Hybrid Vehicle Supervisory Controller Development Process to Minimize Emissions and Fuel Consumption in EcoCAR 2

Trevor Crain

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Mechanical Engineering

University of Washington
2014

Committee:
Brian C. Fabien – Chair
Per G. Reinhall
R. Bruce Darling

Program Authorized to Offer Degree:
Department of Mechanical Engineering

University of Washington

Abstract

Hybrid Vehicle Supervisory Controller Development Process
to Minimize Emissions and Fuel Consumption in EcoCAR 2

Trevor Crain

Chair of the Supervisory Committee:
Professor Brian C. Fabien
Department of Mechanical Engineering

This thesis presents the design process used to create and validate a hybrid vehicle supervisory control system for a plug-in hybrid electric vehicle in the EcoCAR 2 competition. The vehicle utilized a Parallel through the Road hybrid architecture with a B20 biodiesel engine-powered front axle and electric motor-driven rear axle. The primary goal of this work is to present a selection of the processes used by the controls team throughout the competition to define the control system requirements and platforms, model the components of the vehicle and validate each model, and estimate the effects of various control system strategies and parameters on overall vehicle performance. The advantages of using a version-controlled Simulink model for supervisory controller development are discussed along with an explanation of the software architecture and primary hybrid control modes. The models developed to simulate the primary drivetrain components are detailed in addition to their parameterization and validation testing methods. The final chapter presents an analysis on dynamometer test data used to quantify the effects of various controller parameters and strategies on the results of the Emissions and Energy Consumption (E&EC) event’s dynamic drive testing. The electrical energy consumption of the vehicle during charge depleting mode testing is used to select a state of charge value for transitioning to charge sustaining mode. Various implementations of engine stop start are also analyzed, showing the potential for a 3.8% improvement in B20 fuel consumption over the course of the City/Highway E&EC drive cycle.

Table of Contents

List of Figures

List of Tables

Acknowledgments

This work was made possible by an enormous collective effort from the students on the UW EcoCAR 2 team, the University of Washington's Mechanical Engineering department, and the competition sponsors and organizers. All of the competition sponsors have contributed incredible amounts of support, funding, and inspiration over the years, but special thanks must be made to the US Department of Energy and General Motors in addition to the Argonne National Labs Executive Steering Committee for their tireless work.

Thank you as well to the team's excellent faculty advisors. Professor Brian Fabien and Per Reinhall have been avid supporters of the program and the impact it has had on our students since its inception, and have worked endlessly behind the scenes to ensure our team has the departmental support, managerial oversight, and technical knowledge we have needed to succeed. Professor Bruce Darling has helped the team make huge steps in the past year towards being a vital part of the Electrical Engineering department, and all of their contributions in the upcoming EcoCAR 3 competition will be invaluable.

To the team itself, thank you for your countless hours of support and immense drive to learn and excel in both school and the competition. This thesis represents much more than one individual's research. It would not have been possible without the tireless efforts of dozens of students and faculty members that worked to make the UW vehicle a success, and gain as much knowledge as possible in the process.

Chapter 1 - Introduction and Vehicle Architecture Description

This thesis presents the design process used to create and validate a hybrid vehicle supervisory control system for a plug-in hybrid electric vehicle (PHEV) in the EcoCAR 2 competition. The EcoCAR 2 competition challenged 15 universities across North America to reduce the environmental impact of a 2013 Chevrolet Malibu without compromising consumer acceptability. The project took place over a three year design cycle, where teams select a hybrid architecture in year one, implement the hybrid components and control system on the vehicle in year 2, and refine the control system operation and implementation in year 3. The primary goal of this thesis is to present a selection of the processes used by the controls team to define the control system requirements and platforms, model the components of the vehicle and validate each model, and estimate the effects of various control system strategies and parameters on overall vehicle performance in the various competition events.

The first task in constructing a HSC program to coordinate all of the torque sources is to determine what configuration of hybrid vehicle will be used. Hybrid technology generally involves adding one or more high voltage electric motors to a vehicle along with an electric energy storage system (ESS). The motors may be integrated into the drivetrain in a variety of ways. In some situations, a motor is linked directly to the front or rear axle with a gearbox and is the sole torque source for that axle. In other cases, the motor works in conjunction with an internal combustion engine (ICE) to provide a combined torque to an axle. Some vehicles may even have an ICE and motor combination that are completely decoupled from the wheels, so that the ICE is only used to generate electricity. In order to most effectively introduce the topics covered in this thesis, it is first necessary to provide some background information on the architecture of the vehicle the supervisor was developed for.

