UNIVERSITY *of* WASHINGTON | BOTHELL

Auto-Ranging Capacitance Meter Project

Designed by: Forrest Zhang

I. Objectives

The objective of this design project is to build an Auto Ranging Capacitance Meter that accurately measures capacitance of unknown capacitors, with accuracy requirement of: 1. 2nF to 2uF with 2% accuracy. 2. 10% accuracy for 20pF to 2nF and 2uF to 200uF.

II. Design Process

1. Theory of operation

There are multiple ways to design a device to measure a capacitor. From the design project specification page, Dr. Berger listed 3 different ways to measure capacitance:

1- Tuned LC circuit

2- Astable multivibrator

3- Constant current source

However, after researching capacitor measuring methods, I decided to use the RC time constant method which wasn't listed. This design is based on a formula about voltage across capacitance relative to charge time which we learned during one of the labs in EE 233 Circuit Theory course:

$$V_C(t) = V_{\text{supply}}\left(1 - e^{-\frac{t}{RC}}\right)$$

By looking at this formula, we can see that given the known supply voltage and known resistance value, if we can find the charge time and voltage across the capacitor, we can calculate the capacitance value.

In order to make this method more practical, we can isolate the capacitance C using the time constant. The graph below shows the voltage across a capacitor as it charges over time in an RC circuit:

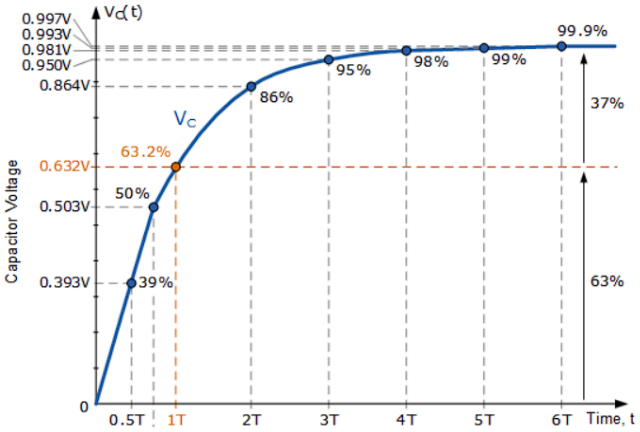


Figure 1: Vc/Charge Time Graph

Based on the percentage of supply voltage shown in the graph, we see that 1 time constant happens when Vc reaches 63.2% of supply voltage. We then get the below equation:

$$V_C(\tau) = V_{\text{supply}} \left(1 - e^{-1}\right)$$

Since $e^{-1} \approx 0.3679$,

$$V_C(\tau) = V_{\text{supply}} \left(1 - 0.3679\right)$$
$$V_C(\tau) = V_{\text{supply}} \times 0.632$$

This implies that at time t = 1 time constant, Time constant = RC.

By analyzing the formulas, I designed the basic logic of capacitance measurement. that is:
charging the capacitor with a known resistor, and starting a timer at the start of charging. Use an analog input to determine if the capacitor voltage reaches 63.2% of known supply voltage (5V). Once Vc reaches 63.2% of 5V supply voltage (3.16V), stop the timer and return the time for Vc to charge from 0% to 63.2%, then use the formula: C = t/R to calculate the capacitance.

By using this design, we can avoid the issue of Arduino A/D converters not being fast enough, since the speed of voltage change doesn't affect as much as other designs. However, in order to design a device that accurately measures small to medium range of capacitance, there's factors of charge time and accuracy to consider. The simplest way to analyze this is the worst case scenario.

With only one resistor in the circuit to charge the capacitor, if the capacitor is very large (uf range) and the resistor value is relatively large, it gives a relatively small current and it will take too long for the capacitor to charge up to 63.2%. Similarly, if the capacitor is very small (pf range) and the resistor values are relatively small, there's too much current in the circuit and charging the capacitor is going to be almost instantaneous, which means there's not enough time for accurate measurements. To improve this design, there needs to be different resistors in the circuit, and the software will determine which resistor to choose.

After improving initial designs based on practicality, I designed the auto ranging and discharging circuit as below.

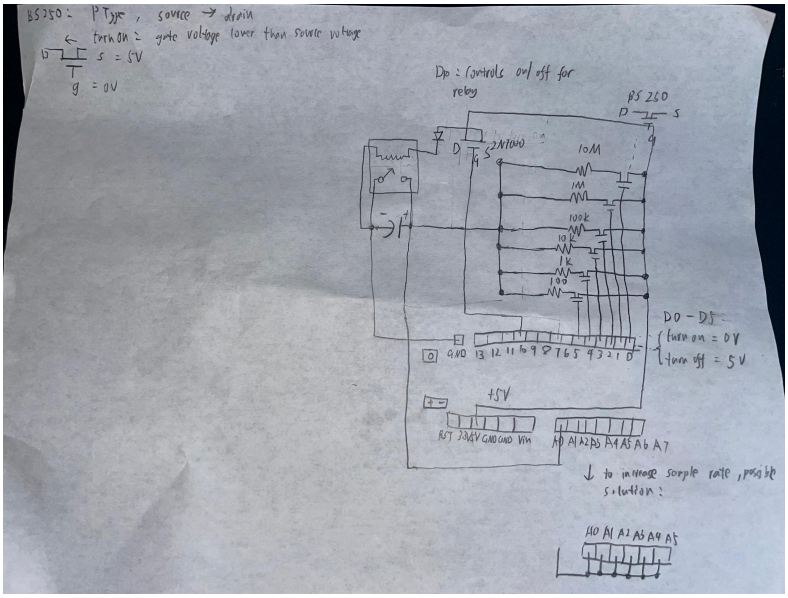


Figure 2: On Paper Design Schematic

Based on this design, there will be 6 different resistance values that the Arduino ATMega 2560 microcontroller can choose from by turning on the PMOS on that branch.

