The ADC accepts an analog signal ranging from 0.5-2.5 V, converting it into a 24-bit binary string with a 0.24 µV precision that accurately represents the received signal. This digital representation is then transmitted to the Arduino through the SPI protocol for further processing.

In this project, an operational amplifier functions as a voltage follower, which prevents current from flowing between two points in the circuit. To amplify the voltage drop across the G-FET, an instrumention amplifier is utilized. Unlike an operational amplifier, an instrument amplifier boasts a higher Common-Mode Rejection Ratio (CMRR), which enhances precision when comparing signals on the inputs.

The step-up converter elevates voltage magnitude from a lower value to a higher one. In this project it is used to ensure a 5V output from the Arduino This transformation is achieved through a circuit, seen in Figure 2.3, comprising an inductor, capacitor, diode, and a rapid switch. During the switch's on phase, energy from the input source charges the inductor. Upon switch-off, the inductor discharges its stored energy. Guided by the diode, current flows solely towards the output, augmenting the voltage magnitude.

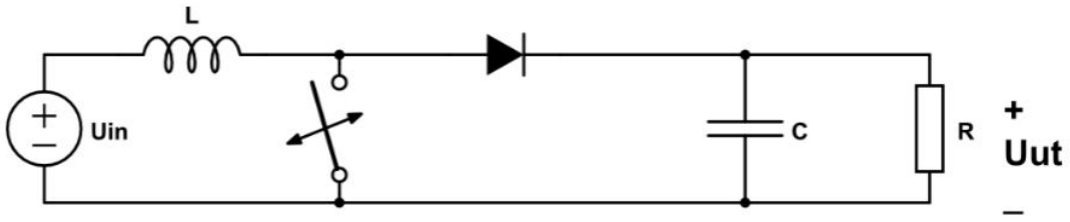


Figure 2.3: *Schematic of a step-up converter.*

2.4 Signal Processing and Communication

In this project, three communication methods are employed: UART (Universal Asynchronous Receiver/Transmitter), SPI (Serial Peripheral Interface), and I2C (Inter-Integrated Circuit).SPI and I2C are synchronous communication protocols, employing either one or two data lines in addition to a clock signal line. This

synchronization ensures that the sender and receiver are aligned during data transmission. Contrary to these two, UART is asynchronous and require a predefined frequency for synchronized communication between the sender and receiver.

Signal processing is vital for integrating hardware and software within an embedded system. This is achieved through the Arduino, DAC, and ADC. Although the Arduino does not directly perform signal processing, it serves as a communication conduit with the DAC and ADC, facilitating data transfer between the hardware and software layers of the system.

2.5 Prototype Circuit

Visualised in Figure 2.4, the G-FETs drain pin will be supplied with 5 V from one of the DACs via the variable series resistance R1 and the other DAC will supply the gate pin. The source is connected to a constant voltage follower of 2.5 V, serving as the system's reference point. During measurements, the DAC on the gate will adjust the voltage from 2.5 V up to 5 V, then down to 0 V, and finally back to 2.5 V. This process causes the Vgs voltage to range from 0 V, up to 2.5 V, down to -2.5 V, and back to 0 V. During this voltage sweep, the voltage drop across the G-FET is amplified by the instrumentation amplifier and converted by the ADC. The gain is determined by the variable resistance R2, controlled by the Arduino. The reference point is established through the a step-up converter ensuring a precise 5 V output, and a voltage divider with a voltage follower.

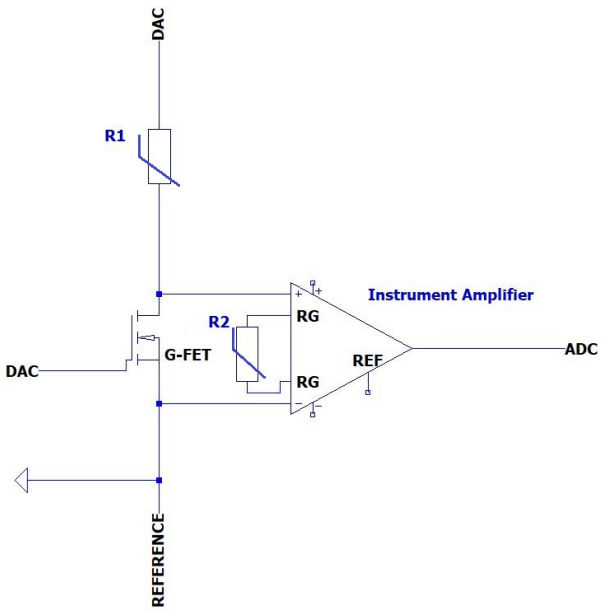


Figure 2.4: *The first prototype circuit. The reference point is 2.5V in comparison to ground.*

3
Method and System Overview

For the creation of a working sensor, the following steps where followed:

1. Design circuit and look for adequate components.
2. Simulate using LTspice.
3. Test components.
4. Write code.
5. Build and test circuit.
6. Calibrate and test system.
7. Design PCB, calibrate and test system with PCB.
8. Final testing.

This is a standard approach when designing an embedded system. However, the project requirements evolved during the process, necessitating modifications to both the software and hardware to meet these new demands. To save time, these changes were tested directly on a breadboard with hardware instead of conducting additional simulations in LTspice. Consequently, the original circuit design and simulations are not fully comparable to the final result and lack some functions.

3.1 Components

For the two DACs, the pre-designed MCP4725 circuit was chosen. To avoid the time-consuming process of designing and testing a custom conversion circuit. The MCP4725 communicates with the Arduino via I2C and has a 12-bit resolution.

The same rationale was applied when selecting the ADC, the ADS1220. This 24-bit ADC communicates using SPI, operates at 2000 samples per second, and supports a supply voltage range of 2.6-5.5V, fitting the system's specifications. It also features two channels, allowing it to function as an ADC in two separate circuits.

The LM324N operational amplifier was selected for its suitable minimum single-supply voltage of 3 V and its integration of three separate op-amps within a single component, enabling the connection of additional op-amps if required. As for the instrument amplifier, the INA129PA was chosen for its low offset voltage of 50 µV, single supply voltage of 4.8 V, and maximum gain capability of 10000.

To be able to connect a G-FET transistor to the circuit a Molex micro SD card connector was selected. The two G-FET are produced to fit inside a micro SD-card

for easy connection to the hardware.

Simulating the first circuit shown in Figure 2.4 was straightforward. An appropriate gain was selected, and the outputs from the instrumention amplifiers were measured. Subsequently, the difference should be calculated inside the software.

3.2 Test of Components

Four component tests were of particular interest: the operational amplifier, instrumentation amplifier, DAC, and ADC. The functionality of the op-amp and instrumentation amplifier was assessed using a circuit depicted in Figure 3.1.

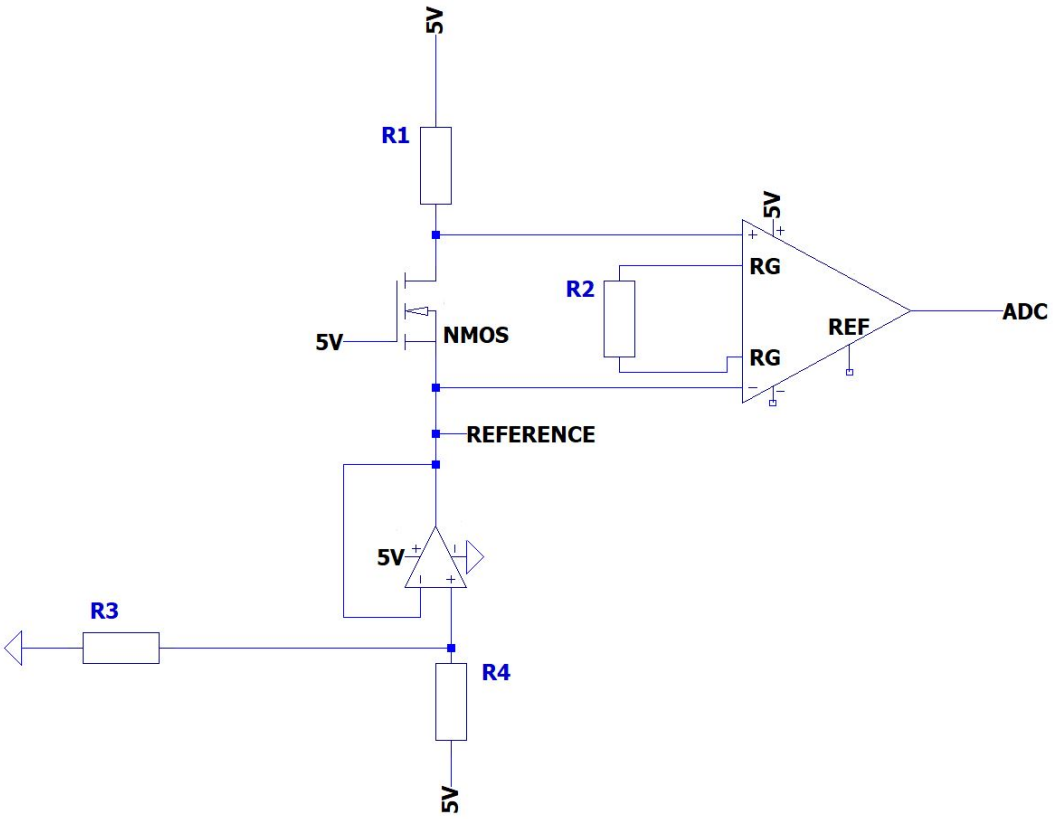


Figure 3.1: *The schematic of the circuit used when testing the amplifiers. R1=820k, R2=50, R3=R4=100k.*

The purpose of testing the instrumentation amplifier was to determine the maximum and minimum voltage drop the component could operate with and identify where the output linearity ends. This was crucial because the linearity of the equation G*Insignal declines when the output from the instrumentation amplifier begins to approach the positive or negative voltage supply, a phenomenon known as satura-

tion. In this project, the supply is single-sided, meaning the negative voltage supply is grounded, and the positive supply is 5 V. During the test, R1 was adjusted until both ends of saturation were identified, and the linear operating range was determined to be between 0.3-2.2 V.