1.1 - PTTR Architecture Description

The University of Washington (UW) EcoCAR 2 team’s vehicle is a Parallel through the Road (PTTR) diesel-electric PHEV. In the PTTR architecture, two primary drive systems are operated in parallel to deliver torque to the ground, with one drive system on the front axle and one drive system on the rear axle. The only physical between the outputs of the drive systems is the road itself, hence the name of the architecture.

On the rear axle the vehicle utilizes a 150kW traction motor with a single speed gearbox. The motor is driven using a high-capacity battery energy storage system (ESS), giving the vehicle around 50 miles of all-electric charge depleting driving range. This RTM system is shown highlighted in Figure 2, with the ESS in green, the motor in yellow, and the gearbox in purple.

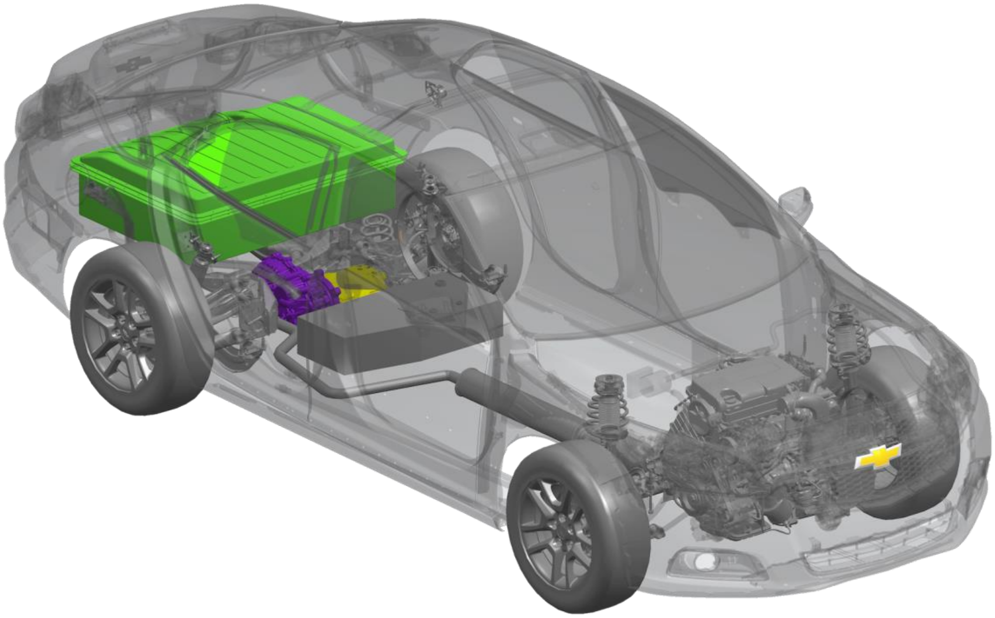


Figure 1. Rear axle motor and ESS system highlighted

On the front axle of the vehicle is a General Motors 1.7L turbo diesel engine paired with a conventional automatic transmission. The diesel engine is run using B20 biodiesel, giving the vehicle approximately 200 miles of extended hybrid charge sustaining driving range. This engine transmission system is shown highlighted in red in Figure 2.

