Introductory Motion and Control
DAQ-Based Software for Phase-Lead, Closed-Loop Position Control

Introduction

- These notes describe the LabVIEW software used for ***phase-lead, closed-loop position control*** in the lab. The overall software characteristics are described here. This software is ***different*** from the proportional closed-loop control software (discussed earlier) in that it demonstrates the use of ***difference equations*** for ***dynamic compensation***.
- This software was written specifically to interface with the electro-hydraulic trainers in the lab. It interfaces with National Instruments ***data acquisition*** cards (NI 6052E or NI 6251) to ***send*** voltage commands to a ***linear proportional valve*** to ***control*** step ***position changes*** of a hydraulic cylinder. It also ***reads*** and ***stores*** the valve command signal and the valve spool and hydraulic cylinder positions.

Front Panel

- The front panel of the closed-loop LabVIEW program for ***step position changes*** is shown in Fig. 1. The ***Desired Position*** box on the upper left specifies the desired position (***in inches***) of the hydraulic cylinder.
- The program assumes the valve commands are also available on analog ***input Channel 2***, and the hydraulic cylinder and valve spool signals are available on analog ***input Channels 0*** and ***1***, respectively.

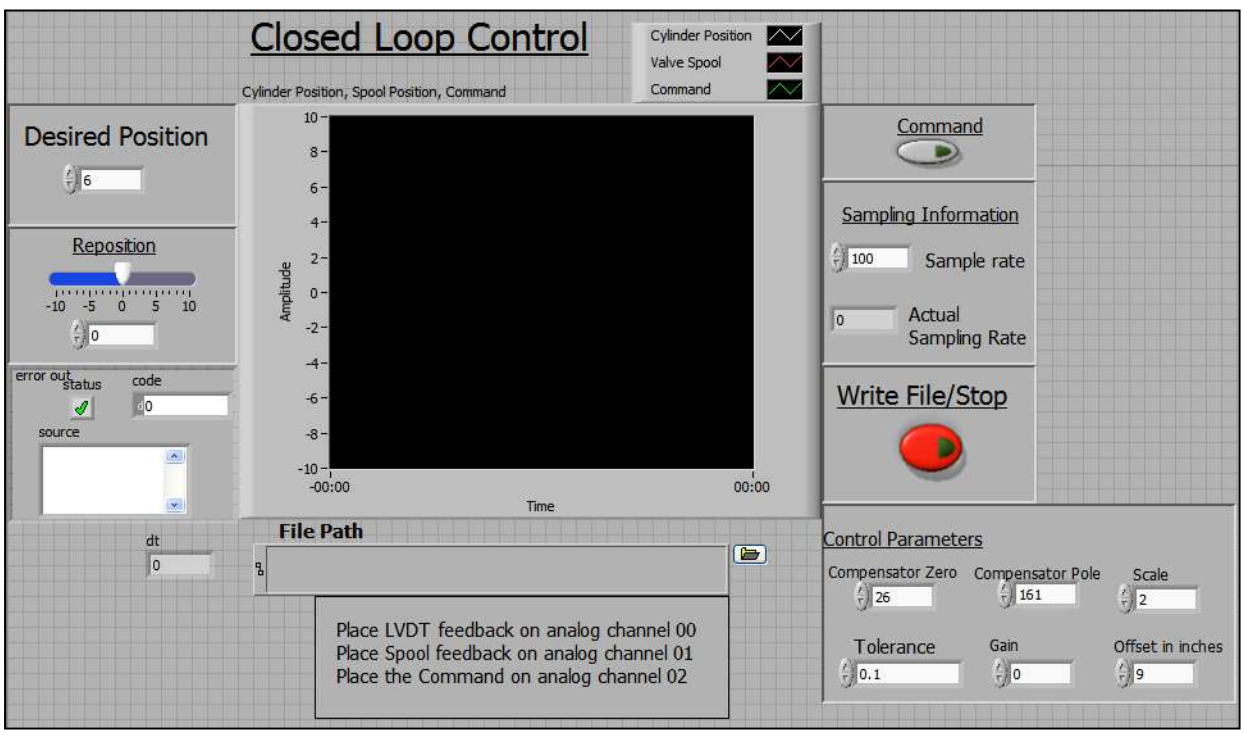

Fig. 1. The Front Panel (Graphical User Interface)

- The ***rate*** at which the analog input and output channels are ***sampled*** (in ***samples/sec***) is shown in the ***Sampling Information*** box. Also contained in this box is the ***actual*** sampling rate achieved as determined by the LabVIEW code.

