International Journal of Sensors and Sensor Networks
2018; 6(1): 1-7
http://www.sciencepublishinggroup.com/j/ijssn
doi: 10.11648/j.ijssn.20180601.11
ISSN: 2329-1796 (Print); ISSN: 2329-1788 (Online)

Analysis on Control System Design of Plant Simulator for Hardware-In-The-Loop Simulation Using MATLAB

Pa Pa Aye, Hla Myo Tun

Department of Electronic Engineering, Yangon Technological University, Yangon, Republic of the Union of Myanmar

Email address:
d.papaaye@gmail.com (Pa Pa Aye), hlamyotun.ytu@gmail.com (H. M. Tun)

To cite this article:
Pa Pa Aye, Hla Myo Tun. Analysis on Control System Design of Plant Simulator for Hardware-In-The-Loop Simulation Using MATLAB. *International Journal of Sensors and Sensor Networks*. Vol. 6, No. 1, 2018, pp. 1-7. doi: 10.11648/j.ijssn.20180601.11

Received: October 19, 2017; **Accepted**: October 28, 2017; **Published**: January 11, 2018

Abstract: This paper proposes to develop a simulation framework that can be used in the development of such digital control systems in academic environments. The Control System Plant Simulator simulates the object a digital control system designer wishes to control – referred to as a "Plant". The Control System Plant Simulator follows the Hardware-In-The-Loop concept in that it takes the place of a physical plant. Designed and implemented controllers are attached to the Control System Plant Simulator, which will behave just as an actual plant will from the viewpoint of the controller. The simulator read input from a controller external to the host computer of the simulator. It evaluate the input in real-time, and provide output to the controller just as the actual plant will. Individuals can use the tool to prototype control systems without being forced to use expensive or limited physical resources. This allows for the education of the next generation of control system designers and implementers in a more realistic setting. The Control System Plant Simulator (CSPS) as a flexible framework for simulating plant models in control system implementation projects.

Keywords: HIL Software, Control System Plant Simulator, MATLAB, GUI, Motor Control System

1. Introduction

Control System simulators have been developed in the past. Current plant simulation frameworks are very expensive. The Control System Plant Simulator is an open source project, provided at no cost. The Control System Plant Simulator runs a hardware-in-the-loop simulation that takes the place of a physical plant. It is designed to be connected to an implemented controller, and behave as a real plant would. This eliminates the expense or danger of testing actual equipment, while fully exercising the implemented controller. Hardware-in-the-loop test simulations have been made for specific situations, products, or markets. The Control System Plant Simulator provides a general framework upon which plants of different natures may be simulated by simply providing a model of the plant. The Control System Plant Simulator is extendable in that user interfaces, plant descriptions, and physical interfaces may be updated or customized with little difficulty. Hardware-In-The-Loop simulation is a technique that is used in the development and testing of complex embedded systems. In a HIL simulation, represented system consists of the simulated part and a real part, the "hardware-in-the-loop". Most hardware-in-the-loop simulators require specialized equipment or test boards to run.

The Control System Plant Simulator is designed with academic environments in mind. It is of the utmost importance for students to be able to implement their designs as physical controllers, but it is often too expensive to test these controllers on physical targets. The targeted plant may be expensive, fragile, limited, or dangerous. The Control System Plant Simulator (CSPS) can be used in place of this equipment, and provide results for the students to monitor and determine the success of their controller design. Because of the low cost goal, the system may be used to prototype controllers developed by each student individually. This allows students the ability to work on controllers, correcting mistakes as they go, without the constant use of limited or unavailable lab equipment. The CSPS is designed to simulate many different kinds of plants, and as such has a flexible front end. The graphical user interface may be swapped out for any other designed by end users and implementers. These interfaces may be used to display animated versions of the plant (such as watching the level of water in a tank raise and lower) and are designed such that there is no impact to the system simulation.

The CSPS is designed to be portable. All operating system calls are abstracted in a separate OS Layer.