2. Hardware component and embedded software flow explanation

a. Hardware Components

- Arduino Mega R3: Microcontroller used to control the operation of the capacitance meter.
- 16x2 LCD Display: Used to display messages, current, and measured capacitance values.
- Resistors: A set of resistors (finalized resistor values: 100Ω, 1kΩ, 10kΩ, 100kΩ, 1MΩ, 50MΩ) used for the charging circuit.
- BS250 PMOS Transistors: Used to switch the resistors in and out of the circuit.
- Relay module (containing diode and NMOS necessary to control the relay): Used to discharge the capacitor by shorting it to the ground.
- Capacitor: The component whose capacitance is to be measured.
- Power Supply: The constant +5V voltage from an external power supply module

b. Pin Configuration

LCD Display Pins:

- rs = 51, en = 49, d4 = 47, d5 = 45, d6 = 43, d7 = 41

Analog Pin:

- A0 = Connected to the positive terminal of the capacitor to measure voltage VcV_cVc

Resistor Control Pins:

- Pins 1-6 control the resistors (100Ω, 1kΩ, 10kΩ, 100kΩ, 1MΩ, 50MΩ) These pins are connected to the gates of BS250 PMOS transistors

Relay Control Pin:

- Pin 10 controls the relay which discharges the capacitor by shorting it to the ground

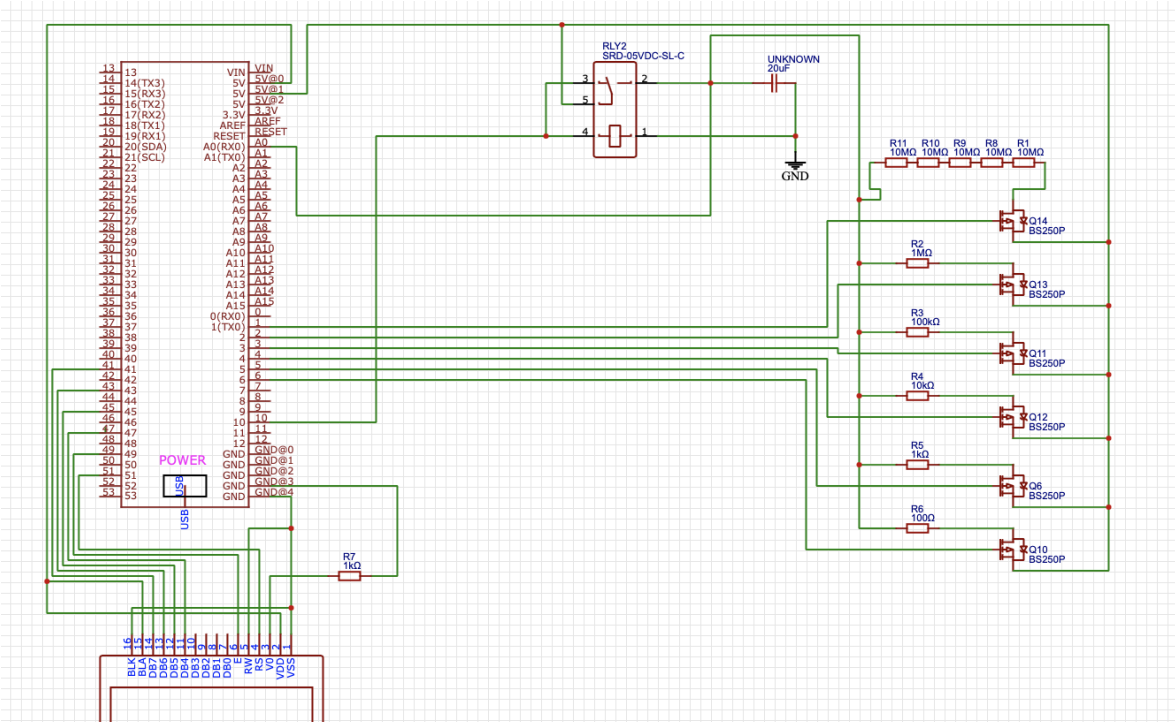


Figure 3: Finalized Design Schematic

c. Software Operational Flow

1. Initialization:

The Arduino powers on, initializing the LCD and displaying a message prompting the user to connect the capacitor. The message remains for 3 seconds.

2. **Discharging the Capacitor:**
   The relay is activated (Pin 10 HIGH) to discharge the capacitor, ensuring it starts from 0V.
   All resistor control pins (Pins 1-6) are set HIGH to turn off the power to the charging circuit.
   The system waits for 3 seconds to ensure full discharge of the capacitor.