- Once data is collected, it is ***stored*** in a ***file*** as specified in the ***File Path*** input box. The file folder ***icon*** on the right of the box allows the user to ***browse*** for a file location. The size of the file generated is determined by the ***sampling rate*** and the ***time duration*** of the measurement.
- The ***Control Parameters*** box is in the lower right portion of the front panel. It allows input of the following parameters.
  - Compensator parameters: ***Gain***, ***Compensator Zero***, ***Compensator Pole***
  - ***Scale***: scale factor for the position sensor voltage
    (position sensor on trainers are 1 volt/inch, so the scale factor is 1)
    (position sensor on the sled system is ½ volt/inch, so the scale factor is 2)
  - ***Tolerance*** is the ***position error*** in ***inches*** the controller is trying to achieve. The ***smallest*** achievable ***tolerance*** is determined by the ***deadband*** of the valve and the ***resolution*** of the cylinder's position measurement.
  - ***Offset*** is the value added to the position to allow the user to shift the zero-position location. Given a scale factor (S), offset (O), and position sensor voltage (V), the position x of the cylinder (in inches) is calculated as follows: $x = (S \cdot V) + O$
- Once values for ***desired position***, ***sampling rate***, ***file name***, and the ***control parameters*** are entered, the user ***starts*** the program ***execution***. The program then acts in ***open-loop mode***, sending the ***reposition*** voltage (initialized to zero) to the control valve.
- Clicking the ***Command*** button (turning it ***ON***) shifts the program to ***closed-loop mode***, moving the cylinder to the desired position. Data collection also occurs in this mode.
- Clicking the ***Command*** button, a second time (to turn it ***OFF***), returns the program to ***open-loop mode*** and sends the ***reposition voltage*** to the valve. This allows the user to reposition the hydraulic cylinder to a convenient starting point.
- After repositioning the hydraulic cylinder, clicking the ***Write File/Stop*** button ***writes*** the collected data to a text file and ***stops*** program execution.

The Block Diagram

- The LabVIEW ***code*** is depicted in the form of a ***block diagram***. Descriptions of the basic operation of the various segments of code follow.

<u>Analog Input Channel Initialization</u>

- The most recent form of data acquisition available in LabVIEW is called ***DAQmx***. LabVIEW includes many subroutines (VI's) to support this feature. ***Fig. 2*** shows the first segment of the code that utilizes *DAQmx* VI's to create a ***closed-loop control/data acquisition task*** and ***initializes*** the analog input ***channels***.
- The ***initialization*** of each channel includes defining the ***physical device*** (NI 6052E or NI 6251), the ***channels*** (in this case, analog input channels *0, 1,* and *2*), the ***minimum*** and ***maximum*** voltages for each channel, the ***type of measurement***, the ***sampling rate***, and the ***sampling mode***.

- The ***sampling mode*** determines whether sampling is based on ***software*** or ***hardware timing***. (Hardware timing is the most accurate.) This code utilizes ***hardware-timed, single point mode***. This mode ensures that for each value ***read*** from the analog input channels, a control command value is ***sent*** to the analog output channel.
- The ***DAQmx Timing*** block measures the ***actual*** sampling ***rate*** achieved. This rate is ***displayed*** to the output box in the Sampling Information section on the right side of the front panel.

