# MECHENG 705: Mechatronics Systems

# Laboratory Exercise: QNET DC Motor Control Trainer

# Introduction

DC motors are very commonly used electromechanical actuators for mechatronics systems. Advantages of a brushed DC motor include low initial cost, high reliability, and simple control of motor speed. Disadvantages are high maintenance and low life-span for high intensity uses.

In this Lab you will be working with the QNET DC Motor Control Trainer (DCMCT) module which is integrated with the NI-ELVIS II board and NI Labview software. This kit is a simple platform which is used for modelling, control design and implementation of a DC motor system.

In this lab you will first get to know how the DCMCT system works, develop interfacing with Labview software through Data Acquisition (DAQ) and then carry out the first step in the process of developing a full control system: system identification and modelling.

# The QNET-DCMCT and NI-ELVIS System

This section outlines the basic system setup and components for the DCMCT system which will be used in this Lab and the upcoming assignment.

## Setup procedure

Follow the steps below to correctly connect up the DCMCT and NI-ELVIS. The complete DCMCT system is shown in Figure 1 below and some of the components are described in Table 1. Please DO NOT make any of the connections when there is power supplied to the hardware!

1. First place the DCMCT module, (7), over the ELVIS board and then slide the PCI connector of the DCMCT module into the female connector on the ELVIS board, make sure this is connected securely.
2. Connect the ELVIS power supply cable, (5).
3. Connect the ELVIS USB cable to the PC, (6).
4. Connect the DCMCT module to the transformer and to then to the mains power, (9).
5. Turn on the power switch at the rear of the board near the power cable (5) as well as the ELVIS board power switch, (2). If the motor begins to turn, switch the power off.
6. The POWER and READY LEDs, (3) and (4) should be lit.
7. The 4 green DCMCT power LEDs, (8), should be lit to verify the +15V, -15V, +5V and +BV are all supplied OK. This indicates the system has been setup correctly.

