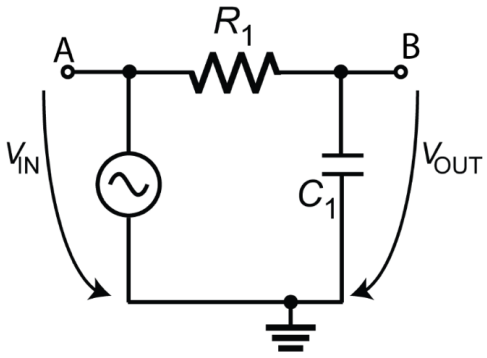
Capacitor Charging and Discharging

A first order circuit[1] is a circuit that only contains one energy storage element (either a capacitor or an inductor). Lab 2 focuses on a first-order circuit where we have a voltage source in series with a resistor and a capacitor. Imagine switching on V_{IN}. Immediately after initially applying V_{IN} to our circuit, all of V_{IN} appears across the resistor because the initially zero voltage across the capacitor cannot change instantaneously (unless, of course, we had an infinite current available). Per Ohm's law, V_{IN} across the resistor produces a current of V_{IN}/R Amperes, which begins to charge the capacitor. As the capacitor charges, however, the voltage across the resistor drops by Kirchhoff's Voltage Law, thereby decreasing the current and the rate of change in capacitor voltage. This process produces an exponential slowing in the rate of increase of the capacitor voltage.



The charging rate in an RC circuit depends on the product of resistance and capacitance, which is typically called the "time-constant", represented by the Greek letter tau:

$$\tau = RC$$

In one time-constant, a charging capacitor will move 63% of the way from its current voltage to the voltage applied through the resistor. The capacitor voltage, V_{OUT}, in an RC circuit with an applied step voltage of V_{IN} is governed by the following equation:

$$V_{OUT}(t) = V_{IN}\left(1 - e^{-\frac{t}{\tau}}\right)$$

In one time-constant the capacitor reaches approximately 63% of the applied voltage, and in five time-constants it reaches approximately 99.3% of the applied voltage. The time it takes to fully charge a capacitor, from a practical perspective, is often taken to be five time-constants. Assuming $R_1 = 100k\Omega$ and $C_1 = 47\mu F$, the time-constant then is $\tau = 100k\Omega \times 47\mu F = 4.7s$, making the time to fully charge the capacitor equal to about 23.5 seconds. Time-constants are the same for charging and discharging, so it would take about 23.5 seconds to fully discharge the capacitor when V_{IN} is reduced to zero volts.

[1] Portions of this lab builds upon wiki.analog.com/university/courses/engineering_discovery/lab_2