The CSPS is not a physical controller, but it behaves as one. This paper provides a framework upon which a plant may be created and run on a Windows XP environment. This plant simulation may be used in place of a real physical plant, making physical equipment less important. Students may perfect their designs at their own workstations without the need of additional equipment.

This work provides a low-cost flexible plant simulator that can be used for the development of digital controllers. Furthermore, this plant simulator provides benefits in academic teaching and research environments. Digital Controllers are control systems that are implemented using digital computers to control a subject system, called a plant. Figure 1 shows a stepper motor control system. A stepper motor positions the read-write heads in a floppy drive. They were once used for the same purpose in hard drives. All of the commutation must be handled externally by the motor controller, and typically, the motors and controllers are designed so that the motor may be held in any fixed position as well as being rotated one way or the other.

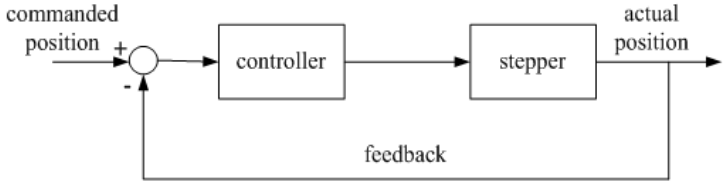

Figure 1. *Block Diagram of Stepper Motor Control System.*

2. HIL in the Design Cycle

Traditional testing, referred to as *static* testing, is where functionality of a particular component is tested by providing known inputs and measuring the outputs. Today there is more pressure to get products to market faster and reduce design cycle times. This has led to a need for *dynamic* testing, where components are tested while in use with the entire system, either real or simulated. Because of cost and safety concerns, simulating the rest of the system with real-time hardware is preferred to testing individual components in the actual real system. Dynamic testing also encompasses a larger range of test conditions compared to static testing. Employing this strategy for dynamic testing is known as Hardware-in-the-Loop (HIL) simulation. HIL is an integrated part of the design cycle. Figure 2 below represents the design cycle of embedded control applications common to automotive, aerospace, and defense industries.

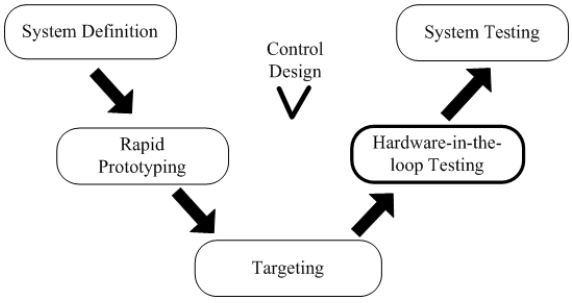

Figure 2. *Control Design Cycle.*

The control design cycle consists of several different stages of design and testing. The System Definition step documents the needs and requirements of the system. The Rapid Prototyping step involves development of prototype controllers to test the basic ideas and concepts in an actual system. The actual system can be simulated using software or a combination of software and hardware. The idea of HIL testing is similar in concept to rapid prototyping in that part of the system is simulated while connected to the rest of the real system. After a design has been prototyped and verified, the program must be deployed on the final production controller in the Targeting phase. After the controller has been developed in the targeting phase, it needs to be tested as part of the actual system. Often the actual systems are very expensive. It then becomes imperative to test the controller on a simulation of the actual system. This stage is referred to as HIL simulation. Once the controller performs satisfactorily in the HIL simulation stage, it is then tested on actual systems.

3. Building HIL System

Such an HIL system involves the use of hardware to generate and receive realistic signals and software to simulate the dynamics of the system. There must choose the appropriate components for the application and for integration into a complete HIL system. Use sensors and actuators to receive and generate signals to and from the EDCU. Use a processing unit to perform the calculations for the simulation. Use a human-machine interface and a development machine for the simulation. Often the same machine is used for both tasks. These can use a subsystem for post simulation analysis. Figure 3 presents the architecture of a typical HIL system.

