National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Department of Electrical Engineering and Computer Science

Faculty Member: Ms. Neelma Naz Dated: 21/02/2023

Semester: 6th Section: BEE 12C

EE-371: Linear Control Systems

Lab 4: Data Acquisition in LabVIEW

Lab Instructor: Yasir Rizwan

Group Members

Name	Reg. No	Lab Report Marks	Viva Marks	Total
		10 Marks	5 Marks	15 Marks
Danial Ahmad	331388			
Muhammad Umer	345834			
Tariq Umar	334943			

1 Table of Contents

2 Table of Figures

3 Data Acquisition in LabVIEW

3.1 Objectives

The objectives of this lab are:

- Introduction to data acquisition
- Learn data acquisition in LabVIEW using NI ELVIS

3.2 Introduction

Data acquisition refers to the process of capturing real-world physical signals and transforming them into digital numerical values that can be processed by a computer. For instance, the humidity of a room can be considered as a physical signal that changes over time. Most humidity sensors produce an analog voltage or current, which cannot be directly processed by computers or microcontrollers.

To enable digital machines to analyze physical signals, a data acquisition system is utilized to sample the analog signal and convert each sample into a digital numerical value. This process is crucial for analyzing physical signals such as temperature, pressure, fluid flow rate, motor speed, pendulum angle, wind speed, etc. Without data acquisition, it would not be possible to harness the analytical power of computers when dealing with physical systems or signals.

3.3 Software

LabVIEW is a graphical programming environment that uses icons to represent instructions and commands. Its graphical approach makes it easy to understand and manipulate data. It allows the user to tailor data inputs and outputs to meet their specific needs. The Control and Simulation module in LabVIEW allows the user to simulate and analyze complex systems, such as those found in robotics, mechatronics, and autonomous systems.

4 Lab Procedure

4.1 Data Acquisition using LabVIEW

We have given a brief introduction to the NI ELVIS and QNET DC motor. Now we will see how we can use LabVIEW for data acquisition. In this section we will only show the data acquisition for the DC motor. However, the data acquisition for any device is similar in LabVIEW especially if the DAQ is manufactured by National instruments (NI). DAQmx is the LabVIEW driver for the programming of data acquisition hardware in LabVIEW. It is a single programming interface for programming analog input, analog output, digital I/O, & counters on hundreds of multifunction DAQ hardware devices. The following are the steps for data Acquisition using DAQmx:

- Create a virtual channel and task using the NI-DAQmx Create Virtual Channel VI.
- Write or Read data using DAQmx Write or DAQmx Read VI respectively

4.1.1 Generating Voltage for DC Motor using Analog Output Channel AO#0

i. Open LabVIEW and create a new VI by going to File » New VI
ii. Right-click on the block diagram and select Measurement I/O
iii. Drop the "NI-DAQmx Create Channel VI" to the block diagram and choose "AO voltage"
iv. Right click on the physical channel input of "NI-DAQmx Create Virtual Channel VI" and create constant. Here you will select the channel you want to use. According to the schematic in QNET User Manual, the DC motor voltage is generated at analog output channel 0.
v. Create constants for maximum and minimum value and enter 8 and -8 respectively
vi. Right-click on the block diagram and select Measurement I/O
vii. Drop the "NI-DAQmx Write" to the block diagram and select Analog-DBL- 1 Channel-1Sample.
viii. The VI that we have created will only generate a single sample for the output voltage. To continuously generate the sample, we must insert a simulation loop. Insert a Control and Simulation loop outside the DAQ to execute the simulation diagram until the Control & Simulation Loop reaches the simulation final time by right-clicking on Block Diagram and using path shown in figure:
ix. Insert the Control and Simulation loop outside the DAQmx Write
x. Double-click on top left of the loop. A pop-up window will appear. Configure same simulation parameters and timing parameters as shown in the figure:
xi. When setting the voltage with the slider input, we are not considering the gain of the amplifier between AO#0 and the DC motor. To accommodate for this gain, we insert a reciprocal of this gain in our VI. In the block diagram, insert a gain block by going to Functions Palette >> Control Design and Simulation >>
xii. Simulation >> Signal Arithmetic >> Gain. Set the gain to 1/2.3 = 0.43
xiii. This completes the voltage generation part of the VI. Run the VI and change the slider to apply different voltages to the motor. The speed of the motor should change.