The tests conducted on the DAC and ADC primarily aimed to ensure that communication from the Arduino was executed correctly. While the DAC functioned as intended, the ADC exhibited some unexpected behavior. Although there were no communication errors, the digital signal sent from the ADC was incorrect. Additionally, the input range was not 0-5 V as expected, but rather 0.5-2.5 V. LayerLogic was informed about this issue and stated that if it could be resolved without purchasing a new component, then it should be addressed accordingly. Despite this discrepancy, this ADC was used because the values it sent exhibited linearity in comparison to the analog signal it received. As a result, the following equation (3.1) was derived from measurements on the ADC to address the situation. The straight line equation (3.2) was used to derive the equation. Where x is (3.3), k = 1.008 and m = 452.

$$\frac{1.008 \times \text{data\_in} \times 2.048 \times 1000}{2^{23} - 1} + 452 \tag{3.1}$$

$$\text{kx} + \text{m} \tag{3.2}$$

$$\frac{\text{data\_in} \times 2.048 \times 1000}{2^{23} - 1} \tag{3.3}$$

3.3 Adaptation

Since the resistance in the G-FET changes with Vgs, the two hardware limitations significantly impacted the results, necessitating system adaptations. As previously mentioned, the linear range of the instrument amplifier is 0.3-2.2 V output, and the input range for the ADC is 0.5-2.5 V. This effectively restricts the instrument amplifier's output range to 0.5-2.2 V. If the gain remains static during the measurement, the system would fail if the G-FET's resistance changes by more than a factor of 4.4. Given that it is common for a G-FET to change resistance by a factor of 10, it was necessary to adjust the gain during measurements. This is done by the variable resistor, R2 in Figure 2.4. To be able to control the resistance from the Arduino, a resistor network was connected in parallel to the instrumentation amplifier's gain input. This circuit was can be seen in Appendix C.

Another adaptation was made for connecting the G-FETs . Since there can be up to two G-FETs in the SD-card connector, a circuit was designed to select which G-FET to measure. This circuit, also controlled by the Arduino, is shown in Appendix C.

3.4 Software

The software was coded with C+ in Arduino IDE. To create a successful software for an embedded system it is important to implement one function at the time to easily troubleshoot the system. The first function to implement was the communication from Arduino to the DAC and ADC. Both components had a downloadable library and a user manual [5] [6] with example code inside. The components were tested as described in 3.3 to verify that the communication worked.

The second function was to control the previously illustrated circuit, Fig C.1, for the instrumention amplifier. The adequate gain could be achieved using 7 different "if" statements. Different output pins from the Arduino became activated to achieve a certain gain. With these two functions implemented, it was possible to see if the sensor system could calculate the resistance in the transistor.

The third function was the voltage sweep on the gate pin of the transistor. This function used one DAC to produce the required voltage on the gate. After measuring one G-FET, the Arduino sends signals to transistors that close the connection between the first G-FET and the rest of the circuit. It then opens the connection to the second G-FET, initiating a new measurement cycle. The software can be activated by pressing a button or by sending a start protocol from a computer.

4
Results

This chapter will describe the finalised system, both hardware and software. It will also include verification tests and compare the results with a professional measurement system.

4.1 Function Description - Circuit

To understand the operation of the complete system, let's start with the voltage supply. The two DACs provide voltages at different points in the circuit. One DAC supplies the gate pin of the G-FET and performs the voltage sweep. The second DAC supplies a voltage between 0-5 V to the series resistance selected by the switch. In this project, 5 V is used, resulting in 2.5 V at Vds, but this can be adjusted in the software if needed. Since different GFETs operate with varying current levels, the first resistor ensures that the current remains within an acceptable range. Depending on the specific GFET being tested, researchers will use different initial resistances, as shown in Table 4.1.

R1 (Ω)	Current (μA)
22k	114
220k	11.4
1.2M	2.08
2.2M	1.14

Table 4.1: Current values when using each resistance connected to the DIL-switch when $V_{ds} = 2.5$ V.

The current flows through the G-FET, and onwards to the system's reference point, which is set at 2.5 V. This voltage also serves as the reference voltage in relation to the gate pin. It's important to note that the voltage sweep on the gate pin should start at 0 V, reach the maximum value and then go down to the minimum value. The reference point subtracts 2.5 V from the voltage supplied to the gate pin, allowing for a negative total voltage. This 2.5 V reference voltage is obtained through an op-amp configured as a voltage follower, utilizing the supply voltage from the Arduino and voltage division using two resistors.

A voltage drop occurs across the G-FET, and this signal is amplified through an instrumention amplifier. The measurements are taken at the drain and source pins of the G-FET. Since the source pin is connected to the reference point of 2.5 V, the instrumention amplifier effectively amplifies the voltage difference between the G-FET and the reference point.
As explained in Chapter 3.3, during the gate-voltage sweep, the resistance of the G-FET undergoes significant changes, necessitating adjustments to the amplification level of the instrumention amplifier. This adjustment is achieved by controlling the circuit described in Chapter 3.3, which involves opening and closing different transistors to introduce various resistances connected to the amplifier. The lowest amplification level of the amplifier was 6. The resistance was calculated using Equation (4.2).

$$G = 1 + \frac{49.4k\Omega}{R_g} \longleftrightarrow R_g = \frac{49.4k\Omega}{G-1} \tag{4.1}$$

The resistor closest to the calculated value was then soldered onto the PCB. To enable the measurement of two G-FETs via the SD-card connector, a series of transistors were connected to the SD-card connector, shown in the Apeednix B, Figure B.5. This arrangement allows the software to determine which of the two G-FETs will be measured.

4.2 Function Description - Software

The software program can be found in Appendix A.

4.3 KiCad and PCB

KiCad was utilized for creating the schematic and Gerber files for the PCB. Many of the components in this project were already available in KiCad, while some had to be custom-built. The KiCad schematic follows the block diagram shown in figure 4.1.

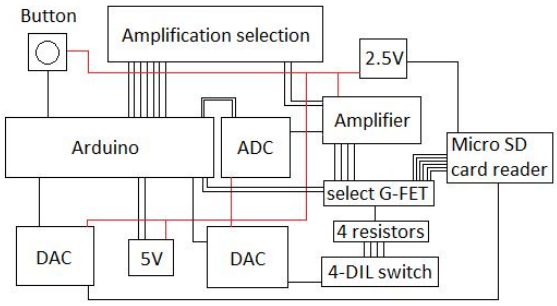


Figure 4.1: *A block diagram of the circuit.*

Figure 4.2 shows all the components in KiCad and how they are wired. Improved images of all the components can be found in Appendix B.

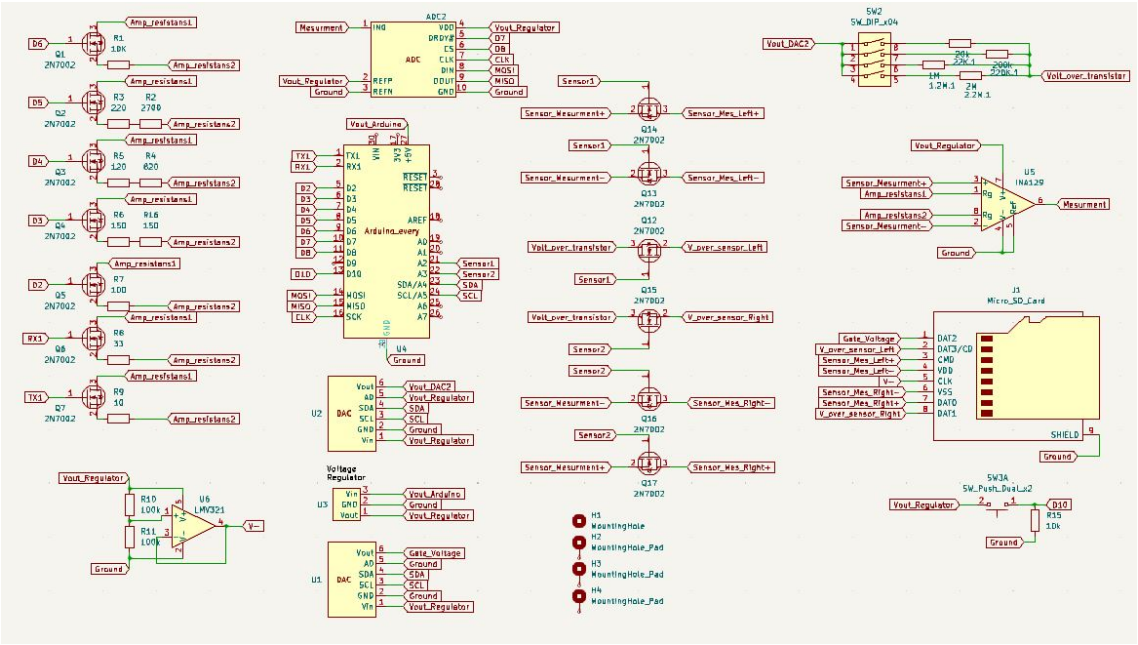


Figure 4.2: *All the components in KiCad.*

All components from the schematic must be manually wired in one of the programs in KiCad, as shown in 4.3. It is also in this program that the design of the PCB is carried out.