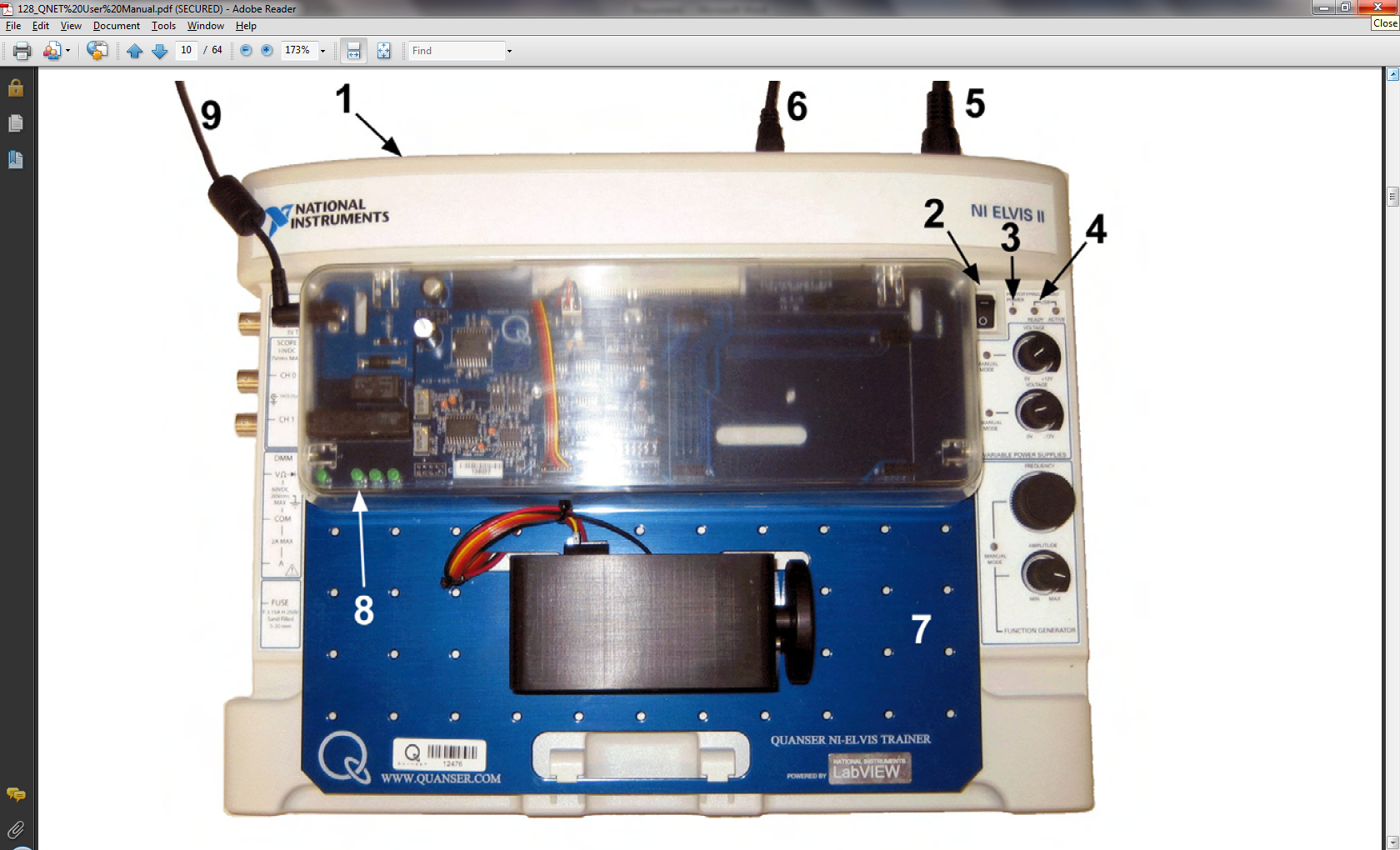


Figure 1. DCMCT and ELVIS board setup

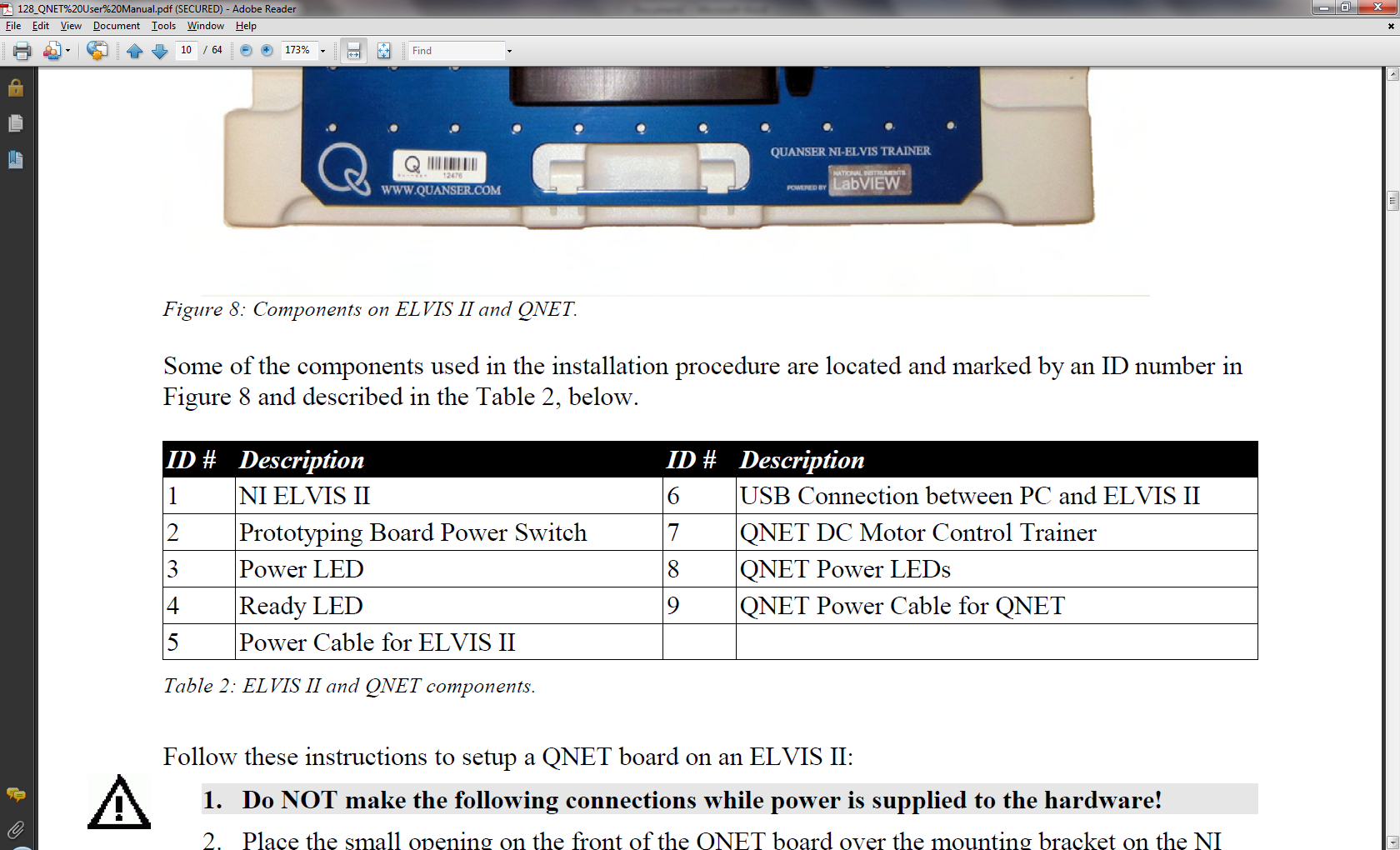


Table 1. DCMCT and ELVIS components.

## Layout and components of the DCMCT

A photograph and general layout of the DCMCT system is given in Figure 2 (a) below and a close up of the DC motor system components is given in Figure 2 (b). The components are described in Table 2.

|  |  |
| --- | --- |
|  |  |
| **(a)** | **(b)** |

Figure 2. QNET-DCMCT layout and DC motor components.