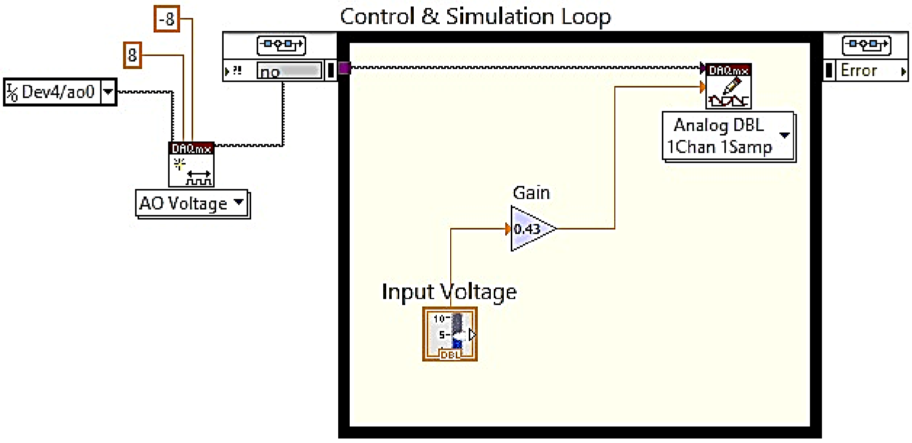

Figure 1: Block diagram for applying input voltage to motor

4.1.2 Acquiring for DC Motor Speed using Analog Input Channel AI#4

i. Follow similar steps as for the voltage generate to configure the DAQ Assistant for acquiring the motor speed. The differences are:

ii. Select analog input channel 4 instead of analog output channel 0 and use DAQmx read instead of DAQmx write

iii. Set the signal input range from -10 to 10 volts

iv. Now go to the front panel and insert a numeric indicator. Name this indicator as “speed”

v. In the block diagram, connect the data output of the “DAQmx Write” to the numeric indicator block

vi. Run the VI and apply some voltage to the motor.

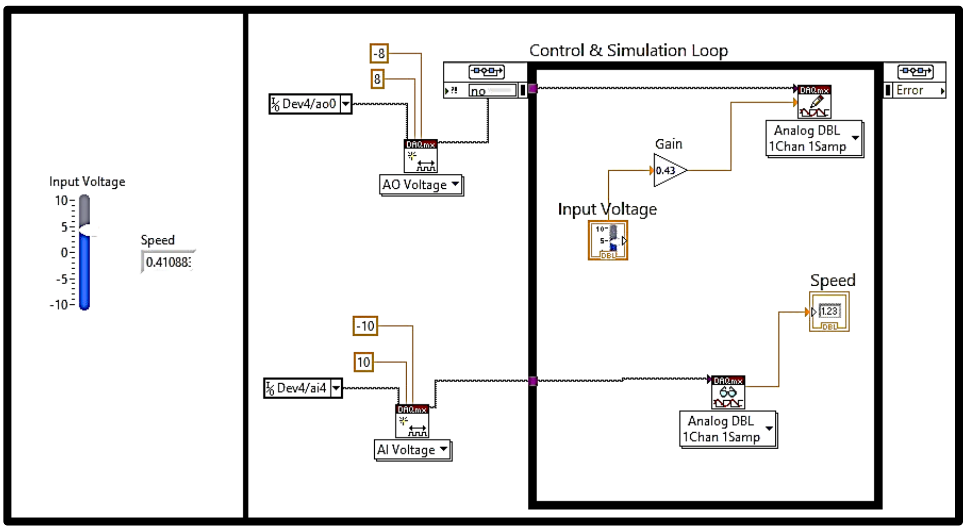

Figure 2: Interface for displaying speed upon applied voltage

vii. According to the QNET User Manual, the tachometer calibration is 2987 RPM/volts. There 2987 rotations in a minute, the tachometer generates 1 volt. Since we are acquiring the voltage from the tachometer, we are not seeing the speed of the motor in any standard units on our numeric indicator. To accommodate for the calibration of the tachometer, we will multiply the acquired voltage by a factor of 2987. This will give us the speed of the motor in RPM.

viii. Since in the earlier labs we derived the transfer function for motor speed in rads/sec, we would also like to have the speed in these units. Insert another numeric indicator in your VI and name it "Speed rads/sec". Scale the speed in RPM to find the speed in rads/sec. The formula is given below:

$$1 \text{ RPM} = 2*\text{pi}/60 \text{ rads/sec}$$

ix. Test your VI again. Now you should be able to see the speed in both units.