3. **Auto-Ranging:**
   The system begins auto-ranging by iterating through the resistors from highest (10MΩ) to lowest (1kΩ). During each iteration, the selected branch first turns on by switching LOW, then after charging it for 2 seconds, if Vc reaches threshold, stop iteration and return the branch selected. Before the next iteration, turn off power by switching the branch HIGH.

```
58  //--------------------------------------------------------------------------
59  int autoRangeResistor() {
60    lcd.clear();
61    lcd.print("Auto Ranging...");
62    int i;
63    for (i = 5; i >= 0; i--) { //Auto ranging loop
64      lcd.setCursor(0, 1);
65      lcd.print("Testing R");
66      lcd.print(i + 1);
67      lcd.print("   ");
68      digitalWrite(resistorPins[i], LOW); // Turn on the resistor
69      delay(2000); // Allow some time for charging
70      if (analogRead(analogPin) > thresholdVoltage / VCC * 1023) {
71        break;
72      }
73      digitalWrite(resistorPins[i], HIGH); // Turn off the resistor
74    }
75    return i;
76  }
```

Figure 4: Auto-Ranging Loop

4. **Charging the Capacitor:**
   The system charges the capacitor using the selected resistor and starts a timer. The capacitor charges until it reaches approximately 63.2% of the supply voltage. If the capacitor does not reach the threshold voltage within the maximum charge time (5 seconds), the system switches to the next lower resistor and repeats the process.

5. **Measuring Capacitance:**
   Once the correct resistor is selected, and the capacitor reaches the threshold voltage, the timer is stopped. The capacitor is discharged again. Then the capacitance is calculated using the time constant formula: $C = t / R$ where t is the elapsed time, and R is the resistor value.

```
//--------------------------------------------------------------------------------
//Phase 2: Measure capactance
float measureCapacitance(int resistorIndex) {

  dischargeCapacitor();
  lcd.clear();
  lcd.print("Measuring...");

  digitalWrite(resistorPins[resistorIndex], LOW); // Turn on the selected resistor
  unsigned long startTime = millis();

  while (analogRead(analogPin) < thresholdVoltage / VCC * 1023);
  unsigned long elapsedTime = millis() - startTime;
  digitalWrite(resistorPins[resistorIndex], HIGH); // Turn off the resistor

  float resistance = resistorValues[resistorIndex];
  float capacitance = (float)elapsedTime / resistance;

  return capacitance / 1000.0;
}
```

Figure 5: Capacitance Measurement

6. **Displaying Results:**
   The capacitance unit is calculated and displayed correctly on the LCD indefinitely, holding the result for the user to read.

```
100  //Phase 3: Display capactance on LCD
101  void displayCapacitance(float capacitance) {
102    lcd.clear();
103
104    if (capacitance >= 1e-6) {
105      lcd.print(capacitance * 1e6, 2);
106      lcd.print(" uF");
107
108    } else if (capacitance >= 1e-9) {
109      lcd.print(capacitance * 1e9, 2);
110      lcd.print(" nF");
111
112    } else {
113      lcd.print(capacitance * 1e12, 2);
114      lcd.print(" pF");
115    }
116  }
```

Figure 6: Capacitance and Units Display

3. Breadboard Testing

Developing the ACM breadboard prototype include several stages and problems:

1. Initial Circuit Construction

To verify the experimental model of the capacitance meter, I first built a simplified version of the circuit on a breadboard. This initial setup included a single PMOS transistor, a resistor, and a capacitor. The objective was to test the voltage changes across the capacitor to ensure the fundamental charging and discharging mechanisms worked correctly. The model functioned as expected, with observable voltage changes corresponding to the capacitor charging and discharging cycles. After proving that this measuring model works, I built the first version of breadboard prototype and Arduino code. In the initial version of the breadboard, I used 6 2N7000 NMOS transistors to control the resistors. However, this component caused an issue due to the voltage drop across it, interfering with the accurate measurement of the capacitor's voltage, leading to errors in the capacitance calculation. Another issue of 2N7000 is that NMOS transistors typically have a higher on-resistance and require a certain voltage drop (Vgs) to turn on fully, which can affect the precision needed for measuring small capacitances accurately. To resolve this, I redesigned the circuit using 6 BS250 PMOS transistors, which has a much lower voltage drop when turned on, especially in the configuration used for this project. This minimizes the interference with the voltage measurements across the capacitor, leading to more accurate results.This redesign eliminated the voltage drop issue, improving the accuracy of the capacitance measurements.

2. Resistor Selection and Timing Calculations

After confirming the basic functionality, I calculated the best and worst-case charging times for the capacitors and selected appropriate resistors for the auto-ranging mechanism. The chosen resistors (R1-R6) are as follows:

R1: 100 Ω
R2: 1 kΩ
R3: 10 kΩ
R4: 100 kΩ
R5: 1 MΩ
R6: 50 MΩ

These resistors were selected to cover the capacitance range from 200 μF to 20 pF.

3. Second Version of Code and Breadboard Setup

I developed the second version of the Arduino code and assembled the complete circuit on the breadboard. However, when powered on, the LCD backlight illuminated, but no text was displayed. This issue was resolved by adding a 1 kΩ resistor to the potentiometer pin of the LCD screen.

