CODEN:LUTEDX/(TEIE-5443)/1-41(2020)

Industrial Electrical Engineering and Automation

Continued Integration of Simulated Vehicle Model for Software Verification in CANoe and Simulink

Johan Persson

Division of Industrial Electrical Engineering and Automation
Faculty of Engineering, Lund University

Continued integration of simulated vehicle model for software verification in CANoe and Simulink

Johan Persson

Division of Industrial Electrical Engineering and Automation

Abstract

This Master's thesis investigates if a previously integrated Simulink based vehicle model works for testing and verifying software on an ECU (electronic control unit) used to control a four-wheel drive coupling in cars. The testing is done through a specific program called CANoe that allows the tester to both develop and run test cases. This work also investigates the possibility to use static evaluation requirements in a dynamic model. Static evaluation means that different vehicle parameters are set to a certain value, and the answer generated from the software is checked with a predetermined value. The simulated model is dynamic, which means that by setting one vehicle parameter to a specific value might change the value of another parameter. By using a simulated vehicle instead of a real one it is possible to get a good estimation of how the product will fare to a fraction of the cost. The investigation showed that the model was working as intended, and that using static evaluation requirements is possible but not optimal compared to dynamic requirements. This thesis was done at TVR-SW (software verification) department at BorgWarner in Landskrona, Sweden.

Sammanfattning

Det här examensarbetet undersöker om en redan integrerad, Simulink-baserad, fordonsmodell fungerar för testning och verifiering av mjukvara på en ECU (electronic control unit) som används till att styra kopplingen till ett fyrhjulsdrift-system i bilar. Testningen sker i ett program som heter CANoe som låter testaren både utveckla och köra testfall. Arbetet undersöker även möjligheten att verifiera statiska krav med en dynamisk modell. Statiska krav innebär att olika fordonsparametrar sätts till ett specifikt värde, och att svaret som genereras av mjukvaran jämförs med ett förutbestämt värde. Fordonsmodellen är dynamisk, vilket innebär att när en fordonsparameter sätts till ett specifikt värde så ska en annan parameter ändras. Genom att använda ett simulerat fordon istället för ett riktigt går det att få hög noggrannhet till en bråkdel av kostnaden. Undersökningen visade att modellen fungerade som avsett, och att det är möjligt att använda statiska krav men att dynamiska krav är att föredra. Det här examensarbetet utfördes på avdelningen för mjukvarutest (TVR-SW) på BorgWarner i Landskrona, Sverige.

Acknowledgements

Firstly, I would like to thank my supervisor at BorgWarner, Richard Pendrill, for always answering my questions and offering vital support whenever I got stuck, as well as providing continuous feedback on the structure of the project. I would also like to thank Måns Andersson and BorgWarner for coming up with the idea for this project and for supplying the equipment required. The rest of the TVR-SW team needs a big thank you as well for offering help whenever Richard was unavailable. Lastly, I would like to thank Gunnar Lindstedt at Lund University for always offering good advices and calming answers.

Table of Contents

Abbreviations

AWD - All Wheel Drive
CAN - Controller Area Network
CAN FD - Controller Area Network Flexible Data rate
CANoe – CAN open environment
CAPL – CAN access programming language
CSW - Control Software
ECU - Electronic Control Unit
ETC – European Tech Center
FWD - Front Wheel Drive
FXD – Front Cross Differential
TVR-SW – Test, Validation and Reliability - Software
LIN – Local Interconnect Network
OEM – Original equipment manufacturer
RWD - Real Wheel Drive
SPA – Scalable Product Architecture (Volvo Vehicle Platform)
PDS - PowerDrive Systems

1. Introduction

BorgWarner Inc. is an American automotive component and parts manufacturer. They produce a wide array of components ranging from turbos to traction control systems and a lot in between. There are different subsidiaries that focus on different products and the company has facilities in 67 locations in 19 countries[1].

1.1 BorgWarner European Tech Center

This thesis was done at the European Tech Center (ETC) in Landskrona. ETC is a part of BorgWarner PDS (PowerDrive Systems), a subsidiary of BW which mainly specializes in torque transfer solutions in cars and trucks. By transferring torque from two wheels to all four wheels, which is what BorgWarner's solutions do, it is possible to increase traction and stability. The way the torque is transferred is the same across most of BorgWarner products, but varies with vehicle configuration. Different vehicle configurations will be covered in more detail later in Chapter 1. The most common configuration and most manufactured coupling in Landskrona is the front wheel drive (FWD) car with all-wheel drive (AWD) capabilities.

The torque is transferred through an electro-hydraulic coupling which is controlled by an ECU(electronic control unit). Based on information from sensors on the car the ECU controls a hydraulic pump that engages or disengages the coupling in order to transfer torque to all four wheels, thus increasing traction. The ECU is mounted directly on the pump which in turn is located on the coupling. This can be seen in Figure 1-1.

Figure 1-1:Electro-hydraulic AWD-coupling. Pump is located on the bottom, with ECU sticking out from the side[2].

The torque transfer systems developed in Landskrona are of the "on-demand" type, which means that torque is only sent to all four wheels when needed. This reduces the overall mechanical losses from the power transfer which in turn improves fuel economy compared to systems with constant AWD, while still offering the same amount of control and stability.

1.1.1 Front wheel drive with AWD

The most common coupling manufactured by BorgWarner PDS in Landskrona is the FWD with AWD capabilities. The engine is situated at the front of the car and drives the front wheels. The rear wheels are disconnected from the powertrain during normal driving conditions. When the ECU attached to the coupling detects that more traction is needed to keep the car controlled it engages the rear wheels via an electro-hydraulic coupling. The ECU gathers information from a whole range of sensors on the vehicle to determine when AWD is required. For example, the ECU constantly watches values in real time like wheel speed for each individual wheel, lateral acceleration and steering angle amongst others. This is done in order to give the driver of the vehicle full control and confidence despite varying road conditions.

1.1.2 Rear wheel drive with AWD

The electro-hydraulic coupling together with the ECU can be positioned in other ways in different vehicle configurations to adapt and give similar control as the FWD to AWD case. One example that is more common in high performance vehicles like the Lamborghini Huracán and the Audi R8 is the rear wheel drive (RWD) with AWD capabilities. In the Audi R8 for example the engine is located in the middle of the car and the torque is sent to the rear wheels. This configuration be seen in Figure 1-2. The BorgWarner coupling is then positioned so that it can send torque to the front wheels instead and in that way obtain AWD control.

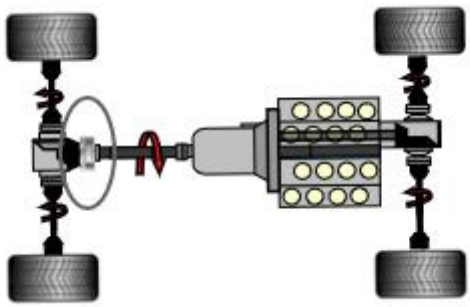

Figure 1-2: Rear wheel drive vehicle with AWD capabilities. Engine in the middle of the car [3].

1.1.3 Rear wheel drive with Transfer case

Some vehicles use a transfer case instead of the simpler FWD/RWD to AWD coupling. The transfer case solution is more common in heavier vehicles with off-road capabilities. These vehicles have the engine positioned longitudinal at the front and sends torque the rear wheels through the transfer cars, this can be seen in Figure 1-3. If and when more traction is required, the transfer case transfers torque to the front wheels by engaging a clutch inside the case with the hydraulic pump. This causes the front drive propshaft to spin until it reaches the same speed as the rear propshaft. When equal rotational speed between the two propshafts is achieved a dog-clutch is engaged and transfers torque to the front wheels. Compared to the other solutions mentioned above this feature can also be engaged manually by the driver if deemed necessary.

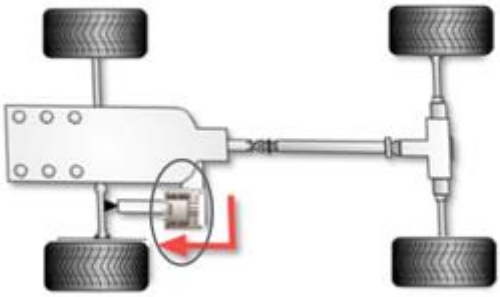

Figure 1-3: Rear wheel drive vehicle with engine in front and with transfer case [4].

1.1.4 Front cross differential

A front cross differential (FXD) is another approach to solving the issue with traction compared to the other versions mentioned above, which can be applied in vehicles with a transversally mounted engine in the front and front wheel drive. The configuration can be seen in Figure 1-4. An issue with this configuration is that during acceleration the two front wheels have different levels of grip, which will cause the vehicle to steer towards the side of lower grip. This is commonly called torque steer and is especially a problem in high powered front wheel drive vehicles. Front wheel drive cars can often suffer from understeer during cornering as well. The FXD coupling counters these behaviours by distributing up to 100% of the power to either wheel through an electrically activated limit slip differential (eLSD) and thus improving traction and stability.

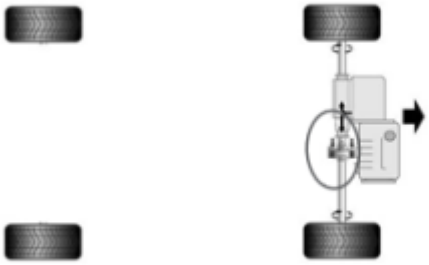

Figure 1-4: Front cross differential (FXD) on a FWD vehicle [5].

1.2 Testing at BorgWarner

In order to ensure that the products delivered fulfils the specifications set by both BorgWarner and its customers they go through extensive testing. Components used in the automotive industry are exposed to varying and tough conditions during their life cycle, and a failure at some point can lead to lethal consequences. At BorgWarner PDS in Landskrona there are two different departments that perform testing. One of the teams are responsible for mechanical testing and the other for software testing. This thesis was done at the department for software test.

At their disposal, the test teams of BorgWarner have two different test tracks available for testing. One is located near ETC at Ljungbyhed Motorbana and provides testing on tarmac. The other testing facility is located far north in Sweden and is well known in the car industry for the place where testing in cold and icy conditions are conducted. A picture of this testing ground can be seen in Figure 1-5.

Figure 1-5: Ice test tracks in the northern Swedish town Arjeplog [6].

The majority of the testing is not performed on these two tracks, however. In order to maximize efficiency and reduce cost, most of the testing is performed in a simulated environment.

1.2.1 Software testing

Most of the software testing done at BorgWarner in Landskrona is conducted by the software test team. The testing process can be split into two major groups. The first group are tests conducted in systems created by the car manufacturers, and the second group are tests that are done in an environment sold by Vector Informatik GmbH[7]. The test environment is a program called CANoe, which is a tool for designing and conducting software testing. Both groups have similar hardware setups however. In both cases an ECU, with the software that requires testing, is connected to a software test rig which is used to imitate signals that would be found in a vehicle. The software rig can also be used to induce specific errors with the purpose of testing error handling functionality. A computer running CANoe is also connected to the test rig. CANoe allows the tester to run specific test cases based on what part of the software that needs testing. The test cases control the rig which emulates signals generated by a car. These signals get sent to the ECU which response returns to the rig and then the computer for evaluation. A schematic of this setup can be seen in Figure 1-6.

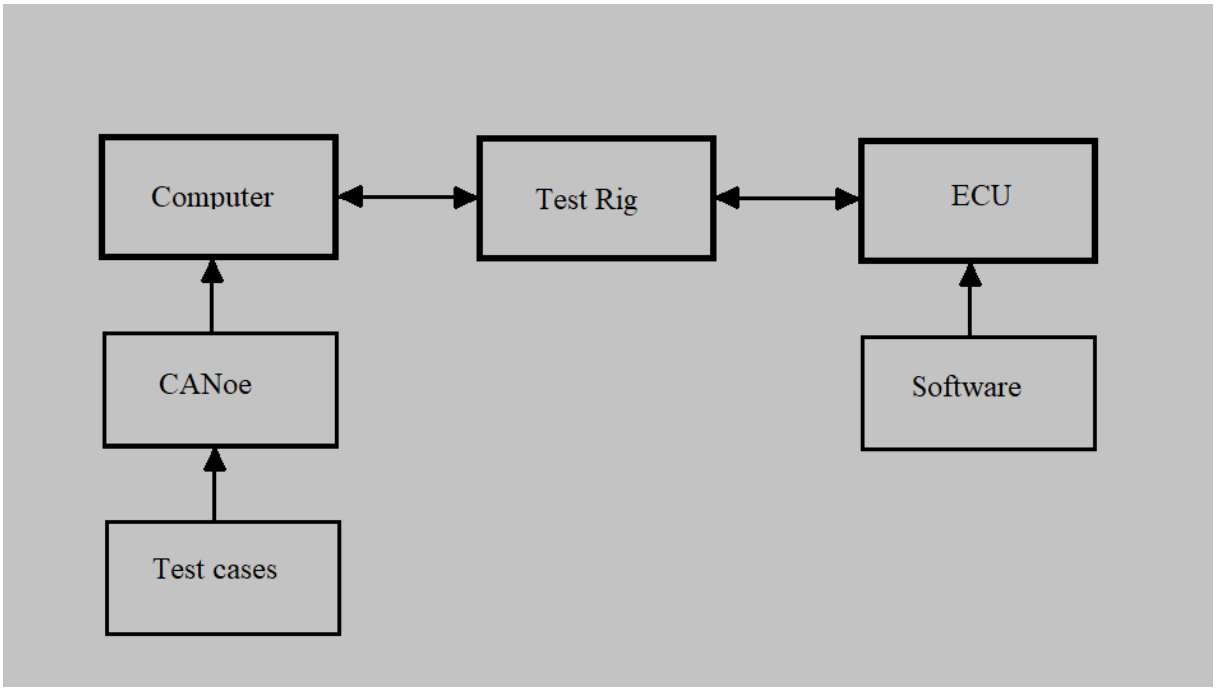

Figure 1-6:Schematic of software test setup.

The in-house testing can further be split into different test suites that cover different areas of the software. There are suites that cover the underlying functionality in the ECU and makes sure that the ECU is communicating with the other nodes in the car and properly responds to input. There are also test suites that test diagnostics, safety and error handling. The different test suites can then be further divided into different cases, but this will not be covered in this thesis.

One of the test suites handles the control software (CSW). This part of the software handles how the AWD-system is supposed to behave when subjected to different parameters like wheel speeds, steering angle, and lateral acceleration to mention a few. The CSW testing is conducted either manually through a panel in the testing environment where an engineer change different parameters and check the ECU for the correct response, or through automated test cases where the same parameters are cycled and the responses analysed. This does not represent how an actual car is driven in the real world where multiple parameters change at once and hence is there a need for simulated vehicle model to assist in testing. Most of the thesis will focus on the CSW part of testing.

1.3 Electronic Control Unit, ECU

An ECU is an embedded computer system which controls different parts of a vehicle. The ECU used for this thesis use multiple inputs from different sensors in the vehicle to determine if the AWD should be active or not. Displayed in Figure 1-7 are two different generations of ECUs used at BorgWarner. The smallest one is the latest generation (Gen VI) and is the one used for this project, and the larger ones are an older generation that is still used in some projects (Gen V).