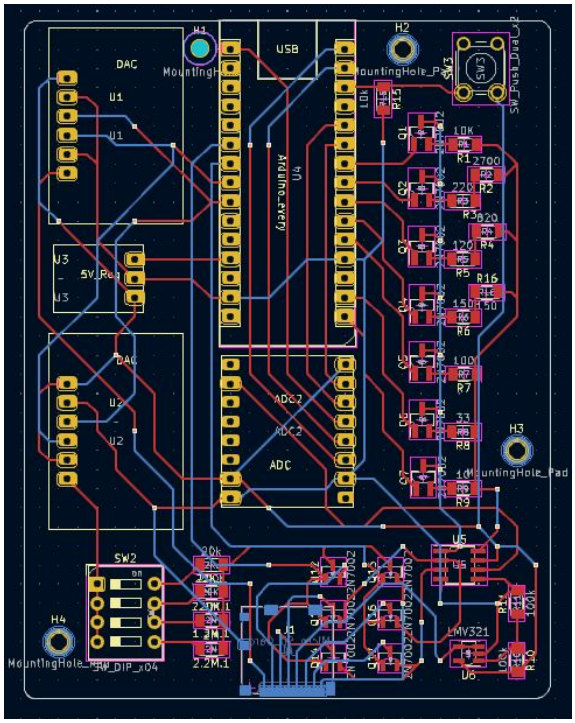


Figure 4.3: *The schematic of all the components and cables for the PCB.*

Figure 4.4 shows the finished PCB when all the components are soldered to it.

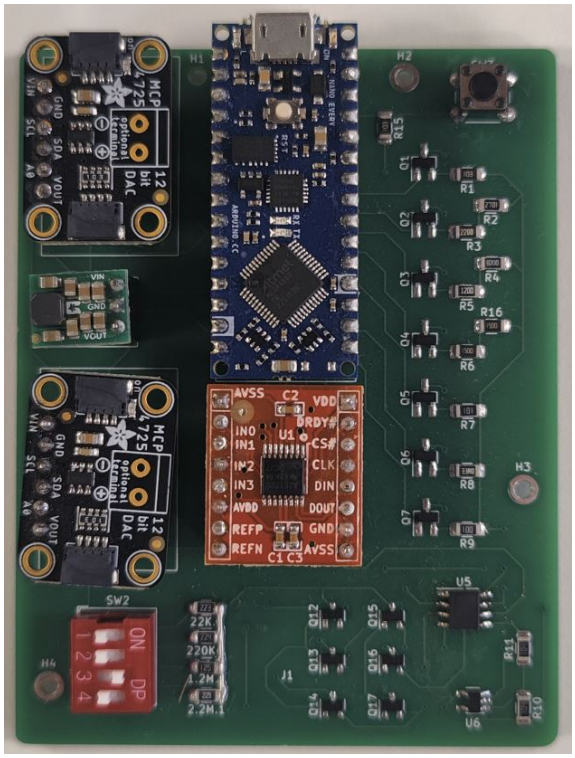


Figure 4.4: *The PCB with all components soldered to it.*

Figure 4.5 shows the finished PCB when all the components are soldered to it.

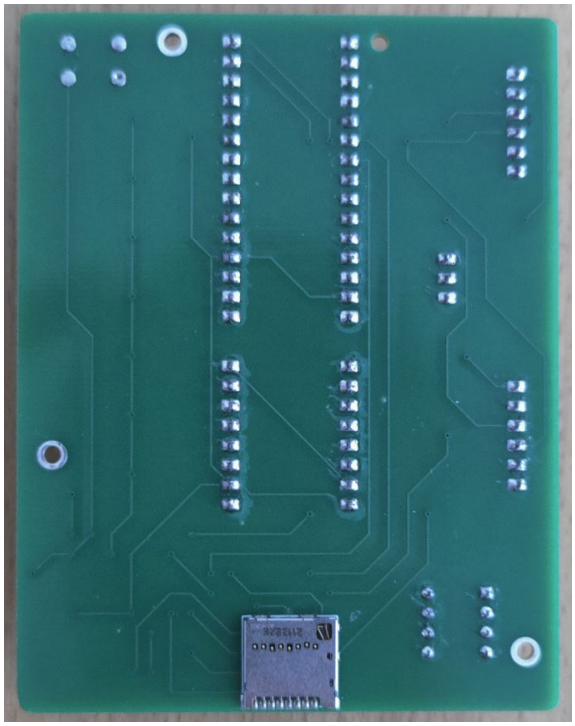


Figure 4.5: *Backside of the PCB with all components soldered to it.*

4.4 Tests

The first test conducted to verify the system involved measuring various resistors. Several different resistor values were tested and compared with measurements obtained using a multimeter to ensure that the deviation from the expected value was no more than ±5%. Through this test, the total possible resistance range for each of the 4 series resistances could be determined, as shown in Table 4.2.

R1 (Ω)	Min measurement of G-FET (Ω)	Max measurement of G-FET (Ω)
22k	3.22	3850
220k	31.8	38183
1.2M	174	More than 100k
2.2M	324	More than 200k

Table 4.2: Measurement range of each resistance when $V_{ds} = 2.5$ V.

An N-MOS transistor was measured to confirm that the voltage sweep over Vgs functioned as intended. The Vgs began at 1.5 V and increased by 0.012 V in each step, as illustrated in Figure 4.6. Initially, the resistance of the transistor remained very high until approximately 1.8 V at Vgs. Subsequently, the resistance dropped towards zero, validating that the voltage sweep functioned as expected.

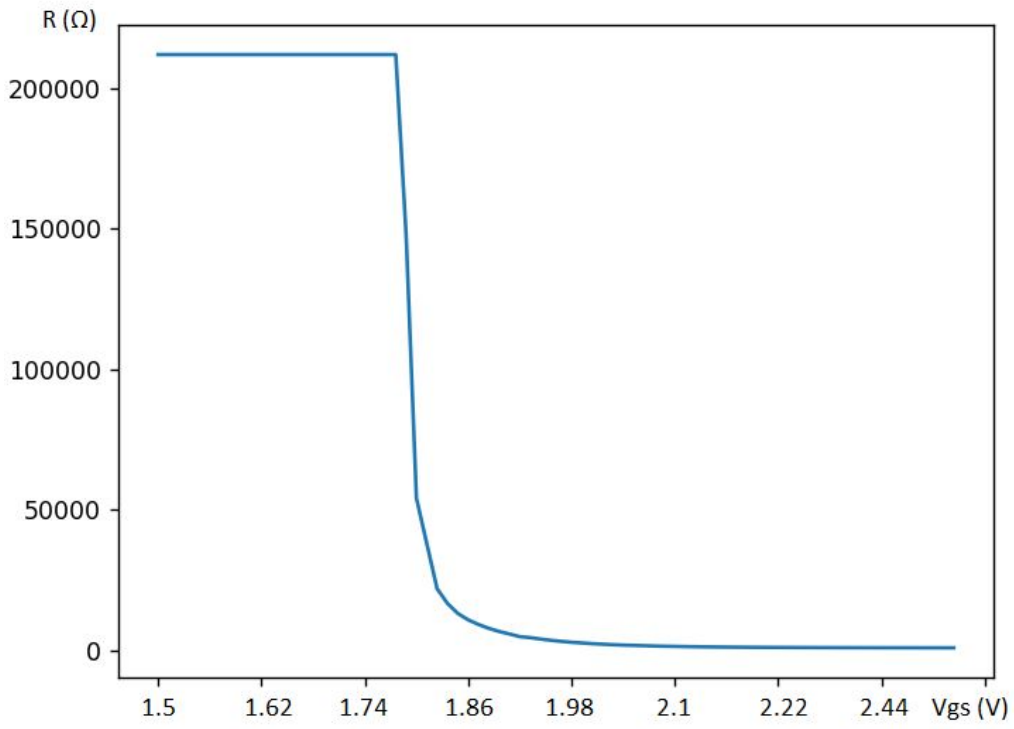


Figure 4.6: *Voltage sweep test over a N-MOS transistor.*

With these two tests confirming the system's readiness, the final test involved measuring a G-FET. This G-FET was measured using 2uA and the voltage sweep was done between -0.6 and +0.6 V with 0.012 V step size. The Vgs voltage starts at 0 V, increases to 0.6 V, then decreases to -0.6 V after finally returning to 0 V.
The graph, Figure 4.7, shows when water is applied to the graphene and connects the gate to the transistor and measurements are made.

