4-2016

Model Based Vehicle Simulation for Electrically Actuated Transmission Test Cells

Kenneth L. Gruel

MODEL BASED VEHICLE SIMULATION FOR ELECTRICALLY ACTUATED TRANSMISSION TEST CELLS

by

Kenneth L. Gruel

A thesis submitted to the Graduate College
in partial fulfillment of the requirements
for the degree of Master of Science in Engineering
Electrical and Computer Engineering
Western Michigan University
April 2016

Thesis Committee:

Janos Grantner, Ph.D., Chair
Damon Miller, Ph.D.
Ralph Tanner, Ph.D.

MODEL BASED VEHICLE SIMULATION FOR ELECTRICALLY ACTUATED TRANSMISSION TEST CELLS

Kenneth L. Gruel, M.S.E

Western Michigan University, 2016

The Eaton Vehicle Group is responsible for the development of a large family of automotive products, including manual and automated manual transmissions for light, medium, and heavy duty vehicles. To further expand the transmission testing capabilities at Vehicle Group headquarters, the Vehicle Simulator system was developed and integrated into an existing motor-based test cell. System development time and cost were reduced by leveraging readily available vehicle models and real-time hardware. The Vehicle Simulator system allows a user to control the speed and torque settings of the test cell motors in manual mode, or switch the motors to vehicle emulation mode. All operation modes can be controlled through a user interface, Controller Area Network (CAN) bus messages, physical vehicle pedals, or a script. A simulated test cell was also developed, allowing the user to test system functionality without the need to actuate the physical test cell and transmission. Details of the Vehicle Simulator system integration are provided along with test results showing operation of the system in manual mode as well as vehicle emulation mode.

ACKNOWLEDGEMENTS

To start, I would like to thank my thesis committee, in particular Dr. Grantner, for taking the time to answer my long list of questions and provide assistance through the thesis process. With the extended time period it took to complete this thesis, the committee kept me on track, and helped me refocus when the final project did not quite match my initial vision. Also, I would like to express gratitude to the committee for the time spent reading the thesis and generating feedback.

Since this thesis was developed as a project for my employer, Eaton, there are a great number of colleagues whose technical expertise and knowledge significantly contributed to the foundation of this project. In particular, I would like to thank Frederick James for his key technical knowledge that allowed the project to move forward each time a wall seemed to be blocking the way. I would also like to thank Scott O'Neill, whose talent allowed for the integration of the Vehicle Simulator system into the existing test cell within a short timeframe, and was critical to the success of this project. Justin Griffiths, as one of the original developers of the vehicle plant models, provided crucial assistance integrating the existing vehicle models into the system. The following individuals also provided key support to the project: Joe Holeman, Shrirang Agahese, Ryan Pauls, Michael Galloway, Jacob Spitzner, Marie Maddix, Christopher DeBore, Gregory Huff, Yeidei Wang, and many others.

Finally, I would like to thank my family and friends for all of their encouragement and support throughout this project. I appreciate my parents for instilling in me the importance of education, and always motivating me to do the best work that I could possibly do. I also appreciate, more than I can accurately express, all of the effort my wife Jen has put forth caring for our son Harrison during the long hours spent writing and revising this thesis. Without her tireless effort, I doubt much more than a paragraph would have been written.

Kenneth L. Gruel

TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....ii

LIST OF TABLES.....v

LIST OF FIGURES.....vi

1 INTRODUCTION.....1

1.1 The Vehicle Simulator.....2

1.1.1 Vehicle Simulator System Specifications.....3

1.1.2 Vehicle Simulator System Realization.....4

1.2 Scope of Original Work.....6

1.3 Literature Review.....7

2 IMPLEMENTATION STRATEGY.....11

2.1 Vehicle Simulator and Test Cell Interface.....12

2.1.1 High Level System Overview.....12

2.1.2 Vehicle Simulator Interface.....14

2.2 Stand Interface Model.....22

2.2.1 Motor Function Selection.....24

2.2.2 User Input Mode Selection.....24

2.2.3 Analog Signal Conditioning.....28

2.2.4 Stand Interface Safety Features.....33

2.2.5 Bypass Development Mode.....38

2.3 Simulated Test Cell.....41

2.3.1 Main Simulated Test Cell System.....42

2.3.2 Simulated Motor and Drive Interconnection.....43

2.4 Vehicle Plants Models.....45

Table of Contents - Continued

2.4.1 Engine Plant Model 46

2.4.2 Chassis Plant Model 48

2.4.3 Input and Output Adapters 52

3 IMPLEMENTATION VERIFICATION 53

3.1 Data Collection and Analysis Software 53

3.2 Test 1: Input Motor Speed Control – Output Motor Torque Steps 54

3.2.1 Test 1: Test Procedure 54

3.2.2 Test 1: Results and Analysis 56

3.2.3 Test 1: Comparison with the Simulated Test Cell 62

3.3 Test 2: Input Motor Torque Steps – Output Motor Speed Control 65

3.3.1 Test 2: Test Procedure 65

3.3.2 Test 2: Results and Analysis 67

3.3.3 Test 2: Comparison with the Simulated Test Cell 72

3.4 Test 3: Dynamic Shifting of a Transmission 75

3.4.1 Test 3: Data Normalization 75

3.4.2 Test 3: Procedure 75

3.4.3 Test 3: Results and Analysis 76

3.5 Additional Verification Items 84

4 CONCLUSION 85

4.1 System Performance Summary 85

4.2 Future Work 86

BIBLIOGRAPHY 88

LIST OF TABLES

2.1: Vehicle Simulator expansion card utilization 14
2.2: APCIe-3121 specifications 16
2.3: APCIe-2200 specifications 16
2.4: Vehicle Simulator inputs 17
2.5: Input motor analog feedback scaling 17
2.6: Output motor analog feedback scaling 18
2.7: User console pedal analog scaling 18
2.8: Stand instrumentation scaling 19
2.9: PLC feedback signals 19
2.10: Vehicle Simulator outputs 20
2.11: Input motor drive analog command scaling 20
2.12: Output motor drive analog command scaling 20
2.13: Vehicle Simulator test cell outputs 21
2.14: Input selection mode 25
2.15: User console pedal scaling variables 30
3.1: Test 1 – Data plot channel definitions 57
3.2: Test 3 Graph signals 76

LIST OF FIGURES

1.1: Vehicle Simulator high level diagram 4
1.2: Vehicle Simulator system overview 5
2.1: Vehicle Simulator system diagram 11
2.2: Vehicle Simulator and test cell signal routing 12
2.3: Motor rotation convention 13
2.4: APCIe-2200 card layout 15
2.5: Test cell user console pedals 18
2.6: Stand Interface model 22
2.7: Stand Interface Simulink model 23
2.8: Stand Interface user selection mode 25
2.9: CAN command message layout 27
2.10: CAN command bit layout 27
2.11: User console pedal scaling 29
2.12: Motor Drive feedback scaling 31
2.13: Motor command signal scaling 32
2.14: Safe state check 33
2.15: Generic safe mode setting selector 34
2.16: Stand operational logic 35
2.17: Input motor shutdown 36
2.18: Heartbeat pulse 37
2.19: Bypass mode logic 38
2.20: Bypass mode input 39
2.21: Bypass mode output 40
2.22: Simulated drive and motor 43
2.23: Simulated motor drive 44
2.24: Vehicle model utilization strategy 45
2.25: Engine Plant overview 46
2.26: Chassis Plant system 49

List of Figures - Continued

2.27: Simple inclined plane 51

3.1: Test 1 - Overall test 56

3.2: Test 1 - Speed ramp detail 58

3.3: Test 1 - Speed signal detail 59

3.4: Test 1 - Input and output motor speed detail 60

3.5: Test 1 - Output motor torque detail 61

3.6: Test 1 - Simulated test cell 62

3.7: Test 1 - Simulated test cell comparison of input motor values 63

3.8: Test 1 - Simulated test cell comparison of output motor values 64

3.9: Test 2 - Overall test 67

3.10: Test 2 - Speed ramp detail 68

3.11: Test 2 - Speed signal detail 69

3.12: Test 2 - Input and output motor speed detail 70

3.13: Test 2 - Input motor torque detail 71

3.14: Test 2 - Simulated test cell 72

3.15: Test 2 - Simulated test cell comparison of input motor values 73

3.16: Test 2 - Simulated test cell comparison of output motor values 74

3.17: Test 3 - Overall dynamic shifting 77

3.18: Test 3 - Launching detail 79

3.19: Test 3 - Shifting detail 80

3.20: Test 3 - Input motor torque command detail 81

3.21: Test 3 - Input motor torque command additional detail 82

3.22: Test 3 - Output motor speed command detail 83

NOTATIONS

Product names used throughout the text

MATLAB® / Simulink®	http://www.mathworks.com/
Microsoft® / Windows®	https://www.microsoft.com/en-us/windows
ETAS® / LABCAR®	http://www.etas.com/en/products/solutions_labcar_component_overview.php
imc® / FAMOS®	http://www.imcdataworks.com/products/measurement-software/imc-famos/
Vector® / CANape® / CANcaseXL®	http://vector.com/vi_canape_en.html
J1939®	http://www.sae.org/standardsdev/groundvehicle/j1939a.htm
ADDI-DATA®	http://addi-data.com/

Acronyms

CAN	Controller Area Network
GUI	Graphical User Interface
HIL	Hardware-in-the-loop
MIL	Model-in-the-loop
PCIe	Peripheral Component Interconnection Express
PLC	Programmable Logic Controller
RTPC	Real-Time Personal Computer

UNITS

Hz	Hertz
$kg \cdot m^2$	Kilogram Meters Squared
$\frac{m}{s^2}$	Meters per Second Squared
N	Newton
$N \cdot m$	Newton Meter
$N \cdot m/s$	Newton Meter per Second
RPM	Revolutions per Minute
RPM/s	Revolutions per Minute per Second
s	Seconds
V	Volts

1 INTRODUCTION

The Eaton Corporation Vehicle Group develops and manufactures a large variety of automotive products aimed at improving vehicle efficiency and performance [1]. Part of the Eaton Vehicle Group product line includes a series of transmissions designed for light, medium, and heavy duty commercial truck applications. These transmissions are offered in automated and manual configurations and are designed to be used in line haul, refuse, agriculture, delivery, bus, and other applications [2]. The Vehicle Group develops the transmission hardware, and in the case of the automated products, the software-based control algorithms as well.

A key component of the development cycle is testing [3]. Within the Vehicle Group, there are different options available for evaluating hardware and software performance of various transmissions. One test method involves installing a transmission in an actual truck and driving it around under various conditions. Truck testing can evaluate the hardware and software, but requires a commercially licensed driver, a truck, and a product mature enough to operate safely in a vehicle. Although truck testing best reflects how the transmission will eventually be used, it also suffers from a lack of repeatability with a human controlling the test, and is subject to the whims of the weather. A truck does provide a large number of system controllers, like an engine controller or Anti-lock Braking System controller, which send messages on the vehicle communications bus. Without these messages, the transmission controller will throw fault codes, and possibly not allow the transmission to operate.

Another assessment method, that does not require a driver or a truck, involves operating the transmission in a test cell. The Vehicle Group test cells are available in several different varieties that allow for different testing scenarios. The test cell configuration most like in-vehicle testing is the diesel cell, which is comprised of a diesel engine and a dynamometer or a large inertia absorber. The engine supplies torque to the transmission while the dynamometer or absorber acts as a load. A couple advantages of the diesel cell include its ability to produce all the same input dynamics as a truck and the expected messages, since there is an actual engine, in addition to producing full rated torque. Also like a truck, a diesel cell is disadvantaged by consuming fuel and producing emissions, but can produce consistent test

result when controlled by an automated system. The diesel cell also has large inertias capable of storing energy that could potentially damage prototype hardware if the hardware were to malfunction. With an engine used to supply torque, there is not a good method of granularly controlling the speed or torque applied to the transmission. For dynamic testing where engine characteristics and full torque are important, the diesel cell is a good choice. For more steady state tests, or those benefitting from less torque, the diesel can be costly or risky.

An alternate test cell, called a motor cell, uses of two electric motors to provide the input torque as well as the load torque. These motors can be a variety of different capacities, allowing for an assortment of speed and torque combinations. Some motors are sized to duplicate the speed and torque values of in-vehicle testing. Other motors are able to match the same speeds of engine dynamics, but are far less capable in their torque capacity. An advantage of testing a transmission with a reduced torque capacity motor is that mistakes are much more easily forgiven. For example, if a new control algorithm malfunctions in a way that would normally damage the transmission, the less capable motor will not be able to supply the same energy required to negatively affect components. The electric motors also do not have the large inertia of an engine, a dynamometer, or an actual truck. The reduced inertia lessens the amount of stored energy that could be transferred to a transmission, also protecting it in the case of a control algorithm error. Electric motors also do not require diesel fuel to operate and produce no direct emissions. Typically, the motor cell provides the user the ability to control the speed or torque settings for the motors, which can be useful for simple tests not requiring truck-like dynamics. To combine the advantages of a motor cell with the dynamics of a diesel cell, an additional system needs to be employed to control the motors with the desired behavior. This thesis focuses on such a system that will allow for vehicle dynamics emulation in a motor cell.

1.1 The Vehicle Simulator

The Vehicle Group headquarters in Galesburg, Michigan has diesel and motor based test cells that allow for a variety of different transmission testing options. To increase testing capabilities, a request was made to develop a system that could be integrated into an existing motor cell,

adding the ability to emulate certain vehicle dynamics. Thus the Vehicle Simulator system was developed, enabling vehicle emulation as well as expanded user control capabilities.

1.1.1 Vehicle Simulator System Specifications

The original requirement for the Vehicle Simulator was to provide sufficient vehicle emulation, so that a transmission installed in a motor cell would act the same as if it were installed in a truck. This system would be used primarily to test automated transmissions, as a manual transmission would need to be shifted by a user or by some other automated shifting system. To meet the vehicle emulation requirement, the Vehicle Simulator would need to send the requisite vehicle J1939 Controller Area Network (CAN) bus messages to the transmission controller, in addition to providing vehicle emulation. After conducting several meetings with potential users of the system, several more requirements were generated around user control options. The following list provides the high level specifications for designing the Vehicle Simulator system.

1. Provide vehicle emulation
    - 1.1. Send required engine and vehicle J1939 messages to the transmission controller
    - 1.2. The input motor will emulate an engine
        - 1.2.1. The motor speed will decrease when the clutch is opened
        - 1.2.2. The torque-speed map will be configurable
        - 1.2.3. The motor will idle at a defined speed when enabled
        - 1.2.4. The motor will respond to a throttle command
    - 1.3. The output motor shall emulate vehicle load dynamics
        - 1.3.1. Road grade
        - 1.3.2. Combined vehicle weight
        - 1.3.3. Air resistance
        - 1.3.4. Rolling resistance
        - 1.3.5. Vehicle brakes
2. Use existing hardware and software components where available
3. Allow the user to control the input and output motors in several different modes

3.1. Vehicle emulation

3.2. Manual motor control

3.2.1. Speed control mode

3.2.2. Torque control mode

4. Provide several different user control options

4.1. Manual motor control through the user interface

4.2. Control of the user interface through a standard scripting language

4.3. CAN message control

4.4. Manual motor control through the throttle and brake pedals

1.1.2 Vehicle Simulator System Realization

To meet the system specifications, the Vehicle Simulator was realized. Hardware capable of running deterministic software plant models with high execution speed while interfacing with the existing test cell infrastructure was acquired.

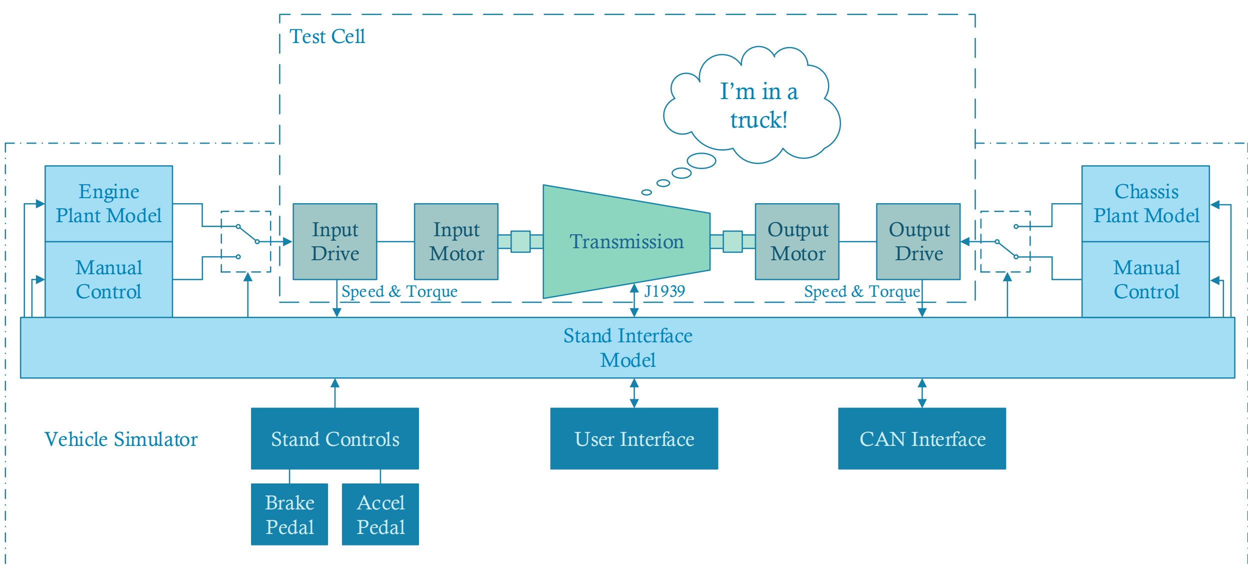

Figure 1.1: Vehicle Simulator high level diagram

Figure 1.1 shows a high level view of the implemented Vehicle Simulator system and how it interfaces with the existing test cell. The entire Vehicle Simulator system is comprised of

several software models, namely the Engine Plant Model, Chassis Plant Model, and Stand Interface Model. In particular, the Stand Interface Model coordinates the various user inputs with the Engine and Chassis Plant models, that then sends the control signals to the input and output motor drives. With this system running in vehicle emulation mode, the transmission believes it is in a truck and is able to dynamically shift gears. The motor speeds and torques are reported back to the Stand Interface Model from the motor drives, and the Engine Plant Model provides the requisite J1939 CAN messages to the transmission controller. The Vehicle Simulator also allows the user to run the input and output motors in manual control mode, directly setting the speed or torque values.

To meet the specification of utilizing existing components, a real-time system was selected from ETAS which was similar to systems already in use for the Hardware-in-the-Loop (HIL) test stands [4]. Using a familiar ETAS system meant there was already local expertise with the hardware and configuration software. Additionally, several MATLAB/Simulink plant models were extracted from the software used on the HIL stands. Namely, the Engine and Chassis Plant models used in the Vehicle Simulator are the same as those on the HIL stand.

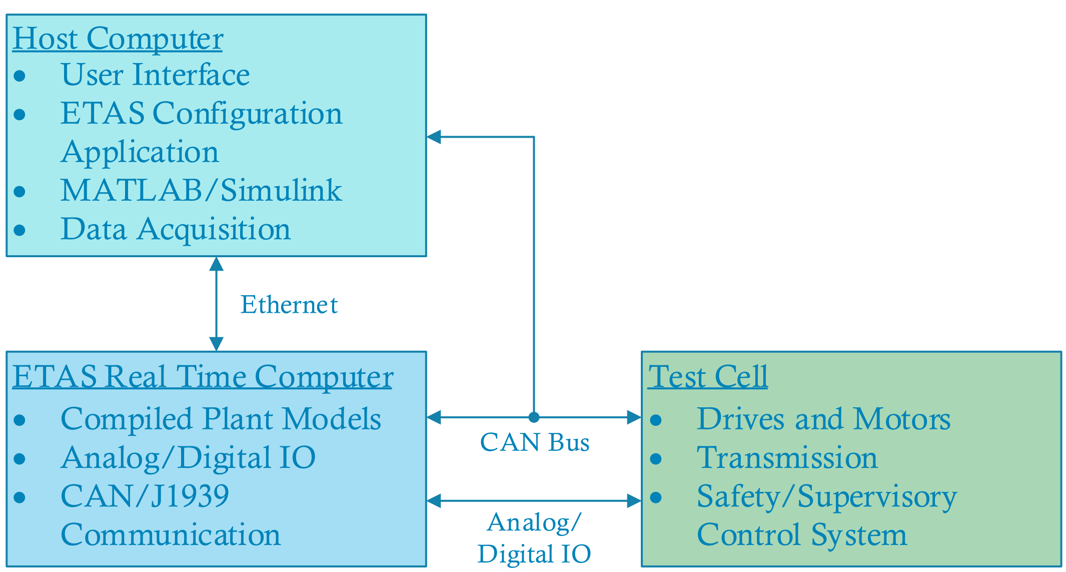

Figure 1.2: Vehicle Simulator system overview

Figure 1.2 shows a high level view of the Vehicle Simulator system as developed with the exiting ETAS hardware and software. There is a host computer running the Microsoft

Windows operating system, connected via Ethernet to an ETAS Real-Time Computer, otherwise known as a Real-Time Personal Computer (RTPC), running the compiled vehicle plant models. The RTPC also contains the hardware required to interface with the existing test cell, and can directly control the motors. The host computer runs the ETAS and MATLAB software necessary for developing the plant models, configuring the system hardware and software connections, and running the user interface. Development and code generation for the plant models is done with MATLAB/Simulink on the host computer, which is then downloaded to the RTPC for compilation into a binary executable. The ETAS configuration application coordinates the code generation and compilation. An additional part of the ETAS software running on the host computer is an automation suite that allows the user to write scripts in C# or Python that can automate the user interface.

1.2 Scope of Original Work

As previously stated, one of the system specifications was to reuse components where available. To meet this specification, a commercially available real-time system was chosen, along with the reuse of existing vehicle plant models. Aside from those system components, the author of this paper mainly contributed to the development of the Stand Interface model (Section 2.2), the Simulated Test Cell model (Section 2.3), the test cell integration strategy (Section 2.1), and the integration of all the system components. The Stand Interface model is effectively the foundation of the entire Vehicle Simulator system, combining all of the different user control options with the vehicle plant models while making sure the system operates safely. The Stand Interface model also scales all of the analog input and output signals from voltages to useable engineering units. The author also developed the system user interface, along with examples of how to control the Vehicle Simulator through external commands from the CAN bus. The Simulated Test Cell allows a developer to try out system changes in pure simulation before running in the test cell. Executing in simulation first will help insure the changes do not cause any adverse system operation. To integrate the system with the test cell, the author also created schematics that detailed the electrical connections from the Vehicle Simulator input and output

hardware to the test cell motor drives and monitoring system. The actual test cell wiring was completed by talented electrical technicians.

1.3 Literature Review

Others have seen the advantages of test cell based vehicle simulation and implemented similar systems. Such systems employ the same strategy of testing a physical transmission by way of engine and vehicle models controlling drive and load motors. Each system is slightly different, but each follows the same overall strategy of model based transmission testing. In each, engine and vehicle models run on commercially available real-time targets while motors apply torques to the input and output of the physical transmission.

Ahlawat et al. [5] describe a similar system for testing a front wheel drive transmission using one input motor and two load motors so wheel slip can be modeled in addition to normal driving situations. Their system can emulate engine torque pulses while running on commercially available hardware, and is verified with measured data over a single standard drive cycle. Diesel engines are not immediately addressed, but mention is made the system can easily be adapted to other propulsion methods. The engine model has the ability to communicate over a CAN bus with the transmission controller, allowing for the actual transmission controller to be incorporated in the testing.

A flexible system that can be used to test hybrids as well more traditional transmissions is detailed by Steiber et al. [6]. Vehicle simulation is accomplished with an AC motor as the transmission input and a DC motor as the output. Their system is comprised of a series of hardware and software nodes that work together to simulate an entire vehicle. The engine model can simulate several different power sources, such as an internal combustion engine, or gas turbine, while the transmission node can handle manual, automatic, or continuously variable transmissions. The system also allows for energy storage models and hardware for hybrid testing. To demonstrate system operation, a 4L60E automatic transmission found in GMC cars and trucks was verified with measured data. They incorporate a generic transmission controller that allows users to manually control the transmission using switches and dials, or automatically through production control algorithms, while handling CAN

communications. Test schedules that define the grade, vehicle speed, brake commands, and shift state can be executed by a cycle model, which feeds into the driver model and allows the vehicle simulation to operate in an automated fashion.

Castiglione et al. [7] outline a sophisticated transmission testing system employing electric motors connected to the input and output of a physical transmission. The entire test rig is capable of tilting +/- 45 degrees to simulate extreme grades while also heating or cooling the transmission to simulate environmental factors. The engine model is capable of reproducing torque pulses up to 250 Hz and inertia down to $0.042\ \mathrm{kg} \cdot \mathrm{m}^2$. The system model equations allow for the addition of a trailer to be configured in the vehicle simulation, defined through a graphical user interface (GUI). Prior to testing the user also defines a wide range of vehicle parameters through the GUI as well, such as tire radius, frontal area of the vehicle, vehicle weight, trailer weight, road rolling resistance, engine map file location, and etc. In manual operating mode, the GUI provides control of the throttle and brake levels, or engine RPM set points. For automatic operation, the user can upload a test schedule file that the driver model uses to control the vehicle. The test schedule contains parameters like time, vehicle speed, throttle position, grade, elevation, road surface coefficient, gear, and ambient temperature. The system was verified against gasoline engines ranging in capacity from 2.2 to 6.0 liters, and compared against actual vehicle data. The entire system is monitored for safety conditions, and can shut down the test if necessary.

Others have made progress in providing the user with a realistic experience while simulating an entire vehicle, as described by Allen et al. [8]. Their system provides an in depth user experience with physical pedals, a force feedback steering wheel, and 3D rendered environments. Even though they are using dated equipment compared to today's standards, their full vehicle simulator performs adequately and they theorize it could be used in real-time hardware testing. An example is presented describing the simulation of a truck with a trailer which is able to be jack knifed with appropriate maneuvering.

Boissinot et al. [9] describe a system developed by AVL Powertrain Engineering Inc. that simulates a vehicle with real-time dynamic models and drives a powertrain with low inertia dynamometers. The system is also able to automatically run many different vehicle maneuvers,

adjusting the powertrain calibration parameters for each test. Shift quality data is collected and used to determine the ideal powertrain calibration values, prior to any in-vehicle testing. For several of the maneuvers, the shift quality was calculated for both the simulated vehicle and an actual vehicle operating at a test track. Once confidence had been established that the simulated vehicle would provide adequate results, other maneuvers were performed to generate shift quality data from the simulated vehicle.

The system developed by Li et al. [10] provides vehicle emulation to an automated manual transmission using a drive dynamometer connected to the transmission input and load dynamometer attached to the output. The torque generated by the engine model is comprised of a mean torque, inertia torque, and combustion torque, while the drive dynamometer is capable of producing the same fully rated speed and torque as seen in the vehicle with the actual engine. The engine mean torque is based on a lookup table, with the inertia and combustion torques based on presented mathematical equations. The road load model is based on vehicle mass, aerodynamic resistance, road grade, and rolling resistance. Details are also provided on the drive control strategy, and data is presented showing successful dynamic shifting of the transmission.

Although the previously mentioned articles do not specifically call out diesel engines for use in their models, others have done work in the advancement of modeling diesel engines for use in real-time HIL tests. The previous articles mainly use a lookup table or engine map strategy for modeling the engines, while other more detailed models exist. Guvenc et al. [11] provide a mean value engine model for a diesel engine and test it using a simplified engine controller. Isermann et al. [12] present a diesel engine model combining superimposed cylinder models and a turbo charger model, then compare it to actual test data. Their advanced model helps testing situations of increased vehicle dynamics. They also use a simplified vehicle model for complete HIL tests of a couple of vehicle control units, with the addition of a user interface that includes physical brake and accelerator pedals for user interaction. Another advanced diesel engine model is presented by Zhang et al. [13]. For their model the diesel engine is decomposed with fine granularity into its subcomponents, then simulated with a mean value

model, in-cylinder model, and neural network NOx emission estimator. This model is verified for use as part of a HIL system, and is verified against several real diesel engines.

2 IMPLEMENTATION STRATEGY

The implementation strategy brings together all of the individual system components into a functioning entity. The abstract concepts of software models running on a real-time system meet the realities of physical motors and vehicle hardware. The Vehicle Simulator is integrated into the existing test cell, and a user interface is designed for control.