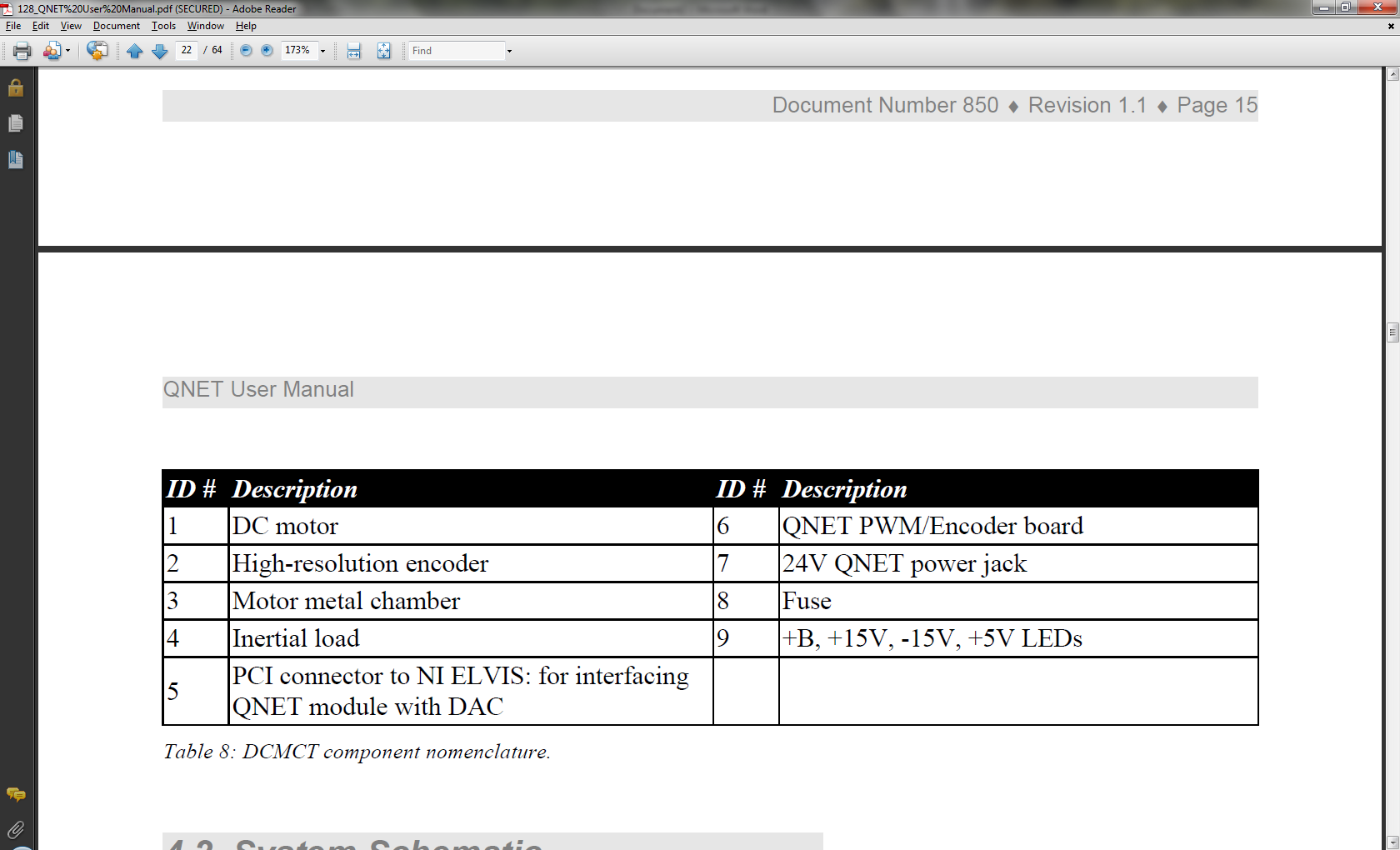


Table 2. DCMCT components.

## System DAQ Schematic

A schematic of the DCMCT system which is connected to the PC through the inbuilt DAQ in the ELVIS board is given in Figure 3. The inputs and outputs to the DCMCT can then be accessed in the same way that the standard NI DAQ cards are used through the Labview software, as you have done before in past courses.

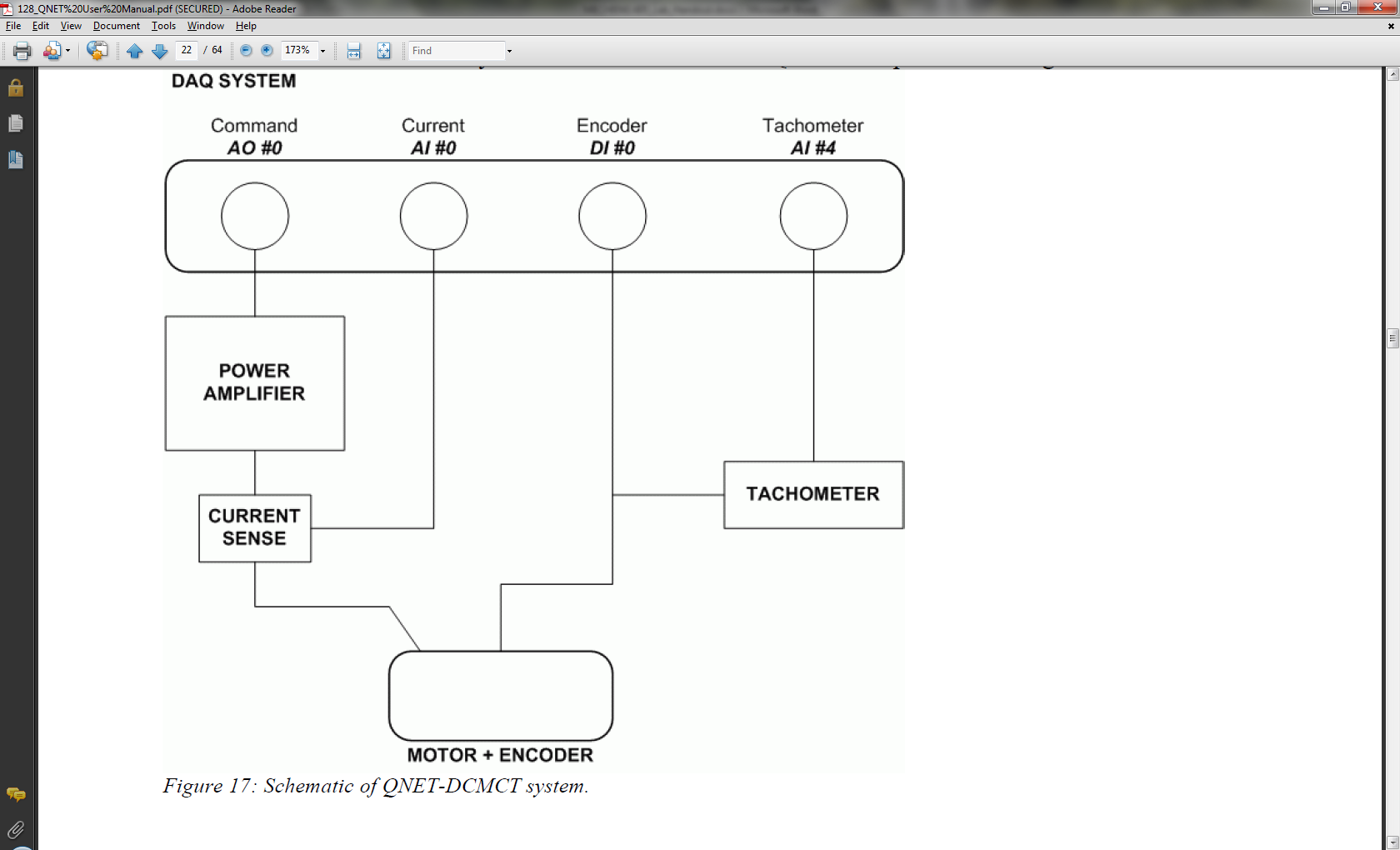


Figure 3. DAQ system schematic.

## DCMCT Component description

*DC Motor*

The system consists of a 12V DC motor which has 5 commutator segments, 64 windings per pole and has a flux ring. The Coulomb friction corresponds to a voltage between 0.5 and 1.5V.