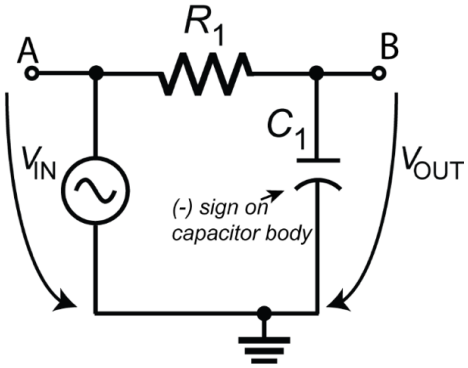
LAB 2

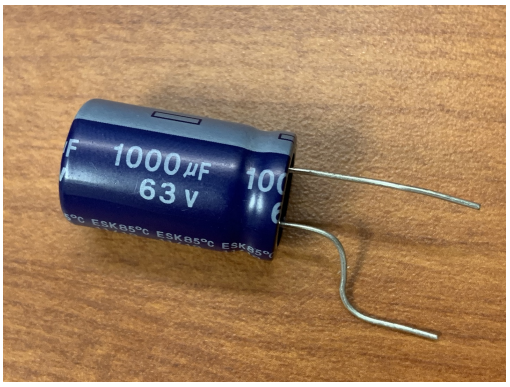
In this lab similar to lab 1, we'll be using Arduino as a replacement for Oscilloscope, allowing us to observe the behavior of RC circuits in real-time. Please download and open the Arduino script named "Capacitor.ino" from Canvas. Don't worry if you are not familiar with C++ code! This is a generic script with a few adjustable values in the first 4 lines, and throughout this lab you only have to understand the purpose of these values and be able to edit them as needed. The code is also included below, however, copy-pasting code from a PDF file might not always work as expected.

```
#define SAMPLING_PERIOD 100             // Time between consecutive analog readings in milliseconds
#define TOTAL_SAMPLES   0               // Total number of analog readings, 0: unlimited, 500: one window
#define INITIAL_STATE   0               // Initial state of the capacitor, 0: none, 1: charged, 2: discharged
#define PULSE_PERIOD    0               // Period of the pulse waveform in milliseconds, 0: no pulse

#define BAUD_RATE       2000000         // Serial transfer rate in bits/s

unsigned long startTime   = 0;
unsigned long currentTime = 0;
unsigned long counter     = 0;
unsigned int  pulse       = 0;

// Custom function for reading the analog voltage on A0  and sending out via serial port to PC
void measure() {
    int val = analogRead(A0) / 1023. * 5000;                // Read analog value and convert it to units of mV

    Serial.print(val);                                      // Send out the read value of A0 over serial port to PC

    if (PULSE_PERIOD) {                                     // Send out the value of the pulse on A1 to PC
        Serial.print(" ");
        Serial.print(pulse * 5000);
    }

    Serial.println();
    counter += 1;                                           // Count total number of measurements
}

// Arduino setup function, called once when Arduino turns on
void setup() {
    if (PULSE_PERIOD) {                                     // Prepare for outputting pulses on A1 if set
        pinMode(A1, OUTPUT);
        digitalWrite(A1, LOW);
    }

    if (INITIAL_STATE) {                                    // Set initial state for the capacitor (charged or discharged)
        pinMode(A0, OUTPUT);
        digitalWrite(A0, HIGH ? INITIAL_STATE==1 : LOW);
        delay(500);
    }

    Serial.begin(BAUD_RATE);                                // Initialize serial communication

    pinMode(A0, INPUT);                                     // From now on A0 is only used for reading voltages
    digitalWrite(A0, LOW);

    if (PULSE_PERIOD)                                       // Send out name of the channels for labeling in Serial Plotter
        Serial.println("A0 A1");

    startTime = micros();                                   // Read the start time and use it as reference

    measure();                                              // Make the first measurement before going to loop
}

// Arduino loop function, called repeatedly until Arduino turns off
void loop() {
  if (TOTAL_SAMPLES && counter == TOTAL_SAMPLES)            // Stop measuring if a limit is set for total number of samples
    return;

  currentTime = micros() - startTime;                       // Get the current system time in microseconds

  if (PULSE_PERIOD) {                                       // Adjust pulse output on A1
    pulse = (currentTime / (PULSE_PERIOD*1000L/2) +
      (INITIAL_STATE+1) % 2) % 2;
    digitalWrite(A1, pulse);
  }

  if (currentTime / (SAMPLING_PERIOD*1000L) != counter - 1)  // Measure analog input on A0
    measure();
}
```

1 Charging Large Capacitor

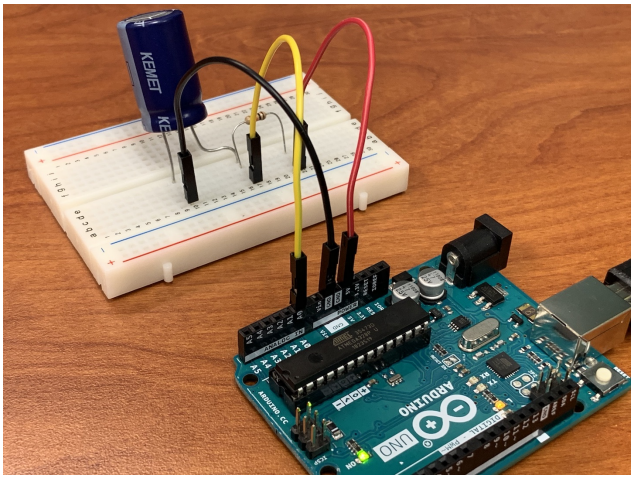
In this exercise our goal is to study charging of a large capacitor from a DC voltage supply in a series RC circuit. Find a $1000\mu F$ capacitor from the extra lab kit (in plastic bag) and try bending its positive pin very gently as shown in the picture below. The capacitors used in this lab are electrolytic capacitors which are polarized. Look for the "$-$" sign on the side of the capacitor, also note that the negative pin is slighter shorter that the positive pin.