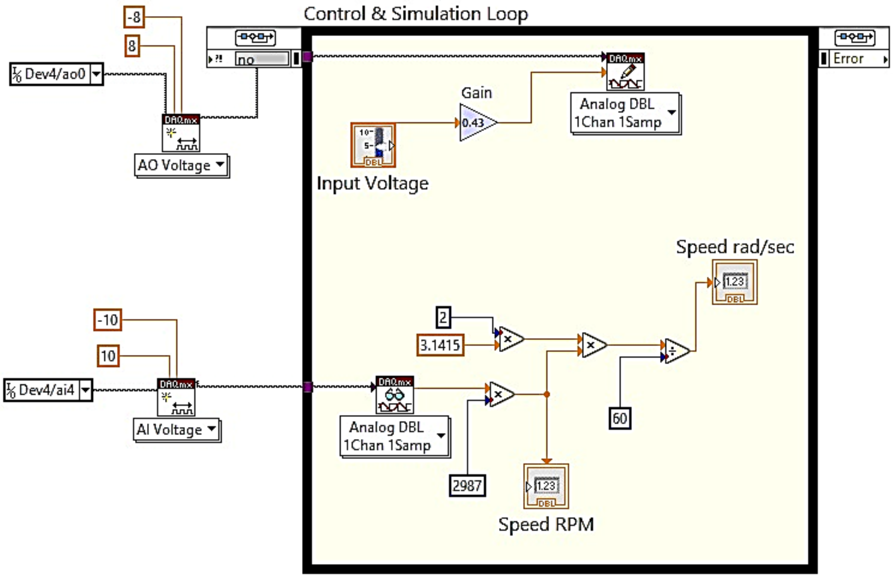

Figure 3: Block diagram for speed in both units

x. Acquiring for position of the DC motor speed using digital input channel DI#0

xi. The data acquisition of the encoder is more complicated than acquisition or generation of analog signals. Therefore, we have provided you with a VI that contains blocks for data acquisition of the encoder. Open the encoder.vi provided to you. Go to the block diagram. Copy the "encoder config" and "pos read" blocks to your VI. Connect the block as shown in the figure below.

xii. Connect the outputs of the "pos read" block to two numeric indicators. Name them "position degrees" and "position rads".

xiii. Every time we connect the NI ELVIS to the PC, it is assigned a device number. A device number is required for data acquisition. The DAQ driver that we have used automatically picks the device number of the connected device. In this section, instead of DAQ Assistant we will use another block for data acquisition. Therefore, we will need to specify the device manually. This can be done by using the device block in front panel.

xiv. In the front panel, right click to open controls palette. Go to Modern >> I/O >> DAQmx Name Controls >> Device. Place a device block in your VI and connect the "device name" block to "dev" input of encoder config VI.

xv. Run the VI and rotate the motor with your hand. The position should change.