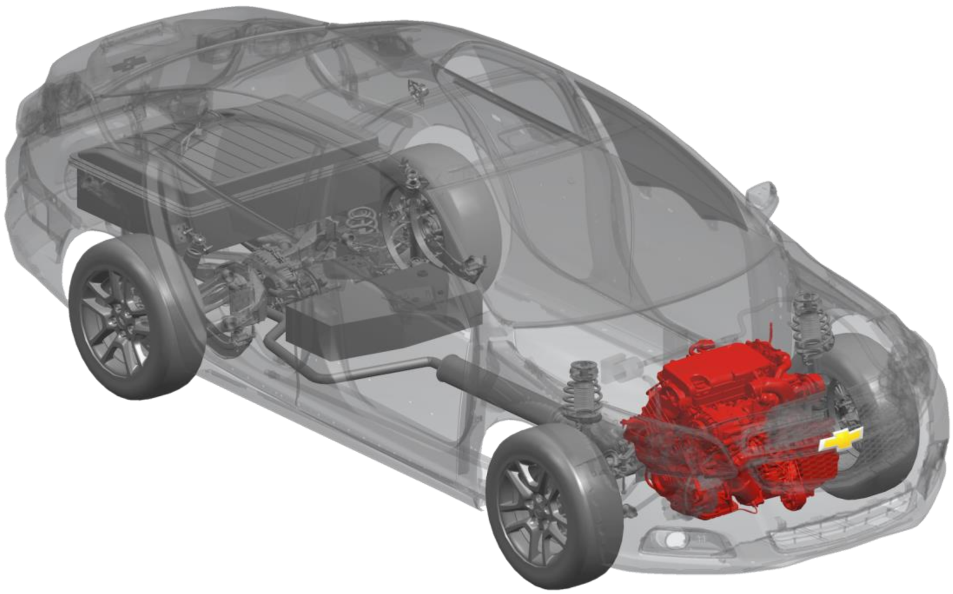


Figure 2. Front axle engine and transmission system highlighted in red

This architecture provides many benefits during normal operation, including [1]:

- All-electric mode for complete petroleum usage displacement during short trips (<50mi)
- Effective load leveling during CS mode without a power-split transmission device
- The ability to use both electric and ICE power in parallel for a boost to tractive power
- Four-wheel drive while using both drivetrains

The UW team used a dSPACE MicroAutoBox II (MABx) vehicle prototyping controller as the primary HSC responsible for distributing torque requests intelligently between the two powertrains. The HSC model was created using Simulink due to the many advantages it presents when creating embedded control system programs.

1.2 - Vehicle Technical Specifications

During the Architecture Selection phase executed in Year 1, the team elected to build a Parallel through the Road (PTTR), plug-in hybrid electric vehicle. Components were selected on the basis of both availability and degree of manufacturer support, in addition to an analysis of which fuel choices would best meet the team goals. The final design utilizes a 1.7L turbo-charged diesel engine coupled with a 6-speeed MH8 transmission to power the forward wheels and a 150kW AC induction motor with a 9.7:1 gear ratio and 18.9kWh ESS to power the rear wheels. Effectively, the platform consists of two independent powertrains designed to work both in unison or independently to give the vehicle multiple modes of operation, maximize fuel efficiency, and reduce greenhouse gas emissions. Inspired by redefining the perception of a hybrid vehicle, more powerful drivetrain components were selected to improve dynamic performance of the stock Chevrolet Malibu while maintaining a charge-depleting range greater than 40 miles. With two powertrains to design, install, and refine, following the EcoCAR vehicle development process has been instrumental in keeping the project on track.

Based on the team requirements developed throughout Year 1, vehicle technical specifications (VTS) were defined as the target values that the vehicle should achieve once implemented. The tables below demonstrate the team’s VTS for both general and competition driving conditions.

Table 1: Vehicle Technical Specifications

Specification	Competition Target VTS	UW Target VTS
Acceleration 0-60 mph	9.5s	7.0 s
Acceleration 50-70 mph	8s	3.4 s
Braking 60-0 mph	143.4 ft	143.4 ft
Hwy Gradeability @20 min	3.5% @60mph	>7%
Cargo Capacity	16.3 ft3	>7ft3
Passenger Capacity	>/= 4	5
Starting Time	<2 s	<2 s
Ground Clearance	155 mm	155 mm
Vehicle Mass	<2250 kg	2160 kg
Vehicle Range	322 km	594 km
Charge-Depleting Range	N/A	76.6 mi
Charge-Depleting Fuel Consumption	N/A	0 Wh/km
Charge-Sustaining Fuel Consumption	N/A	619 Wh/km
UF-Weighted Fuel Energy Consumption	634 Wh/km	215 Wh/km
UF-Weighted AC Electric Energy Consumption	N/A	148 Wh/km
UF-Weighted WTW PEU	624 Wh PE/km	189.4 Wh PE/km
UF-Weighted WTW GHG Emissions	204 g GHG/km	151.6 g GHG/km

Because a very specific drive trace is used with the addition of an approximately 800lbs gas analyzer trailer during the Emissions and Energy Consumption (E&EC) event, technical specifications are shown adjusted in the following table to better reflect the vehicle's performance in that event.

Table 2: Emissions and Energy Consumption VTS

Specification	On road trace at test mass	On-road trace with trailer
Vehicle Range (mi)	369	274
Charge-Depleting Range (mi)	47.6	40.0
ABC - Weighted Total Energy Consumption (Wh/km)	316.8	482.6

VTS were determined using MATLAB, Simulink, and Autonomie software to model and simulate any vehicle architecture’s efficiency and performance given a wide variety of powertrain components. Architectures were simulated against preset drive cycles representing real world driving conditions. In the case of the UW EcoCAR, the platform was developed in Autonomie and tested against a blend of city and highway driving cycles in order to determine the VTS targets listed above.