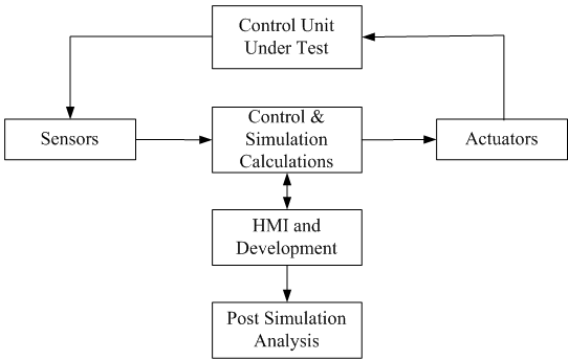

Figure 3. *Architecture of a Typical HIL System.*

4. Flow Chart for Hardware-in-the-Loop Simulation

In this section, the flow charts are provided for understanding software integrating to get HARDWARE IN THE LOOP function. The MATLAB control flow chart works that MATLAB first selects the serial port and initializes serial port settings. And then MATLAB transfers the control code to serial port according to user input condition. If the user closes the MATLAB GUI, MATLAB deactivate the serial port. The system flowchart is illustrated in Figure 4.

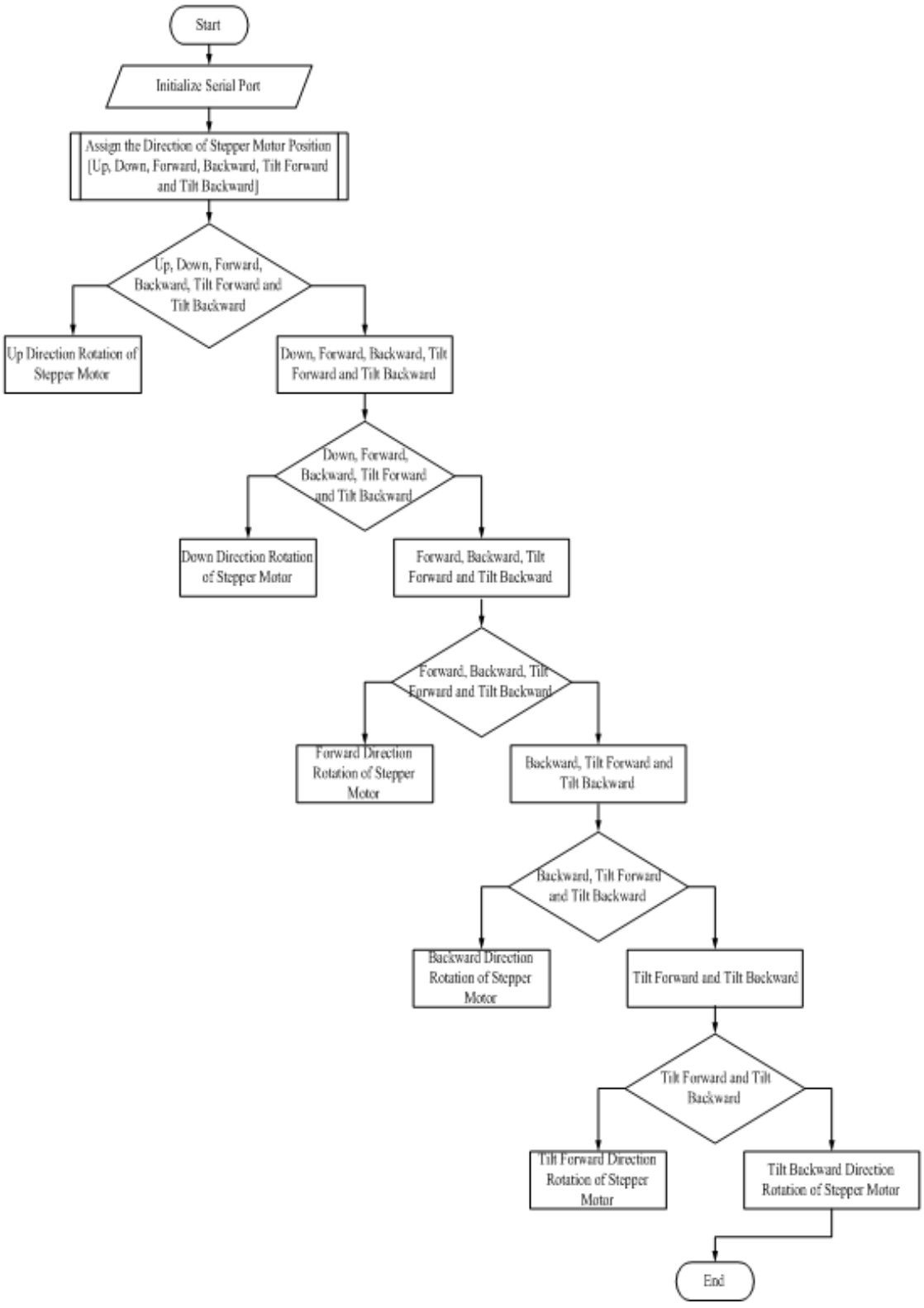

Figure 4. *MATLAB Control Flow chart for HIL simulation.*

5. Simulation Results

HIL simulator plant is getting useful in research development. But the development of an HIL simulation can be a complex and time-consuming process. Real HIL simulator consists of software controller, hardware part to convey information and software plant. In a closed hardware-in-the- loop simulation MATLAB controls the real time motor and driving electronics. The motor controller parameters can be changed on run time and the results immediately be visualized in a MATLAB command window. The HILS is a combination of a simulation (control model) and an experiment (motor) part. The key to HILS is that simulation and experiment work is done together in the same time as a real time simulation.

A. MATLAB Plant Controller

Plant Controller is created with MATLAB GUI. GUI is graphical user interface method. It consists of interface part and coding part. The user can develop interface and coding using MATLAB GUI tool. Basic GUI tools are static text, edit text, push button, slider, radio button, check box, and others. The user drags and places these tools to workspace to create GUI interface. The user can edit the GUI tools using properties window. The coding of the any GUI tool can be written in coding window. The program coding is flexible and reliable for any motor controller. To build Control System Plant Simulator, the Data Translations DT-Open Layers libraries must be included. To successfully run this application, a Data Translation 9812 data acquisition device must be attached to the host PC via USB. In this paper, MATLAB coding is only used for stepper motor controller because required tools for the real full package developing of HIL simulator plant is still hard and expensive.

B. Discussion on Simulation Results

Figure 5 shows the main GUI window for the control system design. This design is composed of two stepper motors which one for horizontal position and one for vertical position. The details development is made by using the appropriate MATLAB function for simulation purposes.