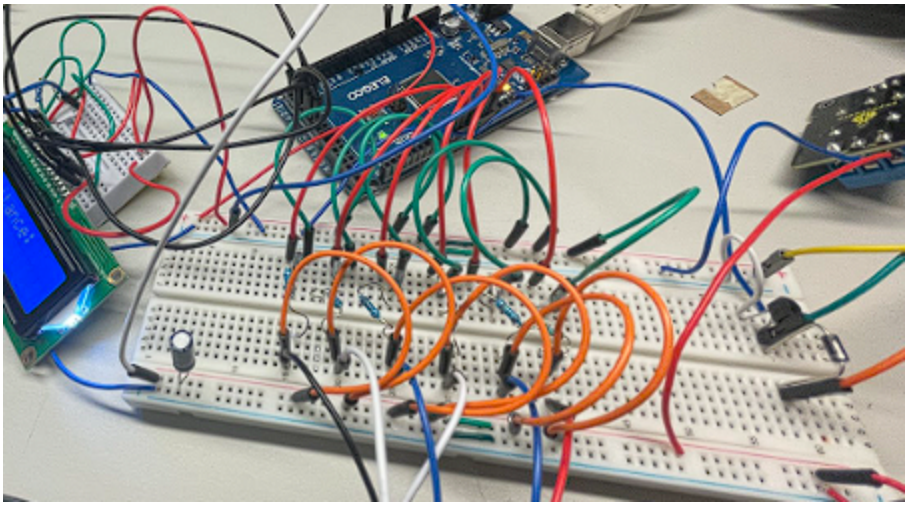


Figure 8: Breadboard Prototype Second Version (BS250)

4. . Debugging Auto-Ranging Loop

During initial testing, the breadboard setup with the first version of the code failed to operate correctly, becoming stuck in an infinite auto-ranging loop. This problem was traced to an incorrect relay connection. The schematic I designed utilized a relay with an NMOS transistor and a diode, but the breadboard setup used a relay module directly.

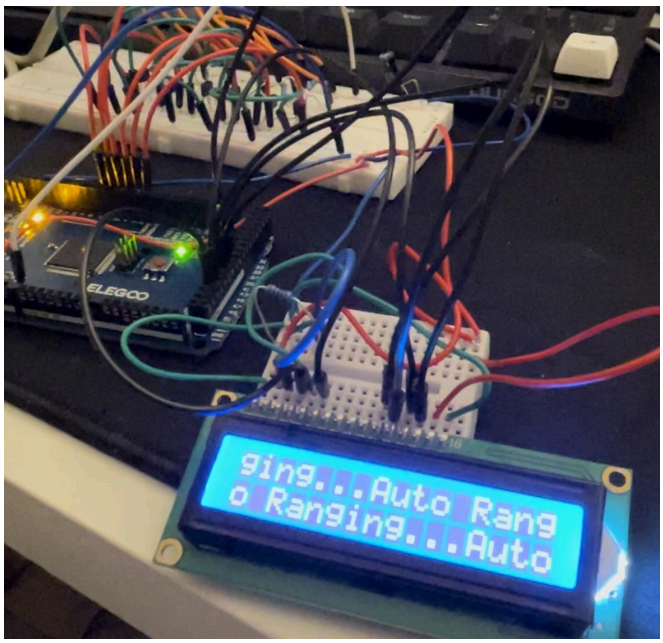


Figure 9: Auto-Ranging Error

To resolve this:

- I removed the diode and NMOS transistor from the breadboard setup.
- Corrected the relay connections according to the relay module specifications.

Once these adjustments were made, the auto-ranging mechanism functioned correctly, and the circuit began providing accurate measurements.

5. Measurement Accuracy and Breadboard Capacitance

After finalizing the embedded code and testing different capacitance on the breadboard, the circuit provided accurate measurements from 220uf down to 1000 pF.

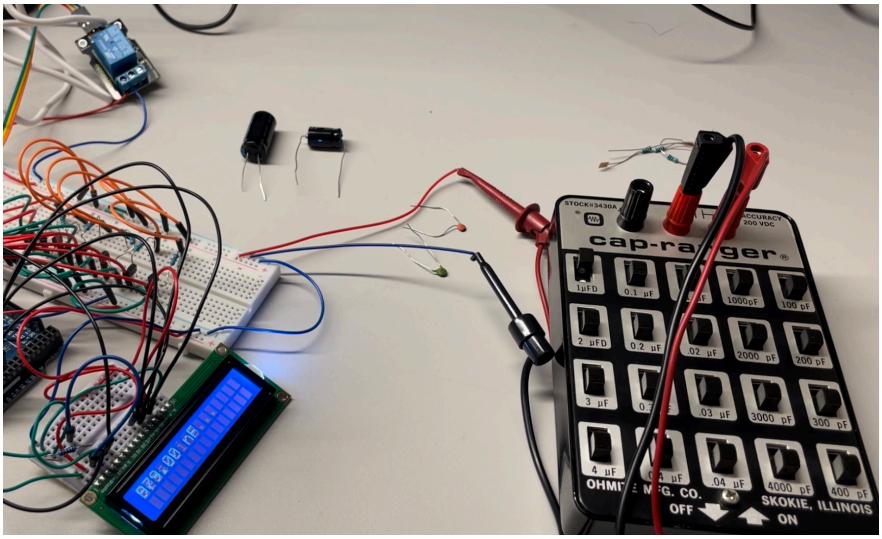


Figure 10: Capacitance measurement (rated 1 uf)

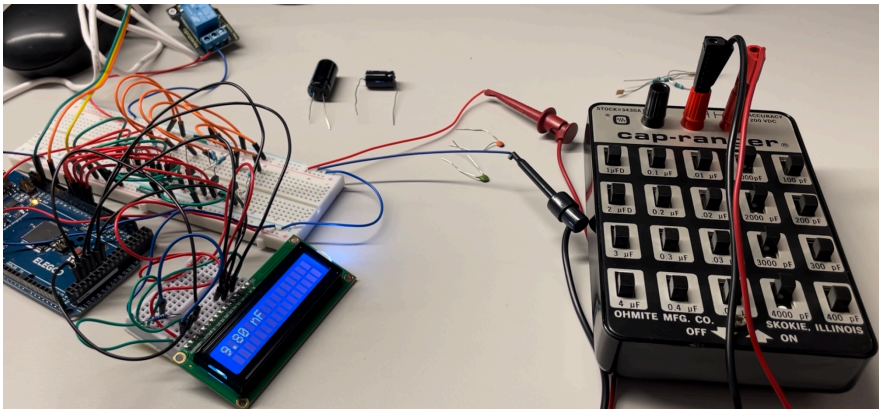


Figure 11: Capacitance Measurement (Rated 10000pf = 10nf)