1.1. We are trying to make an RC circuit with a time-constant of $10s$, given the $1000\mu F$ capacitor, calculate what should the resistor value be? What are the color bands for this resistor? (take a look at the color table from lab 1)

1.2. Find your calculated resistor from the Electronic Components box of the Arduino kit, and build a series RC circuit on breadboard. Double check for the "$-$" sign. Remember that connecting capacitors in reverse might fry them! (when the voltage exceeds reversal threshold)



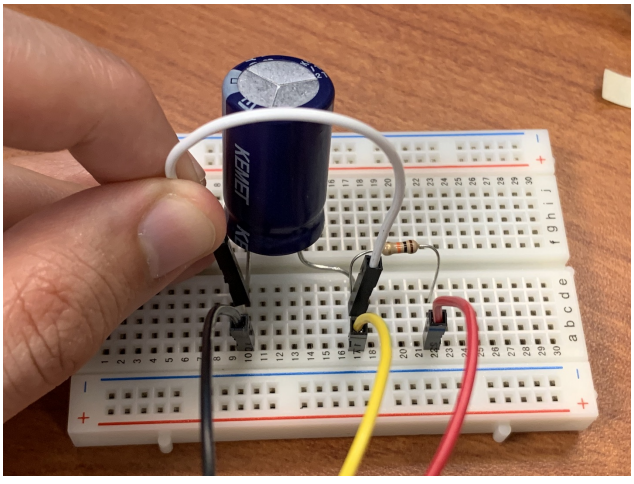
Note that the black wire is connecting GND to the negative pin of the capacitor, the yellow wire is connecting A0 to the positive pin of capacitor, and the red wire is connecting 5V to the right side pin of the resistor.

Snap a clear photo of your circuit on breadboard and its connections to Arduino for your report.

1.3. Connect Arduino via a USB cable to your computer and upload the program (you may use the shortcut Ctrl+U on PC or Cmd+U on Mac), and then open Serial Plotter (the shortcut is Ctrl+Shift+L on PC or Cmd+Shift+L on Mac). Since we will be performing this process (uploading program and opening Serial Plotter) over and over again throughout this lab, using the shortcuts could significantly speed things up for you! From the drop-down list on the bottom left of Serial Plotter, select “2000000 baud”.

What is the voltage of A0 shown on Serial Plotter? Take a screenshot for your report. Briefly explain why the voltage is as you measured it.

1.4. In the previous step your capacitor is already either partially or fully charged. In order to observe the entire charging curve we have to first discharge the capacitor. This is done by simply shorting the two capacitor pins. While keeping Serial Plotter open, use another jumper wire to momentarily connect the two pins of the capacitor to each other like shown in the picture and then remove the wire.



Explain how does the voltage of the capacitor change after removing the white wire? Wait for the curve to complete an entire window and then snap a picture of Serial Plotter. Hint: To pause Serial Plotter for taking a picture or screenshot you can simply unplug the USB cable at the appropriate time!

1.5. What value does the voltage of capacitor seem to be converging to? Write the mathematical formula for the capacitor's voltage, replacing V_{IN} and τ with their appropriate values.