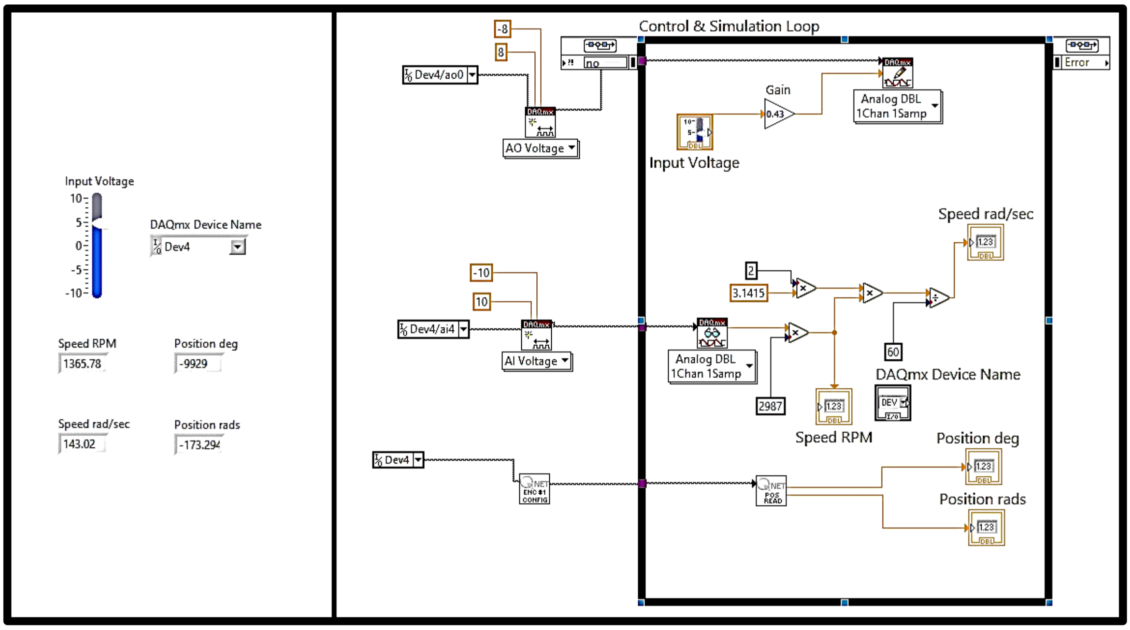

Figure 4: Interface for speed and position

4.2 Exercise 1: A Simple Test of Our Data Acquisition VI

We have created a VI for data acquisition. However, we haven't tested if the signals that are being generated are of the correct value or not. Similarly, we haven't tested if the acquired signals are of the correct value or not.

Stop the VI. Adjust the voltage input to zero volts. Run the VI. Rotate the motor by approximately 90 degrees. See whether the VI shows the same value. Repeat this for various other angles like 180, 360, -90 and -180 degrees etc.

Attach a small marker to one side of the rotating disc attached with the motor. Use a small piece of white tape or something similar. Run the VI and apply a small voltage so that the motor is rotating at around 30 to 60 RPM according to your front panel indicators. Now using a stopwatch or time and by looking at the motor itself, try to count how many rotations the motor is making. It should match with the one shown in your front panel.

The motor speed is approximately 158 rads/sec when a voltage of 5 volts is applied. Apply 5 volts to the motor and verify if this really is the case.