*Pulse-Width Modulated (PWM) Power Amplifier*

A PWM power amplifier is used to drive the motor based on the input from the digital-to-analog converter (D/A) of output channel *AO#0* on the DAQ. The maximum output of the amplifier is 24V and the gain is 2.8V/V (Make sure you take this gain into consideration before you send a voltage output from your software so you don’t give the motor more than ±12V). The maximum peak current is 5A and the maximum continuous current is 4A.

*Analog Current Measurement-Current Sense Resistor*

A 0.1Ω load resistor is connected in series with the output of the PWM amplifier to measure the current drawn by the motor. The signal is amplified by the hardware internally and results in a sensitivity of 1.0A/V which can be collected at the analog-to-digital (A/D) channel *AI#0* of the DAQ.

*Digital Position Measurement: Optical Encoder*

A high-resolution optical encoder is mounted at the rear of the DC motor. The encoder count is available through the digital input channel *DI#0* of the DAQ. The TTL encoder runs 0.25 deg/count.

*Analog Speed Measurement: Tachometer*

An analog signal proportional to the motor speed is digitally derived on the board hardware from the encoder count; this is available through A/D input channel *AI#4* on the DAQ. The tachometer calibration is 1070 RPM/V.

*Fuse and Power Supply*

The DCMCT power amplifier has a 250V 3A fuse. The DCMCT module has a 24V DC power jack to power the on-board PWM amplifier. The +B LED shines green when the amplifier is powered.

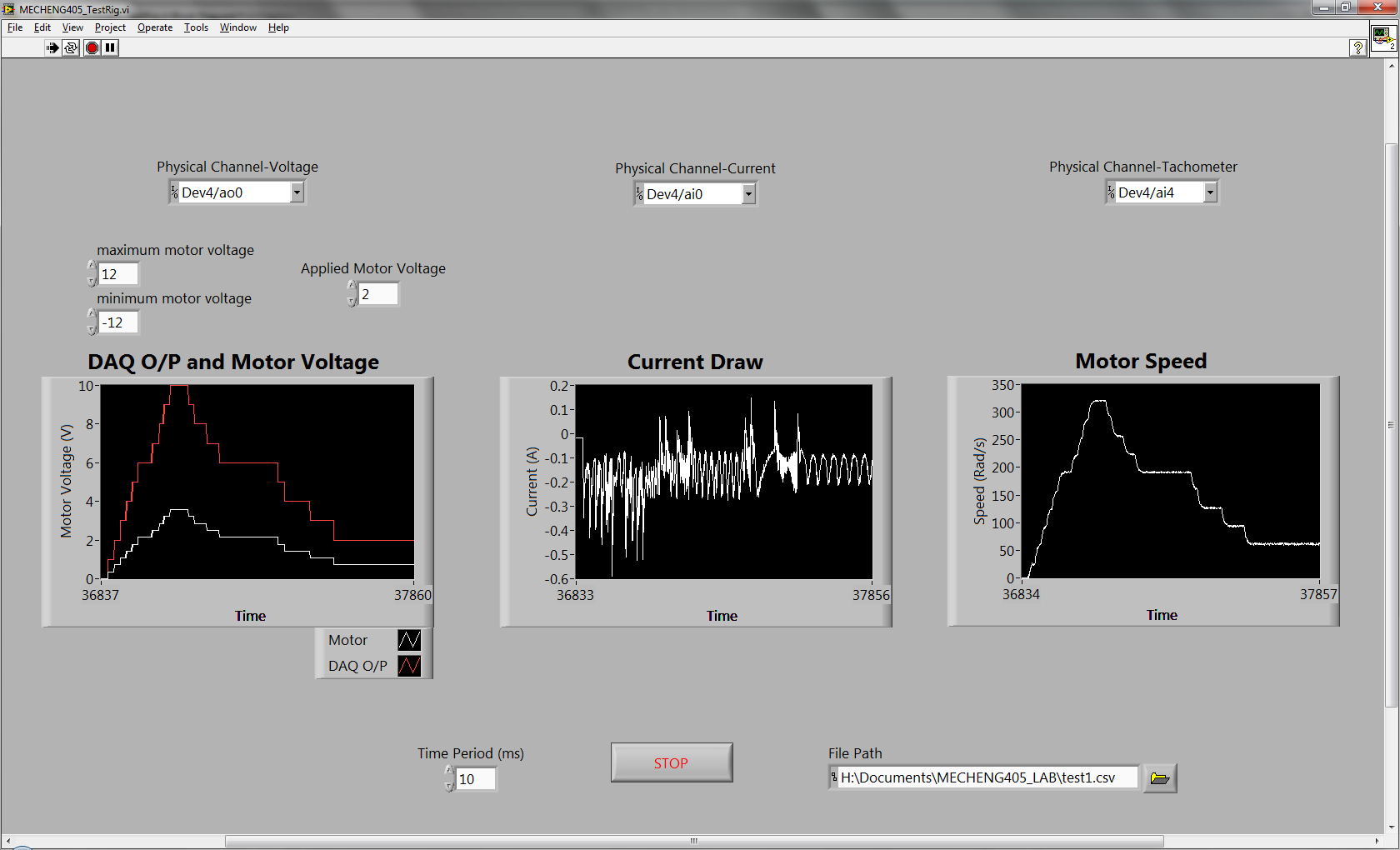
# Interfacing with the DCMCT System

In order to control a mechanical actuator to perform some desired task, you must send output signals to the system and read the feedback signals to determine the actual response of the system. Labview software with the DAQ hardware is used to achieve this for the DCMCT system. A voltage output to the motor must be created and sensor inputs must also be created to determine the state of the motor, i.e. position/speed, by reading the sensors.

There are many ways in which you can write a Labview program to interface with the DAQ and motor. Labview programs are made up of Virtual Instruments (VI) and other maths and I/O functions. The user can change variables in the system through controls on the front panel and can view results through indicators on the front panel. The Lab and upcoming assignment concentrate on system modelling and control and so this Lab does not go into depth with Labview programing. For more information on building Labview programs, you can read the Help menu or open example programs which are inbuilt with Labview. Also if you cannot find a certain VI or function, use the search on the functions palette.

Below in Figure 4 is an example of a simple Labview program which can send a voltage signal to the PWM amplifier to power the motor and also read the current draw and speed from the tachometer.