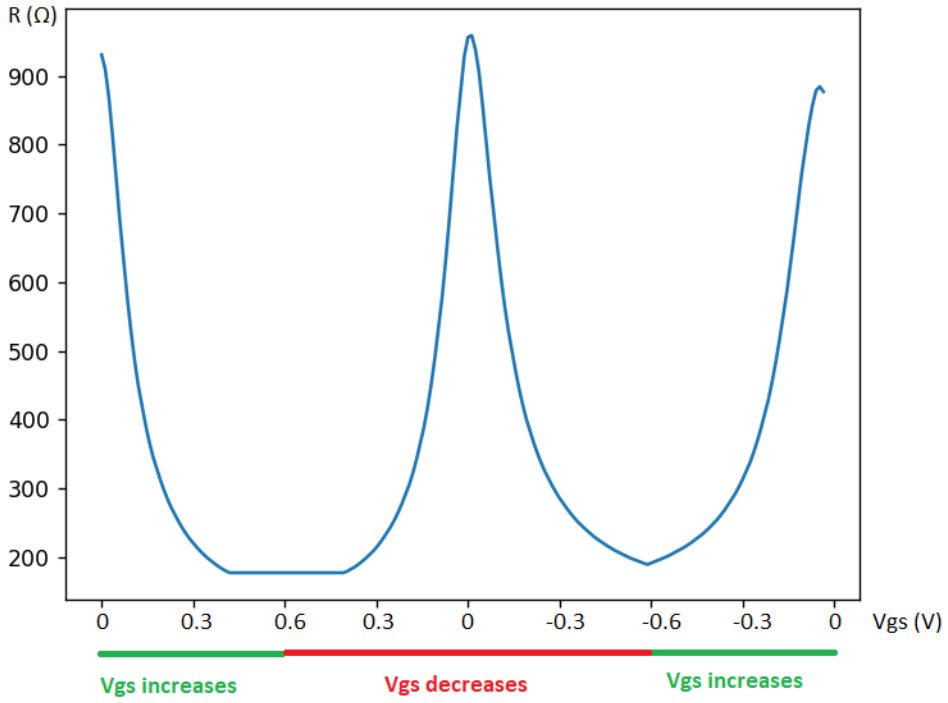


Figure 4.7: *The resistance of the G-FET during the voltage sweep when the gate pin is not connected.*

This test correlated well with Munis Khan's test on the graphene seen in Figure 4.8. This test was done with an Hewlett Packard 34401A Multimeter.

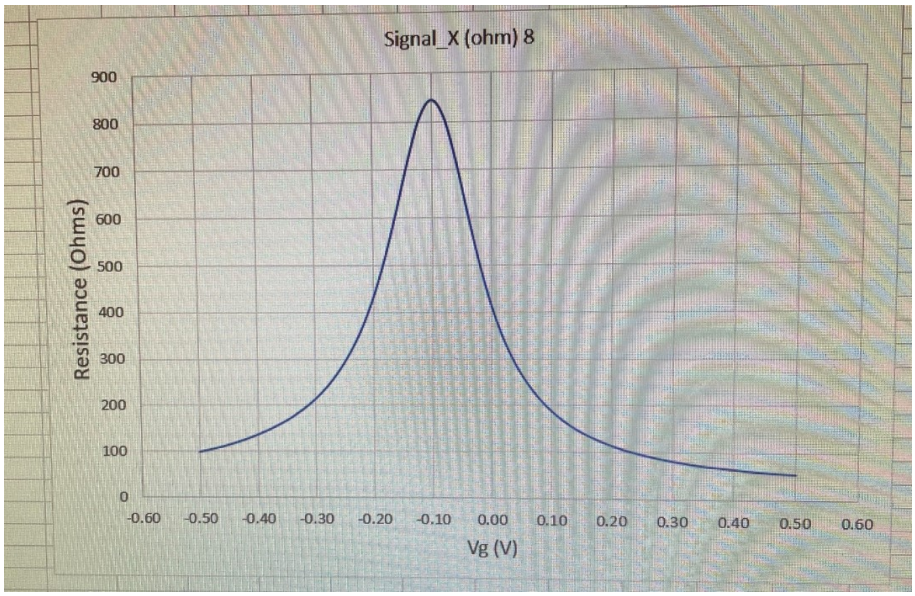


Figure 4.8: *A picture of Munis Khan's resistance test on the same G-FET.*

5

Conclusion and Discussion

The outcomes of this project successfully met the objectives, even though some parts of the system did not work perfectly. Overall, the system functions as intended, offering users the flexibility to adjust starting resistance, voltage output from the DACs, number of G-FETs to be measured, and measurement duration. This adaptability enables the system to accommodate various sizes of G-FETs.

The two most significant issues with the system were the instrumentation amplifier and the ADC. As described in Chapter 3.3, the instrumentation amplifier became saturated in a narrower range than anticipated, and the ADC could not measure signals below 0.5 V. Since these two problems were successfully worked around, no further action was taken to resolve them. However, these issues would need to be addressed more thoroughly if the project were to progress in the future.

Test results generally fell within the required $\pm 5\%$ range, with the exception of the G-FET test, which showed a deviation of $+7.8\%$ compared to measurements obtained with a Hewlett Packard 34401A Multimeter. However, this discrepancy was deemed acceptable by Munis Khan, possibly attributed to slight differences in measurement points or changes in Graphene characteristics.

One area for potential improvement in the project is the consideration of the low voltage output from the Arduino, which significantly limited the voltage range of the amplifier and necessitated the adaptations discussed in Chapter 3.3. This limitation could be addressed by implementing a step-up converter to increase the voltage in the system. Additionally, a closed-minded approach to circuit design from the outset may have hindered exploration of alternative solutions. In future projects, seeking inspiration from similar endeavors and learning from their experiences could prove beneficial.

Moving forward, there are opportunities for further enhancements. Minimizing noise in the system could be explored to improve measurement accuracy. While noise was acknowledged during the project, efforts to detect and mitigate it were not pursued due to the acceptable measurement range variance. Additionally, for commercialization beyond academia, modifications to the software are necessary to streamline user interaction. Setting predetermined values for resistances, voltages, and other variables would simplify operation for consumers and ensure consistent performance. This proactive approach would optimize the system for widespread use while maintaining its research capabilities.

From a sustainability perspective, the system's adaptability and flexibility contribute to its longevity and relevance. By enabling users to accommodate various G-FET sizes and measurement parameters, the system promotes efficiency and reduces the frequency of upgrades or replacements. Once consumerized, the product will eliminate the need for unnecessary transportation of infected fish and off-site testing, further enhancing its sustainability impact.

During the final revisions of the report, AI was utilized to enhance grammar, resulting in clearer and more concise text, thereby improving readability. This collaborative effort with technology refined the presentation, ensuring a polished and professional outcome. Incorporating this process earlier in the writing stage would have provided additional time to focus on the technical aspects of the project.

Bibliography

[1] L. Fusco, A. Gazzi, G. Peng, Y. C. Shin, S. Vranic, D. Bedognetti, F. Vitale, A. Yilmazer, X. Feng, B. Fadeel, C. Casiraghi, and L. G. Delogu, "Graphene and other 2D materials: a multidisciplinary analysis to uncover the hidden potential as cancer theranostics," Theranostics, vol. 10, no. 12, pp. 5435–5488, Apr. 2020.

[2] S. Pandit, M. Li, Y. Chen, S. Rahimi, V. Mokkapati, A. Merlo, A. Yurgens, and I. Mijakovic, "Graphene-Based Sensor for Detection of Bacterial Pathogens," Sensors, vol. 21, no. 23, p. 8085, 2021.

[3] E. W. Hill, A. Vijayaragahvan, and K. Novoselov, "Graphene sensors," IEEE Sensors J., vol. 11, no. 12, pp. 3161–3170, Dec. 2011.

[4] Arduino, "Arduino Libraries," `https://www.arduino.cc/reference/en/libraries/`, Accessed: 2024-04-21.

[5] Adafruit Industries, "MCP4725 12-Bit DAC Tutorial," `https://learn.adafruit.com/mcp4725-12-bit-dac-tutorial?view=all`, Accessed: 2024-04-21.

[6] OLIMEX, "OLIMEX ADS1220 Arduino Library," `https://www.electrokit.com/upload/product/41013/41013934/OLIMEX-ADS1220-arduino.zip`, Accessed: 2024-04-21.

