5-2018

Developing a HIL-Based Software Platform for Testing Electric and Hybrid Vehicle Powertrains

Daniel Schwartz
University of Arkansas, Fayetteville

Developing a HIL-Based Software Platform for Testing Electric and Hybrid Vehicle Powertrains

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Electrical Engineering

by

Daniel Harrison Schwartz
University of Arkansas
Bachelor of Science in Electrical Engineering, 2016

May 2018
University of Arkansas

This thesis is approved for recommendation to the Graduate Council.

_____

Juan Carlos Balda, Ph.D.
Thesis Director

_____

H. Alan Mantooth, Ph.D.
Committee Member

_____

Simon Ang, Ph.D.
Committee Member

ABSTRACT

The objective of this thesis is to present a dynamometer test stand with hardware in the loop (HIL) testing capabilities for the evaluation of electric and hybrid vehicle powertrains under realistic driving conditions. The traction inverter and dc/dc converter are two crucial subsystems of the powertrain and new prototypes require significant validation before implementing into production vehicles. As fundamental technologies such as power modules, thermal management systems, and passive components improve; the performance improvements to the overall system must be demonstrated. The plug and play nature of the testbed makes it ideal for making direct comparisons of design iterations under realistic conditions while mitigating the complexities and safety concerns of mobile testing platforms.

In this thesis the mathematical models of the major vehicle components such as the electric motor, tires, and vehicle body will be developed. The vehicle simulation that was developed based on these mathematical models will then be presented with a discussion of the additional simulation blocks needed to accommodate the real-time simulator and the simplifying assumptions that were made in order to make the model suitable for HIL simulation. Then the individual components and physical layout of the dynamometer test system are presented. It is followed by a detailed description of the graphical user interface and communication protocol implemented between the testbed control system and the prototype inverter. Finally, the simulation and measurement results are compared to the output of NREL's Advanced Vehicle Simulator to prove that the system can emulate realistic driving conditions.

ACKNOWLEDGEMENTS

I would like thank my advisor Dr. Juan Carlos Balda for this opportunity to complete a Master's Degree in Electrical Engineering and his support and guidance throughout my time as a graduate student. I would also like to express my gratitude to my committee members Dr. H. Alan Mantooth and Dr. Simon Ang for their support and encouragement. I would especially like to thank Chris Farnell for his advisement and mentorship throughout my graduate career.

I would also like to thank my friends and fellow students Andrea Wallace, Audrey Dearien, David Rojas, Hazzaz Mohammad, Rana Alizadeh, Ross Gaudet, Tristan Evans, Vinson Jones, and Zeke Zumbro.

DEDICATION

To Mom, Dad, and Wes. Their unyielding love, support, and encouragement in all areas of life has helped me become the first in my family receive a graduate degree. For that, and many other reasons, I am eternally grateful.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Motivations for Research: Electrification of the Automotive Industry

The electric vehicle was first developed alongside the internal combustion engine (ICE) in the early 1900's and for a brief period in the late 1920's EV registrations outnumbered combustion engines 3:1. By the 1930's however, combustion engine technology had matured to a level where they could be mass-produced at a cost much lower than that of an EV. The Ford Model T was the first mass produced vehicle that was widely available to the public. With its combination of affordability and increased performance, the ICE vehicles surpassed the EV in the market. In addition, the lack of reliable electric power transmission and charging infrastructure coupled with the public's desire to travel long distances and the overabundance of fossil fuels meant the EV designs of the time no longer fit the customer demand. As a result EV production stalled from the late 1930's to the mid 60's when the U.S. government enacted tougher fuel economy standards for industry in an effort to promote energy independence from foreign sources. The energy crisis in the 70's prompted the US postal service to order an EV test fleet, however yet again the technology had not matured to a level where they could meet all customer requirements [1].

From the mid 90's to present day there has been a large resurgence in research and production of EVs and HEVs. Both an increased initiative to reduce our greenhouse gas emissions and the limited supply of fossil fuels have contributed to this shift in the industry. As the resources of our planet continue to be depleted, energy security has become a major concern for governments across the globe [2]. In his 2006 State of the Union address, George Bush stated

"Keeping America competitive requires affordable energy. And here we have a serious problem: America is addicted to oil, which is often imported from unstable parts of the world. The best way to break this addiction is through technology" [3]. The automotive industry now has more than 20 different EV and HEV models for sale. A total of more than 500,000 EVs have been sold between their introduction into the market and December of 2016. According to a report published by ChargePoint, in 2012 the number of EVs on the road totaled 73,000 [4]. This statistic illustrates the incredible growth between 2012 and 2016 in this section of the industry. It is worthy to note that during this time frame many state and local governments began to offer tax incentives for those who purchase an EV or HEV [2].

The impact of the electrification of the vehicles on the road does not only affect the automotive industry. Grid enabled vehicles such as EVs and plug-in HEVs have the capability to contribute power to the grid. This would allow the consumer to operate their vehicle as a distributed generation source when electricity demand/prices are high and charge their vehicle when demand is low. However, the effects on the power distribution industry are not all positive. The widespread incorporation of EV and HEVs will drastically increase power demand but this is expected to be relatively gradual to allow the power distribution companies time to upgrade their distribution networks [2]. In addition, the majority of the power grid at the distribution level in the US was designed to be unidirectional. With the adoption of more Smart Grid technology, household meters or local substations will be able to control power flow from distributed generation sources. As the overall demand for energy across the globe continues to rise, innovative and efficient solutions like EVs and HEVs are necessary to fully utilize our natural resources.

The United States Environmental Protection Agency (EPA) reports that burning one gallon of gasoline results in approximately 8.9 kg of CO_2 emissions. The typical passenger vehicle emits about 4.7 metric tons of carbon dioxide per year along with other greenhouse gasses such as methane, nitrous oxide and hydrofluorocarbons [5]. One of the most prevalent greenhouse gasses in the atmosphere is CO_2. These tailpipe emissions can drastically be reduced or eliminated altogether by substituting a hybrid electric (HEV) or electric vehicle (EV) for a traditional ICE car. All EVs and HEVs that use an internal battery pack to provide energy require power electronics to convert this electric potential from DC to AC. To meet customer demands, these converters must have high power ratings to achieve the desired vehicle performances while remaining efficient to extend the autonomy range of the vehicle.

For hardware validation of EV and HEV powertrains, a typical dynamometer consists of a propulsion machine, a load machine, and a load motor drive [6]. By connecting the two motor shafts together, the electrical energy delivered to the propulsion machine will be converted to mechanical energy at the shaft, and then back to electrical energy by the load machine. The load inverter is controlled to provide a force in the opposite direction of the propulsion machine's rotation, thus creating a load torque for the system. This thesis will discuss the development of a hardware in the loop (HIL) based software platform for safe, realistic, and efficient testing of EV and HEV powertrains.

1.1.1 Hardware Validation of Large Power Systems

Whether it is the automotive, aviation, power distribution, or power generation industry there is an increased focus on high power electronics and increasing power density. Current research is looking into optimizing layouts and topologies as well as utilizing new power semiconductor devices based on wide bandgap materials in order to extend the limits of

performance. A prototype must be tested to the design specifications to validate a new design. For most applications, it would be incredibly costly and potentially dangerous to test unproven designs in an end product such as a mobile vehicle or power distribution network. For this reason, an intermediate testing platform between low-power proof of concept testing and mobile or end application platforms is needed. Most academia or industry research proposal budgets do not provide additional funds for procuring test equipment and often times the cost of the test equipment would be a substantial portion of the overall budget leaving little funding for the actual prototype. The dynamometer testbed at the University of Arkansas was constructed to address this need of both industry and academia. The testbed provides an efficient and cost-effective solution for testing powertrains up to 100 kW. The only power consumed by the test can be attributed to the losses within the system, the remaining power is circulated back onto the power grid. The testbed is augmented with HIL testing capabilities which allow real-time simulation of the vehicle physical dynamics which results in accurate speed and torque test profiles for the powertrain.

The testbed must be able to emulate real world conditions to give the designer the best possible feedback on the validity and performance of their design. For EV powertrain testing, non-idealities such as ambient temperature and other environmental parameters must be incorporated as well as a realistic vehicle and driver model. The data acquisition system must also be adequate to capture signals that have very fast transients or high frequency harmonic components.

1.2 Advanced Vehicle Modeling Software (ADVISOR)

The National Renewable Energy Laboratory first developed the software ADVISOR in 1994 to support the U.S. Department of Energy (DOE) hybrid propulsion system program. This software is an empirical model designed to quickly and accurately simulate the performance of a HEV. It incorporates drive train component performance to estimate fuel economy/emissions during driving cycles as well as overall range of a vehicle. In order to meet the goals of the DOE hybrid propulsion system program, ADVISOR is accurate so that meaningful comparisons of different drivetrain configurations can be accomplished. The software allows high-speed analysis of vehicles and design space investigations, and its flexibility allows evaluation of vehicles with various control strategies and components. In addition, it is publicly available allowing for collaborations and to foster HEV development and understanding. Lastly, it is easy to use even without technical knowledge of vehicle modeling and can simulate conventional, electric, and multiple hybrid configurations.

ADVISOR software uses a hybrid backward/forward simulation approach that is unique in the way it handles component limits. In essence, a backward facing approach assumes the vehicle has met the required trace, and works in the opposite direction of power flow to determine the requirements of the drive train components. A forward facing approach uses a driver model and PI controller to work in the same direction of power flow to determine the vehicle speed achieved with the specified components. ADVISOR's combination of the backward/forward approaches resulted in two overriding assumptions. First, the drive train component will require less torque of power from its upstream component than it can use. Second, a component is as efficient in forward facing calculations as it was computed in the backward facing calculations [7].

Since its creation, ADVISOR has been very well documented and examined. Many case studies have tested the validity of the models. And its predictions of EV and HEV behavior constantly fall within the bounds of uncertainty for each step of the simulation process. ADVISOR's EV model is the most accurate since the transient effects of an electric drive train are fairly small. In HEV models, the accuracy decreases as the transient effects of the ICE are increased. This means that the simulation would be accurate for small changes in ICE loading, but accuracy would decrease as the changes in load became larger [8]. With its few limitations and accurate simulation of EVs and HEVs, ADVISOR is a good candidate for comparison to the vehicle model that is developed later in this thesis.

1.3 Real Time Simulators for Hardware in the Loop Capability

Hardware-in-the-loop (HIL) simulation requires dynamic models for the system being evaluated. It must include all significant mechanical and electrical interactions within the EV or HEV. For this application, a real-time simulator will be used to generate realistic torque and speed profiles for the dynamometer based on a user selected vehicle type and user defined environmental input variables. Typhoon HIL, dSPACE, and OPAL-RT were considered for the real time simulator used for this testbed [9-11].

Typhoon HIL [9]

Typhoon HIL 6 series can use up to 6 cores for processing a model in real time. The number of cores necessary to achieve the desired time step varies based on model complexity, however the minimum time step achievable is 500ns. It has 16 analog inputs, 32 analog outputs, and 32 digital inputs and outputs. Typhoon HIL requires the use of its HIL software and does not interface with other simulation software. This results in limited part libraries and an additional amount of time to learn a new simulation software.

dSPACE [10]

dSPACE uses commercial off the shelf components to build its HIL systems. Their mid-sized simulator is based on their DS2211 HIL board which is designed for automotive applications with ICEs rather than EV simulations. It has a special processing unit ideally suited for handling crankshaft angle signals. It has 10 analog inputs or output ports and 32 digital I/O ports. The datasheets and information provided do not include a figure for minimum time step, as this depends on model complexity. dSPACE also offers its Real-Time Interface software that implements MATLAB/Simulink™ models on the hardware.

OPAL-RT [11]

The OPAL-RT OP5031 simulator has the latest generation of Intel processors with up to 32 cores. When paired with an OP4520 expansion, a Xilinx Kintex 7 FPGA and I/O capabilities are added. It can provide up to 128 high performance analog/digital channels with signal conditioning as well as 4 optical high speed lings for interfacing with hardware. With these powerful processors, the OP5031 and OP4520 expansion can reach a minimum of 200ns. It also has Ethernet, RS-232, USB, JTAG, and VGA connections for interfacing with hardware. Like dSPACE, OPAL-RT's software package RT-Lab will interface with Matlab/Simulink with minimal modifications to the stand alone Simulink model.

OPAL-RT was chosen for the testbed for its industry leading minimal step size, multitude of hardware interface connections, and its versatility as a simulator. In addition to the OP5031 unit, the OP4520 FPGA [12] expansion unit was purchased to provide additional I/O capabilities for communication with the data acquisition system and for future controller hardware in the loop (CHIL) testing.

1.4 EPA Driving Schedules

The U.S. EPA is responsible for designing government programs to reduce and prevent air pollution. To do so, it has developed and standardized emissions testing for ICE vehicles to ensure they meet the current standards. The National Vehicle and Fuel Emissions Laboratory (NVFEL) in Ann Arbor, Michigan, is used to measure emissions on a wide range of vehicles. In order to standardize the testing, they needed driving profiles for the vehicles that would emulate common driver behavior in real world environments like driving in a city, highway driving, etc. Figure 1.1 below illustrates the most common schedule used for state emissions certifications in California [13].

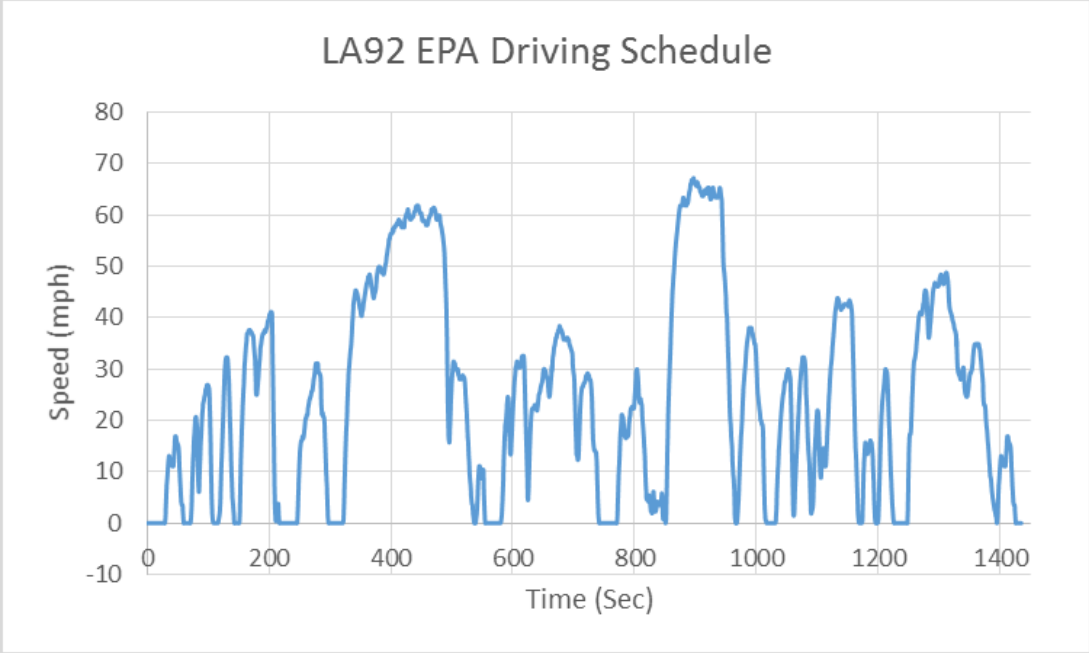

Figure 1.1: California EPA Air Resources Board LA92 Driving Schedule

1.5 Organization of Thesis

This thesis is organized as follows: Chapter 2 will develop the mathematical models for the major vehicle components that will be considered in the OPAL-RT HIL simulation. Specifically, the modeling of the physical dynamics of the vehicle body, the tire-to-road interaction, electric motors, and the battery pack. Chapter 3 will focus on the Matlab/Simulink™ vehicle model developed by providing a block-by-block walkthrough of the block diagram. The vehicle powertrain components such as the battery, electric motor, and power split device (PSD) will also be discussed along with the assumptions that were made during development of the vehicle model. Chapter 4 will discuss the dynamometer testbed, including a topology overview, the load motor drive, the LabVIEW™ data acquisition and control interface, the OPAL-RT real time simulator, and testbed to unit under test (UUT) communication protocols. The fifth chapter will present an overview of the testing capabilities, driving schedules, and maximum acceleration limits of the system before discussing test results from proof-of-concept testing with a Baldor H2 Vector drive. Conclusions and recommendations for future work will be addressed in Chapter 6.

1.6 References

[1] L. Situ, "Electric Vehicle Development: The Past, Present & Future", in *3rd International Conference on Power Electronics Systems and Applications*, 2009.

[2] A. Boulanger, A. Chu, S. Maxx and D. Waltz, "Vehicle Electrification: Status and Issues", *in Proceedings of the IEEE* Vol. 99, No. 6, IEEE, 2011.

[3] A. Shah, "Energy Security — Global Issues", Globalissues.org, 2009. [Online]. Available: http://www.globalissues.org/article/595/energy-security. [Accessed: 06- Oct- 2017].

[4] J. Bhuiyan, "There have now been over 540,000 electric vehicles sold in the U.S.", Recode, 2017. [Online]. Available: https://www.recode.net/2016/12/21/14041112/electric-vehicles-report-2016. [Accessed: 09- Oct- 2017].

[5] "Greenhouse Gas Emissions from a Typical Passenger Vehicle | US EPA", US EPA, 2017. [Online]. Available: https://www.epa.gov/greenvehicles/greenhouse-gas-emissions-typical-passenger-vehicle-0. [Accessed: 13- Oct- 2017].

[6] "Dynamometer", En.wikipedia.org, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Dynamometer. [Accessed: 05- Oct- 2017].

[7] K. Wipke, M. Cuddy and S. Burch, "ADVISOR 2.1: A User-Friendly Advanced Powertrain Simulation Using a Combined Backward/Forward Approach", in *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY*, VOL. 48, NO. 6, 1999.

[8] R. Senger, "Validation of ADVISOR as a Simulation Tool for a Series Hybrid Electric Vehicle Using the Virginia Tech FutureCar Lumina", MSME, Virginia Polytechnic Institute and State University, 1997.

[9] T. HIL, "Hardware in the Loop 602 | Typhoon HIL", Typhoon-hil.com, 2018. [Online]. Available: https://www.typhoon-hil.com/products/hil602. [Accessed: 05- Oct- 2017].

[10] "DS2211 HIL I/O Board", Dspace.com, 2018. [Online]. Available: https://www.dspace.com/en/pub/home/products/hw/phs_hardware/i_o_boards/hil_i_o_board.cfm. [Accessed: 05- Oct- 2017].

[11] "Real-time digital simulation platform │ Real time system │ OP5031", OPAL-RT, 2018. [Online]. Available: https://www.opal-rt.com/simulator-platform-op5031/. [Accessed: 05- Oct- 2017].

[12] "FPGA Expansion Box IO Expansion Box", OPAL-RT, 2018. [Online]. Available: https://www.opal-rt.com/hardware-io-expansion-box/. [Accessed: 05- Oct- 2017].

[13] "Dynamometer Drive Schedules | US EPA", US EPA, 2017. [Online]. Available: https://www.epa.gov/vehicle-and-fuel-emissions-testing/dynamometer-drive-schedules. [Accessed: 10- Oct- 2017].

CHAPTER 2
MODELING VEHICLE COMPONENTS

2.1 Vehicle Body

The equation for the physical dynamics of the vehicle is derived from Newton's second law, also known as the equation for solid body motion. This simplistic equation (Eq. (1)) can be extended to include all of the forces that act on a vehicle. Equation (2) provides a good method for describing straight line motion of a vehicle.

$$F = ma \tag{1}$$

$$F = mgC_{rr} + \frac{1}{2}\rho C_D A v^2 + ma + mgsin(\theta) \tag{2}$$

Where mgC_{rr} corresponds to the force required to overcome the rolling resistance of the vehicle. $\frac{1}{2}\rho C_D A v^2$ refers to the drag force the vehicle must overcome at speed and depends on the drag coefficient of the vehicle as well as the frontal surface area. The mass inertia of the system is given by ma, and $mgsin(\theta)$ describes the force requirement to drive a vehicle on a non-level surface. Equation (2) is widely used as the base equation for most vehicle modeling software [1]-[2] and is a good starting point for fully describing the motion of the vehicle. However, there is some room for improvement. Figure 2.1 shows the graphical representation of the vehicle body in motion and shows the model variables with their associated vectors. This figure will be used to develop the equations of motion used in the MATLAB/Simulink model discussed in Chapter 3. As discussed previously, vehicle motion is the result of the net effect of forces and torques within the vehicle. The weight of the vehicle is directed downward through the center of gravity

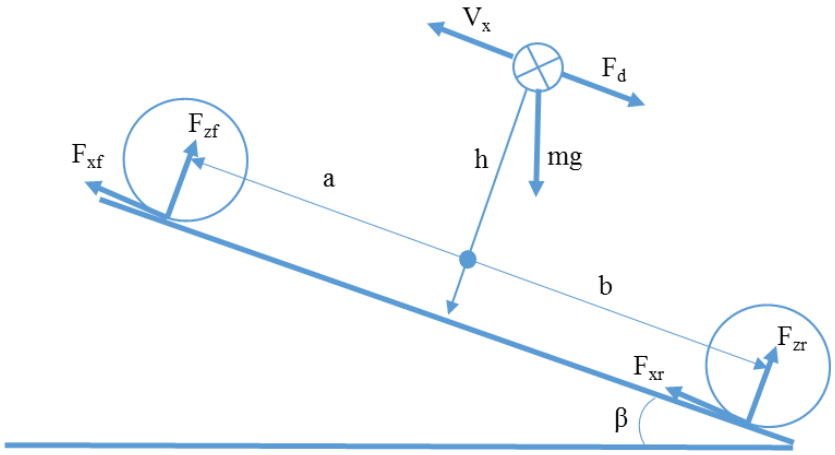

Figure 2.1: Vehicle Body in Motion

and depending on the road angle beta (β) the vehicle weight either pulls the vehicle to the ground (β=0), forwards (β<0), or backwards (β>0). In either direction of travel the aerodynamic drag force will work in the opposite direction of motion and will slow down the vehicle. The drag force will increase exponentially with vehicle velocity and wind velocity as seen in Eq. (5). It is assumed that the drag force also acts through the center of gravity. The improved vehicle equations are shown below. Table 1.1 lists the variables used in Eq. (3) to (7) and their respective meanings.

$$m\dot{V}_x = F_x - F_d - mg \cdot \sin(\beta) \tag{3}$$

$$F_x = n(F_{xf} + F_{xr}) \tag{4}$$

$$F_d = \frac{1}{2} C_D \rho A (V_x + V_w)^2 \sin(V_x + V_w) \tag{5}$$

The zero normal acceleration and zero pitch torque determine the normal force on each front and rear wheel as shown in Eq. (6) and (7). The torque required to reach a given speed command will be calculated using these equations. Whether the vehicle is front-wheel drive, rear-wheel drive, or all-wheel drive will determine which equation (Eq. (6), (7) or both) will be considered. The normal forces that are calculated for the axles are then used with a tire model that calculates the longitudinal behavior of the tire on a road surface [3].

$$F_{zf} = \frac{-h\left(F_d + mgsin(\beta) + m\dot{V}_x\right) + b \cdot mgcos(\beta)}{n(a+b)} \tag{6}$$

$$F_{zr} = \frac{-h\left(F_d + mgsin(\beta) + m\dot{V}_x\right) + a \cdot mgcos(\beta)}{n(a+b)} \tag{7}$$

Table 2.1: Vehicle Body Motion Variables

Symbol	Description and Unit
g	Gravitational acceleration
β	Incline angle
m	Mass of the vehicle
h	Height of vehicle CG above the ground
a, b	Distance of front and rear axles, respectively, from the normal projection point of vehicle CG onto the common axle plane
V_x	Velocity of the vehicle. When $V_x > 0$, the vehicle moves forward. When $V_x < 0$, the vehicle moves backward.
V_w	Wind speed. When $V_w > 0$, the wind is headwind. When $V_w < 0$, the wind is tailwind.
n	Number of wheels on each axle
F_{xf}, F_{xr}	Longitudinal forces on each wheel at the front and rear ground contact points, respectively
F_{zf}, F_{zr}	Normal load forces on each wheel at the front and rear ground contact points, respectively
A	Effective frontal vehicle cross-sectional area
C_d	Aerodynamic drag coefficient
ρ	Mass density of air
F_d	Aerodynamic drag force

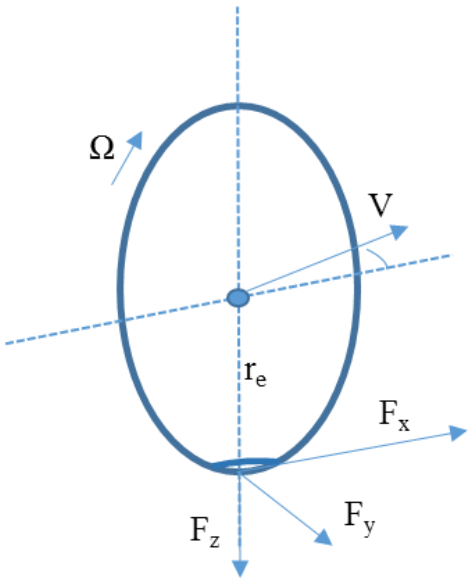

Figure 2.2: Tire Forces

2.2 Tires and Tire-to-Road Interaction

Figure 2.2 shows a diagram of the typical forces that act on a tire. For a freely rolling wheel, the effective rolling radius (r_e) can be obtained by dividing the vehicle forward speed of the wheel center (V_x) by the angular speed of revolution (Ω_o) at the contact point. The effective radius can also be defined for a braked or driven wheel, but this approximation (Eq. (8)) will be well within tolerances for this application.

$$r_e = V_x/\Omega_o \tag{8}$$

$$V_{sx} = V_x - r_e\Omega \tag{9}$$

$$k = -\frac{V_{sx}}{V_x} = -\frac{V_x - r_e\Omega}{V_x} = -\frac{\Omega_o - \Omega}{\Omega_o} \tag{10}$$

When torque is applied about the wheel spin axis, a longitudinal slip arises. The wheel slip velocity can be calculated using Eq. (9). Where V_{sx} is the slip velocity and Ω is the wheel angular velocity. Automotive tires are pneumatic, meaning they are filled with air. Based on the weight of the vehicle, air pressure, and other factors the tires may deform which changes the effective radius of the wheel at the ground point of contact. Ω_o is the angular velocity of the road-tire point

of contact. The tire longitudinal deformation factor u is non-zero if the tire is deformed. If the deformation factor is zero, $\Omega_o = \Omega$. The wheel slip k is a value between -1 and 1 and is defined as the slip velocity divided by the longitudinal velocity (Eq. (10)). If k =-1 the vehicle is attempting to slow down and the wheels are locked and sliding. If k =1, the vehicle is not moving and the tires are spinning. Lastly, k=0 refers to a perfectly rolling wheel [4].

For the case where the vehicle is not moving, but the tires are spinning the resulting k value from Eq. (10) would be infinite. In order to obtain a finite answer in this case at low speeds ($|V_x| \leq |V_{th}|$ where V_{th} is the wheel hub threshold velocity) wheel slip is modified to Eq. (11).

$$k = \frac{2V_x}{V_{th} + {V_x}^2/V_{th}} \tag{11}$$

The interaction between the road and tire and the resulting longitudinal forces have been modeled in many different ways. Figure 2.3 illustrates some of the more common approaches of developing a tire model and shows the benefits and drawbacks of each. The left most category otherwise known as empirical models, include mathematical tire models that describe measured characteristics through tabulated values or mathematical formulas and interpolation schemes. These formulas are structured with the aid of regression procedures to yield a best fit to the experimental data. One example of an empirical tire model is the 'magic formula' developed in Chapter 4 of [4]. The model provides a good fit for F_x, F_y, and M_z which represent longitudinal force, side force, and aligning torque respectively.

The second category, the similarity method, is built on a number of basic characteristics usually obtained from measurements. Rescaling, distortions, and multiplications are necessary to

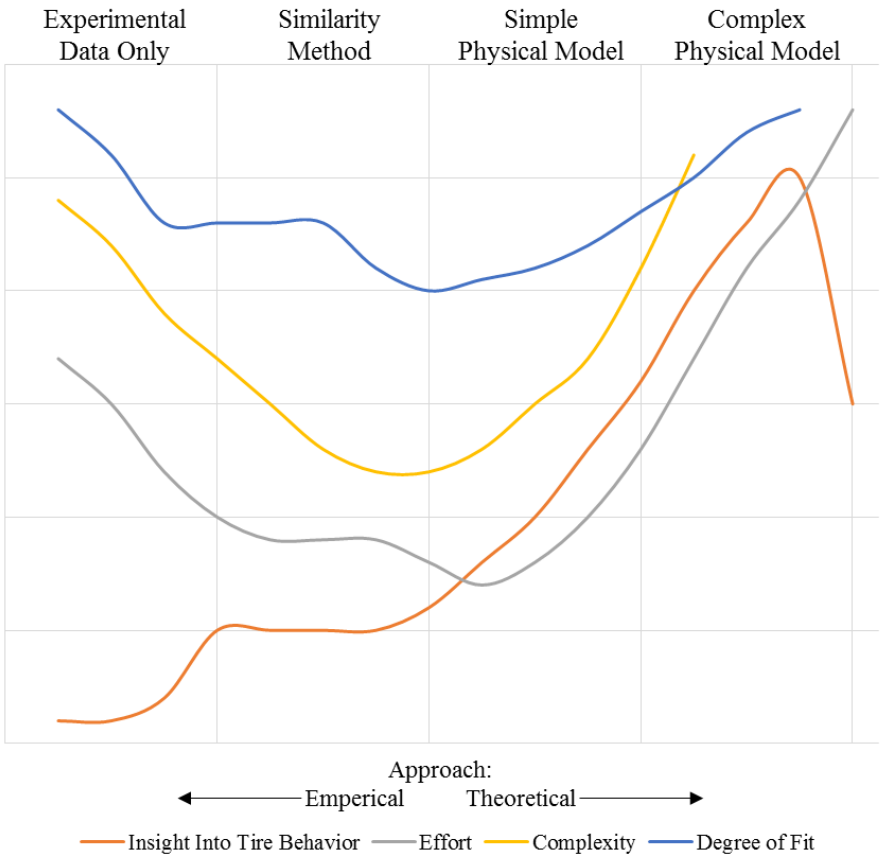

Figure 2.3: Tire-to-Road Modeling Approaches

describe other than normal operating conditions. This method is not computationally expensive and is good for applications that require rapid computation cycles. However, this method is not as accurate as the empirical model.

The simple physical models included in the third category are useful for getting a basic understanding of the tire behavior however, these models do not fit the characteristic curves and are not suitable for use in simulation.

The last group of models is aimed at more detailed theoretical analysis of the tire dynamics. Complex finite element analysis models can be quite computationally intense but are the most accurate and provide considerable freedom in choosing parameters such as pressure distribution, friction coefficient functions, and local contact pressure [4-5].

The magic formula tire model is the most commonly used empirical tire model for steady state tire force and moment characteristics. The 'magic formula' model was developed by Hans B. Pacejka in 1993 [4]. It accurately predicts the longitudinal forces arising from the interaction between the tread and road pavement. It can be represented in two ways: with constant formula coefficients and with load dependent coefficients. The general form for the longitudinal force F_x with four dimensionless constant coefficients is shown below in Eq. (12).

$$F_x = f(k, F_z) = F_z \cdot D \cdot \sin(C \cdot \arctan[\{Bk - E \cdot (Bk - \arctan(Bk))\}] \qquad (12)$$

The coefficients *B*, *C*, *D*, and *E* correspond to stiffness, shape, peak, and curvature, respectively. These coefficients are known for various road conditions and can be set accordingly. Table 2.2 below shows typical constant coefficient values for different road surface conditions [5].

With load dependent coefficients, the final coefficients of the 'magic formula' will be calculated based on other vehicle attributes for the pure longitudinal slip case. This case assumes that all the motion of the vehicle is in line with the positive X axis in Fig. 2.2. Equation (13) shows the resulting load dependent equation followed by the coefficient equations (Eq. (14) to (22)).

Table 2.2: Constant Coefficient Values for Known Road Conditions

Road Surface	B	C	D	E
Dry Tarmac	10	1.9	1	0.97
Wet Tarmac	12	2.3	0.82	1
Snow	5	2	0.3	1
Ice	4	2	0.1	1

Some of the equations include scaling factors (λ) which provide approximations for non-idealities that would otherwise require intensive calculations to determine. For idealized cases, the scaling factors are set equal to 1. The 'p_i' variables in Eq. (14) to (22) represent curve fitting parameters. These are used to match the behavior of the model to the measured response or tire manufacturer information. Detailed information on the curve fitting process is described in Chapter 4 of [4]. Typical values for these coefficients are shown in Table 2.3 [5].

$$F_{x0} = D_x \cdot \sin(C_x \cdot \arctan[\{B_x k_x - E_x \cdot [B_x k_x - arctan(B_x k_x)]\}]) + S_{Vx} \quad (13)$$

$$B_x = \frac{K_{xk}}{(C_x D_x + \varepsilon_x)} \quad (14)$$

$$C_x = p_{Cx1} \cdot \lambda_{Cx} \quad (15)$$

$$D_x = \mu_x \cdot F_z \quad (16)$$

$$E_x = \left(p_{Ex1} + p_{Ex2} df_z + p_{Ex3} {df_z}^2\right) \cdot \{1 - p_{Ex4} sgn(k_x)\} \cdot \lambda_{Ex} \quad (17)$$

$$k_x = k + S_{Hx} \quad (18)$$

$$\mu_x = \frac{(p_{Dx1} + p_{Ex2} df_z) \cdot \lambda_{\mu x}}{(1 + \frac{\lambda_{\mu V} V_s}{V_o})} \quad (19)$$

$$K_{xk} = F_z (p_{Kx1} + p_{Kx2} df_z) \cdot e^{(p_{Kx3} df_z)} \cdot \lambda_{Kxk} \quad (20)$$

$$S_{Hx} = (p_{Hx1} + p_{Hx2} df_z) \cdot \lambda_{Hx} \quad (21)$$

$$S_{Vx} = F_z \cdot (p_{Vx1} + p_{Vx2} df_z) \quad (22)$$

Table 2.3: Typical Values for Load Dependent Coefficients on Dry Surface

Coefficient	Parameters	Default Values
K	$[p_{Kx1}\ \ p_{Kx2}\ \ p_{Kx3}]$	[21.51 -0.163 0.245]
H	$[p_{Hx1}\ \ p_{Hx2}]$	[-0.002 -0.002]
V	$[p_{Vx1}\ \ p_{Vx2}]$	[0 0]
C	$[p_{Cx1}]$	1.685
D	$[p_{Dx1}\ \ p_{Dx2}]$	[1.21 -0.037]
E	$[p_{Ex1}\ \ p_{Ex2}\ \ p_{Ex3}\ \ p_{Ex4}]$	[0.344 0.095 -0.02 0]

2.3 Electric Motors

In EVs and HEVs electric machines are necessary to convert the electrical energy from the battery into mechanical energy and rotational force. Two common types of machines used in EV and HEV applications are induction machines (IM) and permanent magnet synchronous machines (PMSM). Each have their own benefits and drawbacks for this application. Induction machines are low cost and robust, however PMSM are more efficient and are typically more power dense. The downside to PMSM is that they require rare earth metals to create the rotor permanent magnets and fault conditions can cause demagnetization and render the motor inoperable. The fundamental differences in construction between the IM and PMSM result in different circuit models for each.

2.3.1 Induction Machine

Induction machines are named such because the rotor voltage, which produces the rotor current and rotor magnetic field, is induced in the rotor windings rather than being physically connected or provided by permanent magnets. There are two different kinds of rotors that can be used in an IM. Wound rotors are a complete set of 3 phase windings similar to those found in the stator, but this construction requires slip rings that are shorted via brushes. These shorting brushes allow access to the rotor currents for measurement and also for the introduction of external rotor resistances to change motor behavior. Wound rotors have advantages when being used as generators, but are less efficient than cage rotors. Cage rotors are the most common because the rotor conductor shorting rings are included in the rotor construction and does not require external brushes.

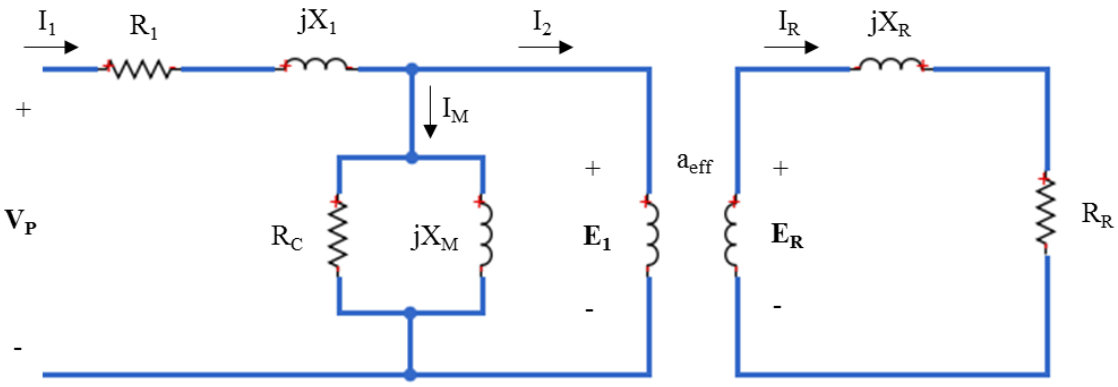

Figure 2.4: Transformer Model of an Induction Machine

Regardless of rotor type, the IM relies on the induction of voltages and currents in its rotor circuit to operate in a way very similar to the operation of transformers. Figure 2.4 shows the transformer model of an induction machine [6]. X_R and R_R represent the rotor impedance. R_1 and X_1 correspond to the stator impedances. X_M and R_C refer to the magnetizing reactance and the transformer core losses such as hysteresis and eddy current losses. The voltages E_1 and E_R correspond to the primary and secondary voltages of an ideal transformer with turns ratio a_{eff}. The voltage E_R that is induced in the rotor in turn produces a current flow in the shorted rotor circuit. The magnitude and frequency of the induced voltage in the rotor is proportional to the slip of the rotor. The IM's slip speed (Eq. (23)) is the difference between the synchronous speed of the motor and the actual frequency. An IM's slip (Eq. (24)) is defined as the relative speed expressed in per units.

$$n_{slip} = n_{sync} - n_{mech} \tag{23}$$

$$s = \frac{n_{sync} - n_{mech}}{n_{sync}} \tag{24}$$

The induced rotor voltage can therefore be expressed in terms of the rotor slip and the induced voltage in the locked rotor condition shown in Eq. (25). Similar relationships exist between the rotor reactance as expressed in Eq. (26). Equation (27) shows the development of the new

equation for the rotor current with the new slip dependent values. Figure 2.5 shows the new rotor circuit model including the frequency effects [6-7].

$$E_R = sE_{R0} \tag{25}$$

$$X_R = sX_{R0} \tag{26}$$

$$I_R = \frac{E_R}{R_R + jX_R} = \frac{E_{R0}}{R_R/s + jX_{R0}} \tag{27}$$

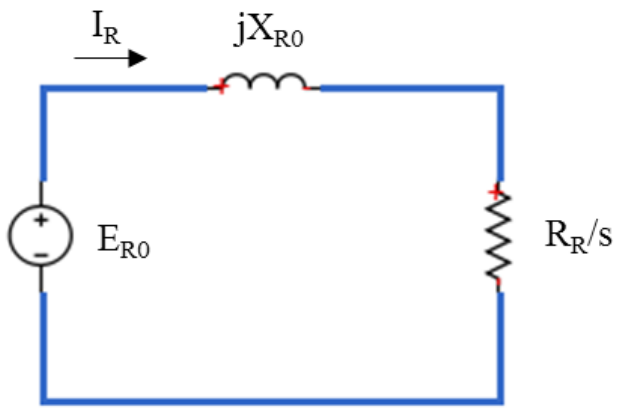

Figure 2.5: Rotor Circuit Model with Frequency Effects

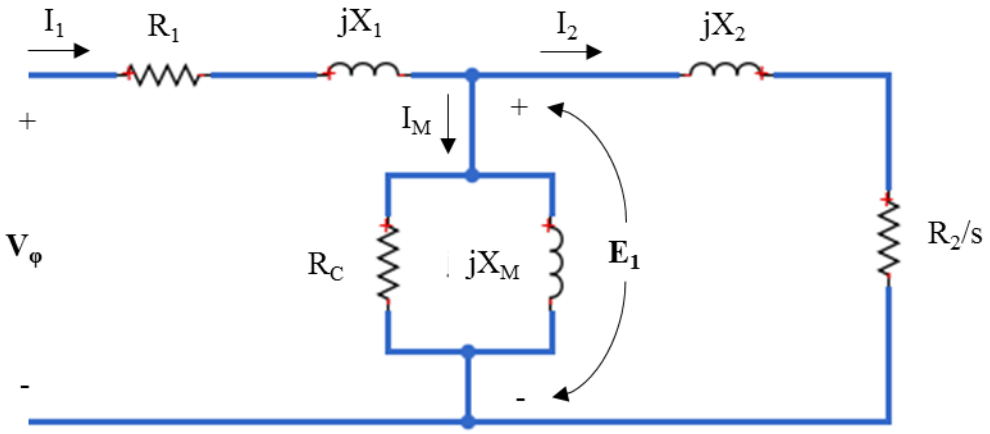

Figure 2.6: Per-Phase IM Equivalent Circuit

The new values for the rotor impedance can now be reflected across the ideal transformer to the stator side of the motor. The voltage on the secondary side is equivalent to the primary voltage divided by the turns ratio. The secondary current is thus multiplied by the turns ratio. Solving for the reflected impedance results in Eq. (28).

$$Z_2 = a_{eff}^2(\frac{R_R}{s} + jX_{R0}) \tag{28}$$

The final per phase equivalent circuit of the IM is shown in Fig. 2.6. Where X_2 and R_2 represent the values of the X_{R0} and R_R after reflection.