The program shown below consists of two “tasks”, one for sending an output and one for reading inputs. A timed loop is used; inside this the actual DAQ tasks place while the experiment is running. For each task a channel must be “created”, then “started” at the start of the program and then “stopped” and “cleared” at the end of the program, i.e. after the loop. While the program is running, i.e. inside the loop the program should either “read” or “write” to the channel. The VIs for DAQ which will be needed can be found in the functions palette by right clicking on the block diagram under, Measurement I/O>>NI-DAQmx. Also all displays should be inside the loop so the plots dynamically update. The data can be saved to a file for processing at the end at the end of the program, VIs for this can be found in Programming>>File I/O.



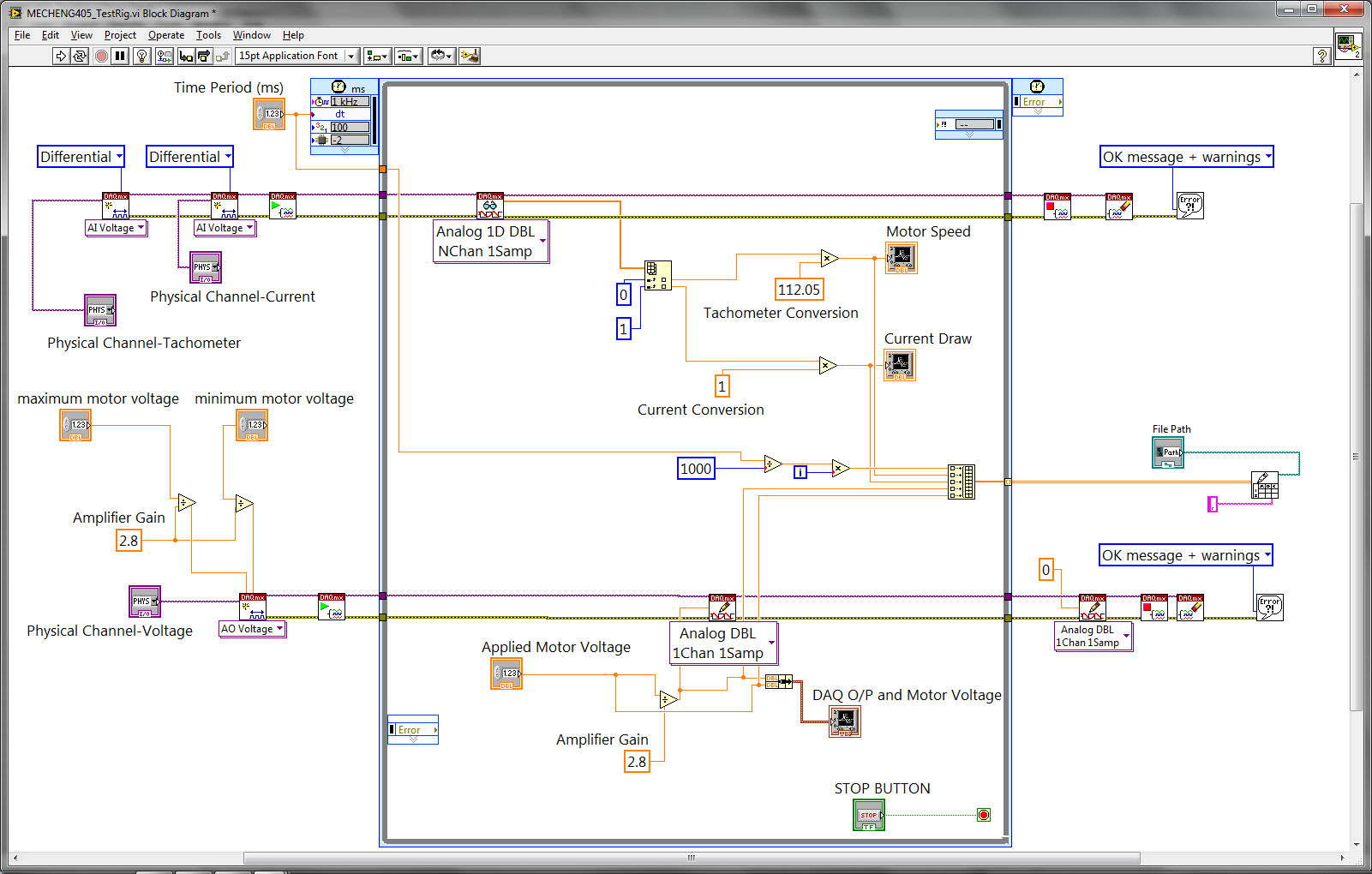


Figure (a) Front panel and (b) block diagram for a simple Labview program to send a voltage to the motor and read the current draw and tachometer.

**TASK 1.**

You are required to write a Labview program which can

1. Send a voltage to the PWM amplifier
2. Calculate the actual voltage applied to the motor from the amplifier
3. Read the analog input from the tachometer
4. Read the input from the current sensor
5. Convert these sensor readings to SI units (speed in rad/s, current in amps)
6. Display the voltage applied, speed and current to the front panel
7. Save the results to a file
8. Set the output voltage to 0V when the STOP button is pressed

**IMPORTANT NOTE:** Ensure the minimum and maximum voltage limits are set for the DAQ voltage output to the PWM amplifier to ensure the voltage applied to the motor does not exceed 12V!

# Motor Modelling

In order to design a control system for the motor it is important to know the motor characteristics to determine how the motor behaves. This is done through modelling the electrical and mechanical properties of the motor.