In the end, the resulting VTS goals are a reflection of the architecture modeled using Autonomie. For instance, using a battery pack with 18.9kWh of storage capacity greatly increases charge-depleting range. All-wheel drive capabilities also greatly improve 0-to-60mph acceleration times, while the ability to transfer between powertrains to either load shift the combustion engine or perform regenerative breaking maintains fuel efficiency.

Chapter 2 - Development of Simulation Platform and Plant Model Requirements

The first step in the modeling and simulation track was to investigate various control system development processes and assess what simulation platforms would be necessary to facilitate efficient HSC development and testing throughout all three years of the EcoCAR 2 competition. Throughout the course of the vehicle development process, a wide variety of simulation platforms were used. Each of these platforms served a particular need in the process, with each one helping increment the team towards full in-vehicle testing. These platforms were identified through a combination of researching publications from industry entities and successful past AVTC teams [1] [2] [3], in addition to receiving direction and advice from competition sponsors and organizers.

The most successful teams in the past have excelled at decoupling controls development from the physical state of the vehicle. Controls development is a particularly challenging track because it is not efficient or possible to wait until the physical vehicle is complete with all wiring intact to begin the development process. Instead, it is highly desirable to utilize multiple testing platforms and vehicle plant models to simulate all the ECUs and associated interfaces that the HSC will interact with once it is in the vehicle. In this way, a development team can produce a reliable and robust set of algorithms and control code that has been proven through a multitude of testing scenarios before even being placed into the actual vehicle. These platforms are summarized in the following sections. It should be noted that the team used MATLAB's Simulink environment for the majority of plant model and supervisory code development.

2.1 - Advantages and Challenges of the Simulink Environment for HSC Development

There were several primary reasons for choosing Simulink as the environment for creating the HSC model. The largest of these is that the MABx and associated dSPACE products heavily utilize Simulink and MATLAB, as outlined in the following section [4].

2.1.1 - Using built-in block libraries for hardware signal transmission

In order to access the I/O on the MABx it was necessary to use Simulink and the premade dSPACE libraries. These libraries provide ways to import analog, digital, and other signals in from each of the MABx's hardware channels. In addition, the libraries also have functions for automatically transmitting a huge number of controller area network (CAN) messages. Most of the components used on the vehicle came with a .dbc file in order to specify the ID's and signal packing of all the required CAN messages. These .dbc files were used to automatically send and receive messages from the vehicle's networks, saving time and ensuring that messages were sent out in the correct format with the validated manufacturer's .dbc files. This was especially necessary with the consideration that, in addition to controlling new electric drive system components, it was necessary to appease both the stock vehicle's controllers and the new engine and transmission controllers.

2.1.2 - Integration with existing vehicle dynamics simulation models

Besides just accessing the hardware and sending out CAN messages, Simulink was also needed because of the model-based control system development process used throughout the competition. It was necessary to begin development of the control system long before any components were installed in the vehicle. To do this, dynamic models of the vehicle's components were used throughout the process to ensure that the system operated in a safe and efficient manner. A combination of dSPACE's Automotive Simulation Models and Argonne National Lab's Autonomie models were used. Both of these model sources used Simulink as their development environment, which made it a requirement to use Simulink for control system development.

Once a model is fully tested with the dynamic models it is ready to be flashed to the MABx. The code for running the program on the MABx is automatically generated using Real Time Workshop in Simulink. This removes any manual creation of C-code, streamlining the progression from model creation in Simulink to executing the code on the target controller on a test bench or in the vehicle.

2.1.3 - Creation of a comprehensible control system for a complex set of components

One of the biggest advantages of using the Simulink environment comes from the ability to create models that are easily understandable by new members on the team. The graphical interface can make it much easier to follow where different signals are going and how different areas of the model work. In contrast, most other programming languages have a significantly higher barrier to entry for new students.

This is even more evident in the UW's HSC model, where it is necessary to interface with nearly a dozen components, perform state of health diagnostics for all of them, and use many different signals to intelligently distribute torque between all the drive systems. Since EcoCAR 2 is a three year long competition, it is necessary to continuously bring new members up to speed to the point where they can develop new algorithms effectively. In the Simulink environment it is easier to dive into all the different areas of the model, achieve a general understanding of how the whole system works, and then use that understanding to develop diagnostics and control code for a particular component.

Even though there are a significant number of benefits to using Simulink, there are also some challenges to overcome when setting up a supervisory model, especially when working with a large team of students that are still learning traditional software development concepts.