The power losses in an induction motor are due to both electrical and mechanical factors. The first source of power losses in motor are the stator resistance (R_1) and stator core losses of the machine (R_c - hysteresis and eddy currents). The power is then transferred across the air gap to the rotor, where the rotor resistance and rotor core losses are. The remaining power is converted to mechanical energy, however some of it is lost due to friction and windage. The rotor core losses are often neglected since they are small in comparison to the stator core losses when operating near synchronous speed. As the electrical frequency increases, the motor speed, friction losses, and windage losses increase. In contrast, as the electrical frequency increases, the core losses decrease. Since these two losses have an inverse relationship, the rotational losses of the induction machine are considered constant with changing speed [6].

2.3.1 Permanent Magnet Synchronous Machine

Synchronous machines can establish rotor magnetic fields in 2 ways, with permanent magnets or by applying a dc current to the rotor windings creating an electromagnet. Regardless of construction, synchronous machines mechanical RPM is synchronized with the electrical frequency and therefore, unlike induction machines, the speed of the motor will be constant regardless of load (i.e. no slip). The fixed rate of rotation is given by Eq. (29), where f_{se} is the electrical frequency and P is the number of poles [6].

$$n_m = \frac{120 f_{se}}{P} \tag{29}$$

Figure 2.7 shows the per phase equivalent circuit for a synchronous machine. The left side of the circuit illustrates the dc exciter used to establish the rotor magnetic field and thus E_A. The phase voltage can be calculated using Eq. (30). The induced voltage E_A can be calculated using Eq. (31). Where N_c is the number of turns in the rotor, ϕ is the airgap flux, and f is the rotational frequency [7].

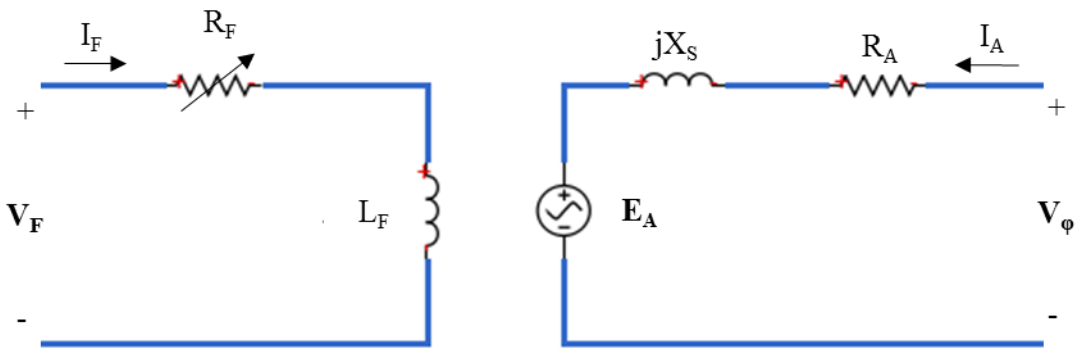

Figure 2.7: Synchronous Machine Equivalent Circuit

$$V_\phi = E_A + jX_s I_A + R_A I_A \quad (30)$$

$$E_A = \sqrt{2}\pi N_c \phi f \quad (31)$$

As previously discussed, PMSMs use interior permanent magnets on their rotors to establish the rotor field rather than inducing the field or using DC exciter windings. The equivalent circuit for a PMSM is shown in Fig. 2.8.

Similar to IMs, PMSMs have power losses associated with the stator resistance, core losses, and windage losses. However the power losses of the rotor are much less than that of the IM because no magnetization current is needed to establish the magnetic field. The power losses in PMSMs can be considered constant with speed since core losses and windage losses have an inverse relationship [6-7].

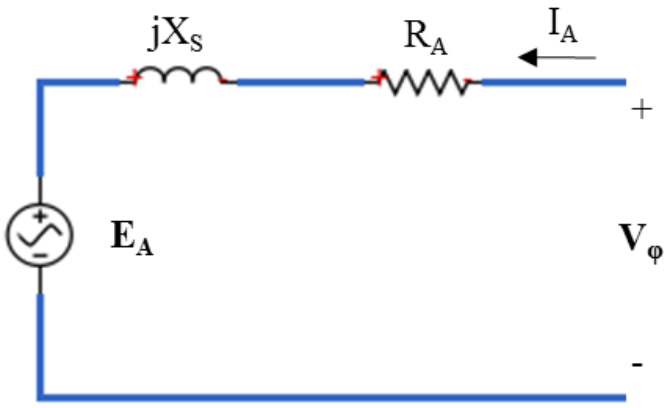

Figure 2.8: PMSM Equivalent Circuit

2.4 Battery

Battery models can be used for state of charge (SOC) calculations, state of health (SOH) estimations, range estimations, and battery management system (BMS) algorithm development. The calculation of remaining discharge energy can be done with an accurate battery model, terminal voltage, and the SOC [8]. Some battery chemistries such as lithium ion, have nonlinear characteristics for low SOC and could lead to miscalculation of remaining driving range. For most EV and HEV applications the BMS will prevent the SOC from falling below the nonlinear threshold, this minimizes the impact on the batteries SOH per charging cycle. Figure 2.9 represents the equivalent circuit model for a battery as developed in [8]. The voltage source represents the open circuit voltage at the current SOC. R_0 refers to the terminal resistance of the battery, and $V_{p,1} – V_{p,n}$ are calculated based on FFT analysis to match the battery impedance and performance characteristics [8, 9, 10]. Equation (32) describes the terminal voltage calculation.

$$V_t = OCV(SOC) - IR_0 - V_{p,1} - \ldots - V_{p,n} \tag{32}$$

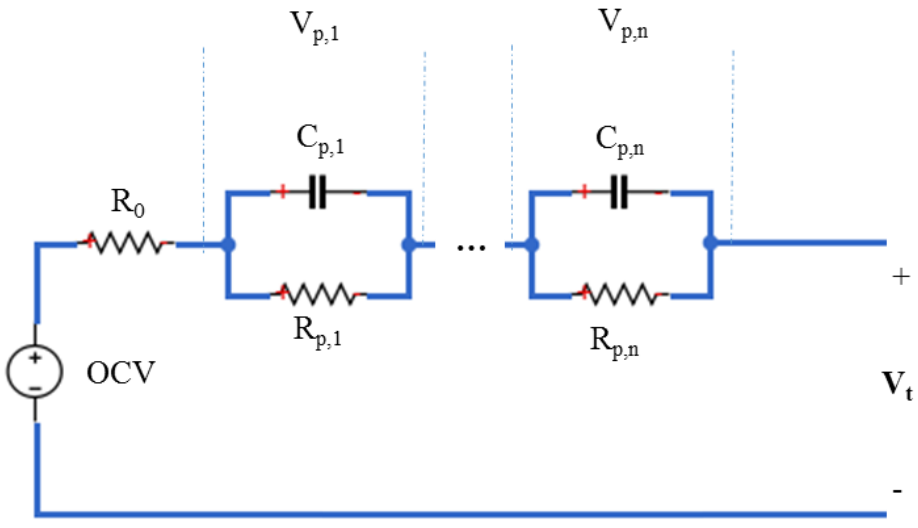

Figure 2.9: Battery Equivalent Circuit

2.5 References

[1] Randall Donn Senger: “Validation of ADVISOR as a Simulation Tool for Series Hybrid Electric Vehicle Using the Virginia Tech FutureCar Lumina” in Fulfillment of Masters Thesis, Virginia Polytechnic Institute and State University, September 1997

[2] K. Wipke, M. Cuddy and S. Burch, "ADVISOR 2.1: A User-Friendly Advanced Powertrain Simulation Using a Combined Backward/Forward Approach", in *IEEE Transactions on Vehicular Technology*, 2017.

[3] "Vehicle Body", Mathworks.com, 2017. [Online]. Available: https://www.mathworks.com/help/physmod/sdl/ref/vehiclebody.html. [Accessed: 14- Oct- 2017].

[4] Pacejka, H. B. Tire and Vehicle Dynamics, Society of Automotive Engineers and Butterworth-Heinemann, Oxford, 2002, chapters 1,4,7, and 8

[5] "Tire-Road Interaction (Magic Formula)", Mathworks.com, 2017. [Online]. Available: https://www.mathworks.com/help/physmod/sdl/ref/tireroadinteractionmagicformula.html. [Accessed: 18- Oct- 2017].

[6] S. Chapman, Electric machinery fundamentals. New York, N.Y.: McGraw-Hill, 2012, pp. Chapters 5 and 6.

[7] N. Mohan, T. Undeland and W. Robbins, Power electronics, 3rd ed. New York, N.Y: John Wiley & Sons, Inc, 2003, pp. Ch 14-15.

[8] G. Liu, L. Lu, H. Fu and J. Hua, "A comparative study of equivalent circuit models and enhanced equivalent circuit models of lithium-ion batteries with different model structures", in *ITEC Asia Pacific* 2014, 2014.

[9] J. Wehbe and N. Karami, "Battery Equivalent Circuits and Brief Summary of Components Value Determination of Lithium Ion", in *Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE),* 2015 Third International Conference, Beirut, Lebanon, 2017.

[10] R. Jackey, A Simple, Effective Lead-Acid Battery Modeling Process for Electrical System Component Selection. MathWorks, Inc., 2007.

CHAPTER 3
VEHICLE SIMULATION

3.1 Overview

The HEV model was developed in Matlab/SIMULINK™ environment and incorporates the Simscape Driveline™ block set. Figure 3.1 shows the top level of the simulation block diagram. This program structure and communication blocks on the inputs and outputs are required for real-time simulation on OPAL-RT. The "SM_vehicle" block will be run on one of the simulation cores inside the simulator, and the "SC_scopes" block will be displayed in the user interface to allow the monitoring of the real-time signals. Figure 3.2 shows the top level of the simulation inside the SM_vehicle block. The major components of the EV powertrain, (such as the battery, motors, transmission, etc.) can be found here as well as the communication setup and the signal limiting logic. Chapter 1 discussed the differences between forward/backward facing simulations and the combination of both used in the creation of ADVISOR. In summary, the backward facing approach answers the question "assuming the vehicle followed the required trace, how must each component perform?" With this approach, no driver behavior model is needed. The forward facing approach better suits our needs: it is ideal for hardware development and detailed control system simulation since it determines what the vehicle can accomplish with the defined components, rather than what component specifications are needed to accomplish a demand. Though the model is generalized for any EV or HEV car, the 2014 Toyota Prius was used as the test vehicle, and the gear ratios and vehicle parameters shown later reflect this vehicle. This chapter will be a block-by-block walkthrough of the vehicle model, and will discuss the mathematical models used and assumptions made in order to simplify the model for real-time simulation [1].

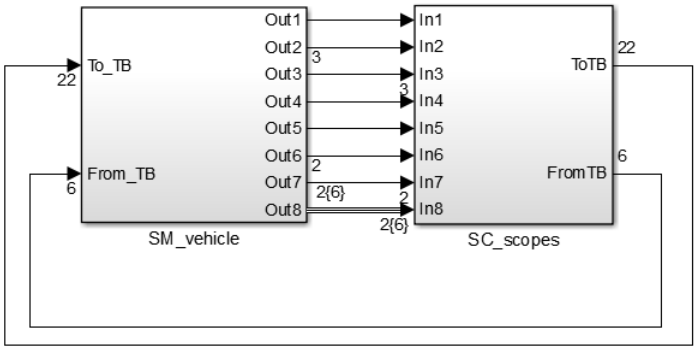

Figure 3.1: Top Level of the Vehicle Model Diagram

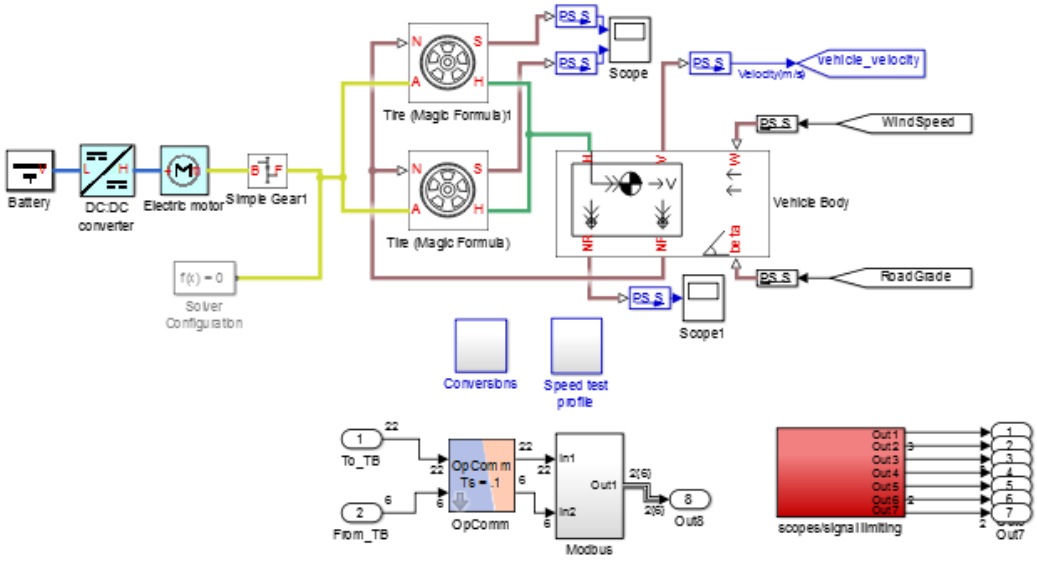

Figure 3.2: Top Level of the Vehicle Simulation

3.2 Battery

The battery block contains the equivalent circuit for a simple battery. The voltage levels, battery impedances, management strategies, and internal chemistry can vary widely from vehicle to vehicle, especially across different vehicle manufacturers. All of those factors affect the structure of the battery model. Figure 3.3 shows the simplistic battery model included in the simulation. The focus of the development work has largely been in the performance of the vehicle power electronics prototypes in a range of vehicle types. Range estimations can be calculated based on production energy storage systems, power electronic efficiency, and power consumption of the dynamometer testbed. The NCREPT facility also does not currently have all of the protective enclosures necessary to test large energy storage systems. For this reason, the generic battery model was not expanded to model a certain chemistry/construction. The battery power, charge and losses are calculated in the subsystem shown in Fig. 3.4.

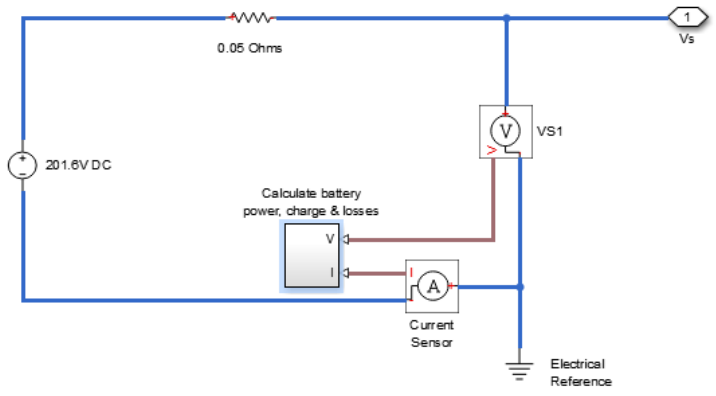

Figure 3.3: Battery Model

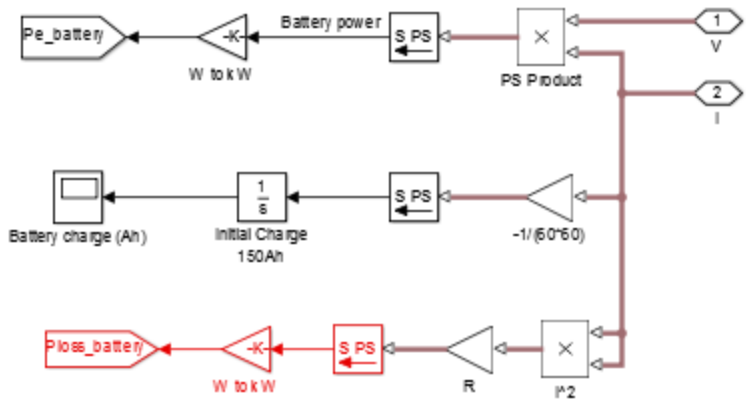

Figure 3.4: Battery, Power, Charge and Loss Calculations

3.3 Electric Motor Model

The electric motor is located downstream from the battery and the dc/dc converter. The dc/dc converter block can be replaced by the user's Simulink design if performance validation of the converter software model is necessary. However, implementing a more detailed dc/dc converter model does not change the torque required to propel the vehicle at a given speed, so the idealized dc/dc converter is sufficient for most inverter testing applications.

Figure 3.5 shows the first level of the electric motor block. The model does not include a Matlab/Simulink rotating machine model, because these are very computationally expensive and would increase the model complexity beyond what a single core OPAL-RT simulator can run in real-time. The reference motor rpm signal is the reference rpm for the motor calculated from the driving schedule. This signal is determined in the 'Speed test profile' block and will be discussed later. The speed controller block houses the proportional integral (PI) controller, which generates a torque signal that is either positive or negative until the desired speed is reached. The PI controller is shown in Fig. 3.6. The controller with the driving schedule is intended to model the

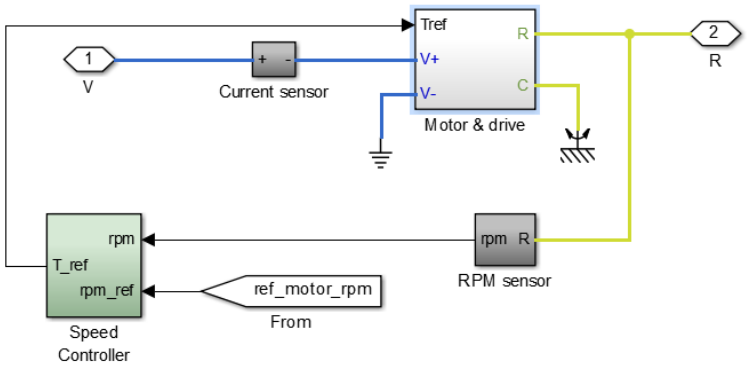

Figure 3.5: First Level of the Electric Motor Block

driver behavior. Equation (1) shows the Laplace domain equation for the output of a PI controller.

$$u(t) = k_p + \frac{k_i}{s} \tag{33}$$

The proportional and integral gains (k_p and k_i, respectively) can be tuned to match the desired characteristics. For linear control systems, the "plant" can be reduced to a transfer function and k_p and k_i can be calculated to fit in the design criterion for overshoot and settling time. However, the electric motor and vehicle body system is nonlinear and is difficult to tune. One common method for tuning is to linearize the system about a single operating point in a region of operation and solve for the controller gains. This often results in a suboptimal solution that may become unstable in different regions of operation. In order to more closely mimic the actual vehicle, the validated torque reference waveforms from NREL's ADVISOR were compared to the reference torque generated in the model. Figure 3.7 illustrates the steps in

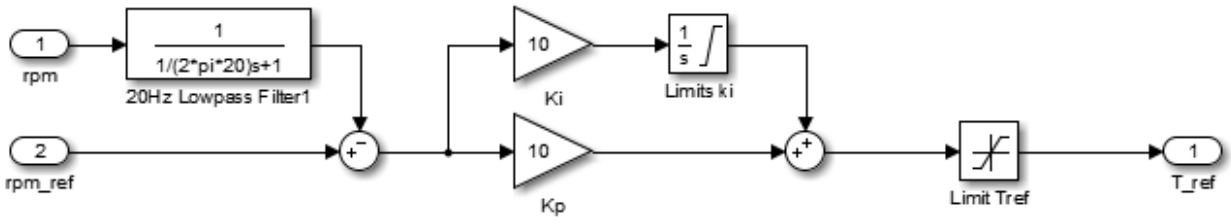

Figure 3.6: PI Speed Controller

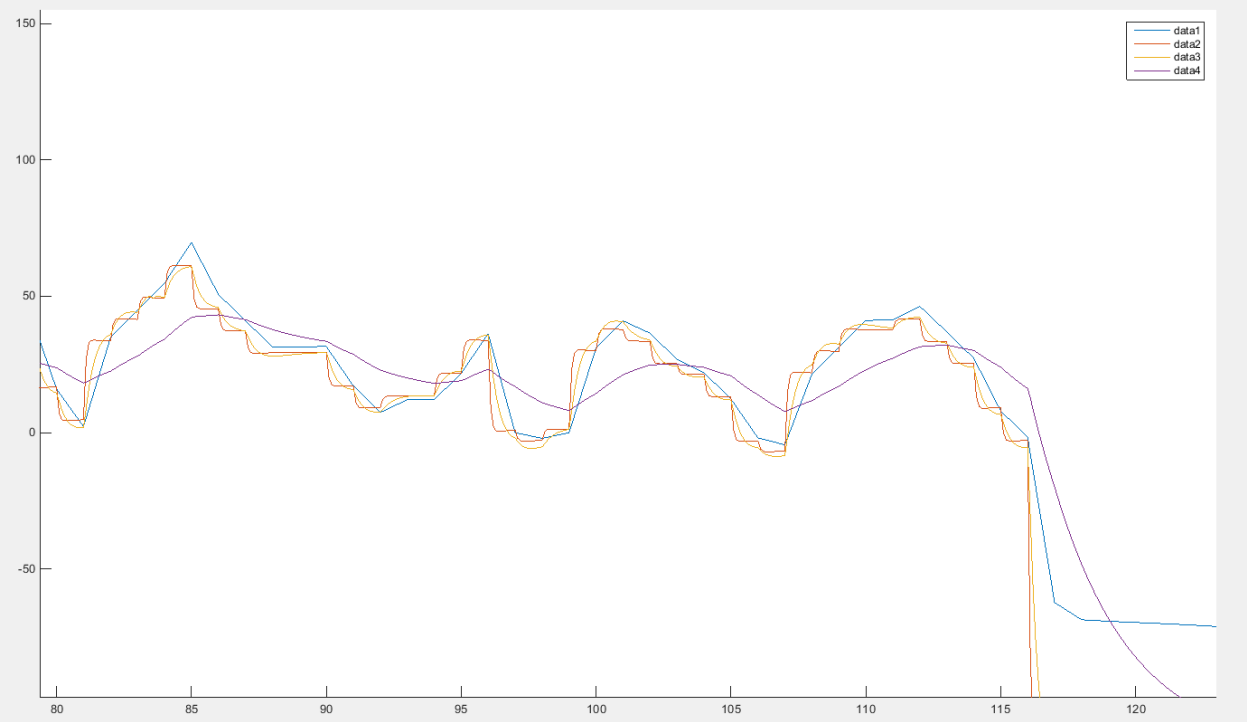

Figure 3.7: Tuning Process - Data1-ADVISOR Data2- Kp=Ki=60 Data3-Kp=Ki=10 Data4- No PI Controller

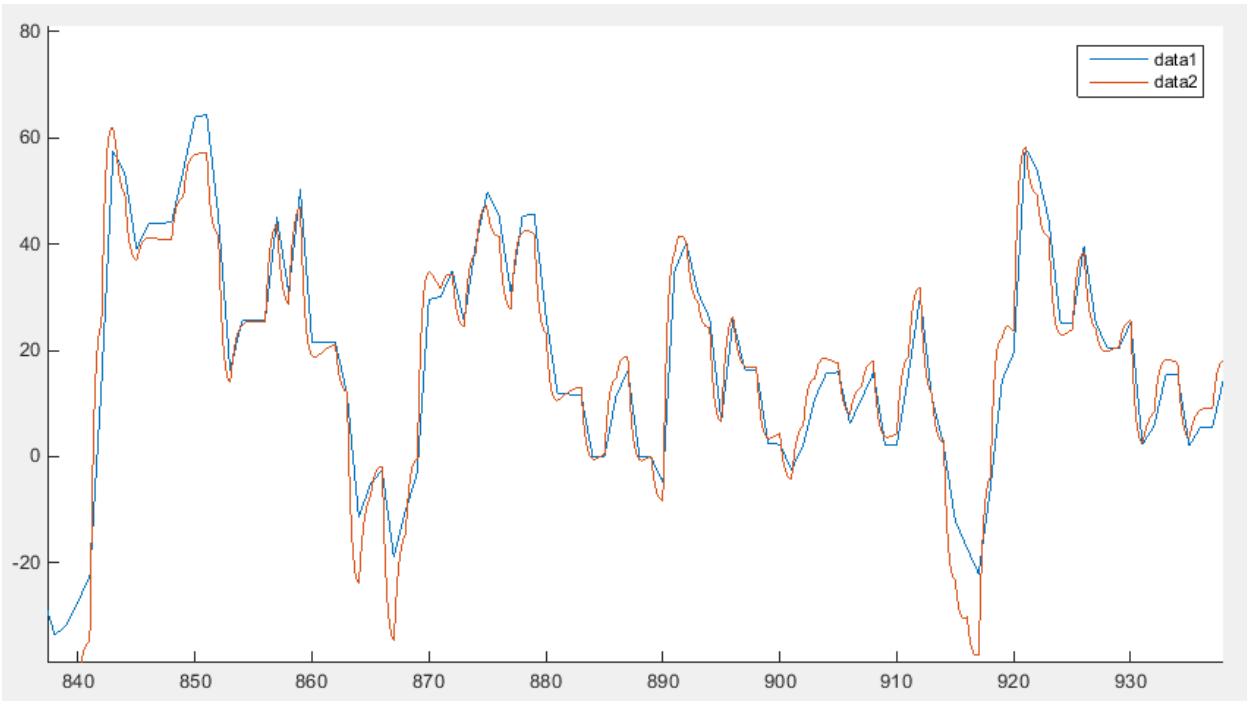

Figure 3.8: ADVISOR and Tuned PI - Blue- ADVISOR // Orange-Tuned PI

the tuning process and based on the results, the yellow waveform corresponding to $k_p = k_i = 10$ was the best fit. ADVISOR has a time step of 1 second, which is 10 times slower than this simulation, meaning that more of the transient behavior between data points is represented in the waveforms (shown in Fig. 3.8 results from ADVISOR and the model developed here).

The reference torque that is generated from the PI controller is then sent to the 'motor and drive' block pictured in Fig. 3.5. This block also uses the voltage level from the dc/dc converter as an input. Figure 3.9 shows the contents of this block where the reference torque signal is converted to a Simulink physical signal and is then fed into a "torque lag" subsystem (Fig. 3.10). Control system theory can be used to determine the transfer function and response of the system.

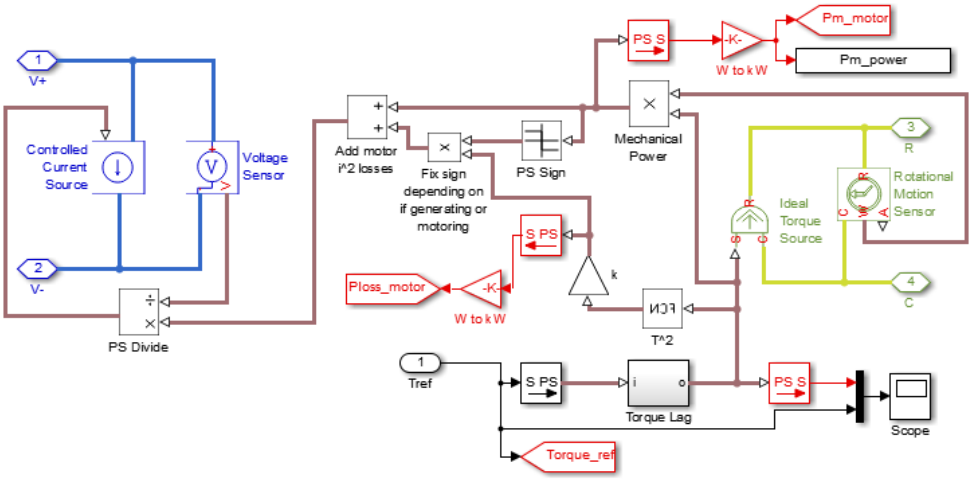

Figure 3.9: Motor and Drive Block

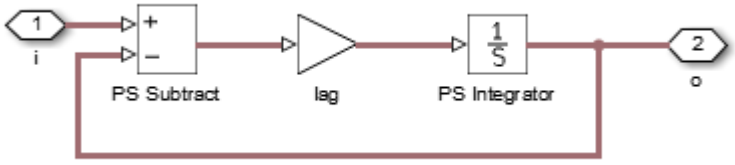

Figure 3.10: Torque Lag block

The open loop gain of the function is represented by Eq. (2) shown below, where *E(s)* is the error of the system in the Laplace domain, *k* is the value of the gain block (1/*Tm*), *G(s)* is the integrator term in the Laplace domain, and *I(s)* and *O(s)* are the input and output of the system respectively.

$$O(s) = E(s) \times k \times G(s) \tag{34}$$

$$E(s) = I(s) - O(s) \tag{35}$$

Inserting Eq. (2) into Eq. (3) yields:

$$E(s) = I(s) - E(s) \times k \times G(s) \tag{36}$$

Solving for *E(s)* yields:

$$E(s) = \frac{I(s)}{1 + kG(s)} \tag{37}$$

Plugging *E(s)* in Eq. (5) into Eq. (2) results in:

$$O(s) = \frac{kG(s)}{1 + kG(s)} I(s) \tag{38}$$

This leads to the transfer function of the system in the Laplace domain:

$$T(s) = \frac{O(s)}{I(s)} = \frac{kG(s)}{1 + kG(s)} \tag{39}$$

Converting Eq. (7) back into the time domain yields:

$$T(s) = \frac{k \times {}^{1}/_{s}}{1 + k \times {}^{1}/_{s}} = \frac{k}{s + k} \rightarrow \frac{1}{T_m} e^{-{}^{1}/_{Tm}} \tag{40}$$

$$\text{Where } T_m = \frac{R_a J_m}{k_T k_E} \tag{41}$$

T_m refers to the mechanical time constant of the motor. This is calculated with the armature resistance (R_a), the inertia of the motors rotor (J_m), the torque constant of the motor (k_T), and the voltage constant of the motor (k_E) [2]. The value of T_m can be set in the simulation parameters by changing the value of the 't_torque' variable. When this torque lag block is introduced into the

system, the output will lag the input of the system. Figure 3.11 shows the input and the output of the torque lag system on the same graph. This output signal of the torque lag block is then used to convert the physical signal to a 'mechanical rotational domain' Simscape variable by the 'ideal torque source' block. This conversion is ideal in the sense that it does not account for inertia, friction, delays, or energy consumption in the motor. The time delays associated with the non-idealities are taken care of in the torque lag block discussed previously. The equation for electromagnetic torque in an electric motor (shown in Eq. (10)) developed in Chapter 12 of Prof. Mohan's "Power Electronics" book is thus reduced to Eq. (11). The J_x and B_x terms refer to the inertia of the system and friction/dampening losses, respectively.

$$T_{em} = \frac{\dot{\omega}_L}{a}[J_m + a^2 J_L] + aT_{WL} + \frac{\omega_L}{a}(B_m + a^2 B_L) \quad (42)$$

$$T_{em} = aT_{WL} \quad (43)$$

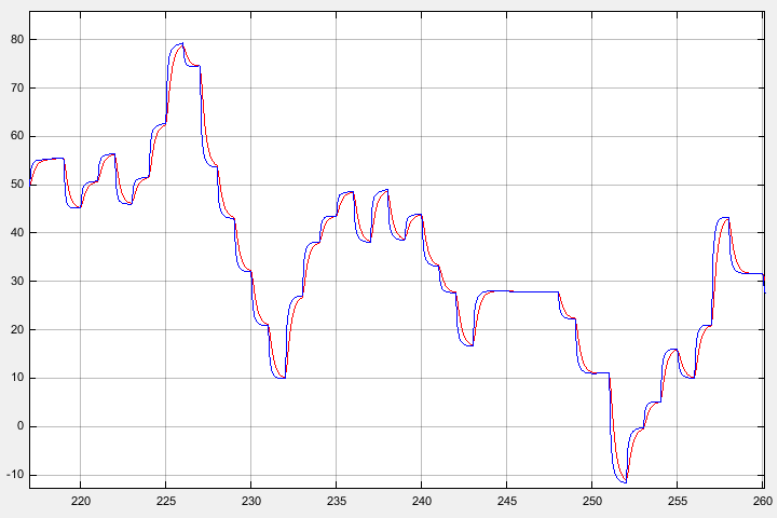

Figure 3.11: I/O of Torque Lag Block - Blue: Input Red: Output

The mechanical rotational domain variable produced by the ideal torque source (yellow traces in Fig. 3.9) is measured with a rotational motion sensor before exiting the "Motor and Drive" block. The angular speed is then used with the torque signal to calculate the output power produced by the motor. The derivation of this equation is shown below in Eqs. (12)-(14).

$$power = \frac{force \times linear\ distance}{time} \tag{44}$$

$$= \frac{\left(\frac{torque}{r}\right) \times (r \times angular\ speed \times t)}{t} \tag{45}$$

$$power = torque \times angular\ speed \tag{46}$$

The power losses in the motor are also calculated here using Eq. (15) [2]. The torque signal is squared with a math function block, then multiplied by a motor loss constant k_2 (Watts/Nm^2).

$$P_R = k_2 T_{em,rms}^2 \tag{47}$$

The motor power produced at the output and the power losses are then used to determine the total power delivered at the terminals of the motor. The mechanical power at the shaft of the motor will either be positive or negative depending on the operating quadrant of the electric motor. The total power is then used to calculate the RMS per phase current draw of the motor. This current value is the control variable for a controlled current source. This current setup mimics a per phase equivalent of a PMSM. If a full three-phase inverter is to be tested with this simulation, this method of current controlled loads can be extended to three phases.

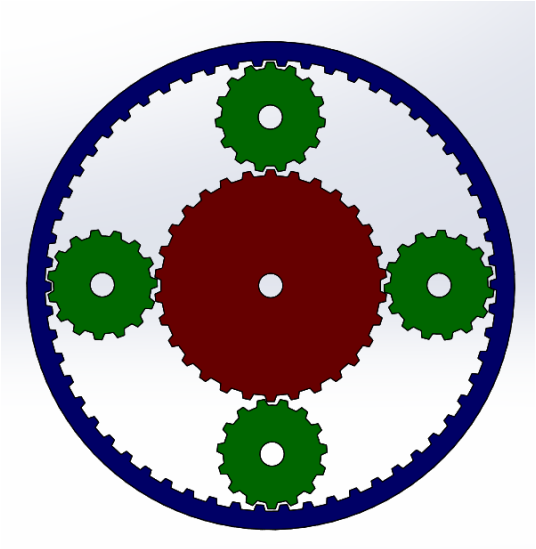

Figure 3.12: Planetary Gear Set

3.4 Power Split Device - Planetary Gearbox

At the heart of most EV or HEVs lies a transmission or power transfer device that changes the often high rpm of the motor shafts to a suitable RPM and torque to propel the vehicle forward. HEVs require a power split device (PSD) in order to utilize or deliver power to/from multiple sources (PMSM, generator, and ICE). Each generation of the Toyota Prius incorporates one or multiple planetary gear sets. The generation 3, which is implemented in the 2009 -2015 models, incorporates a single planetary gear set for power splitting in its electronic continuously variable transmission (eCVT). Figure 3.12 shows a typical planetary gear set. It includes one ring gear (blue), planet gears (green) with planet carrier (which connects all the planet gears), and a sun gear (red) [3]. The ring gear is mechanically connected to the propulsion motor and the wheels. This means that the gear ratio between the electric motor and the speed of the vehicle is fixed and constant. The planet gear carrier is connected to the internal combustion engine. The planet gears spin freely at their connection points to the carrier which means the ICE can provide power to the generator (connected to the sun gear) even when the PMSM is not rotating. The sun gear is connected to the vehicles generator and will change direction of rotation

based on the operating region of the PMSM and ICE [3]. The PMSM can provide power to the wheels on its own or the ICE and PMSM can both provide power to the wheels at the same time. An interactive figure in [3] provides further insight into the PSD operation.

The overall gear ratio of a planetary (epicyclical) gear set can be determined by representing the sun – planet interaction and the planet – ring interactions individually. The sun-planet interaction has a kinematic (Eq. (16)) and geometric (Eq. (17)) constraints associated with the three connected axes.

$$r_C\omega_C = r_S\omega_S + r_P\omega_P \tag{48}$$

$$r_C = r_P + r_S \tag{49}$$

Where r_s, r_p, r_c correspond to the radii of the sun gear, planet gear, and planet carrier respectively. The ω variables represent the angular speeds of the gears and carrier. The planet-sun gear ratio (g_{ps}) is shown below:

$$g_{PS} = {r_P}/{r_S} = {N_p}/{N_S} \tag{50}$$

where N is the number teeth on each gear. In terms of this gear ratio the kinematic constraint for the system shown in Eq. (16) then becomes:

$$\omega_S = -g_{PS}\omega_P + (1 + g_{PS})\omega_C \tag{51}$$

The positive or negative representation of the gear ratio (g_{ps}) refers only to the direction of rotation. The kinematic constraint in the form shown in Eq. (19) and the torque transfer calculation shown in Eq. (20) fully describe the mechanical interaction between the sun and planet gears [4,5].

$$g_{PS}\tau_S + \tau_P - \tau_{loss} = 0 \tag{52}$$

The interaction between the ring gear and the planet gears/carrier has similar kinematic and geometric constraints to the ring-planet gears discussed previously. Equations (20) to (24) show the kinematic constraint, gear ratio, final kinematic constraint and the torque transfer calculation, respectively [4, 6].

$$r_R\omega_R = r_C\omega_C + r_P\omega_P \tag{53}$$

$$g_{RP} = {}^{r_R}/_{r_P} = {}^{N_R}/_{N_P} \tag{54}$$

$$g_{RP}\omega_R = \omega_P + (g_{RP} - 1)\omega_C \tag{55}$$

$$g_{RP}\tau_P + \tau_R - \tau_{loss} = 0 \tag{56}$$

The ring gear of the planetary gear set in the Toyota Prius is considered the output of the PSD, and is connected to the wheels through a final drive gear reduction between the RPM of the motor and the axle speed and has a value of 4.17 [7,8]. As discussed previously, the speed of the ICE has no direct mechanical connection to the final drive and therefore no direct impact on the speed of the vehicle. However, its connection to the PSD allows it spin at a rate suitable for generating power independently of the vehicle speed. That being said, the torque supplied to the PSD by the ICE via the planet carrier is split unevenly between the ring gear and the sun gear. Torque is equal to force times distance from the center of rotation. Since the three gear components (sun, planet carrier, and ring) share the same axis of rotation, more of the torque from the ICE will be delivered to the ring gear than the sun gear. The vehicle power control unit (PCU) determines when torque from the ICE is necessary.

In summary, the ICE can provide torque that is split between the propulsion motor and the generator in the PSD but there is no direct correlation between ICE rpm and wheel rpm. For the real-time vehicle model, a simple gear was used to represent the 4.17 ratio between the propulsion electric motor and the axle rotations (shown in Fig. 3.2 between the motor and tire

models). The torque contribution of the ICE is non consistent and the precise algorithms used in the PCU to determine the amount of ICE input are not made public knowledge by vehicle manufacturers. The hardware associated with a PSD is fairly easy for a competitor to determine by taking apart the vehicle, but it is more difficult to determine the control software. Much of the efficiency gains that have been made in HEVs come from more optimal control of the PCU rather than new mechanical designs [8]. Since the precise control strategy for the Toyota Prius could not be determined, the ICE torque contribution is defined as an average percentage of overall torque provided by the engine for a given vehicle. The torque contribution of the ICE is taken into account in the signal limiting blocks that will be discussed later.