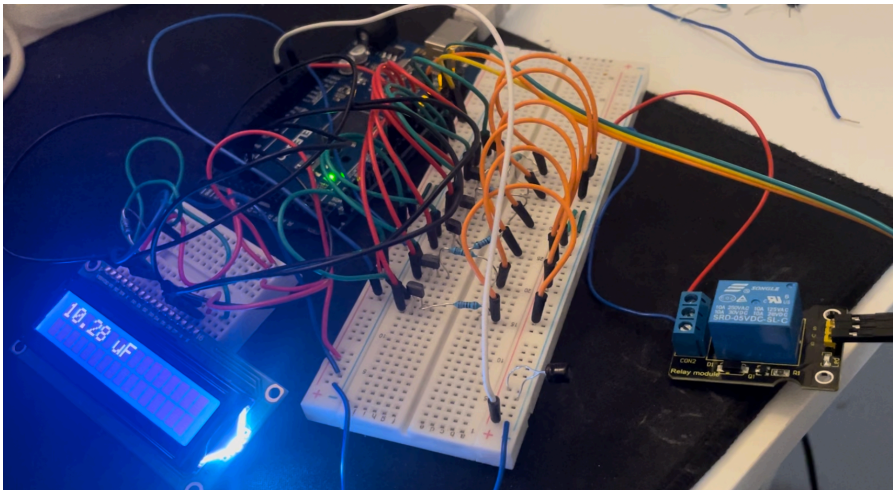


Figure 12: Capacitance measurement (rated 10uf)

Figure 13: Capacitance Measurement (rated 120nf)

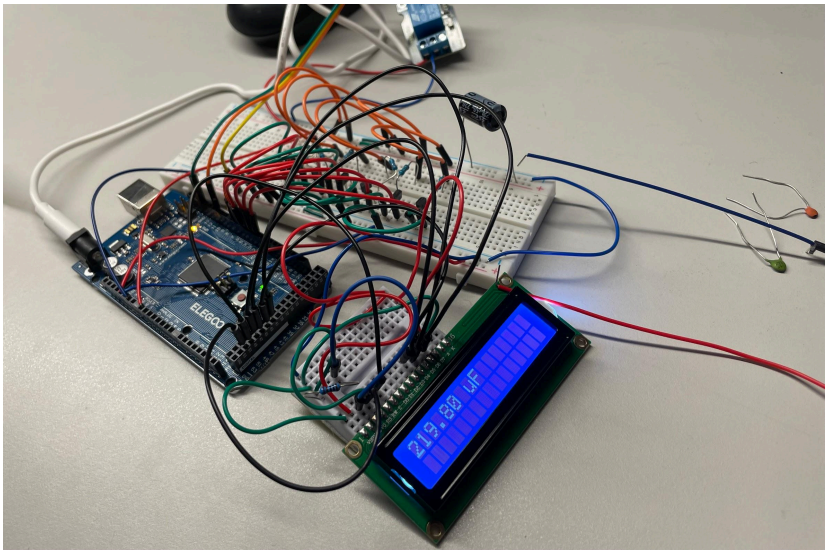


Figure 14: Capacitance Measurement (rated 220uf)

All measurements within different ranges (uf, nf, pf) meet the accuracy requirement. However, measurements for capacitors below 1000 pF were inaccurate due to the inherent capacitance of the breadboard. This issue is only solvable by a customized PCB.

Materials Used

The following supplies listed in Table 1 were used in this project. Some of the materials used such as BS250P, female pin headers and alligator clips were bought in bulk, but I only used a couple of them. Another thing to point out is that the wiring has multiple types such as male to male, female to female, and male to female. The other materials listed are either from the lab or already owned products.

Table 1: Materials Used for Capacitance Meter

Components	**Quantity (pcs)**	**Cost ($)**
BS250P	30 - Only used 6	$20.25
Alligator Clips	6 - Only used 2	$4
Arduino Mega	1	$21
Wires	As needed	Provided
Relay Module: SRD-05VDC-SL-C	1	$4
PCB	5	$54
1602 LCD	1	$5
Male Pin Headers	As needed	Provided
Female Pin Headers	4 - Only used 2	$4
10MΩ Resistor	1	Provided
1MΩ Resistor	1	Provided
100kΩ Resistor	1	Provided
10kΩ Resistor	1	Provided
1kΩ Resistor	2	Provided
100Ω Resistor	1	Provided
Total		$112.25

III. PCB Fabrication

PCB Layout

For our PCB design, I utilized a website called EasyEDA which allowed us to create both a schematic and a PCB layout, including the extra feature of previewing the design in 3D. Through experimentation with this website, I discovered that using LABELs and organizing each component by its functionality significantly improves debugging time, comprehension, and visual clarity. This is illustrated in Figure 15, which shows the schematic of the capacitance meter. The schematic includes sections for autoranging, relay connections, LCD connections, Arduino connections, power and ground connections, and the unknown capacitor.