A

Appendix - Arduino Code

```
#include <Wire.h>
#include <SPI.h>
#include <Adafruit_MCP4725.h>
#include "Protocentral_ADS1220.h"
Adafruit_MCP4725 dac;
// Set this value to 9, 8, 7, 6 or 5 to adjust the resolution
#define DAC_RESOLUTION    (9)
#define PGA          1          // Programmable Gain = 1
#define VREF         2.048     // Internal reference of 2.048V
#define VFSR         VREF/PGA
#define FULL_SCALE   (((long int)1<<23)-1)
#define ADS1220_CS_PIN    8
#define ADS1220_DRDY_PIN  7

unsigned long previousMillis = 0;
const byte numVariables = 8;
float variables[8];
int Which_sensor = 0;
int Serie_Resistance = 3;

Protocentral_ADS1220 pc_ads1220;
double data_in = 0;
double adc_data[1007] = {0};
int point = 0;
int point2 = 0;
int stage = 0;
double calibration = 0;
double calibration1 = 0;
double K_calibration = 0;
float volt_out = 0;
int Dubble_mesurment_stage = 0;
int Sweep_stage = 0;
int total_amount_of_sweeps = 0;

double voltage_out_dif = 0.012;
float volt_out_start = 2.5;
float Amount_of_sweeps = 2;
```

```
float Min_point = 2;
float Max_point = 3;
float volt_ower_transistor = 5;
int number_off_points = 80;
bool test = 0;
//Om denna och test1 = 1 så skriver vi ut vilken förstärkning
som används vid varje tal
bool test1 = 1;
//Denna är = 1 om vi ska skriva ut talen i real tid. Annars
skrivs dem ut i slutet.
bool test2 = 1;
//Skriver ut turn och done om den är 1
bool Multible_sweeps = 1;
int Delay_ms = 1000;

bool Dubble_mesurment = 0;
bool Serial_2Mohm = 0;
//if value = 1 then we use this resistance, do not have more then
one of these at value 1.
bool Serial_1Mohm = 1;
bool Serial_200Kohm = 0;
bool Serial_20Kohm = 0;

volatile bool drdyIntrFlag = false;

void drdyInterruptHndlr(){
  drdyIntrFlag = true;
}

void enableInterruptPin(){
  attachInterrupt(
  digitalPinToInterrupt(ADS1220_DRDY_PIN), drdyInterruptHndlr, FALLING
  );
}


void setup()
{
    pinMode(0, OUTPUT);   //Gain1
    pinMode(1, OUTPUT);   //Gain2
    pinMode(2, OUTPUT);   //Gain3
    pinMode(3, OUTPUT);   //Gain4
    pinMode(4, OUTPUT);   //Gain5
    pinMode(5, OUTPUT);   //Gain6
    pinMode(6, OUTPUT);   //Gain7
    pinMode(10, INPUT);   //Knapp
```

```
    pinMode(16, OUTPUT);  //Sensor1
    pinMode(17, OUTPUT);  //Sensor2

    Serial.begin(115200);
    pc_ads1220.begin(ADS1220_CS_PIN,ADS1220_DRDY_PIN);
    dac.begin(0x63);
    dac.setVoltage((volt_ower_transistor*4095)/5.02, false);// Dac1
    dac.begin(0x62);
    dac.setVoltage((volt_out_start*4095)/5.02, false);// Dac2

    pc_ads1220.set_data_rate(DR_20SPS);
    pc_ads1220.set_pga_gain(PGA_GAIN_1);
    pc_ads1220.set_conv_mode_single_shot(); //Set Single shot mode
}

void loop()
{
  if (stage == 0) {
    if (digitalRead(10) == 1) {
      stage = 1;
      Dubble_mesurment_stage = 0;
      previousMillis = millis();
      volt_out = volt_out_start;
      dac.setVoltage((volt_out*4095)/5.02, false);
      digitalWrite(0, HIGH);
      digitalWrite(1, LOW);
      digitalWrite(2, LOW);
      digitalWrite(3, LOW);
      digitalWrite(4, LOW);
      digitalWrite(5, LOW);
      digitalWrite(6, LOW);
      digitalWrite(16, LOW);              // byta grafen transistor
      digitalWrite(17, HIGH);             // byta grafen transistor
      Sweep_stage = 0;
      total_amount_of_sweeps = 0;
    }

    static byte variableIndex = 0;
  if (Serial.available()) {
    String inputString = Serial.readStringUntil('\n');
    variables[variableIndex] = inputString.toFloat();
    variableIndex++;

    if (variableIndex >= numVariables) {
      //volt_ower_transistor = variables[0];
      voltage_out_dif = variables[0];
```

```
      Max_point = variables[1] + 2.5;
      Min_point = variables[2] + 2.5;
      Delay_ms = variables[3];
      Amount_of_sweeps = variables[4];
      volt_ower_transistor = variables[5];
      Which_sensor = variables[6];
      Serie_Resistance = variables[7];
      Dubble_mesurment_stage = 0;
      previousMillis = millis();
      volt_out = volt_out_start;
      dac.begin(0x63);
      dac.setVoltage((volt_ower_transistor*4095)/5.02, false);// Dac1
      dac.begin(0x62);
      dac.setVoltage((volt_out*4095)/5.02, false);
      digitalWrite(0, HIGH);
      digitalWrite(1, LOW);
      digitalWrite(2, LOW);
      digitalWrite(3, LOW);
      digitalWrite(4, LOW);
      digitalWrite(5, LOW);
      digitalWrite(6, LOW);

      digitalWrite(16, LOW);          // byta grafen transistor
      digitalWrite(17, HIGH);         // byta grafen transistor
      if (Which_sensor == 1) {
        digitalWrite(16, HIGH);       // byta grafen transistor
        digitalWrite(17, LOW);        // byta grafen transistor
      }
      stage = 1;
      variableIndex = 0;
      Sweep_stage = 0;
      total_amount_of_sweeps = 0;
    }
  }
  }

  if (stage == 1) {
    delay(Delay_ms);
    if (Serie_Resistance == 4) {
      calibration = 766;          //(mV) in
      calibration1 = 556;         //(ohm) in
      K_calibration = calibration1/calibration;
    }
    if (Serie_Resistance == 3) {
      calibration = 766;          //(mV) in
      calibration1 = 300;         //(ohm) in
```

```
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 2) {
    calibration = 766;          //(mV) in
    calibration1 = 54;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 1) {
    calibration = 766;          //(mV) in
    calibration1 = 5.45;         //(ohm) in
    K_calibration = calibration1/calibration;
  }

  ADC_Mesurment();

  if (data_in > 7000000) {
    stage = 3;
    digitalWrite(0, LOW);
    digitalWrite(1, LOW);
    digitalWrite(2, HIGH);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
  }

  main_code();

}

  if (stage == 2) {
  delay(Delay_ms);
  if (Serie_Resistance == 4) {
    calibration = 1;          //(mV) in
    calibration1 = 1;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 3) {
    calibration = 1;
    calibration1 = 1;
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 2) {
    calibration = 1;
    calibration1 = 1;
    K_calibration = calibration1/calibration;
```

```
    }
    if (Serie_Resistance == 1) {
      calibration = 1;
      calibration1 = 1;
      K_calibration = calibration1/calibration;
    }

    ADC_Mesurment();

    if (data_in < 500000) {  // 451488 = 0,566V
      stage = 1;
      digitalWrite(0, HIGH);
      digitalWrite(1, LOW);
      digitalWrite(2, LOW);
      digitalWrite(3, LOW);
      digitalWrite(4, LOW);
      digitalWrite(5, LOW);
      digitalWrite(6, LOW);
    }
    if (data_in > 7000000) {
      stage = 3;
      digitalWrite(0, LOW);
      digitalWrite(1, LOW);
      digitalWrite(2, HIGH);
      digitalWrite(3, LOW);
      digitalWrite(4, LOW);
      digitalWrite(5, LOW);
      digitalWrite(6, LOW);
    }

    main_code();

  }

    if (stage == 3) {
      delay(Delay_ms);
    if (Serie_Resistance == 4) {
      calibration = 1367;          //(mV) in
      calibration1 = 2599;         //(ohm) in
      K_calibration = calibration1/calibration;
    }
    if (Serie_Resistance == 3) {
      calibration = 1367;
      calibration1 = 1402;
      K_calibration = calibration1/calibration;
    }
```

```
  if (Serie_Resistance == 2) {
    calibration = 1367;          //(mV) in
    calibration1 = 252;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 1) {
    calibration = 1367;          //(mV) in
    calibration1 = 25.5;         //(ohm) in
    K_calibration = calibration1/calibration;
  }

  ADC_Mesurment();

  if (data_in < 500000) {  // 451488 = 0,566V
    stage = 1;
    digitalWrite(0, HIGH);
    digitalWrite(1, LOW);
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
  }
  if (data_in > 7000000) {
    stage = 4;
    digitalWrite(0, LOW);
    digitalWrite(1, LOW);
    digitalWrite(2, LOW);
    digitalWrite(3, HIGH);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
  }

  main_code();

}

  if (stage == 4) {
    delay(Delay_ms);
  if (Serie_Resistance == 4) {
    calibration = 1022;          //(mV) in
    calibration1 = 5638;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 3) {
```

```
      calibration = 1022;
      calibration1 = 3040;
      K_calibration = calibration1/calibration;
    }
    if (Serie_Resistance == 2) {
      calibration = 1022;          //(mV) in
      calibration1 = 547;         //(ohm) in
      K_calibration = calibration1/calibration;
    }
    if (Serie_Resistance == 1) {
      calibration = 1022;          //(mV) in
      calibration1 = 55.2;         //(ohm) in
      K_calibration = calibration1/calibration;
    }

    ADC_Mesurment();

    if (data_in < 500000) {  // 451488 = 0,566V
      stage = 3;
      digitalWrite(0, LOW);
      digitalWrite(1, LOW);
      digitalWrite(2, HIGH);
      digitalWrite(3, LOW);
      digitalWrite(4, LOW);
      digitalWrite(5, LOW);
      digitalWrite(6, LOW);
    }
    if (data_in > 7000000) {
      stage = 5;
      digitalWrite(0, LOW);
      digitalWrite(1, LOW);
      digitalWrite(2, LOW);
      digitalWrite(3, LOW);
      digitalWrite(4, HIGH);
      digitalWrite(5, LOW);
      digitalWrite(6, LOW);
    }

   main_code();

  }

    if (stage == 5) {
    delay(Delay_ms);
    if (Serie_Resistance == 4) {
      calibration = 804;          //(mV) in
```

```
    calibration1 = 13963;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 3) {
    calibration = 804;          //(mV) in
    calibration1 = 7529;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 2) {
    calibration = 804;          //(mV) in
    calibration1 = 1356;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 1) {
    calibration = 804;          //(mV) in
    calibration1 = 136.8;         //(ohm) in
    K_calibration = calibration1/calibration;
  }

  ADC_Mesurment();

  if (data_in < 500000) {  // 451488 = 0,566V
    stage = 4;
    digitalWrite(0, LOW);
    digitalWrite(1, LOW);
    digitalWrite(2, LOW);
    digitalWrite(3, HIGH);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
  }
  if (data_in > 7000000) {
    stage = 6;
    digitalWrite(0, LOW);
    digitalWrite(1, LOW);
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, HIGH);
    digitalWrite(6, LOW);
  }

  main_code();

}
```

```
if (stage == 6) {
delay(Delay_ms);
if (Serie_Resistance == 4) {
  calibration = 1071;          //(mV) in
  calibration1 = 55912;         //(ohm) in
  K_calibration = calibration1/calibration;
}
if (Serie_Resistance == 3) {
  calibration = 1071;          //(mV) in
  calibration1 = 30146;         //(ohm) in
  K_calibration = calibration1/calibration;
}
if (Serie_Resistance == 2) {
  calibration = 1071;          //(mV) in
  calibration1 = 5432;         //(ohm) in
  K_calibration = calibration1/calibration;
}
if (Serie_Resistance == 1) {
  calibration = 1071;          //(mV) in
  calibration1 = 547.7;         //(ohm) in
  K_calibration = calibration1/calibration;
}

ADC_Mesurment();

if (data_in < 500000) {  // 451488 = 0,566V
  stage = 5;
  digitalWrite(0, LOW);
  digitalWrite(1, LOW);
  digitalWrite(2, LOW);
  digitalWrite(3, LOW);
  digitalWrite(4, HIGH);
  digitalWrite(5, LOW);
  digitalWrite(6, LOW);
}
if (data_in > 7000000) {
  stage = 7;
  digitalWrite(0, LOW);
  digitalWrite(1, LOW);
  digitalWrite(2, LOW);
  digitalWrite(3, LOW);
  digitalWrite(4, LOW);
  digitalWrite(5, LOW);
  digitalWrite(6, HIGH);
}
```

```
  main_code();

}

  if (stage == 7) {
  delay(Delay_ms);
  if (Serie_Resistance == 4) {
    calibration = 961;          //(mV) in
    calibration1 = 150102;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 3) {
    calibration = 961;          //(mV) in
    calibration1 = 80929;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 2) {
    calibration = 961;          //(mV) in
    calibration1 = 14582;         //(ohm) in
    K_calibration = calibration1/calibration;
  }
  if (Serie_Resistance == 1) {
    calibration = 961;          //(mV) in
    calibration1 = 1470.5;         //(ohm) in
    K_calibration = calibration1/calibration;
  }

  ADC_Mesurment();

  if (data_in < 500000) {  // 451488 = 0,566V
    stage = 6;
    digitalWrite(0, LOW);
    digitalWrite(1, LOW);
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, HIGH);
    digitalWrite(6, LOW);
  }

  main_code();

}

if (stage == 9) {
```

```
if (Dubble_mesurment_stage == 1)  {
  //Serial.println(millis()-previousMillis);
  if (test1 == 0){
  for (int i = 0; i < number_off_points; i++) {
    Serial.println(adc_data[i+7]);
  }
  }
  Serial.println("done");
  point = 0;
  stage = 0;
}

if (Which_sensor == 2)  {
if (Dubble_mesurment_stage == 0)  {
  //Serial.println(millis()-previousMillis);
  if (test1 == 0){
  for (int i = 0; i < number_off_points; i++) {
    Serial.println(adc_data[i+7]);
  }
  }
  Serial.println("mid");
  stage = 1;
  volt_out = volt_out_start;
  dac.setVoltage((volt_out*4095)/5.02, false);
  digitalWrite(0, HIGH);
  digitalWrite(1, LOW);
  digitalWrite(2, LOW);
  digitalWrite(3, LOW);
  digitalWrite(4, LOW);
  digitalWrite(5, LOW);
  digitalWrite(6, LOW);
  digitalWrite(16, Serial_20Kohm);
  digitalWrite(17, Serial_200Kohm);
  digitalWrite(16, HIGH);      // byta grafen transistor
  digitalWrite(17, LOW);       // byta grafen transistor
  Dubble_mesurment_stage = 1;
  point = 0;
  delay(5000);
}
}

if (Which_sensor != 2) {
  //Serial.println(millis()-previousMillis);
  if (test1 == 0){
  for (int i = 0; i < number_off_points; i++) {
    Serial.println(adc_data[i+7]);
```

```
      }
      }
      //Serial.println(point-7);
      point = 0;
      stage = 0;
    }
  }
}

void ADC_Mesurment(){
  data_in =
  pc_ads1220.Read_SingleShot_SingleEnded_WaitForData(MUX_SE_CH0);
  adc_data[point] =
  ((1.008*data_in*VFSR*1000)/FULL_SCALE+452)*K_calibration;
}

void main_code(){
  if (Multible_sweeps == 0) {
      if (point > 6) {
       if (test1 == 1) {
          Serial.println(adc_data[point]);
       }
        if (test == 1){
          Serial.println(stage);
       }
       volt_out += voltage_out_dif;
       dac.setVoltage((volt_out*4095)/5.02, false);
      }
      if (point > number_off_points+6) {
         stage = 9;
         //Serial.println(millis()-previousMillis);
       }
       point++;
    }

    if (Multible_sweeps == 1) {
    if (volt_out >= Max_point){
      if (test2 == 1) {
          Serial.println("turn");
       }
      Sweep_stage = 1;
      }

    if (volt_out <= Min_point){
      if (test2 == 1) {
          Serial.println("turn");
```

```
    }
   Sweep_stage = 2;
}

if (Sweep_stage == 0){
  if (point > 6) {
   if (test1 == 1) {
      Serial.println(adc_data[point]);
   }
    if (test == 1){
      Serial.println(stage);
   }
   volt_out += voltage_out_dif;
   dac.setVoltage((volt_out*4095)/5.02, false);
   point--;
  }
  point++;
}

if (Sweep_stage == 1){
   if (test1 == 1) {
      Serial.println(adc_data[point]);
   }
    if (test == 1){
      Serial.println(stage);
   }
   volt_out -= voltage_out_dif;
   dac.setVoltage((volt_out*4095)/5.02, false);
}

if (Sweep_stage == 2){
   if (test1 == 1) {
      Serial.println(adc_data[point]);
   }
    if (test == 1){
      Serial.println(stage);
   }
   volt_out += voltage_out_dif;
   dac.setVoltage((volt_out*4095)/5.02, false);
   if (volt_out >= 2.5){
    Sweep_stage = 0;
    total_amount_of_sweeps += 1;
    if (Amount_of_sweeps == total_amount_of_sweeps){
      stage = 9;
      if (test2 == 1) {
      Serial.println("done");
```

```
          }
        }
      }
    }
    if (Serial.available()) {
      String inputString = Serial.readStringUntil('\n');
      if (inputString.toInt() == 99) {
        stage = 0;
      }
    }
    }
}

float convertToMilliV(int32_t i32data)
{
    return (float)((1.008*i32data*VFSR*1000)/FULL_SCALE+452);
    // K = 1.008 and m = 452 (mV)
}
```