3.5 Tire Simulation Model

Section 2.2 discussed the different mathematical models of the interaction between the tire and road and develops some benefits and drawbacks to each method. Simscape Driveline provides component libraries for modeling and simulation rotational and translational mechanical systems [9], including the tire-road interaction represented by Hans Pacejka's magic formula [10]. The overall tire model simulates both transient and steady state behavior while modeling the longitudinal forces on the tire. Figure 3.13 shows the component diagram for the

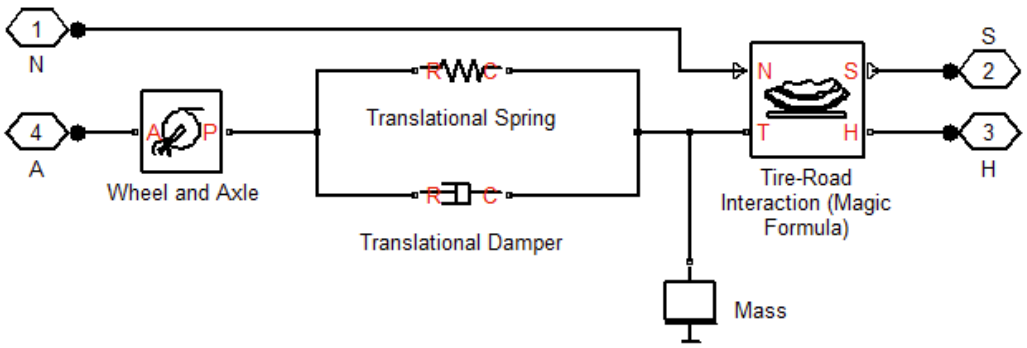

Figure 3.13: Tire Component Diagram

"Tire (magic formula)" block available in Driveline and used in the model. To make the model suitable for HIL/real-time simulation, the compliance portion of the model is turned off. Compliance implies that a time lag exists between the tire response and the forces that act on it. Simulation speed can be significantly reduced if compliance is used. When tire compliance is disabled the translational spring, translational damper and mass inertia in Fig. 3.13 are omitted and the port P of Wheel and Axle is directly connected to port T [11]. The magic formula is modeled with constant coefficients for tire characterization. Those coefficients represent stiffness, shape, peak, and curvature. Table 2.2 in Section 2.2 listed typical coefficient values representing different road surfaces.

3.6 Vehicle Body Model

This block, shown in Fig. 3.2, models a two-axle vehicle taking into account longitudinal dynamics, motion and adjustable mass, geometry, and the drag properties of the user specified vehicle. The free body diagram for the vehicle in motion, the formula derivations, and variable definitions were discussed in Section 2.1. The equations for the normal force on both the front and rear wheels are shown below in Eq. (25) and (26), respectively.

$$F_{zf} = \frac{-h\left(F_d + mgsin(\beta) + m\dot{V}_x\right) + b \cdot mgcos(\beta)}{n(a+b)} \tag{57}$$

$$F_{zr} = \frac{-h\left(F_d + mgsin(\beta) + m\dot{V}_x\right) + a \cdot mgcos(\beta)}{n(a+b)} \tag{58}$$

The drive train characteristics of the vehicle (front, rear, or all-wheel drive) will determine which force port the tire models are connected to. The physical attributes of the vehicle are also represented in these equations; mass (m), vehicle height (h), distance of axles from the center of

gravity (a, b), number of wheels on each axle (n), frontal surface area (A), and drag coefficient (C_d); all contribute to the vehicle response to a speed input [12]. Weather and road conditions are also considered here as wind speed and road grade are user input variables that can be changed in real-time.

3.7 Scopes and Signal Limiting

The scopes and signal limiting block shown in Fig. 3.14 is intended to bring all of the signals that are desirable to observe in real-time or that need to be sent to the LabVIEW™ interface together in one place. In addition, the ramp rate limiter and torque calculations are done at this time. The torque signal generated previously in the "Motor and Drive" block can be thought of as the total torque required to accelerate the user defined vehicle to the commanded speed. Since the speed of the ICE is not directly associated with the speed of the vehicle, the torque contribution of the ICE can be taken into account further downstream rather than at the gearbox. In addition to simplifying the vehicle model, it simplifies the user implementation of a more complex power control algorithm by eliminating the need to use Matlab's "physical signals". The Torque Calculations block in Fig. 3.14 uses the motor RPM and the power generated by the motor to calculate the torque as done previously in Eqs. (12) to (14). Figure 3.15 shows the inside of this Torque Calculations block where the ICE torque contribution is removed from the overall torque produced and the ramp rate limits are determined.

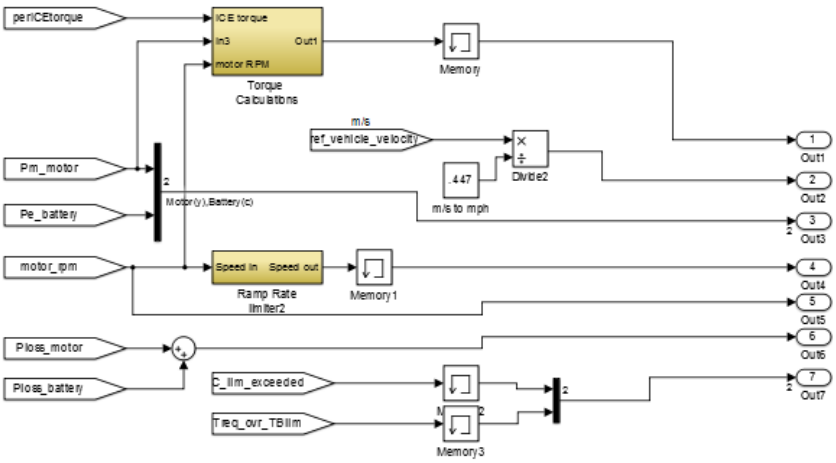

Figure 3.14: Top Level of Scopes and Signal Limiting Block

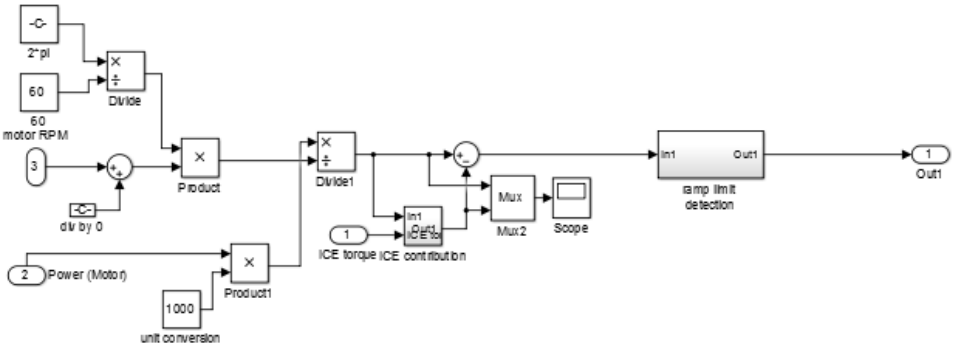

Figure 3.15: Torque Calculations Block

3.7.1 Ramp Limit Detection

Ramp rate limits of the torque and speed curves are imposed on the system due to physical limitations of the testbed equipment. Equation (27) describes the electromagnetic torque required from the motor.

$$T_{em} = \frac{\dot{\omega}_L}{a}[J_m + a^2 J_L] + aT_{WL} + \frac{\omega_L}{a}(B_m + a^2 B_L) \tag{59}$$

where ω_L is the angular velocity of the load, a is the coupling ratio, T_{WL} is the working torque of the load, J_m and J_L are the motor and load inertia, respectively, and B_m and B_L are the dampening associated with the motor and load. Because the motor shaft has a mass, and therefore an inertia,

it cannot change the speed instantaneously and from Eq. (28) the torque required for speed changes (acceleration torque) is directly proportional to the acceleration. J_{eq}, the equivalent inertia of the motor and generator shafts, and a=1 since there is no gearbox between the two shafts; $J_{eq}= J_m + J_{gen}$ [2].

$$T_{acc} = \frac{d\omega_L}{dt}[J_{eq}] \tag{60}$$

In PMSMs and IMs alike the electromagnetic torque produced is dependent upon the phase current of the machine. Thus, a quick change in speed will require a large acceleration torque and will cause a proportional increase in the phase current in addition to the load torque calculated by the simulation and its current requirement. This block uses the measured inertia of the motor/generator set to calculate the acceleration torque required at each time step to ensure that the current required does not exceed the limits for the physical equipment or the user defined maximum current. This prevents the test equipment and unit under test from being damaged by a driving schedule with an unknown power requirement.

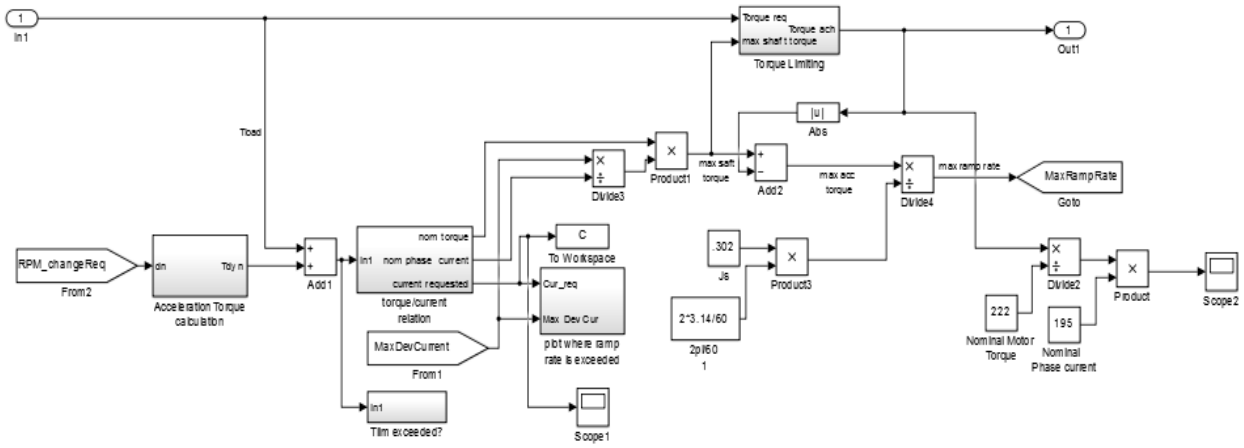

Figure 3.16: Ramp Limit Detection block

Figure 3.16 shows the ramp limit detection block which calculates the total torque in the system and compares it to the user and system limits. The acceleration torque calculation block uses the change in RPM between the previous and current time steps, Eq. (28), and the load calculated load torque to determine the total electromagnetic torque that will be required. The relationship between the electromagnetic torque and the phase current is shown in Eq. (29) and is implemented in the torque/current relation block to calculate the current required to meet the given speed command [2]. If this current exceeds the testbed limit or the user set current limit, the maximum safe value for the torque will be used.

$$\frac{T_{em}}{T_{rated}} = \frac{I_{\phi}}{I_{rated}} \tag{29}$$

The "Tlim Exceeded?" block monitors the total torque requested by the simulation and if it exceeds the testbed limit of 200 Nm the user will be notified by a flashing light in the user interface and the torque will be limited to the 200 Nm value in software. The "plot where ramp rate is exceeded" block monitors the current requested and will notify the user via a light in the user interface when it exceeds the user defined limit. It also saves this information to the Matlab workspace so that where in the driving schedule the current limit is exceeded can be plotted. An extreme example (high current requirement, low user-set current limit) is shown in Fig. 3.17 where even relatively small accelerations of the UDDS driving schedule exceed the limit. However, the ramp rate limiter block discussed later will alter the speed commands in the red areas to limit the current. The user-defined maximum device current is then used to calculate the maximum shaft torque that will be used in the torque limiting subsystem shown in Fig. 3.16.

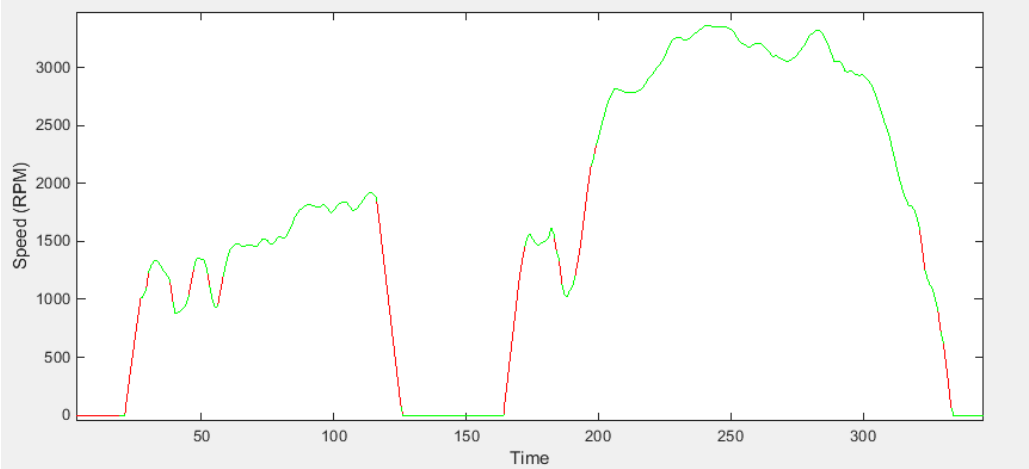

Figure 3.17: Driving Schedule with Areas Exceeding the Current Limit Overlaid

The torque limiting block uses the logic circuit shown in Fig. 3.18 to determine if the torque required is greater than the maximum shaft torque calculated from the current limits and selects the correct command to be sent to the testbed. If the torque required exceeds the limit, 95% of the calculated maximum will be used. The max ramp rate of the speed is calculated after the torque limiting (shown in Fig. 3.16), the torque achieved cannot exceed 95% of the maximum value, otherwise the max ramp rate of the speed would be near zero. A near zero speed ramp will cause the error between the driving schedule and the achieved speed to be quite large. The difference in the achieved torque and the calculated maximum torque is the maximum allowable acceleration torque. Eq. (28) and the system inertia is again used here to calculate the max ramp rate of the speed.

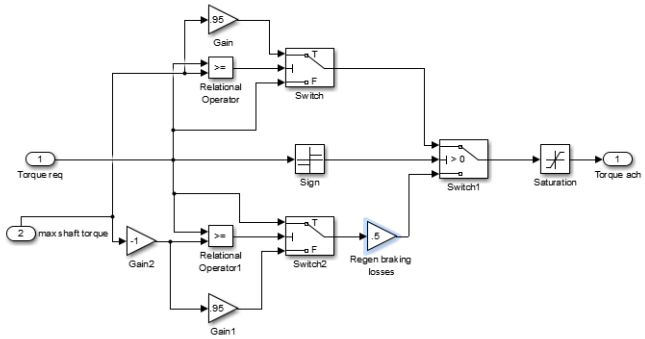

Figure 3.18: Torque Limiting Block

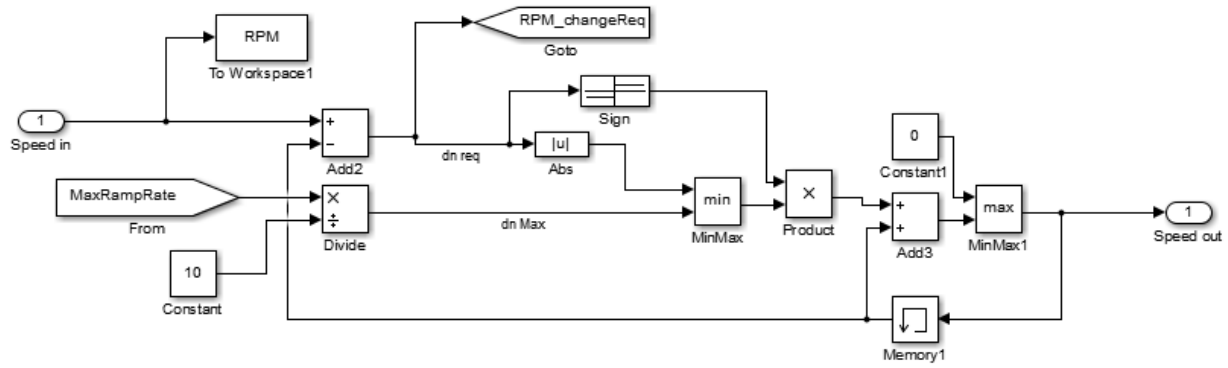

Figure 3.19: Ramp Rate Limiter Block

3.7.2 Ramp Rate Limiter

In this block, shown in Fig. 3.19, the maximum allowable ramp rate (change in RPM per time step) that was calculated in 3.7.1 is compared to the requested time step from the driving cycle. That ramp rate is first divided by 10 to convert the value to RPM per 0.1 second, which coincides with the step time. The minimum of the two ramp rates is used to calculate speed command for the current time step. The output speed is fed back through a memory block which holds the value from the previous time step.

3.8 Simulation I/O

The driver profile for this simulation is a structured driving schedule consisting of speed commands and time stamps. The EPA has created some driving schedules for vehicle emission testing that represent typical driver behavior in certain environments. The Urban Dynamometer Driving Schedule (UDDS) would be similar to city driving and the Highway Fuel Economy test (HWFET) is representative of high-speed highway driving. All of the EPA driving schedules are included in the simulation and can be selected by the user before testing as described in the Testbed User Manual in Appendix A. Custom schedules can also be made to represent any necessary driving scenario.

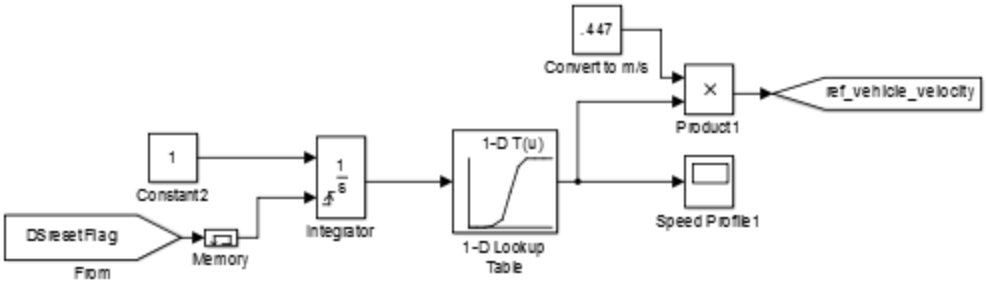

Figure 3.20: Speed Test Profile block

The driving schedules are .csv files and are read into the Simulink program in the initialization stage and saved as a workspace variable. “Speed test profile” block in the top level of the simulation, and shown in Fig. 3.20, uses a lookup table to index the appropriate speed command. The lookup table block uses a linear interpolation method to determine the speed commands between 1-second intervals of the driving schedule. To iterate through the lookup table, a constant value of 1 is integrated continuously. The driving schedule can therefore be reset by utilizing the external reset of the integrator which will set the output value to zero when activated.

Communication with the testbed equipment and the LabVIEW data acquisition software is accomplished using serial communication. MODBUS TCP protocol uses MODBUS messaging in an internet environment by attaching TCP/IP addresses to the PLC’s, I/O modules, and other equipment on the network. Two-way communication between the testbed hardware and the OPAL-RT simulator is used to communicate the speed and torque commands from the simulation to the hardware and to relay the user’s settings and measurement feedback to the real-time interface. Signals to and from the I/O on the OPAL-RT must go through an “OpComm” block as seen in the Modbus interface on the left in Fig. 3.21. The right image of Fig. 3.21 shows inside the “Modbus” block where the initial conditions for the simulation parameters are set and

the OPAL-RT Modbus TCP slave device is set up. A Modbus configuration file must be made to reflect the communication settings for the Modbus Master device on the testbed hardware as described in [13].

For signals in the simulation to be monitored in real-time or to be available for serial communication, the signals must be located in the "SC_" block at the top level of the diagram. Figure 3.22 shows the "SC_scopes" block where the signals are monitored. Measurement feedback from the testbed can be seen in the registers at the bottom of the window. The register Modbus addresses and data names are shown in Table 3.1 below, in the order that they appear in the registers in Fig. 3.22. Both the commanded motor shaft speed and the limited speed can be seen in the "Speed Profile1" scope. The calculated power, torque and vehicle speed in mph waveforms are also available for real-time viewing.

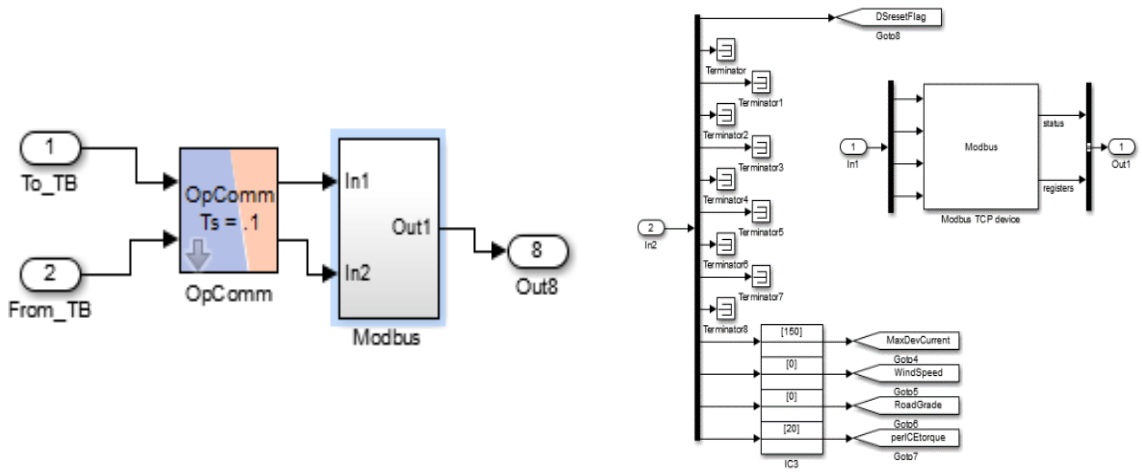

Figure 3.21: Modbus Interface

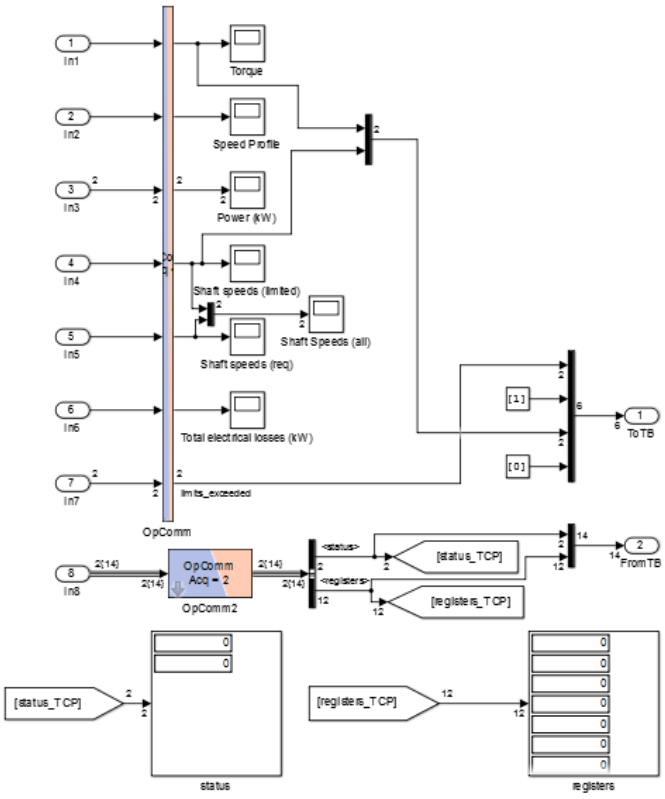

Figure 3.22: SC_scopes Block

Table 3.1: Measurement Feedback Registers in SC_scopes

Address	LabVIEW Data Name	Description
20	UUT_RMS_Ia (A)	Phase A RMS current measured by Yokogawa power analyzer
21	UUT_RMS_Ib (A)	Phase B RMS current measured by Yokogawa power analyzer
22	UUT_RMS_Ic (A)	Phase C RMS current measured by Yokogawa power analyzer
23	UUT_RMS_Vab (V)	Phase A RMS voltage measured by Yokogawa power analyzer
24	UUT_RMS_Vbc (V)	Phase B RMS current measured by Yokogawa power analyzer
25	UUT_RMS_Vcd (V)	Phase C RMS current measured by Yokogawa power analyzer
26	Dyno_Speed (rpm)	Measured speed at the dynamometer shaft
27	Dyno_Torque (N*m)	Measured torque at the dynamometer shaft
28	Max_Dev_Current	User defined max RMS current
29	Wind_Speed	Headwind speed
30	Road_Grade	Road Grade
31	P_ICE_Torque	Percent of total torque at shaft supplied by ICE

3.9 Conclusions

This chapter has discussed the HEV simulation model and how the OPAL-RT real-time simulator interfaces with the LabVIEW testbed control and data acquisition system. The powertrain component mathematical models explained in Chapter 2 were combined and adapted for real time simulation using the Simscape Driveline block set and RT-Lab communication blocks. Chapter 4 will present an overview of the dynamometer testbed system as a whole, then discuss the DTC strategy implemented on the load motor drive, the LabVIEW control interface, OPAL-RT simulator, and Testbed-to-unit under test communication protocols.

3.10 References

[1] K. Wipke, M. Cuddy and S. Burch, "ADVISOR 2.1: A User-Friendly Advanced Powertrain Simulation Using a Combined Backward/Forward Approach", in *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 48, NO. 6*, 1999.

[2] N. Mohan, T. Undeland and W. Robbins, *Power electronics*, 3rd ed. New York, N.Y: John Wiley & Sons, Inc, 1995, p. Ch 12, 13, 14, 15.

[3] "Toyota Prius - Power Split Device", Eahart.com, 2018. [Online]. Available: http://eahart.com/prius/psd/. [Accessed: 06- Aug- 2017].

[4] J. Dicker, G. Pennock and J. Shigley, *Theory of Machines and Mechanisms*, 3rd ed. New York: Oxford University Press, 2003, p. Ch10.

[5] "Sun-Planet Gear Model", *Mathworks.com*, 2017. [Online]. Available: https://www.mathworks.com/help/physmod/sdl/ref/sunplanet.html. [Accessed: 06- Aug- 2017].

[6] "Ring-Planet Gear Model", *Mathworks.com*, 2017. [Online]. Available: https://www.mathworks.com/help/physmod/sdl/ref/ringplanet.html. [Accessed: 06- Aug- 2017].

[7] U.S. Department of Energy, "EVALUATION OF THE 2010 TOYOTA PRIUS HYBRID SYNERGY DRIVE SYSTEM", Oak Ridge National Laboratory, Oak Ridge, Tennessee, 2011.

[8] G. Davies, "Toyota Prius Power Split Device", *Prius.ecrostech.com*, 2009. [Online]. Available: http://prius.ecrostech.com/original/PriusFrames.htm. [Accessed: 11- Sep- 2017].

[9] "Simscape Driveline Documentation - MathWorks United Kingdom", *Mathworks.com*, 2017. [Online]. Available: https://www.mathworks.com/help/physmod/sdl/index.html. [Accessed: 08- Aug- 2017].

[10] "Tire-Road Interaction (Magic Formula)", *Mathworks.com*, 2017. [Online]. Available: https://www.mathworks.com/help/physmod/sdl/ref/tireroadinteractionmagicformula.html. [Accessed: 08- Aug- 2017].

[11] "Tire (Magic Formula)", *Mathworks.com*, 2017. [Online]. Available: https://www.mathworks.com/help/physmod/sdl/ref/tiremagicformula.html. [Accessed: 08- Aug- 2017].

[12] "Vehicle Body", *Mathworks.com*, 2017. [Online]. Available: https://www.mathworks.com/help/physmod/sdl/ref/vehiclebody.html. [Accessed: 14- Oct- 2017].

[13] "How to verify RT-LAB's Modbus slave connection through third party software acting as Modbus master | OPAL-RT Knowledge Base", *Opal-rt.com*, 2016. [Online]. Available: https://www.opal-rt.com/KMP/index.php?/article/AA-01154/0/How-to-verify-RT-LABs-Modbus-slave-connection-through-third-party-software-acting-as-Modbus-master.html. [Accessed: 05- Jan- 2018].

CHAPTER 4
DYNAMOMETER TESTBED SYSTEM

4.1 Overview of the Dynamometer Testbed System

There are many different types of dynamometers ranging from simple springs to sophisticated absorption dynamometers; in all cases they are designed to measure force, power, or speed. For testing traction inverters, a dynamometer test setup would include two rotating machines connected by a gearbox or another sort of shaft coupling. One of these machines will function as the propulsion motor and the other will be the generator recirculating the power back to the power source. This results in a very efficient system: the only losses are the inefficiencies of the components with a large portion of the input power being recycled. The propulsion machine's speed is controlled by the unit under test (UUT), while the load machine's torque is controlled by the load inverter. This means that the motor-generator set can emulate a large range

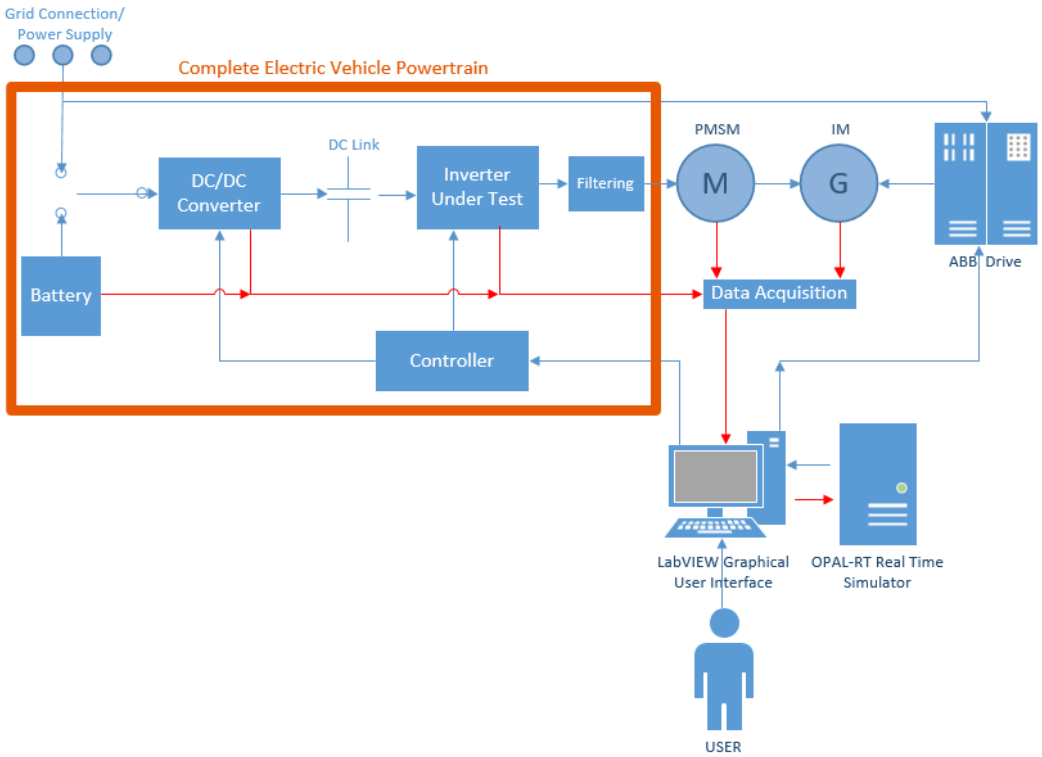

Figure 4.1: Dynamometer Test Setup

of operating points similar to what would be seen in a vehicle. Figure 4.1 shows the dynamometer testbed setup. Both the motor and the generator are equipped with cooling fans to force air into the machines to protect them from heat related damage that may occur when the testbed is operating near its limits. The propulsion motor in this setup is a permanent magnet synchronous motor (PMSM) which was custom made and is indicative of what would be seen in an electric vehicle. It is rated for 75 horsepower and up to 7000 rpm. The load machine is a 100 horsepower, maximum 8000 RPM induction motor controlled by an ABB drive using direct torque control (DTC) [1]. One advantage to the dynamometer testbed setup is that it is capable of testing the propulsion motor and its drive in all four quadrants of operation. However, all 4 quadrants of operation are not commonly represented in EPA Driving schedules, since two of the quadrants refer to the vehicle traveling in reverse. Figure 4.2 shows the forward movement of the motor operating point in two of the operating quadrants during a typical speed ramp. In Fig. 4.2 "forward" refers to clockwise rotation of the machine and forward motion of the vehicle. The red

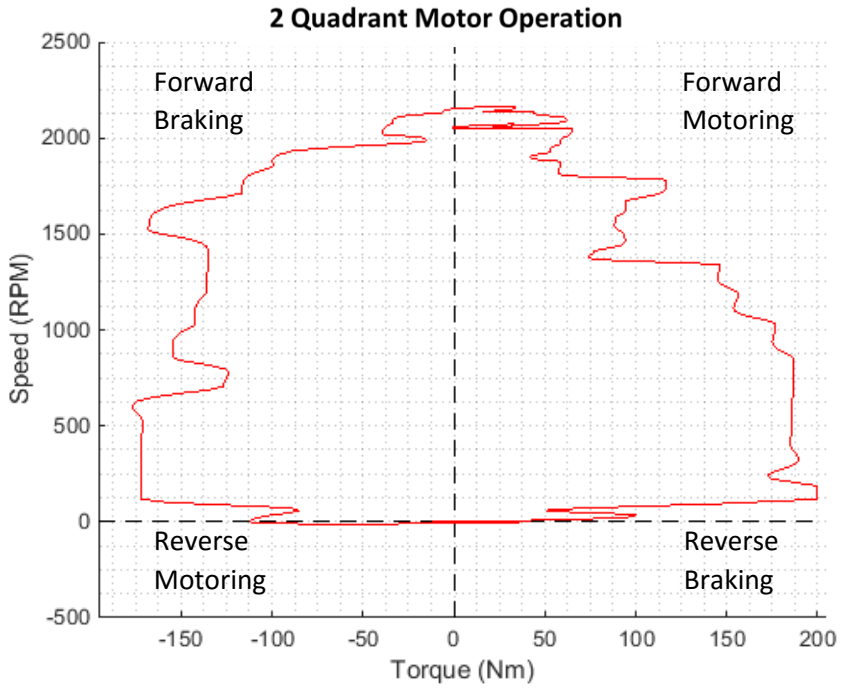

Figure 4.2: Motor Areas of Operation

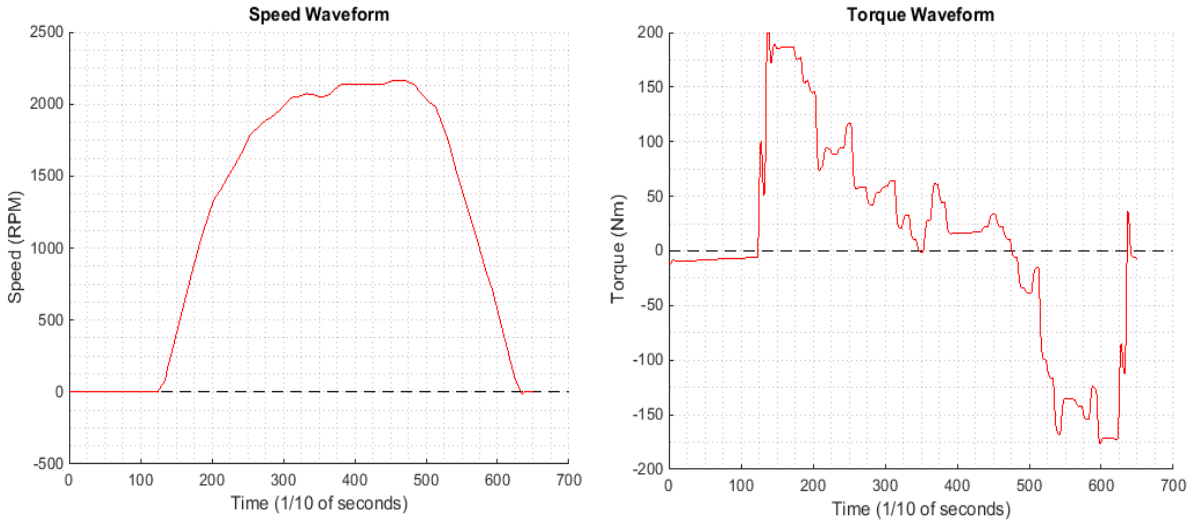

Figure 4.3: Section of UDDS used to Create Fig. 4.2

Speed vs Torque waveform shows that the motor will move frequently from forward motoring to generating throughout a typical driving schedule. The individual speed and torque waveforms were generated for a Toyota Prius running a short section of the EPA UDDS driving schedule and are shown in Fig. 4.3. The points in Fig. 4.2 where the waveform crosses the y-axis (0 speed) of the graph correspond to a change in the direction of power flow. In Quadrant 1 (forward motoring) the power flows from the power supply through the UUT and into the PMSM where it is applied to the load (left to right in Fig. 4.1). In quadrant 2 (forward breaking) the flow is reversed and represents a regenerative breaking scenario where the power flows from right to left in Fig. 4.1.

The vehicle simulation model discussed in Chapter 3 is implemented on an OPAL-RT^{TM} real-time simulator which calculates the speed and torque values for a given vehicle under user-defined conditions in real time. Those values are then relayed to the dynamometer control and data acquisition (DAQ) system via serial communication. The control and DAQ system is a FASTEK CATS-2000 measurement and control system based around National Instruments LabVIEW hardware and software. The OPAL-RT real-time simulator communicates with the

LabVIEW interface by MODBUS TCP/IP serial communication protocol. The speed and torque values are then communicated to the UUT and the load drive, respectively. The LabVIEW system records the shaft speed, shaft torque, line voltages, line currents, temperatures, and other parameters for further analysis.

The remainder of this chapter will address: the load motor drive and the direct torque control strategy used in its control, the LabVIEW data acquisition and control interface, the OPAL-RT real time simulator, and testbed to UUT communication protocol the testbed is currently implemented.

4.2 Load Motor Drive and Direct Torque Control (DTC)

Traction inverters for vehicle applications are typically speed controlled devices that use a form of frequency control based on PID controllers to regulate the motor output speed. This method holds the speed constant to the user input but allows the torque produced by the propulsion machine to fluctuate as needed up to the maximum ratings of the machine. The load machine implements direct torque control (DTC). Traditional methods such as frequency or vector control typically require some form of speed feedback, a coordinate transformation (usually from the d-q reference frame), and PI or PID regulators. DTC is much simpler, in that the feedback variables for the control loop are calculated using a motor model and no speed feedback or coordinate transformation is necessary [1]. Figure 4.4 shows a general torque control loop similar to the one developed by ABB. This DTC control strategy is implemented on the ABB ACS800 series drive to control the load machine [1-2]. The adaptive motor model that is used to calculate the torque and flux produced within the load motor uses motor parameters determined during a motor identification run. The identification run measures motor data such as stator resistance, mutual inductance, saturation coefficients, and inertia.

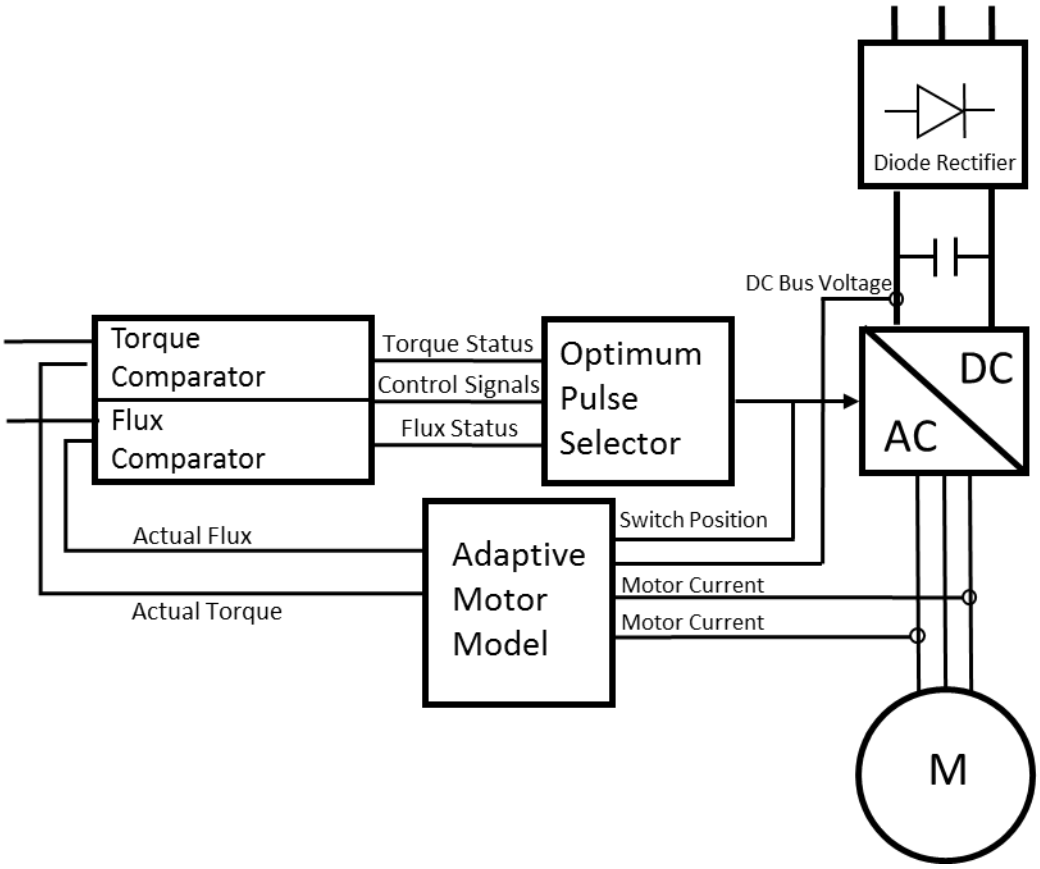

Figure 4.4: DTC Control Loop with Active Motor Model

IMs can be modeled with respect to different reference frames – referred to the stator (as discussed in Chapter 2), rotor, or its synchronous speed. The synchronous speed reference frame is also known as the excitation reference frame (e-frame) where the motor equations can be formulated as a vector with its real element in the d-axis (direct) and imaginary component in the q-axis (quadrature) [1, 3]. The rotor flux and output torque can thus be controlled independently by controlling the d- and q-axis currents, respectively. Equations (1) and (2) show the voltage across the IM stator in the e-frame:

$$\vec{V}_{ds}^{e} = R_s \vec{i}_{ds}^{e} + L_s \frac{d\vec{i}_{ds}^{e}}{dt} \tag{61}$$

$$\vec{V}_{qs}^{e} = R_s \vec{i}_{qs}^{e} + L_s \frac{d\vec{i}_{qs}^{e}}{dt} \tag{62}$$

where $\vec{V}_{ds}^{e}$ and $\vec{V}_{qs}^{e}$ are the stator d-q voltages in the e-frame, R_s and L_s are the stator resistances and inductances, respectively, and $\vec{\imath}_{ds}^{e}$ and $\vec{\imath}_{qs}^{e}$ are the stator currents in the e-frame. The stator flux in each axis can be represented by the $L_s \frac{d\vec{\imath}_{xs}^{e}}{dt}$ terms in Eqs. (1) and (2). Reformulating Eqs. (1) and (2) with the change in stator flux $\frac{d\vec{\psi}_{xs}^{e}}{dt}$ results in:

$$\vec{V}_{ds}^{e} = R_s \vec{\imath}_{ds}^{e} + \frac{d\vec{\psi}_{ds}^{e}}{dt} \tag{63}$$

$$\vec{V}_{qs}^{e} = R_s \vec{\imath}_{qs}^{e} + \frac{d\vec{\psi}_{qs}^{e}}{dt} \tag{64}$$

Solving Eqs. (3) and (4) for the stator flux along each axis yields:

$$\vec{\psi}_{ds}^{e} = \int \left(\vec{V}_{ds}^{e} - R_s \vec{\imath}_{ds}^{e}\right) dt \tag{65}$$

$$\vec{\psi}_{qs}^{e} = \int \left(\vec{V}_{qs}^{e} - R_s \vec{\imath}_{qs}^{e}\right) dt \tag{66}$$

This means that the stator flux can be estimated as long as the stator d- and q- axis currents can be observed. The electromagnetic torque (T_e) and the stator flux vector ($\vec{\psi}_s^{e}$) equations that are used in the estimator in Fig. 4.4 are shown below [1].

$$T_e = \frac{3}{2} P \left(\vec{\imath}_{qs}^{e} \psi_{ds}^{e} - \vec{\imath}_{ds}^{e} \psi_{qs}^{e}\right) \tag{67}$$

$$\vec{\psi}_s^{e} = \vec{\psi}_{ds}^{e} + \vec{\psi}_{qs}^{e} \tag{68}$$

In the ABB drive, both the flux and torque that are calculated in the motor model are fed into the comparators, where the signals are compared every 25 μs to the reference values. The output of the comparators are determined by a two-level hysteresis control method. Unlike traditional PWM drives, DTC has no predetermined switching pattern. DTC has been referred to as “just in time” switching because there are no unnecessary switch changes. Up to 30% of

switch changes in PWM drives can be unnecessary [2]. The stator flux can be controlled by varying the applied stator voltage vector as follows [1, 3]:

$$\Delta\vec{\psi}_s = \vec{V}_s \Delta t \tag{69}$$

Figure 4.5 shows the six control sectors with the voltage vector selections in each. The black vector represents the stator flux vector and the blue indicates the increment necessary in each sector for clockwise rotation of the motor.