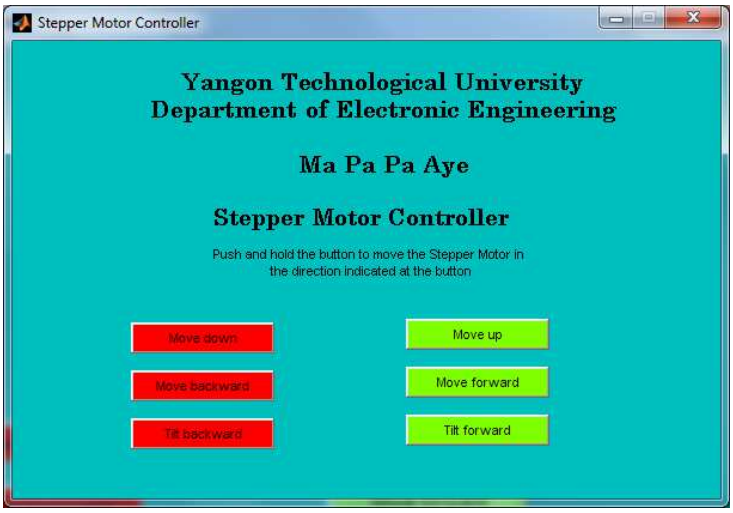

Figure 5. *Main GUI interface.*

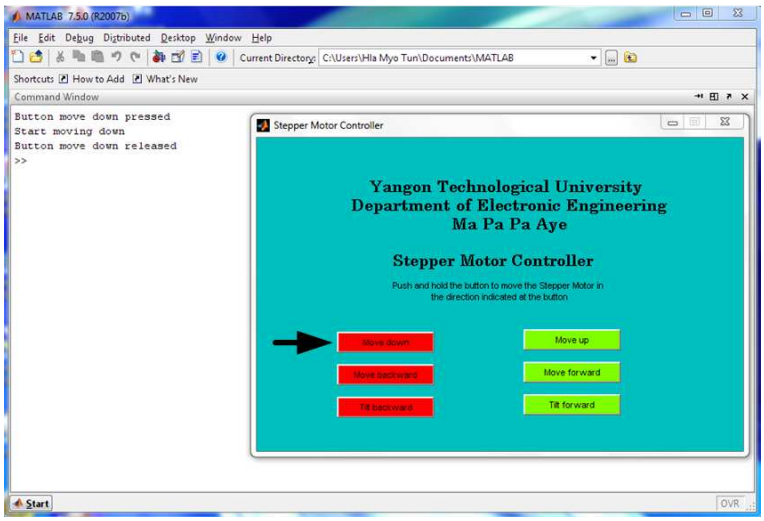

Figure 6. *Result for the command window of 'Move down' control key.*

The 'Move down' and 'Move up' control keys are used for vertical position. Figure 6 and 7 illustrates the Up and Down

direction of the stepper motor rotation according to the command from external panel.

