ECE 403: Computing and Control

Lab Undergraduate Group 08: Jayant Dubey, Raisha Begum, Obi Odeze

Final Project 2: Change Pole of Low-pass RC Filter

Introduction

An open-loop and closed-loop system with a low-pass RC filter implemented on the output are simulated using Simulink. A low-pass RC filter allows a signal to pass for a range of frequencies lower than the cut-off frequency.

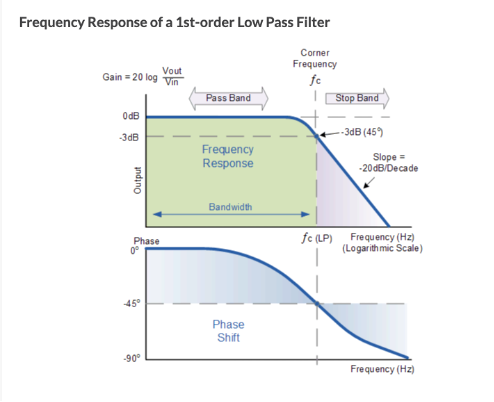


Figure 1. Frequency Response of a 1st order Low Pass Filter

Figure 1 depicts this behavior. One of the defining characteristics of the low-pass RC filter is its pole. The pole affects the system’s phase delay and overshoot, where the phase response is the temporal difference between the input and output signal of the system; and the overshoot is how much the filter’s signal output exceeds the steady-state value before settling. If the pole’s location is changed, subsequently, the cut-off frequency is changed. This results in either greater or lower phase-delay between the system input and output.

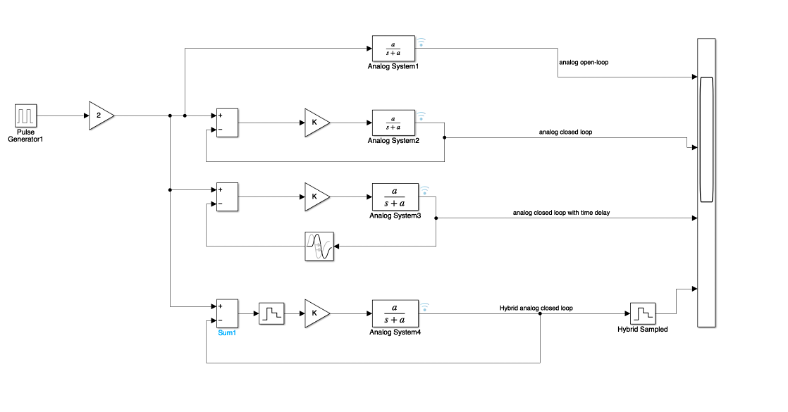


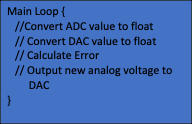
Figure 2. Model in Simulink of Analog Open-Loop, Analog Closed Loop, Analog Closed Loop with Time Delay, Hybrid Analog Closed Loop

Figure 2 shows the following systems that will be modeled in Simulink. The systems will be analyzed for their system response to a step-input and a pulse-input. The models under consideration are split into two categories. The first category is the open-loop system implemented in analog and hybrid (both digital and analog) versions. The open-loop system does not have feedback, so the system does not have the ability to take its output into account and generate a new signal that’s closer to the input signal.

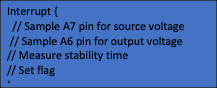
The second category is the closed-loop system, also implemented, in analog and hybrid versions. Within this category, all versions will be modeled with and without time-delay. This is important because the hybrid system will also be built and analyzed in a lab using the Arduino as a proportional-gain controller. The circuit will consist of a resistor and capacitor linked in series, one op-amp placed at the output to realize the gain, and another unity-buffer op-amp on the feedback. Since the hybrid system contains a computer-in-the-loop (Arduino), a time-delay is introduced by the information processing time of the Arduino. The Arduino is responsible for sampling the input and output voltage, calculating the error, multiplying the error by a gain, and outputting a proportional response via A/D converter. It is necessary to include time-delay in the Simulink model for this reason.

Code Implementation

The block-diagram of the code to process the information in the Arduino controller is shown below.







Code

An interrupt samples Arduino’s internal, 10-bit A/D at 500Hz (400Hz). The source voltage is read on pin A0, while the output voltage is read on pin A7. The ADC multiplexer and internal circuitry needs time to switch and stabilize, therefore it is necessary to read the pins multiple times. The interrupt service was timed and took approximately 20ms, and therefore it was necessary to decrease the sample frequency to 400Hz.

This allows for both source and output voltage values to be read and stored in the interrupt. The main loop processes the information and performs calculations. Stability of the system is detected if the error between the input and output signal of the system is less than 0.01V. It must be read less than 0.01V 3 times, for it to be confirmed. Once confirmed, the time to reach the steady state is recorded and displayed.

Build Setup

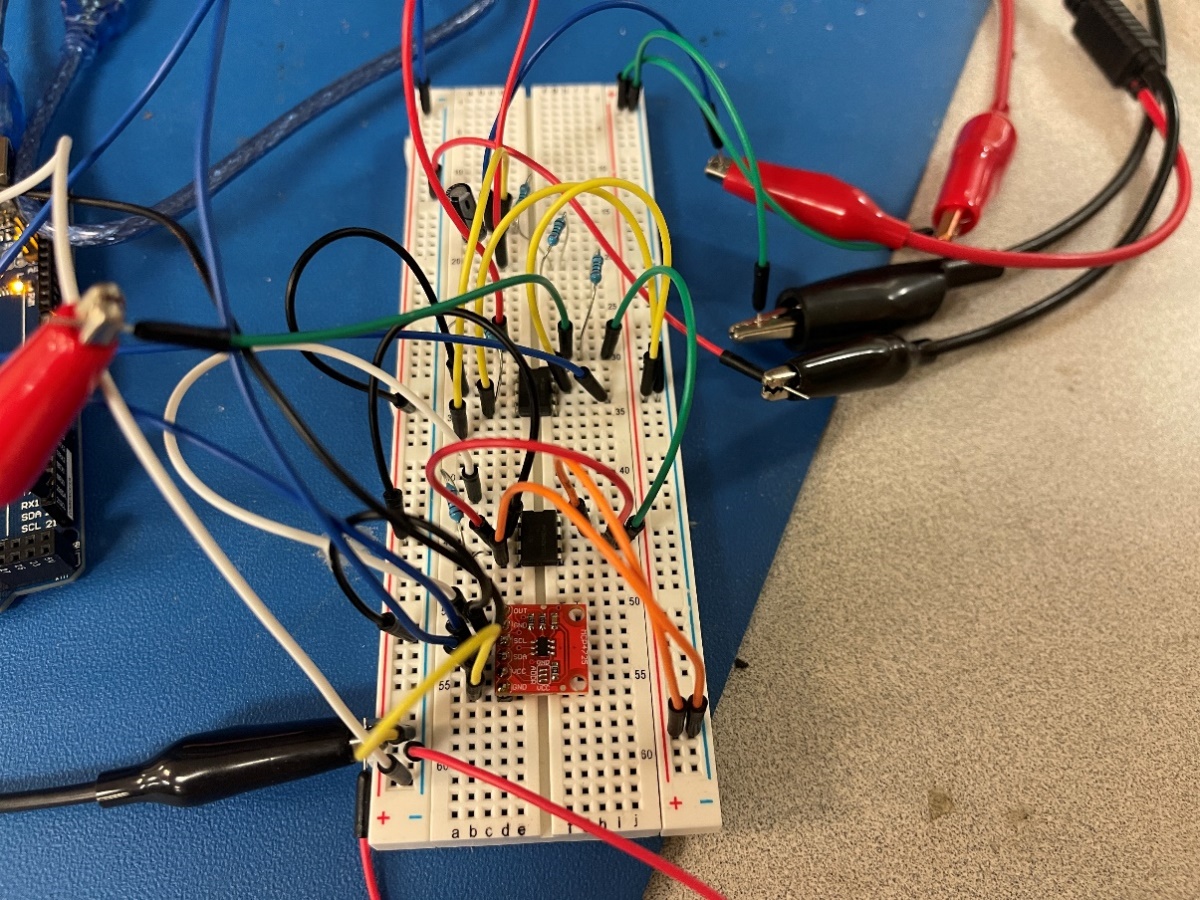


Figure 3. Technical Setup with Focus on Mega 2560 Board

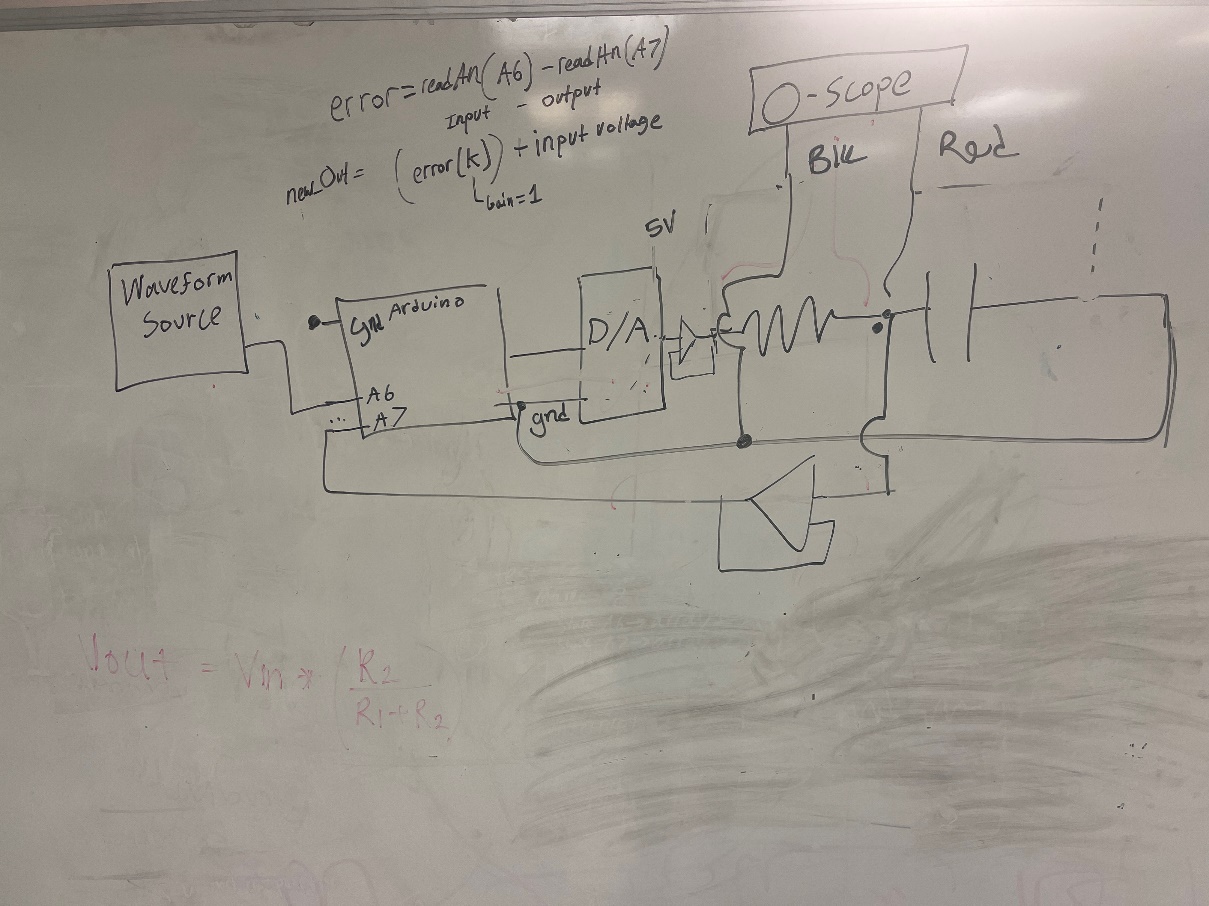
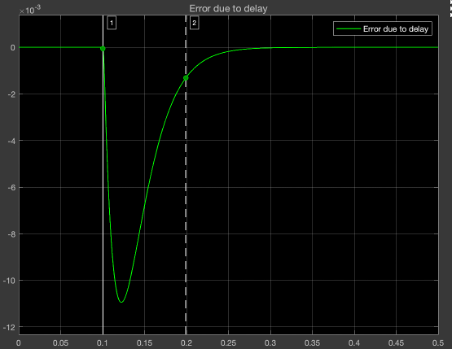
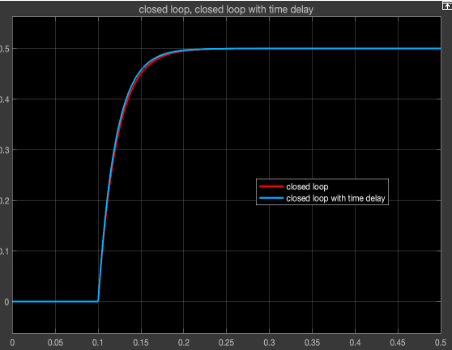


Figure 4. Schematic Configuration of Technical Setup

Simulation/Experiment Results with Discussion

Analog-Only System:



5A. 5B.

Figure 5A. Step response for closed-loop and closed-loop with time delay. Figure 5B. Error between the closed-loop analog system with and without time-delay. K = 1, T = 400Hz, a = 23.09 rad/s.

The time-delay response is faster resulting in larger values than the closed-loop system without time-delay. As a result, the error values are negative.

Pulse response:

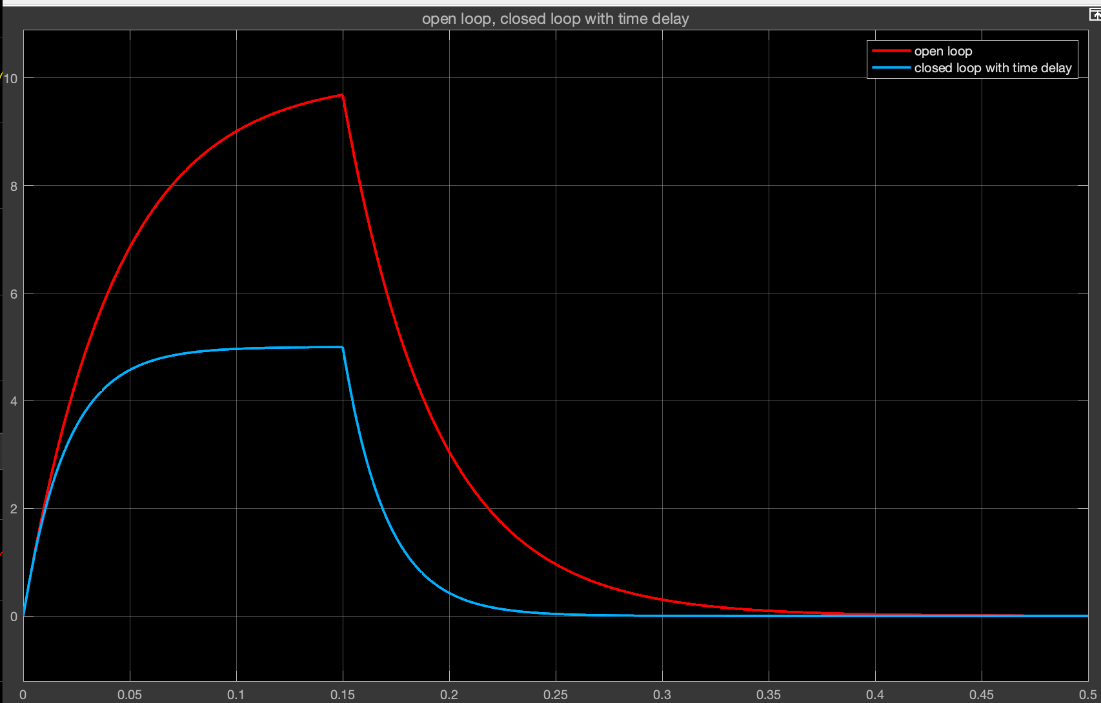


Figure 6.

Figure 6. Pulse response of open-loop and closed-loop system with time-delay. Pulse length = 0.15 seconds, Pulse Amplitude: 5V, K = 1, T = 400Hz, a = 23.09 rad/s.

It can be seen that the proportional-gain controller is working because the steady-state is achieved much faster than the open-loop system. The product of the gain and error is summed with the input until the error reaches zero. The capacitor discharge is also steeper in the closed-loop system because input voltage is zero when the pulse is terminated.

Hybrid vs. Analog Comparison:

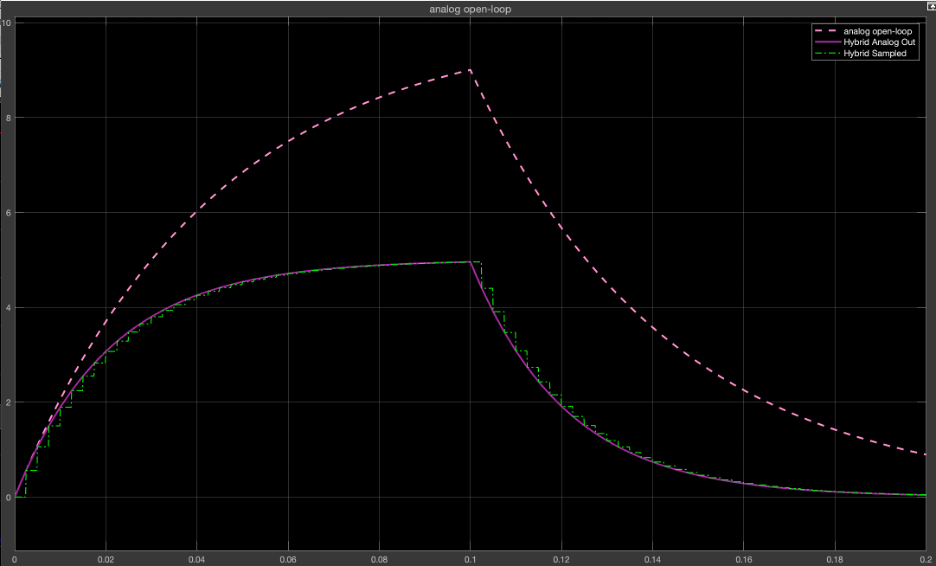


Figure 7. Pulse response of open-loop and hybrid closed-loop system with time-delay. Pulse length = 0.15 seconds, Pulse Amplitude: 5V, K = 1, T = 400Hz, a = 23.09 rad/s.



Figure 8. Hybrid sampled (green), analog closed loop with delay (pink), discrete model (white). The quantizer represents the D/A conversion which is sampled by a zero-order hold.

Experiment Data in comparison to Simulink Hybrid Model:

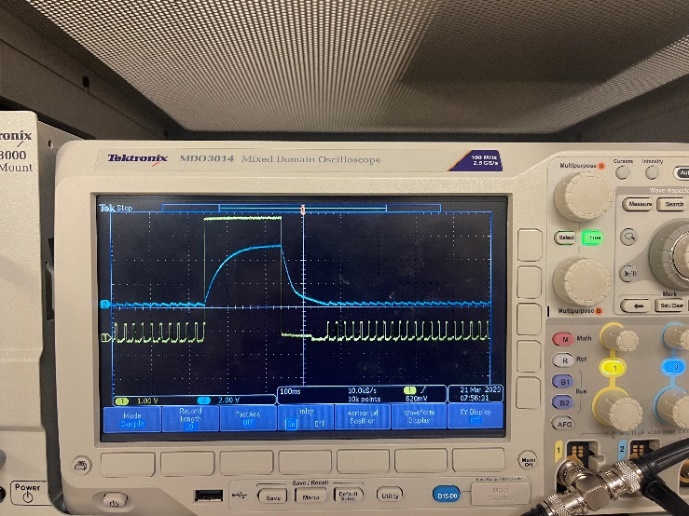
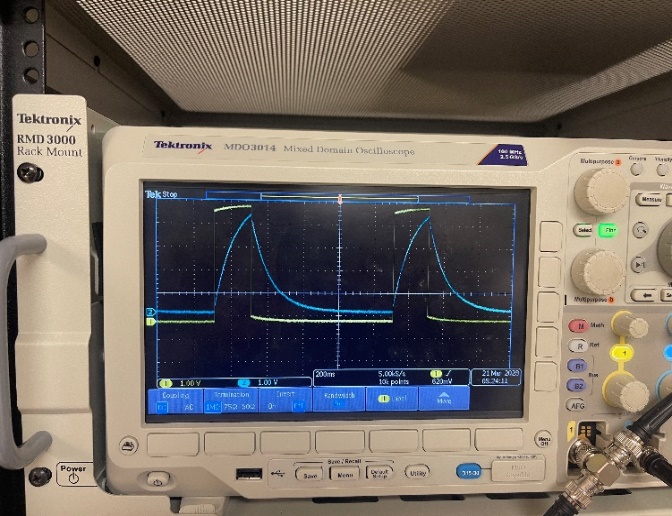


Figure 9. Oscilloscope readings of the implemented model. The left-side image is an open-loop system. The right-side image is the closed-loop with the Arduino. The scale of the closed-loop system is enhanced to show the plateauing effect.

The implemented model shows a strong attenuation of the signal to 5V amplitude and steep drop-off to zero after the pulse has terminated. This shape is similar to the effects we saw in the Simulink model. Unfortunately, the timing data could not be displayed due to the serial monitor’s use of a timer. Additionally, the code’s run time is long in processing, therefore measuring time or using delay statements caused problems.

Conclusion:

An analysis of the RC-filter as the output of a closed-loop system is conducted by simulating the system in Simulink, then building and detecting the voltage outputs. The metrics of the code are not included due to the sensitivity of information processing, whereby, the time it takes to read two analog pins of the Arduino is close to the interrupt timer period. This caused issues in storing and displaying measurements. However, when it came to building the system, the code implemented, is shown to attenuate the signal to the desired setpoint which is the amplitude of the pulse.