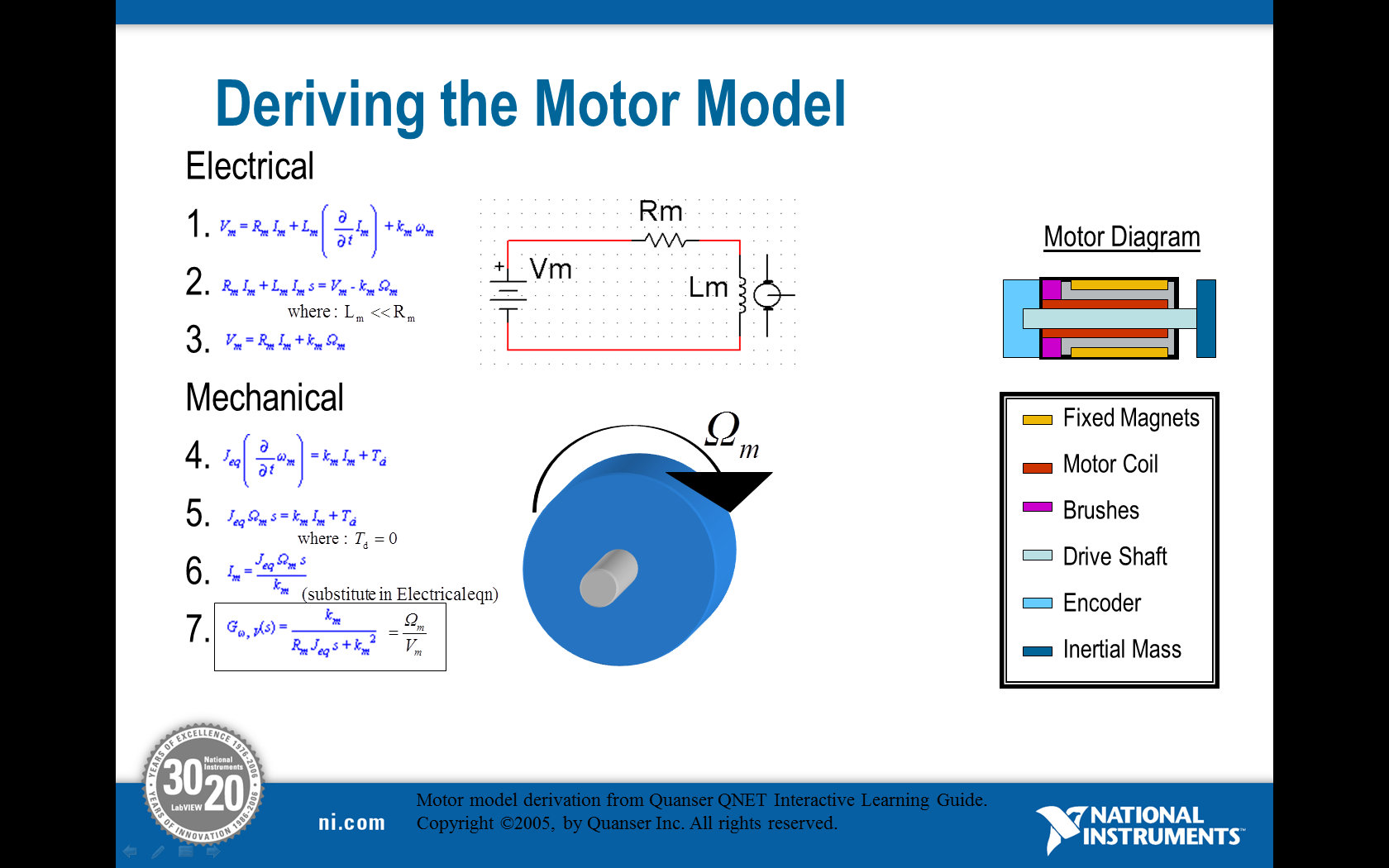


Figure 5. Deriving the motor model

From the above derivation it can be seen the electrical equations governing the open lop response of the motor are:

|  |  |
| --- | --- |
|  | (1) |

and

|  |  |
| --- | --- |
|  | (2) |

The equations governing the mechanics of the motor are:

|  |  |
| --- | --- |
|  | (3) |

and

|  |  |
| --- | --- |
|  | (4) |

where

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Units |
|  | Motor terminal voltage | V |
|  | Motor terminal resistance | Ω |
|  | Motor armature current | A |
|  | Motor torque constant | N.m/A |
|  | Motor back-emf constant | V/(rad/s) |
|  | Angular velocity | Rad/s |
|  | Torque produced by motor | N.m |
|  | Inertia of the motor armature and load | Kg.m2 |

Table 3. Motor model parameters

In order to describe the motor behaviour the motor parameters, , , and must be found.

**TASK 2.**

Find the motor terminal resistance parameter, .

Combining equation (1) and (2) gives

|  |  |
| --- | --- |
|  | (5) |

Which contains two unknown parameters, and . If the motor is constrained to not spin, i.e. then disappears from the equation and can be found from

|  |  |
| --- | --- |
|  | (6) |

where is the current when the motor cannot move.

Fix the motor shaft using an M3 screw through the hole in the inertial load, (4) in Figure 2, to the motor casing, (3) in Figure 2.

**NOTE:** Make sure the screw is secure in the tapped hole in the motor before applying any voltage to the motor to avoid any damage to you or the equipment!

Increase the motor voltage from -5V up to 5V in 1V increments, recording the stall current at each step. Using this data calculate the estimated resistance at each step and then average this out to find the average electrical resistance, .

**TASK 3.**

Find the motor torque constant, .

In SI units the motor torque and back-emf constants which couple the electrical and mechanical domains are equal, i.e. . By combining and rearranging equations (1) and (2) the following relation is found for , which can be used to estimate its value.

|  |  |
| --- | --- |
|  | (7) |

By substituting the average value for found in Task 2, the motor torque constant can be calculated for different input voltages. Increase the motor voltage from -5V up to 5V in 1V increments recording the motor speed and current. Calculate for each voltage and hence find the average motor torque constant for this motor.

**TASK 4.**

Simulate the motor model to find the inertia of the motor armature and load, and verify the developed model.

First you should be able to prove that by combining and rearranging equations (1)-(4) the system transfer function which describes the motor behaviour can be found as

|  |  |
| --- | --- |
|  | (8) |

This is a first order dynamic equation and so the dynamic response must be analysed to calculate the last parameter, . Note that this may also be measured using /2 for a rotating disc. The data sheet for the motor states the inertia for the inertial disc is 0.000015kg.m­­2 but the motor shaft itself also adds some inertia to the system. In this step you will simulate the system model for a step input with the given value for and compare this with the experimental results. Adjust the value for appropriately then so your model matches the actual system reasonably well.