2.1.4 - Challenges Posed by Simulink Control System Development

The largest difficulty encountered by the UW team during the HSC model development process was the task of setting up the model for effective and efficient version control. Using a version control system was necessary to ensure that any changes made by the multiple developers could be made on branched models. These branched versions of the code could be tested for safety and reliability before it was used to operate the actual vehicle. In addition, if any changes were found to be detrimental at a later date, the model could be reverted back to a stable version.

Unfortunately, trying to have multiple developers work out of the same large HSC model resulted in many issues with conflicting changes and failed commits. To address these issues, it was necessary to effectively partition the model using libraries and tightly control the procedure used to edit sections of the model.

2.2 - Using Version Control and Libraries to Edit the HSC

With the intent of supporting the concurrent editing of control system software, using library blocks within Simulink can be a strong starting point. Without proper usage, however, they can be used ineffectively, negating many of the potential benefits. Typical subversion control software allows for line-by-line comparison between code, but because Simulink is a visual-based programming language the slightest changes in a control code can modify many lines of code. Because of this volatility, it is desirable to only change singular files while editing control code, rather than changing the entire HSC model at once.

2.2.1 - Version Control System Selected

The UW team elected to use GitHub for version control. This program offers a cloud-based Git version control system to appropriately manage software changes. This program was selected to enable team members to work freely on any computer as long as it could connect to the internet during commits and syncing operations. Since many members worked on personal computers or switched between computers in the EcoCAR lab, GitHub was used to ensure that the models were kept current on any machine necessary regardless of the member’s physical location. In addition, the program offered a lower cost alternative to other more expensive and Simulink-specific programs while still offering an interface that was approachable for inexperienced team members.

2.2.2 - Team Workflow

In order to keep the changes being made by different editors within separate files, it was necessary that each editor is assigned to work within a different library block within the model. The general strategy to do this on the UW HSC was to have only one or two developers work on blocks associated with a particular component at any point in time. They managed all the algorithms and diagnostics needed to interface with that particular component. The libraries were structured according to this breakdown of work, with each library containing all the blocks used to interface with a single component on the car. The actual process used to edit control code without affecting several models at once is as follows:

- Break the link to the library block
- Edit within the block to make changes as desired
- Push the changes to the library block through the "resolve link" right click menu
- Save the library, but close the supervisory code without saving.

This process ensures that the parent HSC Simulink model is not actually modified except in rare instances. Otherwise, team members would be constantly modifying the HSC model itself rather than the individual libraries, which results in a myriad of conflicting changes and failed commits. These failed commits were encountered constantly before a partitioned architecture was created, and caused a significant amount of lost time in the development process. Members would need to go through the model and identify which conflicting modifications should be saved and which ones should be discarded.

These difficulties are particularly detrimental when students are working with version control systems for the first time and have not yet gotten used to the process or recognized the necessity of version control. The students would often focus on frustrations with the tools rather than learning the concepts and building the understanding required to develop a safe vehicle control system.

2.2.3 - Library Layouts

In order to perform these steps in the HSC model, it was decided to construct a model architecture that separates different subsystems in the model based on what component they are primarily used with. Once these partitioned subsystems were created, libraries were constructed to contain all the blocks needed to interface with a particular component. This setup was created with the idea that an individual developer would be focused on developing diagnostics and control routines for one component at a time. In practice, this approach was fairly successful in helping each developer focus on only one area of the model and not cause conflicts with anyone else's libraries. Figure 3 displays the library used for the DC/DC converter component, which uses energy from the high voltage bus to keep the low voltage battery charged.