The optimal pulse selector used in the ABB DTC selects the appropriate switch combination every 25 μs for reaching or maintaining an accurate motor torque. The efficiency, accuracy of the model, and the processing speed of this DTC inverter produce a system with a torque response of less than 2 ms with an accuracy of 2% [2].

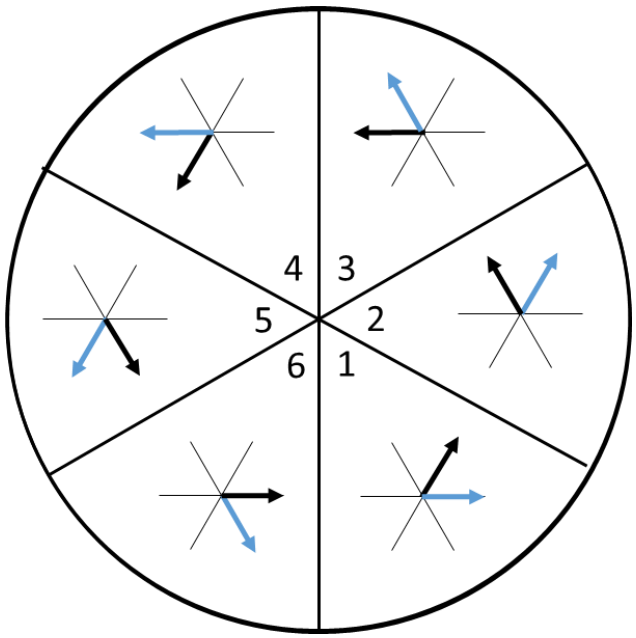

Figure 4.5: Voltage Sectors for DTC

4.3 LabVIEW Data Acquisition and Testbed Control Interface

National Instruments (NI) hardware and their LabVIEW software were chosen for the data acquisition system for the testbed. In addition, the LabVIEW code also controls the flow of speed and torque commands to the appropriate drives. The LabVIEW testbed graphical user interface is shown below in Fig 4.6. The interface is implemented on a PXIe-8102 processor which is part of the Fastek CATS-2000 configurable automated test system. Figure 4.7 shows the wiring diagram for the components of the CATS system. The diagram also includes the NI product names of each component in the system.

The LabVIEW interface controls the load ABB drive, the UUT and the DC power supply. The drives can be operated in multiple modes: user can input specific speed and torque values, they can automatically be selected from a .csv (comma separated values) file, or the OPAL-RT simulator can provide speed and torque values based on a certain vehicle. The user controls for the drives are located in the top left corner of the interface in the “ABB Control” and “UUT Control” sections. The specific controls necessary to enable the UUT may vary based on the design of the control board. For the ABB load drive, Table 4.1 shows the appropriate switch positions to enable the different drive modes. In those same control sections, there are input terminals for the target speed of the UUT and the target torque for the ABB. The real-time simulation parameters can also be changed in the “RT-Lab Control” section on the left side of the interface. The software current limit, wind speed, road grade, and internal combustion engine contribution (if a hybrid vehicle is desired) can be changed mid simulation if necessary. The DS Reset option also provides a quick way to restart the driving schedule once it has been completed. Live readouts of the speed and torque values that are being sent to both motor drives

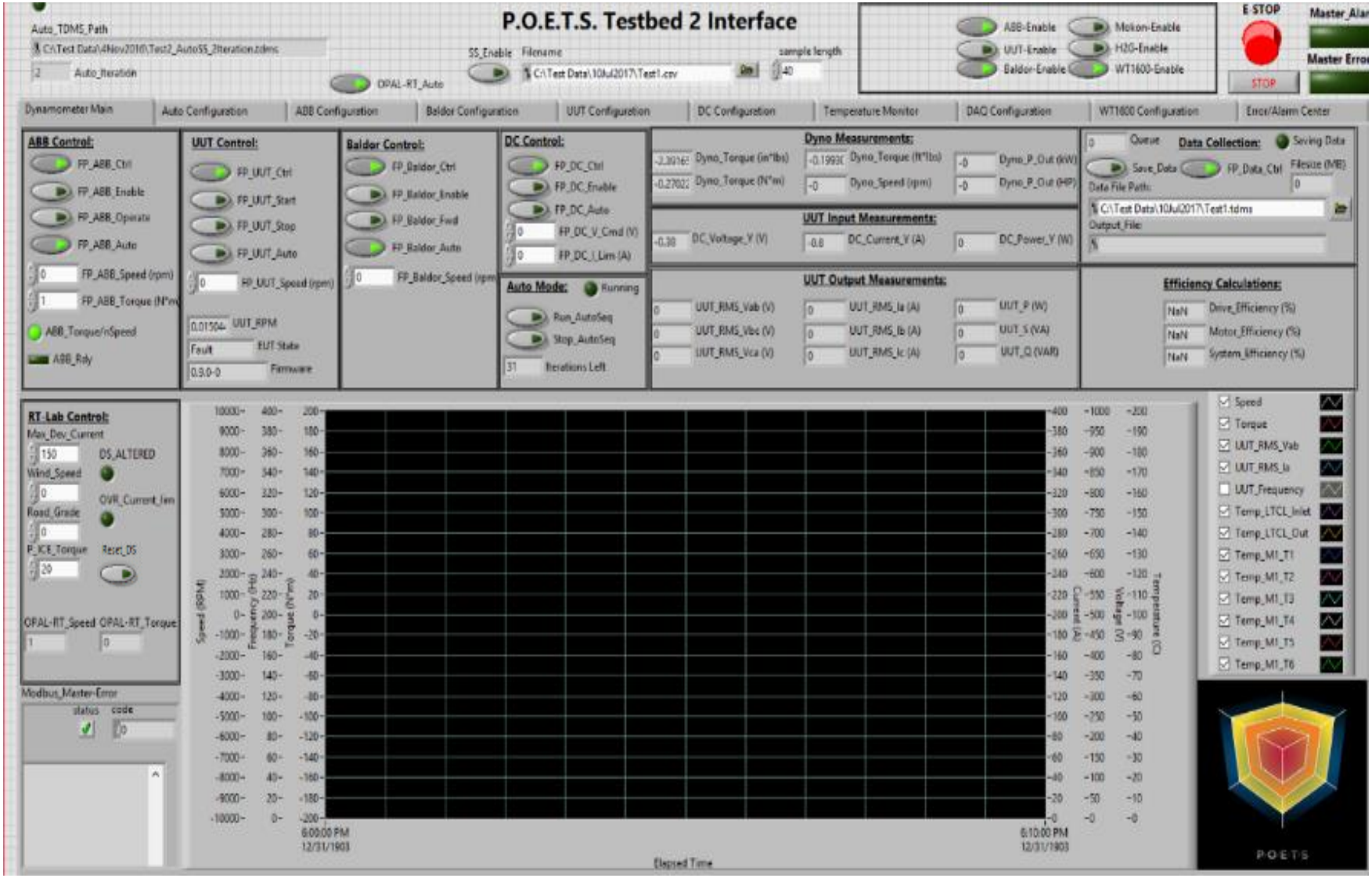

Figure 4.6: Testbed LabVIEW User Interface

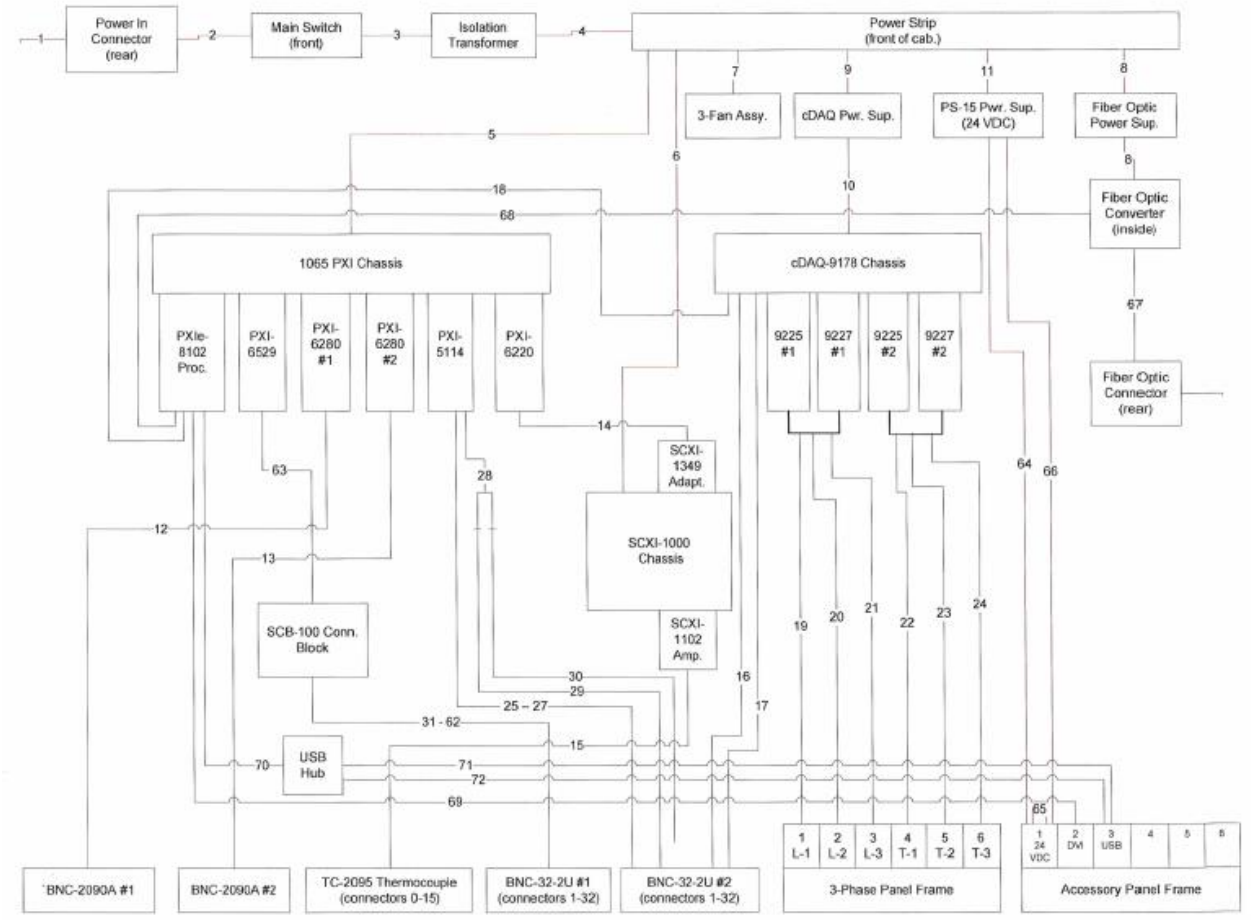

Figure 4.7: CATS 2000 Diagram

Table 4.1: Switch Positions for Testbed Operating Modes

Mode	Enable	Notes
Single user input Speed and Torque	FP_ABB_Control, FP_ABB_Enable, FP_ABB_Operate	Manual input of speed and torque values for UUT and load drive
.csv input of Speed and Torque	FP_ABB_Control, FP_ABB_Enable, FP_ABB_Operate, FP_ABB_Auto, select .csv file in interface	Scheduled speed and torque values every 500ms
Simulator provides Speed and Torque	FP_ABB_Control, FP_ABB_Enable, FP_ABB_Operate, FP_ABB_Auto, OPAL-RT_Auto	Speed and torque values from simulator are updated to the drives every 250ms

are visible at the bottom of the RT-Lab Control section. In the DC control section, the DC power supply voltage and current limit can be set.

The data acquisition system records 145 channels at varying frequencies. The output current and voltage of each phase of the UUT, the shaft torque, and DC bus current are measured and recorded at 80 kS/s (kilo-samples per second). The current measurements are made using a current transformer. The phase voltage measurements are made using a voltage probe and the torque is measured using a rotary torque sensor. The other channels are recorded at either 4 or 8 S/s. A Yokagawa WT1600 power analyzer is also connected to the output lines of the inverter to calculate RMS current, RMS voltage, real power, apparent power, reactive power, and power factor on a per phase basis. The power analyzer data (94 channels) is recorded at 4 S/s by the DAQ. The remaining channels record either thermocouple or pressure sensor information at 4 or 8 S/s. The pertinent data is also displayed in real time in the user interface.

The user must select an appropriate file path to record the data and enable the “Save Data” option in the upper right corner of the interface in the “Data Collection” section (shown in Fig. 4.6). It is important to keep an eye on the file size indicator when operating the testbed and recording data, if this becomes much larger than a gigabyte, then, it may become very time consuming to view the large data files. Simply cycle the Save Data button position and the file will auto increment and continue saving to a new file. LabVIEW saves the data to a .tdms file which is readable with NI DIAdem software. They can also be read into the Matlab workspace by using the TDMS Reader code available from the MathWorks file exchange [4]. This function set converts the .tdms files into structured Matlab data sets for easy manipulation.

4.4 Opal-RT Real-Time Simulator

The Opal-RT OP5031 Simulator and OP4520 FPGA Expansion unit are connected to the LabVIEW system via serial communication. In order for the Matlab/Simulink model to run on the OPAL-RT simulator, certain formatting changes need to be made. The model must be divided into two subsystems, one which includes the simulation and another that includes the variables and scopes which the user wishes to view in real-time. Any signal that is entering either of these subsystems must be routed through an RT-Lab "OpComm" block. These blocks are used by RT-Lab to identify parameters required for communication between the nodes in the hardware configuration [5]. To set up the Modbus TCP/IP communication with the testbed LabVIEW interface, an RT-Lab Modbus TCP device block must be added. This block requires a configuration file that details the communication nodes, their IP addresses, and the protocol data structure, an example of which is shown below in Table 4.2 and in [6]. The configuration file must be included in the RT-Lab project folder. The input lines to the TCP device represent the information that is being sent from the MODBUS slave device (OPAL-RT simulator) to the MODBUS master device (LabVIEW) and correspond to the input variable types listed in the table (coil inputs, discrete inputs, holding register inputs, and input registers). The coil inputs are used as flag variables to notify the user if the torque or speed profile has been limited to prevent the current from exceeding the user defined limit. The holding register inputs are integer numbers that correspond to the simulation speed and torque. The outputs of the block represent the information that is received from the Modbus Master device (LabVIEW). The list of feedback measurements available are listed in Section 3.8. The coil outputs are used for the Driving Cycle reset feature of the testbed. Figure 4.8 shows the LabVIEW code for

Table 4.2: Modbus Configuration File Example

Block name	block1
Device name	modbus_slv_tcp
Device type	Slave
Protocol	TCP
Network Interface	eth0
IP address	192.168.1.51
Slave ID	31
Port	1504
CPU	5
Verbose	1
Addresses of coil inputs	0, 1
Addresses of coil outputs	16, 17
Addresses of discrete inputs	0 1
Addresses of holding register inputs	0; 1;
Addresses of holding register outputs	20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31
Addresses of input registers	0
Polling frequency (ms)	100

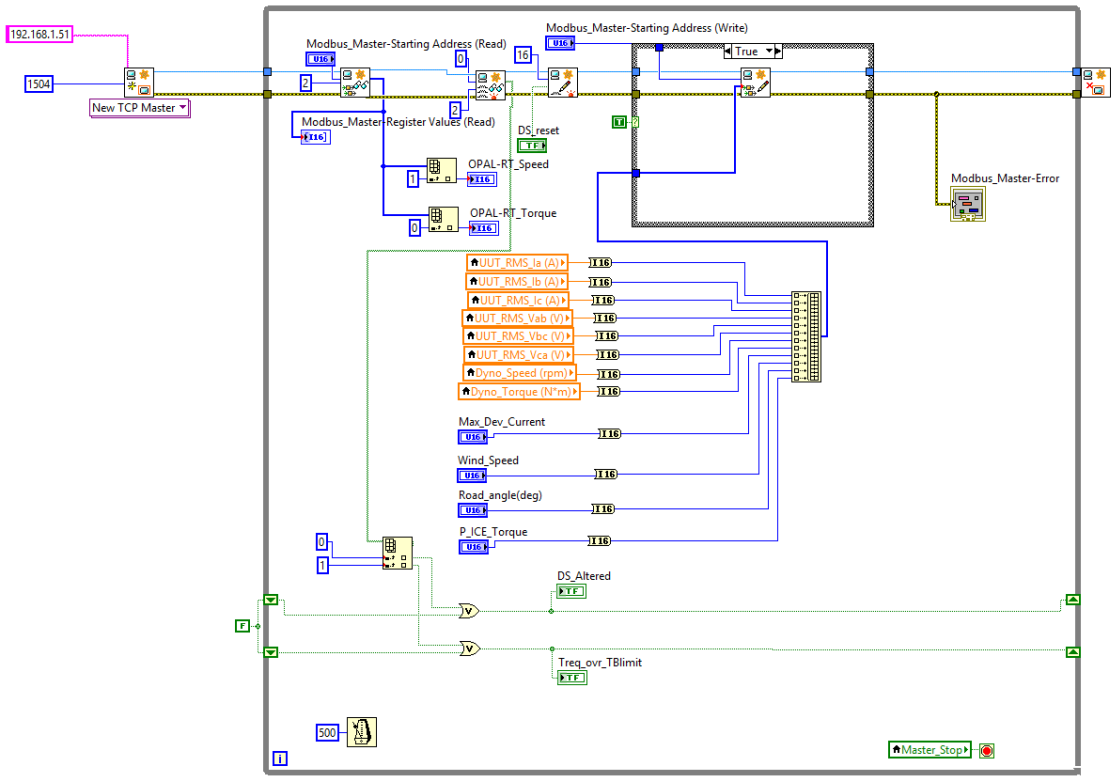

Figure 4.8: LabVIEW Modbus TCP Communication Code

communicating with the OPAL-RT simulator via Modbus TCP/IP. On the far left, the TCP Master is instantiated and the OPAL-RT IP address and port number that the cable is connected to must be set here. Inside the gray “while” loop, the other Modbus blocks read and write to the coils and holding registers. The speed and torque values are set according to the values of the simulation. The measurement feedback data is also gathered and concatenated into a single array of information for transmission back to the simulator. A toggle structure is also created within the “while” loop which notifies the user if the driving schedule has been altered by the simulation to meet the software current limit or if the torque command has been limited because the requested value exceeds the testbed’s torque limit of 200 Nm.

In order to run the Testbed with the simulation generated speed and torque values, the user must initiate the simulation and select the driving schedule before enabling the testbed. To

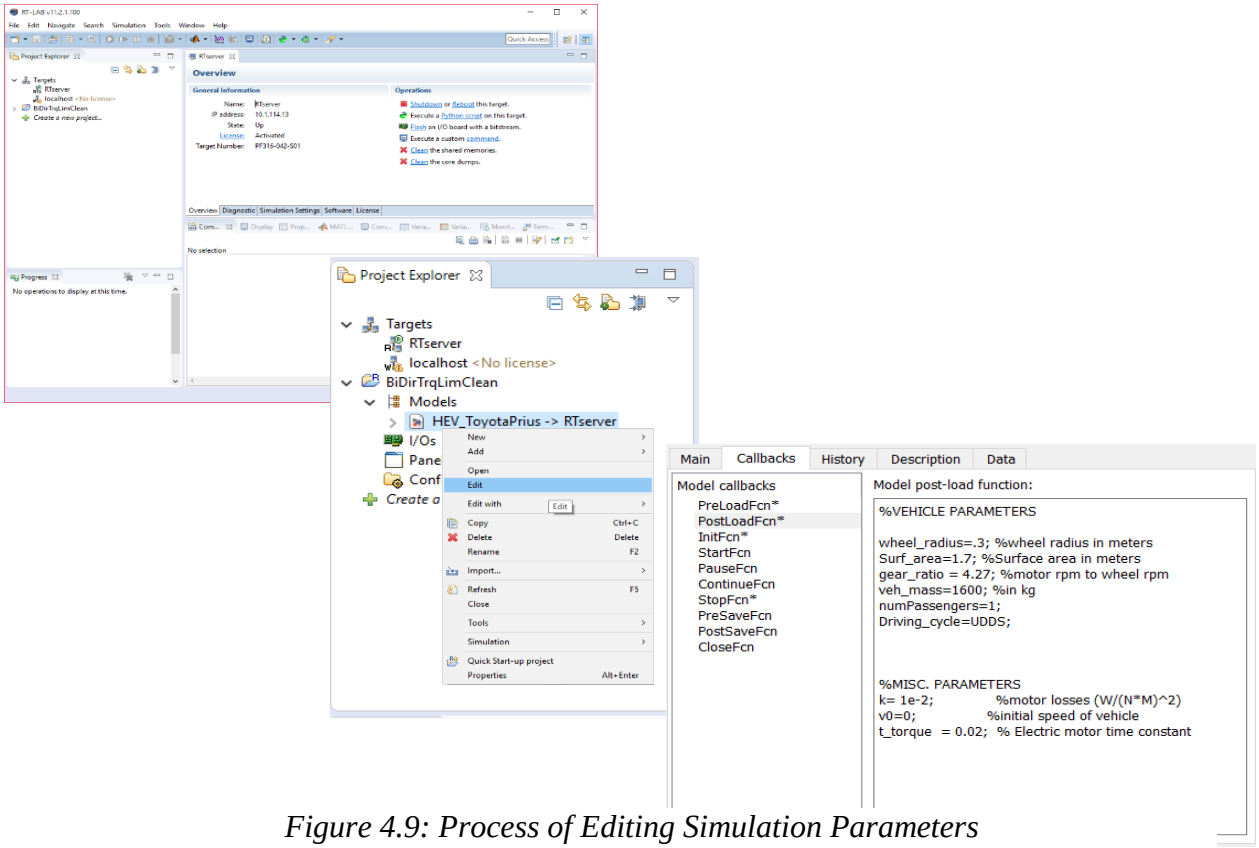

Figure 4.9: Process of Editing Simulation Parameters

do so, open RT-Lab and expand the correct project in the project explorer window. Then expand the models section and right click on the model and select Edit. This will open a new instance of Matlab where you will be able to edit the model. Next, navigate to File > Model Properties >Model Properties then select the Callbacks tab. This displays the model callback functions in the left pane, and the contents in the right hand pane. This process is illustrated in Fig. 4.9. The pre-load function is used to load all of the driving schedules into the workspace. More load statements can be added here to incorporate custom driving schedules. The post-load function sets the vehicle physical parameters such as the wheel radius, surface area, vehicle mass including passengers, and motor constants. The driving cycle is also selected here. Once the parameters are set, select OK then save and exit the model. You must then build the model and load it onto the simulator before executing the LabVIEW code. More detailed explanation for the startup procedure is given in the Testbed Manual in Appendix A.

4.5 Testbed to UUT Communication

The communication between the LabVIEW interface and the present UUT i.e., a Baldor H2 industrial drive is serial communication using Modbus TCP/IP protocol. The NI hardware is capable of both serial communication with various protocols, and CAN (controller area network) communication, which is the standard communication protocol for the automotive industry. Appendix B covers more communication capabilities of the Testbed.

In general, serial communication interfaces stream data one bit at a time and can be implemented on as few as two wires. Serial communication interfaces can be divided into two main categories: synchronous and asynchronous. Synchronous serial communication interfaces usually require more wires because they require a clock signal to be transmitted with the data. The most common type of serial communication interface is asynchronous, thus, not requiring a

clock signal. However, more effort is required to ensure that the data is being transmitted and received reliably. This extra effort comes in the way of serial communication protocols which define the structure of the data including synchronization bits, parity bits and the baud rate or speed at which it will be transmitted. The hardware required for these serial interfaces varies based on type of serial bus. An interface where only one way communication is available is referred to as simplex and only requires a single transmission line. Half duplex refers to two way communication that cannot transmit and receive simultaneously. Full duplex, the most common form of serial interface, allows simultaneous data transmission and receiving as illustrated in Fig. 4.10 [7-8]. The selected communication protocol will have a voltage level associated with it that the UUT hardware must be able to withstand. The required voltage level can vary widely from 0-3.3V for transistor-transistor logic level communication typical between low-level ICs to ±25V that is common in RS-232 communication between computers and peripheral equipment.

Generally speaking each serial interface will have a messaging structure protocol that is implemented in software and a hardware (transport) protocol for the medium of transmission. There are many forms and standards for communication, the remainder of this section will cover Modbus TCP/IP protocol and the hardware necessary for use with the Testbed.

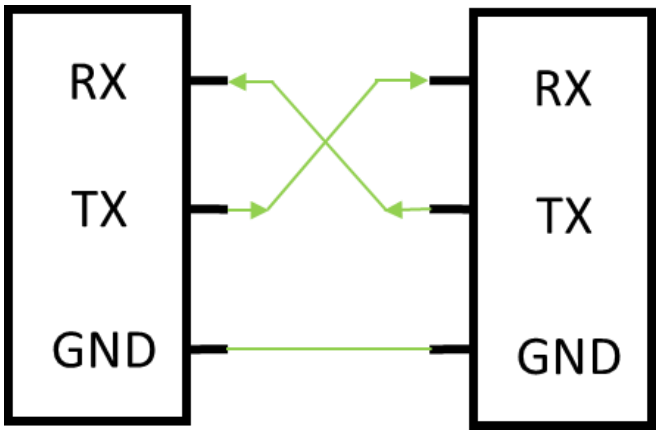

Figure 4.10: Full Duplex Serial Interface

4.5.1 Modbus TCP/IP Protocol

The Modbus messaging structure is the application structure that defines the organization and interpretation rules independent of the medium of transmission. Modbus is a master/slave protocol, meaning the master controls the flow of information from all the slave devices. TCP/IP refers to the Transmission Control Protocol and Internet Protocol which is the medium of transmission for the messages. The primary function of TCP is to ensure that all messages are received correctly while IP is used to ensure that messages are correctly addressed [9]. Each device using TCP/IP is connected with an Ethernet cable and will have an associated IP address similar to a computer on a network. Modbus RTU (remote terminal unit) has the same basic messaging structure as Modbus TCP/IP, however the transport protocol is either RS-232 or RS-485. The only other difference in the transmission packets is that the TCP data structure is six bytes longer to include a header for routing [10] and the cyclic-redundant checksum of the RTU frame is removed. The construction of the TCP/IP data packet or application data unit (ADU) from a traditional Modbus RTU serial frame is shown in Fig. 4.11 [9]. The items from the RTU packet that are not included in the TCP/IP packet have a red background in Fig. 4.11.

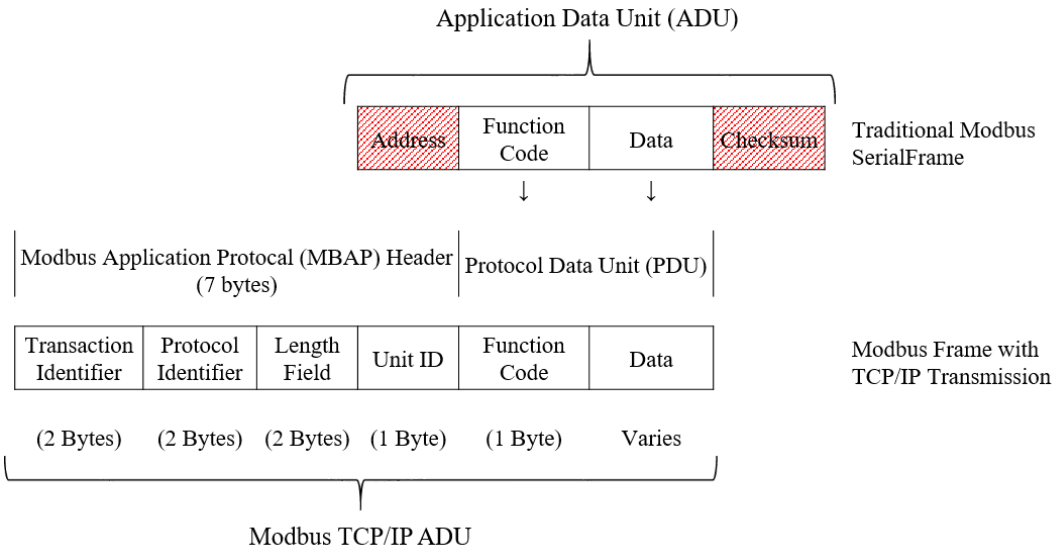

Figure 4.11: Modbus Application Data Unit Construction

With Modbus RTU, once the ADU is constructed it is then ready to be transmitted. With Modbus TCP/IP, a protocol stack of additional information must first be added in multiple layers. The ADU must first be packaged in each of the protocols in the stack before it can be transmitted across the physical hardware, it is then unpacked by the receiving equipment. All of the equipment connected via Modbus TCP must run a protocol stack to interpret this information. Each of the layers on the stack communicate with the receiving equipment's stack via information stored in headers. The information contained in the stack is standard for TCP/IP communications. Figure 4.12 illustrates the five layers of the protocol stack, their associated headers, and their package structure [9].

For TCP/IP communication, the standard physical layer is an RJ45 connector and a Cat5 Ethernet cable. The pinout for the male and female RJ45 connectors is shown in Fig. 4.13 [11]. Modbus TCP/IP communication only needs TX and RX connections thus only lines 1, 2, 3, and 6 are used. The signals sent over the RX and TX lines are ±2.5V in magnitude.

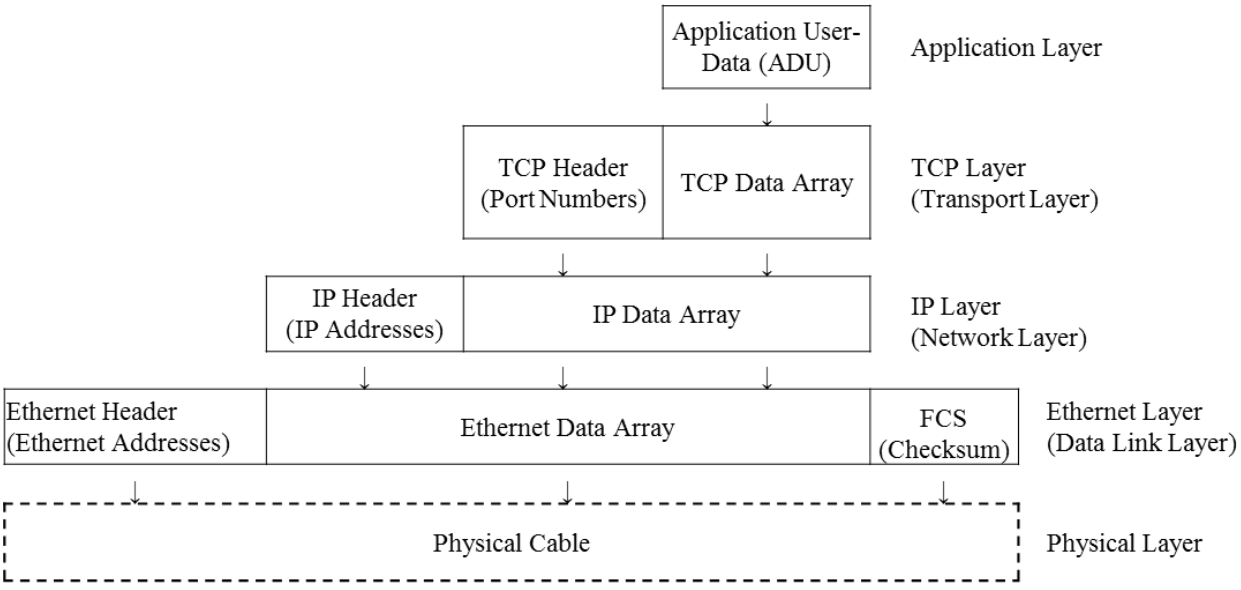

Figure 4.12: Construction of TCP/IP-Ethernet Packet

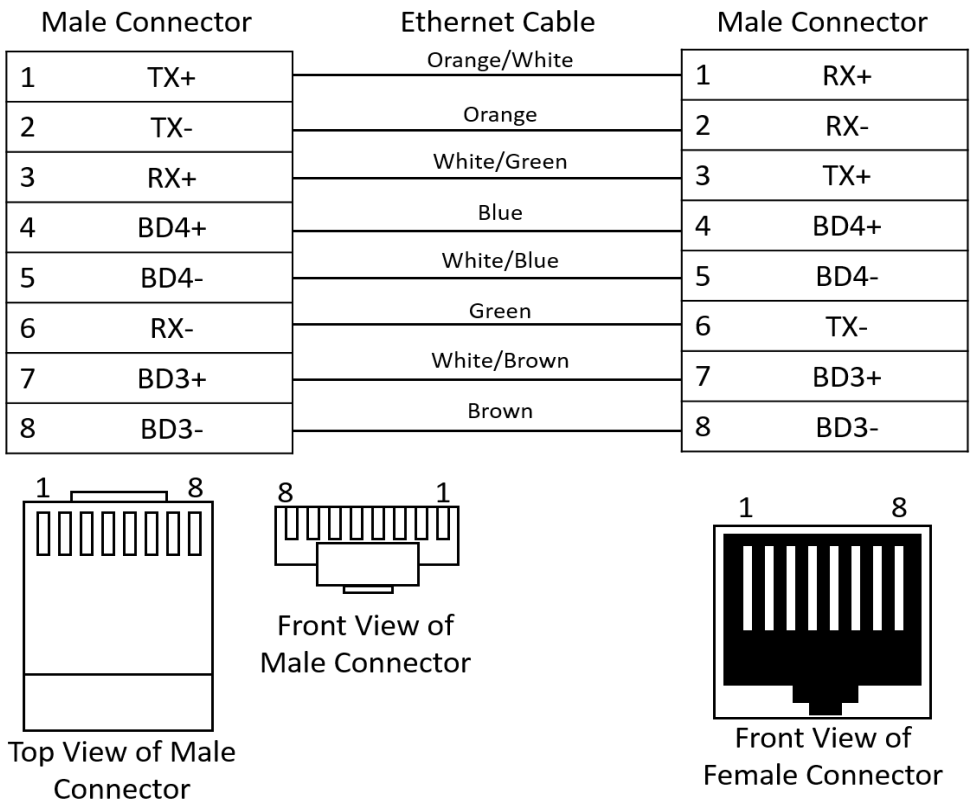

Figure 4.13: RJ45 Connection Diagram

Modbus RTU and TCP/IP can be implemented in the LabVIEW control interface easily for communication with the UUT. The control interface will serve as the Master device and the UUT will be the Slave device for the Modbus communication. The LabVIEW data communication toolbox provides pre-made virtual instruments (VIs) that do many of the communication tasks like reading and writing to holding registers and coils. These virtual instruments, like the “New TCP Master” block shown in the LabVIEW TCP/IP communication code (Fig. 4.14) are pre-made blocks of basic LabVIEW components that perform complicated tasks. They are the equivalent of subsystems in Matlab/Simulink. The LabVIEW Master Instance documentation discusses the different setup options for TCP/IP and Modbus serial type RTU or ASCII [13]. Modbus RTU is the default and communicates the data in binary format. It can also be changed to ASCII format which uses human readable characters to send data.

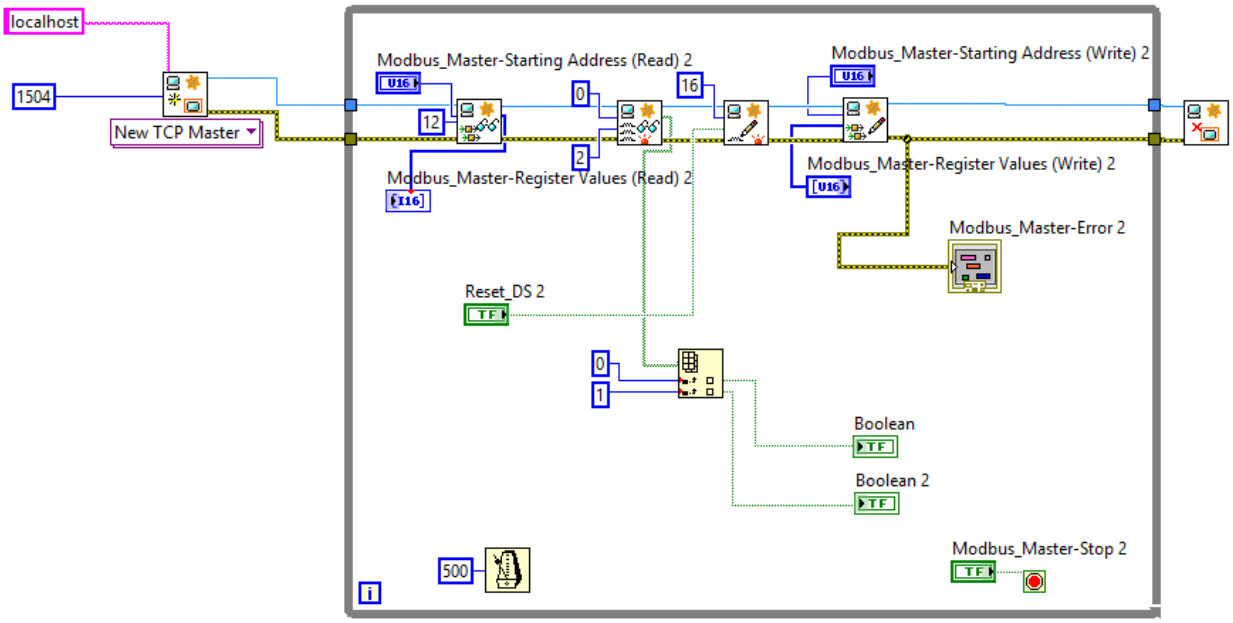

Figure 4.14: Modbus TCP LabVIEW Implementation

4.6 Conclusions

This chapter has presented an overview of the dynamometer testbed, a look at the DTC implemented on the load motor drive, an introduction to the LabVIEW control interface, a look at how the OPAL-RT real time simulator communicates with the system, and communication protocols that are implemented with the testbed. Appendix B includes more information on the different communication protocols that the testbed is capable of using as well as the development of custom communication protocols. Chapter 5 will discuss testing capabilities and show data comparison between the results from ADVISOR, the vehicle simulation, and the measured data from the testbed for multiple test cases.

4.7 References

[1] Z. Alnasir and A. Almarhoon, "Design of Direct Torque Controller of Induction Motor (DTC)", *International Journal of Engineering and Technology (IJET)*, vol. 4, no. 2, 2012.