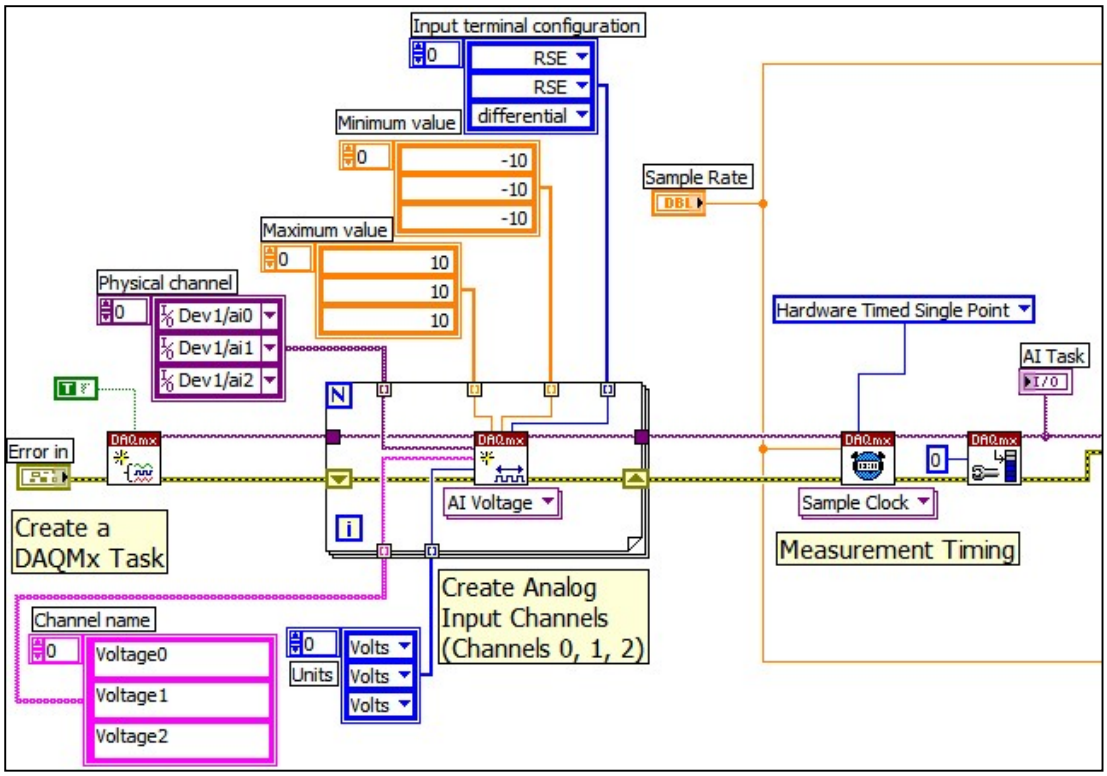

Fig. 2. Segment to Create DAQmx Task and Initialize Analog Input Channels

<u>Analog Output Channel and Control Variable Initialization</u>

- The ***initialization*** of the analog voltage ***output channel*** includes defining the ***physical device*** (NI 6052E or NI 6251), the ***channels*** (in this case, just channel *0*), the ***sampling rate***, and the ***sampling mode***. The block diagram for this section of the program is shown in ***Fig 3***.
- The ***sampling rate*** is ***carried forward*** from the analog input initialization, and as before, ***hardware-timed, single point*** sampling is used to guarantee hardware timing. As wired, the sample clocks for the analog input and output tasks are ***shared*** so the two tasks will be ***synchronized***. The ***analog output task*** is then ***started***; however, since it is ***tied*** to the sample clock for analog input, analog output will not occur until the analog ***input task*** is ***started***.
- This section also initializes the ***reposition voltage*** and the ***coefficients*** in the ***compensator difference equation*** before the repositioning/closed-loop control loop is entered.
- Finally, this section ***opens*** (or creates) a ***data file***.