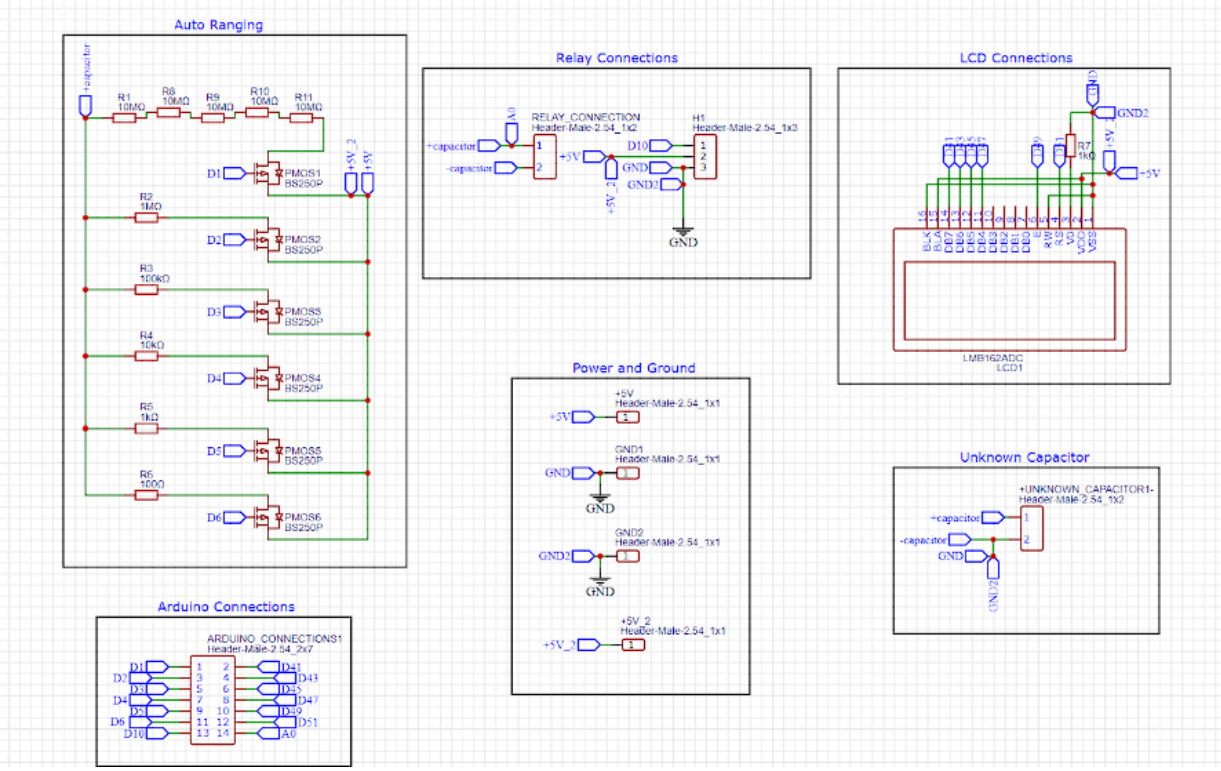


Figure 15: Schematic of Capacitance Meter without the PCB

When designing PCBs, there's multiple factors that I have to take into account such as cost, component placement, layout of the PCB, routing check, and more. One of our primary considerations was the cost, as larger boards are more expensive. To minimize expenses, I excluded the Arduino Mega from the board, which reduced the size by approximately 4.5 inches by 2.1 inches. Instead, I used male header pins and different types of wires to connect the PCB to the Arduino board.

The board arrangement is also vital to consider. I positioned the LCD at the very top. I included two male pins for the alligator clips, which would be used to measure the unknown capacitance. In the center, I added the relay, which is connected to the board, to discharge the capacitance. The autoranging components are next, followed by the pin headers that connect the

PCB to the Arduino and provide power and ground connections. This setup is shown in Figure 16, with the PCB layout on the left and the grounded layers on the right.

The trace that I selected is 0.254 mm, or 10 mils, as seen in Figure 16. I were able to auto-reroute the traces with EasyEDA, which simplified the arrangement. One component that did worry us was the measurements for the relay module's holes; I were unable to locate the precise part on easyEDA and had to use the Collaboratory's caliper to measure the component's length instead.