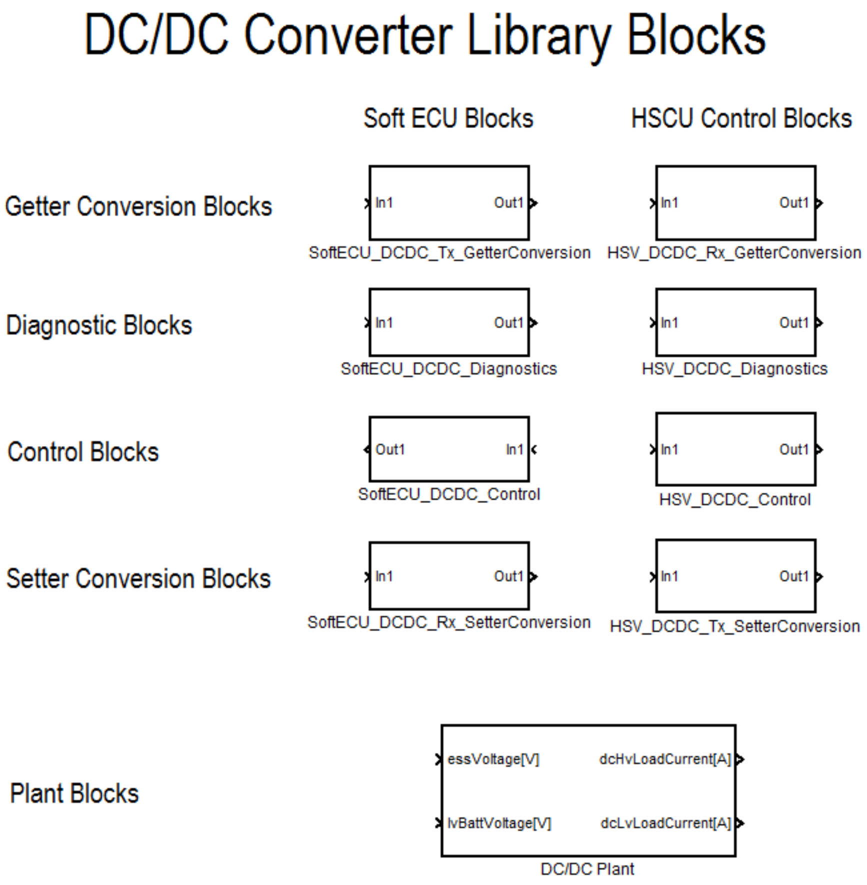


Figure 3. Example library layout for the DC/DC converter component

There are two sets of blocks pictured in the library. The ones detailed in this paper are specifically only the HSC control blocks in the second column. During the development process, Software Electronic Control Units (Soft ECU) blocks were developed to mimic the behavior of different distributed controllers present that the HSC needs to interface with. In addition, plant models were developed to mimic physical behavior of major components. For example, the DC/DC converter Soft ECU blocks produce the same signal set as the real General Motors component on the vehicle, and the plant model is used to model the primary functionality of the physical component itself. This allows for validation that all control system requirements were met for that component before the HSC is installed on the vehicle with the real system of components. It was also helpful to partition the vehicle plant and Soft ECU models during the different phases of control system development.

The following sections go through the overall structure of the UW HSC to show how these different library blocks were utilized, and also to give a basic understanding of how the supervisory control system works.

2.2.4 - Software in the Loop (SIL) Testing Platforms

The most accessible Simulation Platform came in the form of different variations of SIL testing programs and models. During initial architecture selection, the UW team relied heavily on generic plant models and controllers programmed into Argonne National Labs' Autonomie simulation program. This program allows for rapid testing of a variety of vehicle architectures, and includes pre-made vehicle system models, softECUs, and supervisory controllers. It was simply not feasible for a team competing in AVTCs for the first time to adequately design plant models for all the types of hybrid architectures available. Thus, Autonomie offered an excellent platform for comparison studies between a variety of different architectures and components. However, there were some drawbacks to using this software as a simulation platform, most notably the difficulty of modifying the stock vehicle configurations to reflect unique or rare hybrid configurations. In addition, the models used in Autonomie are generally lower fidelity

Torque-Speed Efficiency-based models for the stock vehicle configurations provided by the EcoCAR 2 organizers. This is not necessarily a big issue during architecture comparison and parametric component selection studies, and it also should be noted that higher fidelity models can be dropped into Autonomie vehicle plant models if they become available. The program's models more than adequately served the general requirements developed for the architecture selection phase of the competition, which were focused primarily on utilizing models that could provide comparison numbers on both basic performance metrics and well to wheel (WTW) consumption and emissions results.

Once the architecture was selected, the UW team transitioned to using dSPACE's Automotive Simulation Models to utilize higher fidelity engine models provided by the sponsor and also to have a more easily customizable platform with a lower barrier to entry for beginning students. However, these models also provided a bottleneck during development due to a low number of software licenses needed to run simulations. This was compensated for by developing many of the HSC diagnostics and algorithms outside of the dSPACE system model. Tests were performed in more isolated Simulink test harnesses that only required a MATLAB and Simulink software license, allowing development to proceed with more software developers able to work in parallel.

Overall, SIL as a platform is very useful due to its low barrier to entry. There is no wiring or target controllers to cause additional complications, which is very useful in early development stages. In addition, later platforms utilize real-time processors (RTP) as opposed to the standard processors in normal computers. While this is very useful for accurately simulating the real vehicle environment and Input Output (IO) behaviors of controllers, it usually results in simulations that run slower than possible using SIL simulation. For advanced studies such as parameter optimization or models that use very high fidelity mechanics, it is often useful to simulate in SIL using high-powered traditional computers which can calculate results significantly more quickly. This was especially beneficial in Year 3 of the competition, where more sophisticated Charge Sustaining (CS) strategies required full drive-cycle based optimization of various control parameters.