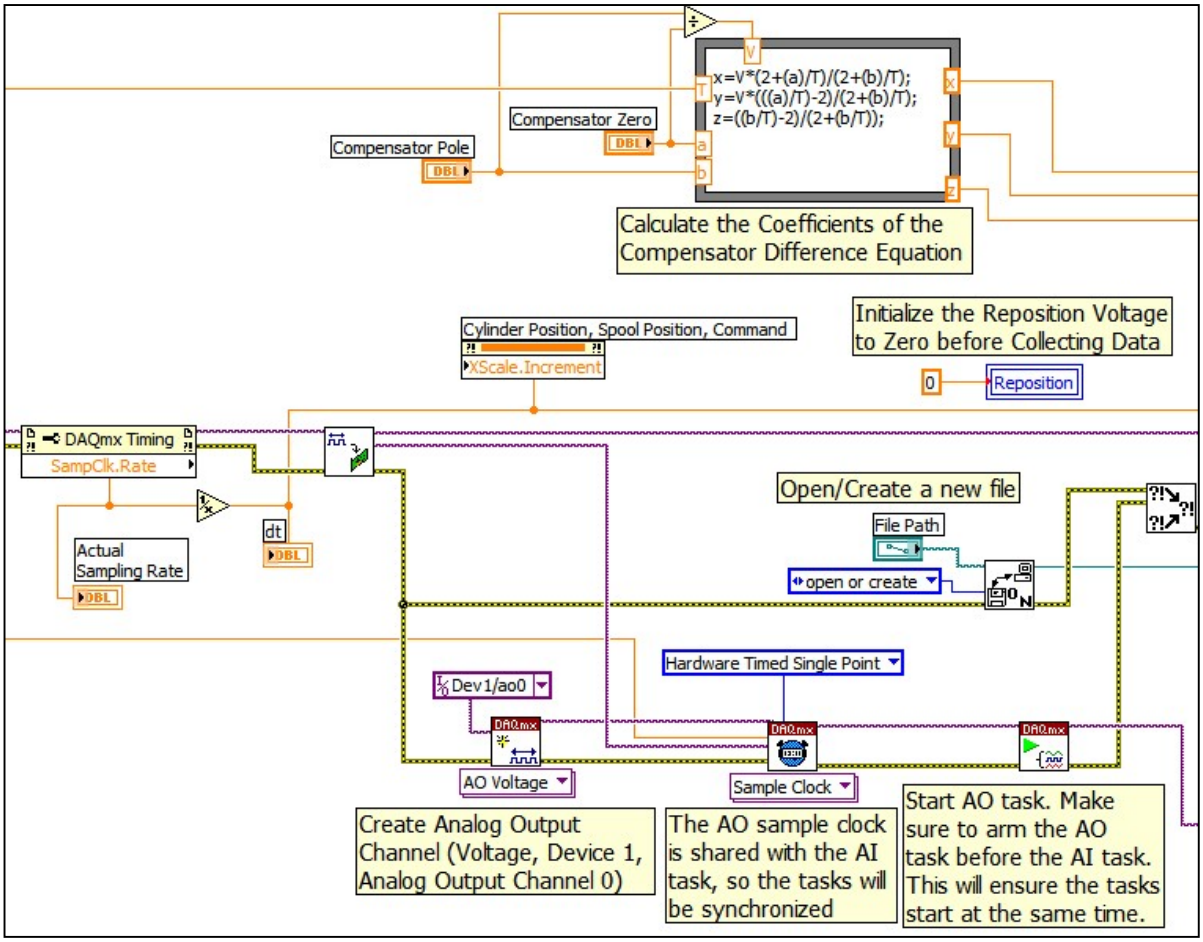

Fig. 3. Segment to Initialize the Analog Output Channel, Synchronize it with the Analog Input Channels, and Initialize important Control Variables

Closed Loop Control and Repositioning

- ***Before*** the ***closed loop control*** and ***repositioning*** "while loop" is executed, the ***analog input task*** (and consequently the analog output task) is ***started***, a ***null array*** is created for the data, and the ***desired position*** from the front panel is identified.
- As execution enters this loop, the ***Command*** button should be ***OFF***, so the program operates in ***open-loop mode*** by sending the ***reposition voltage*** to the analog output channel and data is collected and shown on the ***Waveform Chart*** on the front panel. The block diagram of this segment with the ***Command*** button ***OFF*** is shown in ***Fig 4***.