B

Appendix - Kicad

The first Kicad figure shows an Arduino Nano Every, which controls all the components in the circuit B.1.

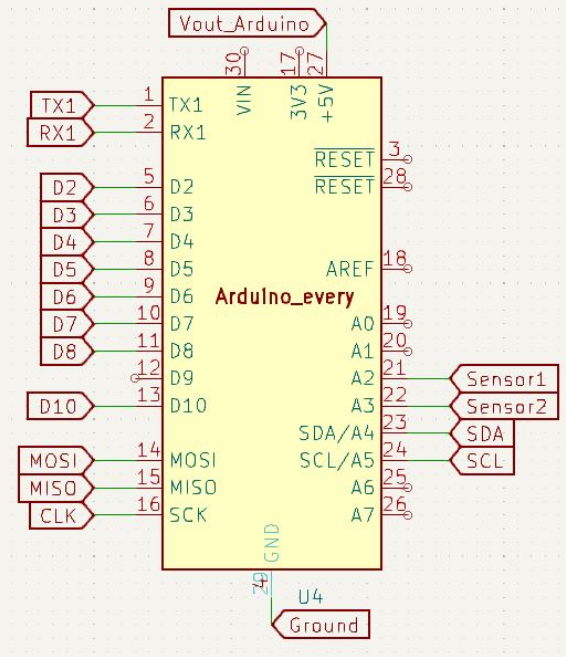


Figure B.1: *The Arduino nano every schematic and its signals.*

The Arduino did not have any DACs of its own, so two additional DACs were used, as shown in Figure B.2. Both DACs are controlled by the I2C protocol, and the address of each DAC is determined by the A0 pin, depending on whether it is connected to GND or Vcc. An additional voltage regulator was also used because the Arduino's voltage was unstable.