1.6. Based on what you've learned about exponential curves from the lecture, explain how one could measure the actual time-constant of the circuit based on the charging curve you recorded in step 1.4? Given the knowledge that the space between two vertical lines of Serial Plotter is 10 seconds long (the entire length of the window is 50 seconds), roughly estimate the actual time-constant of your RC circuit. How does this compare to your theoretical τ?. Please use your favorite graphical software to mark the points of interest and your estimated values on the curve.

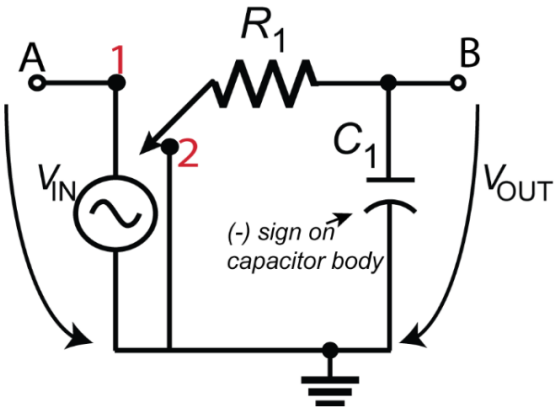
1.7. The Arduino script you are using makes a measurement of the voltage at A0 every $100ms$. This can be changed by adjusting the value of SAMPLING_PERIOD in the first line of the script. Serial Plotter plots these measurements one by one and connects them with a line. (You can see from the labels on the X-axis that between every two vertical line, 100 of these values are getting plotted, and this is where the 10 seconds distance comes from). In order to have a better estimate of the actual time-constant, we have to take the measured values in a professional plotting software, e.g. Matlab or Microsoft Excel. Open up Serial Monitor (using Ctrl+Shift+M or Cmd+Shift+M), and from the drop-down list on the bottom right select "2000000 baud". Short the capacitor by connecting its two pins together momentarily and then disconnect them (similar to step 1.4). Wait about 50 seconds until the capacitor has reasonably reached its final value ($100mV$), then remove the checkbox on "Autoscroll" from the bottom left of screen and use Ctrl+A (or Cmd+A) to select all and Ctrl+C (Cmd+C) to copy the measured voltages in Serial Monitor, and paste them in Excel or Matlab. At the beginning of your data, remove all values before the last 0 so your data starts with only a single 0. Also make sure you don't have too many values at the end that are less than $100mV$ close to your final value.

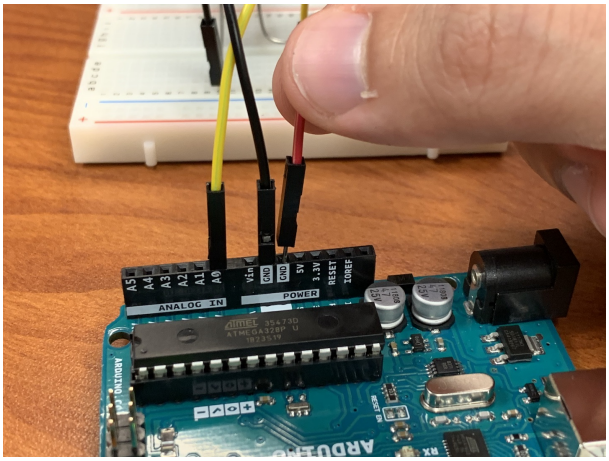
Plot the exported voltage measurements in Excel or Matlab and include it in your report. Estimate how many samples it takes for the voltage of capacitor to change from its initial value (0v) to 63% of its final value. Knowing these samples have been measured $100ms$ apart, what is the new estimated time-constant?

BONUS: Label the X- and Y-axes of your plot appropriately. X-axis should be in units of seconds or milliseconds, and Y-axis in volts or millivolts.

2 Discharging Large Capacitor

So far we've only been characterizing how the capacitor charges when in series with a resistor (discharging was done via shorting). Now lets see how does it behave when discharging in a series RC circuit.