In Labview a library called Control Design and Simulation allows you to simulate a real systems response to a number of input signals as well as view the time domain analysis, frequency response, pole zero plots etc. For this example you will need to simulate the motor response to a step input and compare this with the actual response to tune the parameter.

Start by creating a new Labview program to simulate the system model.

Draw a simulation loop, found under Control Design & Simulation>>Simulation.

In this loop add three controls which represent the system parameters, , and .

Add a SISO construct transfer function block found in, Control Design & Simulation>>Control Design>>Model Construction.

Add a simulation transfer function block found in Control Design & Simulation >>Simulation >>Continuous Linear Systems.

Add a simulated step input found in Control Design & Simulation >>Simulation >>Signal Generation.

Also add a plot so you can view the simulation.

Now connect these blocks up correctly so you can simulate the system response. The SISO construct transfer function block takes array inputs so build arrays with the coefficients for the numerator and denominator in them before wiring these to the block. Also double click on the simulation transfer function block and set the Parameter Source to ‘Terminal’ and wire the SISO construct transfer function block to this terminal. In this way you can dynamically change the motor parameters and the simulation will update automatically.

Before beginning the simulation double click on the box on the top left of the simulation loop to set the simulation parameters, e.g. solver type, simulation time, step size etc.

Figure 6 below shows the block diagram of the program used to run the simulation.

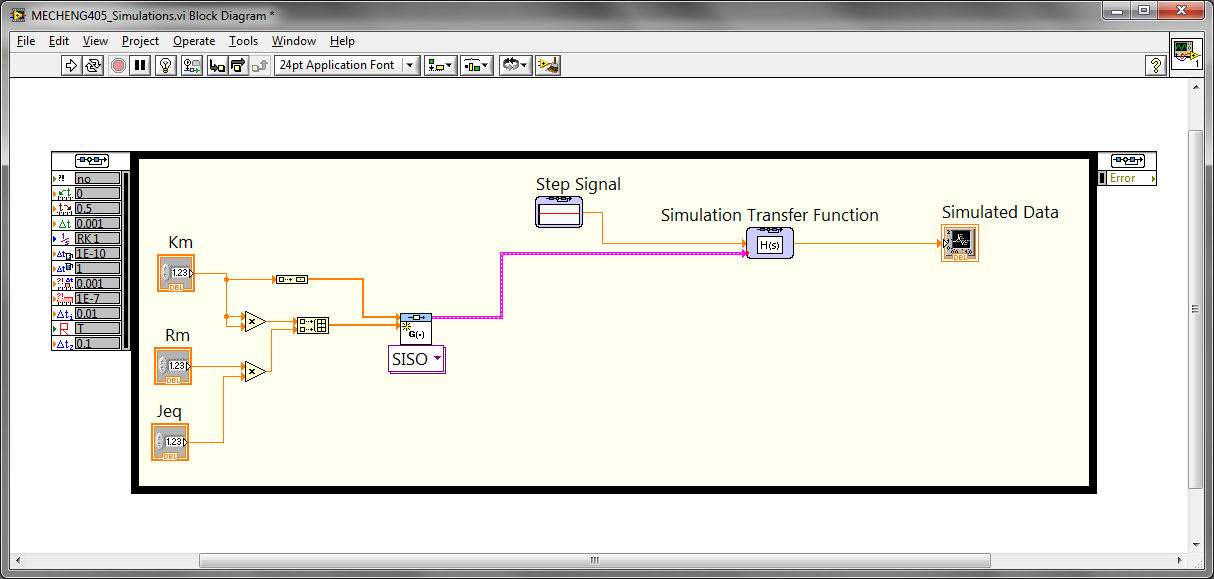


Figure 6. Block diagram of the system to be simulated

View the simulation results and compare this with the experimental step results from the motor to update the motor inertia parameter. Also double check to ensure that the other motor parameters, , , are correct when comparing the simulation with the experiments.