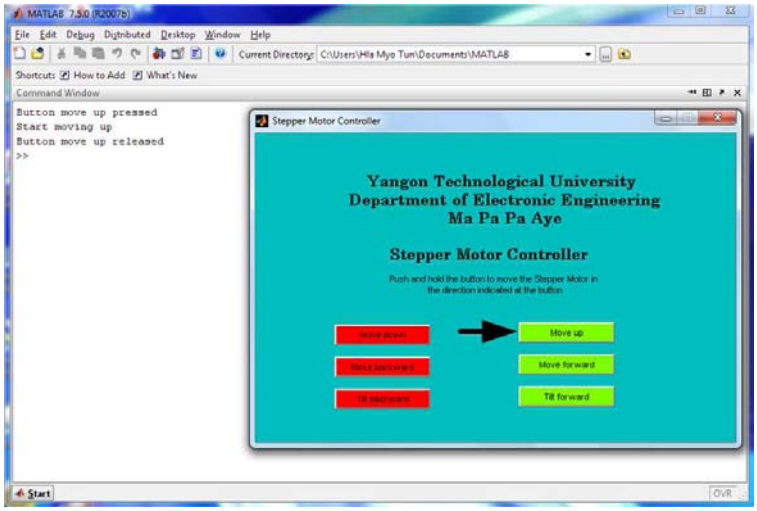

Figure 7. *Result for the command window of 'Move up' control key.*

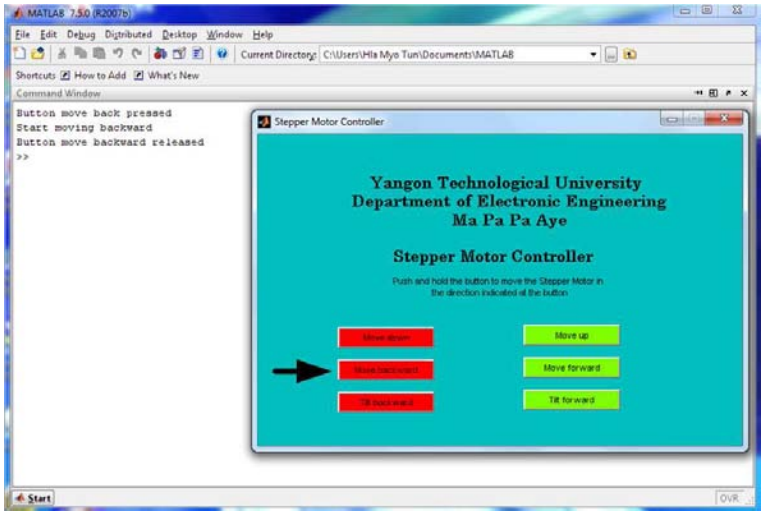

Figure 8. *Result for the command window of 'Move backward' control key.*

The 'Move backward' and 'Move forward' control keys are used for horizontal position. Figure 8 and 9 illustrates the forward and backward direction of the stepper motor rotation according to the command from external panel.

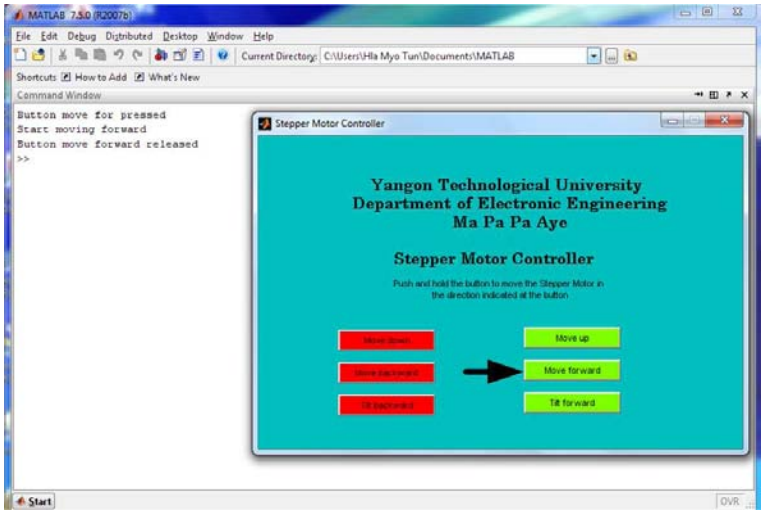

Figure 9. *Result for the command window of 'Move forward' control key.*

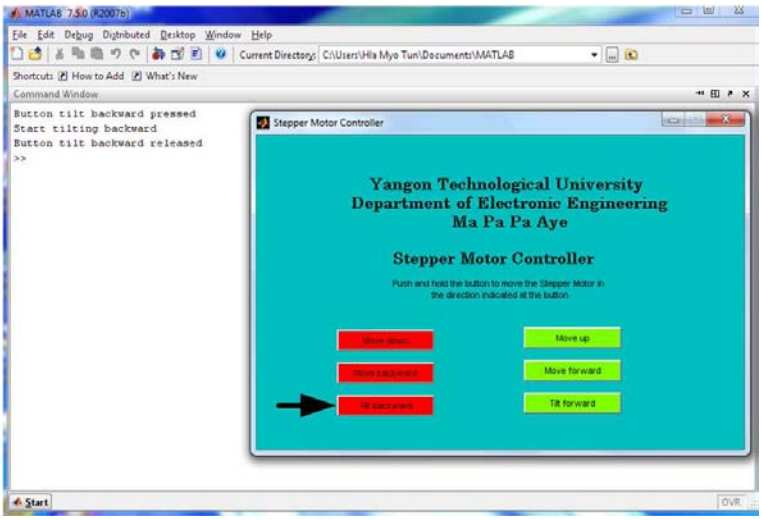

Figure 10. *Result for the command window of 'Tilt backward' control key.*

The 'Tilt backward' and 'Tilt forward' control keys are used to convert from horizontal to vertical position. Figure 10 and 11 illustrates the tilt forward and tilt backward direction of the stepper motor rotation according to the command from external panel.