[2] "Technical guide No.1 Direct torque control - the world's most advanced drive technology", 2018.

[3] G. Buja, D. Casadei and G. Serra, "DTC-Based Strategies for Induction Motor Drives", *Industrial Electronics, Control and Instrumentation*, 1997.

[4] J. Hokanson, "TDMS Reader", *MathWorks File Exchange*, 2012. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/30023-tdms-reader?requestedDomain=true. [Accessed: 26- Aug- 2017].

[5] "RT-Lab Solo Getting Started User Manual", 2003.

[6] "How to verify RT-LAB's Modbus slave connection through third party software acting as Modbus master", *OPAL-RT Knowledge Base*, 2016. [Online]. Available: https://www.opal-rt.com/KMP/index.php?/article/AA-01154/8/HowTo/How-to-verify-RT-LABs-Modbus-slave-connection-through-third-party-software-acting-as-Modbus-master.html. [Accessed: 30- Jan- 2017].

[7] "Serial Communication", *Learn.sparkfun.com*, 2018. [Online]. Available: https://learn.sparkfun.com/tutorials/serial-communication. [Accessed: 31- Jan- 2017].

[8] "BASICS OF SERIAL COMMUNICATION", *What-when-how.com*, 2018. [Online]. Available: http://what-when-how.com/8051-microcontroller/basics-of-serial-communication/. [Accessed: 31- Jan- 2017].

[9] *INTRODUCTION TO MODBUS TCP/IP*. Wixom MI: Acromag, 2005, pp. 4-15.

[10] "Modbus RTU Protocol Overview", *Real Time Automation*, 2018. [Online]. Available: https://www.rtaautomation.com/technologies/modbus-rtu/. [Accessed: 31- Jan- 2017].

[11] "Tech Stuff - LAN Wiring and Pinouts", *Zytrax.com*, 2018. [Online]. Available: http://www.zytrax.com/tech/layer_1/cables/tech_lan.htm. [Accessed: 01- Feb- 2018].

[12] L. Frenzel, "What's The Difference Between The RS-232 And RS-485 Serial Interfaces?", *Electronic Design*, 2016. [Online]. Available: http://www.electronicdesign.com/what-s-difference-between/what-s-difference-between-rs-232-and-rs-485-serial-interfaces. [Accessed: 02- Feb- 2018].

[13] "Create Master Instance VI - LabVIEW 2017 Real-Time Module Help - National Instruments", *Zone.ni.com*, 2017. [Online]. Available: http://zone.ni.com/reference/en-XX/help/370622R-01/lvmve/modbus_create_master_instance/. [Accessed: 02- Feb- 2018].

CHAPTER 5
TESTING AND EXPERIMENTAL RESULTS OF THE TESTBED

5.1 Overview of Testing Capabilities

The real-time simulator, simulated vehicle model, testbed control interface, and testbed hardware provide a system for testing the next generation of power electronic converters for vehicle applications. Before a prototype inverter is used with the simulation-generated speed and torque commands, significant testing should be completed at both low and high power levels with resistive load banks to verify the functionality of the power converter. It is also recommended that the prototype undergo various no-load testing scenarios on a standalone motor without any load torque. Then, the prototype should be used with the testbed. Next, the non-simulation based operating modes for the testbed that were introduced in Section 4.3 should be used to further validate the prototype inverter operation. These operating modes include manually entering speed and torque commands and scheduling speed and torque values in a .csv (comma separated value) file. The vehicle model can be used once confidence has been gained throughout the operating area of the inverter. The simulation should be run offline before using the HIL system with the prototype inverter. The user can then verify the output speed and torque waveforms satisfy the test criteria and are within UUT limits. Incorporating the vehicle model creates harsh, dynamic, and fast changing speed-torque profiles for a vehicle exhibiting typical driver behavior. EPA or custom driving schedules can be implemented to test specific driving conditions or environments.

The vehicle simulation also takes into consideration a user defined current limit, which will limit the torque and speed ramp rates to keep the RMS phase current of the motor below the defined level. This prevents the user from running a driving schedule that requires more power

than the UUT can provide. The torque and speed outputs of the vehicle simulation are also limited to the capabilities of the equipment, meaning that the commanded values for speed and torque will not exceed the limits of the testbed regardless of the simulated vehicle requirements. The PMSM propulsion motor is rated to 75 HP (56 kW), 7000 RPM, and 220 Nm of torque. The shaft of the propulsion motor and the load machine (100 HP induction machine) are connected via a torque coupler that is rated for 200 Nm and a torque transducer to measure the speed and torque values at the shaft. The testbed has bidirectional power flow capabilities ensuring that all four quadrants of electric motor operation can be tested thoroughly. Current transformers and voltage probes are used to measure the output of the UUT at the terminals of the PMSM. Thermocouples can also be placed on the UUT and other powertrain components to monitor key component temperatures.

The LabVIEW control interface communicates the speed and torque commands to the propulsion motor drive and load motor drive via Modbus TCP. The speed and torque commands can be user set, scheduled by a .csv file, or generated by the vehicle simulation. The latter can emulate many different environmental conditions in addition to the driving schedules including headwinds, road grade, and inclement weather road surface conditions (e.g. wet tarmac, snow, and ice). This is done by varying the formula coefficients for the tire model as discussed in Chapter 2. The test facilities also includes an Envirotronics Endurance series environmental chamber, heat exchanger, 105°C coolant loop, and a 10-ton chiller. The ambient temperatures achievable in this thermal chamber range from -75°C to 175°C.

This chapter covers more on the EPA driving schedules, and asserts that analysis of the vehicle simulation results and driver behavior can lead to better powertrain design. The calculation of the maximum acceleration limit of the system as a result of the inertia of both

motor rotating masses follows. Lastly, testing results from the proof-of-concept tests with the Baldor H2 Vector drive acting as a propulsion inverter will also be presented. A comparison of the simulation output shaft speed and torque and ADVISOR is made to ensure that the simulation results are accurate. The measured speed and torque of the physical system is then be compared to the simulation output to show that the system is capable of emulating "real-world" conditions.

5.2 EPA Driving Schedules for Analysis of Driver Behavior

As introduced in Section 1.4, EPA driving schedules were developed to standardize emission testing for combustion engines. In order to accurately measure the emissions in real world conditions, the speed profiles were designed to emulate driver behavior under different environments. The Urban Dynamometer Driving Cycle (UDDS) presents a speed profile that is indicative of light-duty vehicles driving in an urban environment. There are many starts and stops followed by relatively quick accelerations to speeds between 30 and 40 mph. The US06 represents aggressive, high acceleration, highway driving with fast accelerations from a stop to 70 mph and sustained speeds of over 50 mph for a distance of 8 miles. Highway Fuel Economy Test (HWFET) represents a more conservative highway drive than the US06 and covers 10 miles with an average speed of 48 mph. The New York City Cycle (NYCC) emulates one of the most congested cities in America by featuring low-speed, stop-and-go traffic with a total distance of 1.18 miles with an average speed of 7.1 mph [1]. Custom driving cycles can also be used with the vehicle simulation.

Analysis of the driving cycles and driver behavior can provide insights for powertrain component design as was done for a SiC boost converter in [2]. A bivariate histogram, as shown in Fig. 5.1, can be used to visually represent the operating points where the inverter will most

frequently operate. The information in Fig. 5.1 shows that a Toyota Prius in an urban setting (UDDS driving cycle) will spend most of its time operating under 10 kW and around 1500 RPM. This information is useful when determining the power requirements for a vehicle-type specific inverter. For instance, the UDDS and similar schedules would be considered more than the US06 for an electric van designed to make deliveries in busy city centers. In the more aggressive US06 schedule, the inverter may need to be able to handle up to 80 kW as shown in Fig. 5.2 but only for a relatively short period of time. The majority of the operating points in the US06 driving schedule are below 20 kW, which means it could be possible to select less expensive switching devices or perhaps choose an inverter topology that maximizes efficiency in this operating range. However, it must still be capable of reaching the 80kW peak power requirement for short periods of time.

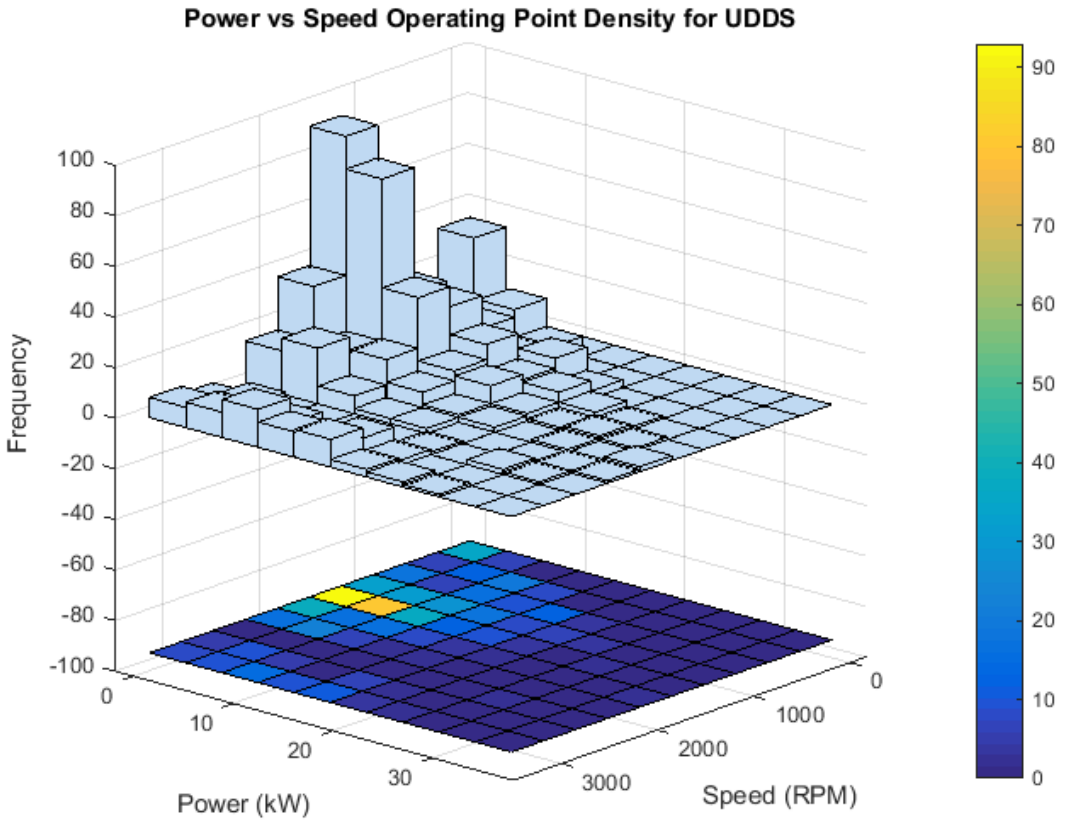

Figure 5.1: UDDS Histogram of Operating Points

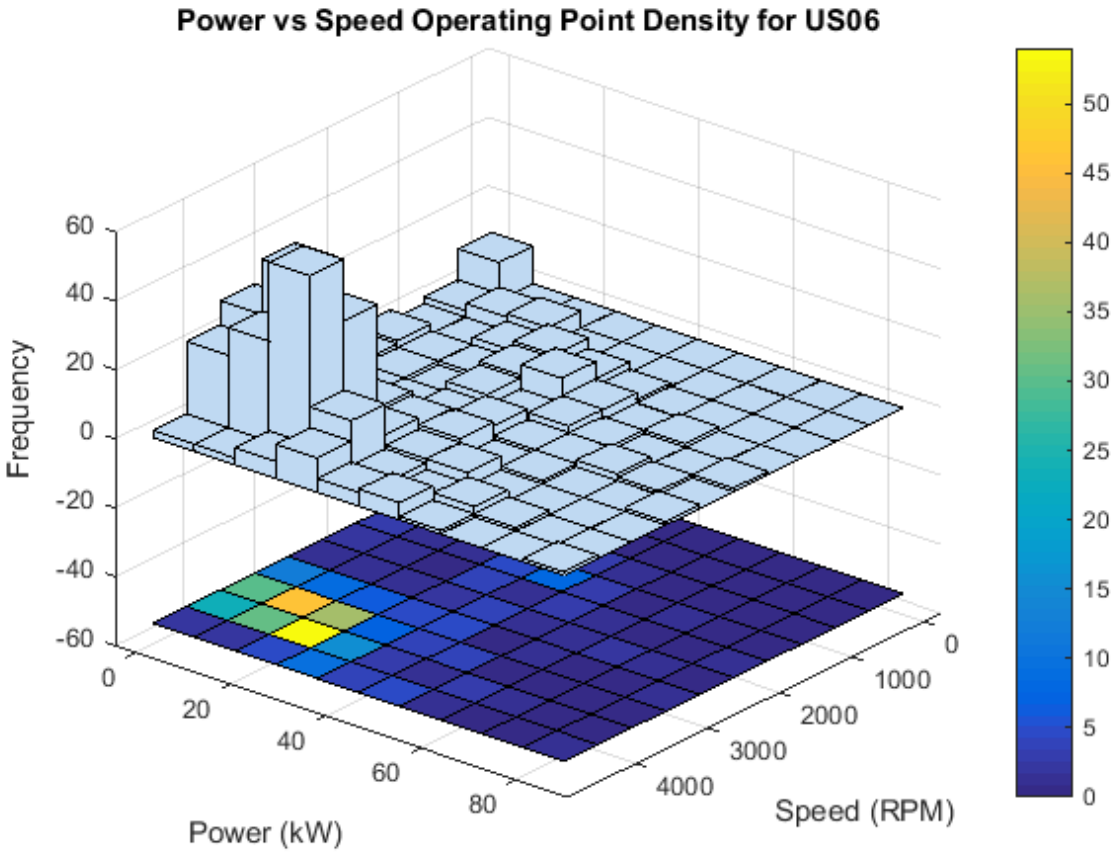

Figure 5.2: Histogram of US06 Operating Points

5.3 Maximum Acceleration Limit and Inertia Measurements

The connected shafts of the motor and generator form a rotating mass. From an analysis of Newton's 2nd Law of motion, a mass cannot be accelerated instantaneously. An instantaneous change in speed results in:

$$a = \frac{dv}{dt} = \frac{dv}{0} = \infty \tag{70}$$

where *a* is acceleration, *dv* is the change in linear velocity, and *dt* is the change in time. This infinite acceleration would therefore result in an infinite force when inserted into Eq. (2).

$$F = ma \tag{71}$$

Table 5.1: Maximum Continuous Current Ratings for the Testbed Equipment

Equipment	Maximum Current Rating (RMS)
Baldor B1137600 PMSM (75hp) (60Hz rated)	202A
Baldor B1137599 IM (100hp)(120Hz rated)	119A
ABB ACS800 motor drive	291A

The force required for a speed change is directly proportional to the required torque, which in an electric motor is proportional to the motor current [3]. This means the limiting factor for motor accelerations is the current carrying capability of the UUT and testbed equipment. Table 5.1 shows the continuous RMS current ratings for the testbed equipment.

The electromagnetic torque produced by the electric motor, the torque required for an acceleration, and the RMS current calculation are given by:

$$T_{em} = T_{acc} + T_{load} \tag{72}$$

$$T_{acc} = J_{eq} \times \frac{2\pi}{60} \times \frac{\Delta n}{\Delta t} \tag{73}$$

$$I_{RMS} = \frac{T_{em}}{T_{nominal}} \times I_{nominal} \tag{74}$$

where T_{em} is the electromagnetic torque produced by the motor, T_{acc} is the torque required to accelerate the motor shafts, T_{load} is the torque calculated by the vehicle simulation and applied by the load drive, J_{eq} is the combined inertia of the two motor shafts, Δn is the change in RPM, Δt is the change in time, $T_{nominal}$ and $I_{nominal}$ are the motor rated torque and current values, respectively [3]. This relationship between acceleration torque, load torque, and electromagnetic torque is illustrated in Fig. 5.3. Though the load torque value was commanded to a constant 50 Nm, there was an acceleration torque when the rotor speed changed in addition to the load torque that was measured at the shaft. The direction of the acceleration torque corresponds to the direction of the speed change. The measured electromagnetic torque and speed are shown in blue and orange,

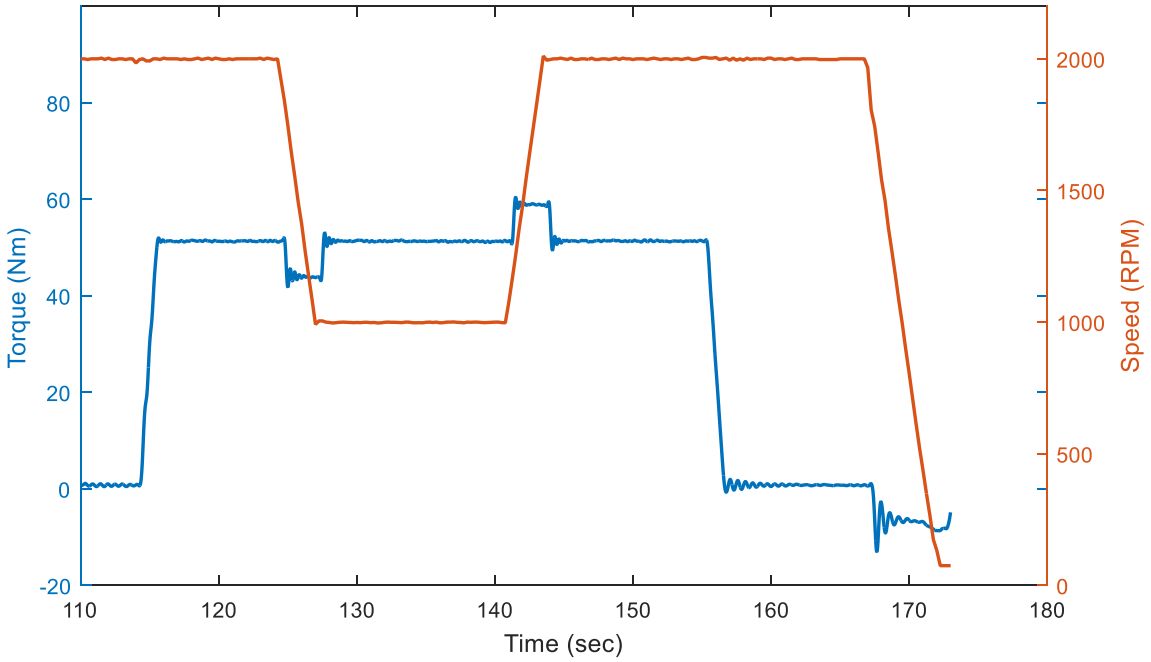

Figure 5.3: Speed and Torque Waveforms Illustrating T_{acc}

respectively. The acceleration torque equation (Eq. (4)) will be used to determine the maximum ramp rates of the system since the limiting factor will be the current carrying capabilities of the testbed equipment. Section 3.7.1 presents the vehicle simulation calculation of the acceleration torque and limits speed ramps in some instances to keep the current below the user defined maximum.

In order to calculate the maximum acceleration possible for the testbed equipment and the UUT using Eq. (4), the equivalent inertia of the two motor shafts must be measured. The retardation test is typically used for measuring the inertias of motor rotors and is discussed further in [4]. To determine the inertia of the combined motor rotors the PMSM is rotated at 2000 RPM and power is removed, allowing the rotor to decelerate unimpeded as shown by the orange waveform in Fig. 5.4. The blue waveform represents the torque measured at the shaft. After determining the slope of the deceleration curve and the torque value at the same operating

point, Eq. (4) can be used to solve for the inertia [4-5]. A point near the midpoint of the curve was chosen for the inertia measurement. The tangent line at this point is shown in Fig. 5.5 and the acceleration torque present at this point is measured at -1.2 Nm. The calculation of the equivalent inertia follows:

$$\frac{\Delta n}{\Delta t}|_{t\approx 60} = \frac{932.8 - 981}{1} = -48.2\ RPM/sec \tag{75}$$

$$T_{acc} = J_{sys,measured} * \frac{2\pi}{60} * \frac{\Delta n}{\Delta t} \tag{76}$$

$$-1.21\ Nm = J_{sys,measured} * \frac{2\pi}{60} * (-48.2) \tag{77}$$

$$J_{sys,measured} = 0.237\ kgm^2 \tag{78}$$

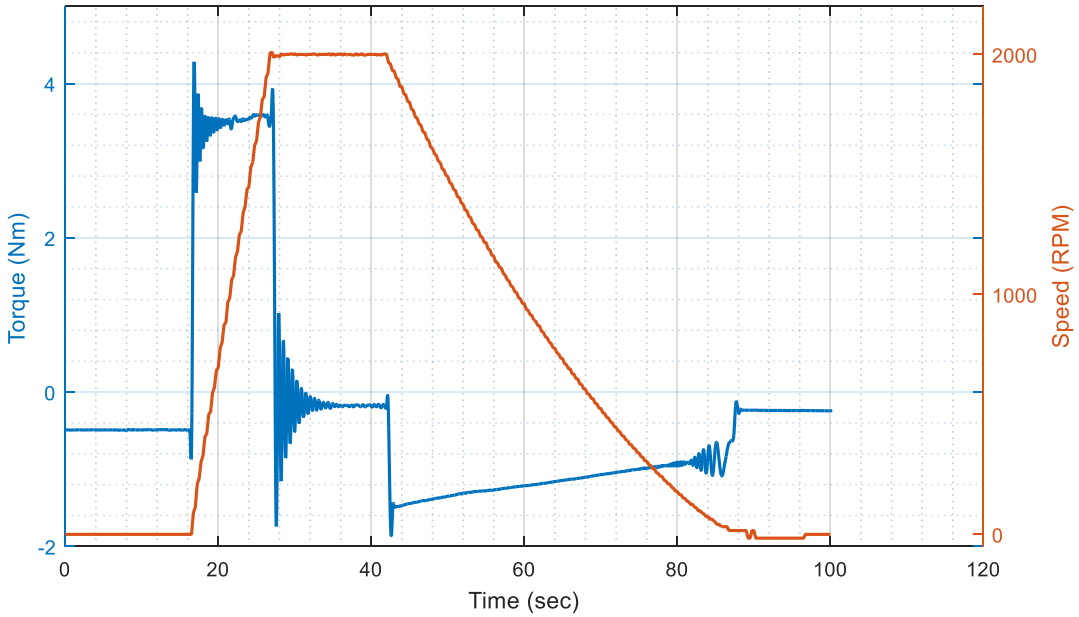

Figure 5.4: Retardation Test Data

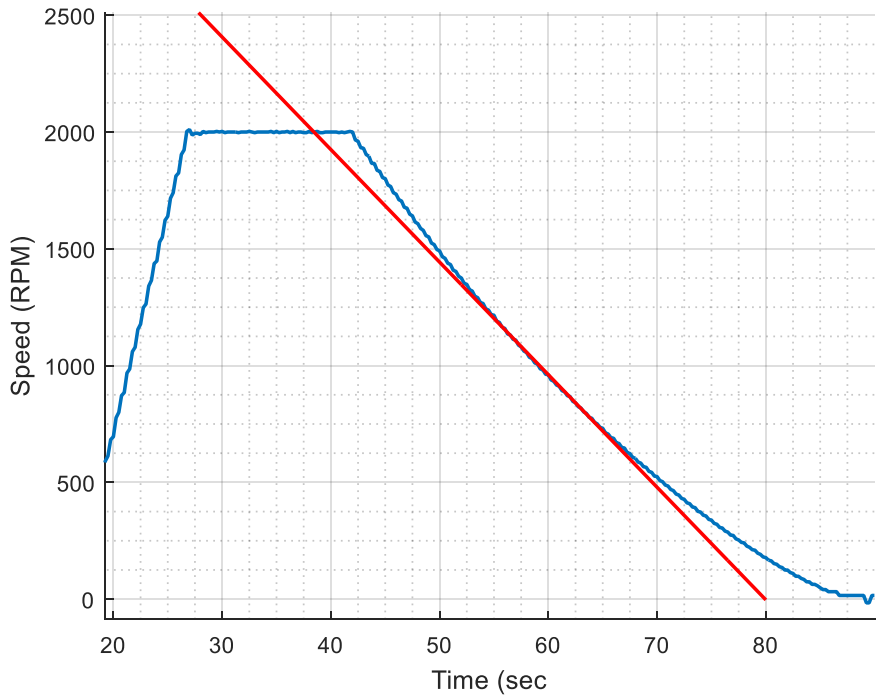

Figure 5.5: Tangent Line Used for Inertia Calculation

The slope of the tangent line at t = 60 seconds was determined by approximating the secant line with points equidistant from the chosen operating point. The measured system inertia was calculated to be 0.237 kgm^2. However, the combined inertia of the rotors as released by the motor manufacturer (Baldor) are 0.302 kgm^2. The measured value is less than the rotor inertias published by Baldor because they neglect friction and windage losses that act opposite to the direction of the shaft rotation, effectively slowing down the rotor faster [5].

With this inertia value, the maximum acceleration ramp rates of the testbed can be determined using Eq. (4). The PMSM has a maximum instantaneous overload current of 3 times the rated current or 606 A, and a maximum speed of 7350 RPM. If the UUT could provide enough current, the testbed could theoretically be accelerated to the max speed in just under 0.4 seconds at no load. However, this would result in an acceleration torque in excess of 460 Nm, so this is limited by the capabilities of the torque couplers that join the two motor shafts; these are

rated for only 259 Nm. Therefore, the theoretical minimum time to accelerate from 0 to 7350 RPM is 0.8 seconds at no load (T_{load}=0 Nm) and 3 seconds at full load (T_{load} = 200 Nm). Table 5.2 shows the acceleration torque required for a 0 to 7350 RPM change in a given amount of time (Δt) and the RMS current required at no load and full load.

Although these very fast accelerations are theoretically possible with the testbed, they are very unlikely to occur in "real world" scenarios. Even the harshest EPA driving schedules, like the US06, only has a maximum ramp rate of 10.82 seconds. As seen from Eqs. (3)-(4), Fig. 5.3, and Table 5.2, quickly accelerating the machines can add a significant amount of torque to the load in addition to the load torque calculated by the vehicle simulation and applied by the load machine. The current limiting feature prevents a driving schedule from requesting a ramp rate that would exceed the user defined current limit.

Table 5.2: Acceleration Torque and RMS Phase Current Calculations

Δt (seconds)	T_{acc} (Nm)	I_{req} (RMS) No Load	I_{req} (RMS) Full Load
0.1	1846.32	1790.9304	1984.93
0.2	923.16	895.4652	1089.465
0.3	615.44	596.9768	790.9768
0.4	461.58	447.7326	641.7326
0.5	369.264	358.18608	552.1861
0.6	307.72	298.4884	492.4884
0.7	263.76	255.8472	449.8472
0.8	230.79	223.8663	417.8663
0.9	205.1467	198.9922667	392.9923
1	184.632	179.09304	373.093
2	92.316	89.54652	283.5465
3	61.544	59.69768	253.6977
4	46.158	44.77326	238.7733

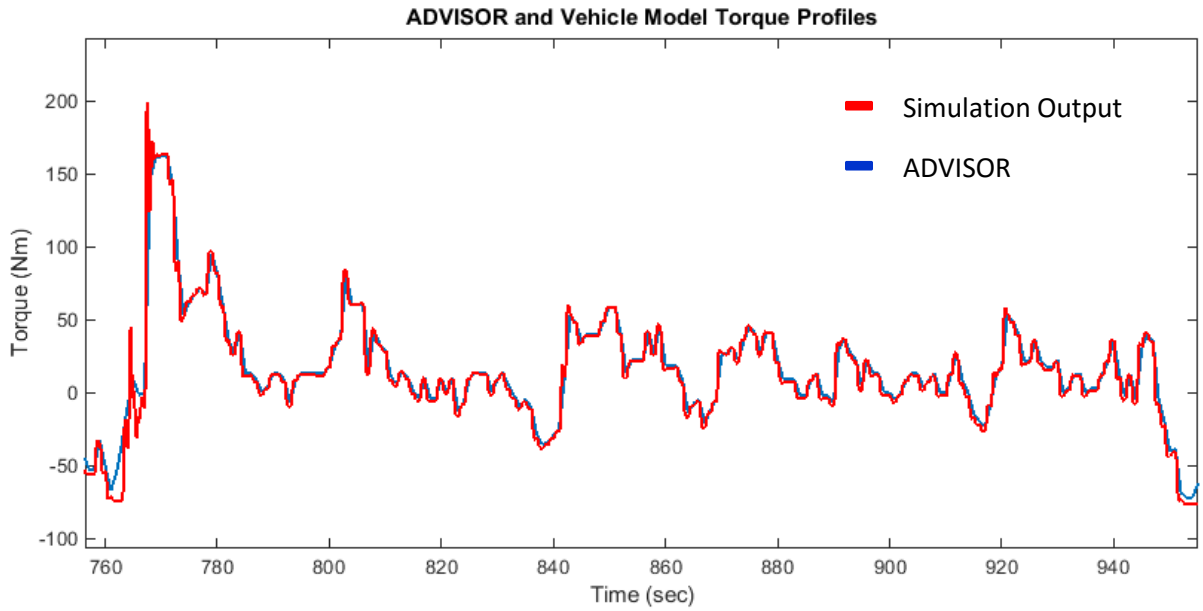

Figure 5.6: Comparison of ADVISOR and Vehicle Model Torque Waveforms

5.3 Proof-of-Concept Testing and Results

There are three different modes in which the dynamometer testbed can be operated: (1) manual input of speed and torque commands, (2) scheduled input of speed and torque commands (from .csv file), and (3) by input from the vehicle simulation. Both manual and scheduled input options are typically used for evaluating powertrain components at one operating point for multiple seconds. When using the vehicle simulation to input the commands, the user should be aware that these profiles are indicative of "real world" scenarios which can include fast accelerations at substantial loads.

First, the Simulation output for a given vehicle was compared with NREL's ADVISOR, a section of which is shown in Fig. 5.6, to ensure its accuracy. The vehicle model developed in this thesis is represented by the red trace and the results from ADVISOR are the blue trace. The vehicle model has a 10 times smaller step size than ADVISOR, which results in a waveform that captures more of the transients between operating points.

In order to validate the vehicle model and HIL system, a 100Hp Baldor H2 Vector drive was used as the UUT. Unlike an inverter designed for vehicle applications, The H2 Vector drive does not have bidirectional power flow capabilities and its protection software severely limits its performance. The H2 drive can only provide power to the system and has no mechanisms other than a small breaking resistor to dissipate the energy from the negative currents generated by the propulsion machine that are common during EV and HEV regenerative breaking scenarios. To further illustrate this scenario, the electromagnetic torque produced at the shaft of the motor for a Toyota Prius on a driving schedule, generated by the vehicle simulation, is shown in Fig. 5.8. For positive torque values, the PMSM (propulsion motor) provides torque in the direction of shaft rotation and moves the vehicle forward. When the torque is negative, the load machine provides torque in the direction of rotation and causes the PMSM to behave like a generator as the vehicle slows down. In this state the load machine is emulating the vehicles forward momentum.

Since the Baldor H2 drive cannot dissipate the energy that would be generated by the negative half of the torque waveform, the torque commands that are sent to the testbed are limited to positive values by the vehicle simulation. The torque limits can be altered by changing the boundaries of the saturation block in the “Torque Limiting” subsystem discussed in Section 3.7.1. For full value waveforms, the saturation limits should be set to 200 Nm and -200 Nm to fit within the safe operating limits of the testbed.

5.3.1 Toyota Prius Under UDDS Driving Schedule

To validate that the system can control the speed of the propulsion machine and the torque of the load machine, the Baldor H2 Vector drive was used as a temporary UUT and subjected to the UDDS load profile. Figure 5.7 shows the UDDS driving schedule that served as the input to the vehicle simulation that was set for an 1100 kg (~2500 lbs) electric vehicle

resembling a Toyota Prius with a frontal surface area of 1.7 m^2, 0.3 m wheel size, and 1 passenger. The wind speed and road grade were set at zero and the RMS current limit was set to 100 A to match the capabilities of the H2 Vector drive. The simulation outputs (the motor speed in RPM and the electromagnetic torque required for the vehicle to achieve the commanded speed) are shown in Fig. 5.8. The speed and torque commands were then limited to 0-2500 RPM and from 0-200 Nm since the Baldor drive speed controller becomes unstable above 2500 RPM and does not have bidirectional power flow capabilities (Fig. 5.9). However, the testbed is capable of ±200 Nm and up to 6000 RPM. With the current limit lower than what is required for the vehicle to follow the speed command, the vehicle speed will deviate from the driving schedule. Figures 5.10 and 5.11 show speed and torque waveforms generated by the simulation that illustrate in red where in the schedule the current limit is exceeded. These simulation results can be generated by the vehicle simulation prior to use of the testbed and can be used in the test plan to describe the expected UUT behavior. Figure 5.12 shows a portion of the driving schedule where the ramp rates were altered to meet the current limit. The orange waveform represents the speed obtained from the driving schedule and the blue represents the speed the inverter can achieve without exceeding the current limit. An inverter that is well suited for the car represented in the vehicle simulation should closely follow the driving schedule. In this test, the driving schedule requirements only exceeded the capabilities of the UUT by a small margin so there is a good agreement between the two waveforms.

Figure 5.13 shows the measured speed and torque (averaged) at the shaft of the propulsion motor which match closely with the simulated values. The raw unfiltered data and the averaged torque waveform are shown in Fig. 5.14. The torque waveform was filtered for noise and averaged to eliminate the torque ripple introduced by the Baldor drive. The averaged

waveform gives a better indication of the performance of the system since the torque ripple is a consequence of the suboptimal UUT design and control. An efficient and well controlled UUT will produce torque waveforms with very little ripple. Figures 5.15 and 5.16 show the vehicle simulation output overlaid on the testbed measurement data. There is a close correlation between the commanded and measured speed that can be seen in more detail in Figs. 5.17 and 5.18. The largest discrepancy between the commanded and measured values occurs in scenarios where the commanded torque is zero. When the simulated vehicle is slowing down the load torque should be negative as seen in Fig. 5.8 but it is limited to zero for this test. The acceleration torque due to the inertia of the motors as discussed in Section 5.3 is still present and causes this negative torque on the shaft when the motors are decelerating. Breaking resistors should be incorporated into the test layout to avoid any potential damage to the UUT power supply due to the power flow direction in this scenario.

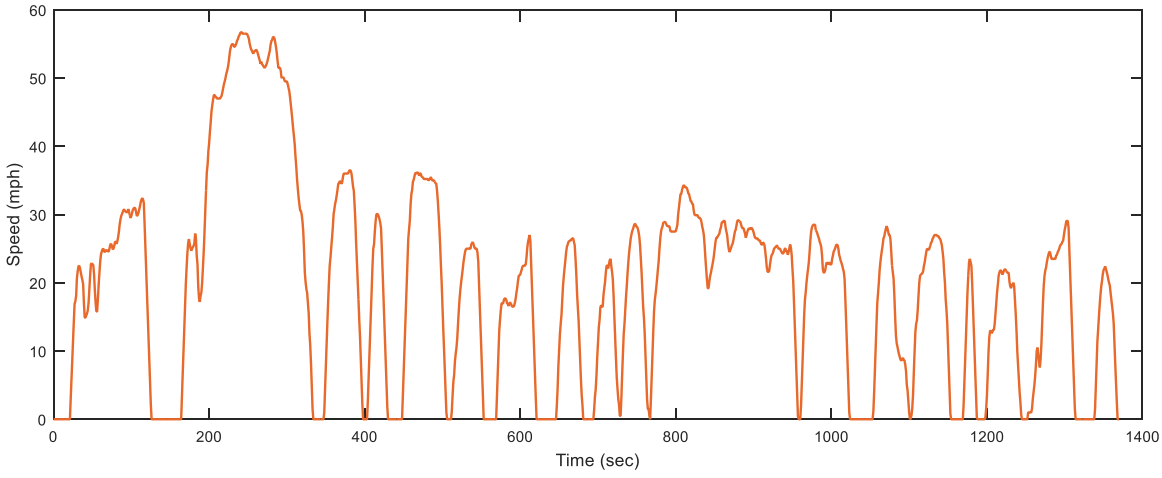

Figure 5.7: EPA UDDS Schedule

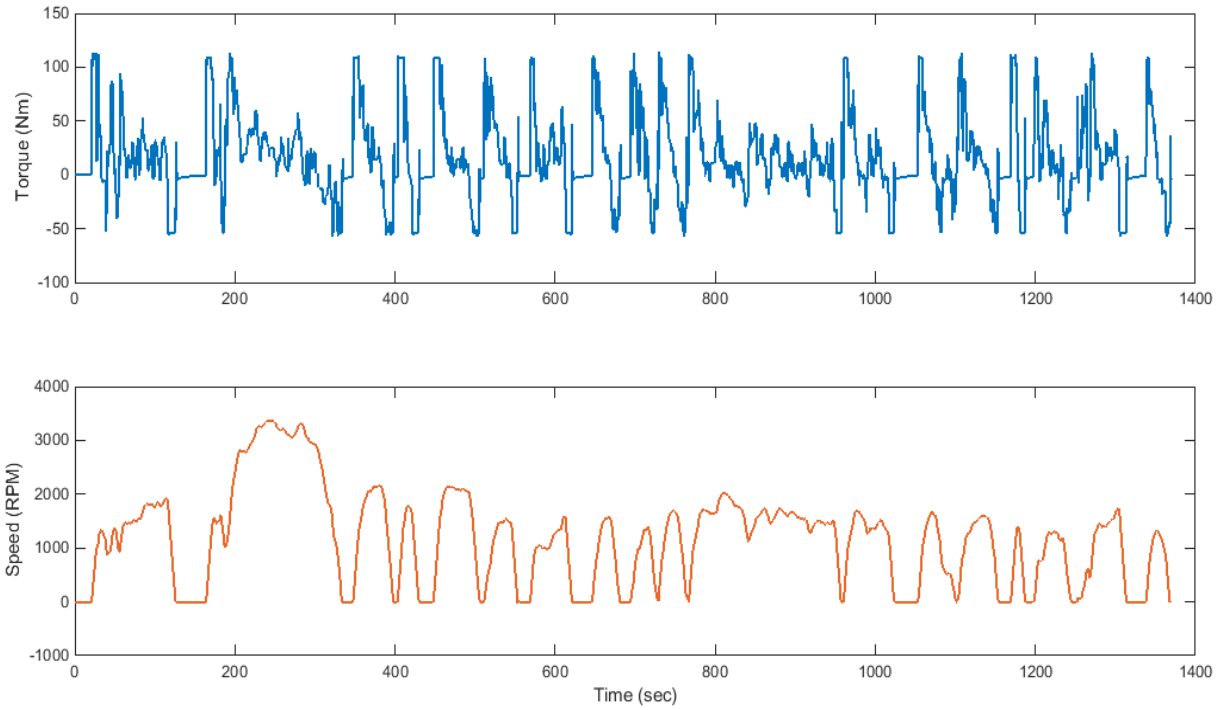

Figure 5.9: Simulation-Generated Torque and Speed Waveforms

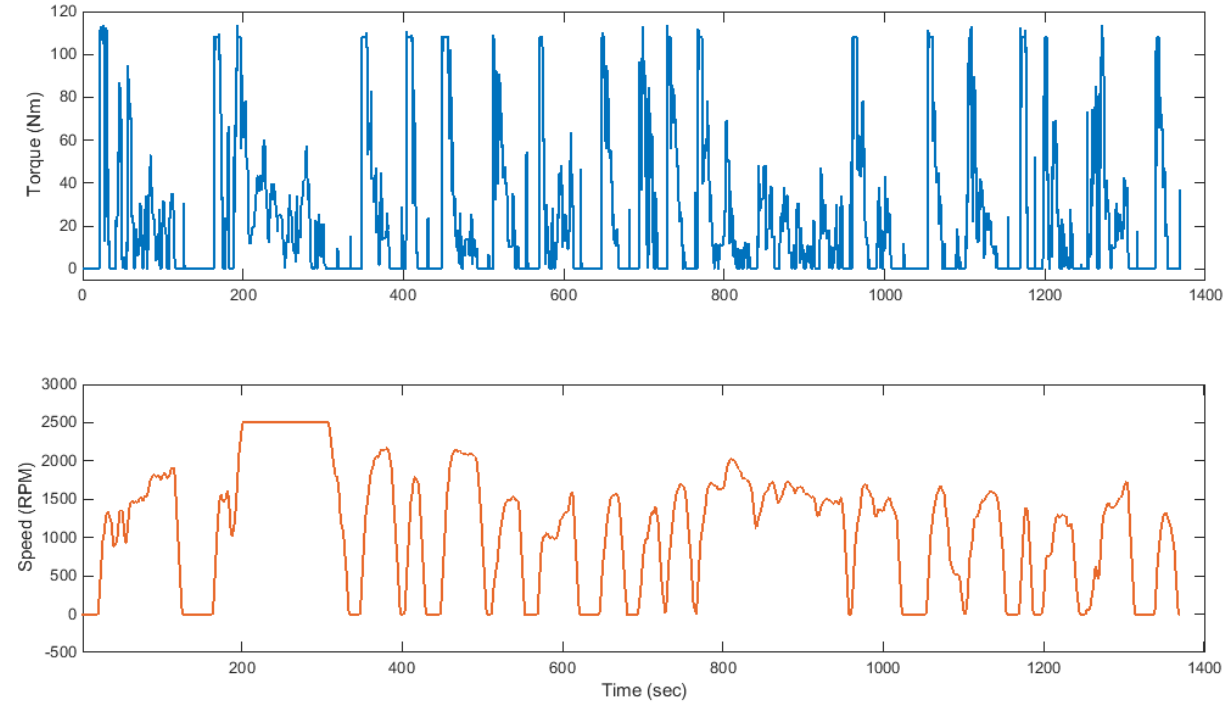

Figure 5.8: Torque and Speed Values Sent to Testbed

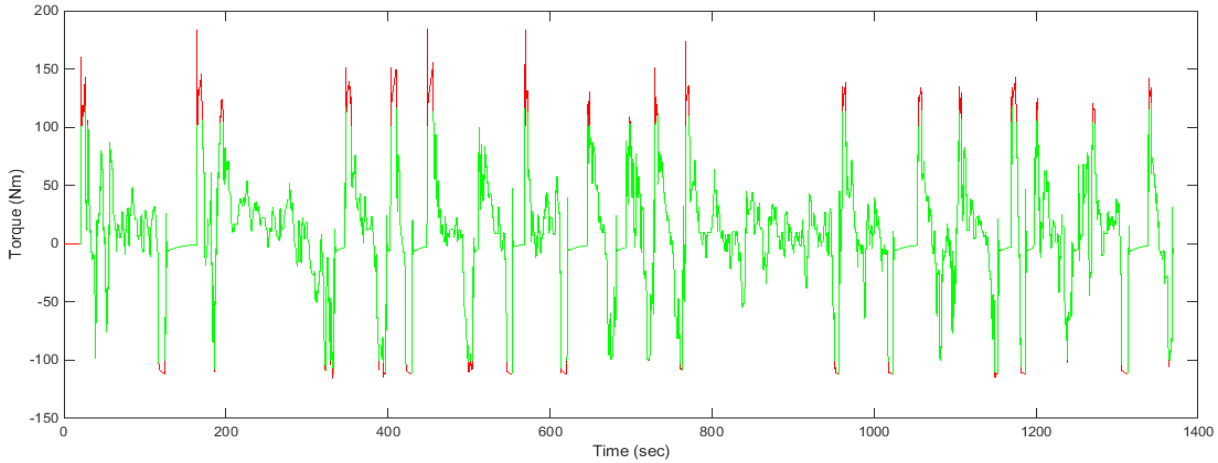

Figure 5.10: Illustrating Where on the Torque Profile the Current Limit is Exceeded