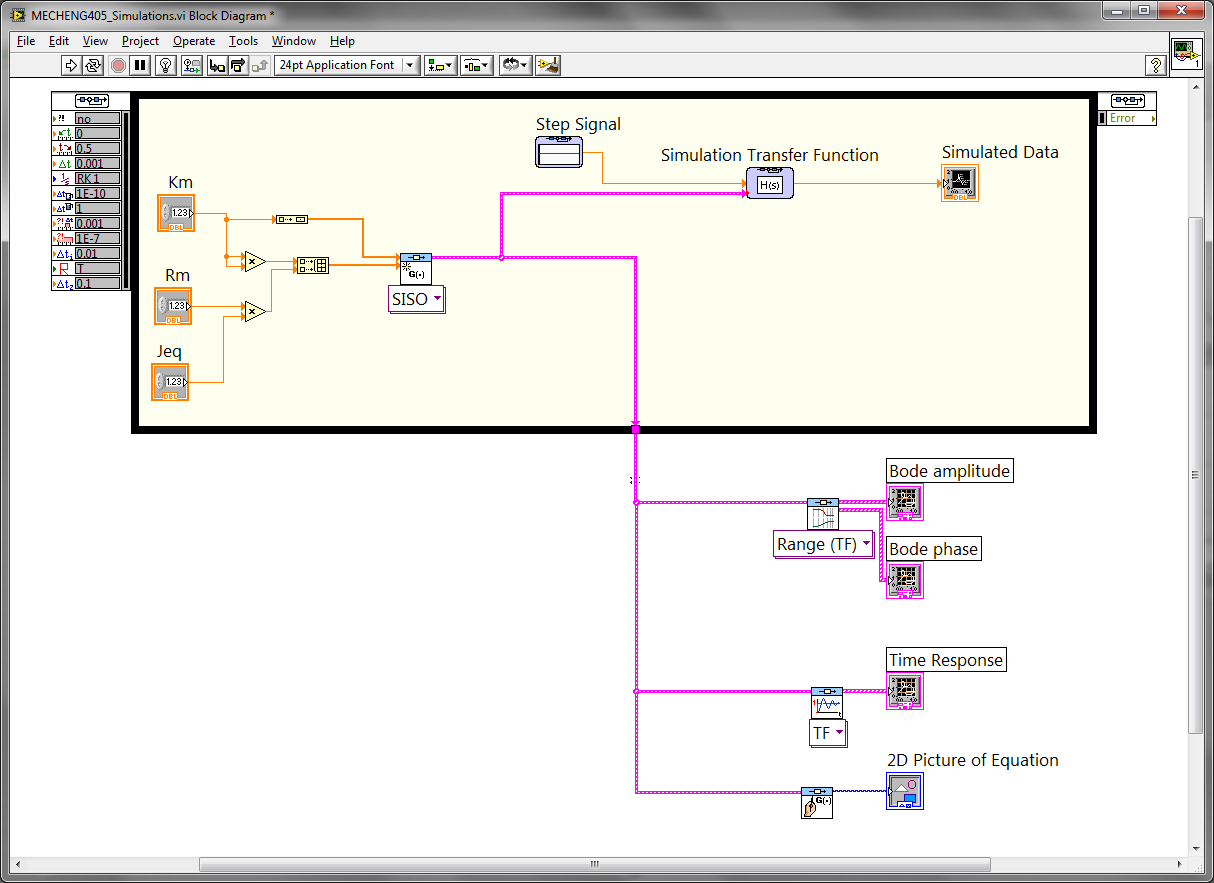
# Conclusions and Extra Analysis

You have now modelled the open loop response of a DC motor. This model can now be used to predict how the motor will act under many circumstances without actually having to undertake the actual experiment. This is very useful for designing mechatronics systems, for example if you have an application with certain requirements it is easy to simulate the motor in the application to see if it can perform as required.

The model can also be used to develop closed loop controllers to control the motor performance in tracking a reference trajectory or rejecting external disturbances.

After you have designed the simulation program it is easy to simulate the motor with different parameters and view the effect they have on the system response, e.g. what happens to the motor output if the resistance is increased? This is very useful for design and analysis.

There are many other features in the Control Design & Simulation library which make designing systems very easy, for example using bode plots for frequency analysis and step and impulse responses for time domain analysis. All of these can be utilized once a model has been developed for the system. Figure 7 below shows an example of bode plots, time response, the simulated data and the a dynamic drawing of the equation for the motor transfer function. The parameters can be dynamically updated and the plots will instantly reflect the change.



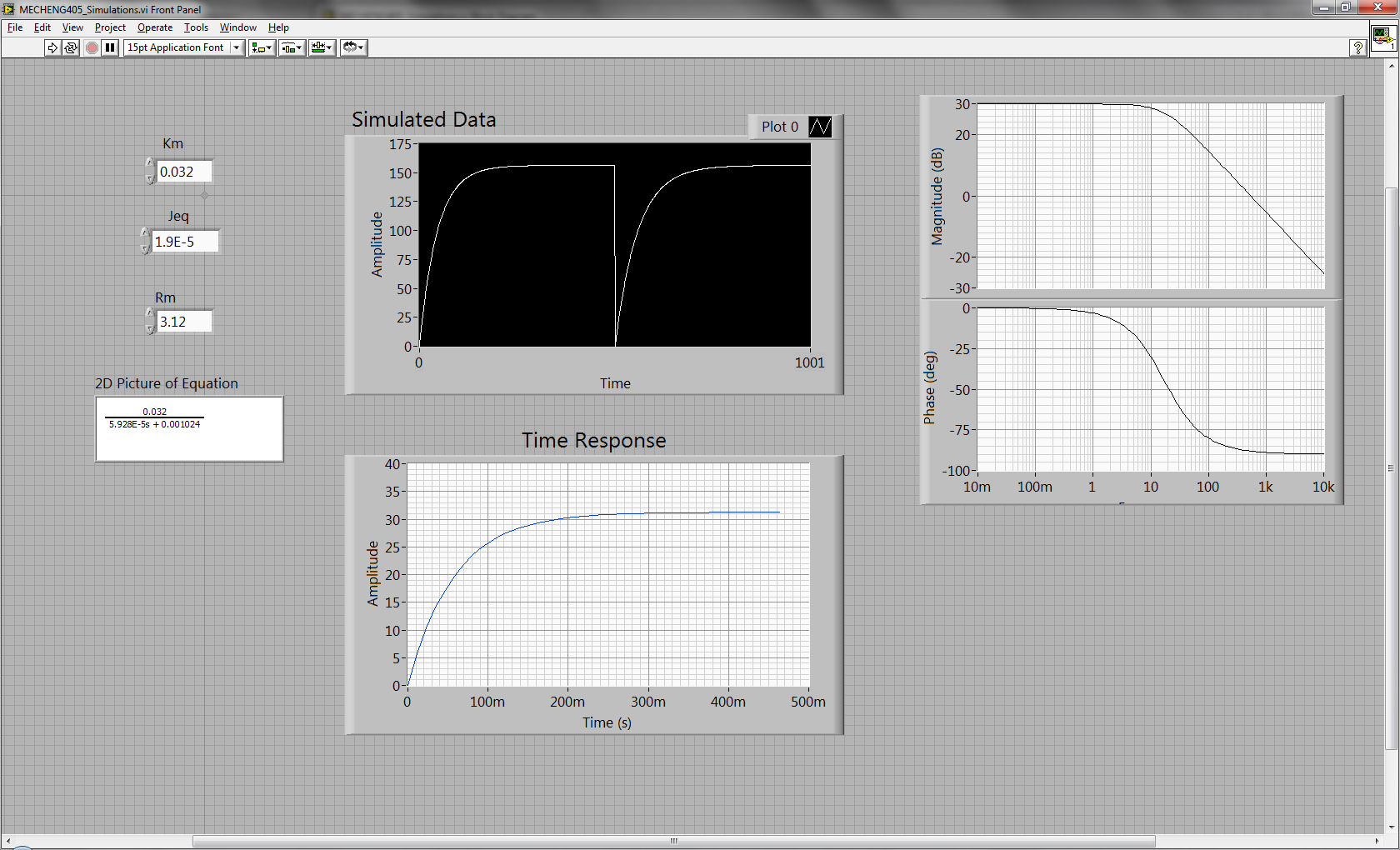


Figure 7. Front panel and block diagram of system analysis in Labview

Figures 1,2,3 and 5 have been reproduced from, Quanser Engineering Trainer for NI-ELVIS: QNET User Manual.