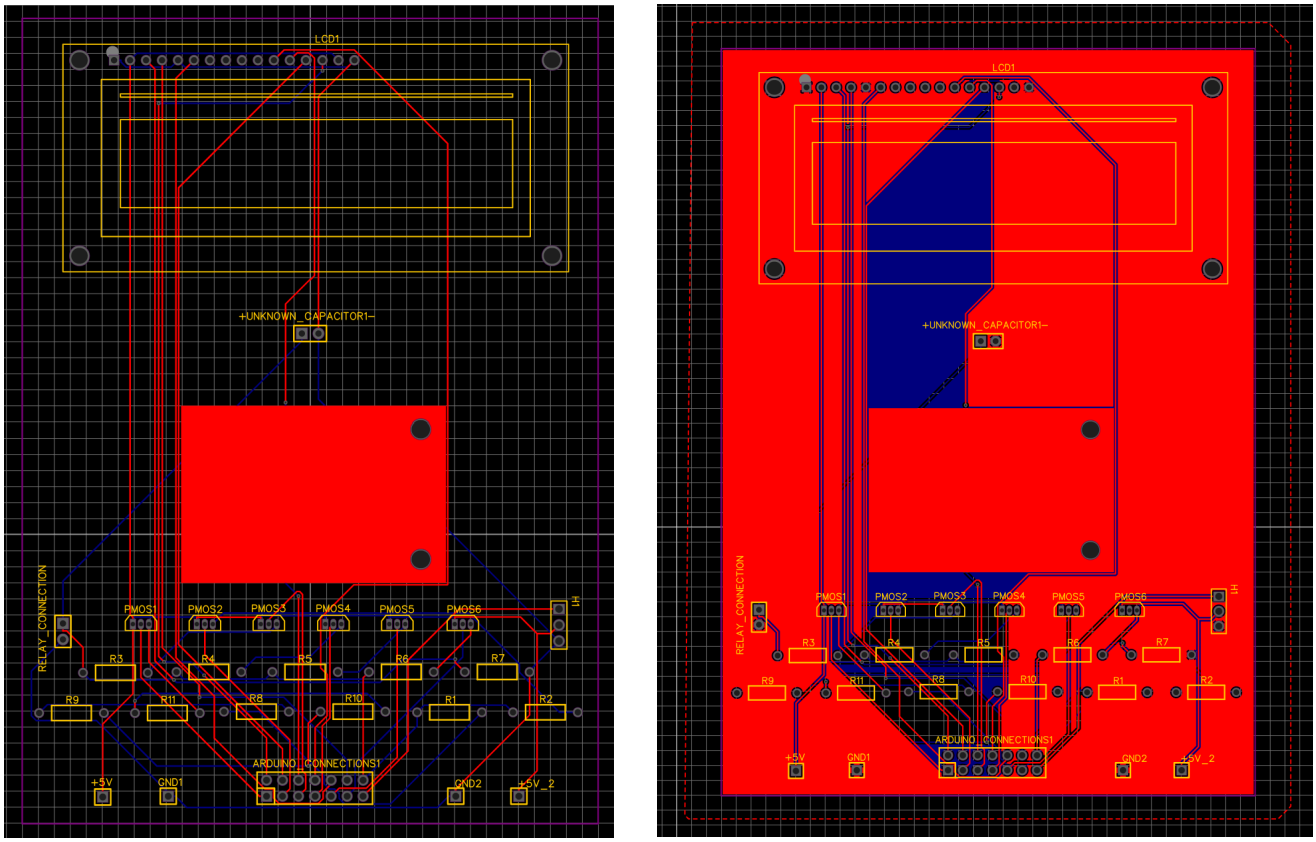
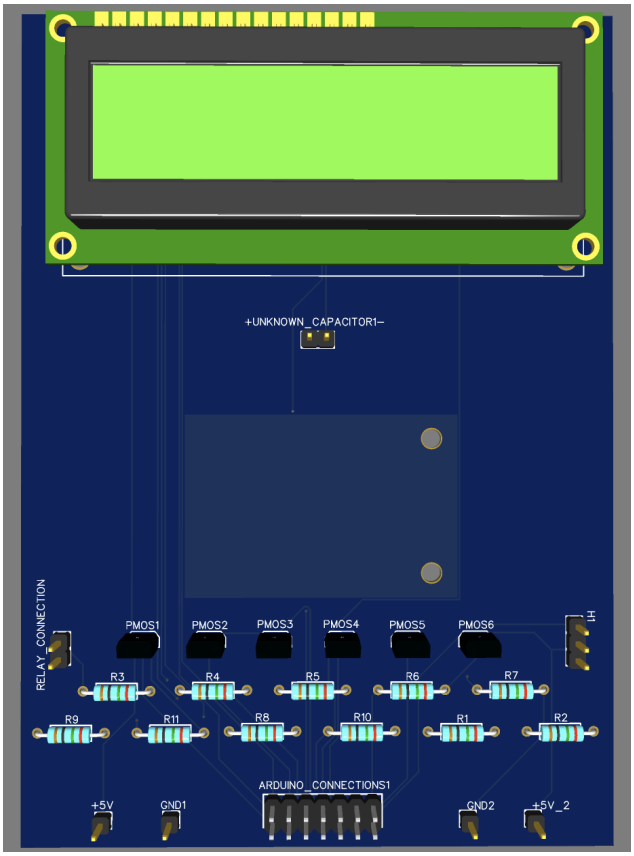


Figure 16: PCB Layout with the 2 Layers Grounded

Figure 17 displays the printed form of the PCB and its 3D model. The printed version was produced by PCBway, who has a document reviewer who checks all the documentation before approving orders.