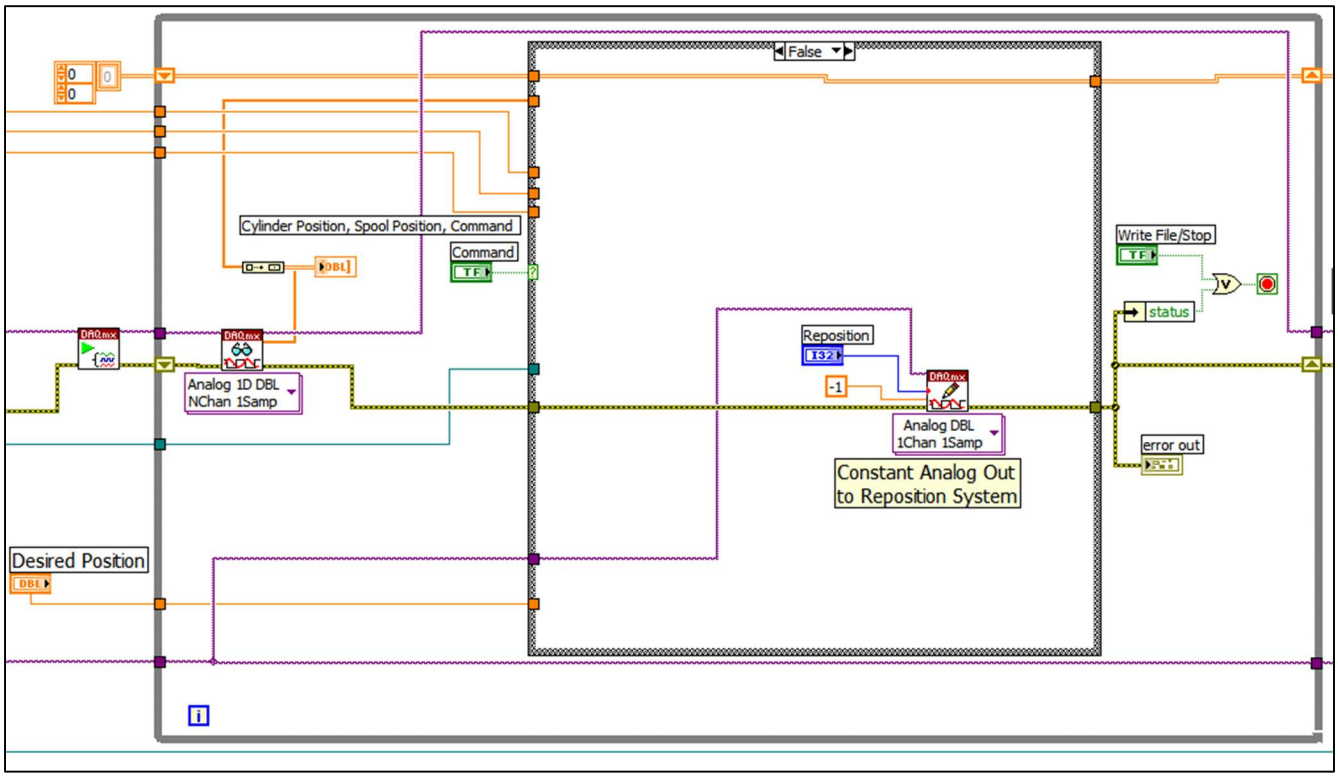

Fig. 4. The Control Loop with the Command Button ***OFF***

- Clicking the ***Command*** button (turning it ***ON***) initiates closed-loop control and collects data. The data is ***appended*** to the null array. Data is collected by the data acquisition card, one point at a time using hardware timing. These values are immediately transferred to the computer's memory.
- As a result of this ***point-by-point*** process, this program must run at much ***slower sampling rates*** than the open-loop data acquisition program. The ***amount*** of data collected (and the size of the file) is dependent on the amount of time the user leaves the ***Command*** button ***ON***. The block diagram of this segment with the ***Command*** button ***ON*** is shown in ***Fig 5***.
- Clicking the ***Command*** button, a second time (to turn it ***OFF***), ***stops*** the closed-loop control process and ***sends*** the reposition voltage to reposition the hydraulic cylinder.
- ***After repositioning*** the cylinder, the reposition voltage should be ***reset to zero***. Then click the ***Write File/Stop*** button on the right of the front panel to stop the "while loop".