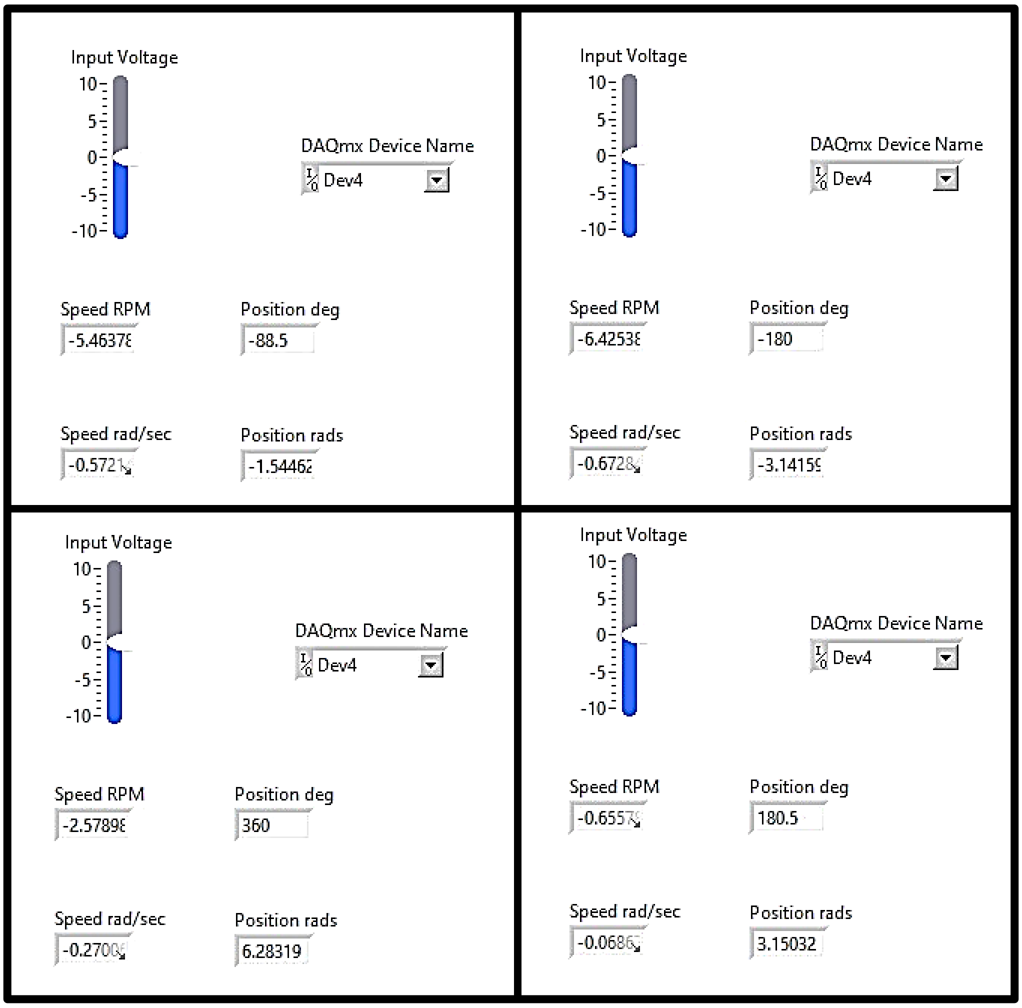

Figure 5: Motor rotated to -90, 180, 360, 180 degrees

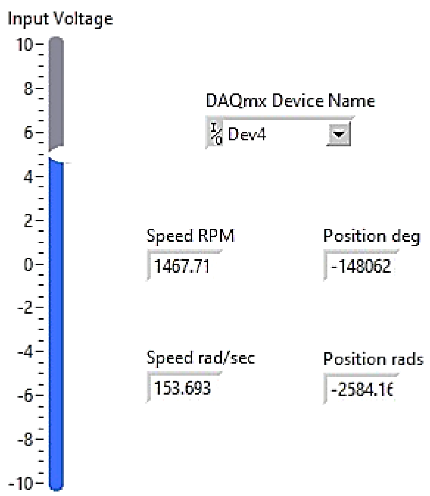

Figure 6: ~158 rad/s speed when 5V is applied

4.3 Exercise 2: Data Acquisition for Inverted Pendulum

The figure below shows which channels of the NI ELVIS board are used by the Inverted Pendulum and for what signal.

1. Analog output channel AO#0 is used to generate the voltage to be applied to the motor.
2. Analog input channel AI#0 is used to acquire the armature current of the motor.
3. Digital input channel DI#0 is connected to the Motor encoder 0 and used to determine the position of the Motor Arm.
4. Digital input channel DI#1 is connected to the encoder 1 and used to determine the position of the Pendulum Link.
5. Analog input channel AI#4 is used to acquire the speed of the motor.

Create a VI which:

- Selects device number
- Applies Voltage to the Pendulum's Motor
- Measures Arm Angle in Degree and rad/sec
- Measures Link Angle in Degree and rad/sec
- Measures Motor speed in RPM and rad/sec