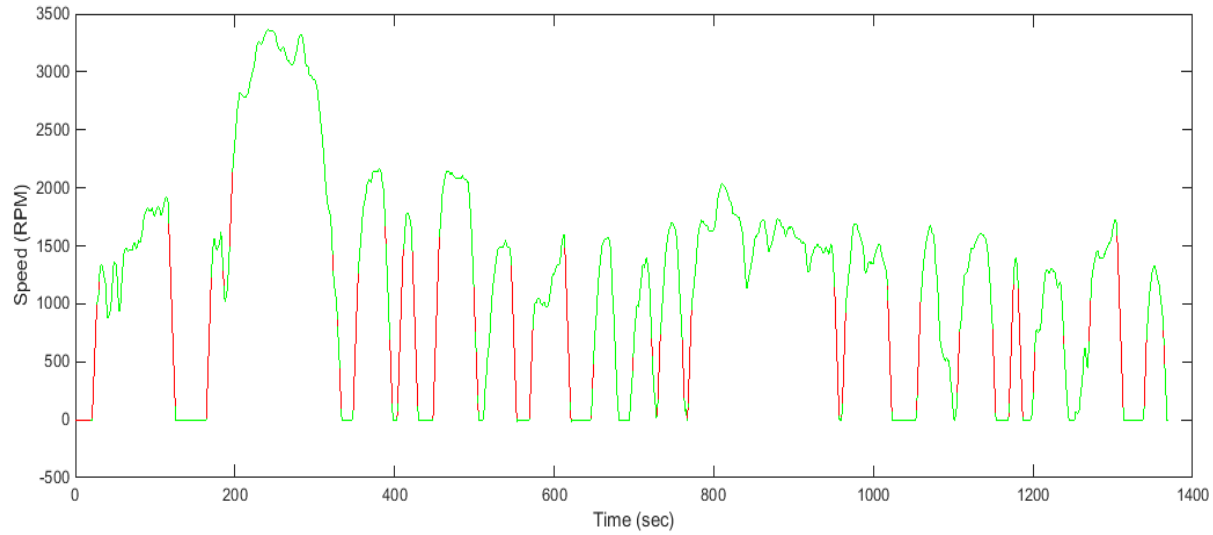

Figure 5.11: Illustrating Where on the Speed Profile the Current Limit is Exceeded

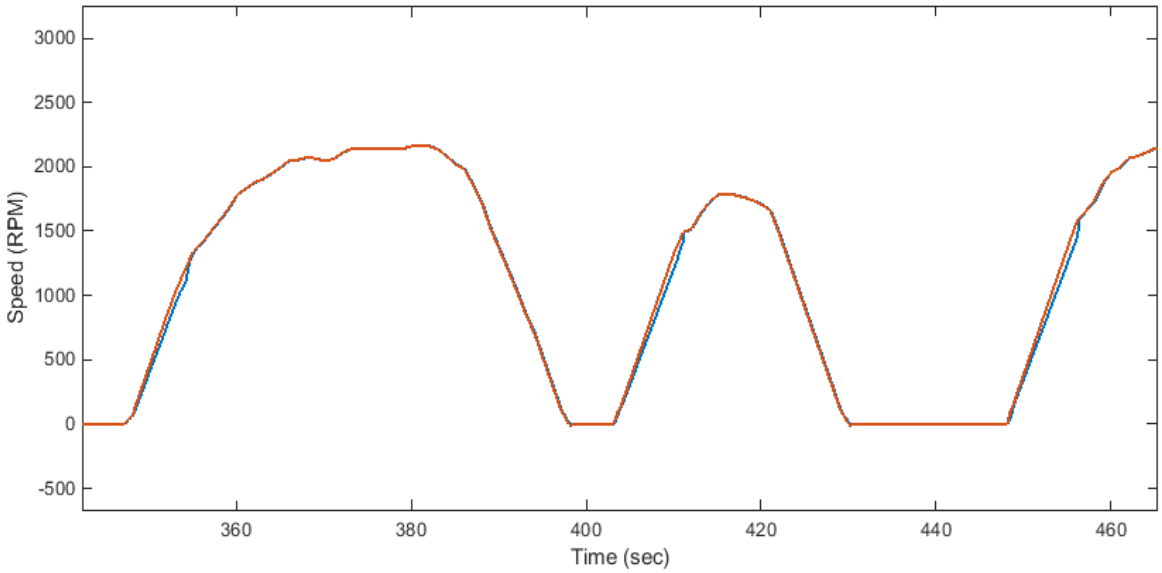

Figure 5.13: Driving Schedule Motor Speed (Orange) and Achieved Motor Speed (Blue)

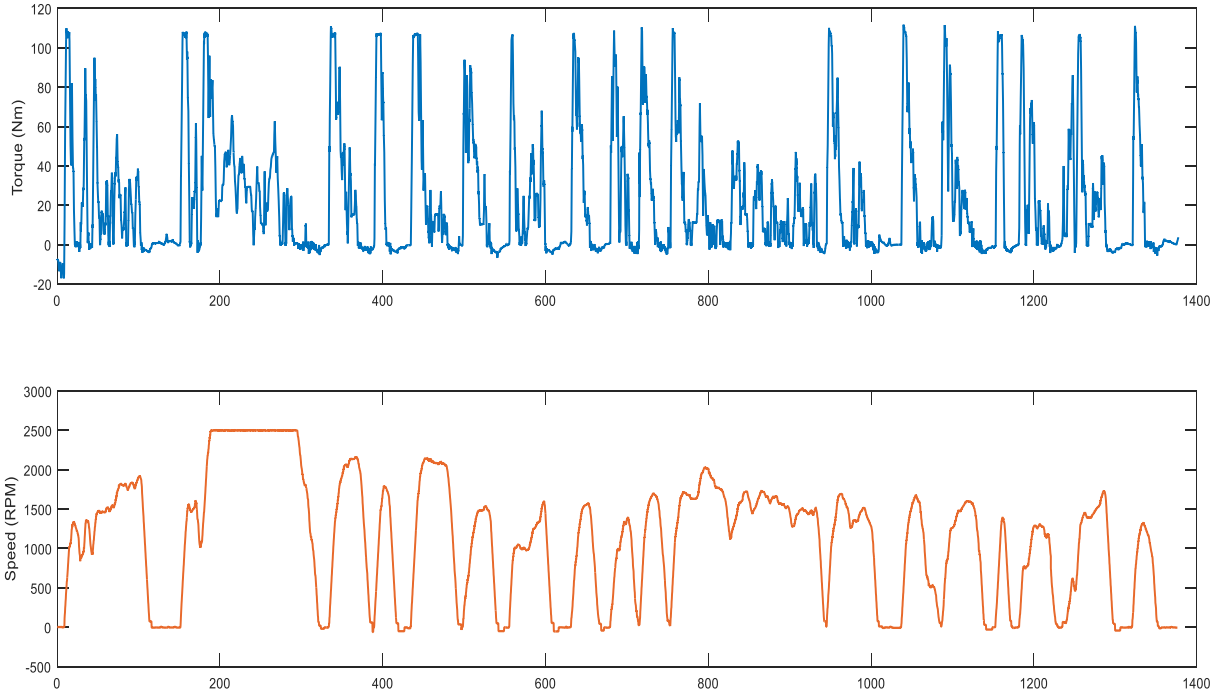

Figure 5.12: Averaged Torque and Speed Measurement Waveforms for the UDDS

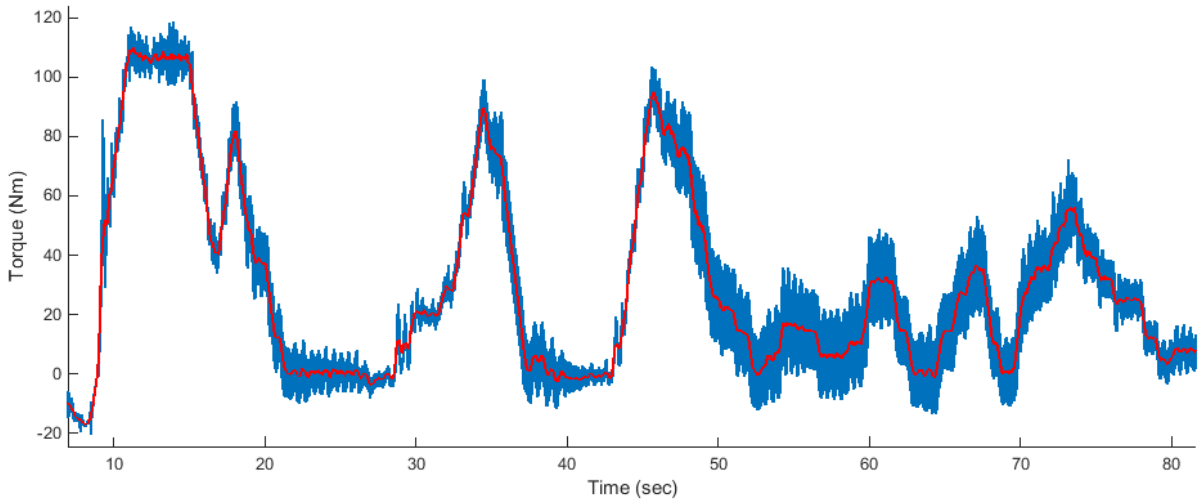

Figure 5.14: Comparison of Raw Torque Waveform (Blue) and Averaged Torque Waveform (Red)

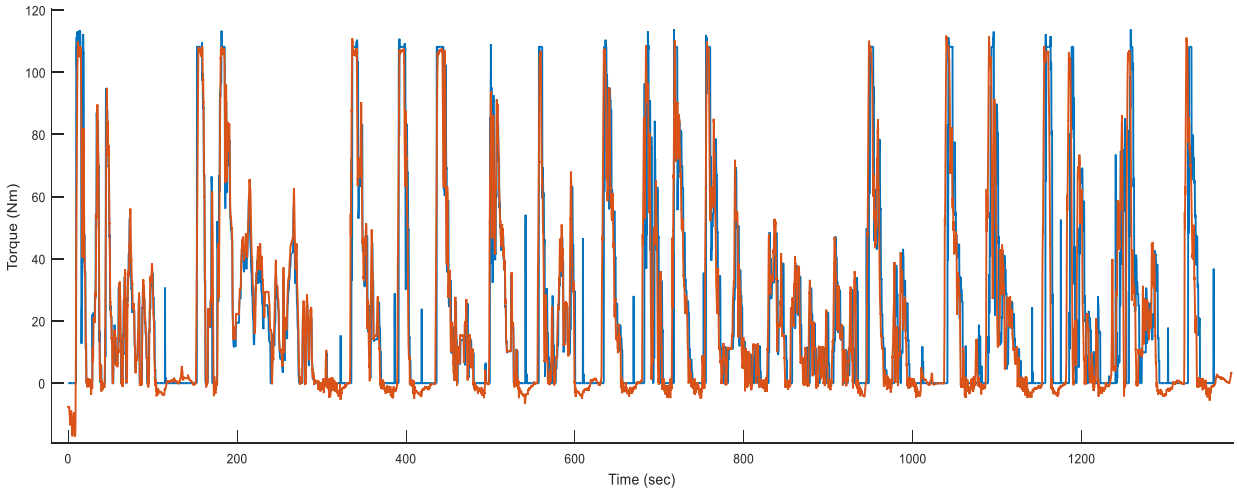

Figure 5.15: Simulated Torque (Blue) and Measured Torque (Orange) for the UDDS

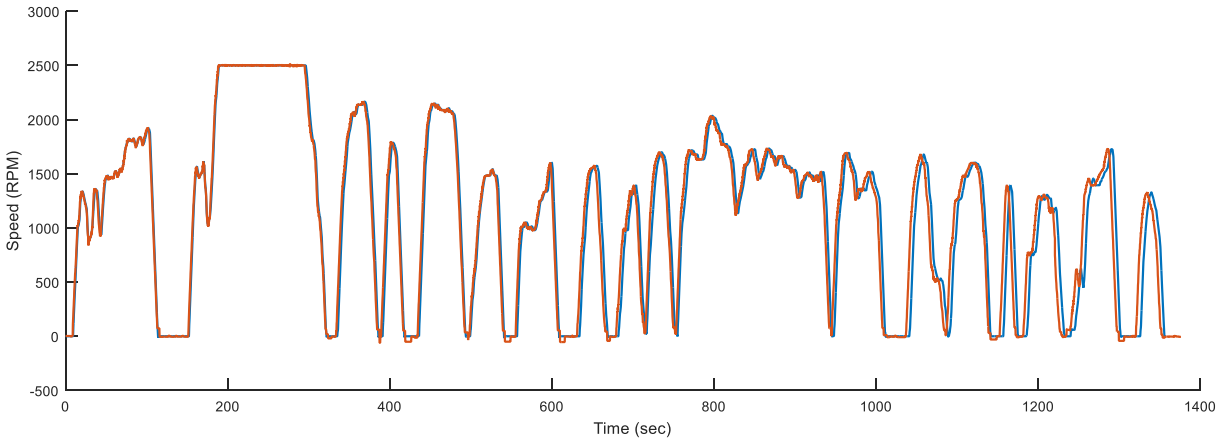

Figure 5.16: Simulated Motor Speed (Blue) and Measured Motor Speed (Orange) for the UDDS

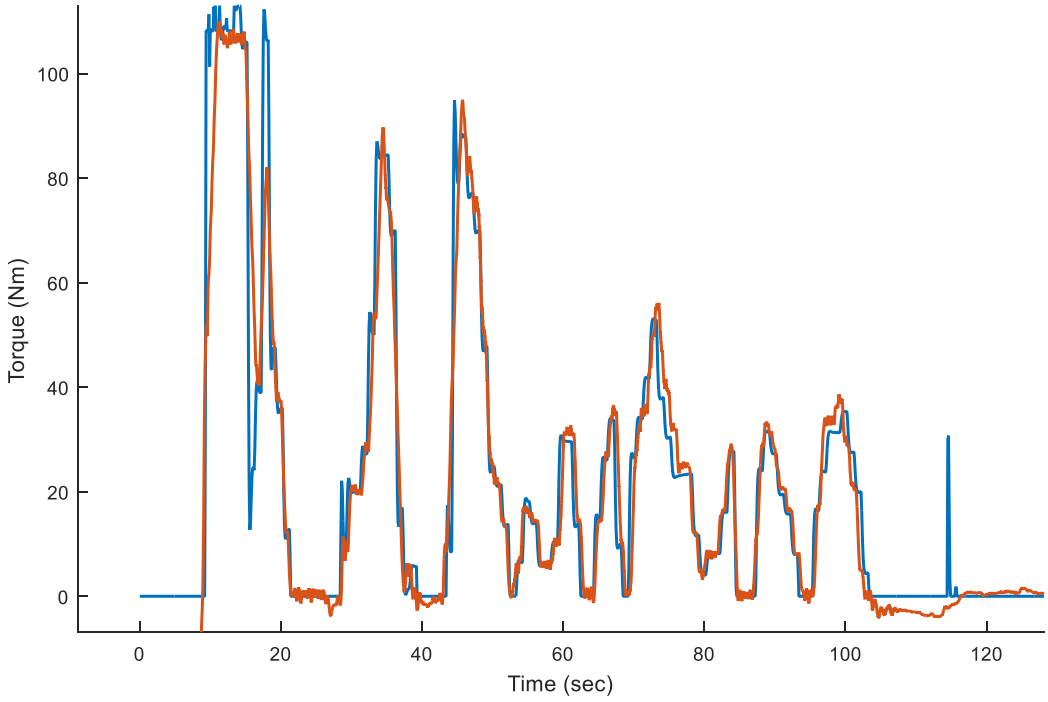

Figure 5.17: Close up - Simulated Torque (Blue) and Measured Torque (Orange) for the UDDS

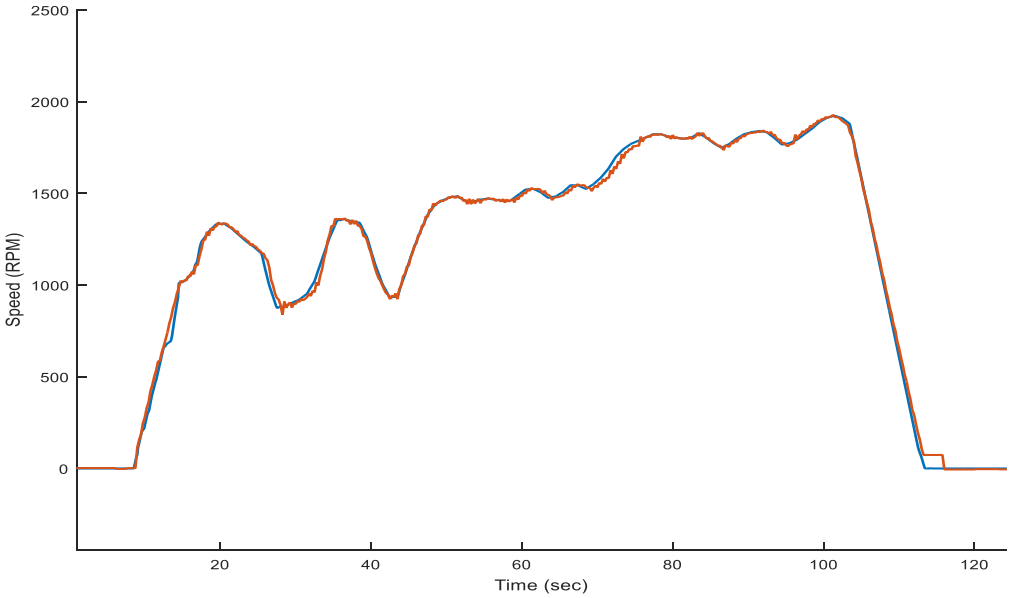

Figure 5.18: Close up - Simulated Motor Speed (Blue) and Measured Motor Speed (Orange) for the UDDS

As mentioned in Section 4.3, in addition to capturing the measured speed and torque of the propulsion machine, the testbed DAQ also captures 145 other signals from voltage, current, temperature, and flowmeters set up in the testbed. The output waveforms of the UUT were measured by current transformers and isolated voltage probes then recorded by the DAQ. The same sensor measurements are also input to the Yokagawa Power Analyzer, which calculated the RMS values for the signals. The Testbed Manual in Appendix A includes a table of all the DAQ measurements. The RMS phase current and voltage waveforms are shown in Fig. 5.19. Due to the properties of the electric motor, the RMS current waveform resembles the torque waveform (Fig. 5.17) and the RMS voltage resembles the speed waveform (Fig. 5.18).

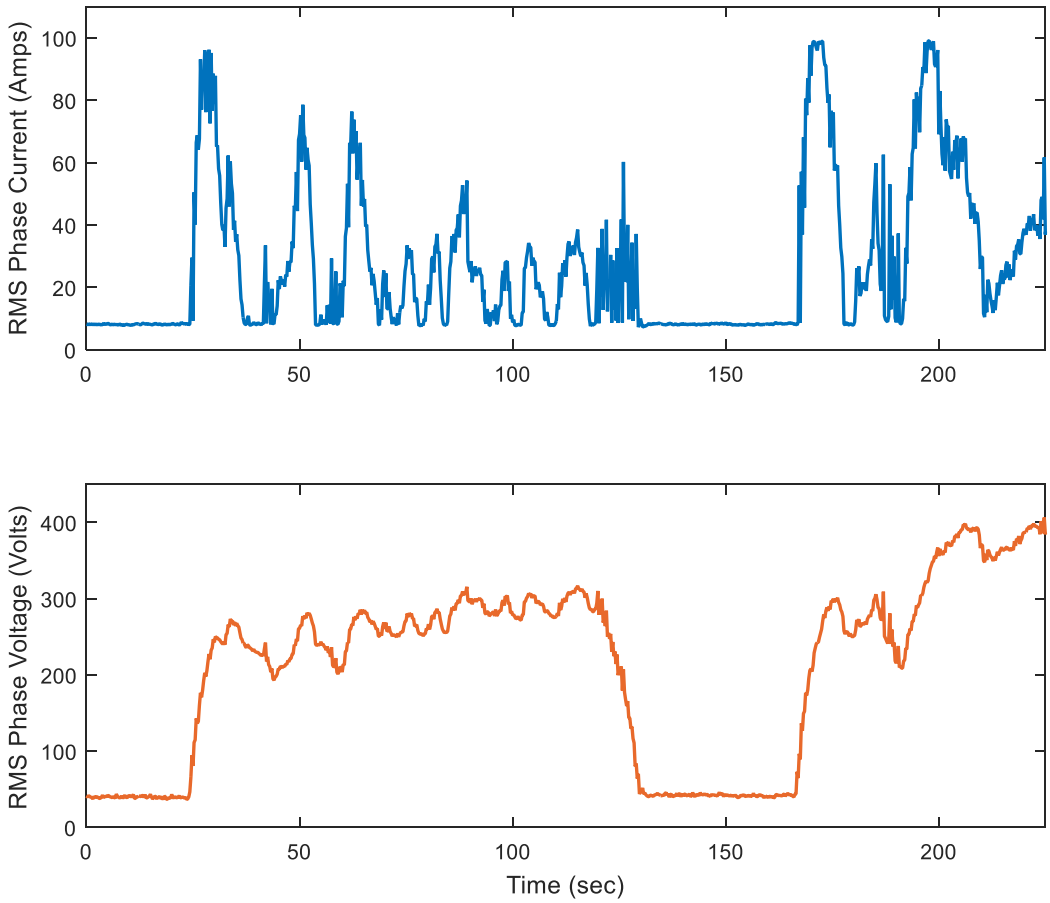

Figure 5.19: RMS Phase Current and RMS Phase Voltage for the UDDS

5.4 Conclusions

This chapter presents an overview of the testing capabilities, EPA driving schedules for analysis of driver behavior, the maximum acceleration limits of the testbed, and lastly the results from the proof of concept testing. The data presented shows that the simulation output was consistent with the results from ADVISOR, and that the testbed system can successfully control the speed and torque of the propulsion motor to emulate the behavior of a vehicle in "real-world" conditions. The software current limiting feature of the testbed, as described in Section 3.7.1, successfully limited the RMS current to the user defined limit of 100 A in the previous test, as illustrated by the current waveform in Fig. 5.19. Ch. 6 will present conclusions for this research.

5.5 References

[1] "Dynamometer Drive Schedules | US EPA", US EPA, 2017. [Online]. Available: https://www.epa.gov/vehicle-and-fuel-emissions-testing/dynamometer-drive-schedules. [Accessed: 10- Oct- 2017].

[2] H. Kim et al., "SiC-MOSFET composite boost converter with 22 kW/L power density for electric vehicle application," *2017 IEEE Applied Power Electronics Conference and Exposition (APEC)*, Tampa, FL, 2017, pp. 134-141.

[3] N. Mohan, T. Undeland and W. Robbins, Power Electronics Converters, Applications, and Design, 3rd ed. New York, N.Y: John Wiley & Sons, Inc, 1995, p. Chapters 12, 13, 14, 15.

[4] Ion Ilina, "Experimental Determination of Moment to Inertia and Mechanical Losses vs. Speed, in Electrical Machines", ATEE, Bucharest, May 12-14, 2011.

[5] Warachart Sae-kok, Pichit Lumyong, "Characteristics Evaluation of 3 Phase induction Motors Based on an Acceleration Method with increasing Moment of Inertia Technique", *SDEMPED 2003*, 24 Aug.

CHAPTER 6
CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

6.1 Conclusions

The benefits to EVs and HEVs such as reduced dependence on non-renewable resources, minimization of carbon footprint, efficiency, and responsiveness have been major factors in the resurgence of the EV and HEV market as well as industry and academic research [1-2]. There are now more than 20 different EV and HEV models for sale and more than half a million EVs have been sold since their reintroduction into the market in 2012 and December of 2016 [3]. To test the next generation of EV and HEV powertrains, the 100kW dynamometer at the University of Arkansas was constructed. The testbed is augmented with an OPAL-RT real-time simulator that generates realistic torque and speed profiles for any user-defined vehicle.

Chapter 2 discussed the major vehicle components and how to mathematically model them. In particular, the forces that act on the vehicle body while it is in motion were derived from Newton's 2nd law and extended to include the forces associated with wind resistance, rolling resistance and an inclined or declined road surface. Four different types of tire models ranging from empirical to theoretical were introduced, and the empirical model was chosen because of its accuracy and suitability for HIL simulation.

The two most common types of propulsion motors for vehicle applications, induction machines and permanent magnet synchronous machines, were then presented with their equivalent circuits and a discussion of the differences in construction and operation. Chapter 2 concluded with mathematical and circuit modeling of batteries.

Chapter 3 provided a block-by-block walkthrough of the MATLAB/SIMULINK^{TM} vehicle simulation and discussed the implementations of the mathematical models presented in

Chapter 2 as well as any assumptions made for making the model suitable for real-time simulation. The walkthrough followed the typical flow of energy in an EV or HEV by starting at the battery and progressing through the powertrain to the dc to dc converter, electric motor, then finally the power split device. Then, the physical attributes of the vehicle and the tire-to-road interaction are accounted for by using the vehicle body and tire model described in Chapter 2. Lastly, the ramp limiting and Simulation I/O from EPA driving schedule and Modbus TCP/IP protocol communication with the testbed control system were discussed.

The testbed hardware, load motor drive, control/data acquisition software, OPAL-RT real time simulator, and UUT communication protocols were covered in Chapter 4. The system overview highlighted the individual hardware components and discussed the limits of operation as well as the four areas of operation of the testbed. The direct torque control strategy implemented on the load drive was then presented. The OPAL-RT simulator section covered the simulation capabilities of the system and showed how to set up the Modbus communication with the testbed control interface. The final section of this chapter described the communication hardware and protocol the testbed is using to communicate with the UUT.

The simulation can also be used as a standalone tool for analyzing driver behavior. A better understanding of the operating point densities and frequencies in real world scenarios can lead to powertrain component designs that are better suited for vehicle applications [4]. To validate the simulation results, the torque and speed profiles generated for a Toyota Prius were compared to the output from NREL's ADVISOR. The data showed a close correlation between the outputs of the two simulators. A Baldor H2 vector drive was used as an UUT for proof of concept testing. The measurement data from the testbed data acquisition system, shown in Section 5.3.1, very closely matched the simulation output values and showed that the testbed can

successfully emulate realistic driving scenarios. The measured RMS current and voltage waveforms showed that the software current limiting feature of the testbed can successfully limit the speed and torque commands sent to the testbed to meet the user defined limit.

6.2 Recommendations for Future Work

The work presented in this thesis could be continued with the following recommendations:

- A Matlab model of the physical dynamometer could be developed that would emulate the behavior of the machines and their feedback devices. This would allow for the existing vehicle simulation to be used in a Controller Hardware in the Loop (C-HIL) configuration.
- The OPAL-RT real-time simulator has the capability of generating PWM output signals and a tool or tutorial could be developed to convert user MATLAB/SIMULINK control system models to an RT-Lab model. The proposed control strategy could then be tested with the inverter before being implemented in hardware on the control board.

The integration of the above recommendations would provide additional intermediate testing and troubleshooting opportunities for the inverter and controller. It would also mitigate some of the risk at the integration point if either technology has not been proven reliable in “realistic” driving conditions.

6.3 References

[1] A. Boulanger, A. Chu, S. Maxx and D. Waltz, "Vehicle Electrification: Status and Issues", *in Proceedings of the IEEE* Vol. 99, No. 6, IEEE, 2011.

[2] "Electric Vehicle Benefits | Department of Energy", Energy.gov, 2018. [Online]. Available: https://www.energy.gov/eere/electricvehicles/electric-vehicle-benefits. [Accessed: 11- Mar- 2018].

[3] J. Bhuiyan, "There have now been over 540,000 electric vehicles sold in the U.S.", Recode, 2017. [Online]. Available: https://www.recode.net/2016/12/21/14041112/electric-vehicles-report-2016. [Accessed: 09- Oct- 2017].

[4] H. Kim et al., "SiC-MOSFET composite boost converter with 22 kW/L power density for electric vehicle application," 2017 IEEE Applied Power Electronics Conference and Exposition (APEC), Tampa, FL, 2017, pp. 134-141.

APPENDIX A : TESTBED USER MANUAL

Introduction

POETS researchers are developing enabling technologies that must be tested in order to produce valuable experimental validation data. The traction inverter is an important subsystem in powertrains for hybrid and electric vehicles so it was selected as a platform for demonstrating a variety of enabling technologies reaching readiness levels of 4 to 6 (e.g., 3D power modules, microchannel-based thermal management solutions, soft-switching inverter topologies, capacitor technologies using new materials).

Generation 2 Testbed 2 evolved from a 100-hp dynamometer at the National Center for Reliable Electric Power Transmission (NCREPT) of the University of Arkansas. Features added to this dynamometer are the following:

- Generation of realistic torque and speed profiles given the physical dimensions of a hybrid or electric vehicle,
- Hardware-in-the-loop (HIL) capabilities by interfacing with OPAL-RT real-time simulator,
- LabVIEW® graphical user interface (GUI) to change parameters in real time,
- Limiting the inverter current ramp rates when the torque and/or speed commands for a given driving schedule results in exceeding the current limits of the unit under test (UUT).

The EPA driving schedules are driver models that represent typical driving characteristics in a certain environment (in city or on highway) and are typically used for state mandated emissions testing and fuel economy tests. The Urban Dynamometer Driving Schedule (UDDS) is

indicative of an urban environment with many starts and stops at relatively low speeds for a distance of 7.5 miles. In contrast, the Highway Fuel Economy Test (HWFET) is representative of high speeds, relatively low small fluctuations in speed, over a distance of about 10 miles. With the “plug and play” design of the testbed and the standardized EPA driving schedules a direct comparison of efficiency and performance can be made for the overall powertrain system between its current state of the art and that incorporating POETS enabling technologies or novel control strategies.

The OPAL-RT real-time simulator runs the vehicle simulation in tandem with the testbed operation. The vehicle simulation takes the physical attributes of the vehicle (surface area, weight, wheel size, etc.) and the desired velocity input (from the driving schedule) to calculate the torque required for the given vehicle to achieve the requested speed. The interaction between the road and the vehicle tire is also modeled by the simulation, which adds the capability to test advanced inverter control strategies that incorporate inclement weather traction control algorithms. The speed and torque commands are then sent to the LabVIEW control interface.

The electromagnetic torque produced by the propulsion motor (i.e., a permanent magnet synchronous motor) is proportional to the motor phase current. The acceleration of the simulated vehicle is therefore limited by the current carrying capability of the UUT. Speed and torque limiting is implemented in software to ensure that the limits of the testbed hardware and UUT are not exceeded. The user can set the maximum UUT current limit in the user interface. The simulation will then limit the ramp rates of the speed and torque waveforms and regulate the maximum torque to levels so that the required current is acceptable. This also protects the UUT if no current limiting is implemented and prevents the user from damaging the test setup or UUT if they are run at a driving schedule that requires more power than the UUT can provide. This

feature speeds up the testing process by eliminating the need for detailed driving schedule design.

This manual guide the user through the operating modes and data acquisition system setup of the POETS Testbed 2 dynamometer in the National Center for Reliable Electric Power Transmission (NCREPT) test facility at the University of Arkansas. The testbed control system has three main operating modes:

1. Manual input of speed and torque commands
2. Scheduled speed and torque commands (stair step-like waveform)
3. Simulation generated speed and torque commands (real-world conditions)

The list of operating modes also correspond to the order in which these operating modes should be used when validating a prototype inverter on the testbed. Starting with the manual input of commands allows the speed and torque values to be incrementally increased to determine if any instabilities exist within the unit under test's operating range. Scheduling commands via .csv (comma separated value) file allow for a more rigorous validation schedule for the inverter. Lastly using the simulation to calculate the speed and torque commands provides the most realistic test conditions by incorporating EPA driving schedules.

This manual will begin by discussing the topology of the testbed system. An overview of the LabVIEW^{TM} Graphical User Interface (GUI) will then be presented. The NCREPT facilities emergency stops and safety policies will follow. The operating modes of the testbed will then be presented with step by step instructions on how to set up the test in the LabVIEW graphical user interface. Lastly, methods for importing the data from the testbed into Matlab will be presented.

Testbed Topology

Testbed 2 has a motor generator pair with one acting as the propulsion motor while the other serves as the load for the system. Figure A.1 shows a block diagram of the system. The propulsion motor will be controlled by the unit under test (UUT) and the load machine will be controlled by an ABB ACS800 series drive. The orange box in Fig. A.1 encircles the typical EV powertrain components, almost all of which could be tested with the current testbed setup. NCREPT does not currently have the appropriate safety enclosures to test large battery banks. However, an 800kW DC power supply is available for providing power to the powertrain components. The plug and play style nature of the system and the vehicle powertrain allows for direct comparisons between powertrain components.

The LabVIEW GUI uses serial communication to send speed and torque commands to the UUT and the load ABB drive. It also records and displays the measurement data from the testbeds

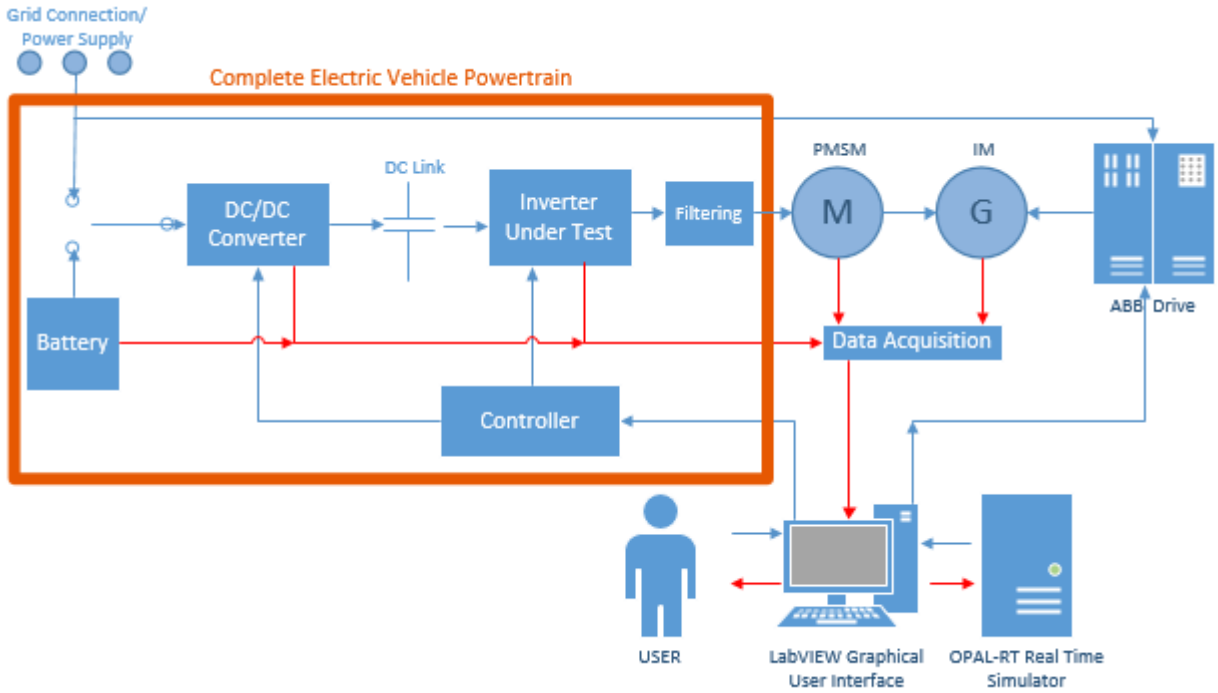

Figure A.1: Testbed 2 Block Diagram

various probes and sensors. When the speed and torque commands are being generated by the simulation the OPAL-RT system will communicate the speed and torque commands to the LabVIEW GUI then on to the inverters. Measurement feedback from the testbed data acquisition system is also available in the simulation.

Power to the UUT is routed through NCREPT circuit breaker F5 and F11. They are operated with a lockout/tag out system and require the facility test engineer to be present when energized. The breakers are also tied in to the facility emergency stops which are located at each exit point.

LabVIEW Interface Overview

Figure A.2 shows the LabVIEW graphical user interface on the "Dynamometer Main" screen where the major dynamometer functions are performed. This screen has controls for the UUT, load machine, DC supply, and data collection. It also has indicators for some variables like the commanded and measured speed/torque values. The large graph in the center of the interface displays the RMS voltage, RMS current, speed, torque, and multiple thermocouple temperatures as indicated by the legend on the right side of the interface. Data collection is initiated by enabling "Save Data" in the Data collection section of the main interface.

The "Auto Configuration" tab has the control information for scheduling the speed and torque commands. The file path to the .csv file is input here along with the start and stop indices for the spreadsheet. The "ABB Configuration" tab is used to set parameters for the drive limits of the ABB ASC800 series load drive.