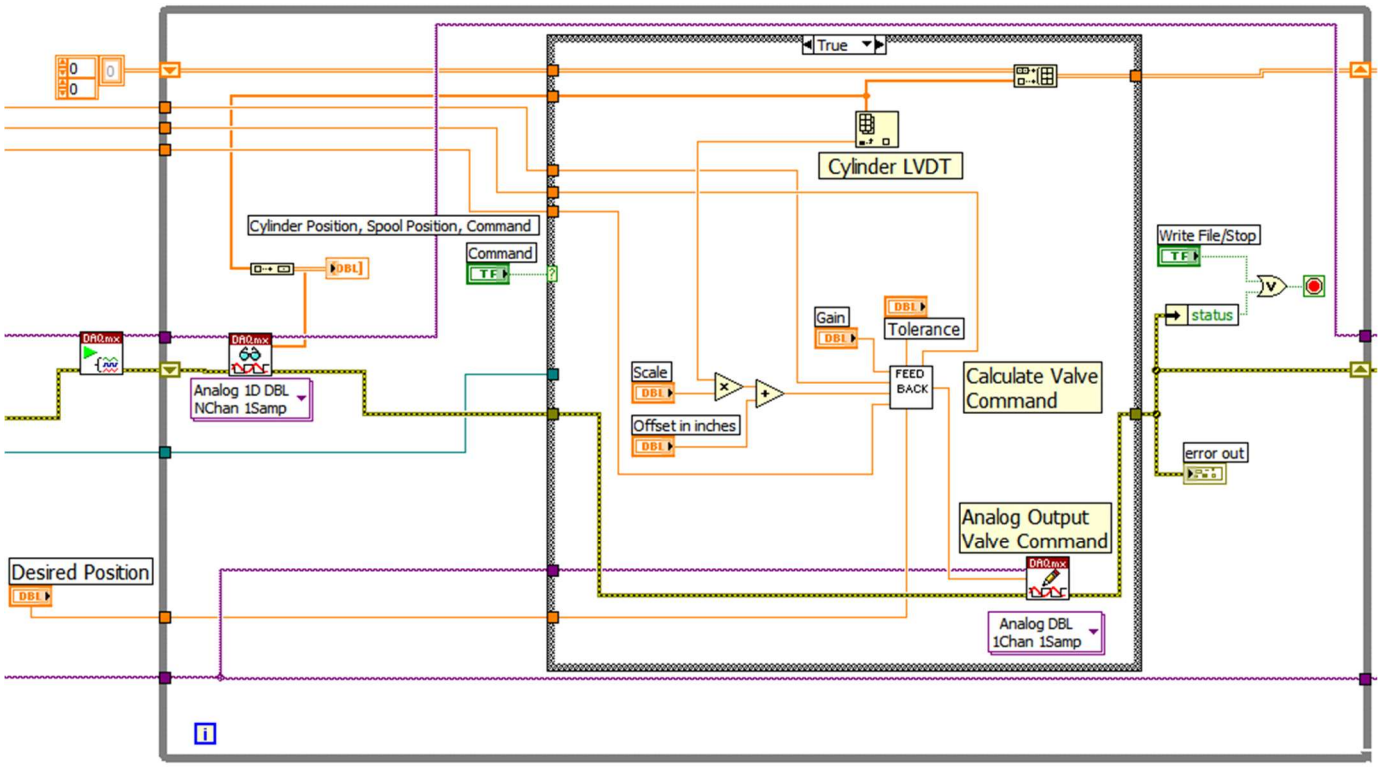

Fig. 5. The Control Loop with the Command Button ***ON***

<u>Feedback Subroutine (VI)</u>

- During closed loop control, the valve command is determined by the ***Feedback*** subroutine (VI). The valve command is found from the compensator's ***difference equation***.
- The valve command is a function of the ***sample time*** (T), the compensator ***zero*** and ***pole*** locations (a, b), the ***current*** and ***previous*** position ***errors*** (er, erold), and the ***previous valve command*** (yold). The ***current error*** is calculated as the difference between the ***desired*** and ***actual*** positions (in inches).
- Once the position of the cylinder is within the ***tolerance*** of the desired position, the valve command is set to ***zero***, keeping the system from trying to make ***small corrections*** to the position. ***Figs 6*** and ***7*** show the feedback block diagram when the system ***is*** and ***is not*** within the ***tolerance*** of the desired position.