2.2.5 - Hardware in the Loop (HIL) Testing

Once the initial phase of SIL testing is nearing completion, it becomes necessary to transition to a platform that more accurately represents the operation of both the target controller and the interfaces it will experience. Various varieties of HIL testing offer an excellent platform for this set of requirements. Not only do embedded processors operate using RTP, but simulators can provide simulated interfaces with these embedded processors to emulate all the signals that the HSC will see on the real vehicle. This facilitates more true-to-life testing, allowing for insertion of faults in the physical interface layer.

The initial phase of HIL bench testing system development can be seen in Figure 4, where the HSC, in this case a MicroAutoBox (MABx) from dSPACE, is connected to a variety of physical inputs that mimic the normal driver inputs found in vehicles. The dSPACE simulator then simulates the plant model and soft ECU system, mimicking the signal transmission that the MABx will see on the vehicle bus.

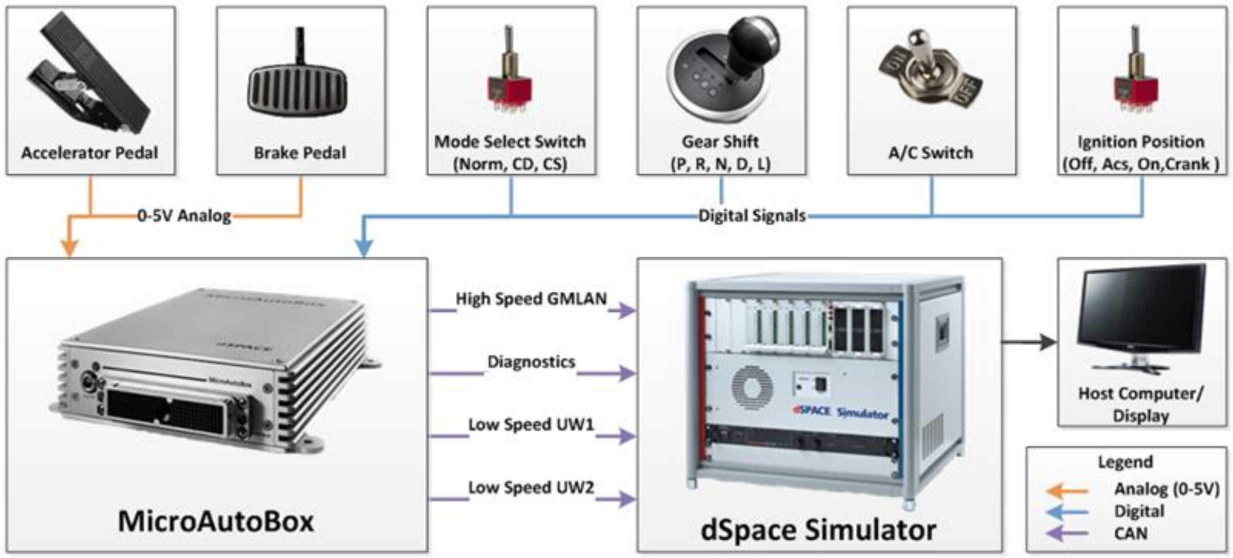


Figure 4: Year 1 HIL Platform Configuration

As component controllers are received, they can be integrated into the HIL harness until there is an entire vehicle system equivalent set up on the test bench. In this manner, when it becomes time to transition to in-vehicle testing, the MABx should be able to be unplugged from the HIL bench and swapped into the vehicle without any significant change in HSC programming. The transition from SIL to HIL to VIL testing is thus significantly smoother, since diagnostics and algorithms can be quickly tested on the HIL bench where components and physical connections are much easier to access and diagnose.

However, there are some drawbacks to using only HIL testing in the development process. There is a noticeably higher overhead for most operations when dealing with embedded controllers. Instead of just playing the model on a PC, it become necessary to establish a link with the target HSC and HIL simulator in order to flash new builds of HSC and plant model programming to the MABx. In addition, the physical nature of CAN and other signals means that it is possible to inadvertently introduce physical faults in the wiring. If a certain segment of code is not working as intended, it becomes harder to root cause the problem to a software or hardware issue. This is why maintaining an updated SIL platform is vital to efficient development, so that code can be validated in the SIL environment first in order to narrow down the possible causes of problems.