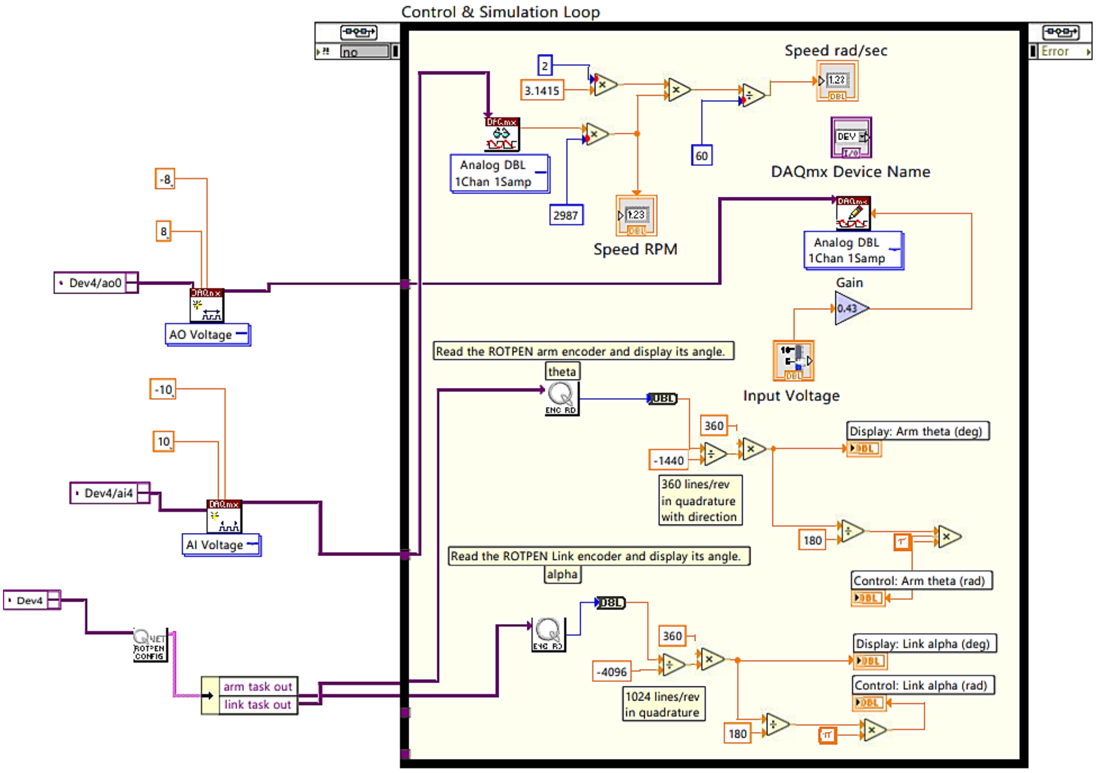

Figure 7: Block diagram of Inverted Pendulum

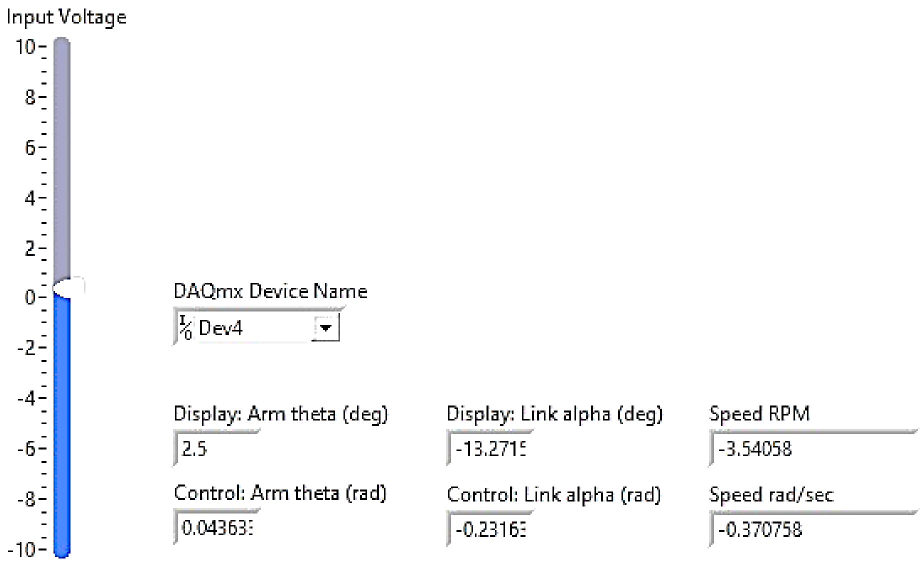

Figure 8: Front panel of Inverted Pendulum

5 Conclusion

In conclusion, this lab report aimed to introduce data acquisition and demonstrate the process of data acquisition using NI ELVIS and LabVIEW. Through this lab, we learned how physical signals can be captured, transformed, and analyzed by computers. We also gained practical experience in sampling analog signals and converting them into digital numerical values using NI ELVIS and LabVIEW. These skills are essential for any engineer or scientist who needs to work with physical systems or signals. In summary, this lab provided an excellent opportunity for us to learn and apply data acquisition concepts and techniques, which are essential for solving real-world problems in various fields.