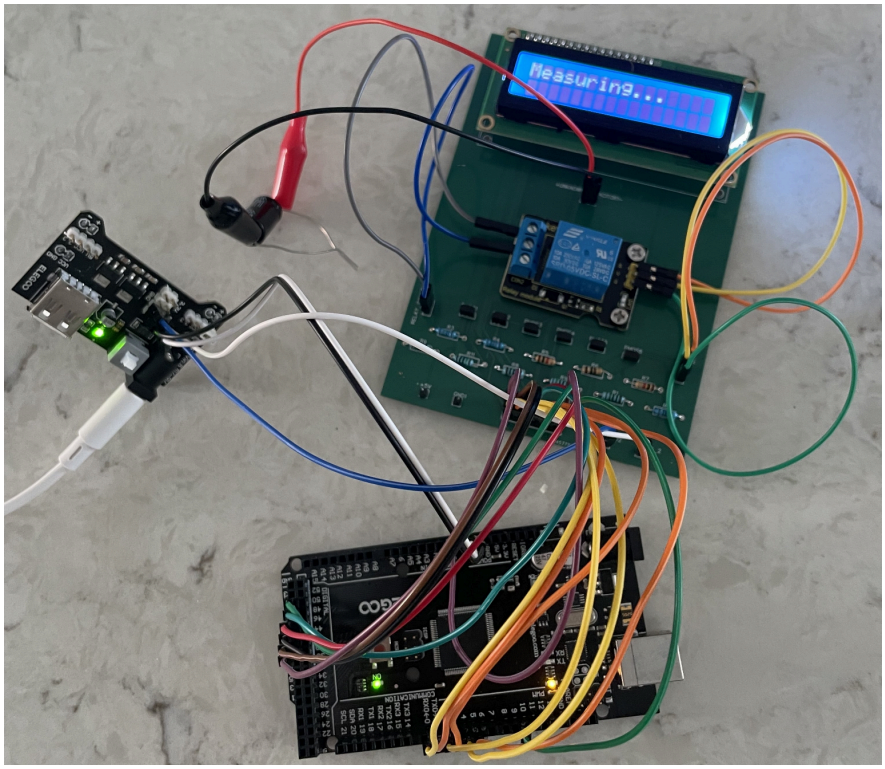


Figure 17: PCB Layout shown in 3D Model and the Printed PCB board

IV. 3D Model Box

Figure 5 shows the 3D model of each piece of the box. The left one in the photo is the main box with the width of 12.2 cm, length of 23.2 cm, and height of 7.2 cm. The middle photo uses the lid of the box with the same measurement as the main box, but only has a height of 3 mm. Lastly, the last photo is a dongle to press the buttons needed. When making the 3D Box, we used the website Fusion360 and printed it at the Collaboratory Place.

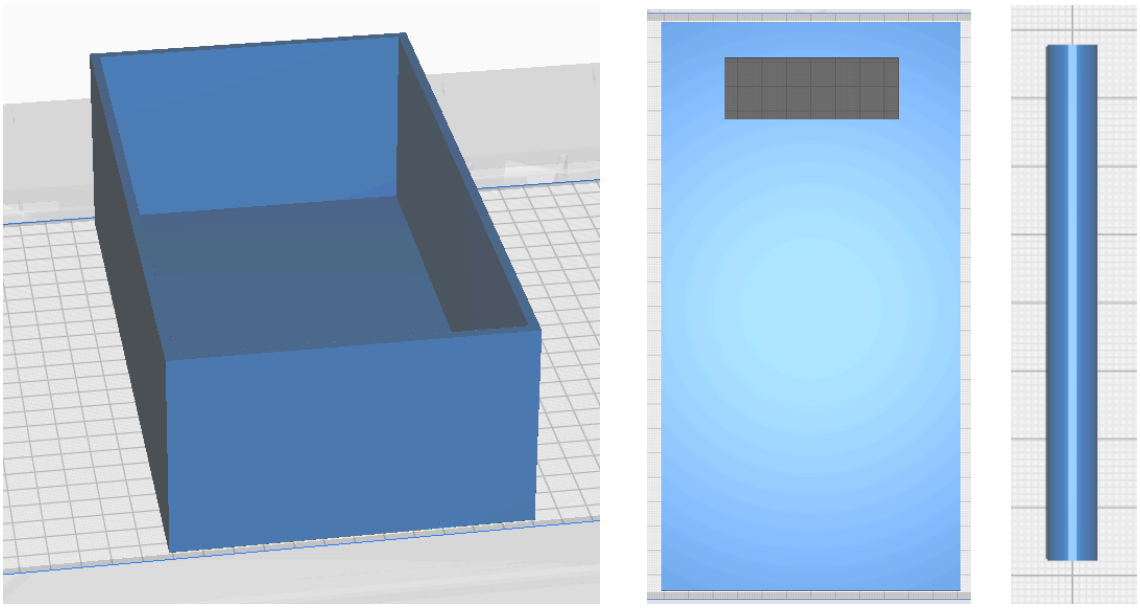


Figure 18: 3D Model of the Box

The finished item, which includes every component, is seen in Figure 19. While the dongles and lid took eleven hours to build, the main box took roughly five hours. The tiny nozzles used in the 3D printer, which produce a better-looking output, are the reason it took so long. Figure 20 shows the box layout. The holes are for the arduino and power module buttons.

Figure 19: Finished Product of the Box

V. Bibliography

- Dr. Berger, A. who helped us with thinking about the factors we needed to take an account when designing this project
- Dr. Decuir, J who helped us with all of our questions and confirmed if our design will work.

VI. Appendix

- Lingib, and Instructables. “Capacitance Meter.” *Instructables*, 12 Jan. 2023, www.instructables.com/Capacitance-Meter-1/. Accessed 06 June 2024.
- *BS250P*, www.diodes.com/assets/Datasheets/BS250P.pdf. Accessed 06 June 2024.
- *SRD-05VDC-SL-C-Datasheet.Pdf*, www.circuitbasics.com/wp-content/uploads/2015/11/SRD-05VDC-SL-C-Datasheet.pdf. Accessed 06 June 2024.
- *SRD-05VDC-SL-C-Datasheet.Pdf*, www.circuitbasics.com/wp-content/uploads/2015/11/SRD-05VDC-SL-C-Datasheet.pdf. Accessed 06 June 2024.
- “Arduino Mega 2560 REV3.” *Arduino Official Store*, store.arduino.cc/products/arduino-mega-2560-rev3. Accessed 06 June 2024.