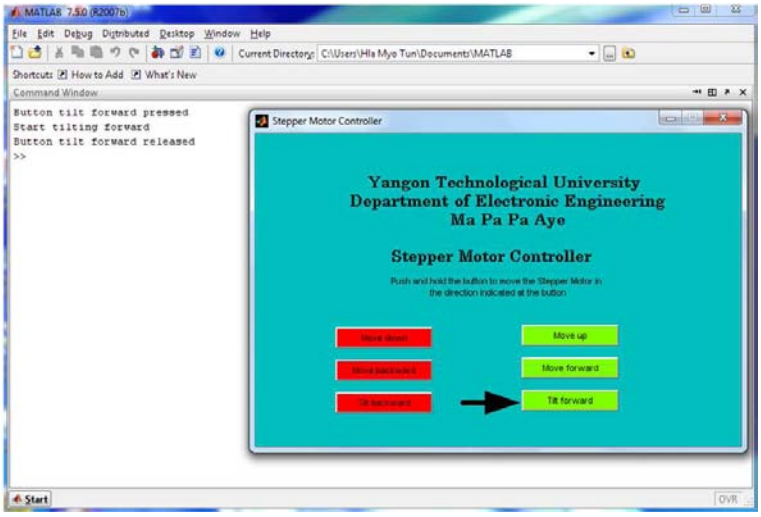

Figure 11. *Result for the command window of 'Tilt forward' control key.*

6. Conclusion

The goal of this work was to provide a framework upon which known modeled plants could be launched and simulated as part of a Hardware-In-The-Loop system with an eye toward aiding experimental works in controls development. Control system experimental works benefit greatly by having students develop real controllers that are connected to real systems to monitor the success or failure of their designs. Such physical systems can be expensive or dangerous, making their use in an educational laboratory environment difficult at best. The Control System Plant Simulator (CSPS) provides a suite of applications that simulate plants with physical output accurate enough that students can design and develop controllers for them. The framework provides plenty of hooks upon which end users may attach their own interfaces and data acquisition systems. It is flexible and can be as complex or simple as needed. In short, the goals for the research were met successfully. An expandable framework for the simulation of plants has been created. The CSPS is provided at no cost to all as an open source system with the hope that end users customize it, make it their own, and share their alterations with all.

References

[1] David Chandler, "Control System Plant Simulator: A Framework for Hardware-In-The-Loop Simulation", (2007).

[2] Gomez, Martin. "Hardware-in-the-Loop Simulation." Embedded.com. http://www.embedded.com/story/OEG20011129S0054 accessed in (2006).

[3] Messner, William C., Smoker, Jason, Tilbury, Dawn, Tseng, Ian, "Modeling Tutorials for MATLAB and Simulink." Accessed in (2006). http://www.me.cmu.edu/ctms/.

[4] Metzger, M., Moor, S. S., Piergiovanni, P. R. “Process Control Kits: A Hardware and Software Resource” in Proceedings of the 35th Annual Conference on Frontiers in Education, (2005).

[5] Zhen Li, Kyte, M., Johnson, B., “Hardware-in-the-loop real-time simulation interface software design” Proceedings on the 7th international IEEE conference on Intelligent Transportation Systems”, (2004).

[6] Murrer, Robert “Technologies for Synthetic Environments: Hardware- In-The-Loop Testing”, SPIE-International Society for Optical Engines (2003).

[7] Dorf, R., Bishop, R. "Modern Control Systems Ninth Edition". Prentice Hall New Jersey, (2001).

[8] Grega, Wojciech. “Hardware-in-the-loop Simulation and Its Application in Control Education”. Department of Automatics, Univeristy of Mining and Mettaluargy Krakow Poland. From Frontiers in Education Conference, (1999).

[9] Messner, William C., Tilbury, Dawn, "Control Tutorials For Matlab and Simulink." A Web Based Approach. 1998 Prentice Hall New Jersey, (1998).

[10] Klee, H “Simulation and Design of a Digital Control System with TUTSIM” IEEE Transactions on Education, (1991).