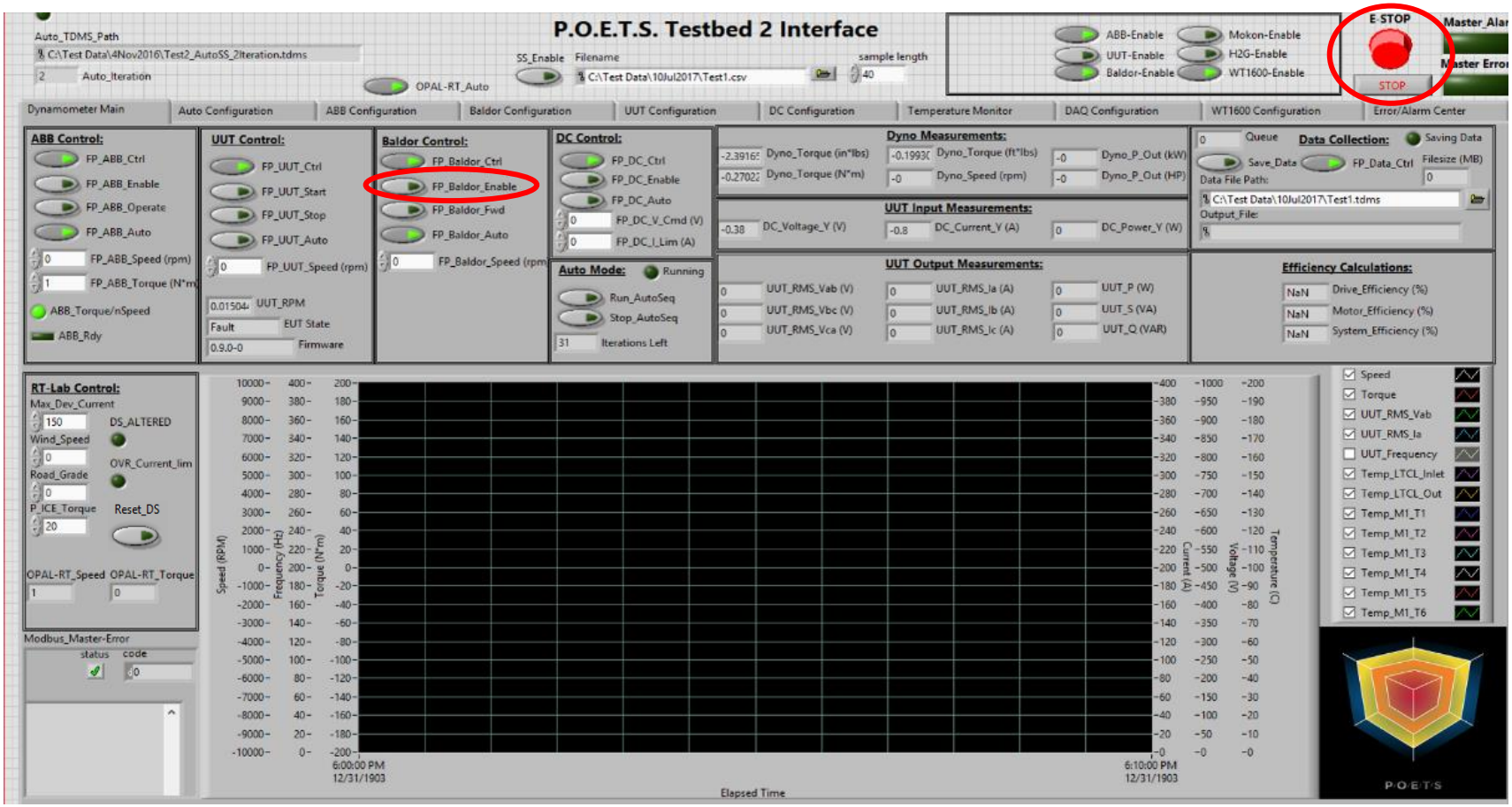

Figure A.2: Testbed 2 User Interface

It also displays some monitored signals available from the drive such as motor power, dc bus voltage, and drive temp in addition to fault and status indicators. The “UUT Configuration” tab will be where any custom indicators, control variables, or settings for the prototype inverter will be found. The “DC Configuration” screen shows the settings available for the DC power supply. “Temperature Monitor” section plots thermocouple temperature measurements placed throughout the system as well as the temperatures of the coolant loops. The “DAQ Configuration” screen controls the channel attenuation and signal multiplication for some of the signals measured by the data acquisition system. The “WT1600 Configuration” tab displays the measured data from the Yokagowa WT1600 Power Analyzer. Lastly, the “Error/Alarm Center” tab displays any error codes that may occur in the test equipment.

Emergency Stops and General Safety Procedures

There are multiple ways to stop the testbed in the event of a fault or other dangerous scenario. The NCREPT test facility has E-Stops located at each exit of the building that will disconnect power from the UUT. If evacuation is necessary, push the E-stop on the way to the predetermined meeting point. There are two other ways to stop operation in non-emergency scenarios where evacuation may not be necessary.

1. GUI emergency stop - The red “E-STOP” button in the upper right hand corner of the interface (circled red in Fig. A.2) will command the testbed to zero speed as fast as possible. This is best in the event that something gets caught in the rotating machines and an immediate stop is necessary. This is not recommended for normal shutdown of the testbed since this very fast deceleration may cause damage to the UUT or testbed components.

2. Disabling propulsion motor drive – This method cuts the power to the propulsion machine and allows it to coast to zero speed. This is done by disabling the unit under test control switch (circled in red in Fig. A.2). Use this method when stopping the testbed in non-emergency scenarios.

Disabling the propulsion drive is the preferred non-emergency shutdown method. This can be used to interrupt a test with little or no adverse effects.

If any of part of the test setup needs to be altered the breakers powering the system must be locked out before anyone can cross behind the baracades. The bay should be cleared of personell during any high power test. The computer next to the dynamometer can be used to control the UUT components, but a remote desktop connection has be set up so that it the test can be run from the control room.

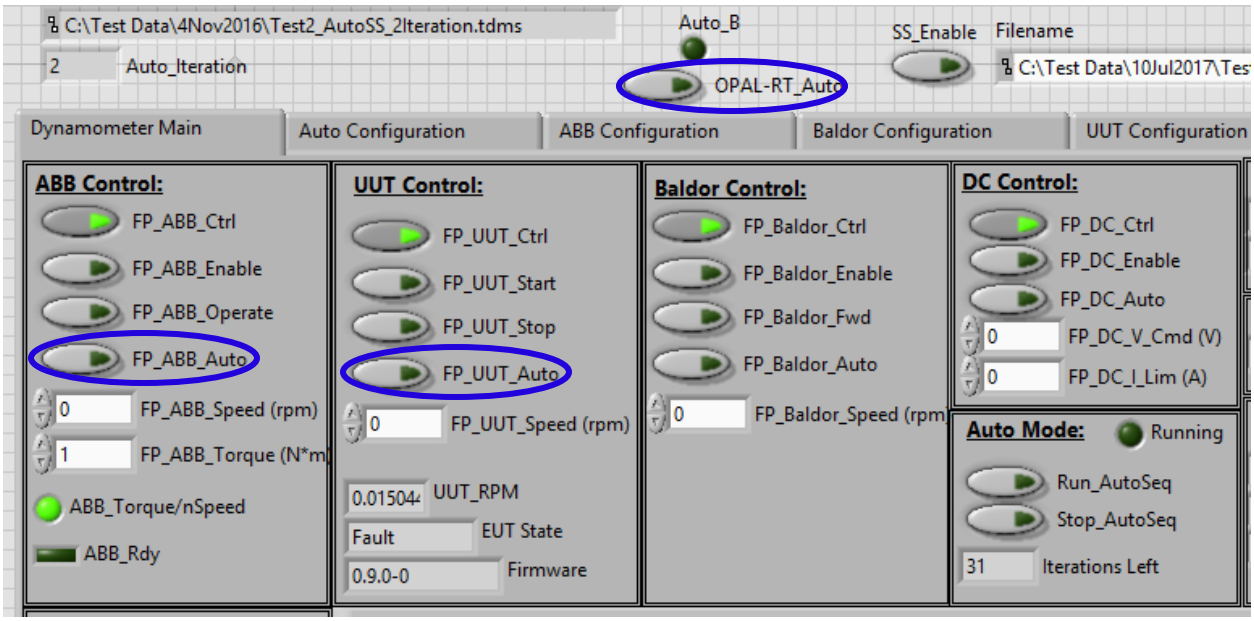

Figure A.3: Setup for Manual Input of Speed and Torque Commands

Testbed Modes of Operation

Operating the Testbed with Manual Input of Speed and Torque Values

1. Set control signals for manual mode
    a. Run the LabVIEW code
    b. Ensure that “OPAL-RT_Auto”, “FP_ABB_Auto”, and ”FP_UUT_Auto” (circled in blue in Fig. A.3) are DISABLED.
2. Energize the system
    a. Make sure the NCREPT Test Engineer is present and the bay cleared of personnel
    b. Turn on the breaker for the ABB load drive
    c. Energize the UUT by enabling breakers F5 and F11 from the NCREPT control room computer
3. Enable the ABB ACS800 load drive
    a. ENABLE “FP_ABB_Enable” and wait for the “ABB_Rdy” indicator to light up
    b. ENABLE “FP_ABB_Operate”

4. Enable the UUT
    a. For the testbed system validation, a Baldor H2 Vector drive was used, its control signals are located in the “Baldor Control” section of the interface.
    b. ENABLE “FP_Baldor_Enable” and “FP_Baldor_Forward”
    c. Depending on the control setup, the prototype being tested will likely have different enable signals than the Baldor drive. However, when the control interface is developed with LabVIEW, the appropriate signal switches should be placed in the “UUT_Control” section of the “Dynamometer Main” screen.
5. Input speed and torque commands
    a. Input the commanded speed value in the “FP_Baldor_Speed” control box located in the UUT Control section
    b. Input the commanded torque value in the “FP_ABB_Torque” control box located in the ABB Control section
6. Disable drives when testing is complete
    a. DISABLE UUT Enable variables (“FP_Baldor_Enable” and “FP_Baldor_Forward” for system validation scenario)
    b. DISABLE “FP_ABB_Operate” and then “FP_ABB_Enable”
7. De-energize and rack out breakers F5 and F11

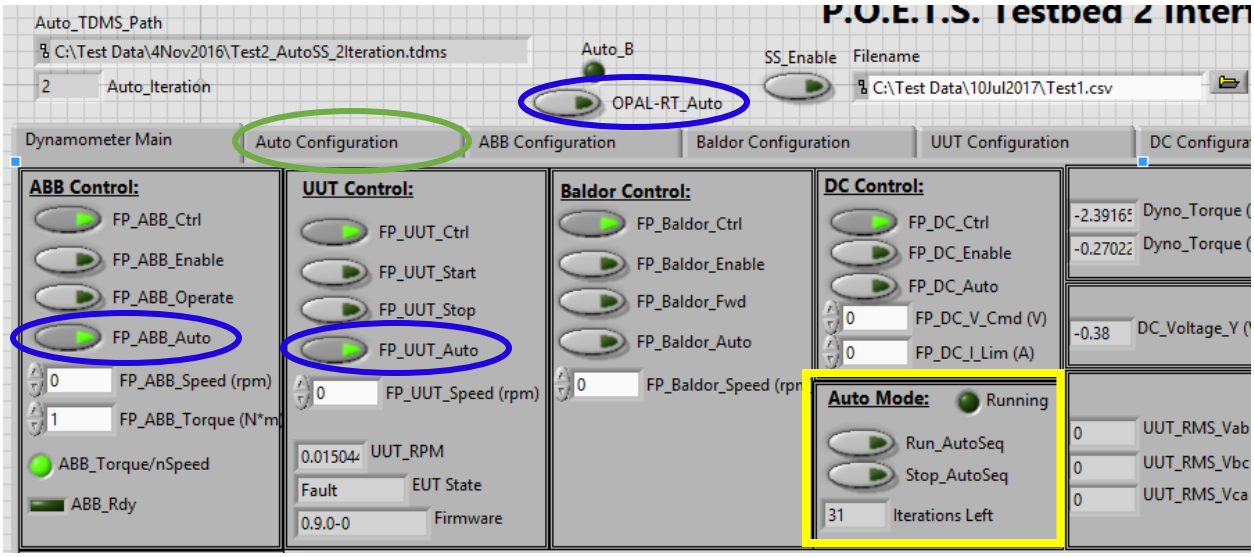

Figure A.4: Testbed Auto-Run Configuration

Operating the Testbed by Scheduling Speed and Torque Values

1. Set control signals for auto input
    a. Run LabVIEW code
    b. ENABLE “FP_ABB_Auto” and “FP_UUT_Auto” (Circled in blue in Fig. A.4)
    c. DISABLE “OPAL-RT_Auto” (Circled in blue in Fig. A.4)
2. Select test schedule .csv file using the filepath control box in the Auto Configuration tab (circled in green in Fig. A.4)
    a. The .csv file must be structured like the following table

Table A.1: Example Test Schedule

Time (ms)	VDC (V)	Speed (RPM)	Torque	Intervals
20000	200	0	1	98
40000	200	500	1	0
30000	200	500	5	0
30000	200	500	10	0

    b. The Time column indicates the duration in ms that system will sit at the operating point described in the VDC, Speed, and Torque columns. The first entry in the Intervals column should represent the number of intervals for the entire test.

3. Energize the system
    a. Make sure the NCREPT Test Engineer is present and the bay cleared of personnel
    b. Turn on the breaker for the ABB load drive
    c. Energize the UUT by enabling breakers F5 and F11 from the NCREPT control room computer
4. Enable the ABB ACS800 load drive
    a. ENABLE “FP_ABB_Enable” and wait for the “ABB_Rdy” indicator to light up
    b. ENABLE “FP_ABB_Operate”
5. Enable the UUT
    a. For the testbed system validation, a Baldor H2 Vector drive was used, its control signals are located in the “Baldor Control” section of the interface.
    b. ENABLE “FP_Baldor_Enable” and “FP_Baldor_Forward”
    c. Depending on the control setup the prototype being tested will likely have different enable signals than the Baldor drive. However, when the control interface is developed with LabVIEW, the appropriate signal switches should be placed in the “UUT_Control” section of the “Dynamometer Main” screen.
6. Begin the auto sequence
    a. ENABLE “Run_AutoSeq” in the “Auto Mode” section of the interface (yellow box in Fig. A.4)
7. Disable drives when scheduled iterations are complete
    a. DISABLE UUT Enable variables
    b. DISABLE “FP_ABB_Operate” and then “FP_ABB_Enable”
8. De-energize and rack out breakers F5 and F11

Operating the Testbed with the Simulated Vehicle Model

The general process flow is shown in Fig. A.5 below. The following section goes into depth on the appropriate procedure to operate the testbed with the real-time vehicle model in a safe manner. Before you use the NCREPT facilities you must contact the test engineer at least two weeks before the desired test date and submit a test plan for review. In addition, a test schedule should be completed describing the schedule of events for the allotted test time. An example test schedule will be shown in a later section.

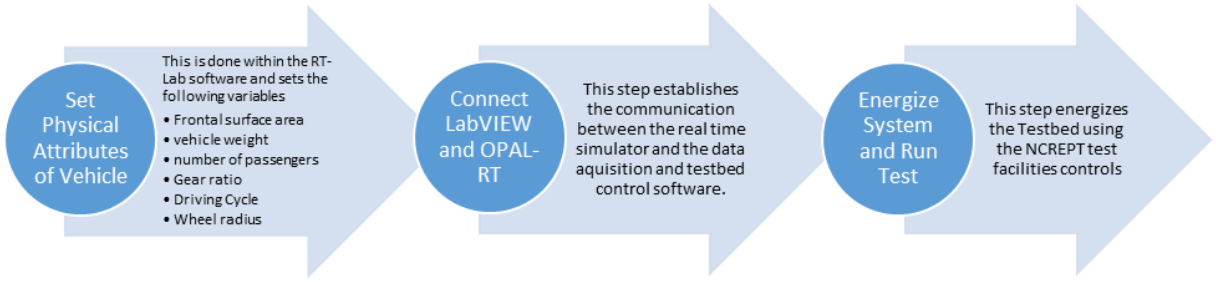

Figure A.5: General Test Procedure Overview

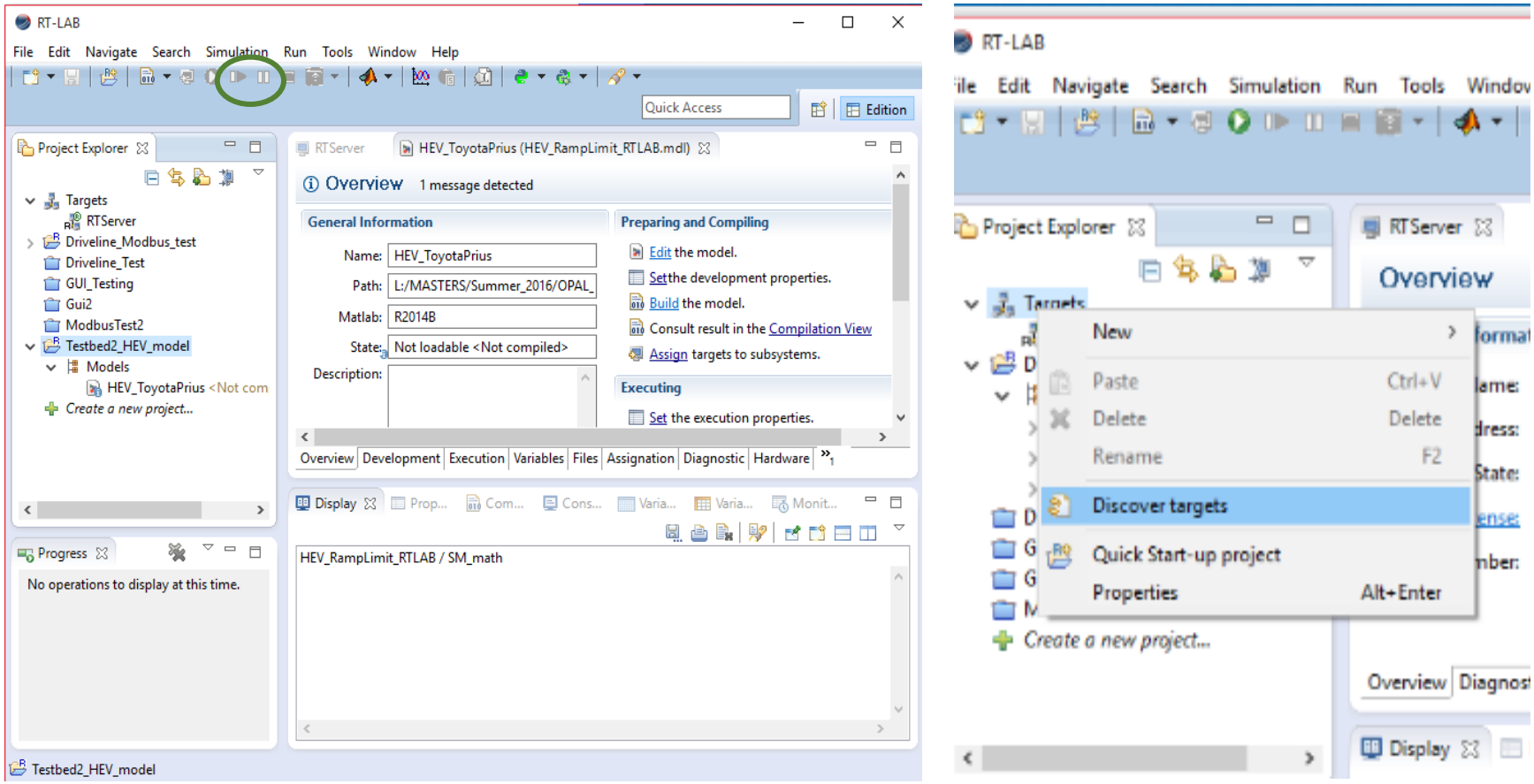

Figure A.6: Opal-RT Interface, Discovering Targets

Use RT-Lab real time simulation software to set the vehicle attributes.

a. Once RT-Lab is open, make sure the target is shown under the “targets” list on the left hand side of the interface (as shown in Fig. A.6).
    i. If there is no target present, simply right click on the “Targets” label and click “discover targets”
    ii. If the “discover targets” does not work either close RT-Lab and reopen it or reset the target manually (push power button twice).

In the project viewer double click the vehicle model to open it - “Testbed2_HEV_model” project in this case.

b. If you wish to change the vehicle parameters, right click on the model you wish to run, in this case it is the “HEV_ToyotaPrius”, and select “edit”
    i. This will open MATLAB/SIMULINK™ and display the model.
c. In Simulink click “File” then “model properties” then “model properties” again. Navigate to the “Callbacks” tab shown in Fig. A.7.
d. Then navigate to the “PostLoadFcn” section in the left pane. Shown in Fig. A.8
    i. This is where the vehicle attribute parameters are located. These parameters refer to the physical aspects of the vehicle and cannot be changed mid simulation. For example: vehicle weight, frontal surface area, wheel radius, and gear ratio are not going to change mid driving schedule in a real world scenario. These parameters shown are for a 2014 Toyota Prius.
e. Next select the driving schedule you wish to run from the available list or load your own custom schedule as shown in Fig. A.8 (Discussed in greater detail in a later section).
f. Once the parameters and driving schedule are set for your vehicle, apply the changes, save the model and close Matlab Simulink

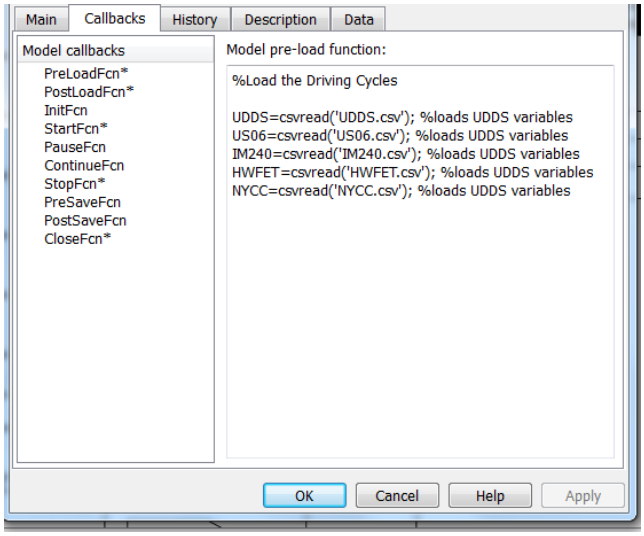

Figure A.7: Callbacks Section Where Vehicle Driving Schedules are Loaded

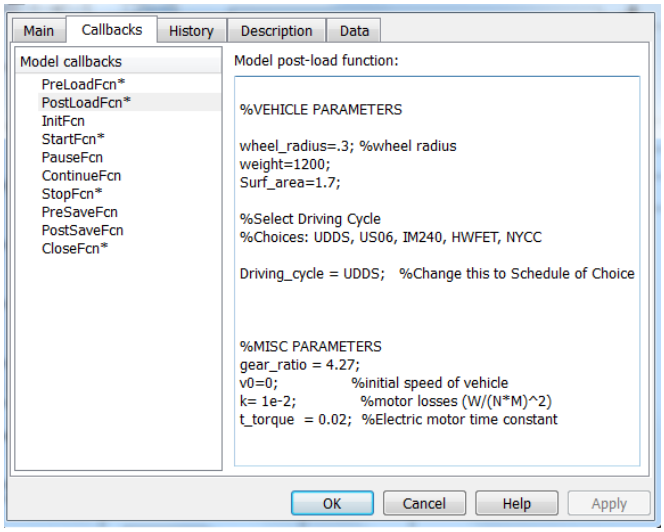

Figure A.8: PostLoadFcn Where Vehicle Parameters are Found

g. In the RT-Lab window, right click on the model in the project explorer window of RT-Lab and select “Simulation” > “Build” from the drop down menu. This will take some time if you have made changes to the model.

h. Once the simulation builds successfully, load the model by right clicking the model again and selecting “Simulation” > “Load”

i. Only one model can be built or loaded at a time on the OPAL-RT server so if anyone else is using it you may receive an error at build or load.

ii. It is important to note that all control of the Simulink file (start, pause, and reset) should be done from the RT-Lab window. If you attempt to use the run/stop/pause buttons in the Matlab Simulink window RT-Lab will no longer be able to control the simulation and you will have to reload it.

i. Once the model is loaded, execute the model and then pause it after it compiles and runs for 10 seconds by using the buttons circled in green in Fig. A.6.

i. The EPA driving schedules have about 20 seconds of 0 speed and 0 torque at the start to allow time for this step. Be sure to pause the simulation before reaching the non-zero values.

ii. This step starts the Modbus Communication and makes the system ready to connect with LabVIEW.

2. Connecting LabVIEW and OPAL-RT

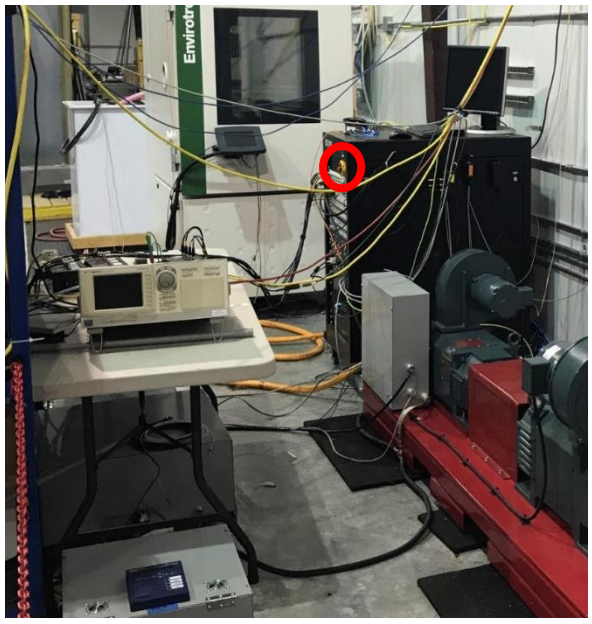

Figure A.9: Power Switch for Data Acquisition System

a. In order to connect to the data acquisition computer, you must first turn on the computer by flipping the large power switch located on the front of the black cabinet indicated by the red circle in Fig. A.9.
    i. You should never cross the red chain unless the system is de-energized and you have notified NCREPT personnel of what you are doing
b. Once the switch is on, you should be able to click on the remote desktop link on the desktop of the computer in the NCREPT control room and open the LabVIEW interface (Fig. A.10).
c. Navigate back to the RT-Lab and verify the program is paused as discussed in Step 1-j. If it is, execute the LabVIEW code by pressing the arrow on the upper left side of the tool bar (Green circle in Fig. A.10).
    i. Since the system is not yet energized you will not have control of the motors at this point in the procedure, this step is to ensure that the communication is established between the OPAL-RT server and LabVIEW
    ii. If there is an issue with the Modbus communication, a red "X" and a message will be visible in the lower left corner of the interface (Circled in blue in Fig. A.10). Reset the target by clicking the red square next to the play button in RT-Lab and repeat Step 1-i, Step 1-j and 2-c to try attempt the connection again.
d. Verify that there are no communication issues – look for green checkmark in lower left corner of the interface (circled in blue in Fig. A.10)

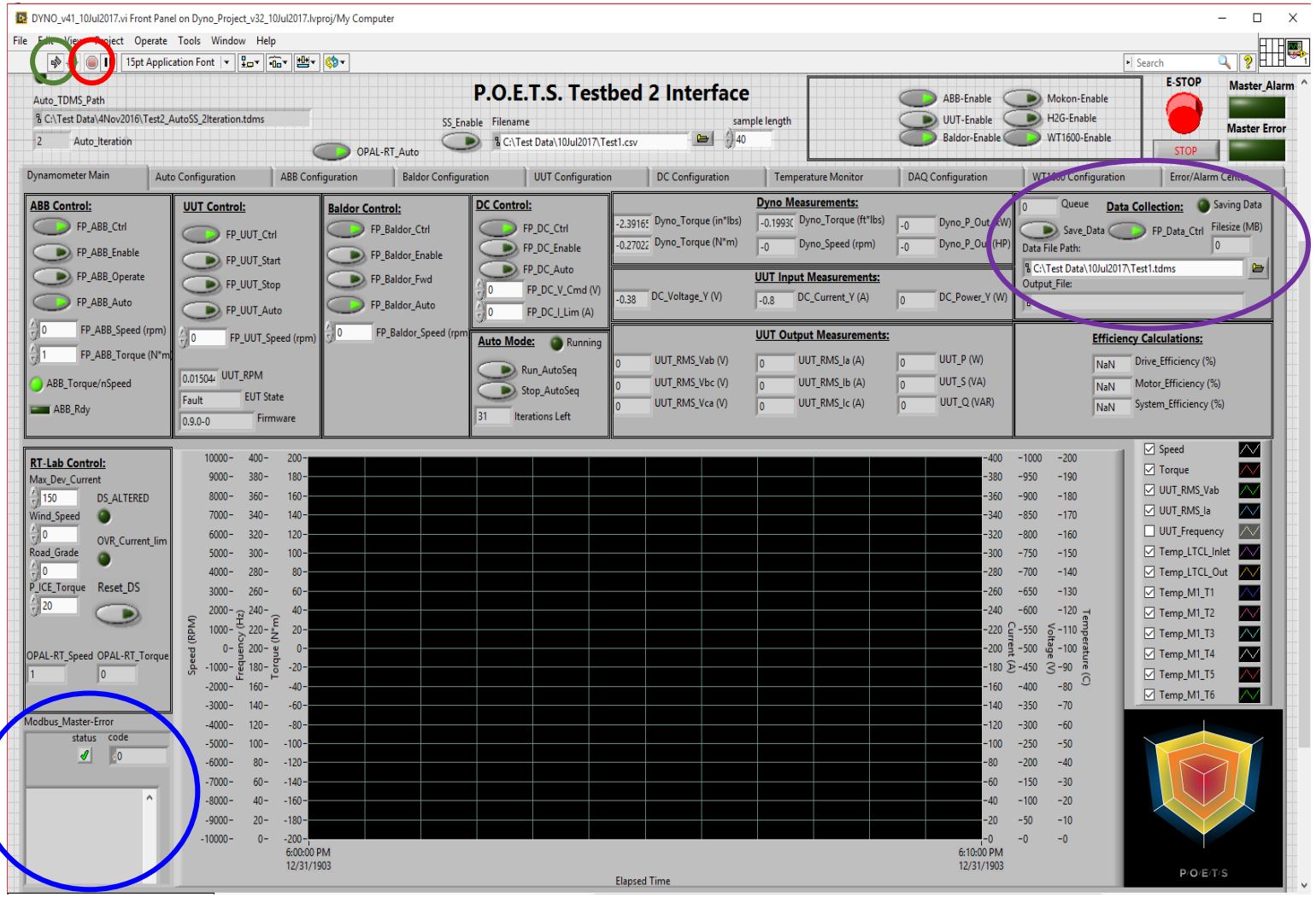

Figure A.10: LabVIEW Interface

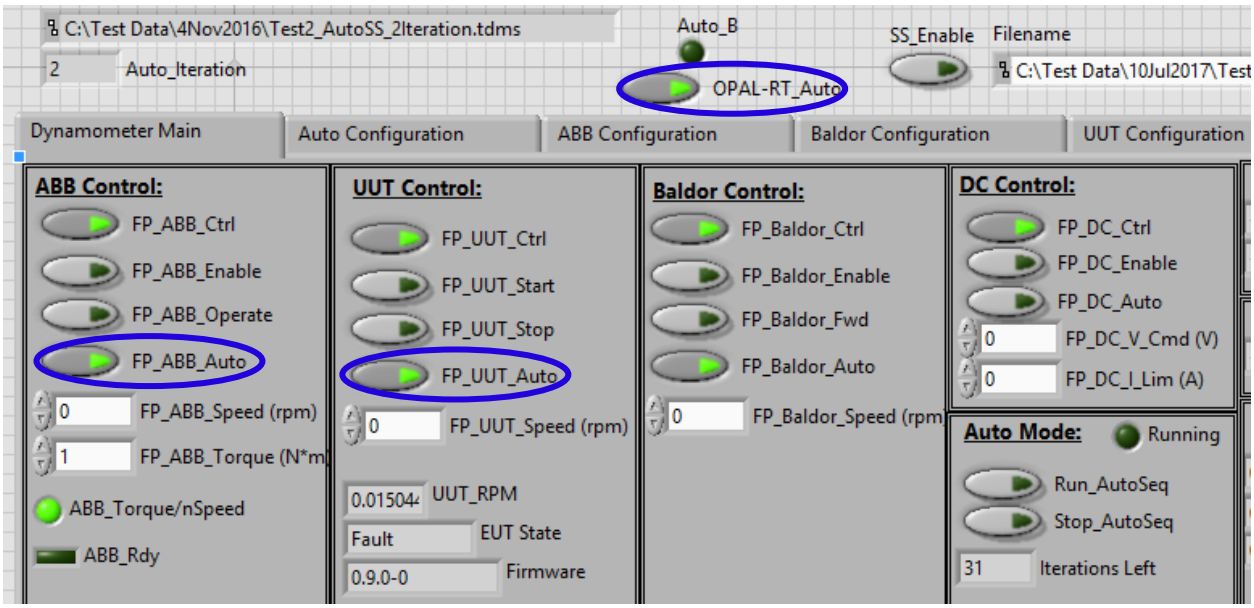

Figure A.11: Control Signals for Simulation Input of Speed and Torque

3. Energizing the system and running test
    a. Set control signals for Simulation Input
        i. Run LabVIEW code
        ii. ENABLE “FP_ABB_Auto”, “FP_UUT_Auto”, and “OPAL-RT_Auto” (as shown in Fig. A.11)
    b. Notify NCREPT personnel of the nature of your test and what you plan to do then listen to the subsequent safety briefing.
    c. Make sure the testing bay is clear of personnel
    d. Energize breakers (NCREPT personnel will be present for this step)
        i. Breaker F5 must be racked in and enabled from the NCREPT control room to supply power to the unit under test
        ii. The breaker labeled “ABB Drive” in the breaker box at the end of the first row of equipment in the test bay must be turned on to power the load motor and drive

e. Execute the LabVIEW code and make sure no Modbus errors are present and make sure that the “Master Error” indicator in the top right corner of the interface is green as shown in Fig. A.10
    i. Sometimes, if all the analog to digital converters are not fully on and ready you may get a Master error.
        1. To view the cause of this, click the “Error/Alarm Center” tab in the interface
        2. This “ADC error” can be resolved by simply stopping (stop sign shaped button circled in red in Fig. A.10) and starting the LabVIEW code over again.
f. Set the Real Time vehicle parameters in the LabVIEW interface (“RT-Lab Control” section)
    i. Max_Dev_Current – maximum RMS phase current you want your devices to see during the test. The program limits speed and torque commands to keep current below that limit.
    ii. Wind_Speed – speed of the head wind experienced by the vehicle
    iii. Road_Grade – angle of road incline.
    iv. P_ICE_Torque – average percent of torque provided by internal combustion engine over a driving schedule. Allows for testing of hybrid electric vehicles.
g. Enable the ABB ACS800 load drive
    i. ENABLE “FP_ABB_Enable” and wait for the “ABB_Rdy” indicator to light up

ii. ENABLE “FP_ABB_Operate”

h. Enable the UUT

i. For the testbed system validation, a Baldor H2 Vector drive was used, its control signals are located in the “Baldor Control” section of the interface.

ii. ENABLE “FP_Baldor_Enable” and “FP_Baldor_Forward”

iii. Depending on the control setup the prototype being tested will likely have different enable signals than the Baldor drive. However, when the control interface is developed with LabVIEW, the appropriate signal switches should be placed in the “UUT_Control” section of the “Dynamometer Main” screen.

i. If you wish to record your data change the file path to the appropriate date in the Data Collection portion of the interface (circled in purple in Fig. A.10) and ENABLE “Save Data”

i. i.e. C:\TestData\DayMonthYear\Test1.tdms

ii. The data files will increment automatically if you stop and start the data collection. If you wish to split your data into smaller files for easy analysis this is the easiest way to do so.

j. Navigate to RT-Lab window, and resume the simulation.

i. The OPAL-RT server will now begin sending torque and speed commands to the drive in real time as the vehicle runs the simulated driving schedule.

ii. Remember: If you need to stop operation, disable the UUT enable signal and “FP_ABB_Operate” to allow the testbed to coast down to zero speed. Refer to the E-Stops section for more information.

k. Driving Schedule Reset
    i. This can be used to start the driving schedule over after it is complete or at any point in the cycle where the speed and torque commands are zero.
    ii. DO NOT reset the driving schedule when non-zero values are present for the “OPAL-RT_speed” and “OPAL-RT_Torque”. These can be monitored in the “RT Lab Control” section just below the reset button (“Reset_DS”)
l. Disable drives when the simulation is complete
    i. Stop data collection
    ii. DISABLE UUT Enable variables (“FP_Baldor_Enable” and “FP_Baldor_Forward” for system validation scenario)
    iii. DISABLE “FP_ABB_Operate” and then “FP_ABB_Enable”
    iv. Navigate to RT-Lab window, pause the program, then reset the target by clicking the red square on the toolbar.
m. De-energize and rack out breakers F5 and F11

Creating and Using a Custom Driving Schedule

1. Create a speed profile in excel with the first column being time and the second column being vehicle speed in mph
    a. Do not include column titles like “time” and “speed” as this will cause problems when it is loaded into Matlab
2. Save the file as a “.csv” format with a descriptive name.
3. Move a copy of the file into the same directory as the real time model
4. Follow Step 1,c-d described above to open the model
5. Next add “VarName=csvread(‘newDrivingScheduleFilename.csv’);” to the preload functions window in Fig. A.7.
6. Select the new driving schedule and save changes by following Step 1, e-g in the “Operating the Testbed With the Simulated Vehicle Model” section

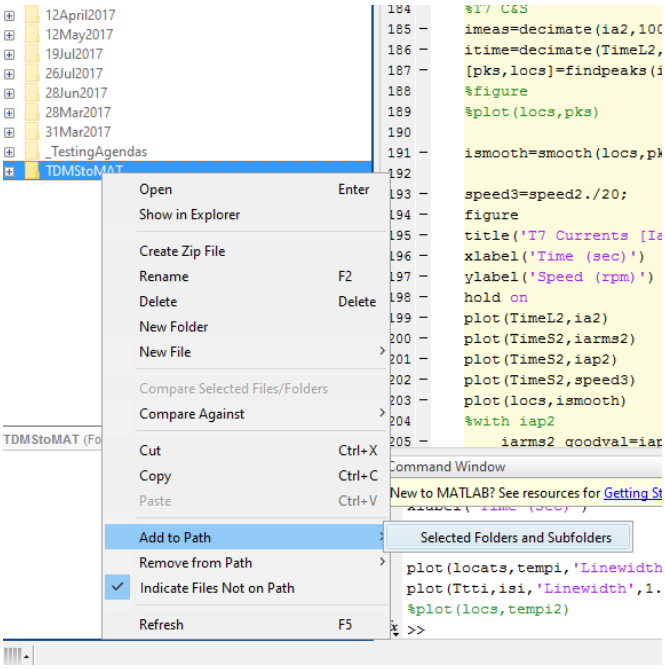

Figure A.12: Add File to Path

Post Processing

Data Analysis

For quick looks at the data National Instruments DIAdem (v2014) software can be used to view the .tdms files on the testbeds computer. However, that software is difficult to use if heavy calculations or analysis is required. The .tdms data files be easily converted to .mat files for use in Matlab but it is recommended that this be done on a personal computer, not the control room computer. On the NCREPT Control room computer desktop there is a file folder that contains the .tdms to .mat file converter which can be copied to a removable drive. Matlab version 2015a or newer should be used for data analysis since they are more memory efficient and can handle larger data files. Once you open Matlab follow the steps illustrated below to convert the files.

1. Be sure to add the "TDMStoMAT" file folder and subfolders to the program path as shown in Fig. A.12
2. Navigate to the file with the data files in it
3. To use the converter type:
    a. *VariableName*= TDMS_getStruct('*dataFileName*.tdms');
    b. This will create a Matlab structure containing the data channels from the .tdms file
4. If you receive an error because your data files are too large, follow the steps in the "Reducing size of data files in DIAdem" section
5. To view the channels, type:
    a. "VariableName." Then press Tab to bring up a list of autofill options. The first level of the structure is separated into two sections, Props and Untitled. Props contains information about who created the file and the date and time it was saved. Untitled contains the data channels shown in Fig. A.13.

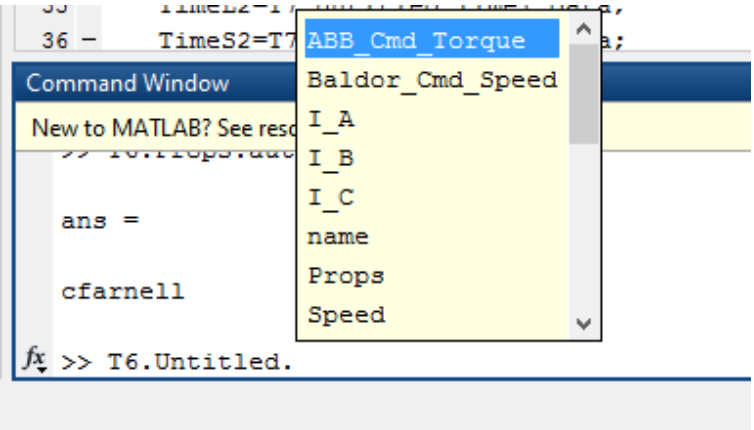

Figure A.13: Navigating Data Structures in Matlab

6. There is one more level to the structure after the channel name which contains fields for "data", "props", and "name".
    a. For example: "T6.Untitled.I_A.data" will return the raw data from the A phase current measurement and shown in the autofill list shown in Fig. A.13 above.
7. Descriptive names can be given to the data so that it is easier to identify and call.
    a. I_a= T6.Untitled.I_A.data; will save the data to the workspace in a new variable that is easier to identify at a glance and read in code if mathematical calculations if necessary.

<u>Reducing the Size of Data Files with NI DIAdem</u>

1. NI DIAdem can be used to remove channels that are not needed if the data files are too large to load into Matlab. To do so:
    b. Open DIAdem then clear the data portal window (right side in Fig. A.14) by right clicking on the top level name, in this case "EXAMPLE" and clicking delete
    c. In the navigation screen, find the .tdms file you wish to edit and drag it to the right hand "Data Portal" plane

d. Once the data file is loaded, the portal window should appear similar to Fig. A.15

e. The DAQ has over 145 channels listed here, channels that are not immediately relevant can be deleted to reduce the file size

   i. Deleting these channels will not permanently delete any channels from the original .tdms file unless you save the new file under the same name

   ii. Once the appropriate channels have been deleted from the portal, right click on the uppermost level in the data portal window and click “save as” and save the file with a different name than the original

   iii. Figure A.16 shows an example of a reduced channel list

   iv. This reduced size file can now be loaded into Matlab using the method described previously in the “Data Analysis” section.