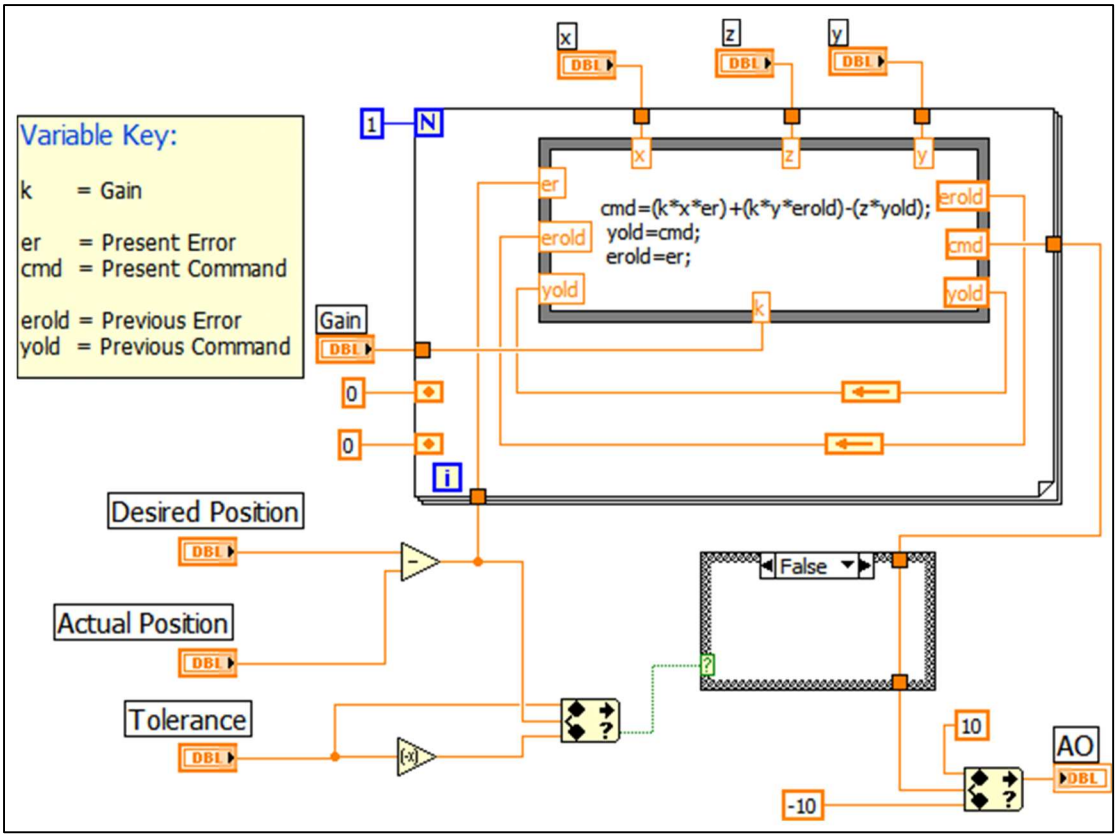

Fig. 6. Feedback VI when Cylinder Position is ***not close*** to Desired Position