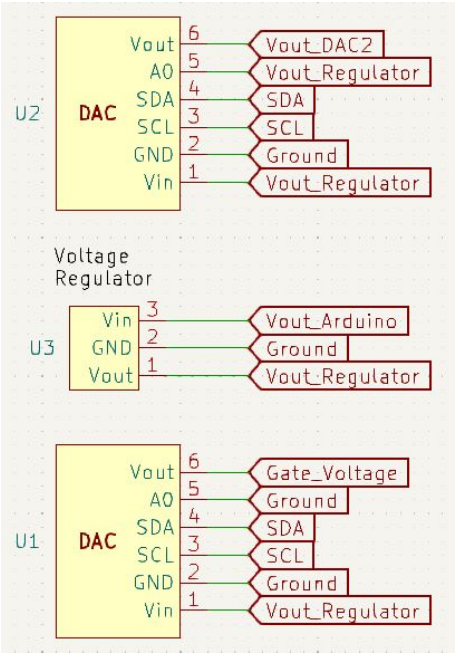


Figure B.2: *Two 12-bit DACs and the voltage regulator.*

An additional ADC was used because the Arduino's built-in ADC does not have sufficient precision. The ADC is controlled by the Arduino through the SPI protocol and is connected to the output of the instrumention amplifier on pin IN0 B.3. The amplifier outputs the measurement from the G-FET, and the ADC sends the result to the Arduino.

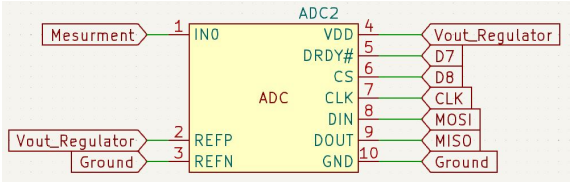


Figure B.3: *The 24-bit ADC.*

The instrumention amplifier has seven different amplification levels, as shown in Figure B.4. All the amplification steps are controlled by different pins from the Arduino. These seven amplification levels allow the system to measure a wider range of resistances.

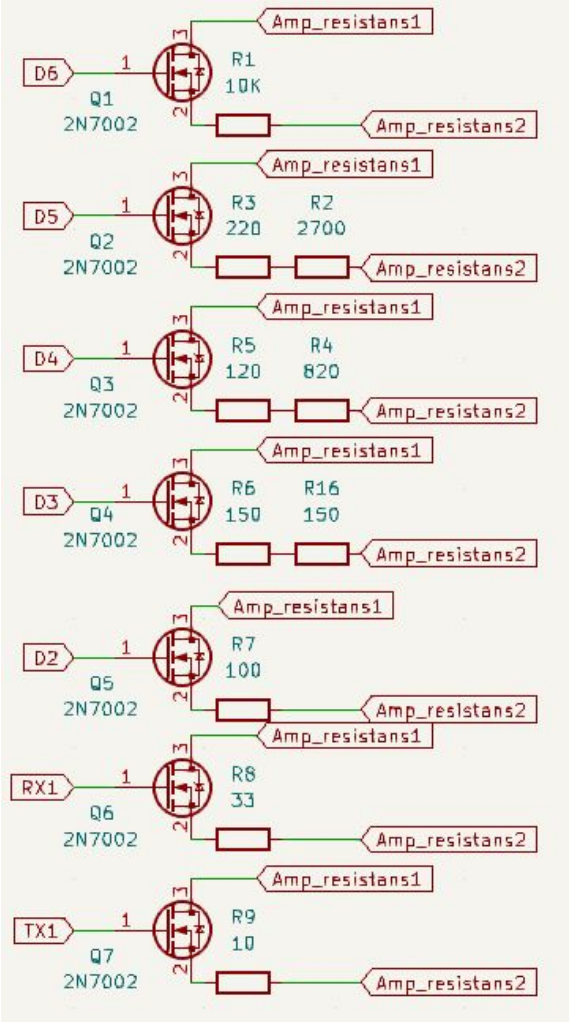


Figure B.4: *Seven amplification steps for the instrumention amplifier.*

Six transistors are placed at the input and output of the micro SD card connector, as shown in Figure B.5. The SD card connector is configured to accommodate two G-FETs simultaneously, and these transistors open and close depending on which G-FET is being measured. They are controlled by signals sent from the Arduino.

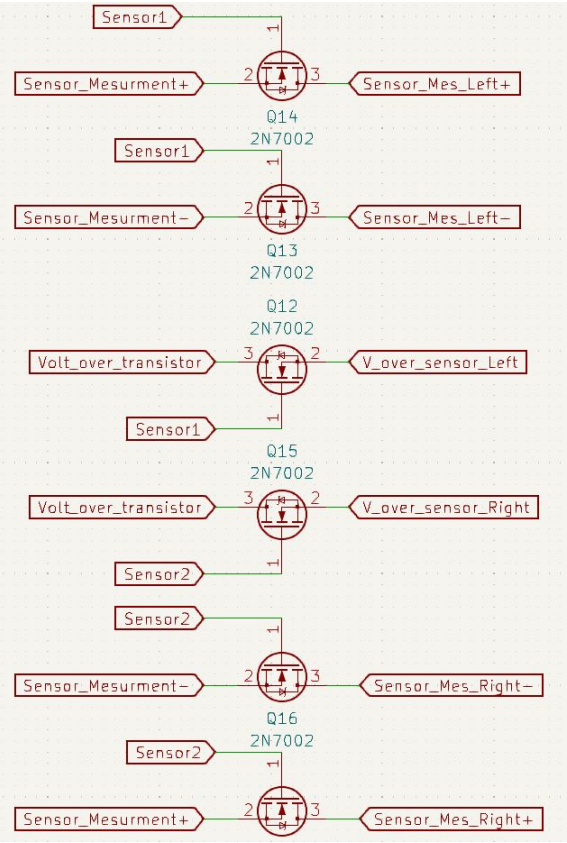


Figure B.5: *Transistors for the measurement ports on the micro sd card connector.*

A DIL switch with four switches was used to select the resistance to be placed in series with the G-FET, as shown in Figure B.6. The combination of the series resistance and the DAC determines the amount of current that flows through the G-FET.

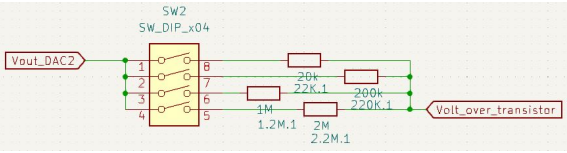


Figure B.6: *Four DIP switches that decide which resistance the circuit uses.*

The instrumention amplifier is connected to seven different amplifications and to the measurement ports on the micro SD card connector, as shown in Figure B.7. One amplifier is able to measure two G-FETs by opening and closing the ports of the G-FETs with transistors. The Arduino opens and closes the transistors when needed.

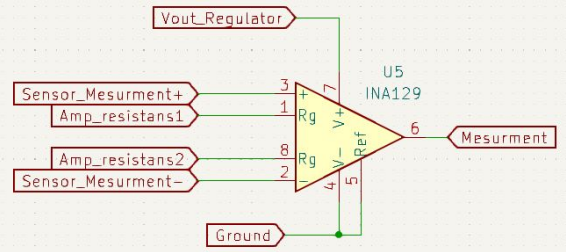


Figure B.7: *The instrumention amplifier.*

The micro SD card connector, shown in Figure B.8 is configured to measure two G-FETs and output a gate voltage. The port for the gate voltage is connected to both G-FETs, and the V- is also connected to both G-FETs. Each G-FET has its own V+ and measurement pins.

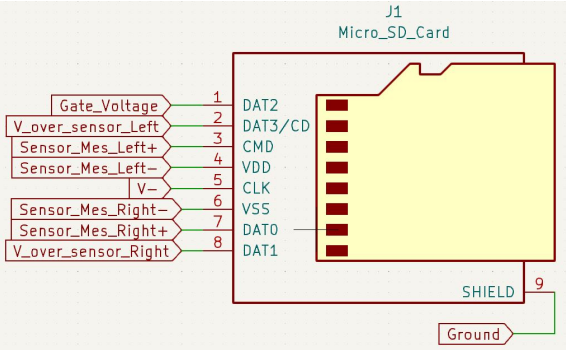


Figure B.8: *The micro SD card connector.*

A button is placed on the PCB to start the measurement, as shown in Figure B.9.

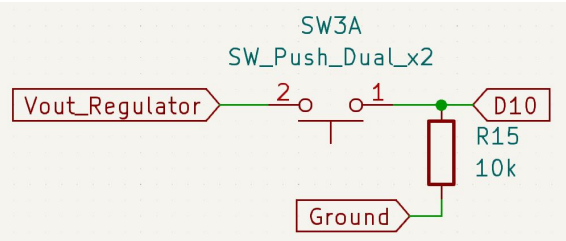


Figure B.9: *A button connected to an Arduino port.*

An operational amplifier (OP amp) is positioned after a voltage divider to generate the V- voltage, as depicted in Figure B.10. The V- is connected to both G-FETs and is intended to be 2.5 volts. This voltage facilitates negative Vgs sensing since the source voltage is 2.5 volts, allowing the gate voltage to decrease below 2.5 volts.