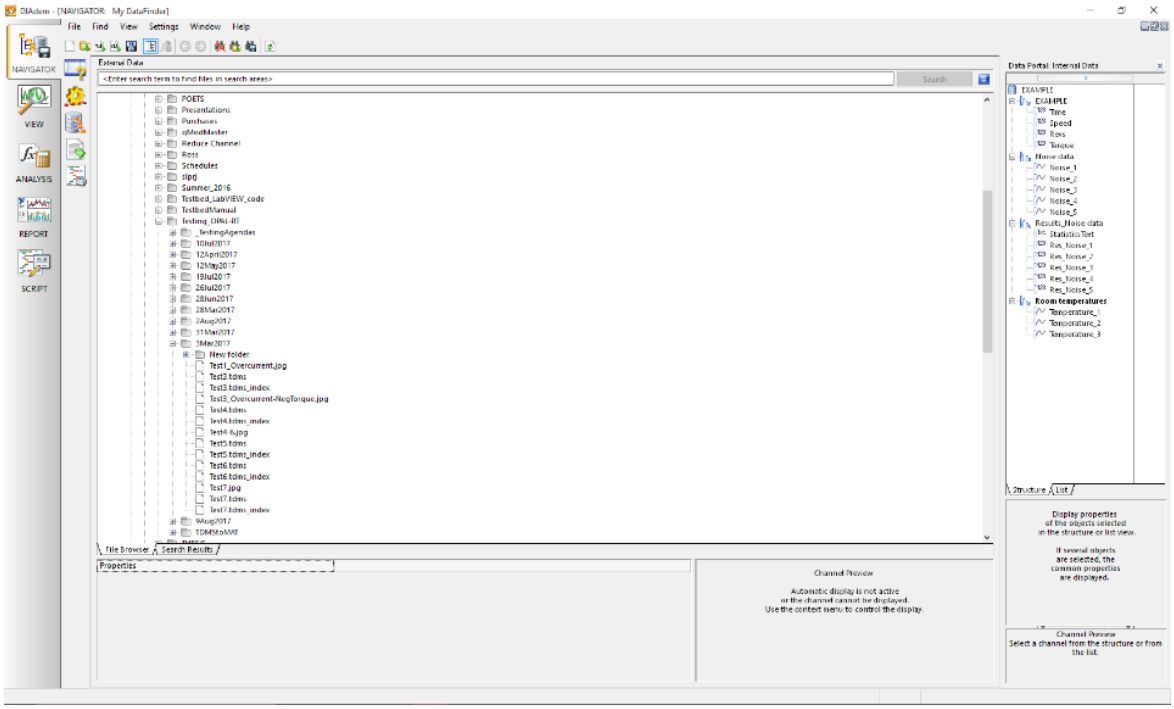

Figure A.14: DIAdem Navigator Screen

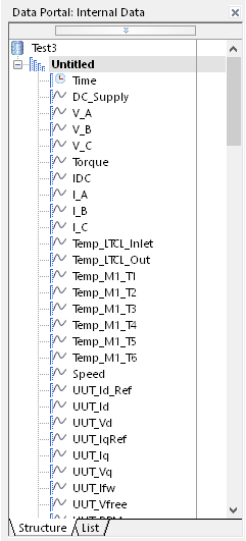

Figure A.15: Data Portal

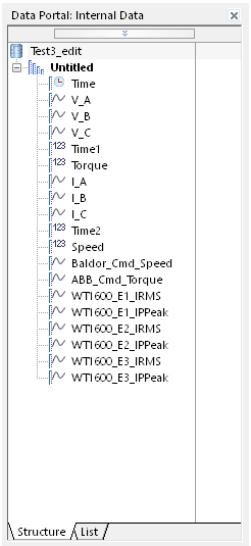

Figure A.16: Data Portal After Reduction

Propulsion Motor Interface

The propulsion motor in the testbed is a 4-pole permanent magnet synchronous machine (PMSM) rated for 75 HP and 7000 RPM. It is equipped with a resolver for absolute position feedback. The resolver is rated for 20V and 22 mA at 2381 Hz. The resolver has a transformation ratio of ~0.5, which means the maximum magnitude of the differential output signals will be half of the excitation magnitude. The feedback cable has a DB-9 connector for interfacing with the unit under test (pinout in Table A.2). Figure A.17 shows the wiring diagram for the resolver.

Table A.2: Resolver to UUT Cable Connection

Resolver pin	Function	DB-9 Connector
A	Excitation +	3
B	Excitation -	1
D	Sin +	8
E	Cos -	6
F	Sin -	9
G	Cos +	7

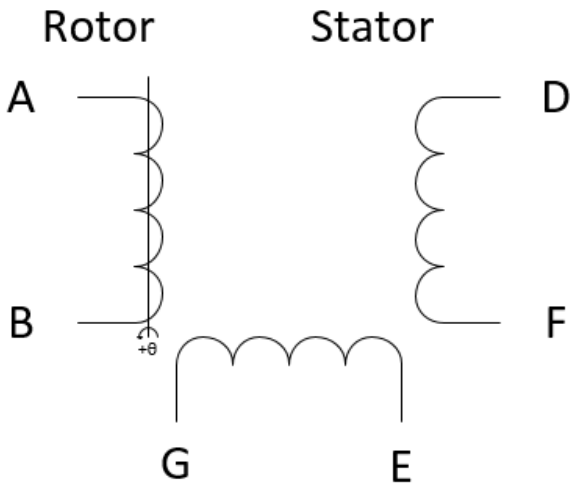

Figure A.17: Resolver Wiring Diagram

Installation of Breaking Resistors

Breaking Resistors are used to protect the dc power supplies by dissipating the energy generated when the PMSM is acting as a generator. Breaking resistors are typically connected on the dc bus in between a dc/ac converter and the dc source. NCREPT has a Bonitron M775RK-H075B dynamic breaking resistor that can be used with the testbed. It senses when the dc bus voltage exceeds a threshold and uses an IGBT to add the resistor bank across the dc link, dissipating the energy as heat. Table A.3 below shows the maximum ratings of the breaking resistor unit.

Table A.3: Breaking Resistor Ratings

Resistor Bank	
Model	BONITRON M3775RK –H075B
Power Rating	42.7 kW
Ohms	10
Breaking Unit	
Model	Yaskawa CDBR-4220D
Rating	400 V Class 220 kW
Input Rating	510-680 V_{dc} and 80 A

Breaking resistors should be installed any time the user is attempting to operate the motor in the forward breaking region of operation as discussed in Section 4.1. Care should also be taken even when testing is restricted to the forward motoring region. As discussed in Section 5.3 and shown in Fig. 5.3, the motors will generate energy as they decelerate even with the torque command set to zero. The slew rates of the driving schedule and the inertia of the system (discussed and measured in Section 5.3) should be used to approximate the torque and current at these deceleration rates. The energy generated should be compared to the limits of the dc power supply in order to determine whether or not the breaking resistors are necessary.

Example Test Schedule

POETS Testbed 2 Experiment Agenda			
Testing Requested by:	Firstname Lastname		
Date/Time of Testing:	8/30/2017 , 1pm		
Facility Resources Needed:	NCREPT, Dynamometer testbed, OPAL-RT server		
Allotted Time for Test:	2 Hours		
Test Objective #1:	Short Consise test objective		
Approach:	Explain the approach to the test here. What schedule you want to run, why you want to run it, and talk about what you hope to gain from the test.		
Test Objective #2:	Test IUT with custom driving schedule		
Approach:	Includes multiple speed ramps increasing in acceleration as the file goes on. This will test the current limiting features of the IUT. 300A software current limit		
Test Objective #3:	Test prototype inverter with US06 driving schedule		
Approach:	Simulate aggressive highway driving with a Nessan Leaf. Set a current limit of 100 A for initial tests. Log data for later analysis.		
Allocation of test time			
Scheduled Start Time:	1pm		
Scheduled End Time:	3pm		
Task	**Start**	**End**	
Setup/ safety briefing	**1:00 PM**	**1:30 PM**	
Test Objective #1	**1:35 PM**	**1:55 PM**	
Test Objective #2	**2:00 PM**	**2:30 PM**	
Test Objective #3	**2:30 PM**	**2:50 PM**	
Power Down System/ Data Retrieval and Storage	**2:50 PM**	**3:00 PM**	

Data Acquisition Variable List

Table A.4: Data Acquisition System Measurement Channels

Number	Variable Name	Sample Rate (S/s)	Source	Description
1	Time	80000	LabVIEW	time data channel, records simulation time from start to finish
2	DC_Supply	80000	Voltage Probe	records DC supply voltage
3	V_A	80000	Voltage Probe	Voltage on phase A
4	V_B	80000	Voltage Probe	Voltage on phase B
5	V_C	80000	Voltage Probe	Voltage on phase C
6	Torque	80000	Torque Sensor	Measured Torque at the shaft by SDI 01424-023-GB0AA rotary torque sensor
7	IDC	80000	Current Transformer	Current on DC bus
8	I_A	80000	Current Transformer	Current on phase A
9	I_B	80000	Current Transformer	Current on phase B
10	I_C	80000	Current Transformer	Current on phase C
11	Temp_LTCL_Inlet	4	Thermocouple	Low temperature coolant loop inlet temperature
12	Temp_LTCL_Out	4	Thermocouple	Low temperature coolant loop outlet temperature
13	Temp_M1_T1	4	Thermocouple	Configurable
14	Temp_M1_T2	4	Thermocouple	Configurable
15	Temp_M1_T3	4	Thermocouple	Configurable
16	Temp_M1_T4	4	Thermocouple	Configurable
17	Temp_M1_T5	4	Thermocouple	Configurable
18	Temp_M1_T6	4	Thermocouple	Configurable
19	Speed	4	Torque Sensor	Measured Speed at the shaft by SDI 01424-023-GB0AA rotary torque sensor
20	Baldor_Cmd_Speed	4	RT-Lab via Modbus TCP	Commanded speed value from RT-Lab vehicle simulation
21	ABB_Cmd_Torque	4	RT-Lab via Modbus TCP	Commanded Torque value from RT-Lab vehicle simulation
22	UUT_Vd	4	UUT	Vd of Unit Under Test
23	UUT_IqRef	4	UUT	Iq reference of Unit Under Test
24	UUT_Iq	4	UUT	measured Iq of Unit Under Test
25	UUT_Vq	4	UUT	Vq of Unit Under Test
26	UUT_lfw	4	UUT	Configurable
27	UUT_Vfree	4	UUT	Configurable
28	UUT_RPM	4	UUT	RPM of Unit Under Test
29	Motor_Eff	4	LabVIEW calculation	Propulsion motor efficiency. Calculated by: (Dyno Output Power / UUT Output Power)*100
30	WT1600_E1_Urms	4	WT1600 Yokogawa Power analyzer	Vrms of phase A measured at motor terminals

Table A.4 (Cont.)

Number	Variable Name	Sample Rate (Hz)	Source	Description
31	WT1600_E1_UMN	4	WT1600 Yokogawa Power analyzer	Vmn of phase A
32	WT1600_E1_UDC	4	WT1600 Yokogawa Power analyzer	Vdc of phase A
33	WT1600_E1_UAC	4	WT1600 Yokogawa Power analyzer	Vac of phase A
34	WT1600_E1_IRMS	4	WT1600 Yokogawa Power analyzer	RMS current of phase A
35	WT1600_E1_IMN	4	WT1600 Yokogawa Power analyzer	Configurable
36	WT1600_E1_IDC	4	WT1600 Yokogawa Power analyzer	Phase A dc current offset
37	WT1600_E1_IAC	4	WT1600 Yokogawa Power analyzer	Phase A ac current
38	WT1600_E1_P	4	WT1600 Yokogawa Power analyzer	Phase A real power
39	WT1600_E1_S	4	WT1600 Yokogawa Power analyzer	Phase A apparent power
40	WT1600_E1_Q	4	WT1600 Yokogawa Power analyzer	Phase A Reactive power
41	WT1600_E1_PF	4	WT1600 Yokogawa Power analyzer	Phase A power factor
42	WT1600_E1_PHI	4	WT1600 Yokogawa Power analyzer	Configurable
43	WT1600_E1_FU	4	WT1600 Yokogawa Power analyzer	Configurable
44	WT1600_E1_FI	4	WT1600 Yokogawa Power analyzer	Configurable
45	WT1600_E1_UPPeak	4	WT1600 Yokogawa Power analyzer	Peak Voltage Phase A
46	WT1600_E1_UMPeak	4	WT1600 Yokogawa Power analyzer	Configurable
47	WT1600_E1_IPPeak	4	WT1600 Yokogawa Power analyzer	Peak Phase A current
48	WT1600_E1_IMPeak	4	WT1600 Yokogawa Power analyzer	Configurable
49	WT1600_E2_Urms	4	WT1600 Yokogawa Power analyzer	Vrms of phase B measured at motor terminals
50	WT1600_E2_UMN	4	WT1600 Yokogawa Power analyzer	Vmn of phase B
51	WT1600_E2_UDC	4	WT1600 Yokogawa Power analyzer	Vdc of phase B
52	WT1600_E2_UAC	4	WT1600 Yokogawa Power analyzer	Vac of phase B
53	WT1600_E2_IRMS	4	WT1600 Yokogawa Power analyzer	RMS current of phase B
54	WT1600_E2_IMN	4	WT1600 Yokogawa Power analyzer	Configurable
55	WT1600_E2_IDC	4	WT1600 Yokogawa Power analyzer	Phase B dc current offset

Table A.4 (Cont.)

Number	Variable Name	Sample Rate (Hz)	Source	Description
56	WT1600_E2_IAC	4	WT1600 Yokogawa Power analyzer	Phase B ac current
57	WT1600_E2_P	4	WT1600 Yokogawa Power analyzer	Phase B real power
58	WT1600_E2_S	4	WT1600 Yokogawa Power analyzer	Phase B apparent power
59	WT1600_E2_Q	4	WT1600 Yokogawa Power analyzer	Phase B Reactive power
60	WT1600_E2_PF	4	WT1600 Yokogawa Power analyzer	Phase B power factor
61	WT1600_E2_PHI	4	WT1600 Yokogawa Power analyzer	Configurable
62	WT1600_E2_FU	4	WT1600 Yokogawa Power analyzer	Configurable
63	WT1600_E2_FI	4	WT1600 Yokogawa Power analyzer	Configurable
64	WT1600_E2_UPPeak	4	WT1600 Yokogawa Power analyzer	Peak voltage Phase B
65	WT1600_E2_UMPeak	4	WT1600 Yokogawa Power analyzer	Configurable
66	WT1600_E2_IPPeak	4	WT1600 Yokogawa Power analyzer	Peak Phase B current
67	WT1600_E2_IMPeak	4	WT1600 Yokogawa Power analyzer	Configurable
68	WT1600_E3_Urms	4	WT1600 Yokogawa Power analyzer	Vrms of phase B measured at motor terminals
69	WT1600_E3_UMN	4	WT1600 Yokogawa Power analyzer	Vmn of phase C
70	WT1600_E3_UDC	4	WT1600 Yokogawa Power analyzer	Vdc of phase C
71	WT1600_E3_UAC	4	WT1600 Yokogawa Power analyzer	Vac of phase C
72	WT1600_E3_IRMS	4	WT1600 Yokogawa Power analyzer	RMS current of phase C
73	WT1600_E3_IMN	4	WT1600 Yokogawa Power analyzer	Configurable
74	WT1600_E3_IDC	4	WT1600 Yokogawa Power analyzer	Phase C dc current offset
75	WT1600_E3_IAC	4	WT1600 Yokogawa Power analyzer	Phase C ac current
76	WT1600_E3_P	4	WT1600 Yokogawa Power analyzer	phase C real power
77	WT1600_E3_S	4	WT1600 Yokogawa Power analyzer	Phase C apparent power
78	WT1600_E3_Q	4	WT1600 Yokogawa Power analyzer	Phase C Reactive power
79	WT1600_E3_PF	4	WT1600 Yokogawa Power analyzer	Phase C power factor
80	WT1600_E3_PHI	4	WT1600 Yokogawa Power analyzer	Configurable

Table A.4 (Cont.)

Number	Variable Name	Sample Rate (Hz)	Source	Description
81	WT1600_E3_FU	4	WT1600 Yokogawa Power analyzer	Configurable
82	WT1600_E3_FI	4	WT1600 Yokogawa Power analyzer	Configurable
83	WT1600_E3_UPPeak	4	WT1600 Yokogawa Power analyzer	Peak voltage phase C
84	WT1600_E3_UMPeak	4	WT1600 Yokogawa Power analyzer	Configurable
85	WT1600_E3_IPPeak	4	WT1600 Yokogawa Power analyzer	Peak Phase C current
86	WT1600_E3_IMPeak	4	WT1600 Yokogawa Power analyzer	Configurable
87	WT1600_E4_Urms	4	WT1600 Yokogawa Power analyzer	dc link RMS voltage
88	WT1600_E4_UMN	4	WT1600 Yokogawa Power analyzer	Configurable
89	WT1600_E4_UDC	4	WT1600 Yokogawa Power analyzer	dc Link Voltage
90	WT1600_E4_UAC	4	WT1600 Yokogawa Power analyzer	Configurable
91	WT1600_E4_IRMS	4	WT1600 Yokogawa Power analyzer	dc link RMS current
92	WT1600_E4_IMN	4	WT1600 Yokogawa Power analyzer	Configurable
93	WT1600_E4_IDC	4	WT1600 Yokogawa Power analyzer	dc link current
94	WT1600_E4_IAC	4	WT1600 Yokogawa Power analyzer	Configurable
95	WT1600_E4_P	4	WT1600 Yokogawa Power analyzer	dc link real power
96	WT1600_E4_S	4	WT1600 Yokogawa Power analyzer	dc link apparent power
97	WT1600_E4_Q	4	WT1600 Yokogawa Power analyzer	dc link reactive power
98	WT1600_E4_PF	4	WT1600 Yokogawa Power analyzer	dc link power factor
99	WT1600_E4_PHI	4	WT1600 Yokogawa Power analyzer	Configurable
100	WT1600_E4_FU	4	WT1600 Yokogawa Power analyzer	Configurable
101	WT1600_E4_FI	4	WT1600 Yokogawa Power analyzer	Configurable
102	WT1600_E4_UPPeak	4	WT1600 Yokogawa Power analyzer	dc link peak voltage
103	WT1600_E4_UMPeak	4	WT1600 Yokogawa Power analyzer	Configurable
104	WT1600_E4_IPPeak	4	WT1600 Yokogawa Power analyzer	dc link peak current
105	WT1600_E4_IMPeak	4	WT1600 Yokogawa Power analyzer	Configurable

Table A.4 (Cont.)

Number	Variable Name	Sample Rate (Hz)	Source	Description
106	WT1600_CA_Urms	4	WT1600 Yokogawa Power analyzer	Configurable
107	WT1600_CA_UMN	4	WT1600 Yokogawa Power analyzer	Configurable
108	WT1600_CA_UDC	4	WT1600 Yokogawa Power analyzer	Configurable
109	WT1600_CA_UAC	4	WT1600 Yokogawa Power analyzer	Configurable
110	WT1600_CA_IRMS	4	WT1600 Yokogawa Power analyzer	Configurable
111	WT1600_CA_IMN	4	WT1600 Yokogawa Power analyzer	Configurable
112	WT1600_CA_IDC	4	WT1600 Yokogawa Power analyzer	Configurable
113	WT1600_CA_IAC	4	WT1600 Yokogawa Power analyzer	Configurable
114	WT1600_CA_P	4	WT1600 Yokogawa Power analyzer	Configurable
115	WT1600_CA_S	4	WT1600 Yokogawa Power analyzer	Configurable
116	WT1600_CA_Q	4	WT1600 Yokogawa Power analyzer	Configurable
117	WT1600_CA_PF	4	WT1600 Yokogawa Power analyzer	Configurable
118	WT1600_CA_PHI	4	WT1600 Yokogawa Power analyzer	Configurable
119	WT1600_CA_FU	4	WT1600 Yokogawa Power analyzer	Configurable
120	WT1600_CA_FI	4	WT1600 Yokogawa Power analyzer	Configurable
121	WT1600_CA_UPPeak	4	WT1600 Yokogawa Power analyzer	Configurable
122	WT1600_CA_UMPeak	4	WT1600 Yokogawa Power analyzer	Configurable
123	WT1600_CA_IPPeak	4	WT1600 Yokogawa Power analyzer	Configurable
124	WT1600_CA_IMPeak	4	WT1600 Yokogawa Power analyzer	Configurable
125	FlowRate	8	Mokon / Pressure sensor	Mokon Flow Rate
126	DiffPress	8	Mokon / Pressure sensor	Mokon Differential Pressure
127	InPress	8	Mokon / Pressure sensor	Coolant loop inlet pressure
128	OutPress	8	Mokon / Pressure sensor	Coolant loop outlet pressure
129	Temp_HTCL_Inlet	8	Mokon / Thermocouple	High temperature coolant loop inlet
130	Temp_HTCL_Outlet	8	Mokon / Thermocouple	High temperature coolant loop outlet

Table A.4 (Cont.)

Number	Variable Name	Sample Rate (Hz)	Source	Description
131	Temp_CP_Top	8	Mokon / Thermocouple	Configurable
132	Temp_BP_ModA	8	Mokon / Thermocouple	Configurable
133	Temp_BP_ModC	8	Mokon / Thermocouple	Configurable
134	Temp_Cap_Ceramic	8	Mokon / Thermocouple	Configurable
135	Temp_Cap_Film	8	Mokon / Thermocouple	Configurable
136	Temp_BB_PhC	8	Mokon / Thermocouple	Configurable
137	Temp_BB_Pos	8	Mokon / Thermocouple	Configurable
138	Temp_PCB	8	Mokon / Thermocouple	Configurable
139	Temp_PS_Xfmr	8	Mokon / Thermocouple	Configurable
140	Temp_Gate_Res	8	Mokon / Thermocouple	Configurable
141	Temp_Gate_Drv	8	Mokon / Thermocouple	Configurable
142	Temp_Inlet_Air	8	Mokon / Thermocouple	Inlet air temperature
143	Temp_Ambient_Int	8	Mokon / Thermocouple	Ambient temperature internal to inverter housing
144	Temp_Ambient_Ext	8	Mokon / Thermocouple	Ambient temperature external to inverter

APPENDIX B : TESTBED TO UUT COMMUNICATION PROTOCOLS

Introduction

The LabVIEW based data acquisition system covered in Chapter 4 has additional communication capabilities other than the Modbus TCP/IP protocol that is currently being used. It is capable of serial communication via RS-232, RS-485, and USB as well as Controller Area Network (CAN) communication. This appendix will discuss each of these protocols and give a brief overview of the hardware interfaces (transport protocols) RS-232 and RS-485 then discuss how to implement a Modbus RTU packet structure on both. Next, Universal Serial Bus (USB) hardware and packet structure will be covered. Lastly, CAN communication and the implementation of custom packet structures and protocols will be discussed.

RS-232

RS-232 is one of the oldest serial hardware interfaces, whose standard defines a logic one as a voltage between -3V and -25V and a logic zero between 3 and 25V. Voltages between -3V and 3V are considered invalid signals which helps with noise reduction [1]. Typical voltage levels for RS-232 communication range from ±5 to ±12V but since the standard specifies ±25V any hardware designed on this standard that is to be operated in the testbed must be able to withstand the maximum range. Figure B.1 shows a standard DB9 connector and a typical signal waveform. The pinout for the female connector is the same as the male, however most standard cables (with the exception of null modem cables) will have their RX and TX lines swapped to make the correct connections for full duplex communication [2].

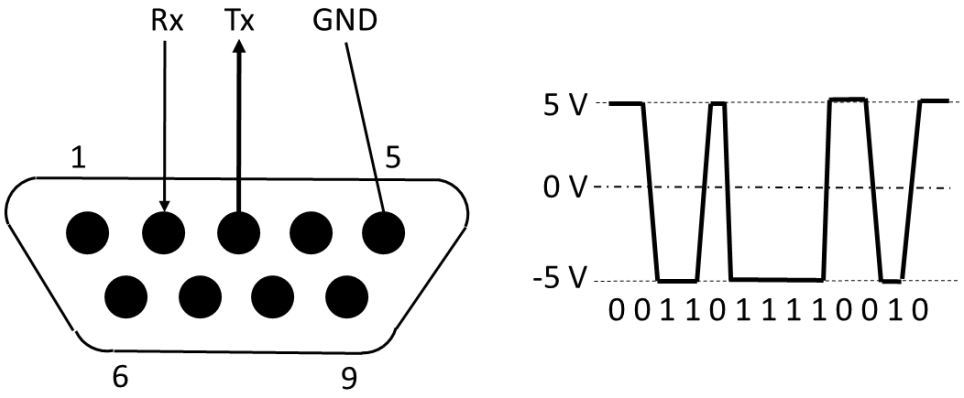

Figure B.1: RS-232 DB9 Pinout

RS-485

RS-485 differs from RS-232 in that it can be used to create a communication bus in addition to device to device communication. It also uses differential lines to transmit signals rather than a single communication line in reference to ground. This differential transmission method produces effective common-mode noise cancellation [1]. For a logic 1, the A terminal is negative relative to terminal B and a logic 0 is represented by terminal A being positive relative to B. A common ground pin may also be included to satisfy the ANSI-485 standard. Figure B.2 illustrates how to set up a simple 2-wire RS-485 communications bus [3] and the DB9 connector

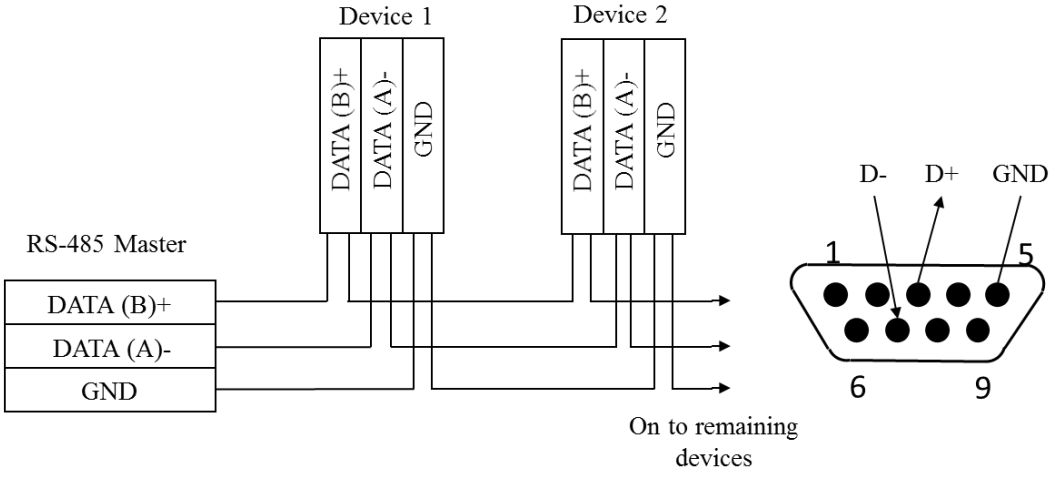

Figure B.2: 2-Wire RS-485 Connections

pinout [4]. The standard defines a logic 1 as greater than -200 mV and a logic 0 as greater than +200 mV. However, typical signal voltages range from ±1.5V to ±6V.

Modbus RTU

Modbus RTU and TCP/IP are both message structures that can be implemented in the LabVIEW control interface for communication with the UUT. Modbus RTU can be implemented on either RS-232 or RS-485. Modbus TCP/IP implementation has its own transport protocol that is discussed in Chapter 4. The Testbed control interface will serve as the Master device and the UUT will be the Slave device for the Modbus communication. The LabVIEW data communication toolbox provides pre-made virtual instruments (VIs) that do many of the communication tasks like reading and writing to holding registers and coils. These virtual instruments, like the "New Serial Master" block shown in the LabVIEW RTU communication code (Fig. B.3) are pre-made blocks of basic LabVIEW components that perform complicated tasks. They are the equivalent of subsystems in Matlab/Simulink. The LabVIEW Master Instance documentation discusses the different setup options for TCP/IP and Modbus serial type RTU or

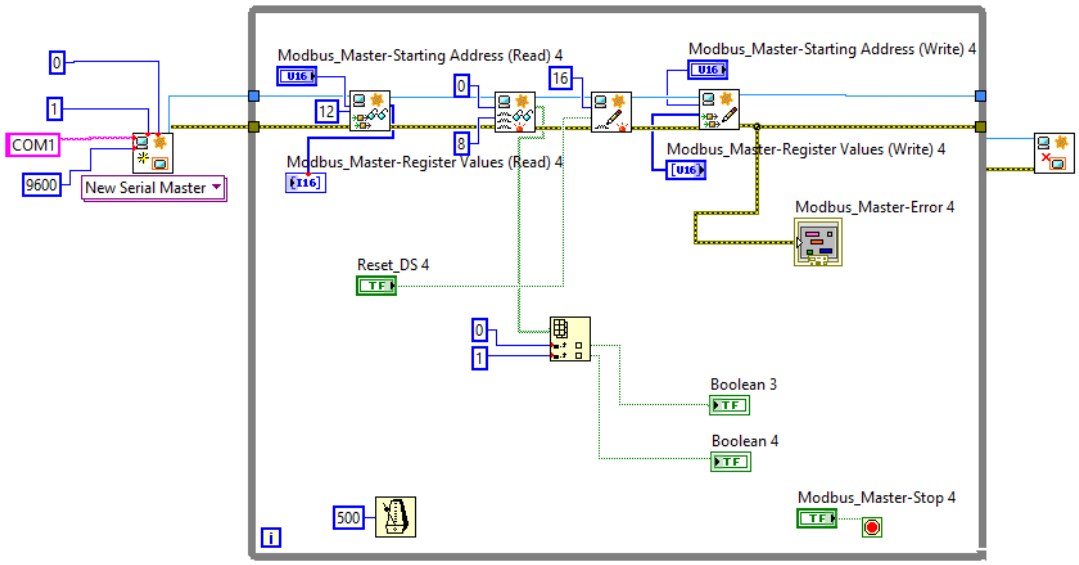

Figure B.3: Modbus Serial Implementation (RTU or ASCII)

ASCII [5]. Modbus RTU is the default and communicates the data in binary format. It can also be changed to ASCII format which uses human readable characters to send the data.

Universal Serial Bus (USB)

USB is another form of serial communication that uses four shielded wires. Two of which are 5V and ground with the remaining two being used as differential data signals. USB is host controlled, meaning one unit is responsible for scheduling bandwidth and undertaking all transactions. There can only be one host per bus. All devices have an “upstream connection” and all hosts have “downstream connections” which are not mechanically interchangeable and prevent illegal loopback connections at hubs. Type A plugs are the most common always face upstream while type A sockets usually find themselves on hosts [6]. In RS-232 and RS-485 the format of the data is undefined by the standard. However, USB like the TCP protocol, uses several layers of protocols that define its overall packet structure. Each USB transaction starts with a Token packet that is used to define what information is to follow. An optional data packet may follow the token packet. Lastly a status packet is transmitted to acknowledge transactions and for error detection. Each packet is made up of a combination of the six data fields: Synch, Packed ID (PID), the address field (ADDR), Endpoint field (ENDP), cyclic redundancy checks (CRC), and end of packet (EOP). Each packet must start with a synch field so that clock of the receiver and transmitter are synchronized. It varies from 8bits in length to 32 bits in length depending on the communication speed. The PID is used to identify the type of packet that is being transmitted. The PID is four bits, but is complemented and repeated to check for errors which brings the overall packet size to eight bits. Table B.1 shows the possible values for the PID field.

Table B.1: PID Packet Identifier Values

Group	PID Value	Packet Identifier
Token	0001	OUT Token
	1001	IN Token
	0101	SOF Token
	1101	SETUP Token
Data	0011	DATA0
	1011	DATA1
	0111	DATA2
	1111	MDATA
Handshake	0010	ACK Handshake
	1010	NAD Handshake
	1110	STALL Handshake
	0110	NYET (no response yet)
Special	1100	PREamble
	1100	ERR
	1000	Split
	0100	Ping

Table B.2: USB Packet Structures

Packet Type	Data Structure					
Token	Sync	PID	ADDR	ENDP	CRC5	EOP
Data	Sync	PID	Data		CRC16	EOP
Handshake	Sync	PID	EOP			
Start of Frame	Sync	PID	Frame Number		CRC5	EOP

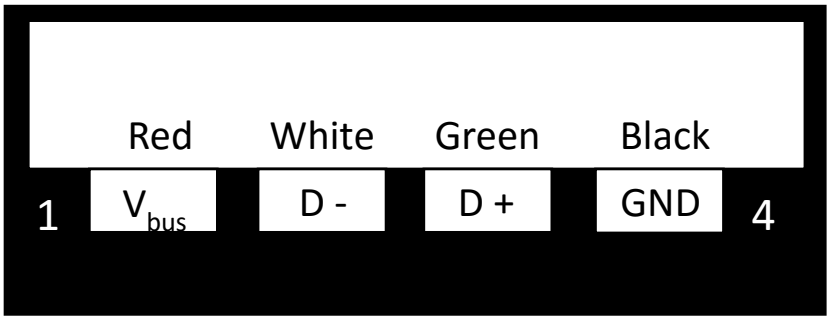

Figure B.4: USB Connector and Pinout

The ADDR field contains the address of the device that the packed is designated for. This field is seven bits long which limits the number of connected devices to 127. ENDP is four bits long which results in 16 possible endpoints. CRC field contains the check information for the data with a maximum length of 16 bits. Lastly EOP which is indicated as a 0V on D+ or D- for two bit times, signals the end of the packet [7]. Table B.2 shows the different USB packet types and their structures. The diagram, cable colors, and pinout for a Type A USB connector is shown in Fig. B.4. A differential 1 is transmitted by D+ over 2.8V and D- under .3V. A transmitted differential 0 is the opposite, D- must be greater than 2.8V and D+ must be less than .3V. The receiver defines a logic 1 as D+ being at least 200mV greater than D- and logic 0 as D+ being 200mV less than D-.

Controller Area Network (CAN) Bus

CAN is a multi-master message broadcast system that was designed for automotive applications. In contrast to traditional networks, it does not transmit large blocks of data point to point. Instead, many short messages are broadcast to the entire network. Since each node in the network could theoretically begin transmitting at the same time, a collision detection protocol is implemented to prevent data loss [8]. If multiple nodes begin to transmit when the bus is free the identifier at the beginning of the data frame will decide which message is transmitted. Each identifier is unique and represents the messages priority level. As the identifier for the message is broadcast, each node compares the bit it is transmitting to the value on the bus. The higher priority identifier will have a dominant bit earlier in the identifier. When the nodes see a dominant bit on the bus they aren't transmitting, it interprets this as another message with higher priority transmitting simultaneously and will suspend transmission before the next bit [9]. Figure B.5 shows the standard CAN bit field. Table B.3 describes each of the bit fields [8].

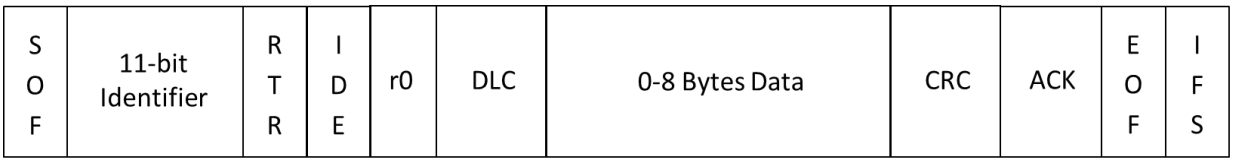

Figure B.5: CAN Bit Field

Table B.3: CAN Bit Field Descriptions

Bit Field	Description
SOF	Start of frame bit and is used to synchronize nodes after the bus being idle
Identifier	Establishes priority of message. The lower the binary value the higher priority of the message
RTR	Remote transmission request is dominant when information is required from another node
IDE	A dominate single identifier extension bit means standard CAN with no extension is being used. A recessive bit means that an 18 bit extension of the identifier follows
r0	Reserved bit
DLC	The 4-bit data length code contains the number of bytes of data being transmitted
Data	Up to 64 bits of application data may be transmitted
CRC	The 16 bit cyclic redundancy check contains the checksum of the application data for error detection
ACK	Every node receiving an accurate message overwrites this recessive bit in the original message with a dominant bit indicating an error free message. Message repeats if no ACK overwrite is done. ACK is two bits, one is the acknowledgement and the second is a delimiter
EOF	End-of-frame, 7 bit field marks the end of a Can frame (message) and disables bit stuffing, indicating a stuffing error when dominant. When 5 bits of the same logic level occur in succession during normal operation, a bit of the opposite logic level is stuffed into the data.
IFS	7-bit interframe space contains the time required by the controller to move a correctly received frame to its proper position in a message buffer

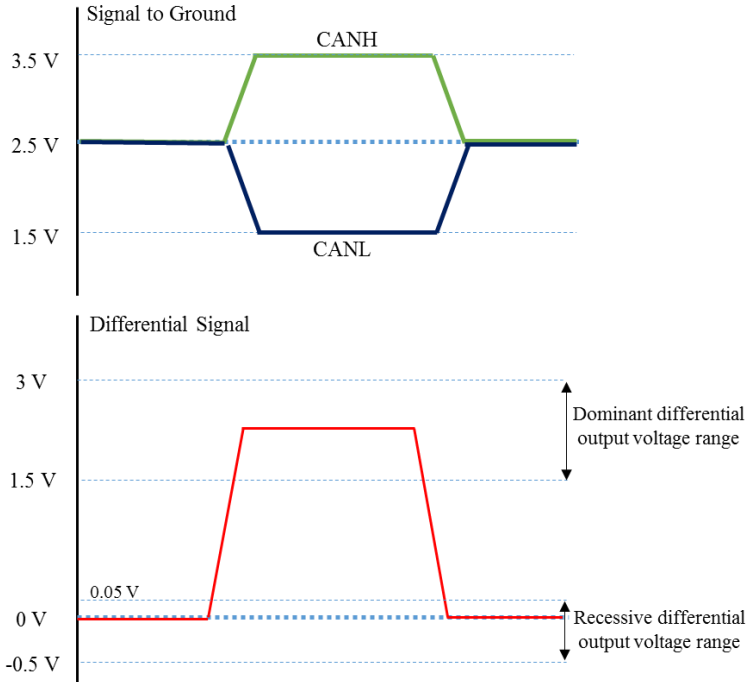

Figure B.6: ISO11898 CAN Bus Voltage Levels

The NI PXI-8531 CANopen interface module is used in the DAQ for CAN communication to the UUT. It follows the ISO11898 standards [10] for electrical compliance. The DC voltage on CANH and CANL must be able to withstand short circuits on the CAN bus from -3V to 32V and transient voltages from -150V to 100V. The recessive differential output bus voltage will range from -.5V to .05V and the differential dominant output voltage will range from 1.5V to 3V [19]. The nominal ISO11898 bus voltage levels are shown in Fig. B.6.

Custom Communication Protocols

Custom communication protocols for the application structure can be incorporated into the testbed however modifications to the LabVIEW testbed control code will be necessary and significant lead time should be given to accommodate this request. Detailed documentation on the data and packet structure must also be included for the protocol. The transport protocols (physical layer) for the testbed however are somewhat limited to serial interfaces via RS232 or RS485, USB, and TCP/IP.

References

[1] L. Frenzel, "What's The Difference Between The RS-232 And RS-485 Serial Interfaces?", *Electronic Design*, 2016. [Online]. Available: http://www.electronicdesign.com/what-s-difference-between/what-s-difference-between-rs-232-and-rs-485-serial-interfaces. [Accessed: 02- Feb- 2018].

[2] "Simply Modbus - About Modbus", *Simplymodbus.ca*, 2017. [Online]. Available: http://www.simplymodbus.ca/FAQ.htm. [Accessed: 02- Feb- 2018].

[3] "RS-485 Connections FAQ, 2 Wire RS485/RS232 - B&B Electronics", *Bb-elec.com*, 2018. [Online]. Available: http://www.bb-elec.com/Learning-Center/All-White-Papers/Serial/RS-485-Connections-FAQ.aspx. [Accessed: 02- Feb- 2018].

[4] "RS232, RS422 & RS485 standard DB connector pinout", *Opengear Help Desk*. [Online]. Available: https://opengear.zendesk.com/hc/en-us/articles/216371923-RS232-RS422-RS485-standard-DB-connector-pinout. [Accessed: 02- Feb- 2018].

[5] "Create Master Instance VI - LabVIEW 2017 Real-Time Module Help - National Instruments", *Zone.ni.com*, 2017. [Online]. Available: http://zone.ni.com/reference/en-XX/help/370622R-01/lvmve/modbus_create_master_instance/. [Accessed: 02- Feb- 2018].

[6] C. Peacock, "USB in a NutShell - Chapter 2 - Hardware", *Beyondlogic.org*, 2010. [Online]. Available: http://www.beyondlogic.org/usbnutshell/usb2.shtml. [Accessed: 08- Feb- 2018].

[7] C. Peacock, "USB in a NutShell - Chapter 3 - USB Protocols", *Beyondlogic.org*, 2010. [Online]. Available: http://www.beyondlogic.org/usbnutshell/usb3.shtml. [Accessed: 08- Feb- 2018].

[8] S. Corrigan, *Introduction to the Controller Area Network (CAN)*. Texas Instruments Inc., 2016.

[9] S. Maradana, "CAN Basics", *AUTOMOTIVE BASICS*, 2012. [Online]. Available: https://automotivetechis.wordpress.com/2012/06/01/can-basics-faq/. [Accessed: 02- Feb- 2018].

[10] P. Richards, *A CAN Physical Layer Discussion*. Microchip Technology Inc., 2002.