2.1. Reconnect Arduino via USB to your computer, and open Serial Plotter. Wait a bit (about 50 seconds) for the capacitor to become fully charged. Then, Move the red jumper wire from 5V to GND as shown in the picture above and wait for the capacitor to fully discharge. You may want to disconnect the USB to pause plotting. If you missed your chance for taking a screenshot, you can always switch the red wire to 5V, wait a while and the reconnect to GND.

Take a screenshot of the discharge curve with the initial and final voltages of the capacitor fully visible.

2.2. What value does the voltage of the capacitor seem to be converging to? Write the mathematical formula for the capacitor's voltage, replacing V_{IN} and τ with their appropriate values.

2.3. Roughly estimate the actual discharging time-constant based on your captured curve, similar to step 1.6. How does this compare to the charging time-constant?

2.4. In the first line of your Arduino script, change SAMPLING_PERIOD to 500 (i.e. measure every $500ms$ or half a second). We are trying to slow down measurement so we can observe the behavior of capacitor for a longer duration without going outside the bounds of plotting window. Upload your program and open Serial Plotter. Connect the red wire to 5V and wait for the capacitor to get fully charged. Then disconnect the red wire, making sure we have an open circuit without either charging or discharging the capacitor. Observe the voltage of capacitor for about a full time window of the plotting screen.

Explain what is happening to the voltage of capacitor as time goes by. Does it stay the same for an extended period of time? Explain why, and capture a screenshot from the plot for your report.

2.5. Again charge the capacitor fully by connecting the red wire to 5V. When ready, disconnect both the red wire from 5V and yellow wire from A0. Wait about 2 minutes and reconnect the yellow wire to A0.

Ignoring the duration that A0 was disconnected, how has the voltage of the capacitor changed after 2 minutes? Explain why, and capture a screenshot of the plot.

3 Small Capacitor

In this section we will be assessing how a smaller capacitor behaves when in series RC configuration.

3.1. We are trying to build an RC circuit with a time-constant of $1s$. Given the resistor from previous sections, how much should the capacitance value be?

3.2. Find a capacitor with the value you calculated in previous step from the extra lab kit, and replace it with the large $1000\mu F$ capacitor. Since a smaller time-constant means faster change in voltage, we have to also speed up our measurements with Arduino. In the first line of Arduino script, change SAMPLING_PERIOD to 10 (i.e. $10ms$). Connect the red wire to 5V, short the capacitor with another jumper wire, upload program, open Serial Plotter (close and reopen if it's already open) and immediately disconnect the shorting wire.

What do you see on Serial Plotter? How does the voltage of capacitor change? What value does it seem to be converging to?

3.3. An issue with the previous step is that the capacitor charges up rather quickly, and timing the opening of Serial Plotter and disconnecting the shorting wire is tricky. Moreover, the plot window fills up very fast, making it almost impossible to take a screenshot or even disconnect the USB cable in time without the initial value of the curve going out of the plotting window. This issue is solved with "triggering". Both the beginning and ending of the plot should be designated by a trigger. When using an Oscilloscope, there are multiple trigger related settings that allow plotting rapid time-varying circuits very easily. In this lab, we use a smart Arduino script to take care of the trigger for us.

In the second line of script, by setting TOTAL_SAMPLES to 500, we force Arduino to stop sending out values to computer after 500 measurements have been completed. Remember that Serial Plotter has a plotting window that's exactly 500 samples long! We can also use Arduino itself to discharge or charge the capacitor fully at startup using the same A0 pin that we've been using for measurements. In line 3 of the script, set INITIAL_STATE to 2 (i.e. discharged). The first few lines of your code should look like this:

```
1  #define SAMPLING_PERIOD 10          // Time between consecutive analog readings in milliseconds
2  #define TOTAL_SAMPLES   500         // Total number of analog readings, 0: unlimited, 500: one window
3  #define INITIAL_STATE   2           // Initial state of the capacitor, 0: none, 1: charged, 2: discharged
4  #define PULSE_PERIOD    0           // Period of the pulse waveform in milliseconds, 0: no pulse
```

Making sure the red wire is connected to 5V, close Serial Plotter, program Arduino and reopen Serial Plotter.

Take a screenshot of the resulting plot, and roughly estimate the charging time-constant similar to step 1.6. How does it compare to the theoretical value of τ?

3.4. Repeat step 1.7 with the small capacitor: close Serial Plotter, program Arduino, and open Serial Monitor. When Arduino stops sending out values, copy all the recorded values into Excel or Matlab, plot the curve and estimate the time-constant. Note that for this step we have changed the distance between each two measurement to $10ms$.

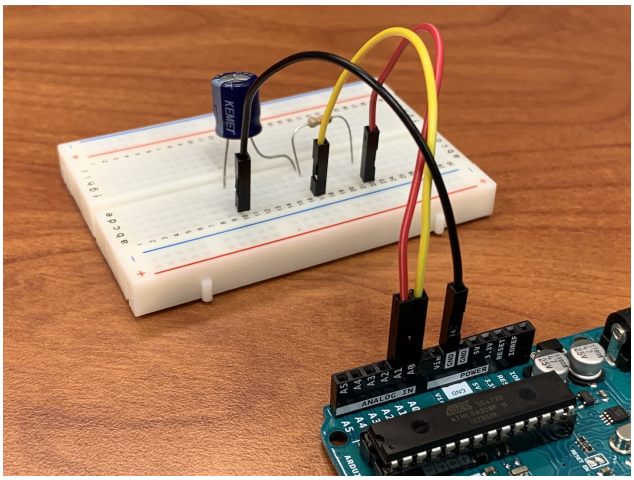
Include the charging plot in your lab report. What is the estimated time-constant?

BONUS: Label the X- and Y-axes of your plot appropriately.

3.5. To observe discharging of the capacitor, connect the red wire to GND, and simply change INITIAL_STATE to 1 (i.e. fully charged). Close Serial Plotter, program, and reopen the Plotter.

Take a screenshot of the resulting plot, and roughly estimate the discharging time-constant similar to step 1.6. How does it compare to the charging time-constant?

3.6. So far by switching the red wire between GND and 5V, we have been manually generating a pulse waveform (i.e. rectangular waveform). As explained in the lecture, in lab we use a Function Generator to produce time varying voltage outputs such as a pulse, sinusoidal or sawtooth waveform. In this lab we resort back to our versatile Arduino which is fully capable of generating a pulse waveform. By changing the value of PULSE_PERIOD in the 4th line of Arduino script we can generate a pulse waveform on pin A1 with almost any desired period (T, duration between two consecutive rises or falls). So, we can connect the red wire to A1 as in the picture below.



Adjust the settings of your Arduino to the values below:

```
#define SAMPLING_PERIOD 50          // Time between consecutive analog readings in milliseconds
#define TOTAL_SAMPLES   500         // Total number of analog readings, 0: unlimited, 500: one window
#define INITIAL_STATE   2           // Initial state of the capacitor, 0: none, 1: charged, 2: discharged
#define PULSE_PERIOD    10000       // Period of the pulse waveform in milliseconds, 0: no pulse
```

Close Serial Plotter, program, and reopen the Plotter.

Take a screenshot of the Serial Plotter and describe what you see. What do the blue and red curves represent?

3.7. Based on the period of the pulse waveform ($T = 1000ms$), what is the frequency of the output pulse? What is the ON duration of the pulse (i.e. the amount time that it's HIGH or at 5v)? What is the OFF duration of the pulse (the time that it's LOW or at 0v)?

3.8. Change the value of PULSE_PERIOD to 5000, 2000, 1000, 500 and 100. For the 500 and 100 periods, set SAMPLING_PERIOD to 10 (keep using 50 for the rest). Each time close Serial Plotter, program and reopen the Plotter. Take screenshots of all 5 curves.

3.9. Describe what happens to the voltage of the capacitor as pulse period is reduced (pulse frequency is increased). Is the capacitor able to follow the pulse voltage faithfully? If not, what value does it seem to be converging to? You may try decreasing the pulse width further down to $1ms$ and maybe decreasing sampling period as well.