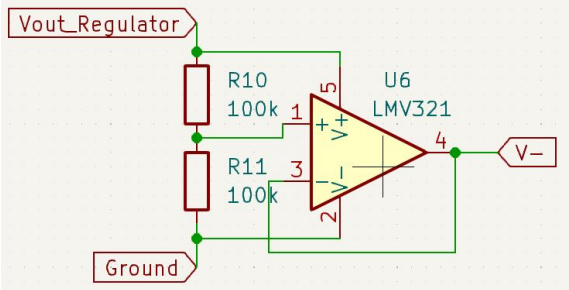


Figure B.10: *A voltage divider which outputs 2.5 Volts.*

To facilitate securing the PCB to a container using screws, four mounting holes were incorporated, as depicted in Figure B.11. Three of these holes feature an additional metal border to withstand greater force. However, one hole had to be positioned in close proximity to the Arduino pins, preventing the inclusion of a metal border. Consequently, these four mounting holes were distributed across four different locations on the PCB.

Figure B.11: *Four mounting holes to attach the PCB to something.*

In Figure B.12 the front side of the PCB is depicted when a 3D model is generated from KiCad.

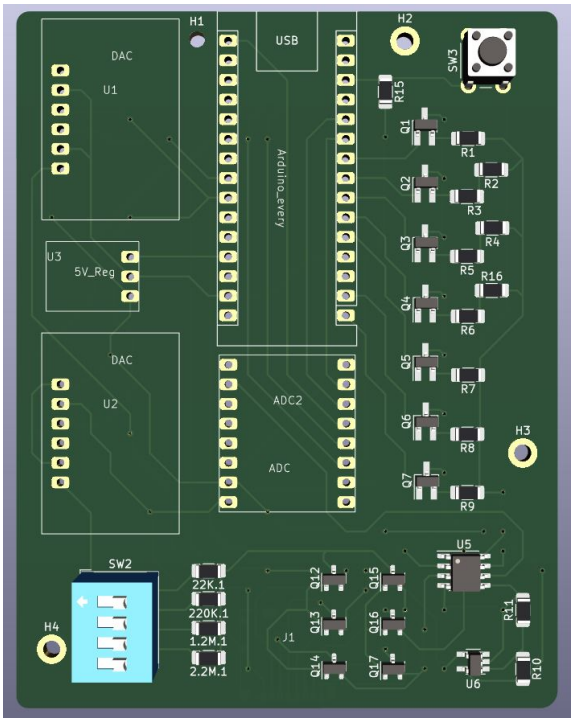


Figure B.12: *A 3D view of the front side of the PCB.*

In Figure B.13 the back side of the PCB is shown when a 3D model is generated from KiCad.

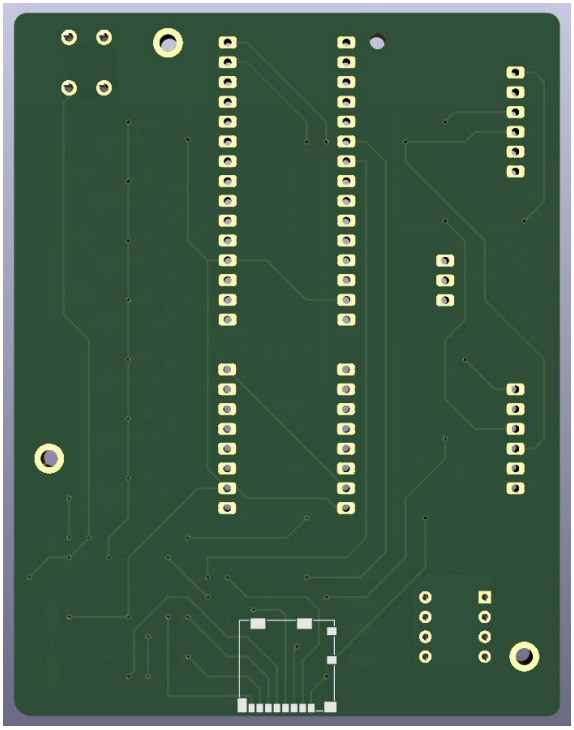


Figure B.13: *A 3D view of the back side of the PCB.*

C

Appendix - Circuit

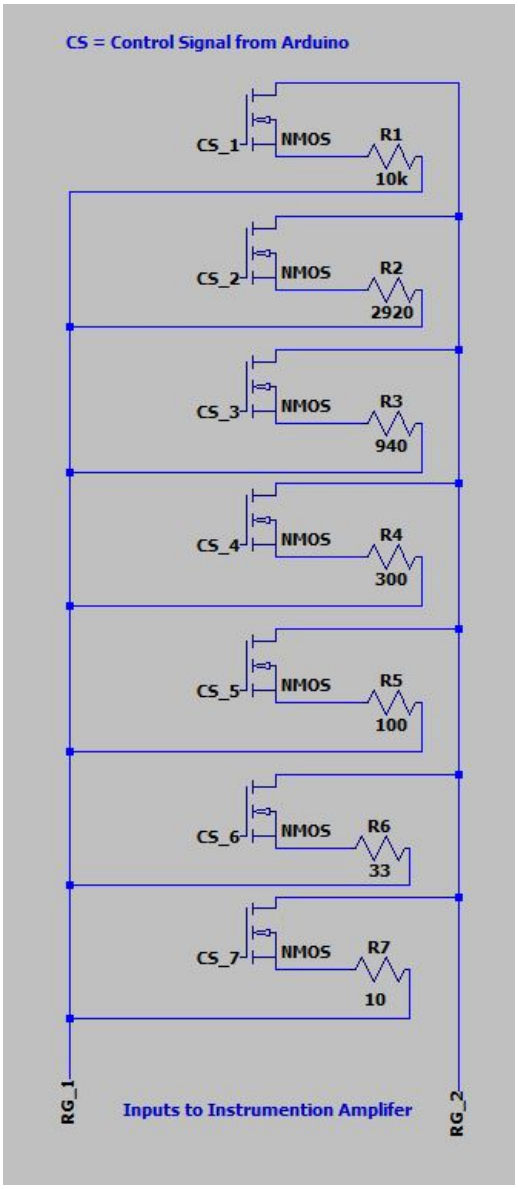


Figure C.1: *Schematic of the variable resistance network. Only one transistor will be open at a time, resulting in the correlating resistance plus 2 ohm from the transistor.*

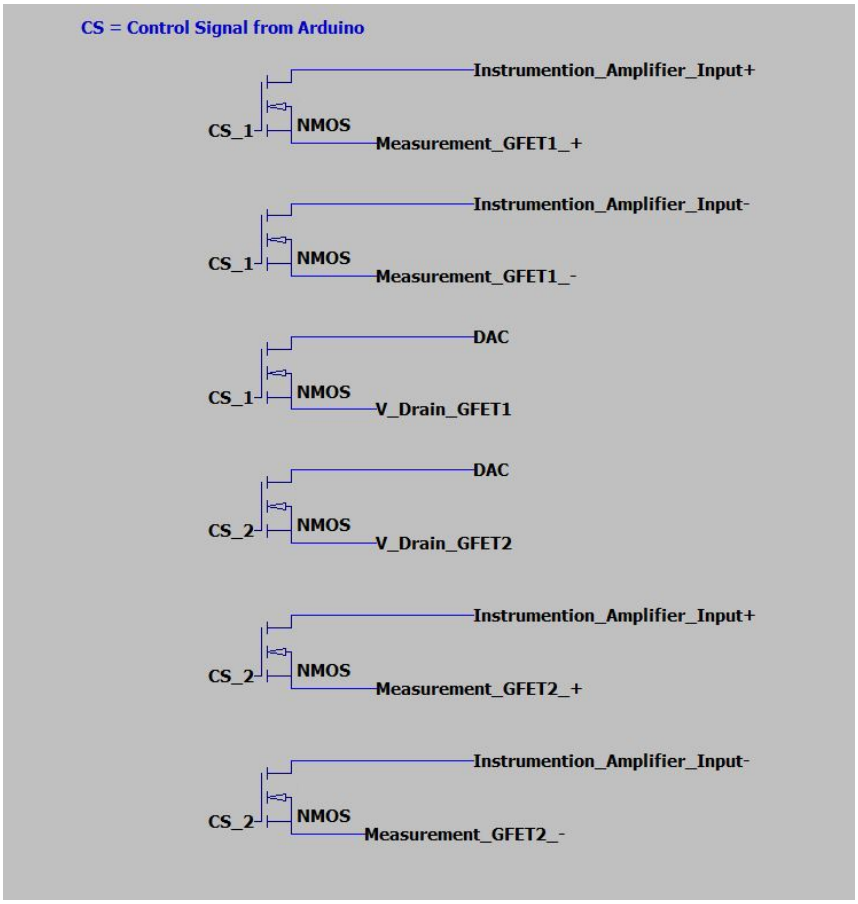


Figure C.2: *Schematic of transistors connected to the pins on the micro SD card connector. One of the two signals, CS1 or CS2, will be activated and connect one G-FET to the circuit.*

DEPARTMENT OF SOME ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS