Trenton Cathcart
Date: 11/1/2023

Motor Transfer Function, Second Order System and PD Controller Design

Table of Contents

I. List of Tables:

II. List of Figures:

III. Objectives:

In this laboratory experiment, our aim was to determine the positional transfer function for the QUBE-Servo and subsequently design and implement a controller that facilitates precise position control of the motor. This experiment built upon the models created in prior sessions, allowing us to delve deeper into the control systems realm, focusing on second-order systems and the PD (Proportional-Derivative) controller design.

IV. Equipment Used:

- Matlab Program
- Simulink and Quarc Software
- Computer

V. Background Theory:

The Quanser QUBE-Servo 2 is a rotary servo system with its DC motor shaft connected to a load hub, and a disk load attached to the output shaft. The back-emf voltage of this system is influenced by the motor's speed and the back-emf constant. Using Kirchhoff's Voltage Law, we formulated an equation relating motor voltage, resistance, current, and back-emf. We then explored second-order systems and their step responses, understanding the dynamics affected by natural frequency and damping ratio. In the context of this experiment, we used a unity feedback loop for controlling the position of the QUBE-Servo. Finally, we delved into PD control, where a QUBE-Servo 2 voltage-to-position transfer function was established, and the functionality of the PID controller was analyzed with an emphasis on the potential challenges posed by real-world noise in measured signals.

VI. Preliminary Calculations:

No Preliminary calculations required.

VII. Procedure/Result/Analysis:

I. First Principles Modeling:

In the initial stage, we used models that were constructed in the 6th experiment. Our objective was to devise a model capable of applying a 1-3 V, 0.4 Hz square wave to the motor and simultaneously read the servo velocity using the encoder.. A subsystem named 'QUBE-Servo 2 Model' was created, as represented in Figure 7.6 in the lab manual. This subsystem incorporated essential blocks to emulate the QUBE-Servo system's behavior. Using the equations delineated in Section II, we built a block diagram in Simulink, ensuring the inclusion of crucial components such as Gain blocks, a Subtract block, and an Integrator block, aiding in transitioning from acceleration to speed.

To enhance efficiency and avoid numerical value entries in Gain blocks, we deemed it beneficial to script a brief MATLAB code. This script primarily sets various system parameters within MATLAB, enabling the use of symbols. In 1.3, we built and executed the QUARC controller in tandem with our QUBE-Servo 2 model. We anticipated the scope response to mirror the one presented in Figure 7.8 from the lab manual. A screen capture of our scopes taken. Based on our observations and evaluations, we deduced that our model represented the QUBE-Servo 2's characteristics. The results from this phase of our experiment can be seen from **Figure 1** provided below.

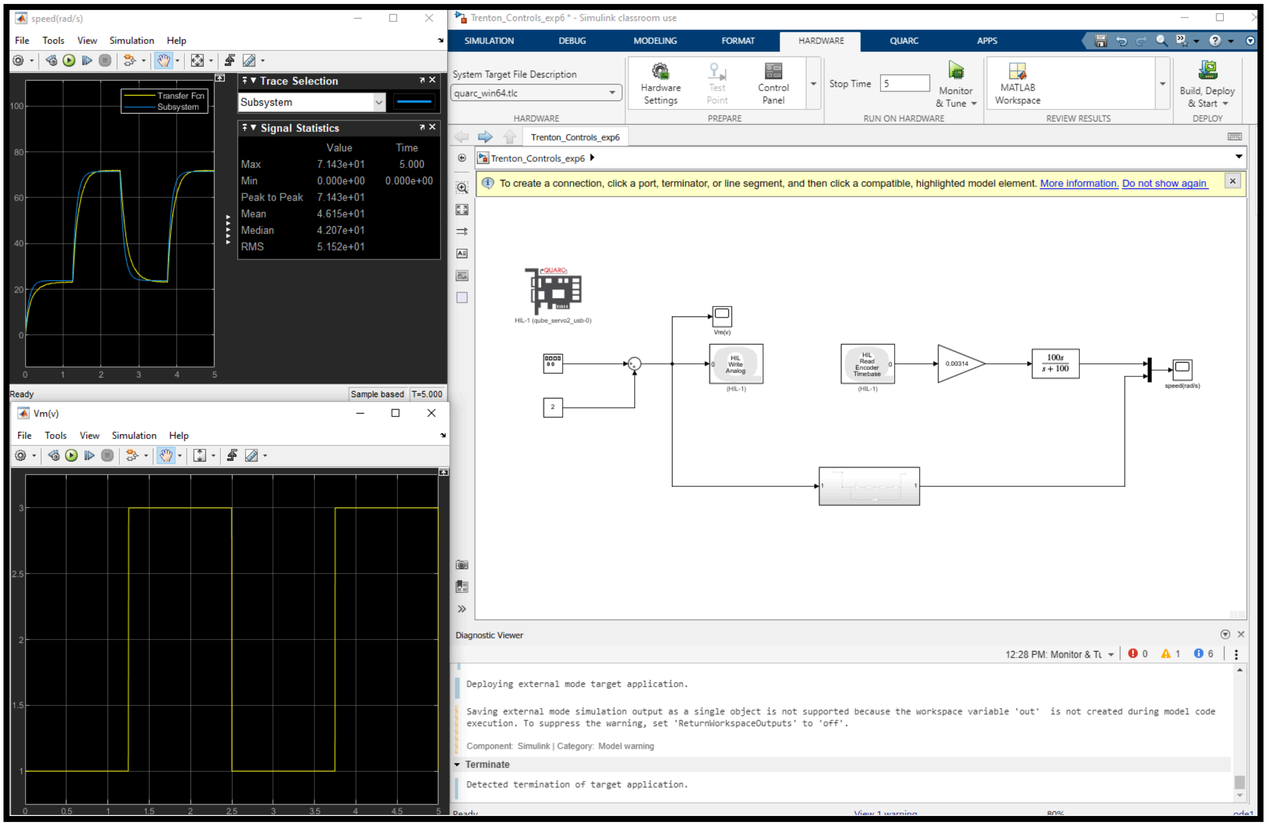

Figure 1: First Principles Simulink and Output

II. Second Order Systems:

Part 2: Unity Feedback Control and Analysis of the QUBE-Servo 2 System

In this phase, we designed a Simulink model as depicted in the manual. This design was focused on implementing the unity feedback control within Simulink. We applied a step reference, which represents the desired position or setpoint, of 1 rad at 1 second, allowing the controller to operate for 2.5 seconds.

In the subsequent section, 2.1, we examined the QUBE-Servo's closed-loop equation under unity feedback as presented in Equation 7.15 from the lab manual. Using either the accurate parameters from experiment #6 or the default parameters provided in section III, we successfully determined the system's natural frequency and damping ratio.

For section 2.2, armed with our calculated values for ωn and ζ, we projected the expected peak time and percent overshoot for the system. This helped set the benchmark for the expected system response.

Moving to 2.3, we constructed and executed the QUARC controller. We anticipated our scope results to resemble the patterns shown in Figure 7.10.

In 2.4, we presented the QUBE-Servo's position response, ensuring it showcased both the setpoint and measured positions within a singular scope. Moreover, we provided details on the motor voltage, aligning with the visual references provided. For data retention for offline scrutiny, we leveraged the QUARC documentation, which enabled us to generate essential MATLAB figures via the MATLAB plot command.

Section 2.5 involved the empirical measurement of peak time and percent overshoot from our system's response. We then contrasted these real-world results with our earlier predictions. For precise measurements, we employed the cursor measurements tool within the scope image, which facilitated direct data acquisition from the scope.

Finally, in 2.6, we pondered upon potential reasons for discrepancies between our predicted values for peak time and percent overshoot and the actual measured results. One plausible explanation we surmised is that inherent system nonlinearities or unaccounted external disturbances might have impacted the QUBE-Servo's response, leading to variations from our initial calculations.

From our endeavors, we observed that the Delta y, representing the overshoot, was 0.3485 or equivalently 34.85%.

The detailed calculations and system responses from this segment of our investigation can be observed in **Figures 2 and 3** provided below.

2.1

Equation 7.15

$$\frac{Y(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

$$2\zeta\omega_n = 1/\tau \rightarrow \zeta = 0.289$$

$$\omega_n^2 = K/\tau \rightarrow \omega_n = 13.3$$

$K = 23$ rad/sec

$\tau = 0.13$ s

Equation 7.9

$$\frac{\theta_d(s)}{V_m(s)} = \frac{K/\tau}{s^2 + s/\tau + K/\tau}$$

2.2

$$T_p = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}}$$

$$T_p = 0.247 \text{ sec}$$

$$\%OS = 100e^{-\pi\zeta/\sqrt{1-\zeta^2}}$$

$$\%OS = 38.74\%$$

Figure 2: Parameter Calculations

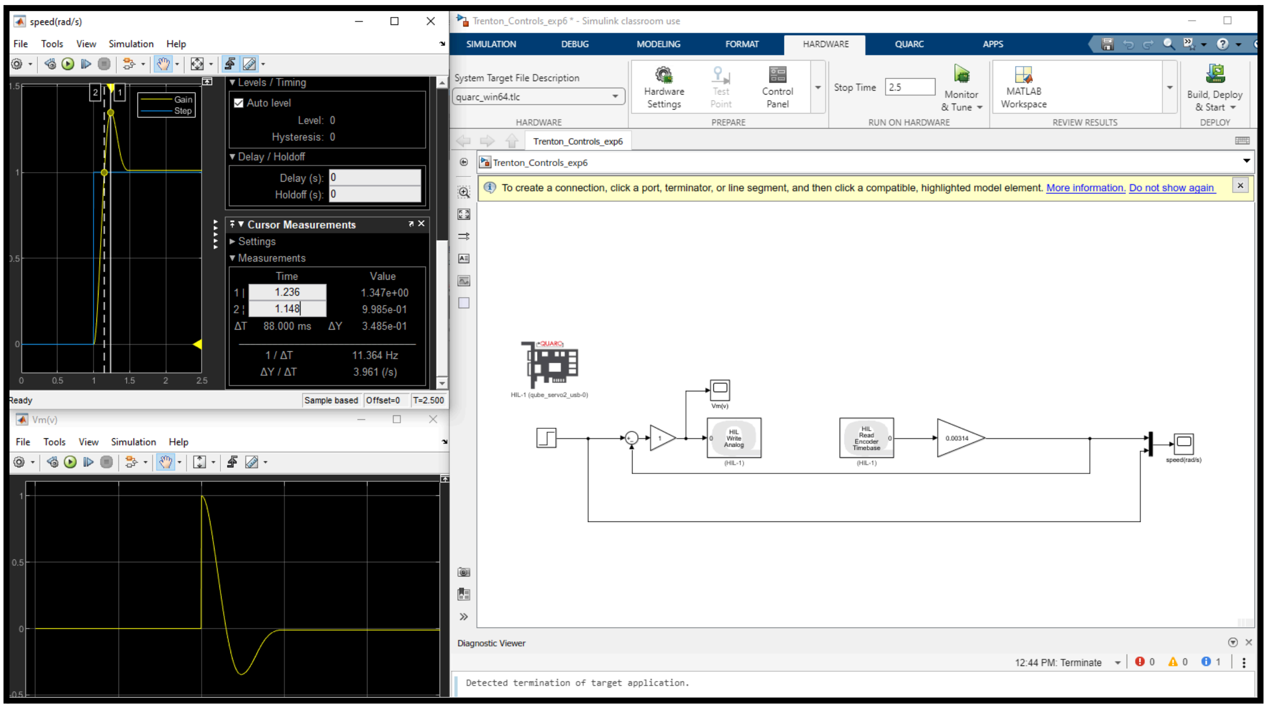

Figure 3: Simulink Model & Oscilloscope Results

III. PV Control:

In this phase, we constructed a Simulink model following guidance from the manual, implementing the proportional plus rate feedback, or PV control. Importantly, this control incorporated a filter alongside the derivative, taking the form 100s/(s + 100) instead of a direct derivative, as shown in **Figure 4**.

In 3.1, we configured the Signal Generator block so that the servo command (the reference angle) materialized as a square wave, having an amplitude of 0.5 rad and a frequency of 0.4 Hz.

Moving to 3.2, we set the proportional gain to kp = 2.5 V/rad and the derivative gain to kd = 0.05 V-s/rad.

In 3.3 and 3.4, we built and ran the QUARC controller, anticipating the output to resemble the response from the manual. The outcomes can be seen in **Figure 5**.

For 3.5, we fixed kp at 2.5 V/rad and set kd to 0. We then adjusted the kp value between 1 and 4 to analyze its effect on the servo position control response. The respective responses are showcased in **Figures 6**, **7**, and **8**.

During 3.6, with kp set at 2.5 V/rad, we modified the derivative gain kd between the values 0 and 0.15 V/(rad/s). We observed the influence of this derivative gain on the servo position control response in **Figures 9, 10**, and **11**.

After concluding our tests in 3.7 by stopping the QUARC controller, we proceeded to 3.8 and 3.9 to calculate the necessary proportional and derivative gains. This was based on the QUBE-Servo 2 system specifications and the desired response characteristics defined in the manual.

With our newly designed gains, we executed the controller in 3.10, producing results displayed in **Figure 12.**

In 3.11, we measured the QUBE-Servo 2 system's percent overshoot and peak time, utilizing the cursor measurements tool for precision. Initially, we observed an overshoot of 1.8%. However, our goal was to match the desired specifications from the manual, specifically, a peak time of 0.15 s and a percent overshoot of 2.5%.

Given our initial results, we embarked on a tuning exercise in 3.12. After some trial and error, we achieved a percent overshoot of 2.5%. The refined response and the steps we took to reach these results are detailed in **Figure 13**.

In summary, through iterative adjustments and evaluations, we fine-tuned our controller to align closely with the desired specifications. The insights gained during this phase greatly enhanced our understanding of the dynamic behavior of the QUBE-Servo 2 system and the effects of different control gains on its performance.

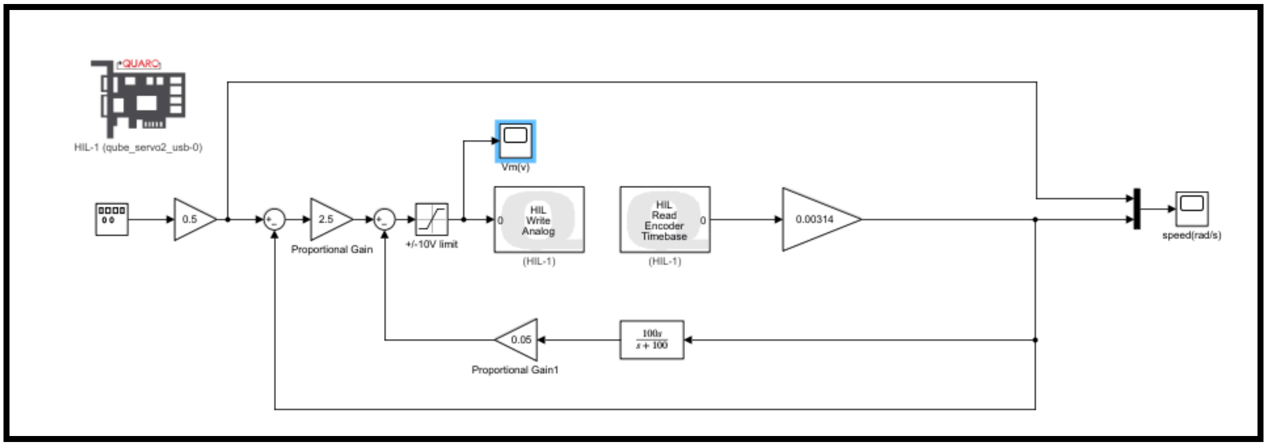

Figure 4: Simulink Model

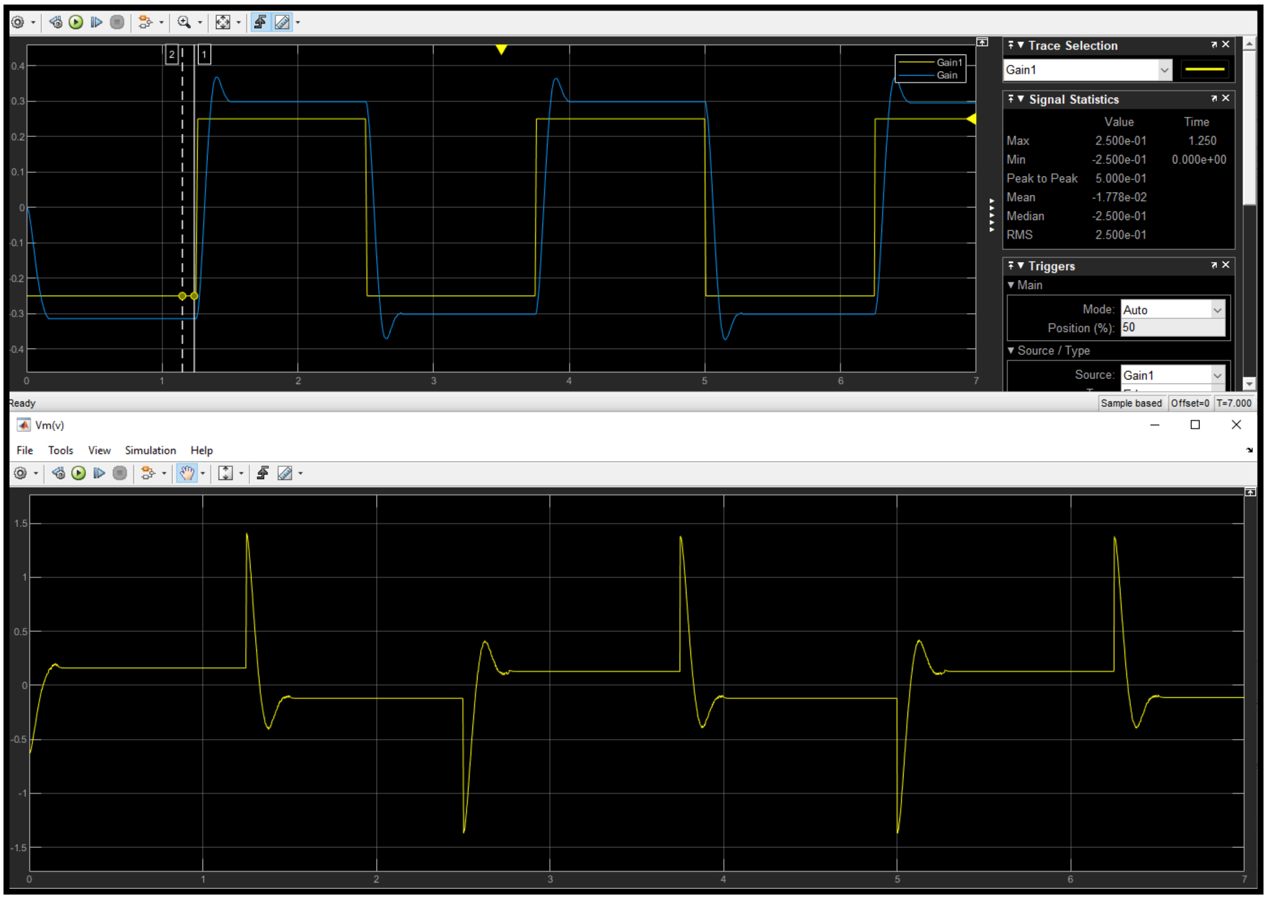

Figure 5: Initial Output

Figure 6: Part 3.5 Output Kd = 0; Kp = 2.5

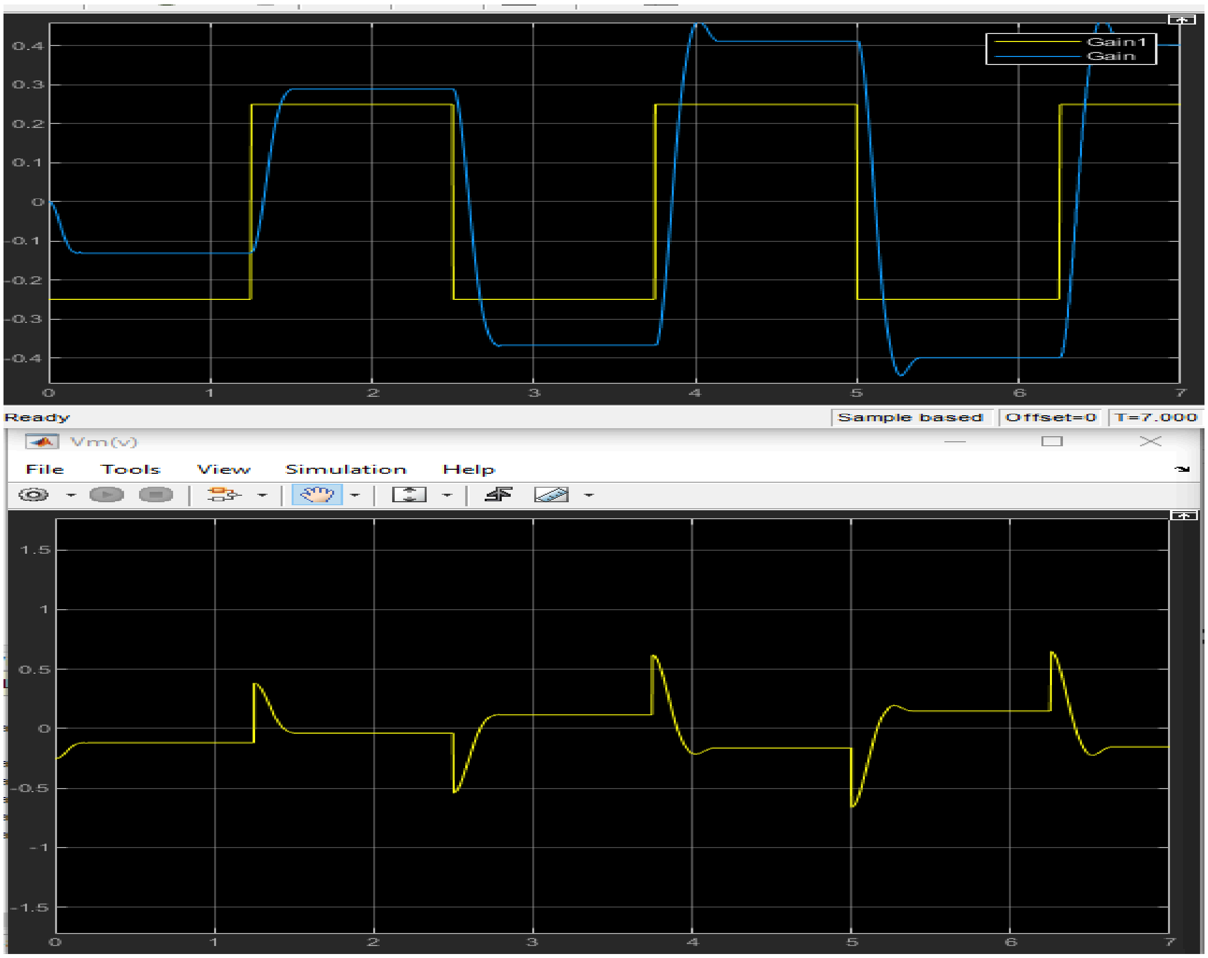

Figure 7: Part 3.5 Output Kd = 0; Kp = 1

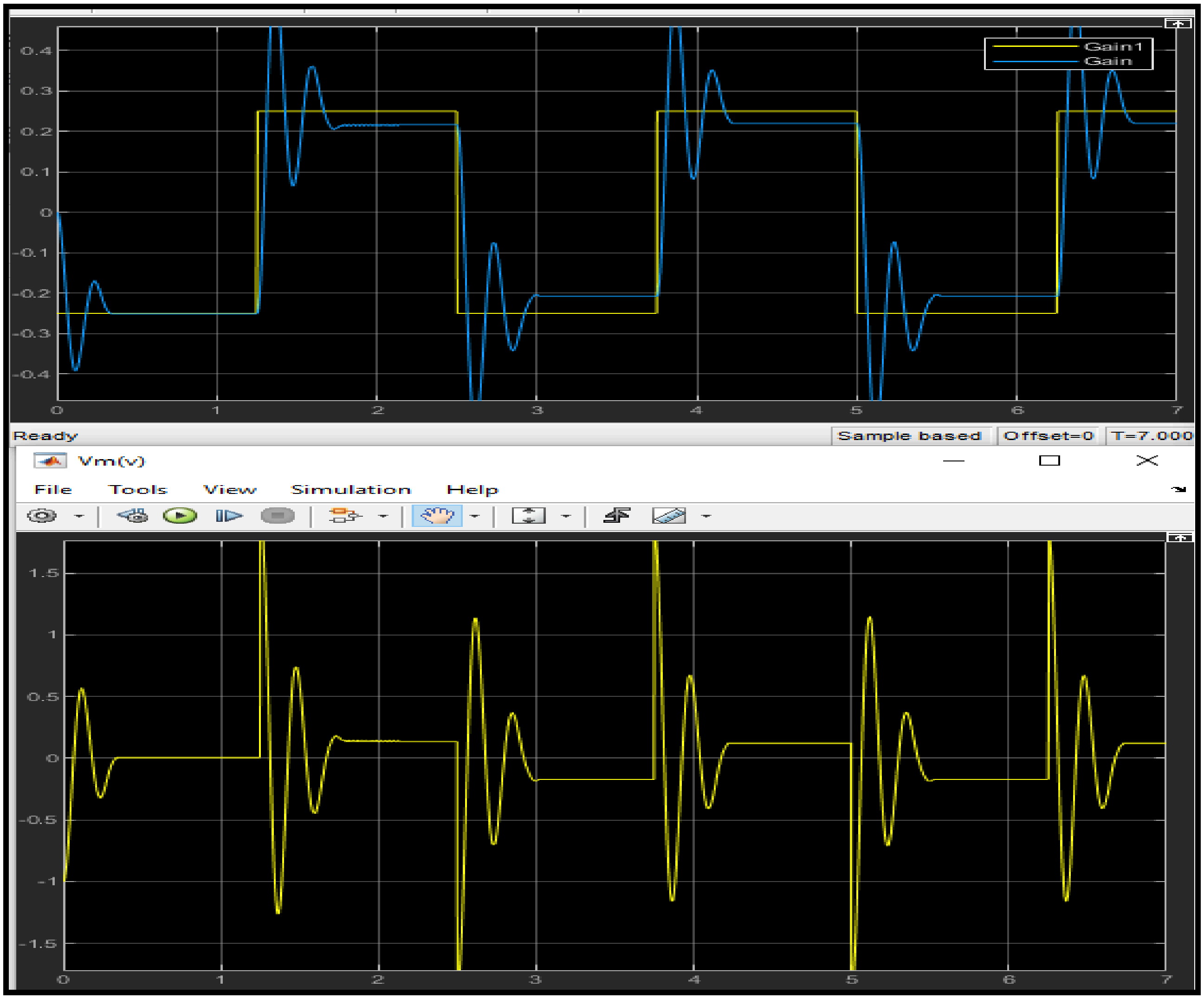

Figure 8: Part 3.5 Output Kd = 0; Kp = 4

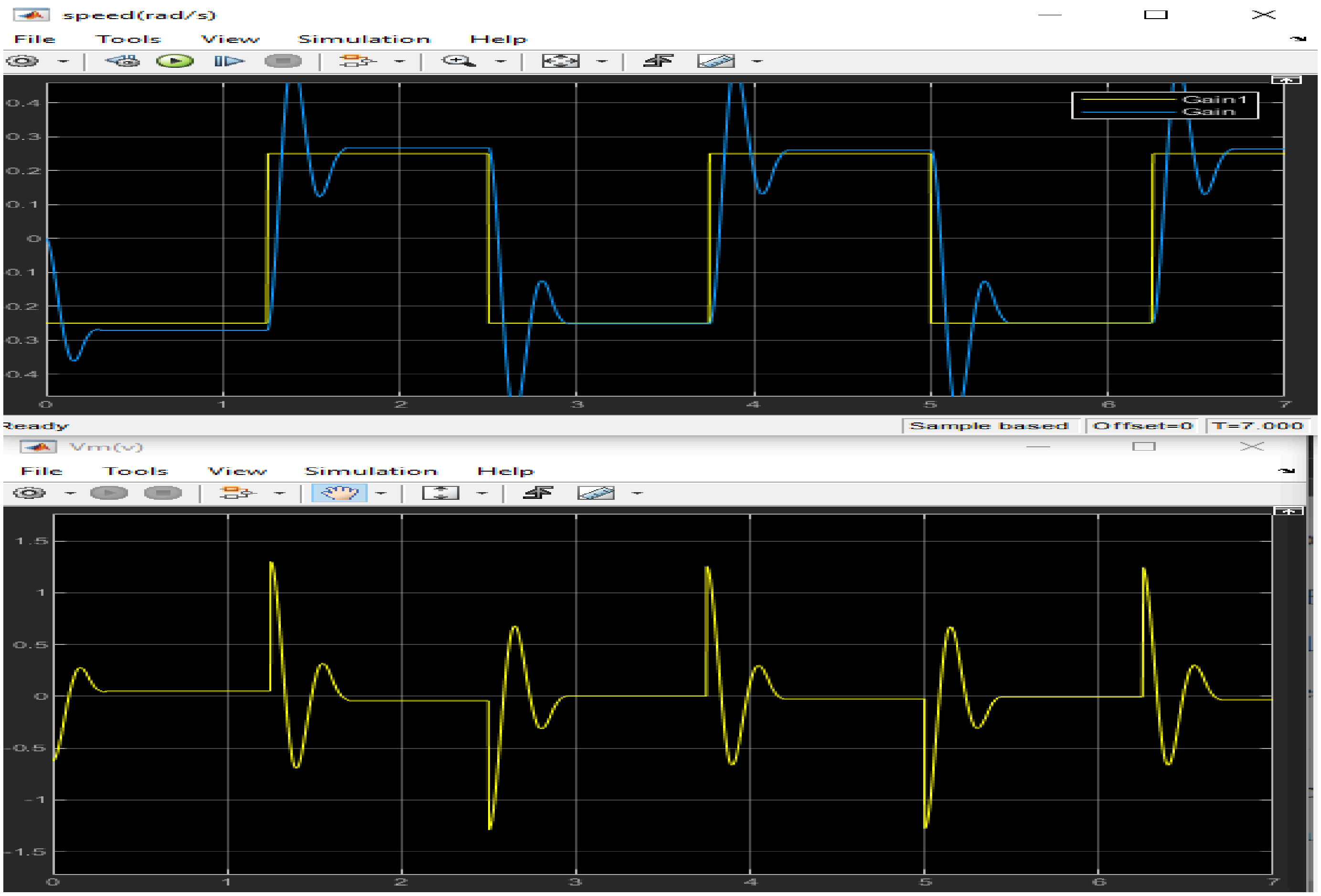

Figure 9: Part 3.6 Output Kd = 0; Kp = 2.5

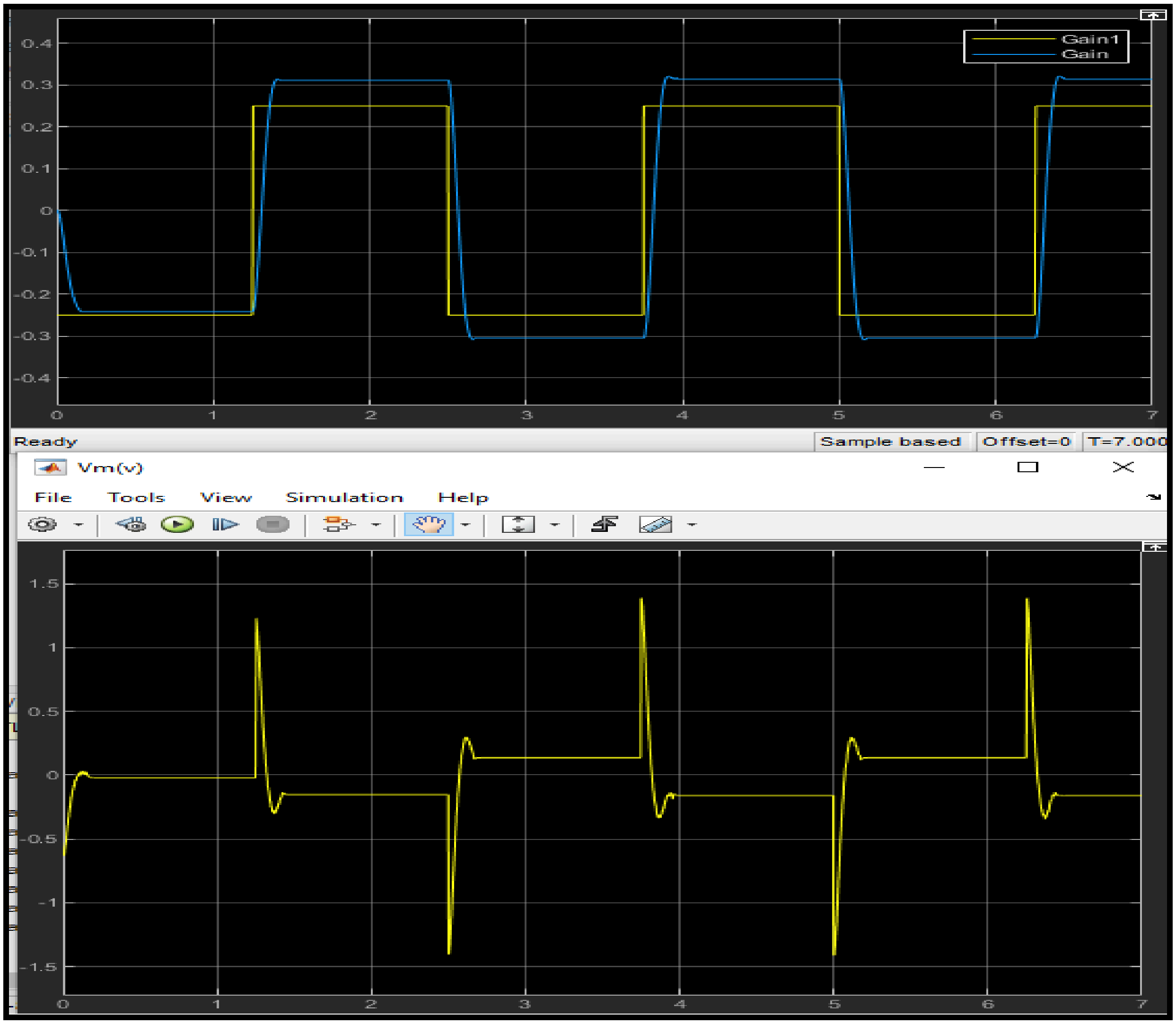

Figure 10: Part 3.5 Output Kd = 0.075; Kp = 2.5

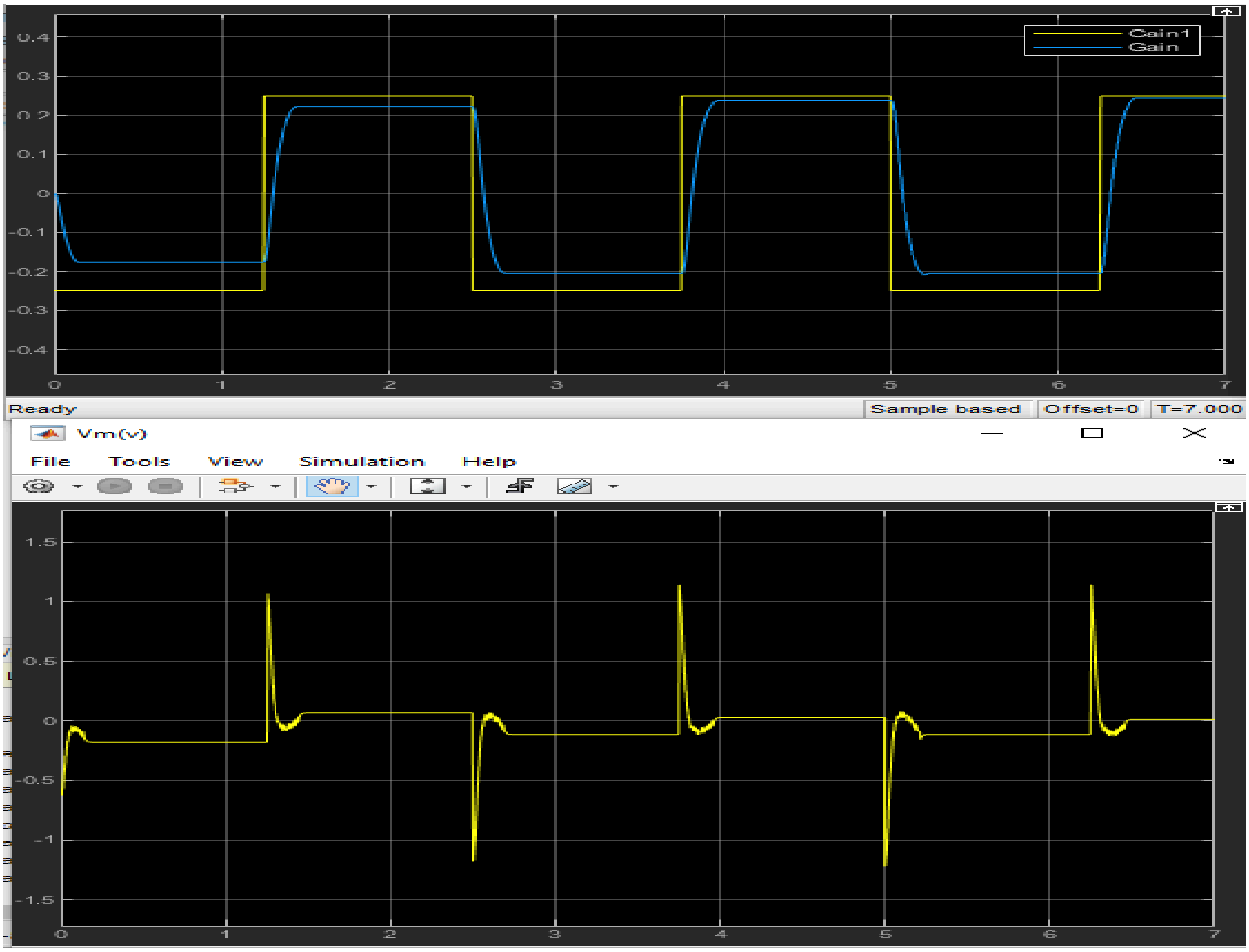

Figure 11: Part 3.5 Output Kd = 0.15; Kp = 2.5

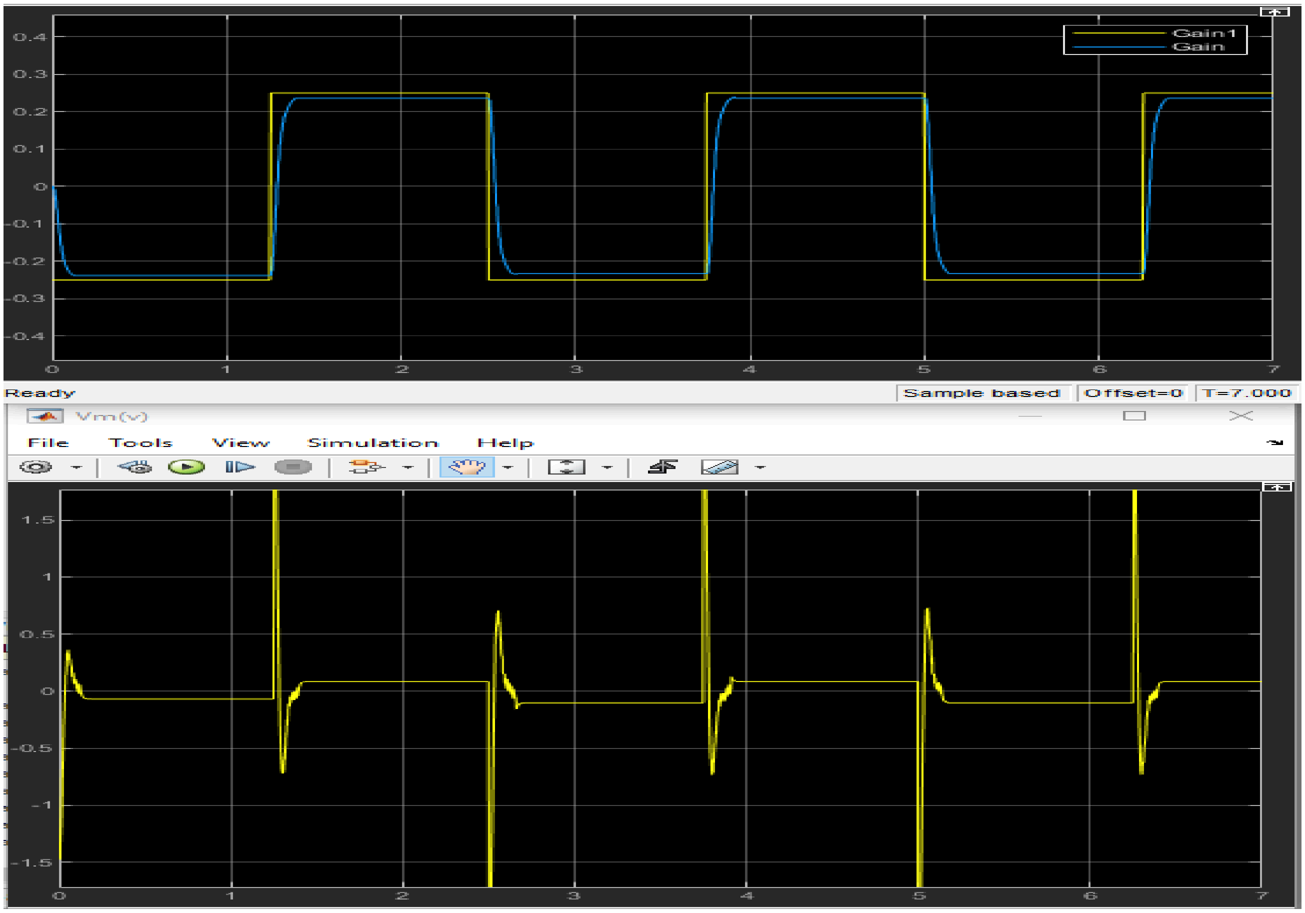