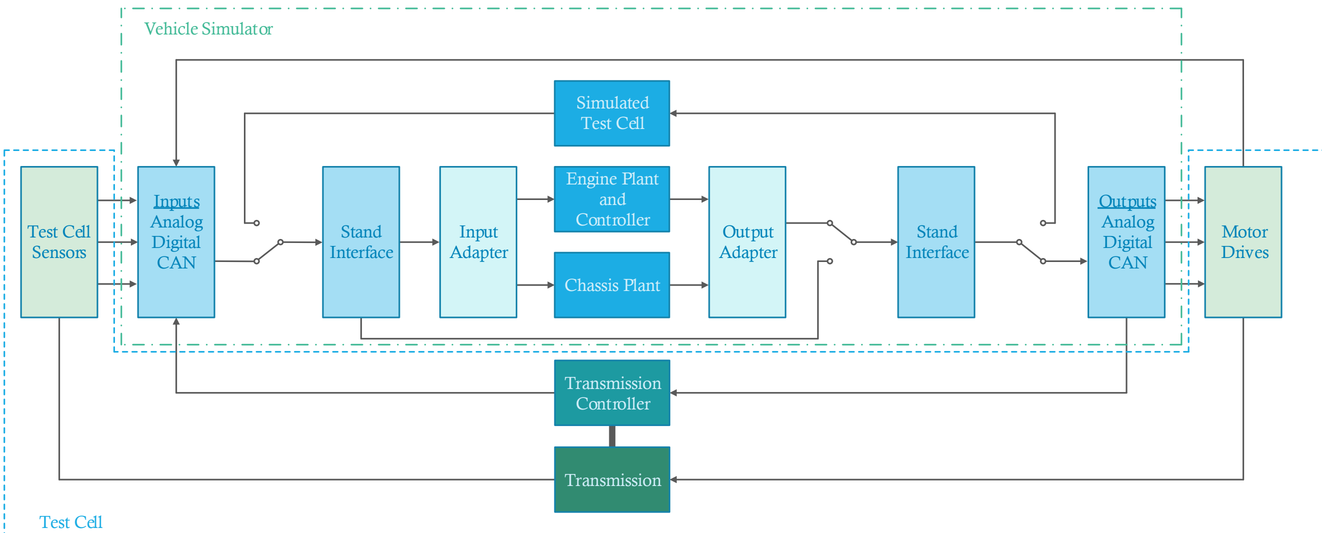

Figure 2.1: Vehicle Simulator system diagram

Figure 2.1 shows a high level overview of how the components of the implementation strategy fit into the overall system, and the distinction between the Vehicle Simulator and the test cell. First, the physical input/output interface between the Vehicle Simulator and the test cell will be discussed, followed by an explanation of the Stand Interface model. Then details about the Simulated Test Cell will be given, which allows for offline development and testing without the test cell being operational. The last part of the Implementation Strategy will focus on the Vehicle Plant model, which includes the Engine Plant, Chassis Plant, as well as the input and output adapters. In all, the implementation strategy pulls together the individual components into a complete system, ready for verification and real-world testing.

2.1 Vehicle Simulator and Test Cell Interface

At the very base level, the Vehicle Simulator interfaces with the test cell through a variety of analog and digital inputs and outputs. To simplify matters, the main focus will be on the input and output from the point of view of the Vehicle Simulator. First, focus will be given to an overview of the entire system, followed by an in depth discussion on the Vehicle Simulator system and its input and output specifications.

2.1.1 High Level System Overview

Various connections interface the Vehicle Simulator and the test cell, all with a different purpose and function. Here, the term test cell describes both the Programmable Logic Controller (PLC) and the Motor Drives. To further reduce system complexity in this overview, the term Motor Drive is comprised of the input motor drive and output motor drive, which are connected to the input shaft and output shaft of the transmission by way of their respective motors.

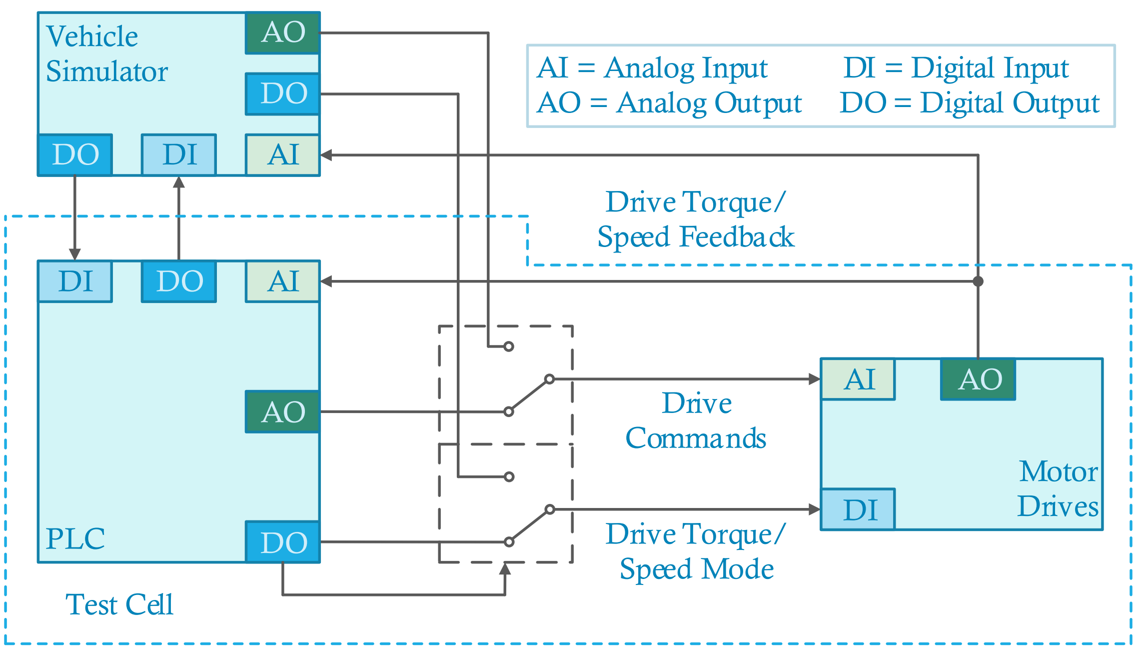

Figure 2.2: Vehicle Simulator and test cell signal routing

Figure 2.2 shows the routing of the analog and digital signals between the various system components. Also shown is the distinction between test cell and Vehicle Simulator. Several

switches, controlled by the PLC, physically swap the control signals from either the PLC or the Vehicle simulator to the Motor Drives.

2.1.1.1 Motor Drive Overview

Regardless of the control signal source, each Motor Drive has two analog inputs, one for speed and another for torque. The active input is set by the Motor Drive mode digital input, which can be either speed control or torque control mode. In speed control mode, the drive will adjust its torque in order to maintain the requested speed. Likewise, in torque mode, the drive will adjust its speed to maintain the desired torque. The Motor Drive speed and torque feedback signals are also sent back to the PLC and the Vehicle Simulator in parallel.

The Motor Drive set point and feedback signals follow the same voltage sign convention to indicate direction. Clockwise (CW) and counterclockwise (CCW) rotation are defined as the shaft direction when looking at the face of the motor, as shown in Figure 2.3 [14]. For the Motor Drives, a CW rotation speed results in a positive voltage, while a CCW speed is a negative voltage. Similarly, torque acting in the CW direction is positive, while CCW is negative.

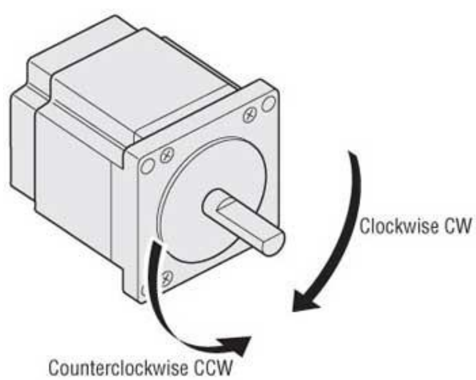

Figure 2.3: Motor rotation convention

To further complicate matters, the input motor shaft is connected to the output motor shaft by way of the transmission, making the shaft directions opposite of one another. For example, when the transmission is in a forward gear and the input motor rotates in a CCW direction, the output motor will turn in a CW direction. The normal convention of engine flywheel rotation is CCW when looking at the back of the engine [15]. To stay with normal

engine rotation convention, the input motor normal rotation values will be negative, and the output motor normal values will be positive.

2.1.1.2 PLC Overview

The overall stand supervisory control comes by way of the PLC, which toggles the Motor Drive speed and torque control signal switch and monitors the Motor Drive feedback for fault conditions. The user will set appropriate speed and torque operating limits for the Motor Drives operation. If any of the limits are surpassed, a fault condition occurs. When a fault occurs, the PLC will disable the Motor Drives, removing torque and allowing them to coast down to zero speed. Then, the PLC will not allow the drives to be enabled again until the fault has been cleared. Since the drive feedback signals are paralleled to the PLC and the Vehicle Simulator, the PLC will still monitor and shutdown the drives even when the Vehicle Simulator is in control. A user interface is provided with the PLC which allows a user to control the motors in speed or toque modes, shutdown the cell, set limits, and switch over to Vehicle Simulator control mode.

2.1.2 Vehicle Simulator Interface

The Vehicle Simulator input/output functionality lends itself to several PCIe cards that reside in the chassis of the RTPC and connect to the test cell through screw terminal breakout boards.

Vehicle Simulator Input/Output Cards			
Slot	Type	Card	Function
1	PCIe 4x	ADDI-DATA APCIe-3121-16-8	Multifunction Input/Output
2	PCIe 16x	ADDI-DATA APCIe-3121-16-8	Multifunction Input/Output
3	PCIe 1x	ADDI-DATA APCIe-2200-16-16	Digital Input
4	PCIe 8x	Spare/2200 Connector	Relay Form C Output
5	PCIe 1x	CAN-IB200	2 Port CAN
6	PCIe 4x	CAN-IB200	2 Port CAN
7	PCI	Spare	

Table 2.1: Vehicle Simulator expansion card utilization

Table 2.1 outlines the input/output card layout in the RTPC chassis, as well as their respective functionality. All of the PCIe cards take up a 1x slot, and can be moved to interchangeably to

any of the available PCIe slots. The APCIe-2200 card takes up one entire slot with the actual card, and then an additional adjacent slot opening for its second connector, as shown in Figure 2.4 [16]. Also shown in the figure are several German terms directly following the equivalent English description.

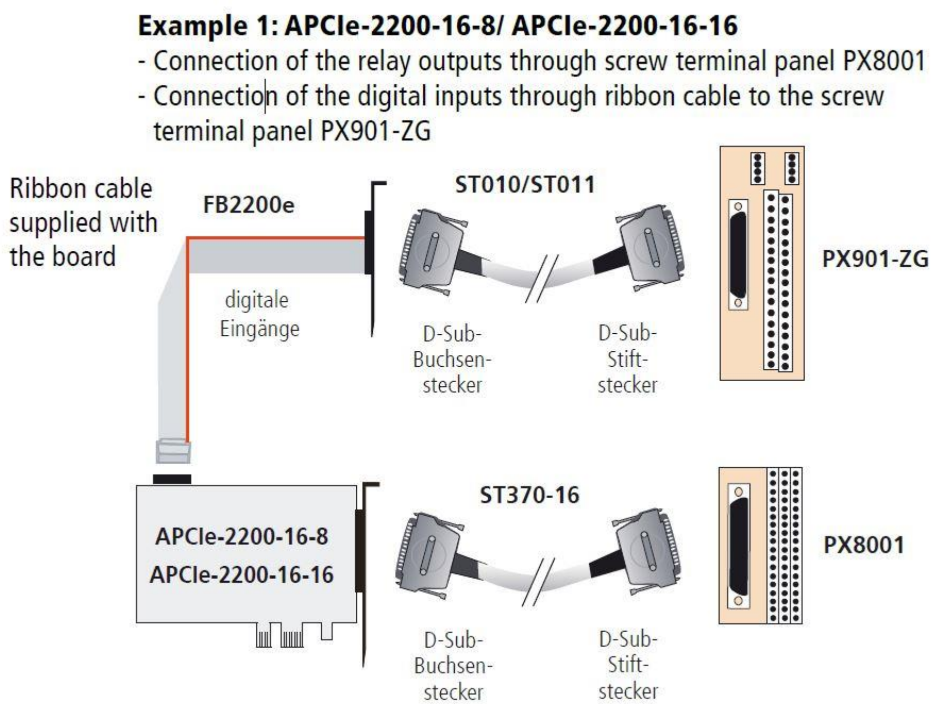

Figure 2.4: APCIe-2200 card layout

With the adjacent slot opening taken up, the slot is unusable for an additional card. The two CAN-IB200 cards provide four high speed CAN ports. The four ports of the CAN cards are reserved as follows: J1939 communication to the transmission controller, proprietary high speed CAN bus communication also to the transmission controller, Vehicle Simulator control and feedback communication, and a spare.

ADDI-DATA APCIe-3121-16-8			
Type	Direction	Count	Specification
Analog	Input	8 Differential or 16 Single Ended	+/- 10 Volts
	Output	8 Single Ended	+/- 10 Volts
Digital	Input	4	24 Volt Logic
	Output	4	24 Volt High Side Driver

Table 2.2: APCIe-3121 specifications

ADDI-DATA APCIe-2200-16-16			
Type	Direction	Count	Specification
Digital	Input	16	24 Volt Logic
	Output	16	Form C Relay

Table 2.3: APCIe-2200 specifications

Table 2.2 and Table 2.3 describe the details of the Vehicle Simulator input/output cards. Two of the ADDI-DATA APCIe-3121-16-8 cards are incorporated to cover all of the analog input/output requirements of interfacing with the Motor Drives, as well as providing several spare channels. The ADDI-DATA APCIe-2200-16-16 cards handle the majority of the digital input/output signals, while also providing several spare channels for possible future expansion. All of the RTPC cards combined form the basis of the Vehicle Simulator input and output functionality.

2.1.2.1 Vehicle Simulator Inputs

The Vehicle Simulator receives a variety of signals from both the PLC and the Motor Drives through its various cards. These signals aid in monitoring the test cell operation as well as allowing for user control of the Vehicle Simulator.

Number	Type	Card	Connection	Function
1	Analog	APCIe-3121 Card 1	Input Motor Drive	Speed Feedback
2	Analog	APCIe-3121 Card 1	Input Motor Drive	Torque Feedback
3	Analog	APCIe-3121 Card 1	Output Motor Drive	Speed Feedback
4	Analog	APCIe-3121 Card 1	Output Motor Drive	Torque Feedback
5	Analog	APCIe-3121 Card 2	User Console	Throttle Pedal Position
6	Analog	APCIe-3121 Card 2	User Console	Brake Pedal Position
7	Analog	APCIe-3121 Card 2	Stand Instrumentation	Output Motor Speed
8	Analog	APCIe-3121 Card 2	Stand Instrumentation	Input Motor Speed
9	Analog	APCIe-3121 Card 2	Stand Instrumentation	Output Motor Torque
10	Digital	APCIe-2200	PLC	Test Cell Running Indicator
11	Digital	APCIe-2200	PLC	Vehicle Simulator Selected Indicator

Table 2.4: Vehicle Simulator inputs

To make use of the analog inputs, their ranges and scaling need to be known. The scaling is what will transform the measured signal voltage, into useful engineering units like Revolutions Per Minute (RPM) and Newton Meters (N·m).

Number	Input Motor Drive		
1	Speed Feedback	-10V = -3000 RPM	10V = 3000 RPM
2	Torque Feedback	-10V = -257.6 N·m	10V = 257.6 N·m

Table 2.5: Input motor analog feedback scaling

Number	Output Motor Drive		
3	Speed Feedback	-10V = -4000 RPM	10V = 4000 RPM
4	Torque Feedback	-10V = -859.6 N·m	10V = 859.6 N·m

Table 2.6: Output motor analog feedback scaling

The input and output motor drive feedback signals in Table 2.5 and Table 2.6, follow the same motor polarity as described in Section 2.1.1.1 Motor Drive Overview, not the engine polarity. All feedback signals are filtered with a single pole, 1 millisecond time constant, low pass filter internal to the Motor Drives.

Number	User Console		
5	Throttle Pedal	1V = 0% Depressed	4V = 100% Depressed
6	Brake Pedal	1V = 0% Depressed	4V = 100% Depressed

Table 2.7: User console pedal analog scaling

Table 2.7 show the scaling for the throttle and brake pedals at the test cell user console. At the core of each pedal is a potentiometer that is supplied by a 5 V source. Each pedal does not make full mechanical use of the potentiometer, which is why the range is reduced from 1 V to 4 V.

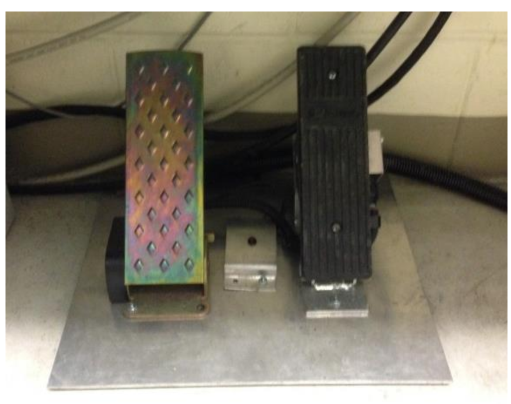

Figure 2.5: Test cell user console pedals

Figure 2.5 shows the actual pedal as installed at the user console of the test cell.

Number	Stand Instrumentation		
7	Output Motor Speed	-5V = -4000 RPM	5V = 4000 RPM
8	Input Motor Speed	-5V = -4000 RPM	5V = 4000 RPM
9	Output Motor Torque	-10V = -451.91 N·m	10V = 451.91 N·m

Table 2.8: Stand instrumentation scaling

Table 2.8 shows the scaling of the test cell speed and torque sensors. This instrumentation operates independently of the Motor Drive speed and torque feedback signals. The torque sensor is a Himmelstein dual range transducer, but only the larger range is used. Again, the same motor signal polarities as the drive feedback signals are employed.

Number	PLC		
10	Test Cell Running	0V = Test Cell Not Running	24V = Test Cell Running
11	Vehicle Simulator Selected	0V = Vehicle Simulator Not Selected	24V = Vehicle Simulator Selected

Table 2.9: PLC feedback signals

The digital signals outlined in Table 2.9 are from the PLC, and give the Vehicle Simulator some indication to the state of the test cell. The Test Cell Running signal will be 24 V when the user has enabled the test cell, which also enables the Motor Drives. The Vehicle Simulator Selected signal is 24 V when the user has selected the Vehicle Simulator for Motor Drive control through the PLC user interface. This signal also indicates that the physical Motor Drive command switches have moved from the PLC, to the Vehicle Simulator.

2.1.2.2 Vehicle Simulator Outputs

The Vehicle Simulator contains several outputs that allow for Motor Drive control, as well as interacting with the PLC. The system outputs come by way of the same PCIe cards previously outlined in Table 2.1.

Number	Type	Card	Connection	Function
1	Analog	APCIe-3121 Card 1	Input Motor Drive	Speed Setting
2	Analog	APCIe-3121 Card 1	Input Motor Drive	Torque Setting
3	Analog	APCIe-3121 Card 1	Output Motor Drive	Speed Setting
4	Analog	APCIe-3121 Card 1	Output Motor Drive	Torque Setting
5	Digital – Solid State	APCIe-3121 Card 1	Watchdog/PLC	Heartbeat Pulse
6	Digital - Relay	APCIe-2200	Input Motor Drive	Torque/Speed Mode
7	Digital - Relay	APCIe-2200	Output Motor Drive	Torque/Speed Mode
9	Digital - Relay	APCIe-2200	PLC	Test Cell Shutdown

Table 2.10: Vehicle Simulator outputs

Table 2.10 shows the entire list of Vehicle Simulator outputs, which card they originate from, and what function they play in the overall system.

Number	Input Motor Drive		
1	Speed Setting	-10V = -3000 RPM	10V = 3000 RPM
2	Torque Setting	-10V = -257.6 N·m	10V = 257.6 N·m
6	Torque/Speed Mode	0V = Speed Control	24V = Torque Control

Table 2.11: Input motor drive analog command scaling

Number	Output Motor Drive		
2	Speed Setting	-10V = -4000 RPM	10V = 4000 RPM
3	Torque Setting	-10V = -859.6 N·m	10V = 859.6 N·m
7	Torque/Speed Mode	0V = Speed Control	24V = Torque Control

Table 2.12: Output motor drive analog command scaling

Both Table 2.11 and Table 2.12 give the analog command scaling from the Vehicle Simulator to the Motor Drives. Much like the analog feedback signals, the polarity is seen from the motor point of view, not the engine polarity. The Torque/Speed Mode signal changes the drive control

mode between torque and speed mode, as described in Section 2.1.1.1 Motor Drive Overview. Whichever mode is active, the opposite input is set to 0 V. For instance, if the Motor Drive is in speed mode, then the torque analog output will be set to 0 V.

Number	PLC and Watchdog		
5	Heartbeat Pulse	No Pulse = Vehicle Simulator Not Running	Pulse = Vehicle Simulator Running
9	Test Cell Shutdown	0 V = Test Cell Active	24V = Shutdown Test Cell

Table 2.13: Vehicle Simulator test cell outputs

The Heartbeat Pulse signal in Table 2.13 is a 24 volt high side driver that comes from the APCIe-3121 and connects to a watchdog timer. When the Vehicle Simulator is actively running, the Heartbeat Pulse toggles the output to the watchdog timer at a rate of 1 Hz. When the Vehicle Simulator is not running, there is no pulse, and a contact opens on the watchdog timer after 5 seconds. When the watchdog contact opens, the PLC is alerted that the Vehicle Simulator is not ready for operation. Since the Heartbeat Pulse will be switching for long periods of time, a solid state output was chosen for a longer life span, as opposed to a mechanical relay output on the APCIe-2200 card. The Test Cell Shutdown signal, when at 24 V, will signal the PLC that the stand needs to be shut down. When the PLC receives the shutdown signal, it will disable the Motor Drives, as described in Section 2.1.1.2 PLC Overview.

2.2 Stand Interface Model

The Stand Interface Model plays the critical role of tying the physical world of the test cell, together with the simulated world of the Vehicle Plant models, and is the source of much of the diverse functionality found the in Vehicle Simulator. Figure 2.1, the overall system diagram, shows the Stand Interface model placed before and after the plant models, but skips over the detailed functionality. What follows is a high level overview of some of the Stand Interface Model features that will be covered in greater detail later in this section.

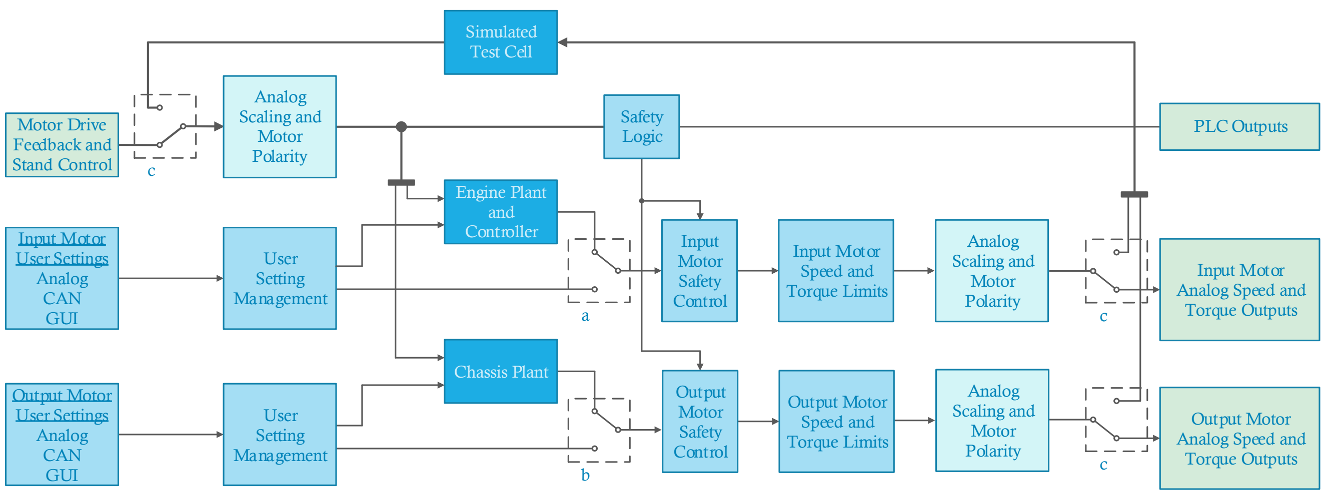

Figure 2.6: Stand Interface model

Figure 2.6 shows details of the signal routing and the functionality of the Stand Interface model. Here, the external components are the Simulated Test Cell, Engine Plant and Controller, and the Chassis Plant; all other blocks are contained within the Stand Interface model. Some of the critical functionality of the Stand Interface model is as follows:

- Switching between manual and plant motor control modes;
- Managing and switching between various user control methods;
- Scaling analog signals from voltage to engineering units;
- Adjusting motor polarity to match engine polarity;
- Safety monitoring; and
- Bypassing all the physical inputs and outputs to the Simulated Test Cell.

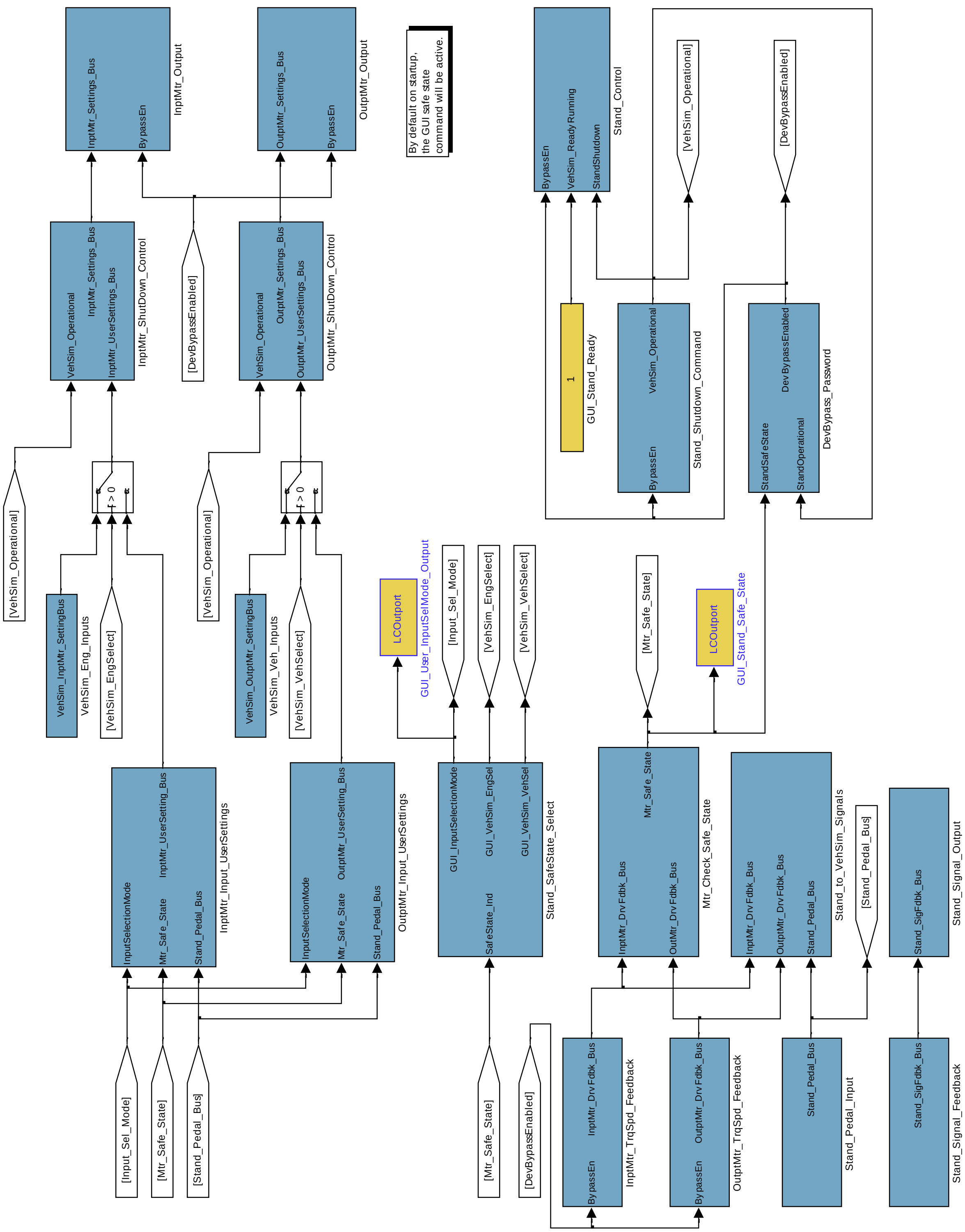

Figure 2.7: Stand Interface Simulink model

Figure 2.7 shows the Stand Interface Simulink model in its entirety, which Figure 2.6 represents in a simplified manner. For the remaining sections, greater detail of this Simulink model will be provided, typically showing subsections at a deeper level than the overall model.

2.2.1 Motor Function Selection

The general flow of the Stand Interface model commences with the input and output motor drive setting outputs, found at the right side of Figure 2.6. The Motor Drive commands can source from either the plant models, or the manual Motor Drive user settings, using software switches *a* and *b* of Figure 2.6. Regardless of the source, those Motor Drive settings are subject to speed and torque limits, as well as safety shutdown controls. Switches *a* and *b* are able to independently switch the Motor Drive control from manual user control, to plant model control. In the case of the input motor, the Engine Plant could be enabled with switch *a*. For the output motor, switch *b* will change over to the Chassis Plant model.

Keeping switches *a* and *b* independent gives the user a great amount of flexibility for utilizing the Vehicle Simulator system. In one instance, the user could put the input motor in Engine Plant mode, and the output motor in manual torque control mode. In another situation, the input motor could be in manual control mode, and the output motor could be set to Chassis Plant mode. Setting both motors to the same mode is also possible. This way, the Vehicle Simulator can cover a larger variety of testing situations, from simple durability tests, to full vehicle emulation.

2.2.2 User Input Mode Selection

When a Motor Drive is in manual control mode, the Stand Interface allows the user to set the Motor Drives in either speed or torque mode. In whichever mode, the user has the ability to request an appropriate speed or torque setting, and the motor will react as described in Section 2.1.1.1 Motor Drive Overview. The speed and torque settings can be requested through the stand pedals (analog), CAN command messages, or through the Vehicle Simulator user interface (GUI).

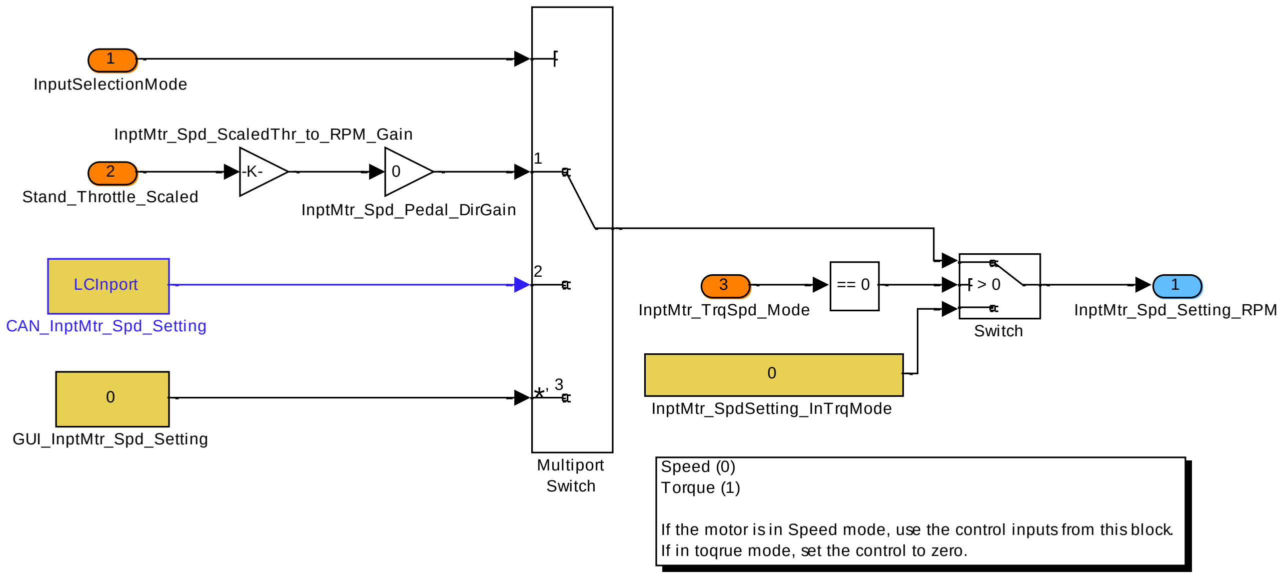

Figure 2.8: Stand Interface user selection mode

Figure 2.8 shows how the user input selection mode works for the input motor speed setting. This same template is copied for the input motor torque setting, as well as the output motor speed and torque settings. Whichever user input mode is selected, the second switch block before the output clamps the setting at zero if that mode is not selected. In the case of the input motor speed setting Figure 2.8, if the input motor is in torque mode, the speed setting output will always be zero regardless on the user speed setting.

The first Multiport Switch is used to select which speed signal is passed on to the rest of the Stand Interface model. The InputSelectionMode input comes from a user input elsewhere in the Stand Interface model, and is routed to several other blocks.

InputSelectionMode Value	User Setting
1	Analog value from the user console pedal input
2	CAN Command
3	User interface value - default

Table 2.14: Input selection mode

Table 2.14 shows the three possibilities for user control, which will be discussed in further detail. Whichever input control method is selected, it applies to both the input motor and the output motor. For instance, if both Motor Drives are set to manual control mode and CAN control is selected, a user cannot control the input motor with the stand pedals, and the output motor with CAN commands. The control mode is universal across both motors. This was done to avoid the confusion of keeping track of the individual motor control modes.

2.2.2.1 Analog Pedal Control Input

If the CAN command or GUI modes are selected, the values are already scaled in the appropriate engineering units. For either of the analog user console pedal inputs, the signal will enter this block scaled from 0% to 100%, and will be converted to the engineering value using a gain block. A 100% pedal input is equivalent to the maximum torque or speed setting, depending on the selected motor mode. Also, the throttle pedal is always used to control the input motor, while the brake pedal is used to control the output motor. In the case of the input motor speed setting, 100% throttle pedal would command the input motor to 3000 RPM, if in speed control mode.

Since the pedals can only operate in one direction, and the motors can run CW or CCW, there is an additional gain block after the pedal scaling to indicate direction. This direction gain could also be used to reduce the command value when the pedal is pressed to 100%. For example, the output motor direction gain could be set at -0.2; then, when in torque mode, this would only request 20% of full rated torque in the CCW direction. The value of the direction gain, like almost all the model parameters, is able to be set in the user interface, on the fly, during runtime.

2.2.2.2 CAN Command Control Input

The CAN command originates from a block called LCInport in Figure 2.8, which is used by the ETAS LABCAR software as a hook into the Simulink model. In this case, the LCInport is tied to a signal in a message of the CAN software module in the LABCAR configuration utility, which handles all the CAN communication overhead.

Name	Message	Startbit	Length [Bit]	By...	Value Type	Initial Value	Factor	Offset	Minimum	Maximum	Unit
InptMtr_Spd_Setting_RPM	StandInt_Rx_InptMtr_Settings	0	16	Intel	Signed	0	0.0915555	0	-3000.09	3000	RPM
InptMtr_Trq_Setting_Nm	StandInt_Rx_InptMtr_Settings	16	16	Intel	Signed	0	0.00793481	0	-260.008	260	Nm

Figure 2.9: CAN command message layout

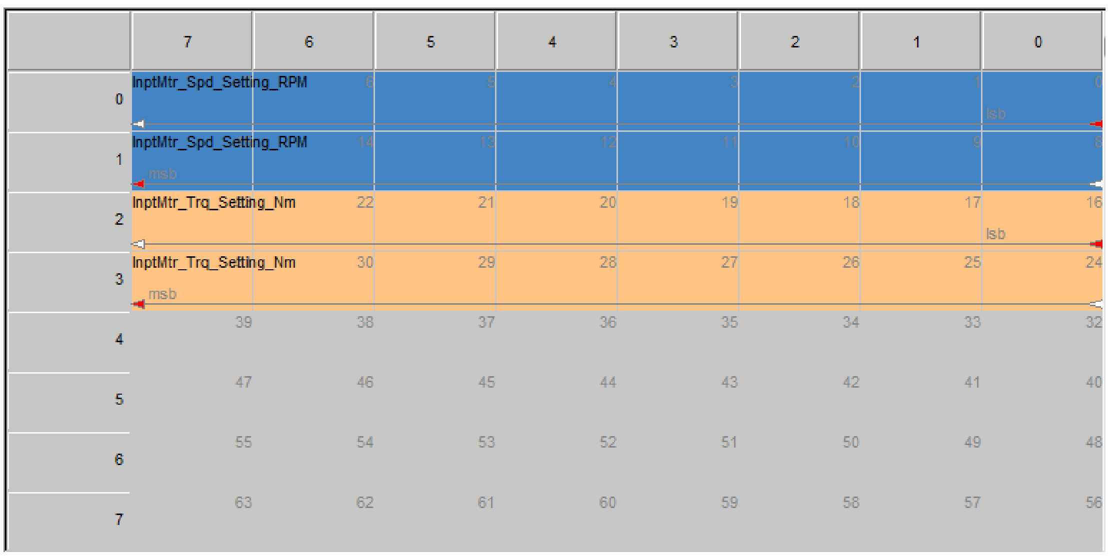

Figure 2.10: CAN command bit layout

Figure 2.9 and Figure 2.10 show the input motor CAN message layout and bit layout, respectively. As can be seen, one CAN message contains both the speed and torque values. Although both the speed and torque are sent, only one setting that corresponds to the motor mode is utilized.

2.2.2.3 User Interface Control Input

The user interface, or GUI, can also be used to control the motor speed and torque settings. The LABCAR utility used to create the GUI is able to access and change most any Simulink parameter. Once configured, the user will have several virtual instruments available within the

GUI that can be used to change the parameters, like text boxes or numeric sliders. For the user interface control input, a simple constant block is employed, which allows access to its one parameter through the GUI. In the case of Figure 2.8, that parameter is the speed setting.

With an additional piece of LABCAR software, Python or C# scripts can be written to automatically change any available Simulink parameter within the model. This would allow the script to adjust the user interface control input, among any other available Simulink parameters, when executed. When properly configured, a script would have capability to automatically control the torque or speed command of both the input and output motors. This automated capability lends itself well to long duration durability testing, where a repetitive test is performed for many cycles, sometimes continuously for weeks on end.

2.2.3 Analog Signal Conditioning

The Vehicle Simulator interacts with the test cell through a series of inputs and outputs as outlined in Second 2.1.2 Vehicle Simulator Interface. In particular, the analog inputs and outputs of the ADDI-DATA APCIe-3121 card interact with the test cell as +/-10 volt signals. These signals need to be scaled into engineering units before their represented signals can be used in the rest of the Stand Interface, and later the Engine and Chassis Plant models. Similarly, outputs from the Stand Interface also need to be scaled from engineering units to voltages, before they can be used to control the Motor Drives.

2.2.3.1 User Console Pedal Scaling

The user console throttle and brake pedals both send signals to the Stand Interface, scaled as described in Section 2.1.2.1 Vehicle Simulator Inputs. These signals are connected from the APCIe-3121 card to the Stand Interface model through the LABCAR configuration utility application. In the Stand Interface model, the signals are connected to an LCInport source blocks, as a floating point representation of their analog voltage. The user console pedal scaling block takes care of scaling the raw analog voltage to a useful percentage. Since the analog signals contain a small amount of noise, the block also imposes a dead band on the analog input to filter the noise, before the scaling is applied. Without the dead band, the analog noise would

be scaled into throttle and brake pedal signals, potentially causing unintended system actuation.

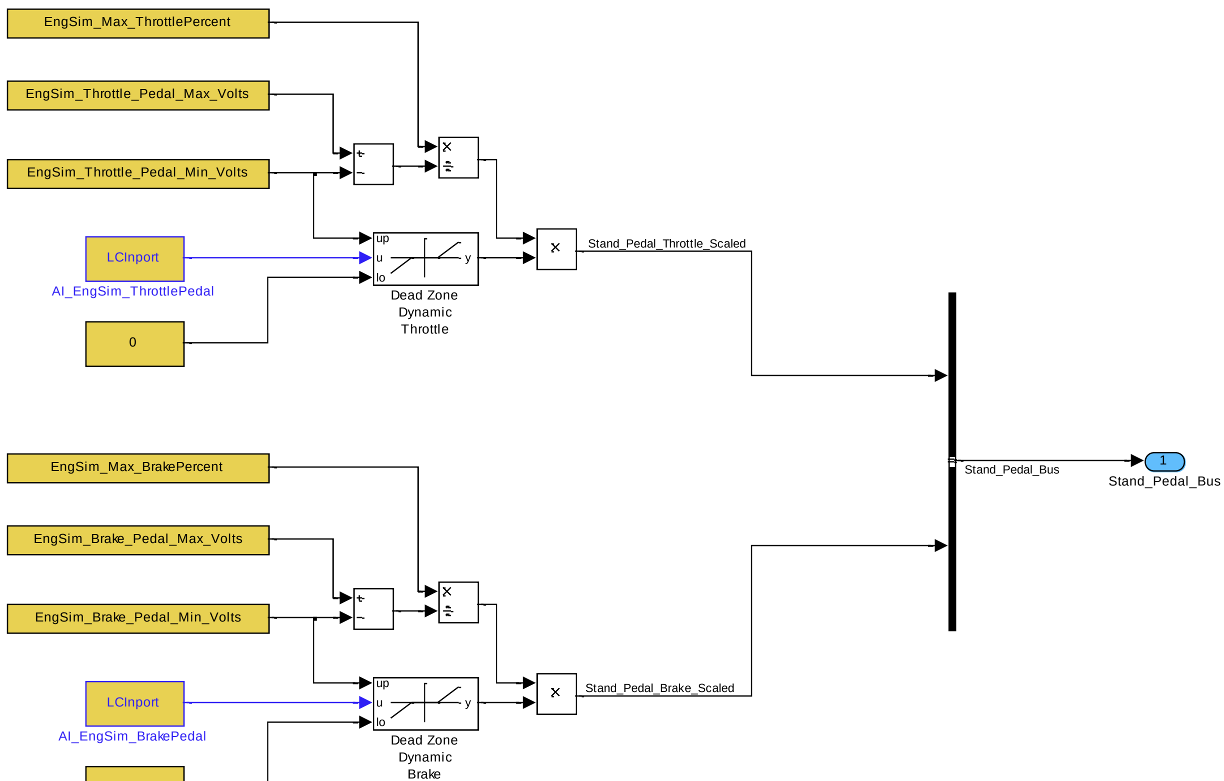

Figure 2.11: User console pedal scaling

Figure 2.11 shows the scaling algorithm for the throttle pedal and brake pedal. The signals originate from the LCInport and are scaled before being combined on a bus. The pedal scaling is

$$PedalPercent(v) = \begin{cases} 0 & if\ v < V_{min} \\ \left(\frac{Percent_{Max}}{V_{max} - V_{min}}\right) v & if\ v \geq V_{min} \end{cases} \tag{2.1}$$

where the variables defined in Table 2.15.

$PedalPercent(v)$	Scaled pedal percentage
$Percent_{Max}$	The maximum percentage value, typically 100%
V_{max}	The measured voltage when the pedal is completely depressed
V_{min}	Upper bound of the dead band voltage, just higher than the voltage when the pedal is not pressed plus signal noise
v	Pedal voltage as read at the analog input

Table 2.15: User console pedal scaling variables

In Figure 2.11, the constant block parameters correspond to the $Percent_{Max}$, V_{max}, and V_{min} variables of Table 2.15, which are all defined at build time in a MATLAB M-File. During runtime, the user has access to all the parameters through the LABCAR user interface. This allows the user to reduce the maximum pedal percentage, or adjust for slight changes in the pedal output voltage. For instance, the user could set the maximum pedal percentage to 20%, so when the pedal is fully depressed, the scaled output is only 20%, and not 100%. This would give a much more granular control over the motor setting or plant model.

2.2.3.2 Motor Drive Feedback Scaling

The Motor Drive feedback signals are also read in as raw analog voltages and require scaling to engineering units before further use in the Stand Interface model. In addition to the scaling, the signals also need to be adjusted to account for the motor polarity and engine polarity difference as described in Section 2.1.1.1 Motor Drive Overview. With the polarity adjusted, the user can interact with the Vehicle Simulator in terms of engine polarity, without needing to adjust for the actual motor polarity.

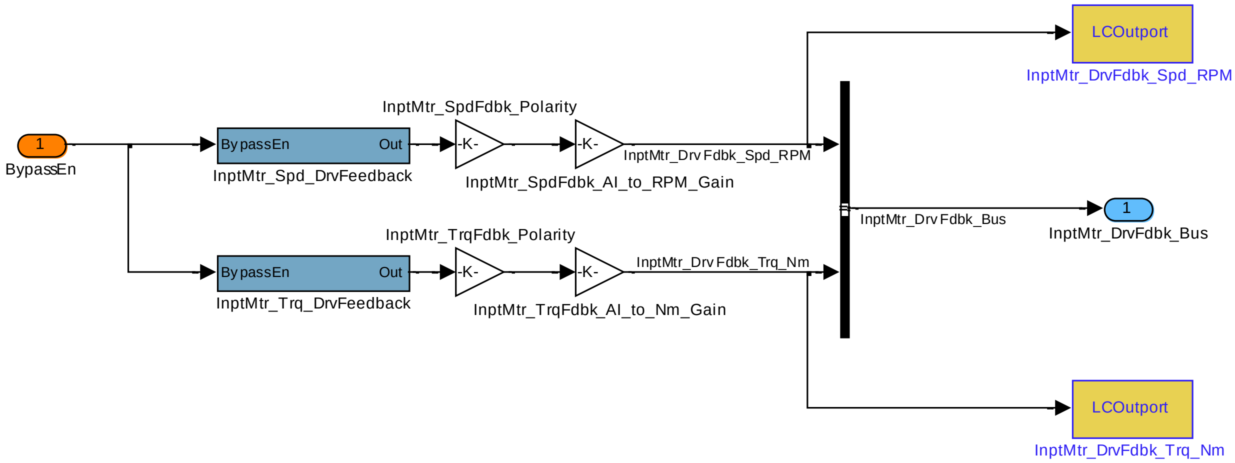

Figure 2.12: Motor Drive feedback scaling

Figure 2.12 shows the Simulink realization for the input motor speed and torque feedback signals. The same strategy is used for the output motor speed and torque feedback signals. As shown, the raw motor feedback signals originate from the blue blocks, and are adjusted for motor polarity and scaled to engineering units, using the scaling of Section 2.1.2.1 Vehicle Simulator Inputs.

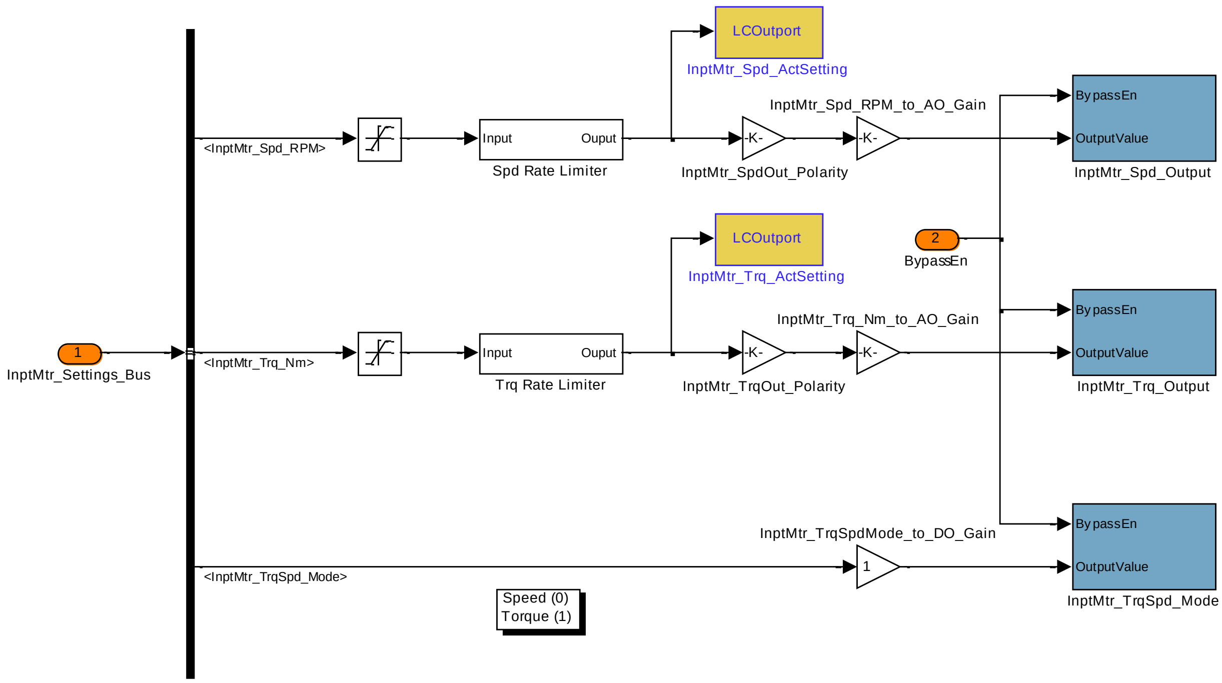

Figure 2.13: Motor command signal scaling

Figure 2.13 shows the Simulink algorithm for a similar scheme followed for the input and output Motor Drive command signals. As described in Section 2.1.2.2 Vehicle Simulator Outputs, the signals to the Motor Drives are analog, and need to be converted from the Stand Interface engineering units to an analog voltage value before sending to the APCIe-3121 card. The scaling and motor polarity adjustments are achieved using simple gain blocks.

This block also shows the output signal saturation limits and rate limiter blocks. Each Motor Drive control signal has these limits applied before the scaling occurs. These limits are in place to ensure more predictable motor reactions, and to provide better user control of the operating parameters. Like many other Simulink parameters, the limits are configured to an initial value, and then are available to the user for adjustment during runtime.

2.2.4 Stand Interface Safety Features

The Stand Interface model plays a part in making sure the test cell is controlled in a manner that is as safe as possible. To do so, there are several features of the Stand Interface that limit motor control functionality based on system conditions, or only allow certain changes to occur when the system is stable. Other safety features alert the PLC that the Vehicle Simulator is not currently running, or put the motors in a safe state if there is an issue.

2.2.4.1 Safe State Check

Certain actions should only take place when the system is in a stable and safe state. This state occurs when both input and output motors are at zero speed and zero torque. Realistically, with the noise in the motor feedback signals, the safe state is defined for a small range of speed and torque values centered on zero.

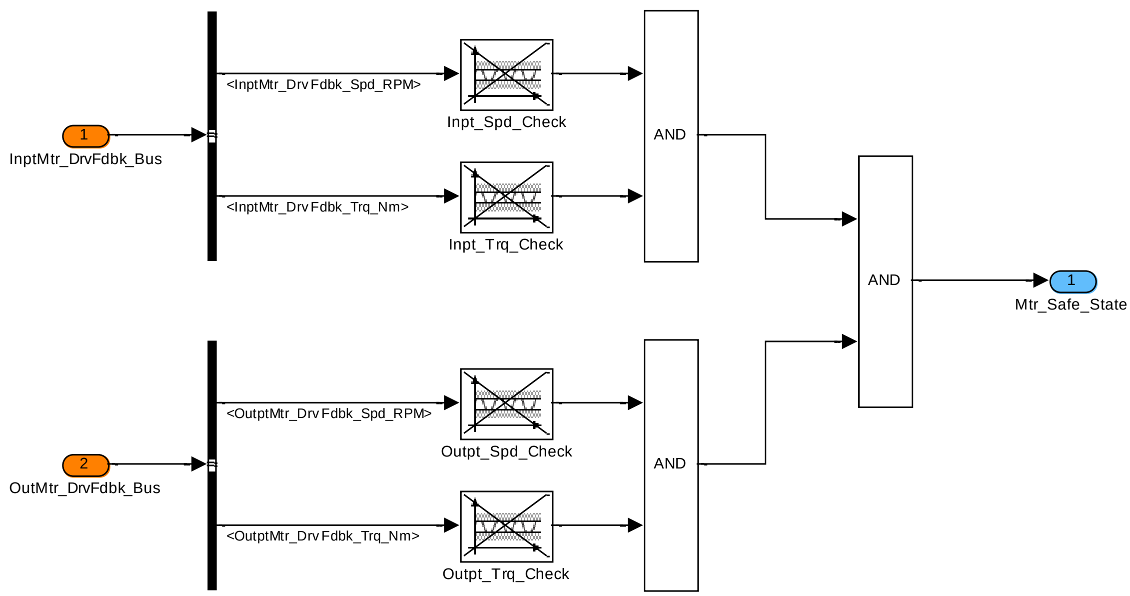

Figure 2.14: Safe state check

Figure 2.14 shows how the safe state check is implemented for the motor feedback signals. The input and output motor speed and torque signals all pass though verification blocks that send a

true value if the signal is within the desired range, and false if the value is out of range. With all the check blocks combined with a logical AND, the safe state will only occur when all feedback signals are within their respective ranges. When the system is in a safe state, the following settings can be changed in the user interface. If the system is not in a safe state, these settings cannot be changed:

- Input motor torque/speed mode;
- Output motor torque/speed mode;
- Input motor manual/plant control mode;
- Output motor manual/plant control mode; and
- User input control mode.

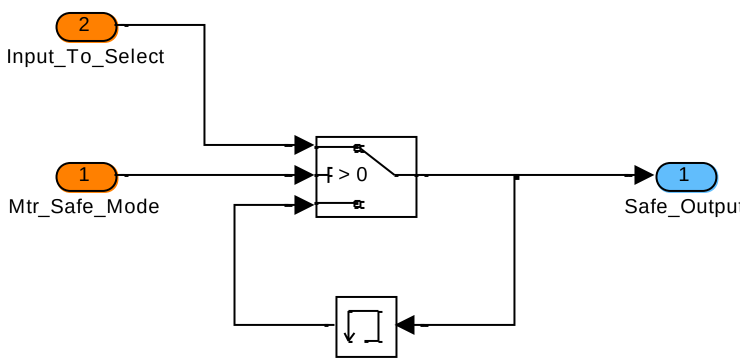

Figure 2.15: Generic safe mode setting selector

Figure 2.15 shows the generic blocks necessary for restricting the settings when not in safe mode. When safe mode is active, input 2 is passed through to the output. When not in safe mode, the signal is held to whatever its previous version has been.

2.2.4.2 Motor Enable and Stand Operational Logic

Another safety check implemented by the Stand Interface will disable the motor control commands unless several criteria are met.

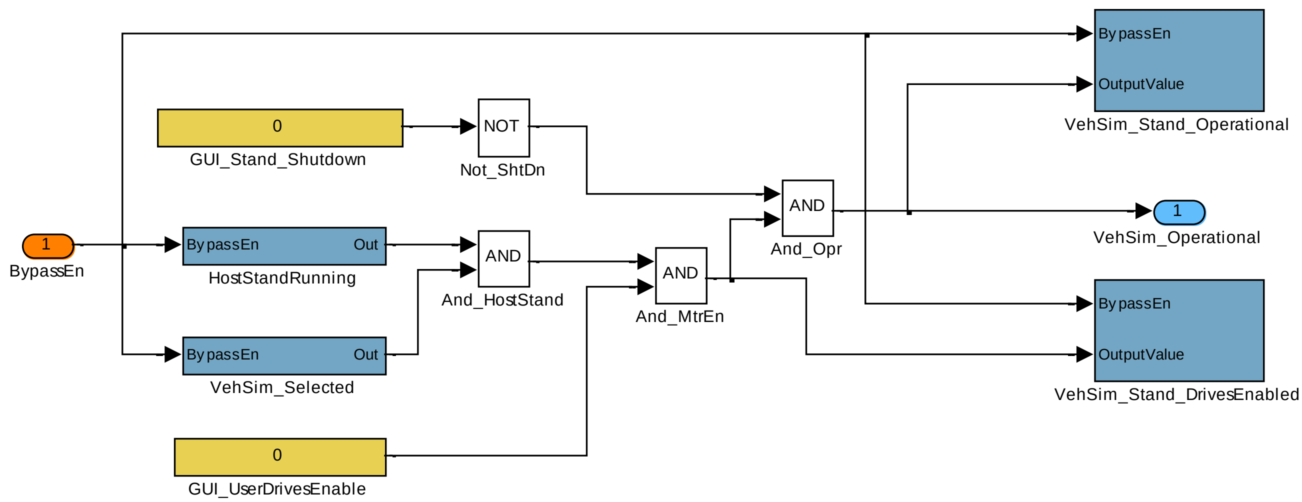

Figure 2.16: Stand operational logic

Figure 2.16 shows that the VehSim_Operational signal will only be true when the following criteria are met:

- The test cell is running;
- The Vehicle Simulator has been selected at the PLC;
- The user has enabled the motors through the Vehicle Simulator user interface; and
- The Vehicle Simulator stand shutdown parameter has not been enabled.

Both the HostStandRunning and VehSim_Selected subsystem blocks contain constant blocks, just like the GUI_Stand_Shutdown and GUI_UserDriveEnable blocks in Figure 2.16. Implementing these settings as constant blocks will make the parameters available to the user to change during runtime in the user interface.

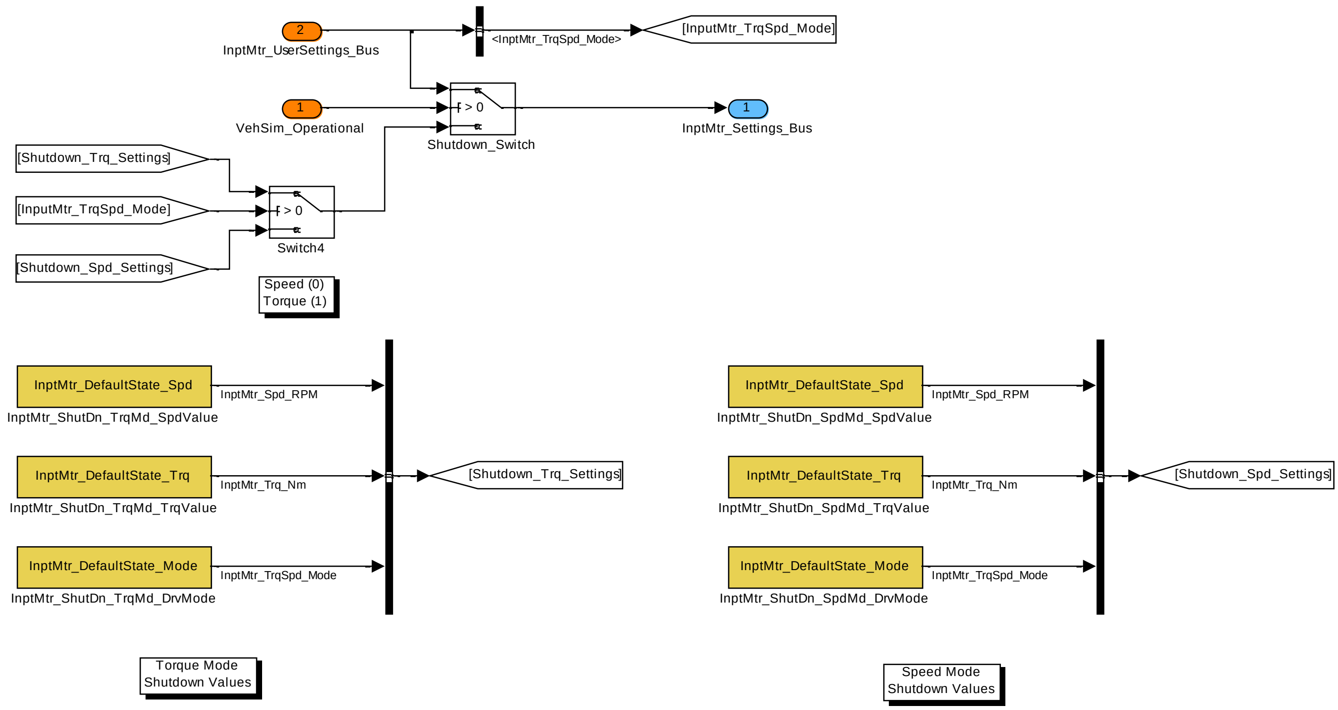

Figure 2.17: Input motor shutdown

Figure 2.17 shows how the VehSim_Operational signal from Figure 2.16 is used to disable the motor commands, in this case, for the input motor. Here, the motor commands will pass through the Shutdown_Switch block when the VehSim_Operational signal is true. Otherwise, the motor commands are set to some default values based on the current the motor mode. For both torque and speed motor modes, it happens that the disabled settings are the same. When the stand is disabled, both the input and output motors are put in torque mode, with a zero torque setting.

2.2.4.3 Heartbeat Pulse

To ensure that the Vehicle Simulator is ready to accept control of the test cell motors, and continues to run while in control, the Stand Interface model outputs a 1 Hz heartbeat pulse when actively running.

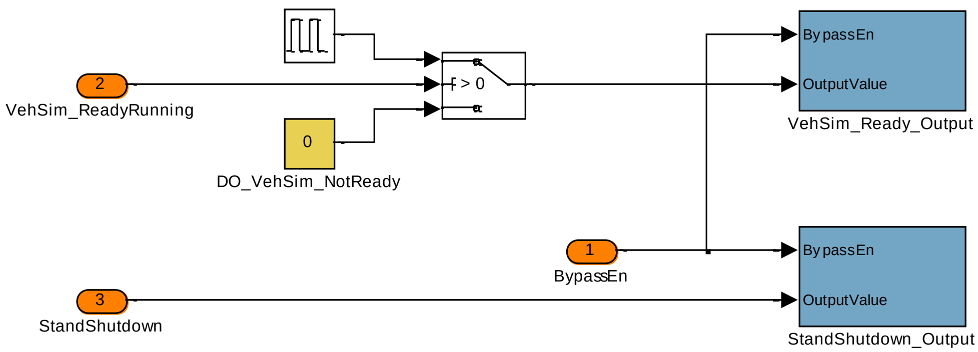

Figure 2.18: Heartbeat pulse

In Figure 2.18, the output of a pulse generator block is passed on to an output when the VehSim_ReadyRunning signal is greater than zero. This is always the case, as a constant block with the value of 1 is tied to the VehSim_ReadyRunning input one level up in the Simulink model. This pulsed output is physically connected to the input of a Red Lion 6HBWDOG1 watchdog timer, which is in turn connected to a PLC input through a relay. Effectively, whenever the pulse stops for a configured amount of time, the watchdog output switches state and signals to the PLC that the Vehicle Simulator is no longer running. This will tell the PLC that the Vehicle Simulator is ready prior to transferring control, and that the Vehicle Simulator continues to run while in control.

2.2.5 Bypass Development Mode

In order to allow for continued development when the Vehicle Simulator system is not physically connected to a test cell, or when there is no transmission present in the test cell, the Stand Interface model allows for the bypassing of its inputs and outputs to an external model. From Figure 2.6: Stand Interface model, the bypassing can be seen by way of the multiple switches, all labeled *c*. These software switches will divert the output commands from actually reading the physical output cards, as well as replacing the physical input signals from the same cards. When in bypass mode, no outputs will reach the test cell, making it safe to test out changes to the Vehicle Simulator software for errors, before actually testing in the test cell.

2.2.5.1 Bypass Mode Logic

There is a certain amount of logic involved in enabling the transition in to, and out of, the bypass development mode.

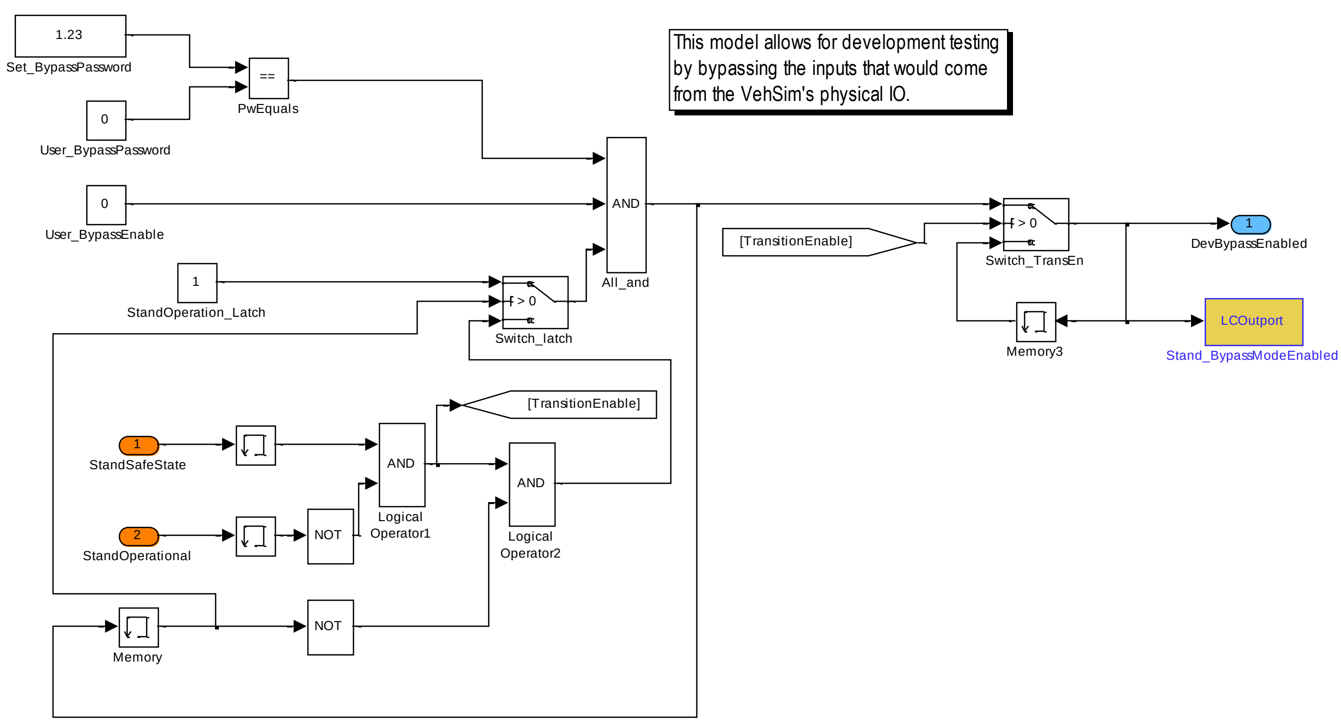

Figure 2.19: Bypass mode logic

From Figure 2.19, it can be seen several criteria have to be met to transition into bypass mode. In particular, the user has to enter a numeric password by way of changing the constant block parameter in the user interface. This password needs to match the hardcoded password of Figure 2.19. Also required is that the Vehicle Simulator is not already operational, and it is in a safe state as described in Section 2.2.4.1 Safe State Check. Once all the criteria has been met, the user also has to manually enable the bypass mode by setting the User_BypassEnable constant block parameter value to 1. Then, to transition out of the bypass mode, the Vehicle Simulator once again has to be in a safe state and not already operational. This is to avoid switching back to the actual test cell if the bypassed controls are set to something other than zero.

2.2.5.2 Input Signal Bypass Logic

When bypass mode is enabled, some of the Stand Interface inputs are switched from the analog inputs, to those from a different model.

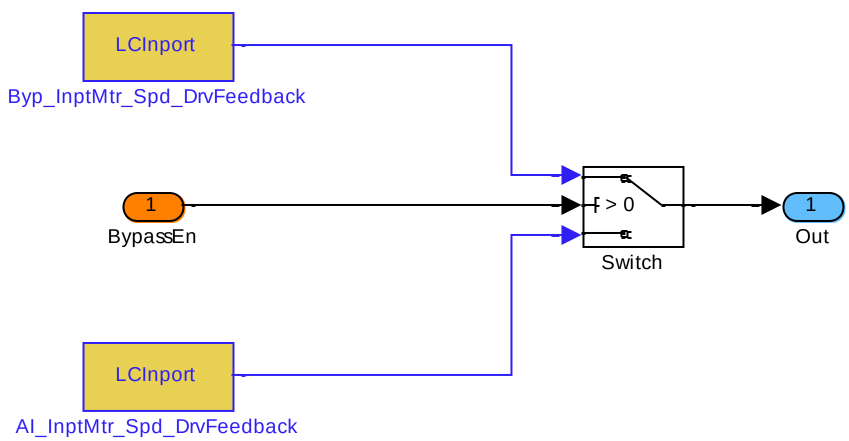

Figure 2.20: Bypass mode input

Generically, Figure 2.20 shows the template used for the inputs that are switchable in bypass mode. Both of the LCInport blocks provide connection points in the LABCAR configuration application for other software modules. For the example in Figure 2.20, the AI_InptMtr_Spd_DrvFeedback block is tied to an analog input on the APCIe-3121 card, and Byp_InptMtr_Spd_DrvFeedback is tied to the output of another Simulink model. When the

BypassEn signal is greater than zero, the bypass input is used. Otherwise, the analog input will be utilized. The same template of the Simulink logic in Figure 2.20 is copied for the following Stand Interface inputs:

- Input motor speed feedback;
- Input motor torque feedback;
- Output motor speed feedback;
- Output motor torque feedback;
- Test cell running; and
- Vehicle Simulator selected.

These inputs are all that is currently required for the Stand Interface to function outside of a test cell, as they are all tied to critical hardware inputs. All bypassed inputs need to be scaled just as the actual hardware input. For instance, the motor feedback signals will be scaled as a +/- 10 volt signal, and not the actual engineering units.

2.2.5.3 Output Signal Bypass Logic

To completely bypass the test cell, the Stand Interface outputs need to be diverted as well. In the output case, additional assurances need to be made so that the bypassed output is set to a safe value when not in use.

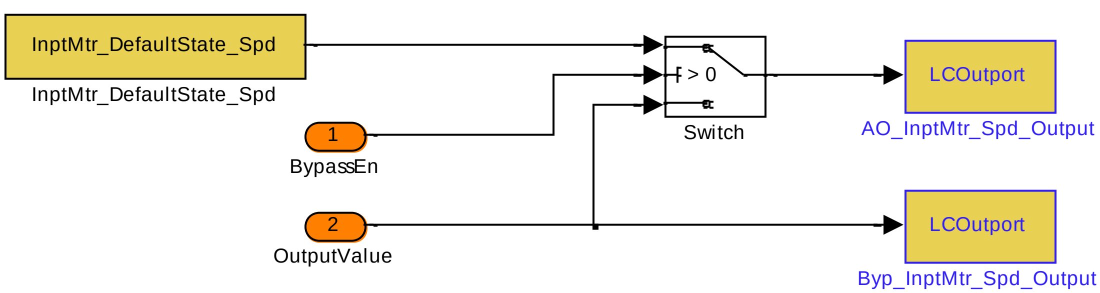

Figure 2.21: Bypass mode output

A bypass mode output example is shown in Figure 2.21, in this case for the input motor speed command signal. Much like the bypass input, there are two LABCAR blocks that allow the Stand Interface model to attach to other software modules in the LABCAR configuration application. Instead of being LCInports, here in Figure 2.21, two LCOutports provide the attachment points. The AO_InptMtr_Spd_Output signal is tied to an APCIe-3121 analog output, while the Byp_InptMtr_Spd_Output signal is tied to another model. As can be seen in Figure 2.21, the bypass output is active all the time, and only the analog output signal is switched with the bypass enable signal. When bypass enable is on, the analog output signal is set to a default and safe value, detaching it from the Stand Interface setting. The following signals all employ similar logic to what is in Figure 2.21 to achieve the bypassed output:

- Input motor speed command output;
- Input motor torque command output;
- Input motor torque/speed mode command;
- Output motor speed command output;
- Output motor torque command output;
- Output motor torque/speed mode command;
- Vehicle Simulator operational;
- Vehicle Simulator drives enabled;
- Heartbeat pulse output; and
- Vehicle Simulator test cell shutdown.

2.3 Simulated Test Cell

To further aid in offline development, the Simulated Test Cell has been developed to tie in to the Stand Interface inputs and output when in bypass mode. The Simulated Test Cell is completely realized in software, running on the same RTPC, in parallel with the Stand Interface model. Both models are allowed to run, and be interconnected, by way of the LABCAR configuration application.

2.3.1 Main Simulated Test Cell System

The Simulated Test Cell is a simplified version of the actual test cell, making the assumption that there is a single speed transmission between the input and output motors, fixed in a 1:1 gear ratio. The motor models used are also very simplistic, with no account for torque drop off at higher speeds or temperature based variations. There is also a simple overall friction model, and a test cell brake, not found in the actual test cell. As with the real test cell, the Simulated Test Cell uses the same motor polarities, so that the adjustments for engine polarity in the Stand Interface are the same. The overall system equation is

$$J\dot{\omega} = T_{InMtr} + T_{OutMtr} + T_{Friction} + T_{Brake} \tag{2.2}$$

where J is the transmission inertia, T_{InMtr} is the torque contributed from the input motor, T_{OutMtr} is the torque from the output motor, $T_{Friction}$ is the friction torque, T_{Brake} is the brake torque, and $\dot{\omega}$ is the angular acceleration. All variables are defined in SI units. The friction equation is

$$T_{Friction} = -1 \cdot ((\mu_c \cdot \tanh(\omega)) + (\mu_v \cdot \omega)) \tag{2.3}$$

where μ_c and μ_v are the Coulomb and viscous friction constants, respectively. The -1 is employed to act in the opposite direction, and the tanh function is used to smooth the typical Coulomb friction discontinuity. The brake torque equation is

$$T_{Brake} = -1 \cdot \mathrm{sign}(\omega) \cdot T_{Max} \cdot Brake_{Percent} \cdot |\tanh(\omega)| \tag{2.4}$$

where T_{Max} is the maximum amount of torque that can be applied, and $Brake_{Percent}$ is a user defined parameter that ranges from 0 to 1 as a fraction of the maximum torque applied. The tanh function reduces the brake torque for lower speeds, and the sign function assures the torque is always acting opposite of the current motion.

2.3.2 Simulated Motor and Drive Interconnection

Just like the real test cell, the Simulated Test Cell also has two motors and two drives, connected through a transmission. Although the models are simple, both drives still accept two continuous control signals for torque and speed, and a discrete signal for torque/speed mode selection. The simulated drives are able to operate in speed control, or torque control modes.

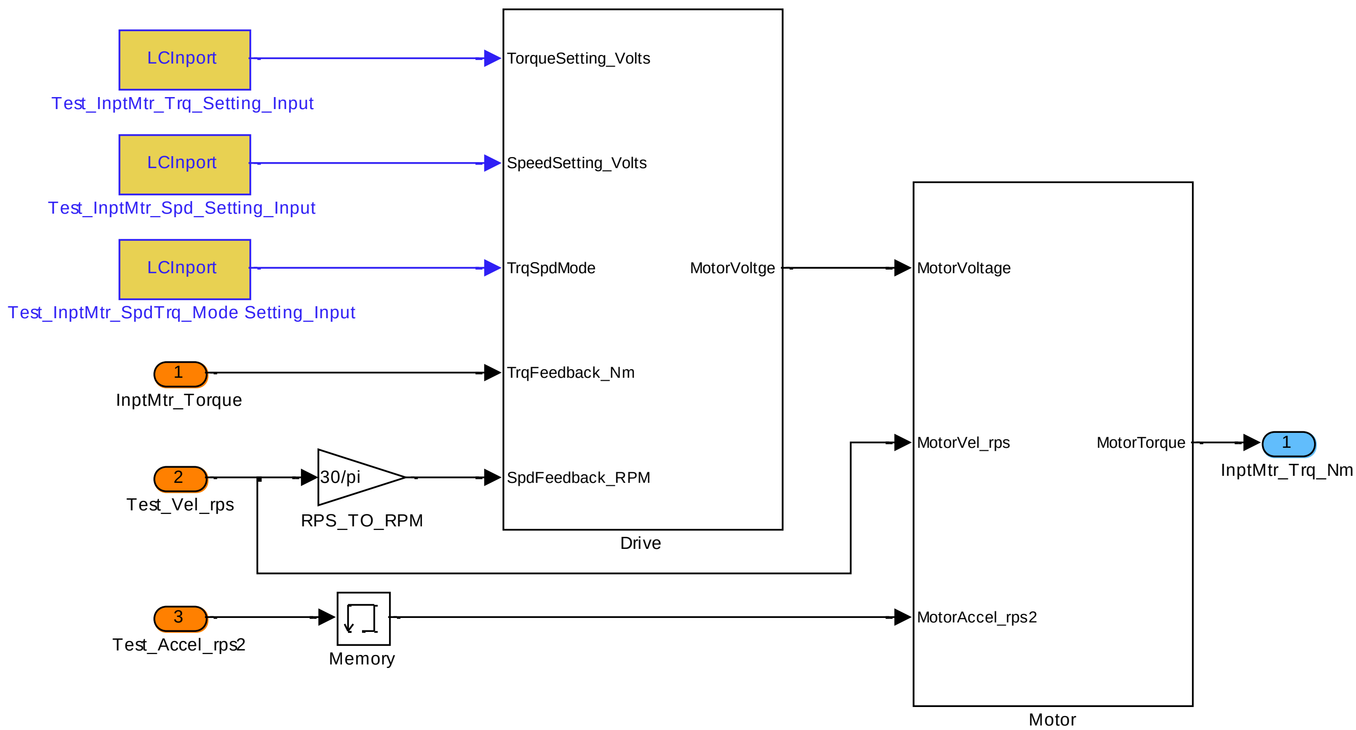

Figure 2.22: Simulated drive and motor

In Figure 2.22, the Simulink interconnection between the simulated drive and motor is shown. This same template is used for the simulated input and output motors, as well as the simulated drives.

2.3.2.1 Simulated Motor

The simulated motor accepts a voltage, angular velocity, and angular acceleration, which are used to calculate generated motor torque. The simulated motor system dynamics equations are

$$T_{Motor} = Ki - J\dot{\omega} - b\omega \tag{2.5}$$

$$L\frac{di}{dt} = V - Ri - K\omega \tag{2.6}$$

and based on the motor equations in reference [17]. In the proceeding equations, T_{Motor} is the torque produced by the motor, K is both the motor electromotive force constant and the torque constant, J is the motor inertia, b is the viscous friction constant, ω is the motor angular velocity, L is the motor inductance, V is the applied motor coil voltage, R is the motor coil resistance, and i is the motor current with all variables defined in SI units.

2.3.2.2 Simulated Motor Drive

To control the simulated motor, a simulated motor drive was developed as well. To imitate the test cell drives, separate speed and torque modes are required. To accomplish the separate control modes, two proportional integral (PI) controllers were utilized.

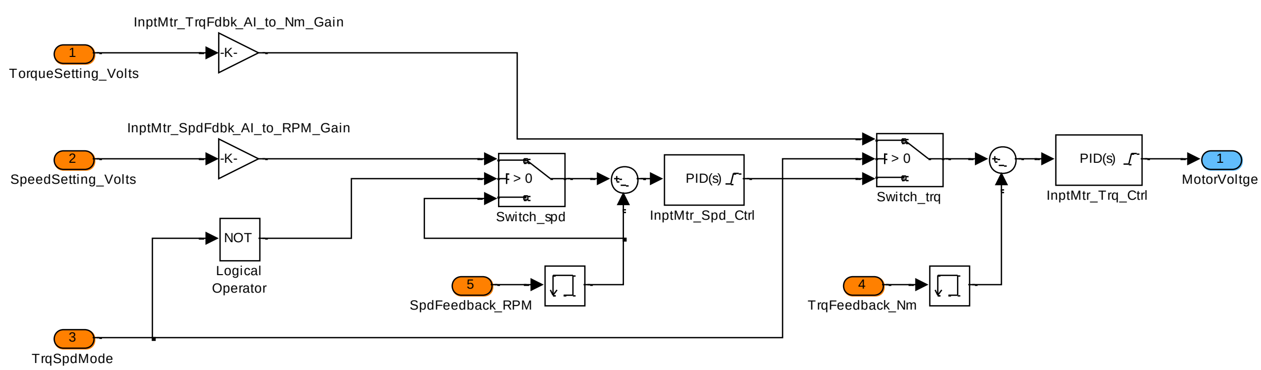

Figure 2.23: Simulated motor drive

Figure 2.23 shows the Simulink model of the simulated motor drive. In torque control mode, only the torque PI controller is in the feedback loop, with the torque set point coming directly from the user input. When speed mode is active, the torque set point switches from the user set value to the speed controller set value, by way of a second speed specific PI controller. There is an additional switch that effectively zeros the speed PI controller error to zero when speed mode is not selected in an effort to reduce integrator wind up.

2.4 Vehicle Plants Models

To aid in controls development work for their automated product, Eaton engineers have developed several different full vehicle models of varying fidelity. These vehicle models allow the engineer to completely test out new transmission controller functionality in a virtual environment before implementing in the real world. Typically, this virtual environment is provided by the engineer's laptop or desktop computer. Higher fidelity vehicle models allow testing of complicated systems in greater than real-time, while lower fidelity models provide quicker execution times, permitting for faster-than-real-time execution. With full virtual development, otherwise known as Model in the Loop (MIL) development, all components of the vehicle are required, namely the engine plant and controller, clutch and transmission plants, transmission controller, and vehicle plant. Figure 2.24 shows how different components of the full vehicle plant can be reused for the different simulation platforms.

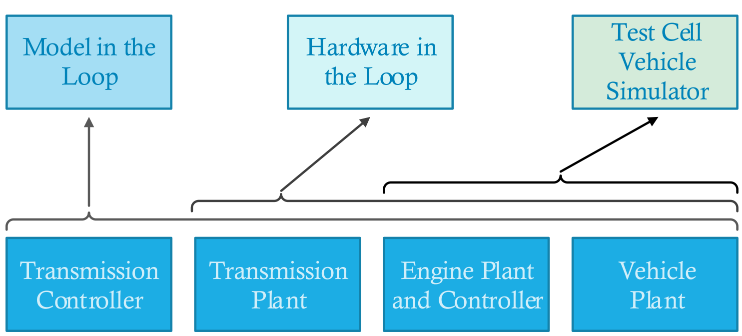

Figure 2.24: Vehicle model utilization strategy

The Hardware in the Loop (HIL) stands utilize a lower fidelity fixed step version of the same vehicle models used in controls development, with the transmission controller removed,

all developed in MATLAB/Simulink. Reusing the models helps to reduce development time for the HIL stands, and increases consistency between the two testing environments. Since the transmission controller interacts with the vehicle plant models via hardware, instead of completely in software, additional software modules are required to interface the hardware with the remaining vehicle plant models.

For the very same reasons that the HIL stand reuses the control development models, the Vehicle Simulator will also make use of a subset of the same lower fidelity Simulink fixed step vehicle models. Since more of the system will be real and connected to the test cell, the Vehicle Simulator will require a reduced subset of the vehicle plant models. No longer is the transmission plant model needed, only the engine plant, engine controller, and vehicle plant.

2.4.1 Engine Plant Model

Typically, the input side of a transmission is connected to a diesel engine, capable of producing large amounts of torque at the expense of consuming fuel. In the test cell case, the input shaft of the transmission is connected to a low inertia motor, capable of producing a percentage of the actual torque an actual engine could produce. The Engine Plant is responsible for making the motor act like a real engine in the test cell. Figure 2.25 show a high level view of the various components of the Engine Plant, developed by other Eaton engineers.

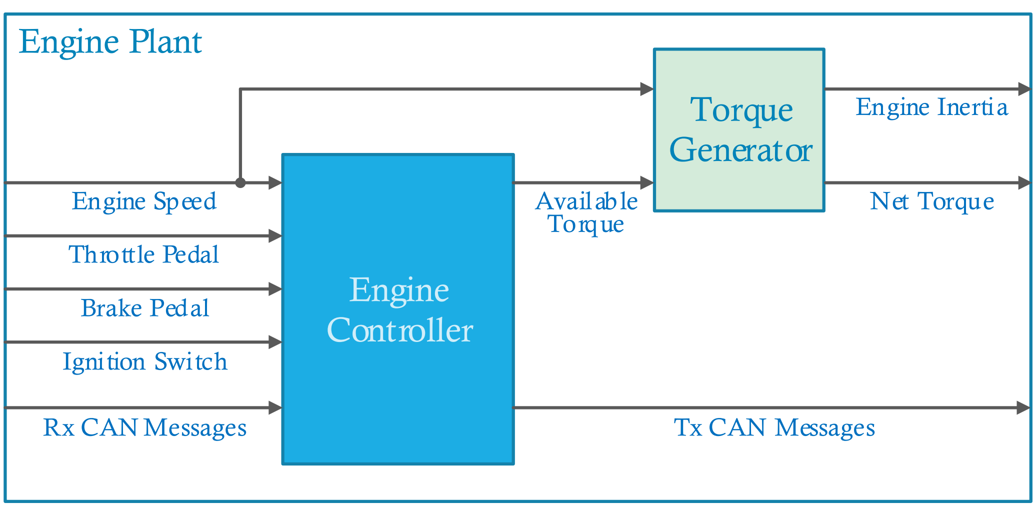

Figure 2.25: Engine Plant overview

2.4.1.1 Engine Controller

The engine controller accepts operator controls, CAN messages, and the engine speed to produce the total available torque from the engine. The controller accomplishes this task by way of a 2D lookup table, taking engine speed and throttle pedal position, and interpolating the total available torque. The lookup table is implemented as a MATLAB workspace parameter that can be adjusted to better mimic the characteristics of different engines. The controller also builds the expected engine CAN messages, making them available to send on the CAN bus. Without these expected engine messages, other controllers, like the transmission controller, would assume there is no engine present, and go into a fault state. An idle governor is present to make sure the engine speed stays at a minimum while the ignition is on.

The engine controller also handles the speed and torque control from several different sources. The transmission controller sends a TSC1 J1939 CAN message, which requests the engine to operate at a desired speed or torque, depending on the message settings. This allows the transmission controller to request zero torque prior to shifting gears, much the same way a driver of a manual transmission would take their foot off the throttle prior to shifting. Similar to the TSC1 message, the engine controller also implements the cruise control system, allowing an operator to request and maintain a consistent speed. The cruise control system will also allow the user to increment or decrement the speed setting.

2.4.1.2 Torque Generator

The torque generator takes the total available torque, boosts it via the turbo, and subtracts off engine loads to produce the net torque available to the clutch. One possible load is the friction torque, which changes based on the state of the simulated engine fan. The friction torque is calculated using two different one dimensional lookup tables based off of engine speed, depending on whether the fan is on or not. Another engine load is the user operated engine brake, which is also calculated by way of an engine speed based lookup table. When the user turns on the engine brake, the fuel is cut to the engine so no torque is produced, and an additional speed based load is applied. The torque generator also contains the starter motor, which generates torque when the user turns the ignition.

2.4.2 Chassis Plant Model

The Chassis Plant is responsible for calculating the loads that are applied to the output shaft of the transmission. In a real truck these loads would be applied to the transmission by way of the drive shaft, which is connected to the truck chassis through the differential. In the test cell, a low inertia motor takes the place of the drive shaft, and applies a load torque within the capability of the motor. The original intent was to reuse the same Chassis Plant model as the MIL and HIL development, but subjecting that model to real world noise proved unstable. Instead, a simpler model was developed by Eaton engineers, which improves the stability of the system while still using the same vehicle load calculations as the MIL and HIL Chassis Plant model. In addition to calculating the vehicle load, the vehicle plant also generates the EBC2 J1939 CAN message to send out on the bus.

2.4.2.1 Chassis Plant Model System

The Chassis Plant model operates with the output motor in speed mode so the drive will cause the motor to act as a large inertia. In the simplest terms, the Chassis Plant model takes the torque generated from the Engine Plant, multiplies it by the transmission gear ratio as reported from the transmission controller, and subtracts off the vehicle load torques to determine the torque available to accelerate the vehicle mass. The engine torque is only applied to the vehicle mass if the Driveline Engaged J1939 signal from the transmission controller says the clutch is closed. The torque reported from the Engine Plant is often much greater than what the input motor can provide. With this, the Chassis Plant is allowed to accelerate as if the full torque from a truck was being applied to the vehicle mass, as opposed to the lessor torque from the motor. From the torque applied to the vehicle mass, the vehicle speed is calculated, which is then commanded to the output motor.

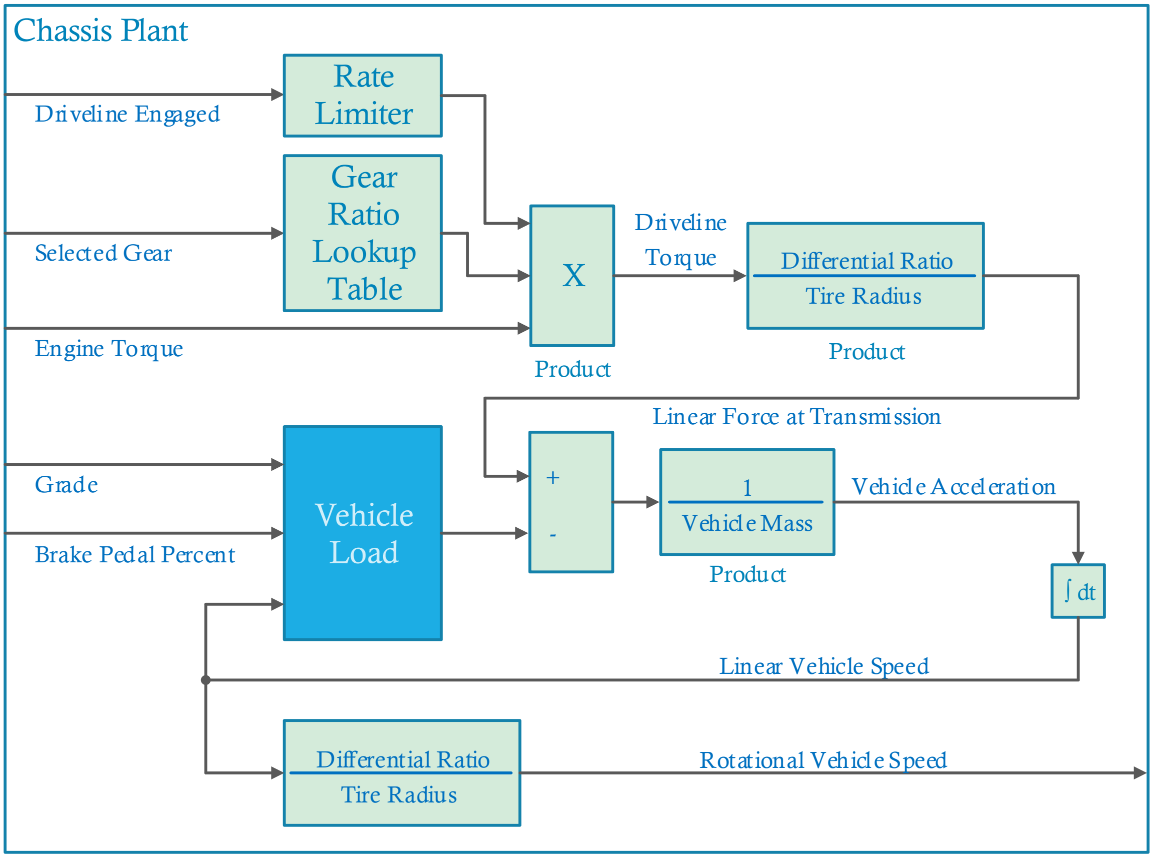

Figure 2.26: Chassis Plant system

Figure 2.26 shows an overview of the Chassis Plant model system, displaying the required inputs and the only rotational vehicle speed output. After the total force on the vehicle mass is known, determining the speed is a simple integration of Newton's Second Law of motion. Once the torque from the Engine Plant model is multiplied by the gear ratio, it is as if that torque is available to the output shaft of the transmission. In trucks, there is commonly a differential that translates the driveline torque to the wheels, which includes another gear ratio. To translate total transmission torque to linear force, it must be multiplied by the differential gear ratio, then divided by the wheel radius in meters. Then, to translate linear velocity to rotational velocity, the same ratio of differential gear to tire radius is employed again.

2.4.2.2 Vehicle Load Equations

To determine how much force is available to linearly accelerate the vehicle, the amount of load force has to be calculated, and subtracted from the torque provided from the engine through the

transmission. The vehicle load is a combination of forces acting linearly on the truck, namely the drag force, rolling resistance, and the force of gravity acting on an incline. Also included in the vehicle load is the brake force applied from the operator. The applicable loads acting on the vehicle are

$$F_L = F_R + F_G + F_{Br} + F_{Air} \tag{2.7}$$

where F_L is the vehicle load, F_R is the rolling resistance force, F_G is the force of gravity felt on an incline, F_{Br} is the brake force, and F_{Air} is the force of air resistance.

The rolling resistance force F_R is a combination of coulomb and viscous forces, defined as

$$F_R = (v_v \cdot R_v + \tanh(v_v)\, R_c)F_N \tag{2.8}$$

where v_v is the linear vehicle velocity in meters per second (m/s), R_c is the coulomb rolling resistance constant, R_v is the viscous rolling resistance constant, and F_N is the normal force in newtons (N). The tanh function smoots the discontinuity typically caused by the coulomb friction.

The incline force F_G is calculated as a function of the road grade percentage G_p and the normal force F_N,

$$F_G = 0.01 \cdot G_p \cdot F_N \tag{2.9}$$

where road grade percentage is a value typically reported by the transmission controller. The road grade percentage, a unit of measure for the amount of height a road will rise over a specified distance, also known as run, is

$$G_p = 100\left(\frac{rise}{run}\right) = 100 \cdot \tan\theta. \tag{2.10}$$

A simple inclined plane demonstrates how the grade angle and forces are related.

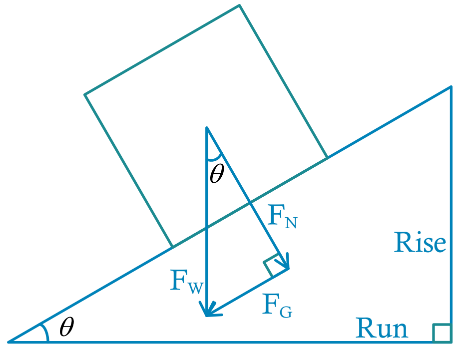

Figure 2.27: Simple inclined plane

The vehicle weight force F_W, is

$$F_W = m \cdot g \tag{2.11}$$

where m is the vehicle mass in kilograms, and g is the acceleration due to gravity, or 9.81 $\frac{\mathrm{m}}{\mathrm{s}^2}$. The truck normal force F_N is

$$F_N = F_W \cdot \cos\theta. \tag{2.12}$$

Although the $\cos\theta$ is easily calculated, it is more convenient to define the normal force as a function of the grade percentage as

$$F_N = \frac{F_W}{\sqrt{\left(0.01 \cdot G_p\right)^2 + 1.}} \tag{2.13}$$

This definition is based on the trigonometric identity [18]

$$\cos\theta = \frac{1}{\sqrt{\tan^2\theta + 1.}} \tag{2.14}$$

The air resistance force F_{Air} is calculated using

$$F_{Air} = C_r \cdot v_v \cdot |v_v| \tag{2.15}$$

where v_v is the vehicle velocity in meters per second and C_r is a proportional aerodynamic constant, determined empirically. The above equation is based around the drag force

$$F_{Drag} = \frac{1}{2} \cdot \rho \cdot C \cdot A \cdot v_v^2 \tag{2.16}$$

where ρ is the fluid density, C is the coefficient of drag, A is the cross sectional area, and v_v is the vehicle velocity [19]. The absolute value is used in equation (2.15 to maintain a directional sign if the vehicle is moving in reverse.

2.4.3 Input and Output Adapters

Since the engine and vehicle plant models were originally developed with a purely simulated environment in mind, a certain amount of software infrastructure is needed to use the plant models in the Vehicle Simulator. The input and output adapters provide the necessary software components for the Engine and Chassis Plant models to successfully interact with the Stand Interface model. Figure 2.1 shows a high level view of how the Engine and Chassis Plant models are connected to the input and output adapters, and then with the Stand Interface model. The input adapter connects signals from the Stand Interface model to those that are required for the Engine and Chassis Plant models to operate. The output adapter takes the signals generated from the Engine and Chassis Plant models, and routes them back to the Stand Interface model as a separate motor control mode. Since there is separation between the models, it would be easy to update the plant models, or change them to other plant models if necessary.

3 IMPLEMENTATION VERIFICATION

The Vehicle Simulator system was evaluated for several different control modes, with various user inputs, via three separate tests. The Simulated Test Cell functionality was also evaluated with several of the same tests to demonstrate how closely it matches the actual test cell. The tests have been selected to show that the Vehicle Simulator can operate in manual motor control mode, as well as in full vehicle emulation mode. The first two tests show how the Stand Interface model provides manual motor control through various user input sources, while the transmission is held in a 1:1 gear ratio. The last test shows dynamic shifting of the transmission through several gears, while utilizing the Engine and Chassis Plant models. For each test, the setup will be discussed, followed by a presentation and analysis of the results. Prior to presenting the verifications test results, an overview of the data collection method will be provided.

3.1 Data Collection and Analysis Software

Throughout all verification tests, several different applications were utilized to collect, analyze, and display test data. In addition to receiving CAN control messages, the Vehicle Simulator is capable of sending several motor drive feedback messages to data acquisition software. Values like motor speed, motor torque, and stand indicators are reported to the data acquisition software through the CAN feedback messages. Vector CANape was used with a CANcaseXL box to collect the drive monitor messages, in addition to the standard J1939 messages from the Engine Plant model, Chassis Plant model, and the transmission controller. Once collected, CANape stores the data as MDF files. CANape was also used to send the CAN user input commands that controlled some of the tests as well. Once the data was collected, the imc FAMOS application was used to analyze and display the data. FAMOS has the advantage of opening many different file formats, while providing powerful graphing utilities that allows the data to be examined in several different ways. The majority of the remaining data graphs will be from FAMOS.

3.2 Test 1: Input Motor Speed Control – Output Motor Torque Steps

The first verification test demonstrates that the input motor can be controlled in speed mode, and the output motor can utilize the torque control mode. The torque and speed commands can be provided by the analog pedals, the user interface, or CAN commands. Only results from scripted CAN commands are shown. The analog pedals cannot provide the accurate timing of a script running from CANape. Running the test through the user interface controls would yield the same timing inaccuracies as the analog pedals if run manually. Controlling the user interface via a Python script would likely produce closer timing to the CANape script, but was not analyzed for this test.

3.2.1 Test 1: Test Procedure

To properly show input and output motor speed and torque control modes, it was determined that the output motor should apply several different toque steps, while the input motor attempts to maintain a constant speed with the test transmission in a 1:1 gear ratio. The direct gear ratio simplifies the test, in that the command torque at the output motor should roughly equal the torque supplied by the input motor. Also, the two motor speeds should closely match. The following test procedure was realized as a CANape script, and used to control the Vehicle Simulator.

- Set transmission in 1:1 gear ratio with clutch closed
- Do torque steps of 30% and 70% of the lessor rated motor torque
    - Input Motor: 172 N·m nominal torque
        - 30% ≈ 50 N·m
        - 70% ≈ 120 N·m
- Input Motor Setup
    - Speed Control Mode
    - Manual Motor Control
    - Rate Limit = 1000 RPM/s
    - Speed Limit = 1000 RPM
    - Torque Limit = 258 N·m

- Output Motor Setup
    - Torque Control Mode
    - Manual Motor Control
    - Rate Limit = 1000 N·m/s
    - Speed Limit = 1000 RPM
    - Torque Limit = 150 N·m
- Test Sequence
    - Start the test
    - Wait 1 second
    - Bring the input motor up to 700 RPM
    - Wait 5 seconds
    - Command -50 N·m of torque on the output motor
    - Wait 5 seconds
    - Command -120 N·m of torque on the output motor
    - Wait 5 seconds
    - Command 0 N·m on the output motor
    - Wait 5 seconds
    - Bring the input motor to 0 RPM
    - End test

3.2.2 Test 1: Results and Analysis

As described in the previously stated test procedure, the Vehicle Simulator was capable of controlling the input motor in speed control and the output motor in torque control.

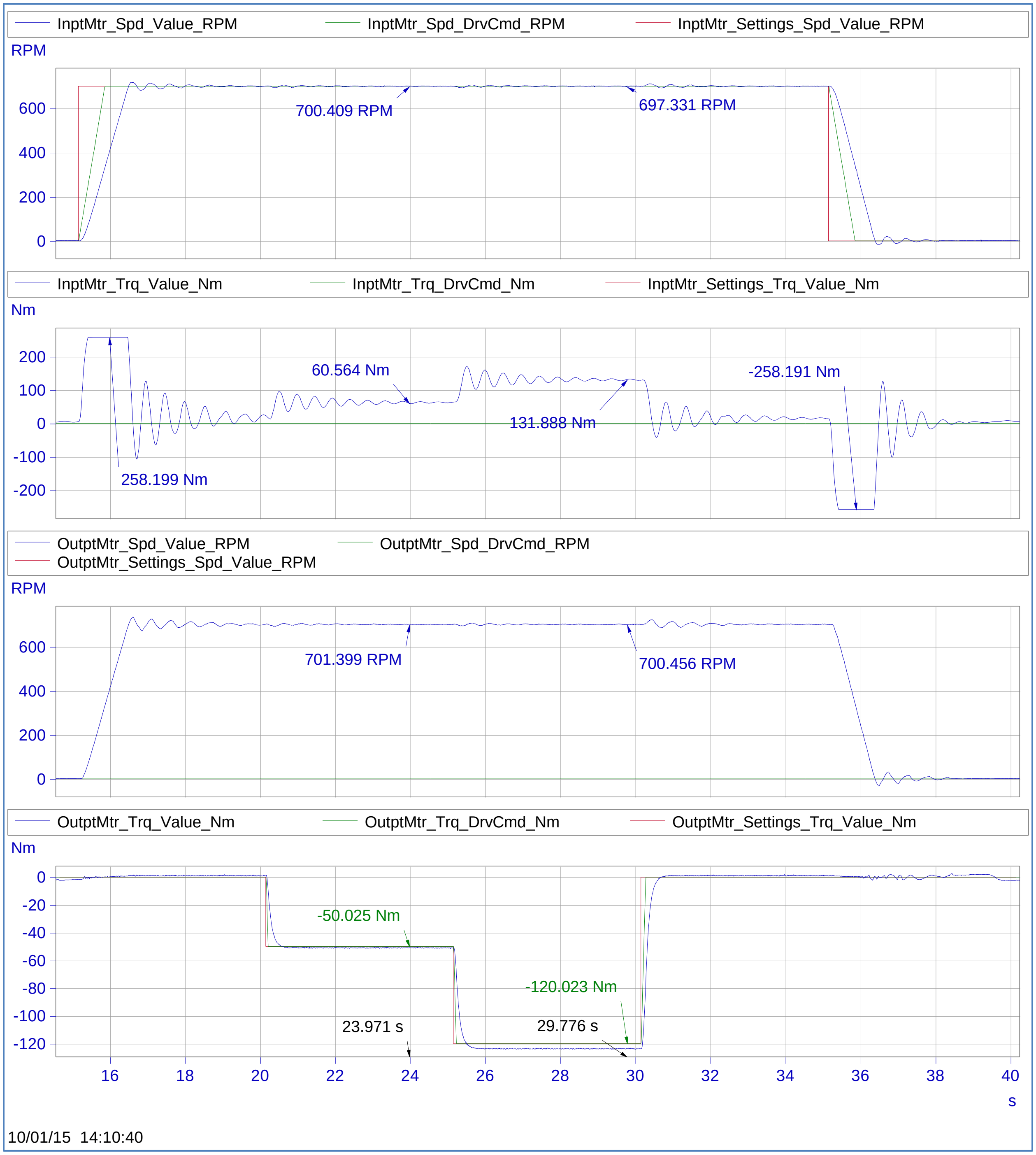

Figure 3.1: Test 1 - Overall test

Figure 3.1 shows the results of Test 1, with the input motor successfully being controlled to roughly 700 RPM all while the output motor progressed through a series of torque steps. As can be seen, the scripted timing between speed commands and torque steps was accurately realized. Also, the input motor speed control loop is underdamped as the system comes up to speed and throughout each torque step. This is in contrast to the critically damped current loop of the output motor, which does not show any oscillations.

Graph Signal Name	Description
InptMtr_Spd_Value_RPM	Input motor speed in RPM, as reported by the input motor drive
InptMtr_Spd_DrvCmd_RPM	The input motor speed in RPM commanded to the input drive from the Vehicle Simulator
InptMtr_Settings_Spd_Value_RPM	The input motor speed command in RPM as commanded to the Vehicle Simulator from CANape
InptMtr_Trq_Value_Nm	Input motor torque in N·m as reported by the input motor drive
InptMtr_Trq_DrvCmd_Nm	The input motor drive torque command in N·m as commanded to the drive from the Vehicle Simulator
InptMtr_Settings_Trq_Value_Nm	The input motor speed command in RPM as commanded to the Vehicle Simulator from CANape
OutptMtr_Spd_Value_RPM	Output motor speed in RPM, as reported by the output motor drive
OutptMtr_Spd_DrvCmd_RPM	The output motor speed in RPM commanded to the outut drive from the Vehicle Simulator
OutptMtr_Settings_Spd_Value_RPM	The output motor speed command in RPM as commanded to the Vehicle Simulator from CANape
OutptMtr_Trq_Value_Nm	The output motor torque in N·m as reported by the output motor drive
OutptMtr_Trq_DrvCmd_Nm	The output motor drive torque command in N·m as commanded to the drive from the Vehicle Simulator
OutptMtr_Settings_Trq_Value_Nm	The input motor speed command in RPM as commanded to the Vehicle Simulator from CANape

Table 3.1: Test 1 - Data plot channel definitions

Table 3.1 provides a link between the signal names of the plot in Figure 3.1 and their descriptions. The same signal names will be used throughout the remainder of this section.

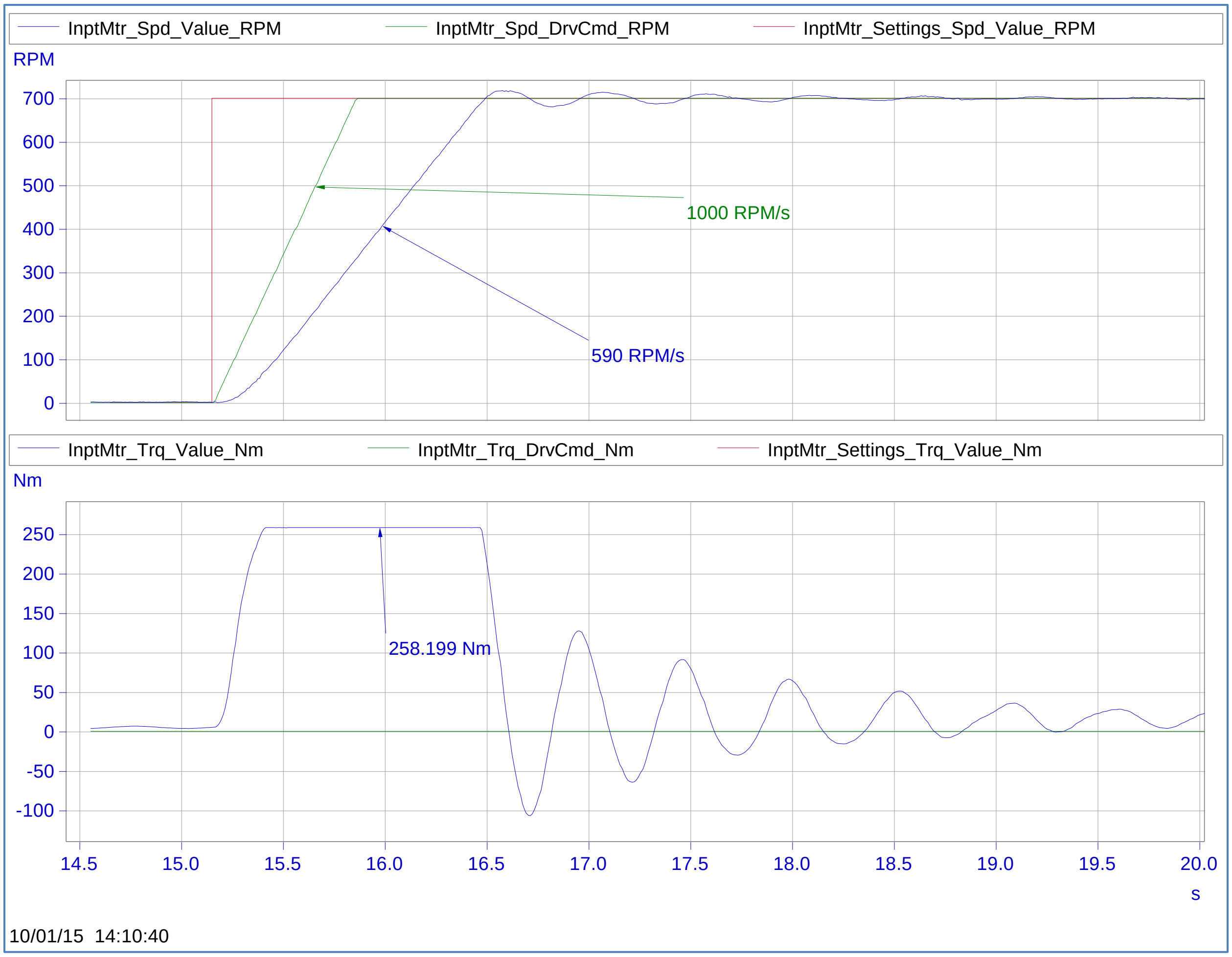

Figure 3.2: Test 1 - Speed ramp detail

Figure 3.2 shows a zoomed in view of Test 1, presenting a detailed view of several system characteristics. First, the 1000 RPM/s speed rate limit on the input motor command can be seen, which is the analog command from the Vehicle Simulator to the intput motor drive. The speed rate limit is part of the Stand Interface model and is user configuratble. Also noticable in Figure 3.2 is the torque limiting of the input drive, as it attempts to bring the system up to a speed of 700 RPM. Due to systems characteristics, the input motor spends a little more than a second at full torque, causing the actual acceleration of the input motor to be around 590 RPM/s. Also in this more detailed view, the damped torque oscillations can be seen at initial excitation, to move between -100 N·m and 100 N·m, before settling.

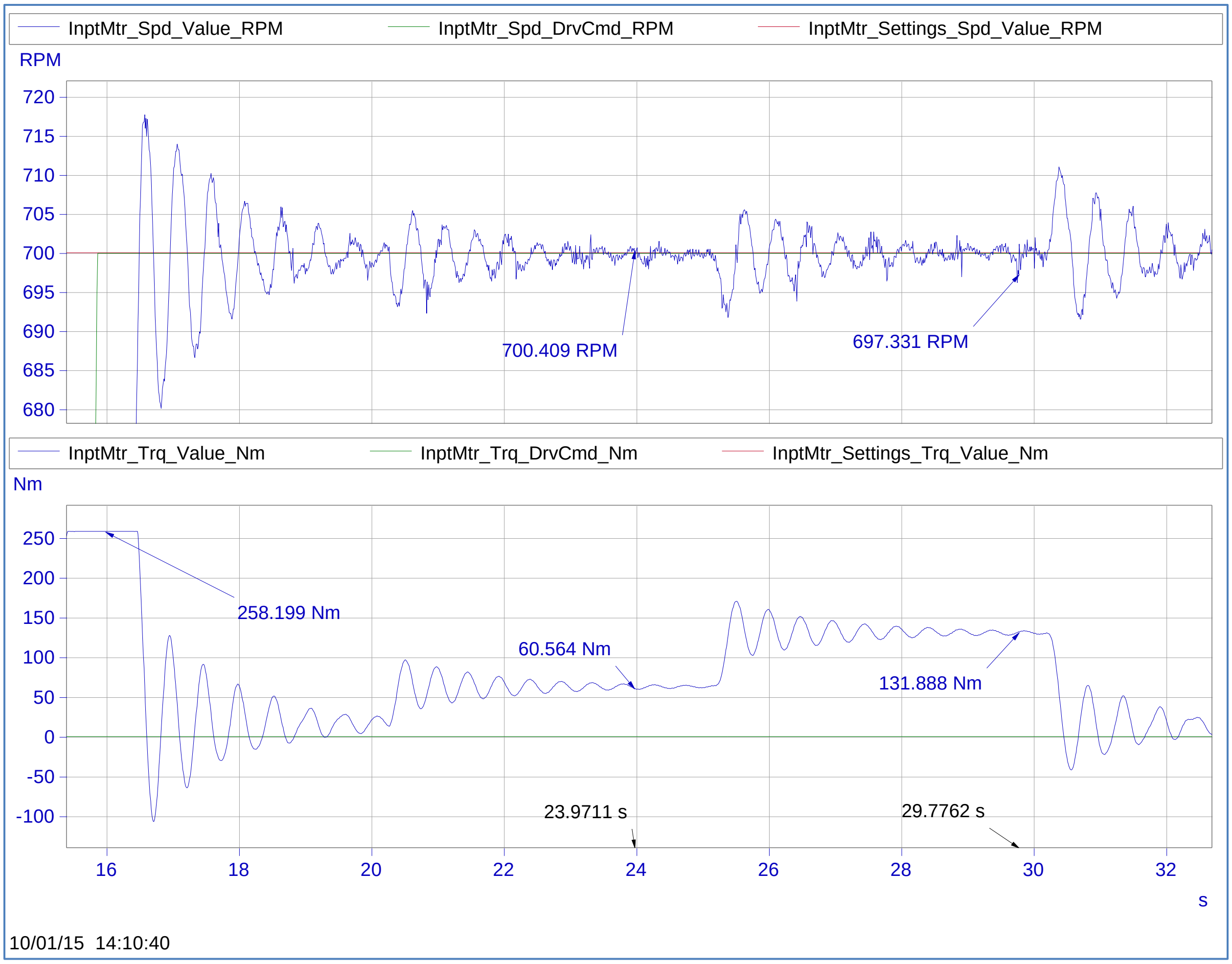

Figure 3.3: Test 1 - Speed signal detail

The signals in Figure 3.3 show a different detailed view of the input motor speed and torque values, as reported by the input motor drive. Here, the y-axis scale has been increased to show the input motor speed oscillations when the different torque steps are applied. Although the torque is oscillating by 200 N·m peak-to-peak as the speed initially comes up to the commanded value, the input motor speed only oscillates to about 35 RPM peak-to-peak. Through the two torque steps from the output motor, the input motor speed oscillations hold to around 10 RPM, and only increase slightly when the finial torque step is removed. Based on the speed oscillations, the input motor drive should be tuned to better match the characteristics of the system under test.

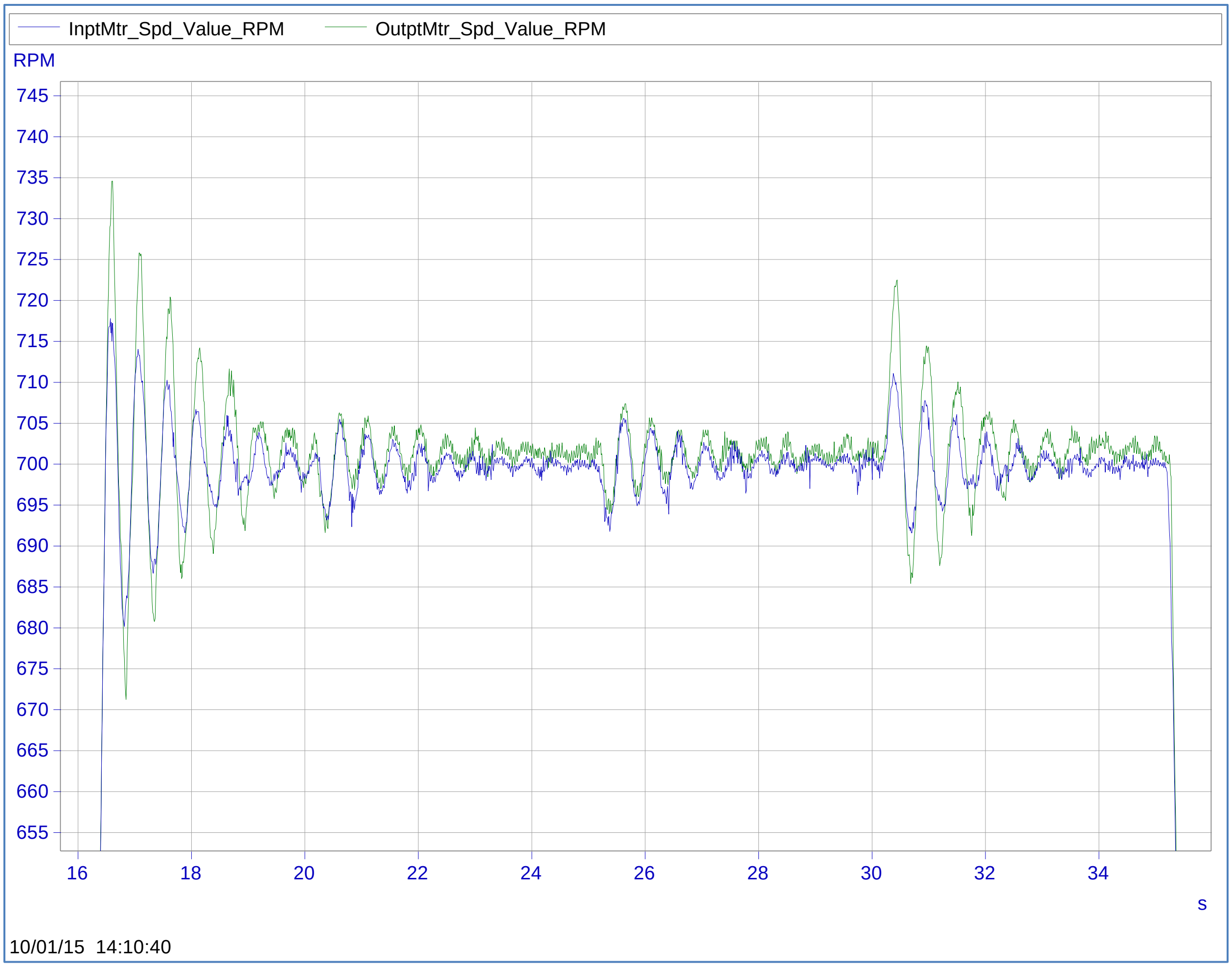

Figure 3.4: Test 1 - Input and output motor speed detail

Figure 3.4 shows details of the input motor speed and output motor speed plotted on the same y-axis, showing the difference in speeds throughout the test. Comparing this plot to Figure 3.3, the speeds are the furthest apart when the torque oscillations are the greatest. During the two torque steps, the input motor and output motor speeds match closely.

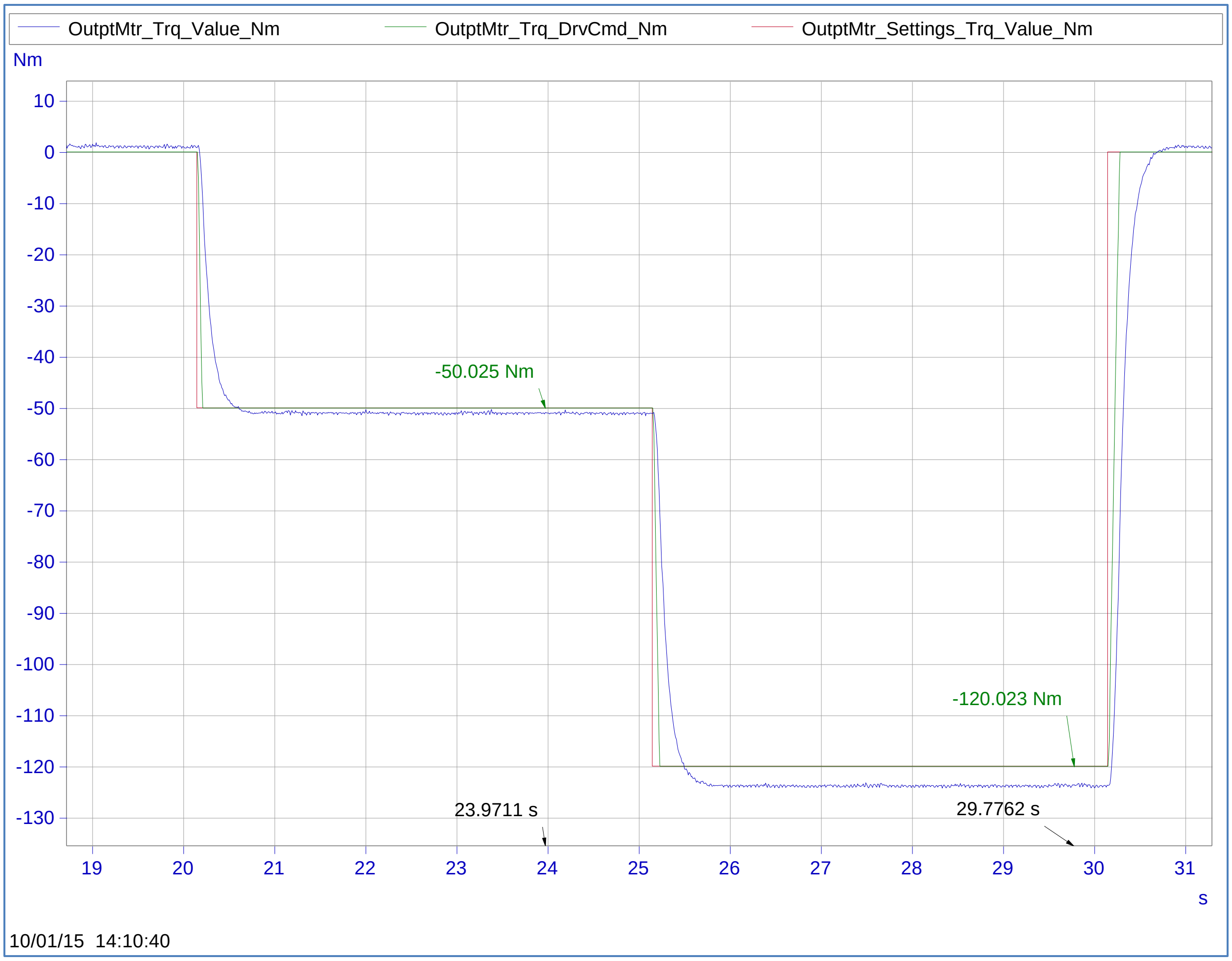

Figure 3.5: Test 1 - Output motor torque detail

In Figure 3.5 shows details of the output motor torque values as reported by the output drive, throughout the test. The torque loop of the output motor drive closely follows the torque commands of the test. In this critically damped system, there are no oscillations like that of the input motor drive speed control. As defined in the test procedure, the output motor is correctly commanded at -50 N·m for 5 seconds, then -120 N·m for another 5 seconds, before returning to 0 N·m at the end of the test. Here, the motor sign convention still holds, where a negative value indicates torque acting against the normal engine motion. A negative torque acts as a load of a vehicle on the engine. Also shown here is a small offset between commanded torque value and the reported torque value. As great care was used to insure proper scaling, the offset is likely from the 0.1s time constant filter on the reported toque values.

3.2.3 Test 1: Comparison with the Simulated Test Cell

The same test procedure was run with the Simulated Test Cell enabled instead of the actual test cell, and data was recorded.

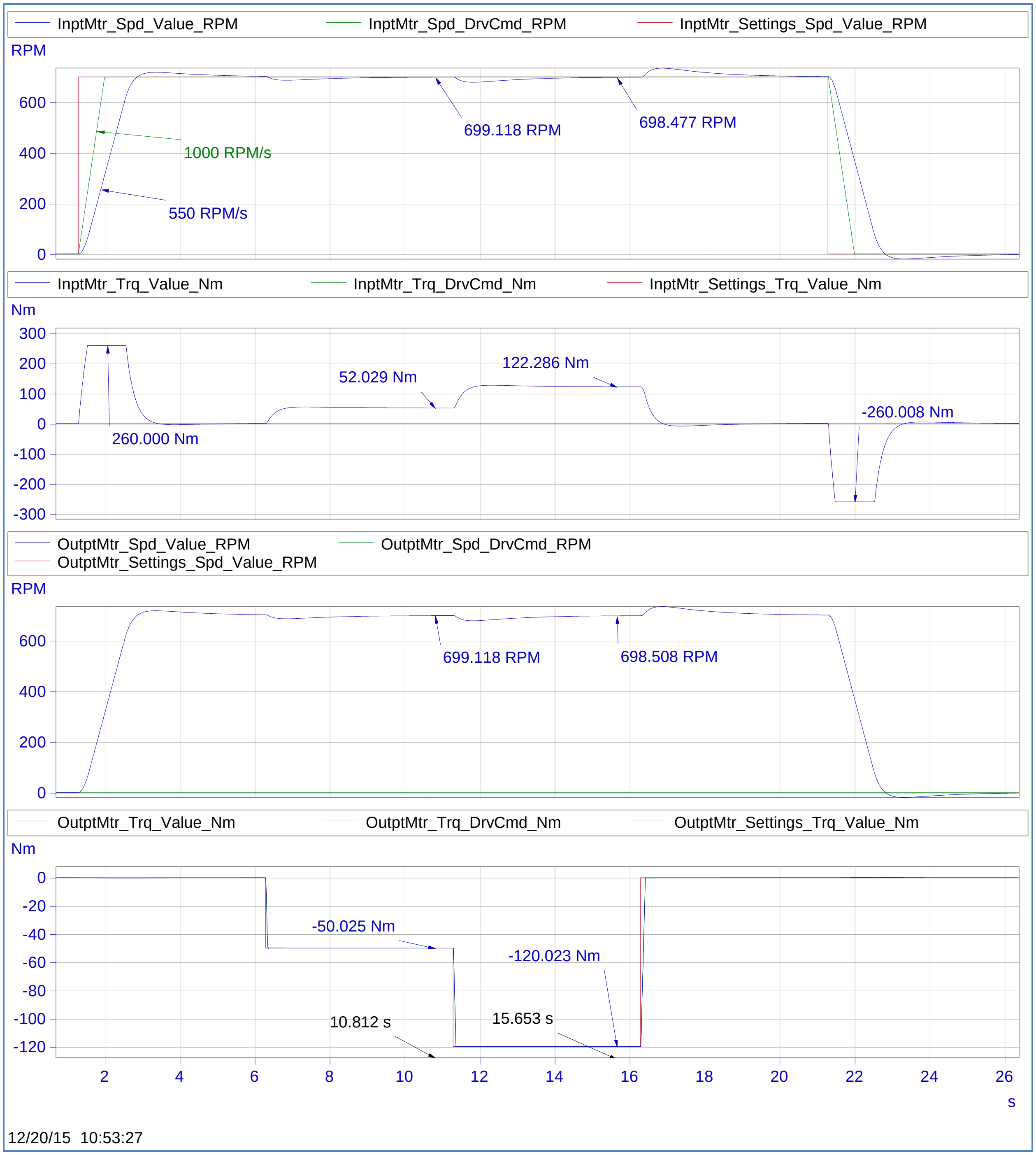

Figure 3.6: Test 1 - Simulated test cell

Figure 3.6 shows the overall results of Test 1 for the Simulated Test Cell. As can be seen, the input motor torque values do not oscillate at all. Other than the toque oscillations, the overall performance of the Simulated Test Cell is very close to the operation of the real test cell.

Figure 3.7: Test 1 - Simulated test cell comparison of input motor values

Figure 3.7 shows a direct comparison of the test cell data overlaid with the Simulated Test Cell data for input motor speed and torque values. Here again it is apparent that the Simulated Test Cell does not have the same oscillations which are present in the test cell data.

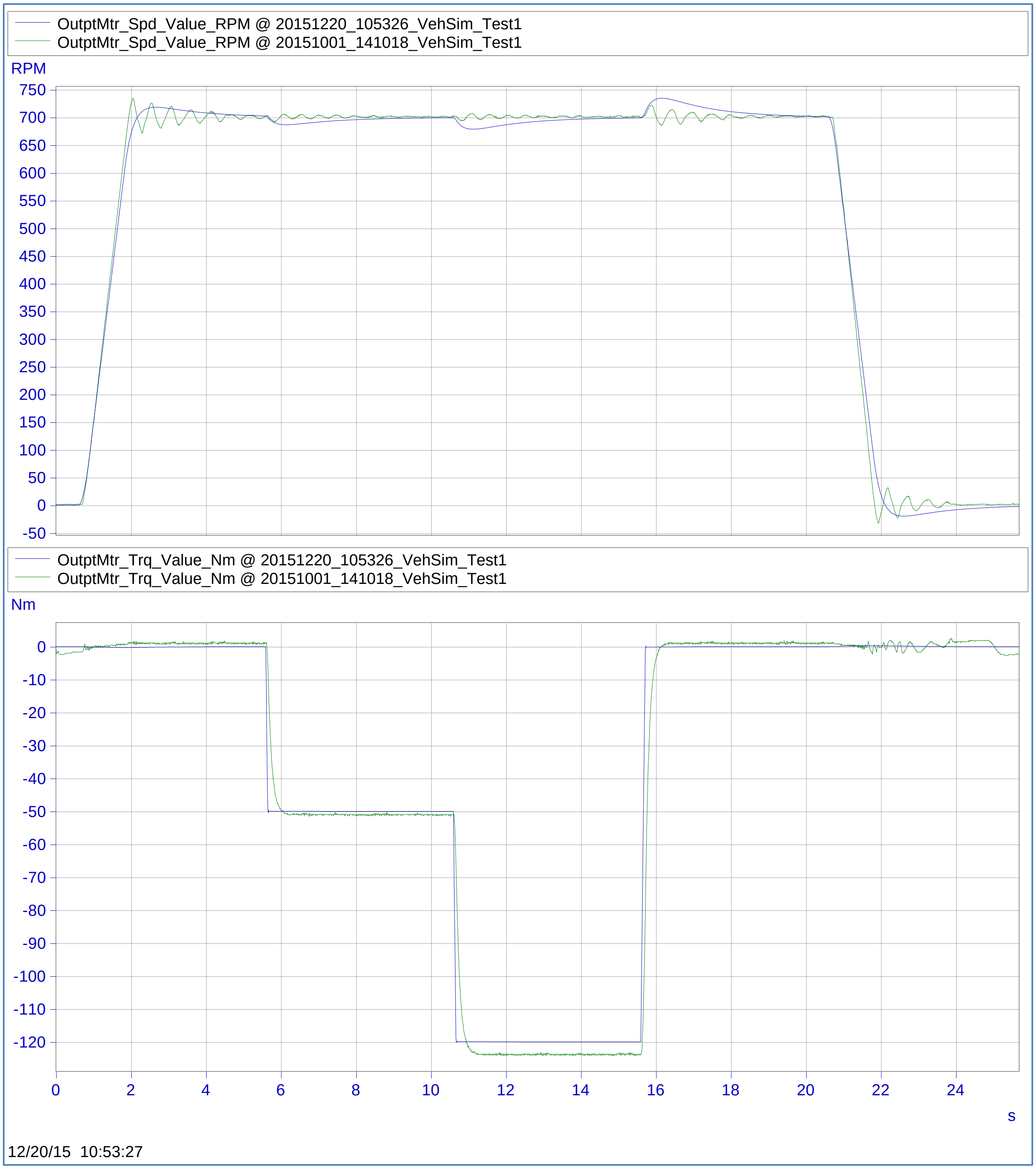

Figure 3.8: Test 1 - Simulated test cell comparison of output motor values

Figure 3.8 shows the comparison of the output motor torque and speed values for the Simulated Test Cell and the test cell. The test cell output motor speed does not contain as many speed oscillations as the input motor, and none are present in the Simulated Test Cell speed data. Also, the torque steps of the output motor are very close between simulation and reality.

3.3 Test 2: Input Motor Torque Steps – Output Motor Speed Control

The second system verification test is very similar to the first test, except the output motor is in speed control and the input motor goes through a series of torque steps. For this test, it is demonstrated that the opposite motor drive control modes of Test 1 are possible, and controllable. Also, similar to Test 1, this test is run by CAN commands issued through a CANape script, sending speed and torque set points to the Vehicle Simulator.

3.3.1 Test 2: Test Procedure

Again, much like Test 1, the procedure for Test 2 is comparable except for the input and output motor control modes. The speed and torque set points are the same as Test 1, and the transmission is still in a 1:1 gear ratio.

- Same as Test 1, except for the following
- Do torque steps of 30% and 70% of the lessor rated motor torque
    - Input Motor: 172 N·m nominal torque
        - 30% ≈ 50 N·m
        - 70% ≈ 120 N·m
- Input Motor Setup
    - Torque Control Mode
    - Manual Motor Control
    - Rate Limit = 1000 N·m/s
    - Speed Limit = 1000 RPM
    - Torque Limit = 258 N·m
- Output Motor Setup
    - Speed Control Mode
    - Manual Motor Control

- Rate Limit = 1000 RPM/s
- Speed Limit = 1000 RPM
- Torque Limit = 700 N·m

- Test Sequence
  - Start the test
  - Wait 1 second
  - Bring the output motor up to 700 RPM
  - Wait 5 seconds
  - Command -50 N·m of torque on the input motor
  - Wait 5 seconds
  - Command -120 N·m of torque on the input motor
  - Wait 5 seconds
  - Command 0 N·m on the input motor
  - Wait 5 seconds
  - Bring the output motor to 0 RPM
  - End test

3.3.2 Test 2: Results and Analysis

With this test, the output motor was driving the output shaft of the transmission at a set speed, in contrast to Test 1. Since the output motor has more torque capacity than the input motor, the Test 2 results differed slightly.

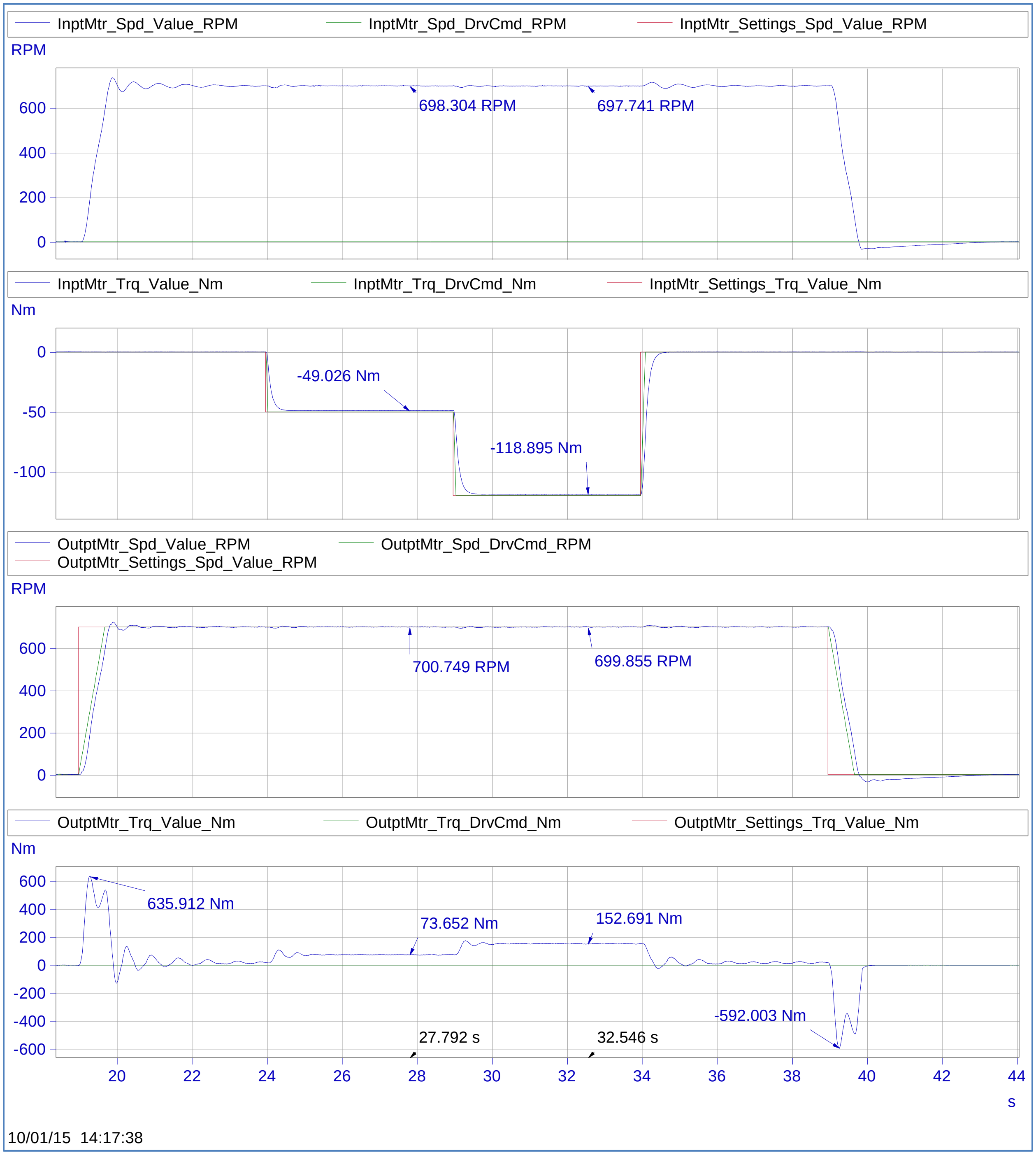

Figure 3.9: Test 2 - Overall test

Figure 3.9 shows the entirety of Test 2, with the Vehicle Simulator successfully controlling the motors as described in the test sequence. The output motor was able to hold a steady speed of nearly 700 RPM while the input motor completed the torque steps. Table 3.1 can also be used for the plots of this section as a decoder of the plot channel names.

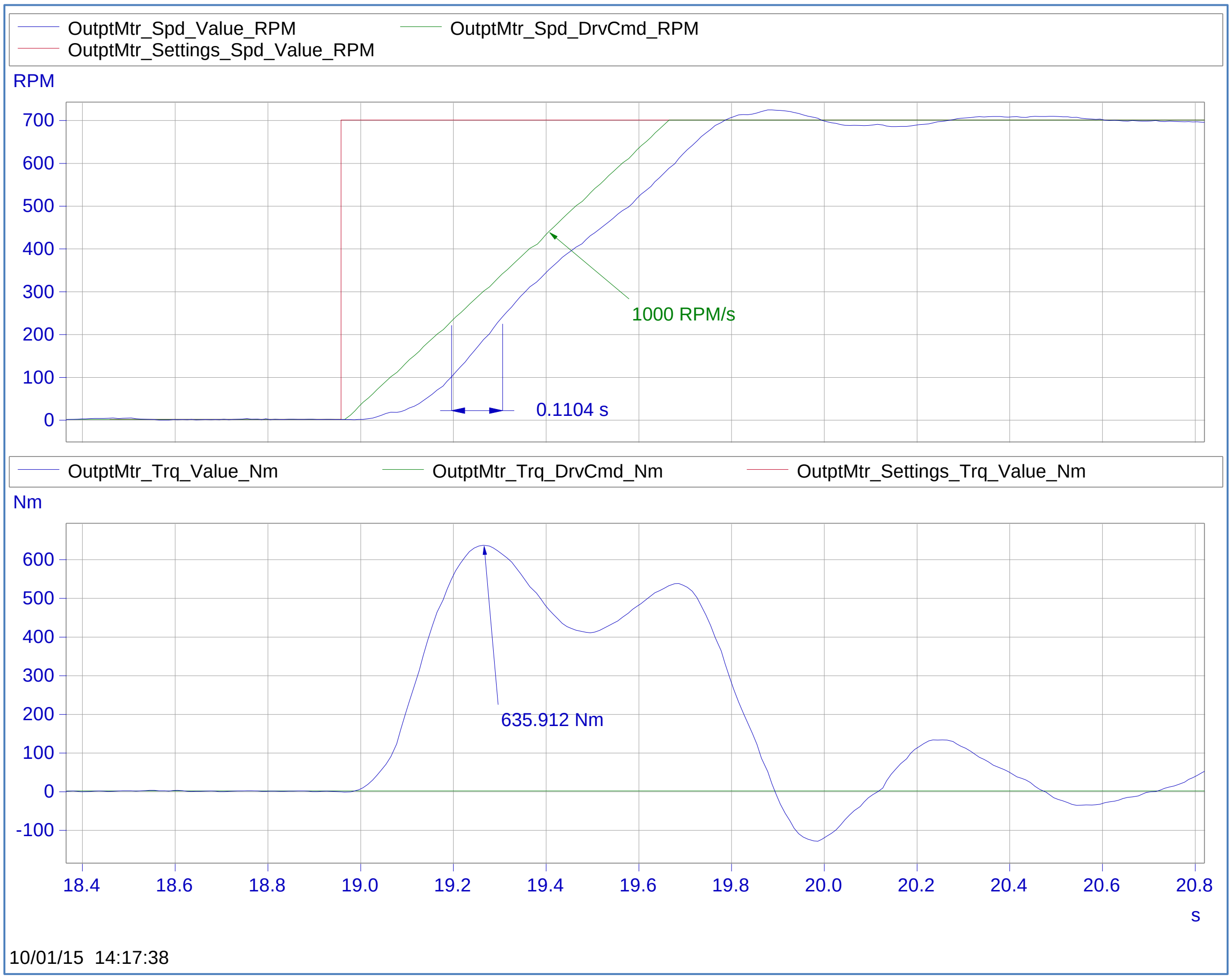

Figure 3.10: Test 2 - Speed ramp detail

Figure 3.10 shows the initial speed ramp of the output motor in greater detail. With the larger torque capacity of the output motor, it was better able to follow the 1000 RPM/s speed command from the Vehicle Simulator. Like the input motor drive speed controller, there is about the same magnitude of torque oscillations as the speed value changes state. Unlike the

input motor, the torque from the output motor does not hold steady at a maximum value, but is instead oscillates slightly as it attempts to modulate the speed. Figure 3.10 clearly shows the 0.1 second time constant filter that is present on all of the analog drive feedback signals, on both the input and output drives. This filter is configured in the motor drives themselves, and helps mitigate noise on the feedback signals, at the detriment of a small delay.

Figure 3.11: Test 2 - Speed signal detail

Figure 3.11 shows more detail of the output motor speed oscillations during the test. After the initial speed settling oscillation of approximately 40 RPM, the rest of the oscillations are only around 10 RPM. These speed oscillations are in line with what was observed from the input

motor drive speed controller, but also appear to be more heavily damped, as the oscillations to not last as long.

Figure 3.12: Test 2 - Input and output motor speed detail

Figure 3.12 presents the relationship between the input and output motor speeds for Test 2. Here, the input motor speed oscillates with greater amplitude than the output motor speed and these speeds are mostly in phase with each other. The input motor speed more closely resembles what was seen in Test 1, except with greater damping. Also, the output motor speed controller responded to the input motor torque steps with less oscillation, when compared to the input motor speed controller of Test 1.

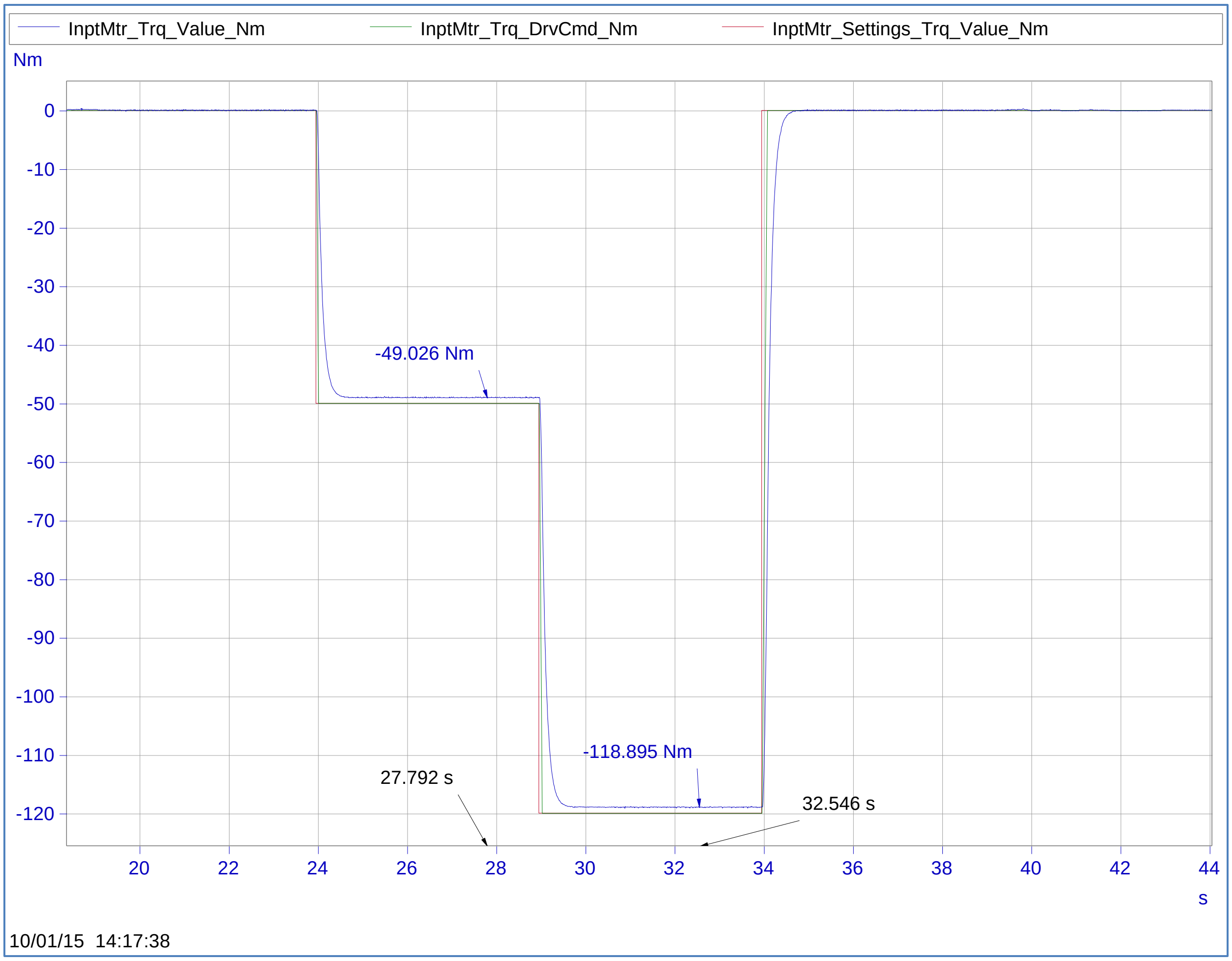

Figure 3.13: Test 2 - Input motor torque detail

Figure 3.13 shows the detail of the input motor toque steps during Test 2. The input motor is successfully commanded to both -50 N·m and -120 N·m during the test. Unlike Test 1, where the output motor torque was slightly higher than the command value, here the input motor torque is slightly below the commanded value. This offset may also be due to the filter, or a slight variance in scaling of the torque signal from analog to CAN message. This difference is only 0.43% of full scale input motor torque for the 120 N·m command, which is below the general target of 1% error or less.

3.3.3 Test 2: Comparison with the Simulated Test Cell

The Test 2 procedure was also run for the Simulated Test Cell and data was recorded.

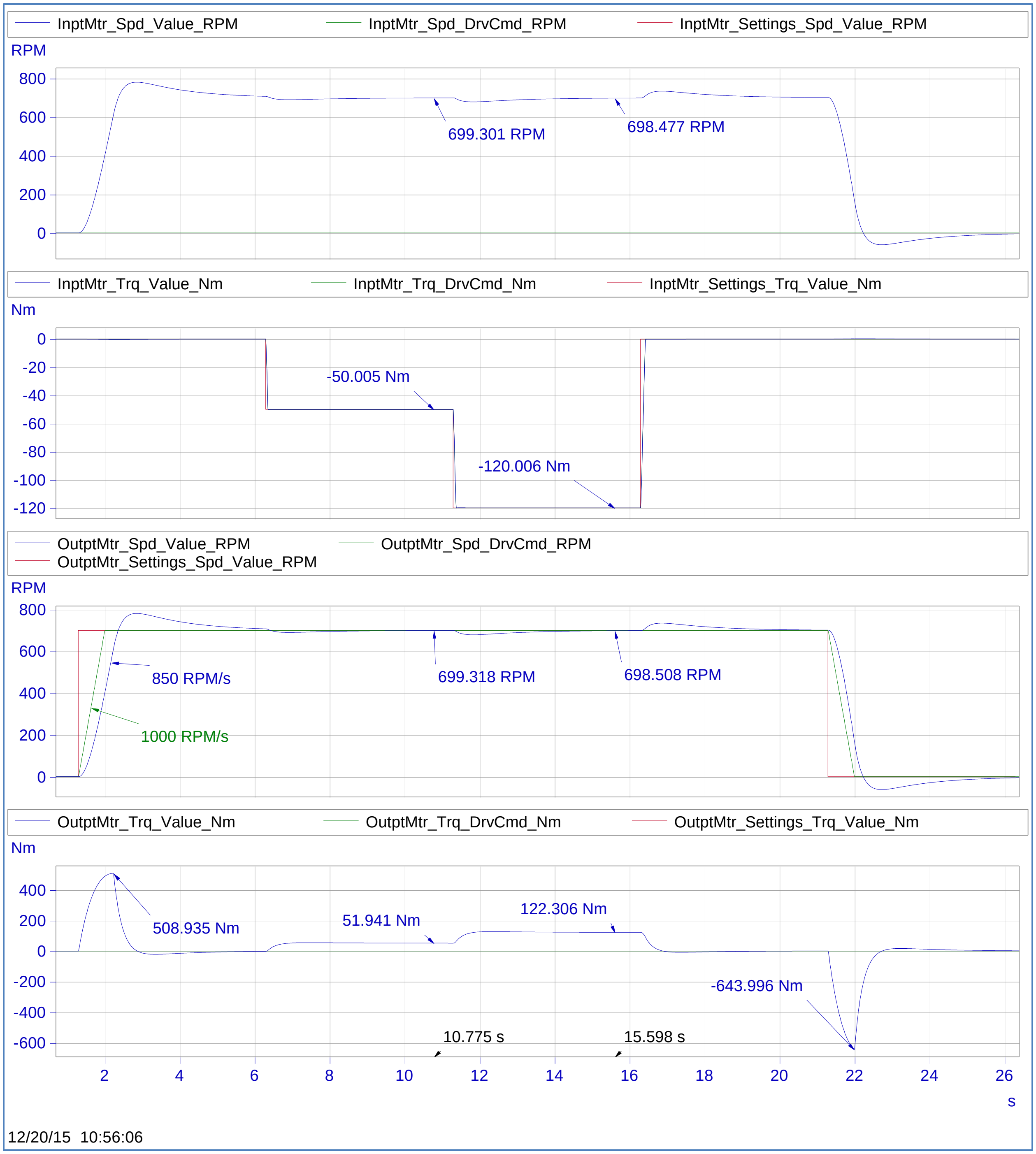

Figure 3.14: Test 2 - Simulated test cell

Figure 3.14 shows the complete execution of Test 2 for the Simulated Test Cell. As can be seen, there are no speed or torque oscillations in the data, in contrast to the actual test cell data. Also, the output motor speed overshoots the speed command set point, but settles to the value within an acceptable amount of time.

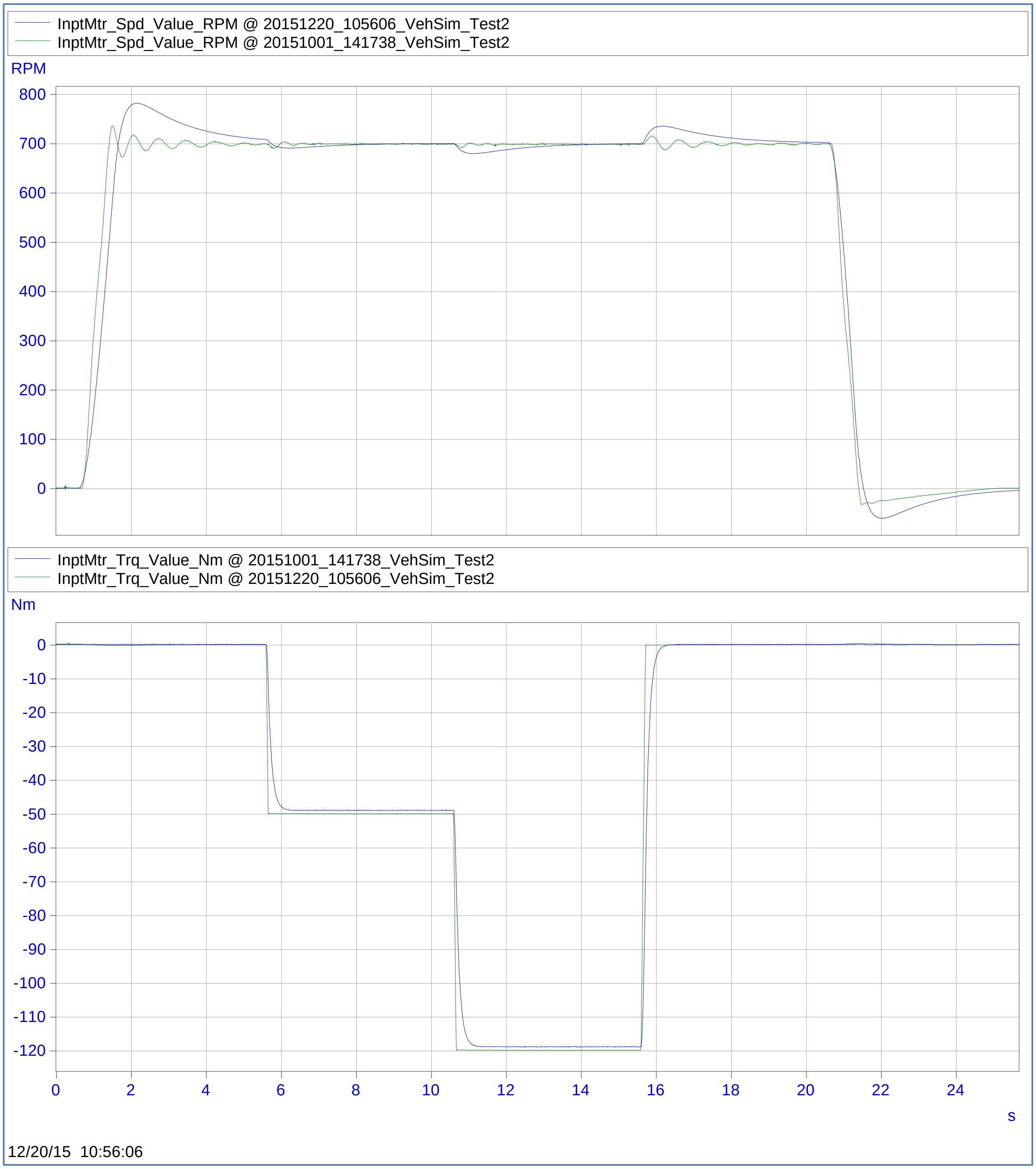

Figure 3.15: Test 2 - Simulated test cell comparison of input motor values

Figure 3.15 show the comparison of the input motor speed and torque values between the Simulated Test Cell and the actual test cell. There are oscillations in the speed and torque values of the real test cell, but not in the Simulated Test Cell; however, there is a speed overshoot in the Simulated Test Cell data, which is not present in the actual test cell data. Other than the previously noted items, the Simulated Test Cell replicates the actual test cell within an acceptable range.

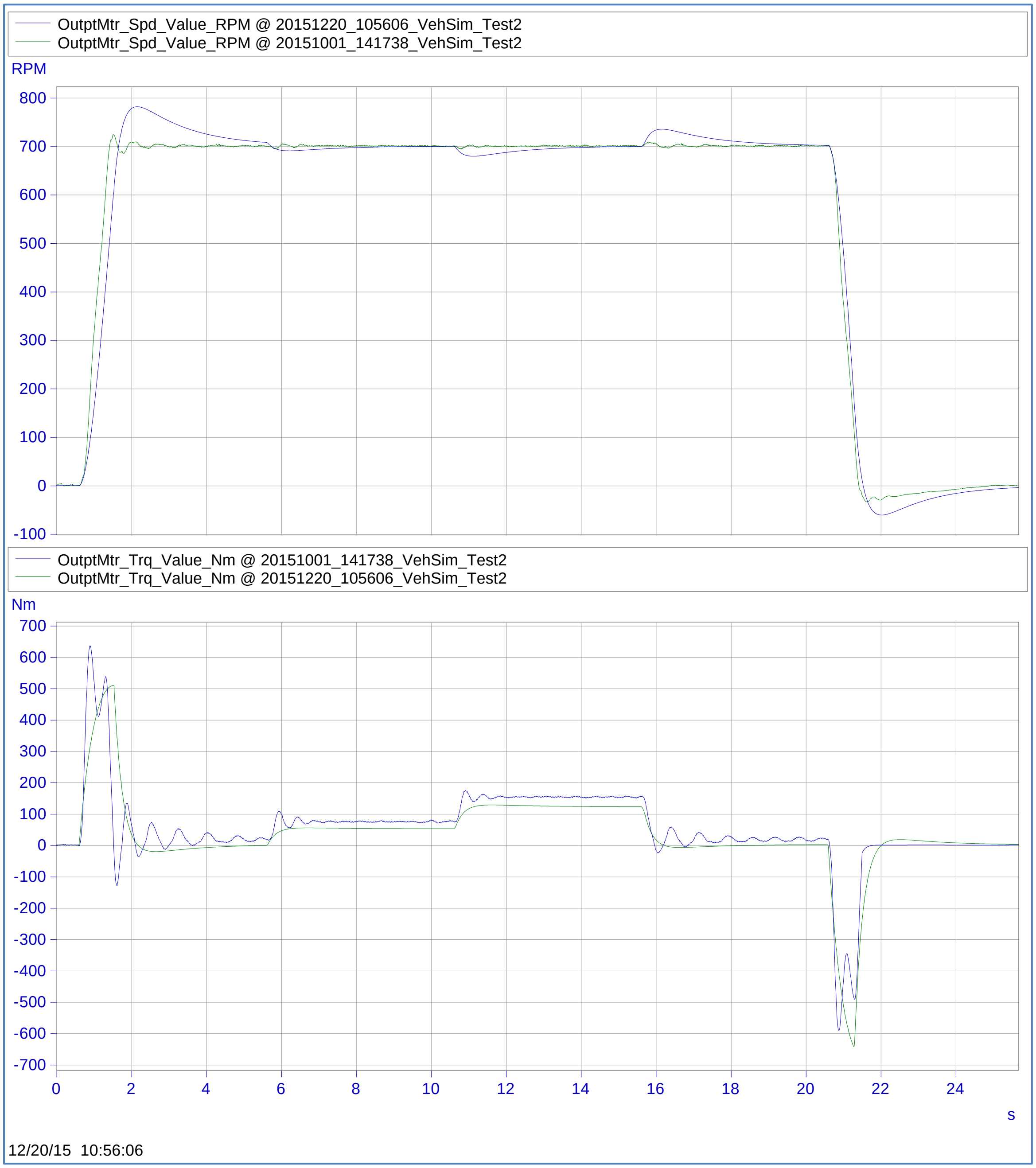

Figure 3.16: Test 2 - Simulated test cell comparison of output motor values

Figure 3.16 shows the comparison of output motor speed and torque values for Test 2 between the Simulated Test Cell and the actual test cell. It is apparent that torque oscillations are present in the actual test cell, which are not in the Simulated Test Cell. Apart from the oscillations, the Simulated Test Cell is able to replicate the actual test cell within an acceptable limit.

3.4 Test 3: Dynamic Shifting of a Transmission

The main goal of Test 3 is to provide all the necessary components that will allow a transmission to dynamically shift through several gears, while in the test cell. To accomplish this goal, the Vehicle Simulator was configured to use the Engine Plant model for the input motor control, and the Chassis Plant model for the output motor control. As described in Section 2.4, the input motor will be in torque control mode with the Engine Plant, and the output motor will be in speed control mode with the Chassis Plant model. Test 2 already demonstrated how it was possible for the input motor to operate in torque mode, and the output motor to operate in speed mode.

3.4.1 Test 3: Data Normalization

Since some of the data presented in this section is of a proprietary nature, the speed and torque signals were obfuscated and normalized before generating the following graphs. Even though the data presentation shows the recorded torque and speed signals aligned in time, the y-axes scales are not truly representative of the engineering units. The normalized signals will still show the relative magnitudes and how the plant models control the input and output motors. Also, the transmission gear is normalized to obfuscate the actual gear number. However, the accelerator pedal position and brake switch signals are not normalized, and are presented as recorded in the data.

3.4.2 Test 3: Procedure

Test 3 was run from the Vehicle Simulator user interface (VSUI), and the test cell console. The accelerator pedal and brake pedal were manually controlled by way of the VSUI, while the transmission was keyed on and put into gear through the test cell console. Generally, the following procedure would be used to run the Vehicle Simulator.

- Press the start button in the VSUI to bring the Engine Plant to idle
- Apply the brake pedal
- Shift the transmission from neutral into gear
- Set the brake pedal to zero, allowing the clutch to close
- Apply the accelerator pedal to at least 15% to start the simulated vehicle accelerating

There was no set brake or accelerator pedal profile that was run for each test. Rather, just a general procedure to provide all the necessary components for the transmission to dynamically shift. Several different test runs were taken, but only one is presented here.

3.4.3 Test 3: Results and Analysis

Since the data presented in this section was normalized, the graph signal names have changed from what was shown in Test 1 and Test 2. In addition to the motor drive feedback signals, there is also the accelerator pedal position and brake switch, as reported by the Engine Plant in their standard J1939 CAN messages.

Graph Signal Name	Description
_InptMtr_Spd_RPM_Norm	The normalized reported speed from the input motor drive to the Vehicle Simulator
_InptMtr_Trq_Nm_Norm	The normalized reported torque from the input motor drive to the Vehicle Simulator
_InptMtr_TrqCmd_Nm_Norm	The normalized torque command from the Vehicle Simulator to the input motor drive
_OutptMtr_Spd_RPM_Norm	The normalized reported speed from the output motor drive to the Vehicle Simulator
_OutputMtr_SpdCmd_RPM_Norm	The normalized speed command from the Vehicle Simulator to the output motor drive
_OutptMtr_Trq_Nm_Norm	The normalized reported torque from the output motor drive to the Vehicle Simulator
_GearNumberNorm	The normalized transmission gear
EEC2_00_AcceleratorPedalPosition	The accelerator pedal position as commanded by the Vehicle Simulator, and reported by the Engine Plant model
CC_00_BrakeSwitch	The brake switch as commanded by the Vehicle simulator and reported by the Engine Plane model

Table 3.2: Test 3 Graph signals

Table 3.2 shows the link between the normalized signal names in the following data graphs, and their descriptions.

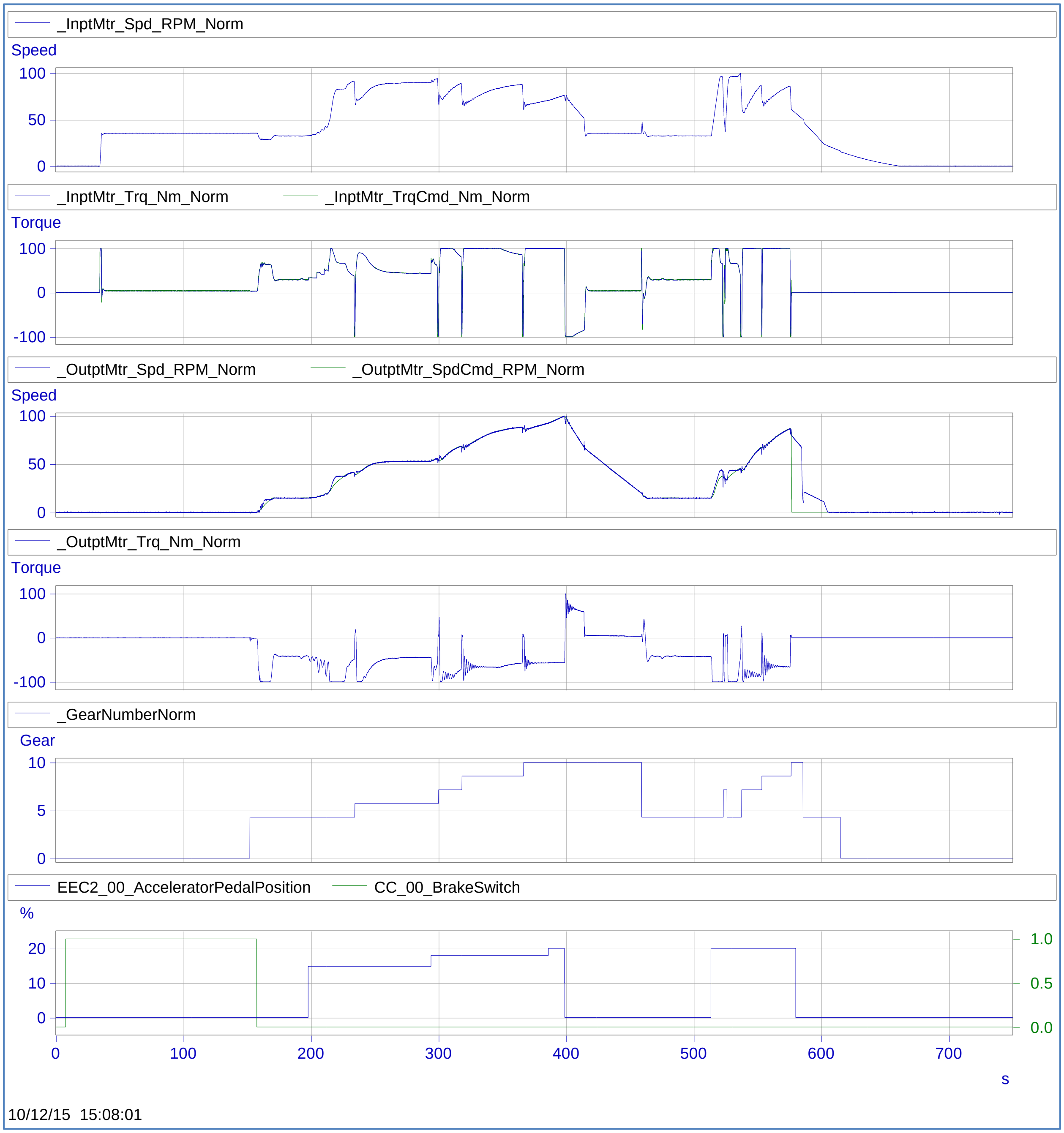

Figure 3.17: Test 3 - Overall dynamic shifting

Figure 3.17 shows the entire test, where the transmission was successfully able to dynamically shift up and down through several gears. Notable in the figure is the input motor coming up to idle speed around the 50 second mark while the brake pedal is applied, the transmission being shifted into gear around 150 seconds, and the accelerator pedal being applied around 200 seconds. For this particular test, the accelerator pedal was only varied between 15% and 20%, which provided enough torque from the Engine Plant to accelerate the Chassis Plant. Also of note, is the vehicle slowing down by way of the vehicle loads simulated in the Chassis Plant model, near the 400 second mark. The throttle pedal is set to 0%, causing the Engine Plant model to create an additional load, and slow the vehicle with the Chassis Plant model loads.

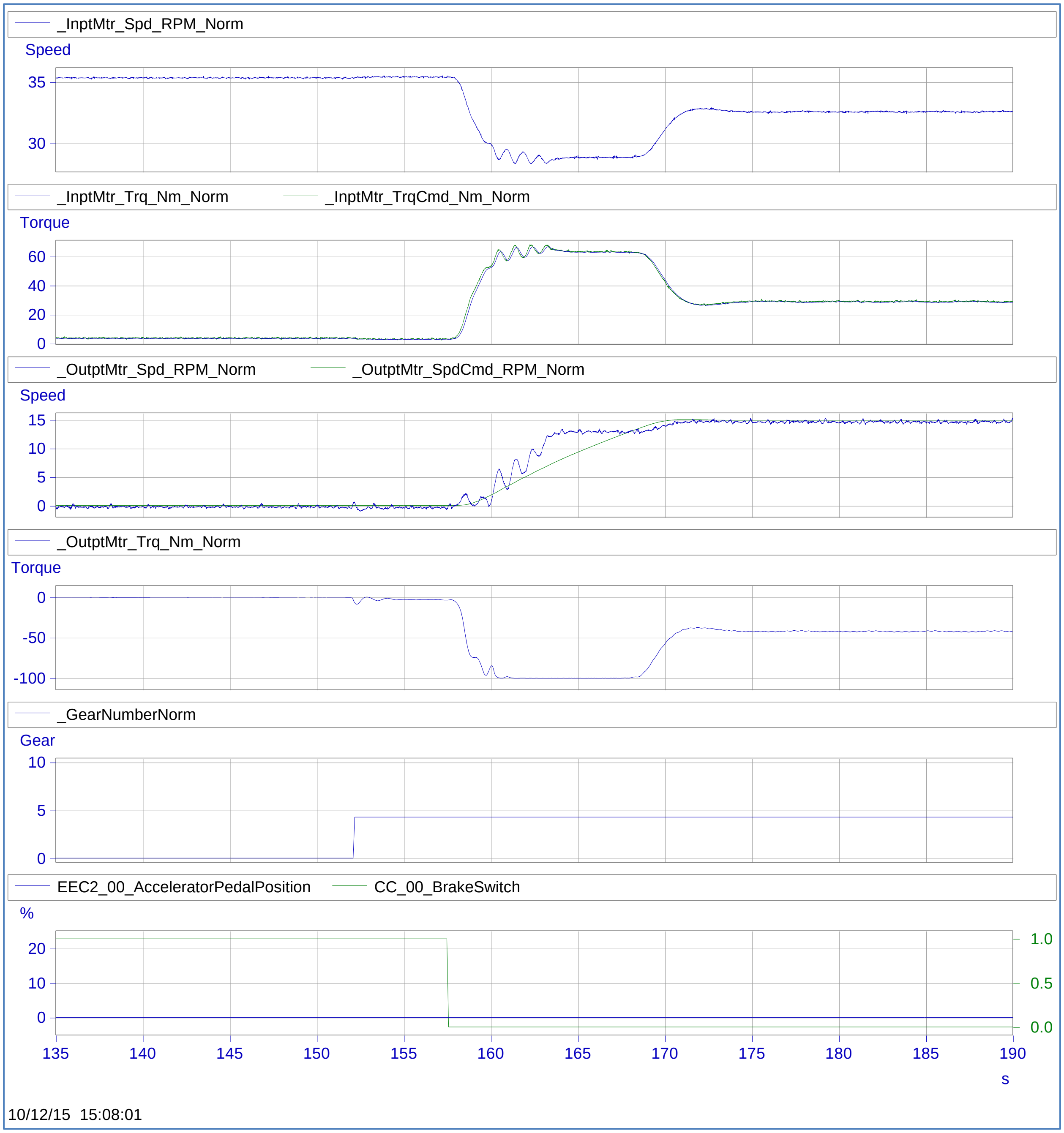

Figure 3.18: Test 3 - Launching detail

Figure 3.18 shows greater detail of the transmission launching, as the brake pedal is released while the transmission is in gear and the engine is at idle. As the clutch closes, the engine applies torque to the system to keep the idle speed constant under the new load. The engine torque also causes the Chassis Plant to increase the speed command to the output motor drive.

The increase in engine torque is enough to bring the output motor speed above the speed command from the Chassis Plant model, prompting the output drive to increase the output motor torque in opposition. Some oscillations exist during the launch, possibly the same as the oscillations observed in Verification Test 1 and Test 2.

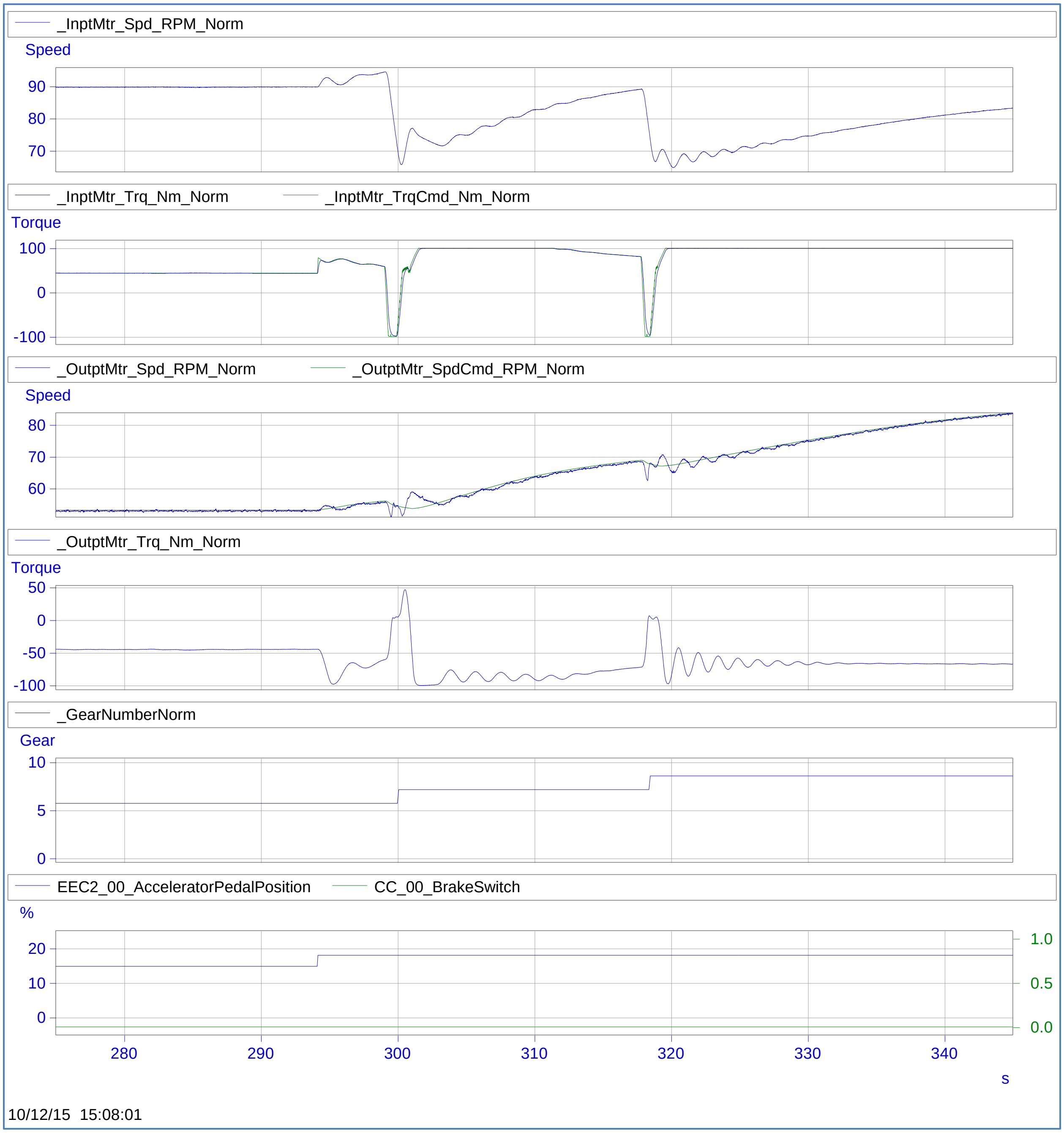

Figure 3.19: Test 3 - Shifting detail

Figure 3.19 shows a detailed view of the system accomplishing several dynamic shifts to new gears. For each shift, the engine torque is commanded to a negative value, then resumes its previous value after the shift completes. The torque change is due to the transmission sending an engine control message that the Engine Plant accepts, as described in Section 2.4.1.1. The transmission is requesting the engine to go to zero torque, but the engine torque actually goes negative due to the engine torque loads. This negative torque caused the engine speed to drop until it matched the target speed of the new gear. When the target is met, the transmission closes the clutch, and releases its control of the engine torque, allowing the engine torque to increase once again to its normal operating value.

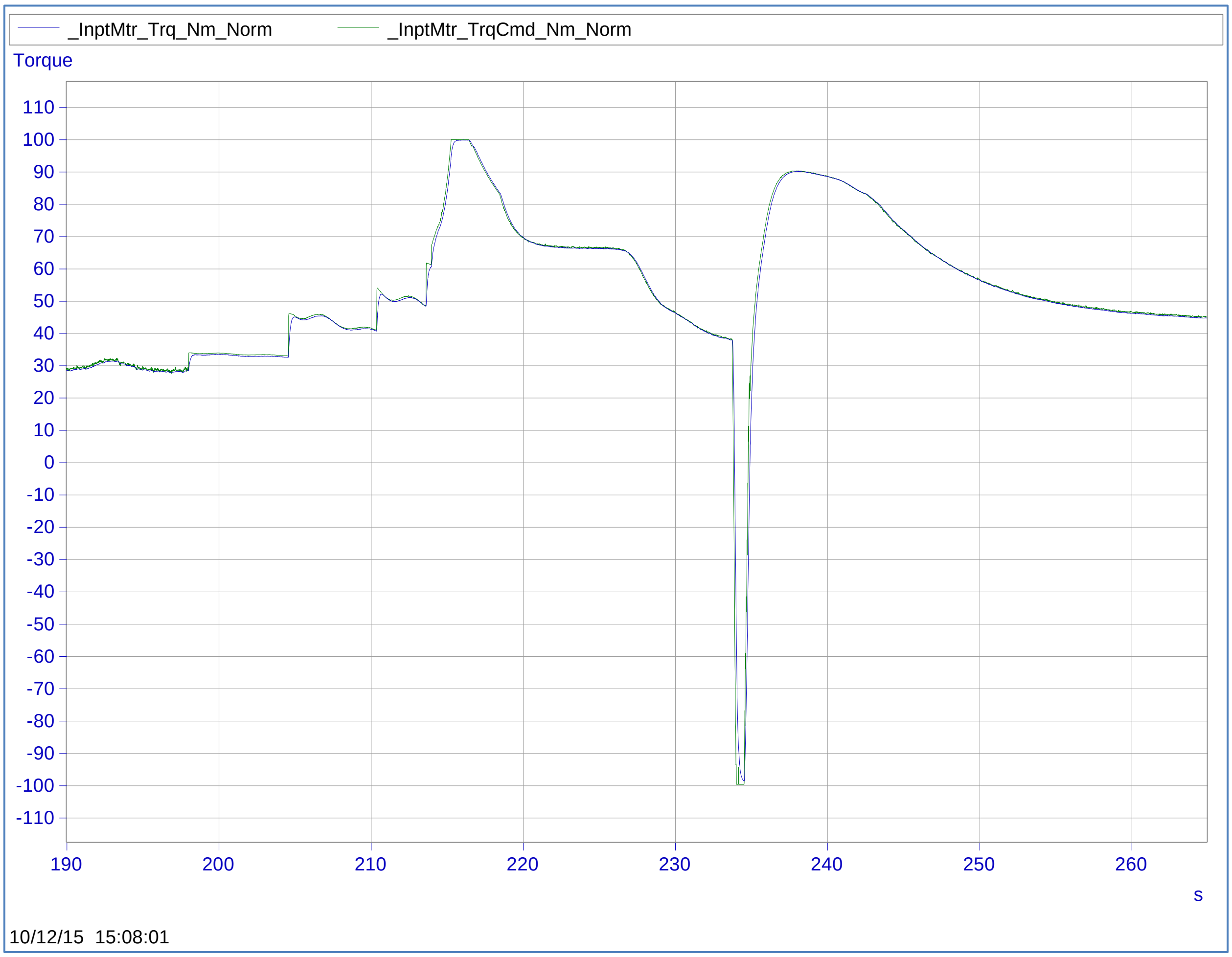

Figure 3.20: Test 3 - Input motor torque command detail

Figure 3.20 shows the detail of the reported input motor toque overlaid with the drive command from the Engine Plant model. With the input motor drive in torque control mode, it

is able to closely follow the command from the Engine Plant model. Near 205 seconds and 211 seconds, it can be seen that there is slight variation, mainly due to the high rate of change of the Engine Plant command signal.

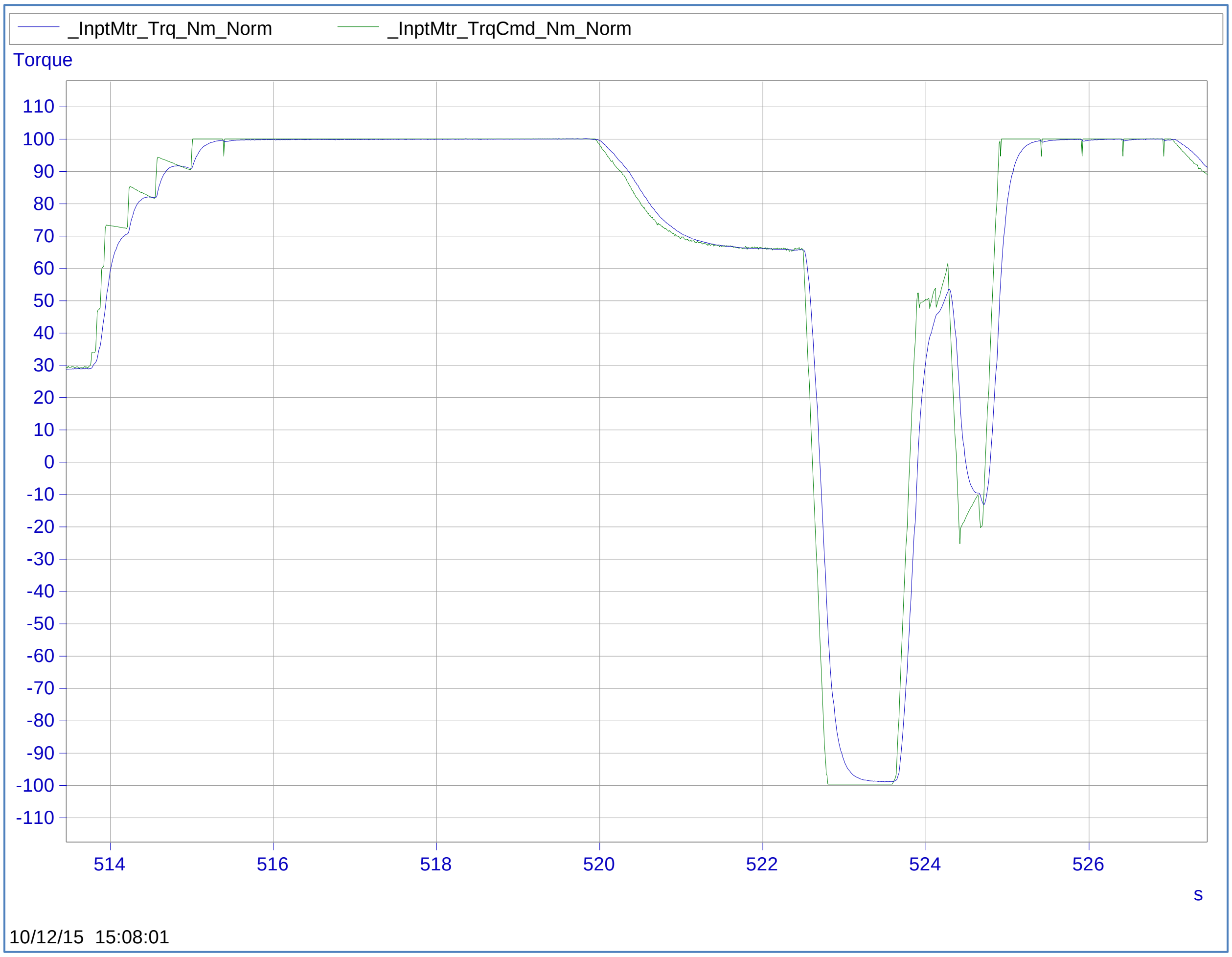

Figure 3.21: Test 3 - Input motor torque command additional detail

Figure 3.21 shows the same signals as Figure 3.20, but at a different point in the test. Here in a slightly expanded time scale, it can be seen in greater detail how the reported input motor torque follows the Engine Plant model command. The input motor torque is still following closely, but affected by the 0.1s time constant filter from the reported torque of the input motor drive. Although the torque is not exactly following the Engine Plant command, it is close enough for the transmission to believe it is indeed being driven by a real engine.

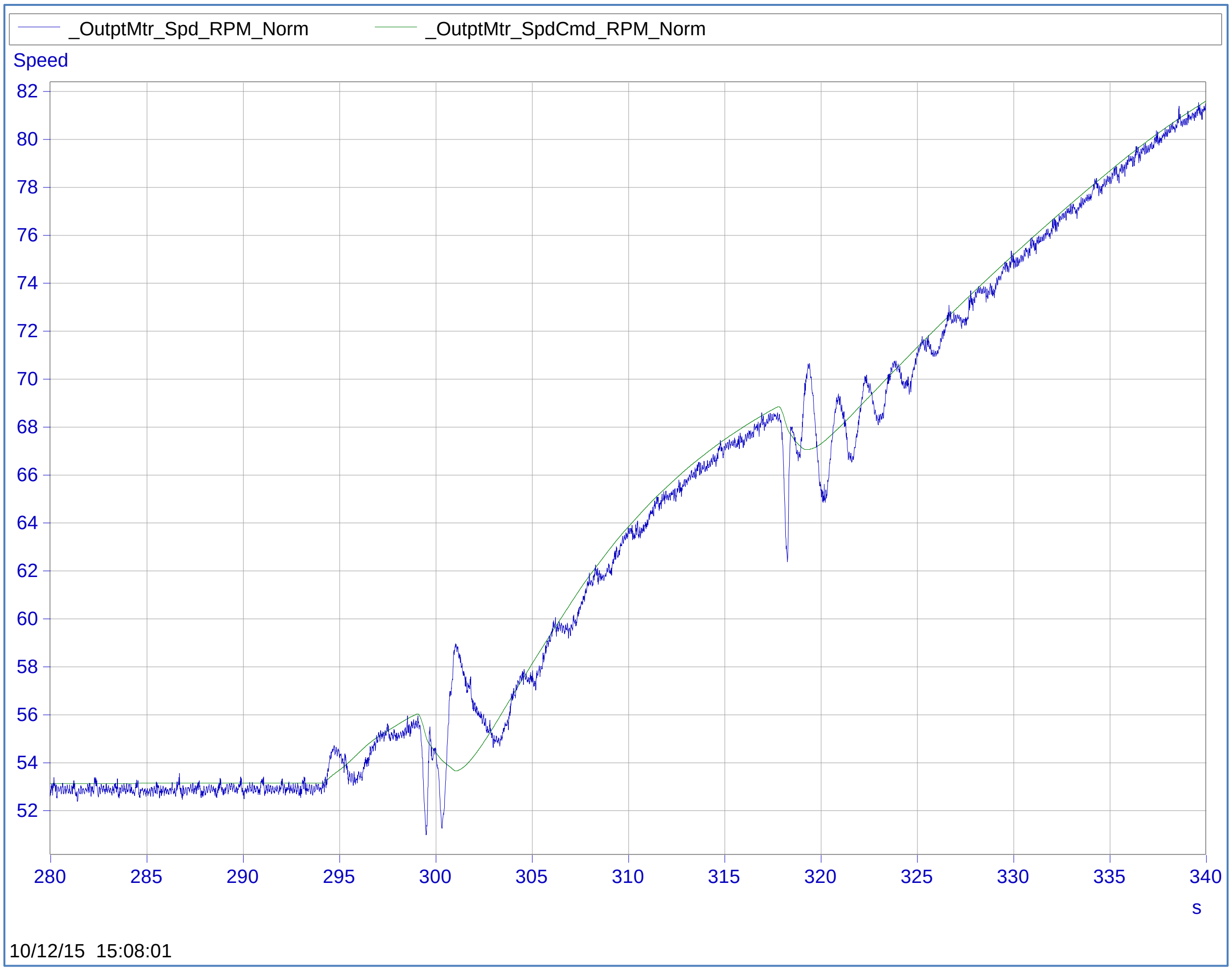

Figure 3.22: Test 3 - Output motor speed command detail

Figure 3.22 shows a detailed view of the reported output motor speed overlaid with the Chassis Plant model speed command. The output motor was able to successfully follow the general trend of the Chassis Plant speed command, in this case, through several shifts. After each shift, the output motor speed will oscillate slightly, until enough time has passed for the damping to take effect.

3.5 Additional Verification Items

In the interest of brevity, some functionally of the Vehicle Simulator has been verified, but intentionally not presented here. To provide test data for all control input scenarios would have required an extensive verification test section. Namely, the following items were verified, although omitted in this paper.

- It is possible to utilize the analog throttle and brake pedals in all motor control modes
  - The throttle pedal was used to control the input motor speed and torque modes, as well as the throttle of the Engine Plant model
  - The brake pedal was used to control the output motor speed and torque modes, as well as acting as the brake for the Chassis Plant model
- The input and output motors can be manually controlled by way of the Vehicle Simulator user interface
- In the Chassis Plant model, applying the brake when the vehicle is moving will apply a brake torque and slow the vehicle down
- Vehicle Simulator safety features
  - The watchdog relay does open when the pulse output stops for a predetermined amount of time
  - The user cannot switch motor control modes unless the Vehicle Simulator is in a safe state

The following items have been implemented, but not fully tested yet.

- Additional vehicle load when the Chassis Plant model grade value is increased
- CAN control of the throttle and brake signals for the Engine and Chassis Plant models, respectively
- Python script automation of the Vehicle Simulator user interface

4 CONCLUSION

The overall goal of the Vehicle Simulator was to implement a flexible system that could facilitate many different transmission testing scenarios, including vehicle simulation, with an emphasis on component reuse where available. Such a system has been developed and is currently providing value within the organization, while also generating positive user feedback. Vehicle emulation was achieved by repurposing plant models that were already in use within the company, and the entire system was implemented with off-the-shelf hardware and software from a proven supplier. It is very likely that the component reuse dramatically decreased the overall system development time, compared with building the entire system from the ground up. With the flexibility of the Vehicle Simulator hardware and software, this system should be easily expandable to other existing test cells within the company. Utilizing off-the-shelf hardware and software increases the likelihood of further system development, and continued technical support. The combination of both traits helps to ensure that the Vehicle Simulator system will remain relevant, and able to deliver real value for some time to come.

4.1 System Performance Summary

Three verification tests were presented, demonstrating the functional integration of the Vehicle Simulator into the test cell, several different control methods, and a number of operating modes. The results of the verification tests allow for a comparison of the Vehicle Simulator system performance to the specifications provided in Section 1.1.1 on page 3. It was shown that the input motor could be utilized in speed control mode, in torque control mode, or as an engine emulator. Likewise, the output motor could be operated in speed control mode, in torque control mode, or as a chassis emulator. Each control mode is mutually exclusive and can be freely combined with any control mode of the other motor, meeting the motor control mode requirements of specification 3. Also shown are the multiple control inputs the user can use to run different tests, addressing the requirements of specification 4, except for the standard scripting language of specification 4.2. Verification Test 1 and Test 2 demonstrated that the CAN message inputs worked for the manual motor control modes and Verification Test 3 showed control the user interface was possible. Verification Test 3 also showed that full vehicle

emulation was possible, and all the components were provided for a transmission to dynamically shift through several gears. The vehicle emulation meets the requirements of specification 1, with the exception of specification 1.3.1 road grade, as mentioned in Section 3.5 Additional Verification Items. The Vehicle Simulator system is comprised of existing engine and Chassis Plant models as well as commercially available ETAS hardware and software, meeting the requirements of specification 2.

4.2 Future Work

Although the majority of system functionality was realized, several items still need to be addressed. As shown in the verification tests, both the input and output motors had some speed oscillations when in speed control mode. Both motor drives should be tuned to better perform when the Vehicle Simulator is controlling the system under test. Along those same lines, the Simulated Test Cell could also be tuned to better match the actual test cell. As it stands now, the Simulated Test Cell does a good job of functionally testing cell operation, but not to the extent of mimicking the oscillations witnessed in the data of the verification tests. As the actual motor drives are tuned, it likely will be easier to adjust the Simulated Test Cell to match the physical test cell characteristics. Another outstanding item is automating the user interface control with Python scripts. A successful proof of concept script has been developed to control the Vehicle Simulator user interface to recreate Verification Test 1 with no safety checks. The initial test script was based on the extensive work already done for the HIL stands, which needs to be expanded to more control inputs and control modes.

Another more extensive item is finishing the test cell integration, to enable full Vehicle Simulator control. In the interest of getting the Vehicle Simulator functional in the timeframe provided, complete control of the transmission user input was not achieved. As currently implemented, the only way the user is able to interact with the transmission is through the test cell console. Here, the user must key on the transmission with a switch much like what is found in an actual vehicle, and put the transmission into gear using a shift interface device. Ideally, the key and shift interface device at the test cell console would be one option for interacting with the transmission, with the other being through the Vehicle Simulator. Once implemented,

the user would be able to key on the transmission and shift into gear through the Vehicle Simulator user interface. With that, additional features could be automated by Python scripts, like shifting into gear, manually selecting other keys, or cycling power many times.

BIBLIOGRAPHY

[1] Eaton, "Vehicle," 2015. [Online]. Available: http://www.eaton.com/Eaton/ProductsServices/Vehicle/index.htm. [Accessed 9 November 2015].

[2] Eaton, "Chart," 2015. [Online]. Available: http://www.roadranger.com/rr/ProductsServices/ProductsbyCategory/Transmissions/Chart/index.htm. [Accessed 9 November 2015].

[3] Syracuse University, "The Seven Steps of the Engineering Design Process," 2015. [Online]. Available: http://engineeringonline.syr.edu/resource/the-seven-steps-of-the-engineering-design-process. [Accessed 9 November 2015].

[4] ETAS, "ETAS - LABCAR Component Overview - LABCAR Component Overview - Test and Validation - Solutions - ETAS Products," 2015. [Online]. Available: http://www.etas.com/en/products/solutions_labcar_component_overview.php. [Accessed 19 December 2015].

[5] R. Ahlawat, S. Jiang, D. Medonza and M. Smith, "On emulating engine and vehicle transient loads for transmission-in-the-loop experiments," *Mechatronics,* vol. 22, no. 7, pp. 989-996, 2012.

[6] J. Steiber, B. Surampudi, M. Rosati, M. Turbett and et al., "Modeling, Simulation, and Hardware-in-the-Loop Transmission Test System Software Development," *SAE Technical Paper 2003-01-0673,* no. doi:10.4271/2003-01-0673, 2003.

[7] M. Castiglione, G. Stecklein, R. Senseney and D. Stark, "Development of Transmission Hardware-in-the-Loop Test System," *SAE Technical Paper 2003-01-1027,* no. doi:10.4271/2003-01-1027, 2003.

[8] R. Allen, T. Rosenthal, B. Aponso, D. Klyde and et al., "A Low Cost PC Based Driving Simulator for Prototyping and Hardware-In-The-Loop Applications," *SAE Technical Paper 980222,* no. doi:10.4271/980222, 1998.

[9] F. Boissinot, J. Bellavoine, A. Shabashevich and S. Puster, "Automated Calibration for Transmission on Powertrain Dynamometers," *SAE Technical Paper 2015-01-1625,* no. doi:10.4271/2015-01-1625, 2015.

[10] W. Li, X. Shi, D. Guo and P. Yi, "A Test Technology of a Vehicle Driveline Test Bench with," *SAE Technical Paper 2015-01-1303*, no. doi:10.4271/2015-01-1303, 2015.

[11] B. Guvenc, B. Sencer, M. Giray and L. Guvenc, "Use of a Simulink engine blockset in real time hardware in the loop simulations," in *IEEE ICM 2004 International Conference on Mechatronics*, 2004.

[12] R. Isermann, J. Schaffnit and S. Sinsel, "Hardware-in-the-loop simulation for the design and testing of engine-control systems," *Control Engineering Practice,* vol. 7, pp. 643-653, 1999.

[13] J. Zhang, S. Gao, F. Jiang and W. Huang, "A Diesel Engine Simulation System Based on Vx Works RTOS," in *IEEE International Conference on Vehicular Electronics and Safety*, Beijing, 2006.

[14] O. M. U. CORP, "Stepper Motor Glossary of Terms," 2015. [Online]. Available: http://www.orientalmotor.com/technology/articles/stepper-motor-glossary-of-terms.html. [Accessed 15 September 2015].

[15] T. Athens, "Which Way Does My Engine Turn?," Seaboard Marine, 1 January 2009. [Online]. Available: http://www.sbmar.com/articles/which_way_does_my_engine_turn/. [Accessed 15 September 2015].

[16] A.-D. GmbH, "PCI-Express 8/16 relay board APCIe-2200 - ADDI-DATA," [Online]. Available: http://addi-data.com/datasheets/datasheet_apcie-2200.pdf. [Accessed 15 September 2015].

[17] P. D. T. Prof. Bill Messner, "Control Tutorials for MATLAB and Simulink - Motor Speed: System Modeling," 2014 . [Online]. Available: http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SystemModeling. [Accessed 19 September 2015].

[18] M. Abramowitz and I. A. Stegun, "Handbook of Mathematical Functions," 1972. [Online]. Available: http://people.math.sfu.ca/~cbm/aands/page_73.htm. [Accessed 13 May 2015].

[19] G. Elert, "Aerodynamic Drag - The Physics Hypertextbook," [Online]. Available: http://physics.info/drag/. [Accessed 15 May 2015].