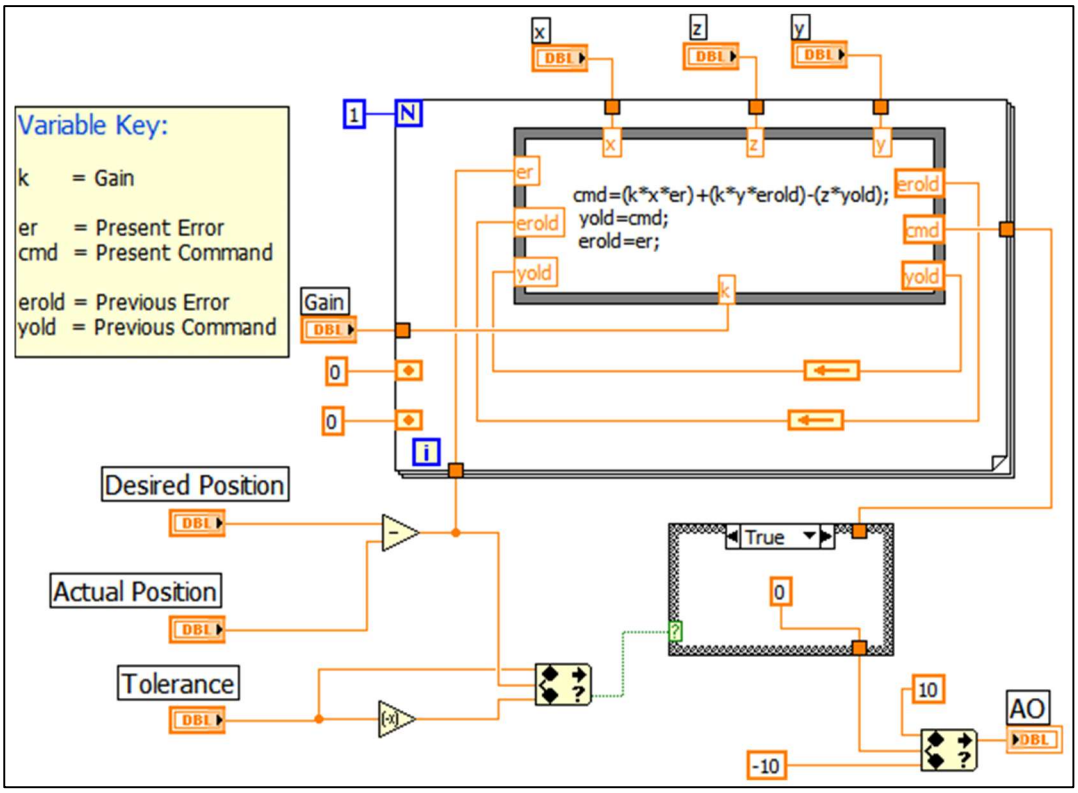

Fig. 7. Feedback VI when Cylinder Position ***is close*** to Desired Position

Storing the Data

- Clicking the ***Write File/Stop*** button (turning it ***ON***), ***ends*** the "while loop", ***stops*** and ***clears*** the I/O ***tasks***, ***writes*** the data to a text file, and ***closes*** the file. This segment of the code is shown in Fig 8.

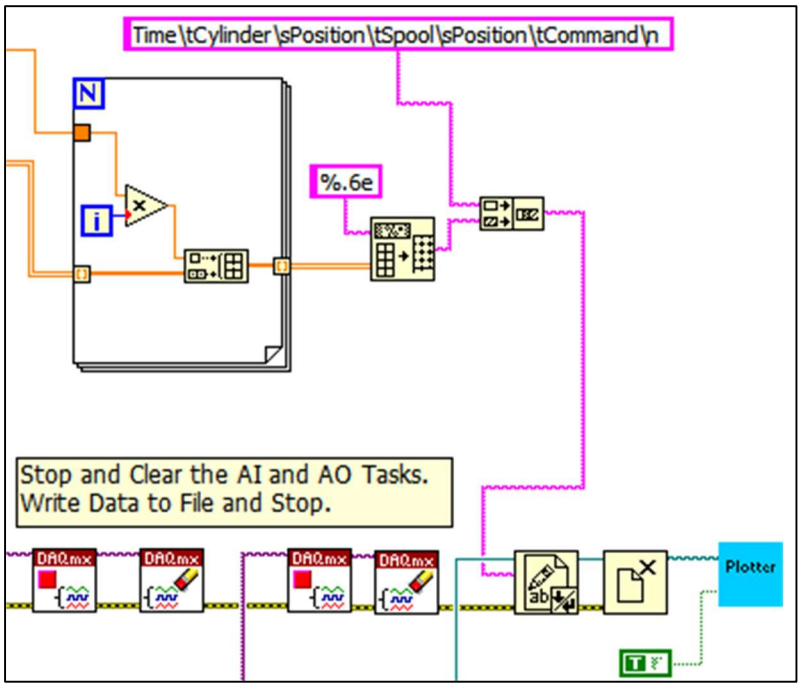

Fig 8. Segment to Record Data and Stop Execution