Figure 12: Part 3.10 Output

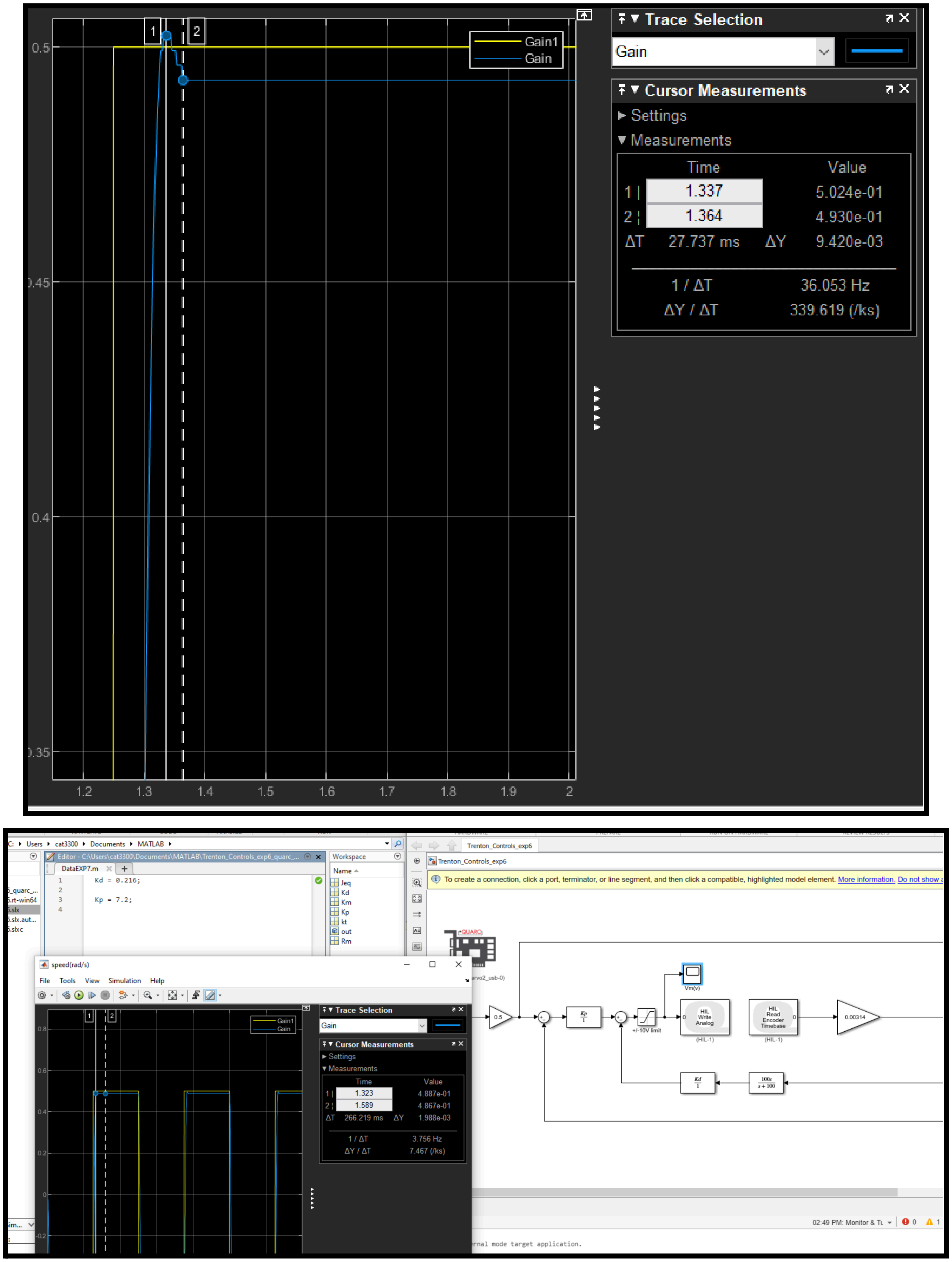

Figure 13: Part 3.11

VIII. Conclusion:

In this experiment, we delved into the intricate dynamics of the QUBE-Servo 2 system, employing both unity feedback control and proportional plus rate feedback control. Through systematic adjustments to the control gains and keen observations of the system's responses, we honed our understanding of the interplay between theoretical predictions and practical outcomes. Notably, our iterative tuning process in the latter part of the experiment, where we methodically aligned our controller's performance with the desired specifications, underscored the importance of real-world validation against theoretical models. The insights garnered from this hands-on exploration provided a tangible understanding of control system design and optimization, emphasizing the significance of both analytical and empirical approaches in engineering applications.