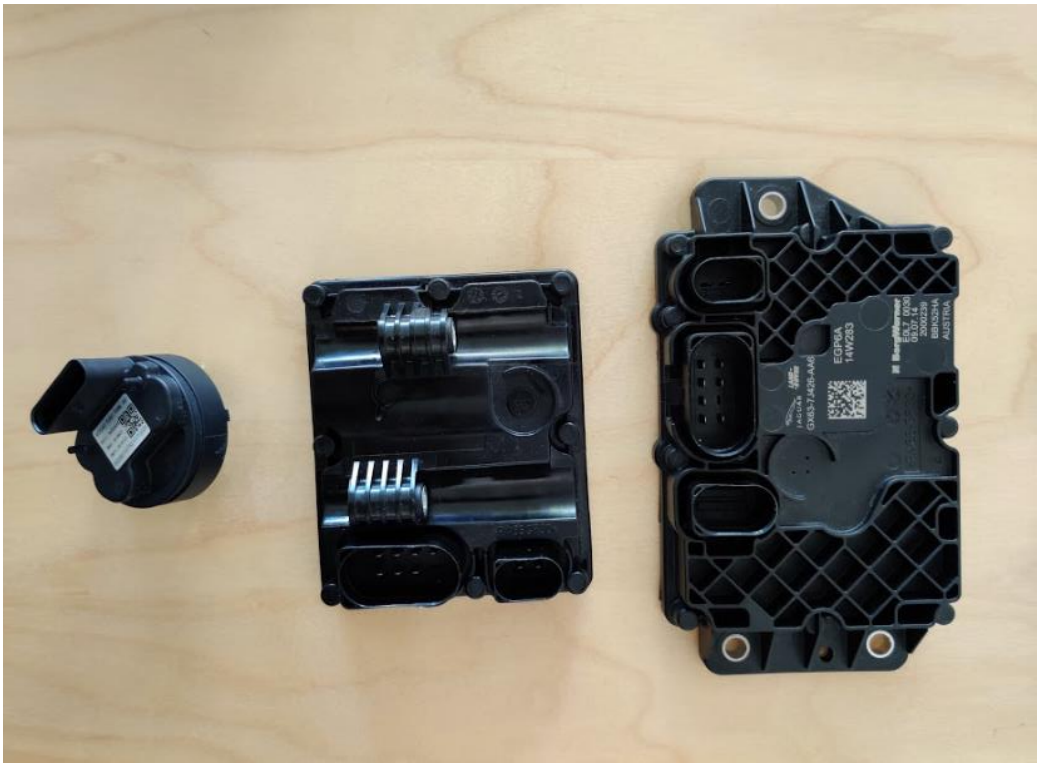

Figure 1-7: Gen VI (left), Gen V (middle), Gen V transfer case (right) ECUs

1.4 Simulink vehicle model

BorgWarner has developed a vehicle model in Simulink primarily for use in vehicle dynamics related work. The model emulates a real car when it comes to signals handled by the ECU. It has two kinds of inputs that get fed through the model which generates an output like signals sensors would feed to the ECU that controls the AWD in a real car. One type of input is the static one, which contains information about the car like its mass or engine size. The other type is dynamic with changing values during the simulation, like the steering wheel angle. The inputs are used in multiple subsystem and functions to calculate how a vehicle would behave given the same input. Based on engine characteristics, current gear and accelerator pedal position one subsection of the model could calculate the vehicle speed. Every subsection is kept simple which means that despite the entire model being complex, it is possible to gain knowledge regarding the overall system by studying the functionality of each subsection.

1.5 Problem formulation

Both the current testing and current evaluation are done with mostly static evaluation requirements. This means that the values that are sent to the ECU for evaluation are predetermined and so are the responses expected by the ECU. For example, a test case could require some vehicle parameters to be set to specific values. These parameters get sent to the ECU which generates a response that tells the AWD coupling how much torque that should be transferred. The next step in the test case would be to increase or decrease the specific values of the parameters and then repeat the entire procedure. The software in the ECU generates the

response to these parameters by interpolating them in a look-up table. In real vehicles it is rare that parameters have distinct levels like that and differences between testing and reality can cause errors to slip through and make it into the manufactured product. By using a simulated vehicle model, it is possible to get a better approximation of a real car in the test environment and thus improving the testing of software.

The simulated vehicle model that has been developed by BorgWarner for a wide range of uses and then further developed towards testing by Nils Espfors in his master's thesis[8] was done to combat the difference between reality and testing. One of the assignments of this thesis is to study the existing model and verify that it works as intended, which includes the special test cases that has been developed for it. There is no prior record of how the test cases perform. If it does not work as desired, then the assignment extends to improving the model further.

The vehicle model is controlled through a panel in CANoe and gives the user the same controls as if driving a car, like the three pedals, steering wheel, gear selector and gauges. The control panel can be seen in Appendix A. The vehicle model is dynamic, which means that the static inputs required for the current evaluation process cannot be set in the same manner. Setting a vehicle parameter to specific value will cause other parameters to change as well because they are all linked in the vehicle model. This makes it difficult to create distinct levels in order to evaluate the responses given by the ECU. In order to use the vehicle model in the testing procedure the evaluation method used needs to be investigated.

Linh Nguyen has done a thesis at BorgWarner, *Automating the Evaluation Process of Software Testing in Vehicles*[9], on an evaluation program that handles dynamic inputs like the ones generated by the vehicle model. Another assignment of this thesis is to investigate if it possible to continuously sample signals instead of setting specific values, and by evaluating using the automatic evaluation program instead of evaluating based on values from a look-up table.

To summarize assignments and questions that this master's thesis aims to address:

- Is the control panel of the vehicle model working as intended?
- Do the test cases developed for the vehicle model work?
- Is it possible to use static validation requirements with a dynamic model?
- Investigate if Linh Nguyen's automatic evaluation program can be used with the vehicle model.
- If yes on the question above, implement the evaluation program and test cases to use it with.
- In case of spare time in the end of the time frame, investigate the possibility of making the vehicle model more advanced and being able to handle other types of drive trains, like hybrid or electric.

1.6 Related work

The work of Nils Espfors and Linh Nguyen both cover topics that are relevant for this thesis. Their work were respectively done at the software testing department at BorgWarner in Landskrona. There has also been work done on the integration between Simulink and CANoe outside of BorgWarner[10]. There is no vehicle model being used in testing as of today, but the department for vehicle dynamics have a couple of different Simulink models for analysis.

1.7 Outline of the report

The report consists of 7 different chapters. The first chapter is an introduction to BorgWarner, their products and the software testing procedure. The first chapter also covers the main assignment. In Chapter 2 the background information required to understand the problem formulation from both a software and a hardware perspective is explained, and in Chapter 3 the required equipment and its function in the test process is clarified. Chapter 4 covers the verification of the current vehicle model and its testing software as well as the implementation of the automatic evaluation program. Chapter 5 explains the evaluation of the verification of the model and the results of the implementation of the evaluation program. It also includes the improvements made to both the old test cases and the improvement made to test cases using the evaluation program. Lastly, Chapter 6 contains the discussion and Chapter 7 is comprised of the conclusions and suggestions on future work.

2 Background

The goal of this chapter is to provide necessary background information regarding this project and why it was developed. The chapter is divided into two different subchapters, one covering the software background and the other covering hardware. In the software part the communication protocol used in automotive industry, the programming language used for this project, the software development tool, and the test execution environment are all explained. In the hardware subchapter the hardware used in this project as well how it relates to the standard testing equipment is covered.

2.1 CAN - Controller Area Network

In the 1980s Bosch developed the Controller Area Network (CAN) for communication in the growing automotive industry[11]. Still to this day CAN is one of the most used communication protocols in the automotive industry and it has even spread to other industrial areas like automation and mobile machines[12]. CAN communicates through two wires, CAN hi and CAN lo. The receiving node compares the two signals and the differential between the two determines if it is 0 or 1. In CAN, every node has permission to access the CAN-bus at any given time and each node has a controller and a transceiver. A CAN network topology can be seen in Figure 2-1. When data is transmitted in the bus it gets transmitted to every node in the network.

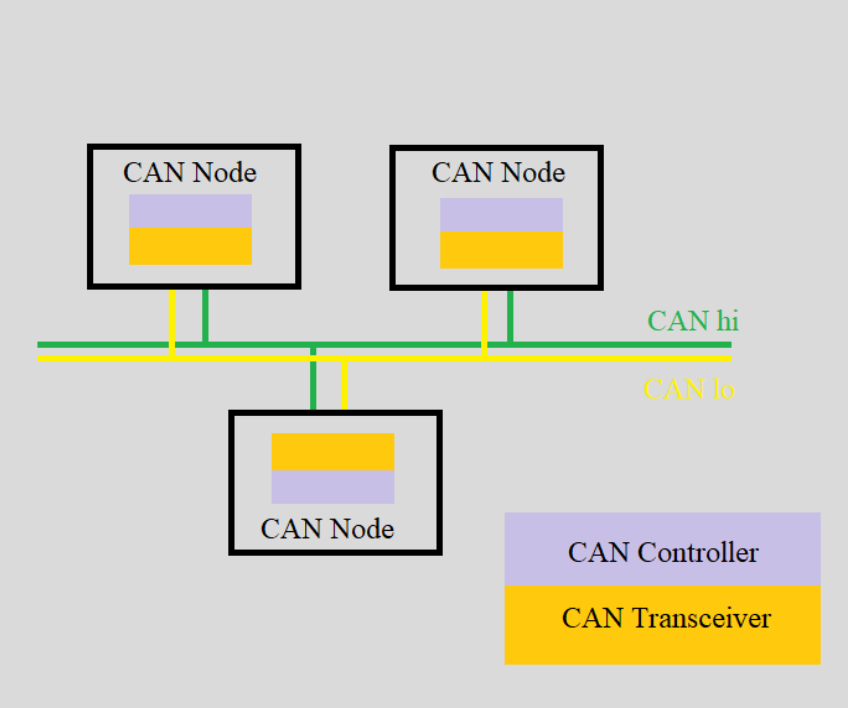

Figure 2-1: Example of CAN network topology.

By managing the communication through only two wires means that the overall requirement for cables in the vehicle can be reduced substantially. This is essential since modern vehicles can have up to 100 different nodes that communicate with each other. The nodes can be

everything from an ECU controlling the mirrors to the sensors measuring the engine speed. The reasons CAN is still used as the standard protocol for communication in vehicles almost four decades after its release are many. One of them is that if the bandwidth limit of one network is reached, automotive manufacturers can simply add another one. Since every node in the network receives all data transmitted, it makes sense to split networks depending on their purpose. Another reason that CAN is still used is that it has been thoroughly tested and can withstand the harsh operating conditions that are linked with automotive operations. On top of that is it a cheap and simple way of communication which further solidifies its position as the communication standard.

In recent years however, with increasing demands for communication the development of improved communication protocol has taken place. Many parts that used to be mechanically controlled are today controlled electronically instead. This, in combination with more systems like driver aid and systems that improve efficiency and emissions being developed means that the bandwidth limit of CAN will be breached. In 2012 an updated version of CAN with flexible data, CAN FD, was released by Bosch to address the bandwidth issue[13]. Another communication protocol worth mentioning is FlexRay. Developed in the 2000s by a consortium of automotive OEMs (original equipment manufacturers) and chip producers[14]. It has a data rate of 10 Mbit/s, which is ten times higher than that of CAN.

2.2 CANoe

CANoe or CAN open environment is a test development tool developed by Vector Informatik GmbH. CANoe is widely used in the industries that use CAN communication for testing purposes. The software and the associated required hardware are both provided by Vector. CANoe makes it possible to develop tests, analyse data and simulate individual ECU:s or even entire networks of ECU:s. On top of being able handle CAN communication, CANoe also supports the use of other communication protocols like FlexRay, LIN, and Ethernet.

The panel that controls the simulated vehicle model is developed and used in CANoe. It has the same controls as normal vehicle, like steering wheel, accelerator pedal, brake pedal and clutch pedal. These can be seen in Figure 2-2. It also has a gear selector and an automatic gearbox setting with different characteristics. The options to set a certain vehicle speed, a certain engine RPM or a certain engine torque also exists. There is also the possibility to control some vehicle parameters like weight and engine size.

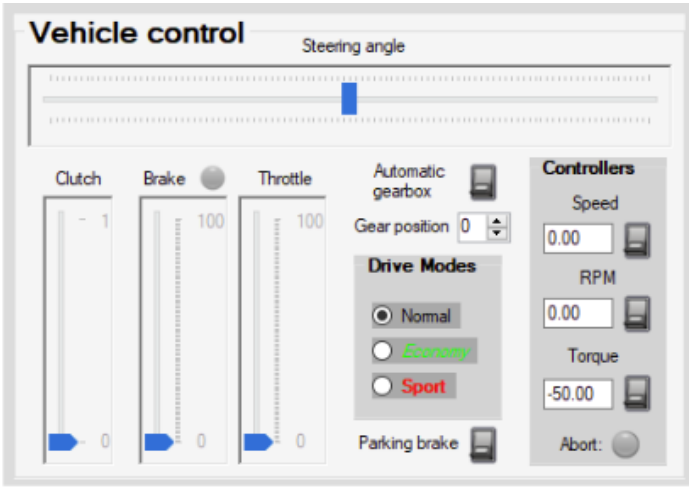

Figure 2-2: Control panel to control vehicle model in CANoe.

As well has having the driver controls, the panel also includes indicators for several of the output signals from the model like engine RPM and vehicle speed. These can be seen in Figure 2-3.

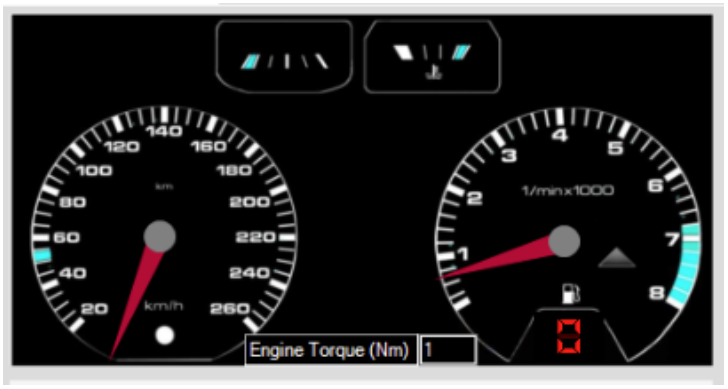

Figure 2-3: Gauges showing speed and RPM, current engine torque and current gear.

There is also a panel that allows the change of the road friction coefficient for either all four wheels or specific wheels to simulate a sudden change in grip. This can be seen in Figure 2-4.

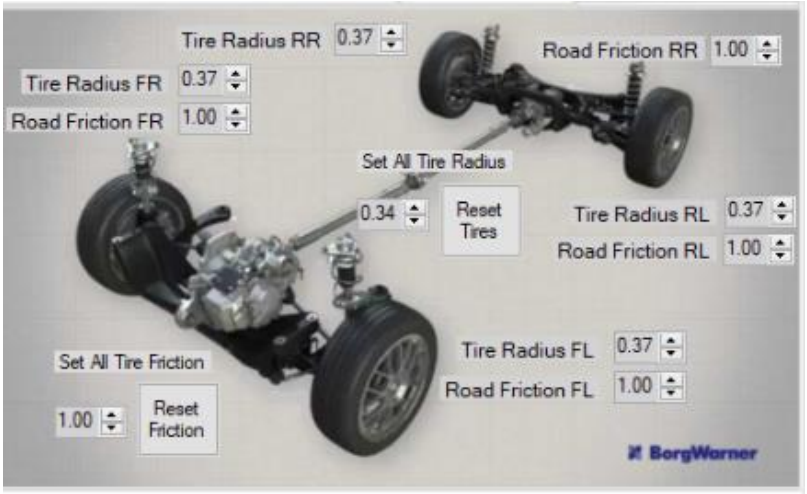

Figure 2-4: Panel that can change road friction coefficients.

2.2.1 CAPL

CAPL, or CAN Access Programming Language, is the programming language used to write code for CANoe. CAPL is based on the C programming language, but it also shares similarities with Java and Python [15]. CAPL is an event driven programming language which means that the applications written in CAPL can respond to different events in the system, like messages on the CAN bus.

2.3 vTestStudio and CAPL Browser

vTestStudio is an environment for test development using CAPL for CANoe configurations. The CAPL Browser is an older version of vTestStudio that fills the same functionality. They both compile the CAPL code before CANoe can use it. vTestStudio is however better than CAPL Browser at displaying and managing the test sequence order. The test structure in vTestStudio is contained by a tree structure in a.vtt file. The .vtt file contains all the test cases active in the test suite, and which functions they use. The foundation for this thesis is test cases specially developed to fit the vehicle model that originates from the CSW test suite.

2.4 MATLAB and Simulink

MATLAB is a software program and programming language first released in the 1980s[16]. It is developed and sold by Mathworks and is used by many companies and universities. MATLAB, which is an abbreviation of matrix laboratory, is a computing environment that for instance allows calculation on matrices, plotting of functions and data and interacting with programs written in other programming languages. One of the programs that MATLAB can interact with is CANoe.
MATLAB has many purpose built add-on programs. One of these programs is Simulink, which is a graphical programming environment used for modelling and simulating dynamical systems. Simulink shares the ability of interacting with programs written in other programming languages, like CANoe, which means that the vehicle model that is built in Simulink can be used for software testing.

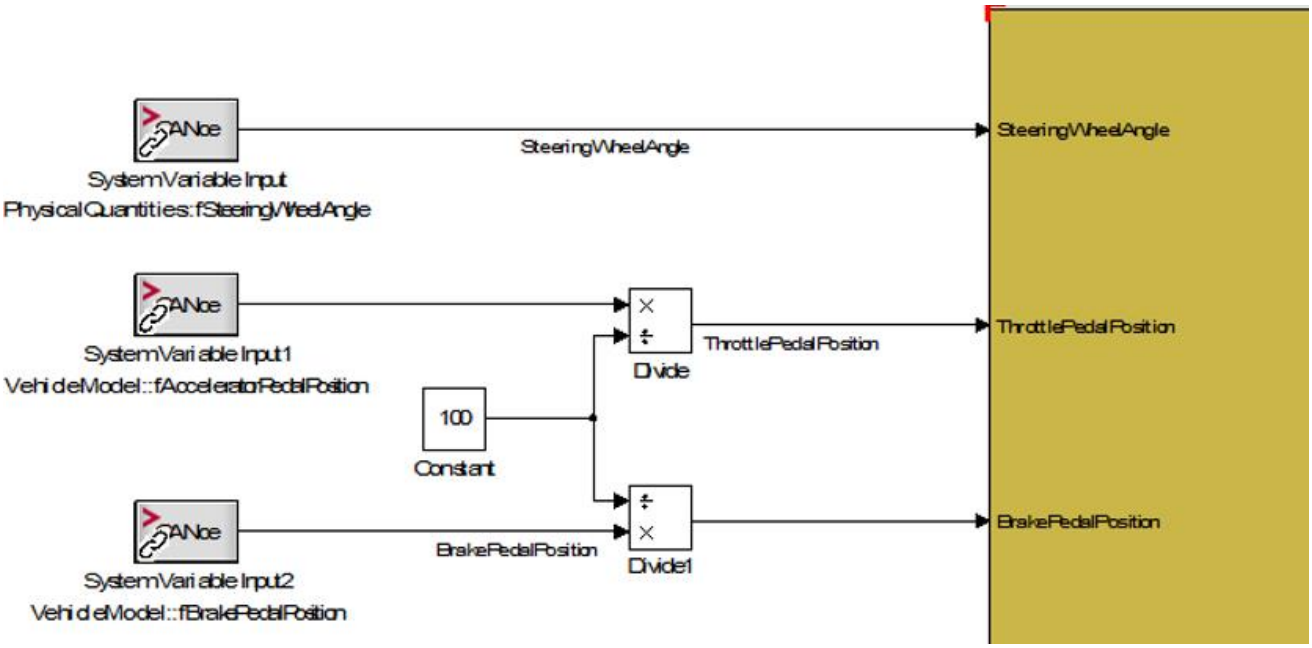

Figure 2-5: Simulink vehicle model. Inputs to the model located on the left.

Figure 2-5 showcases a part of the vehicle model used in this thesis work. In Simulink it is possible for the model to be built in multiple layers which allows the creation of complex system while maintaining an overview of the entire system. This specific model takes input generated by the control panel which controls the vehicle and sends them through all the subsystems of the model to generate an output that gets passed to the ECU.

The model used in this thesis began development in 2014 and has since been used in multiple different projects and by different departments. It is a model of the "FWD with AWD capabilities" type mentioned in Chapter 1.1.1 and it functions by taking two kinds of inputs, static and dynamic, and running it through the subsystems to generate the correct outputs. The static inputs are composed of different vehicle parameters like vehicle mass and size and of different driving conditions like friction coefficients. The dynamic inputs consist of different driver inputs from the CANoe control panel, like steering wheel angle and accelerator pedal position. The model takes these inputs and run them through the different blocks and subsystems to calculate the outputs that normally would be generated by the different sensors in a real vehicle. These outputs are then processed by CANoe and lastly sent to the ECU where the software decides if the AWD should be engaged or not. The Simulink model is running in real time and should not change the outcome of any tests. However, when CANoe is doing highly computationally demanding tasks like graphing, the rate which CANoe samples data from the vehicle model might decrease from which can cause errors. Normally data from the model is updated every 10ms, but during heavy loads that can increase to every 100ms instead.

Most of the signals the ECU receives are sent every 10-20ms. If the simulated model cannot supply CANoe with updated signals fast enough old data will be transmitted to the ECU. It is hard to tell how this will affect the software and the outcome of any tests since not all signals generated by sensors in a real car update at the same frequency. However, CANoe has a built-in warning system for when the real time requirements are no longer fulfilled. It is unknown what the limit for how slow the system can be is. The ECU can also detect when real time requirements are no longer fulfilled and act accordingly.

2.5 Volvo generation VI software for scalable product architecture platform

In order to avoid introducing unnecessary errors to this project a previously tested and verified software was used. By doing that it is possible to conclude that failing test cases depends on the test environment and not the software.

A software for the Volvo platform SPA (scalable product architecture) was used for the entire project. The SPA platform was developed by Volvo for their larger model cars like the XC90 and V90.

2.6 Automatic Evaluation Program

The automatic evaluation program designed by Linh Nguyen is a set of functions that can be used to determine trends from sampled signals. It was designed to automate the analysis of driving logs. Some of the functions used to evaluate the results of driving logs was used in this thesis.

2.7 Test Rig

In the software testing environment, the ECU is not fitted to a car. It is instead connected to a test rig supplied by Vector Informatik GmbH. Depending on the test requirements the rigs come in different sizes with different functionality. One of the smaller versions available is the VN8970 which is based on an embedded Windows 7 system. The VN-system lack some of the functionality required to induce errors related to hardware, like drawing too much current or short-circuiting different components. Because of this the VN-system is suitable for software diagnostics. A picture of a VN-system rig can be seen in Figure 2-6.

Figure 2-6:VN8790 System with power supply.

The other system used, which is larger and more complex, is the VT-system. The VT-system has an on-board operating system based on Windows 7 that enables the user to access and investigate configuration errors related to CANoe. VT-systems has a modular setup which makes it possible to adapt them based on demands. More powerful power supplies and specialised units for inducing hardware errors makes the VT-system suitable for testing hardware related errors. A picture of a VT-system rig can be seen in Figure 2-7.

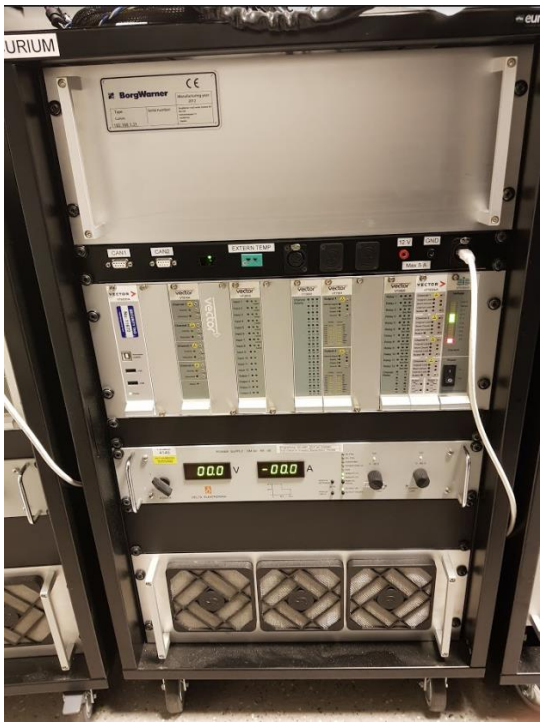

Figure 2-7: Modular VT-system used in testing.

The ECU used in testing is connected to the rig through CAN. This applies to both VN and VT rigs. The rig then sends information to the ECU like a normal vehicle would, emulating the different nodes that could be found on a CAN network in a car. The computer that runs CANoe can be connected through either USB or ethernet to the rig which allow real-time data sampling of both the simulated signals from the rig and how the ECU handles them. How the different components communicate with each other can be seen in Figure 2-8.

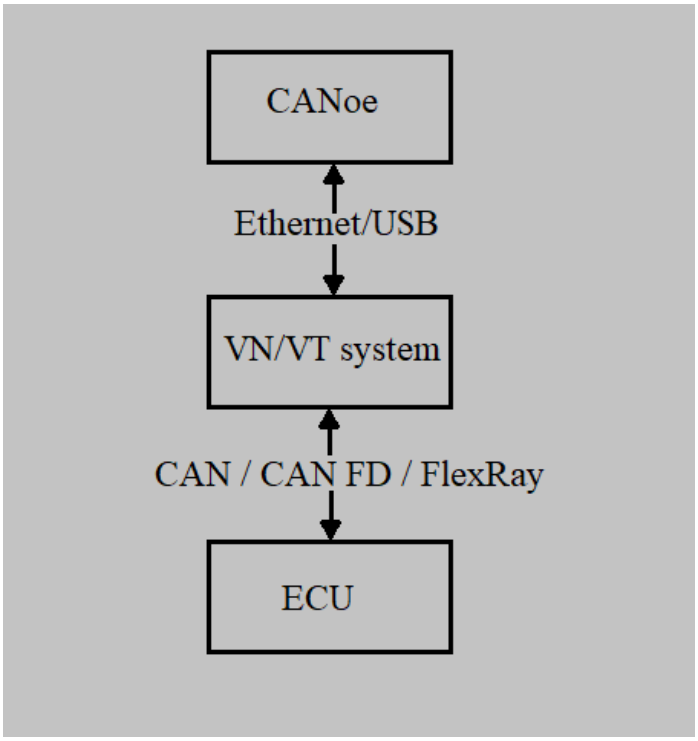

Figure 2-8: Schematic figure of how the components in the test setup communicate.

3 Equipment

The equipment used in this project was a Gen VI ECU with previously verified Volvo software connected to a VN8970-system rig through CAN. The ECU was attached to a hydraulic actuator just like it would be in a real car. Instead of the actuator controlling an AWD coupling however, it was connected to a pump block with oil so it could build pressure similarly to how it would in the real product. The pump block and actuator give the tester additional feedback on top of the signals sampled by CANoe in form of audible noise when running.

The power supply is connected to the VN-system through an analogue I/O-board which also feeds the ECU with power. This can be seen in Figure 3-1. The computer that runs CANoe and thus controls what tests are running is connected to the VN box through ethernet, which also supplies the software license required.

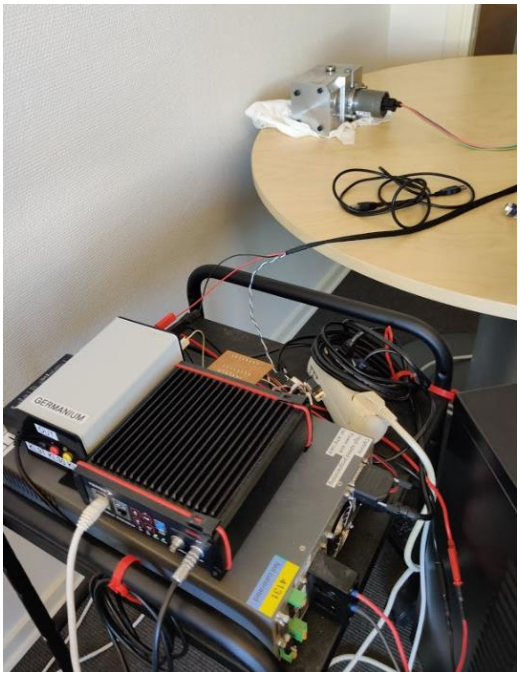

Figure 3-1: Test setup used in this project. VN-System with power supply and analogue input/output board at the bottom. Actuator and pump block on the top.

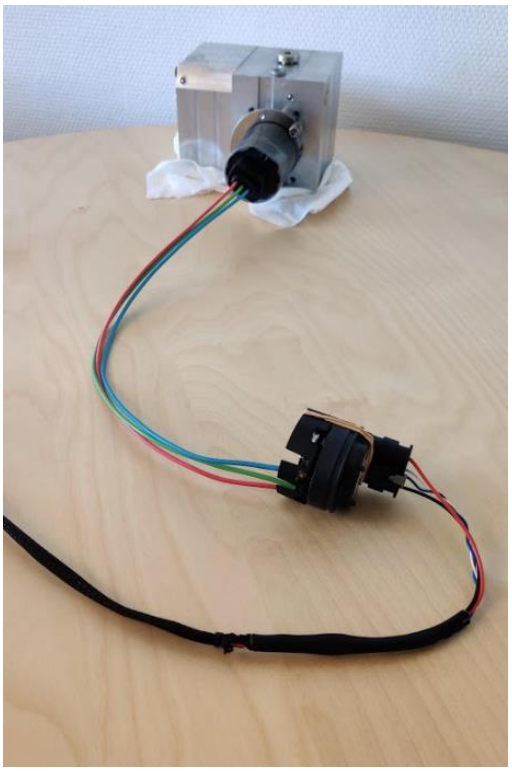

Figure 3-2: ECU connected to actuator and pump block.

In order to make it easier to swap ECUs without having to change the entire actuator and pump block the ECU is not physically attached to the actuator like it would be in a real car, instead there are wires connecting the two. This can be seen in Figure 3-2.

This setup resembles the setup used in the normal test procedure at BorgWarner and thus makes it easier to integrate any discoveries done through this project.

4 Software Verification

This section will be split into two subsections to cover the main assignments of this project. One subsection will cover the verification and improvement of already developed test cases for the vehicle model and the other subsubsection will cover the implementation of the dynamic evaluation program.

4.1 Implementation of the vehicle model

For the vehicle model to work together with CANoe it had to be imported and adjusted to fit the project specific software. Once CANoe was configured to allow Simulink models, the vehicle model was imported as a node in the emulated CAN network. After importing the vehicle model, the associated system variables were also imported and updated to fit the Volvo software that was used. In order to use the vehicle model for testing a test suite had to be linked to CANoe through vTestStudio. The test suite was updated with the correct project specific data bases.

4.1.1 Verification of the vehicle model

The vehicle model has a specially developed test suite that consists of converted CSW tests and integrity tests that check if the inputs from the control panel match up with the variables read by the ECU. The integrity test cases were run to check if the implementation of the vehicle model had been done correctly.

4.1.2 Updating functions

Some of the functions used in the different test cases in the suite were updated. The function that sets the speed of the vehicle was updated with a loop counter to avoid creating an infinite loop if the vehicle was unable to reach the desired speed for any reason. The function was also given a Boolean return value based on if the requested speed was achieved or not. All the test cases using the set speed function were updated to accommodate the change.

The function that sets a specific engine torque was updated to allow values both above and under the specified torque to be considered a pass if they are within the given threshold.

4.2 Implementation of automatic evaluation program

The automatic evaluation program was implemented into the project in vTestStudio. In order to use the program, the test cases had to be modified. The functions require arrays of data as input and to sample the signals a timer was implemented that ran parallel during the test sequences and saved the specific signal at a certain interval into an array. This array could then be passed to the appropriate evaluation function. A test case that checks if the requested torque to the AWD coupling goes down as speed increases was implemented. The torque was sampled with the timer as the vehicle model was accelerating and then passed through an evaluation function that checks for strictly decreasing values in the array.

5 Evaluation of software implementation

This chapter will cover the evaluation of the verification of the model and the results of the implementation of the evaluation program. It will also cover the improvements made to both the old test cases and the improvement made to test cases using the evaluation program as well as the results of said improvements. It will also answer some of the questions asked in the main assignment, see Chapter 1.5.

5.1 Results of verification and implementation

The integrity tests show whether the different controls in the control panel are linked to the correct parameter read by the ECU. In the table below it is possible to see that all the tests except one passed. The failing test failed because one of the features it tests, longitudinal acceleration, was not implemented in the tested software. The results can be seen in Table 5-1 and they show that the implementation of the vehicle model was successful and that the project specific parameters were linked properly.

Test	Result
Accelerator pedal	Pass
Brake pedal	Pass
Gear position	Pass
Clutch pedal	Pass
Steering angle	Pass
Parking brake	Pass
Engine speed	Pass
Engine torque	Pass
Speed and velocity gauges	Pass
Longitudinal acc, lateral acc, yaw rate	Not implemented

Table 5-1: Results of integrity tests.

Some of the converted CSW test cases did not pass. One of three failing test cases failed because a simpler VN rig was used which could not measure the hardware related parameters related to that test. The purpose of the test is to make sure the pump that controls the coupling starts and stops with the engine. This is done by measuring the current in the pump, which the rig used could not do. It is unknown why the other two test cases failed and it was decided that focus and time should be spent on answering the other questions of the main assignment and on implementing new test cases instead.

In Figure 5-1 it is possible to see all the test cases that were run. Some of the initial ones do not leave a verdict since they have no evaluation step and are only there in order to initialize the test. The integrity tests were run separately and their results are covered above. Some of

the test cases were not fully implemented and hence not executed.

Test Results

1	724_132_VehicleModel	fail
1.1	Init	3
1.1.1	CSWInit ()	none
1.1.2	CSWSetParameterSet (iParameterSetToUse=0)	none
1.1.3	tcInitVehParams (iParameterSetToUse=0)	none
	Vehicle Model Integrity	not executed
1.2	CSW Drive with torque reduction	1
1.2.1	tcCSWDriveTorqueReductionWithVelocityVehicleModel (settleTime=5)	pass
1.3	CSW Drive Increase with Driver Req	1
1.3.1	tcCSWDriveTorqueIncreaseWithDriverRequestVehicleModel ()	pass
1.4	CSW Prelock Torque Accelerator Pedal	1
1.4.1	tcCSWPrelockAcceleratorPedalVehicleModel ()	fail
	CSW Prelock Min Base Steering Angle Reduction	not executed
	CSW AYC Dist during power on	not executed
1.5	CSW Brake Torque Distribution	1
1.5.1	tcCSWBrakeTorqueDistVehicleModel ()	fail
1.6	CSW Pump Start Stop Running	1
1.6.1	tcCSWPumpStartStopVehicleModel ()	fail
	CSW Safe Driving	not executed
1.7	CSW Prelock Min Base Torque Velocity	1
1.7.1	tcCSWPrelockMinBaseTorqueVelocityVehicleModel ()	pass
	Debug/Testing	not executed

Figure 5-1: Auto-generated test report from CANoe.

The implementation of the automatic evaluation program worked in the sense that it was possible to sample data and feed it to the evaluation function. However, the function could not detect any decreasing trend in the requested torque and thus the test case failed.

5.2 Main assignment

In order to understand the improvements done and the newly developed test cases it is necessary to answer the questions of the main assignment first. Each question will be given an answer and an explanation.

- **Are the controllers of the vehicle model working as intended?**

Yes, the integrity tests show that different controls available through the control panel do work as intended.

- **Do the test cases developed for the vehicle model work?**

Some of the test cases work as they should. The cases that do pass use the functions of the vehicle model and gives the same results as the CSW tests which they are based on.

- **Is it possible to use static validation requirements with a dynamic model?**

Yes, it is possible as the already implemented test cases use static requirements to evaluate test results. However, using the vehicle model on CSW tests with static requirements is just a more complex way to accomplish the same thing as normal CSW testing. To get full value of the vehicle model and how it represents a real vehicle the test cases developed for it needs to be customised.

Instead of making the dynamic vehicle model static, the test requirements should be made dynamic. This can be achieved by not using the functions designed for the model that set parameters to specific values and instead using the controls in the control panel. Using the control panel would better mimic how a vehicle is driven in the real world and thus decrease the gap between a simulated test environment and reality.

- **Investigate if the automatic evaluation program can be used with the vehicle model.**

Yes and no. The evaluation program uses dynamic requirements to determine the outcome of a test, while the current vehicle model test suite use static requirements. However, it is possible to sample data from the vehicle model and use the program on that data. This means that if test cases were specially made to be evaluated with dynamic requirements it is possible that the automatic evaluation program would work together with the vehicle model.

- **If yes on the question above, implement the evaluation program and test cases to use it with.**

Special test cases with dynamic requirements were designed and implemented, see Section 5.3.

- **In case of spare time in the end of the time frame, investigate the possibility of making the vehicle model more advanced and being able to handle other types of drive lines, like hybrid or electric.**

At the end of the time limit given for this project there was not enough time to answer this question and it will be further discussed in the chapter for future work.

5.3 Implementing test cases with dynamic requirements

Three different test cases were implemented that use dynamic requirements instead of static. The first case checks the correlation between how much torque is transferred through the coupling and the pressure set point as the car engine torque ramps up and then down again. The pressure set point is the pressure set on the lamellas that close the coupling and thus enables torque to be transferred. The engine torque was changed by ramping the accelerator pedal from 0% to 100%, and then back to 0 again. As the engine torque was ramping a timer was set to sample both transferred torque and the pressure set point. The samples were then run through an evaluation functions that check for correlation. The function returns a value

between -1 and 1, and a value greater than 0.8 indicates strong correlation. A value less than -0.8 indicates a strong negative correlation. The correlation value returned by the evaluation function for this test was 0,997 which means the test passed. The three spikes in the red and green graphs in Figure 5-2 are different gears being used.

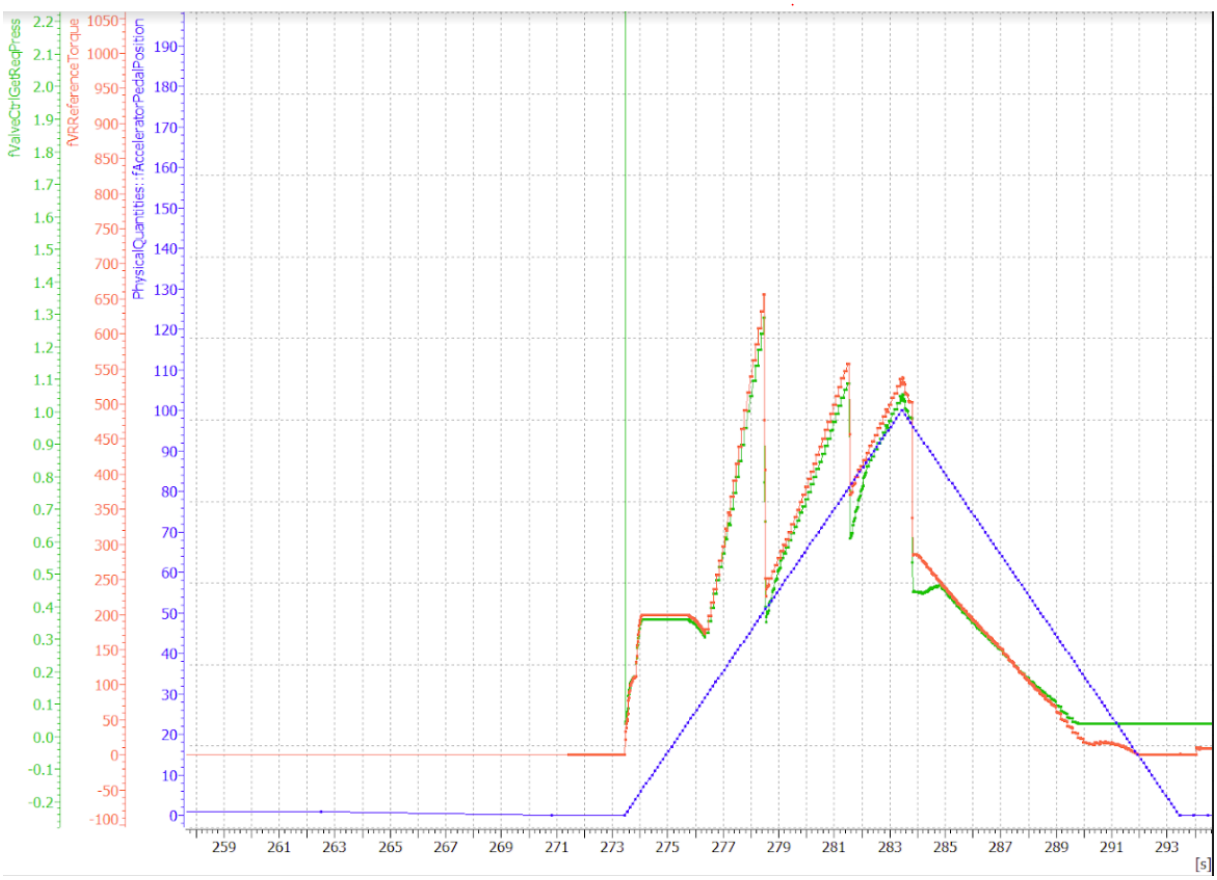

Figure 5-2: Pressure set point (green) in Pa, transferred torque(red) in Nm and accelerator position(blue) in %, as a function of time in seconds. The green spike at the start of the simulation depends on that the default value of the pressure set point is high.

The second test case implemented that use dynamic requirements was the test implemented in Chapter 4.2. The test checks that the transferred torque is reduced with velocity. If the vehicle travels in a straight line at high speed it does not need AWD and by decoupling the mechanical losses can be reduced, thus improving fuel consumption. The parameters and tuning used in this software meant that the transferred torque would not reduce until the vehicle reached a velocity of 35 m/s. This means that the vehicle had to accelerate to 35 m/s before the sampling of the transferred torque signal could begin. Once the vehicle reached sufficient speed the test could begin. The vehicle continued to increase in speed and the transferred torque signal was sampled. The sample was then sent to a function that checks that each element in the array is smaller than the previous one. The function returns true if the values in the array are strictly decreasing. The evaluation function returned false and the test failed. The graphs of the sampled signals and the position of the accelerator pedal as a function of time can be seen in Figure 5-3.

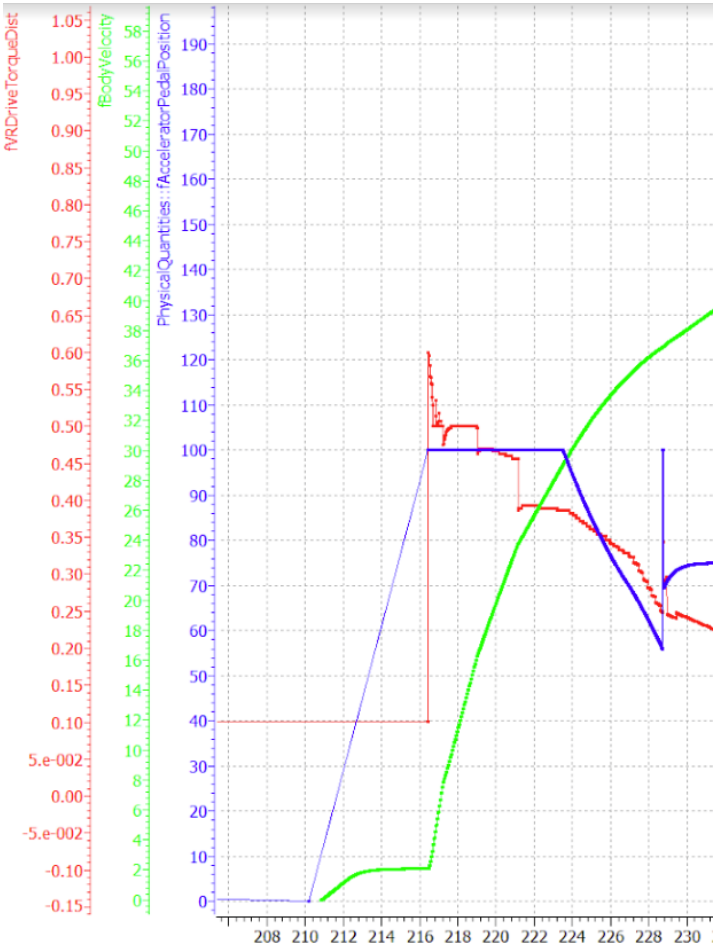

Figure 5-3: Transferred torque(red) in Nm, velocity(green) in m/s and accelerator position(blue) in % as a function of time in seconds.

The third and last test case that was implemented checks that the transferred torque is reduced with increasing steering angle at low speeds. A good example of this driving scenario would be if the vehicle needs to turn into a tight parking space. With AWD enabled the turning radius increases, which makes it harder to park the car. As the steering wheel was turned 3π radians in one direction, then back to 0, and then 3π radians in the other direction the transferred torque was sampled in the same manner as previous test cases. The sampled array was then sent to the same evaluation function as the previous test case. The function returned false and the test case failed. The graph of the sampled data and the steering wheel angle as a function of time can be seen in Figure 5-4.

Figure 5-4:Torque distribution(red) and steering wheel angle(blue) in radians as a function of time in seconds.

5.3.1 Updating evaluation functions

Signals generated by the simulated vehicle may not update as often as they do in a real car. This could be a cause of CANoe not polling data fast enough from Simulink. There were no warnings from CANoe regarding real time requirements not being satisfied however. Compared to the driving logs generated in a real car, on which the automated evaluation program was designed upon, the signals in the simulated environment had plateaus. This can easily be seen in Figure 5-4. In order to use the dynamic evaluation functions, they were updated to allow plateaus instead of strictly increasing/decreasing values. When the two test cases that failed were run again with the updated evaluation function, they both passed.

6 Discussion

There are several benefits to using a simulated vehicle model for software testing purposes. The main one is that it creates an environment where the testing represent the real product as close as possible while still maintaining the advantages of using simulation. By using simulation instead of a real car, it makes testing cheaper and more versatile as well as allowing for the ability to repeat tests in order to replicate errors. When using a simulation, it is for example possible to imitate a change of temperature from -30 to +50 degrees Celsius instantly.

It is also advantageous to make sure that the simulated vehicle is used in such a manner that is emulates a real driving situation as good as possible. An example of this can be seen in Figure 5-2 where it is possible to see the gear shifting. If that test would be conducted with static parameters instead, the test would select a gear and stick to it which does not represent how a real vehicle would be driven. It is very rare that a modern car is driven for a long period of time in a single gear, especially at lower speeds. In order to get a good similarity between reality and simulation it is important to take into consideration what controls a real driver would have available and how a real car would be driven.

The test cases that were already implemented before this project shows that it is possible to create test cases that use static requirements for evaluation with the help of functions that can give specific parameters static values. Those functions have access to the same controls as a driver would, but with advanced regulators and high updating frequencies they can achieve values that no human driver could.

However, there are risks involved with solely using simulated vehicles. Unless the simulated version of the car represents the real one to 100% there is a chance that some errors will not be caught which can lead to accidents. By analysing both the driving logs from the real car and the simulated tests with the automatic evaluation program it is possible to compare the two and thus getting a better understanding on how the two relate. It is important to remember that the purpose of the vehicle model to improve testing of the software and if the model introduces more error sources it might not be worth using at all.

Since the ECU runs in real time it is essential that the vehicle model is working fast enough to satisfy the requirement to keep it a real time system. The issue is that if the vehicle model generates new outputs slower than CANoe sends them to the ECU but fast enough to not trigger any warnings it would be very difficult to for the ECU to detect any errors. This can cause test cases to pass despite being faulty because the test environment is not precise enough. Since no warnings regarding the real time requirements were ever generated during testing of the test cases developed for this thesis is it hard to tell if the execution time of the model had any impact on the outcome of the tests.

In order to use the vehicle model for extensive testing the entire test specification and all the relating test cases must be rewritten to accommodate dynamic requirements. This would of course be very time consuming and as of now nothing is planned. If enough time and work went into it however, it is possible that the vehicle model could be used for software testing in the future, as the test cases implemented in this thesis are proof of concept.

7 Conclusions and future work

This work has provided the software-verification department (TVR-SW) at BorgWarner PDS with a deeper knowledge of how the Simulink based vehicle model can be used for testing and how it can be improved. It has also connected the work of two others master's theses and brought them both closer to a point where they can be used in the daily testing procedure. There is still some work to do in order fully utilize the vehicle model and its test cases compared to its current state.

7.1 Future work

As mentioned earlier, in order to fully use the vehicle model for test purposes it needs some further work on the test cases that uses the model. First and foremost, the type of tests that would benefit from using the model needs to be identified. It is possible to squeeze the vehicle model into most test cases, but if it does not make the tests more accurate nor make them easier to implement there is really no point. After identifying suitable test cases, or finding scenarios where new cases can be created, a test specification for those cases needs to be either re-written for old cases or created for new ones. Once both suitable tests and specifications for those tests has been created, the test case itself needs to be developed and implemented into the specific project in order to use the model.

The question of simulation performance needs to be further investigated as well. The cause and impact of the signal plateaus seen in some test cases needs answering before the model can be used in testing. If the plateaus are a result of the simulation execution time being slow and it is deemed that it could affect the outcome of test cases, then a way of increasing the performance of the model must be developed.

7.1.1 Different vehicle configurations and drivetrains

Only one of four mentioned vehicle configurations were used in this thesis, the front wheel drive with AWD capabilities. This means that it would be possible to use other vehicle models with the three remaining configurations to broaden the use of simulated vehicle models in testing. The four vehicle configurations mentioned in this thesis are not the only products being made. There are different hybrid and fully electric products as well that could benefit from having a simulated vehicle for testing, just like the one investigated in this project. This would require extensive work on creating a simulated model however.

7.1.2 3D Graphics

Another aspect that could be implemented to further improve the vehicle model and how it is used could be to have some sort of 3D graphic interface. This could improve the testing process in two ways. The first way would be to give the tester an additional form of feedback on top of the logs and test reports, that are already being created, through being able to observe how the vehicle reacts in different scenarios. The other way this could improve testing would be through further closing the gap between driving in a real car and simulated testing. With a 3D visualisation it would be possible for the test engineer to drive the simulated vehicle through the control panel in the same manner as the real car and use the same test procedure, which would make it possible to directly compare the two.

8 Bibliography

[1]: BorgWarner ”Company” [Online]
Available: https://www.borgwarner.com/company [Accessed 5 April 2020]

[2]: Borgwaner ”AWD Couplings” [Online]
Available: https://www.borgwarner.com/technologies/awd-cross-axle-systems [Accessed 20 May 2020]

[3]: Bild på RWD med AWD (BorgWarner, “BorgWarner Official Presentation,” Landskrona, Sweden, 2017.)

[4]: Bild på RWD med transfer case (BorgWarner, “BorgWarner Official Presentation,” Landskrona, Sweden, 2017.)

[5]: Bild på FXD (BorgWarner, “BorgWarner Official Presentation,” Landskrona, Sweden, 2017.)

[6]: IceMakers, ”Our history” [Online]
Available: https://www.icemakers.se/en/about-us/our-history/ [Accessed 11 May 2020]

[7]: Vector ”CANoe”[Online]
Available: https://www.vector.com/int/en/products/products-a-z/software/canoe/ [Accessed 5 May 2020]

[8]: Espfors. N. “*CANoe – Simulink Integration of Vehicle Model in Existing Test Environment*”, Master’s thesis, Lund University. Lund.
Available: http://lup.lub.lu.se/student-papers/record/8964000 [Accessed 9 April 2020]

[9]: Internal BorgWarner report. Written by Nguyen. L. To be published under the name “*Automating the Evaluation Process of Software Testing in Vehicles”.*

[10]: J. W. Y. W. C. L. Linlin Yao, “Research on vehicle integrated control algorithm based on MATLAB and CANoe co-simulation,” in IEEE Conference and Expo Transportation Electrification Asia-Pacific, Beijing, China, 2014.

[11]: “Quora,” 23 September 2017. [Online].
Available: https://www.quora.com/What-is-a-CAN-bus. [Accessed 4 May 2020].

[12]: Kvaser, “About CAN”. [Online].
Available: https://www.kvaser.com/about-can/ [Accessed 30 May 2020]

[13]: Vector Informatik GmBH, "Learning Module CAN" [Online].
Available: https://elearning.vector.com/mod/page/view.php?id=333 [Accessed 4 May 2020]

[14]: Vector Informatik GmBH, “FlexRay Consortium” [Online].
Available: https://elearning.vector.com/mod/page/view.php?id=378 [Accessed 15 May 2020]

[15]: Vector Informatik GmBH “CAPL Documentation” [Online].
Available: https://kb.vector.com/entry/48/ [Accessed 27 April 2020]

[16]: Mathworks “A Brief History of MATLAB” [Online].
Available: https://se.mathworks.com/company/newsletters/articles/a-brief-history-of-matlab.html [Accessed 9 May 2020]

Appendix A

Full Control Panel