2.2.6 - Vehicle in the Loop (VIL) Testing

This simulation environment was the ultimate goal of the controls testing development cycle, ending with a controller able to successfully drive the vehicle using a robust set of diagnostics and control algorithms. Since this environment is the real vehicle itself, it is very useful for validation of the simulated models used throughout the rest of the development process. This validation not only aids in future design cycles, but also enables more accurate control parameter optimization studies in the SIL environment.

However, it is also the most rapidly changing environment, with multiple teams making modifications continuously thanks to the accelerated VDP design cycle. This makes it very difficult to maintain full experimental controls, making debugging issues over a long period of time very difficult. Because of this, it was necessary to maintain the ability to switch back and forth at will between the different environments. The next section details efforts to make those transitions smoother and more efficient.

2.3 - Maintaining Consistent Models and Interfaces Across Multiple Platforms

In order to maintain the capability for SIL, HIL, and VIL testing throughout the course of the competition, the UW team made extensive use of library blocks in Simulink and also generated a separate library for each individual component. These libraries facilitate parallel code development between all of the controls team engineers. In addition, they help keep the blocksets up to date across any models that use them. This library is the same for both SIL, HIL, and Vehicle testing, which allows easy transitions between all three testing platforms.

For SIL testing purposes, a local version of the MABx control model has been inserted into the HIL model. The Soft ECUs have been set up to be able to communicate with both the local supervisor for SIL testing along with the remote MABx supervisor for HIL testing. A SIL virtual network bus has been created in order to simulate the cycle times and latency involved in CAN message transmissions for safety-critical serial data messages. This same control model is used in HIL and in-vehicle testing using the library system. The team's version control program is used to keep models up to date on any computers that use them.

For any in-vehicle dynamic testing, only the code contained in the Master branch is used on the MABx. This code is run through a standard set of tests each time a sub-branch is merged in to ensure safety-critical functionality is still intact.

In addition to software compatibility, it was also vital to keep both the HIL and VIL physical interfaces as close to each other as possible. This allowed the team to remove the MABx from the vehicle at any point during the development and plug it into the HIL harness, while still maintaining the same code set for debugging purposes. This was accomplished through thorough documentation of the harness pin-outs in the same Visio diagram. Whenever signals were added to the VIL harness, an effort was made to keep the HIL harness current as well.

2.4 - Identifying and Managing Plant Model Requirements throughout the VDP

At the beginning of the EcoCAR 2 competition, the University of Washington (UW) team identified several primary phases of model development to transition through in order to reach a highly refined vehicle control system at the end of Year 3.

The team elected to use the controls development process defined by the traditional V-Diagram [2]. This diagram outlines a process that fits well within the model-based design that was undertaken, with an emphasis placed on safety critical items within the control strategy. The development process also helped guide the plant model development, with requirements derived from linked HSC system requirements. This process is outlined in the following sections, which explain the adaptation of requirements to changing stages in development throughout the three year competition.

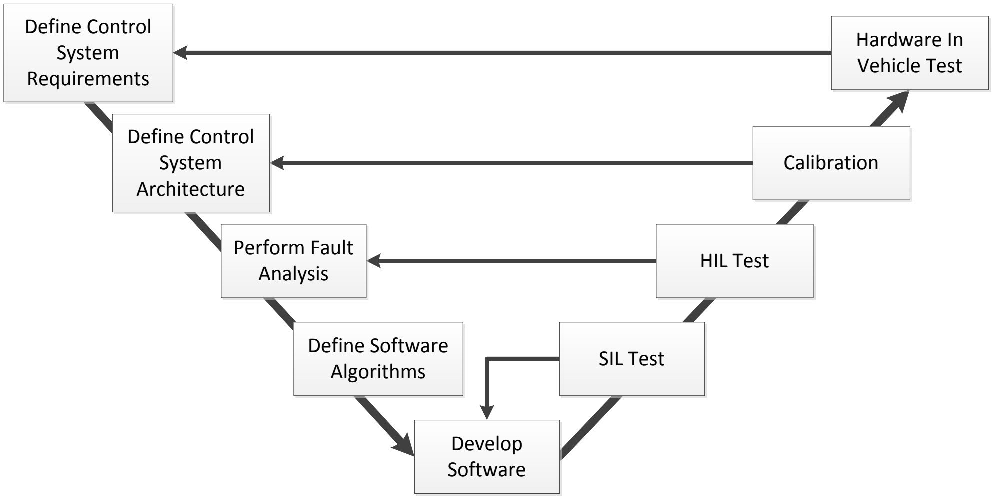


Figure 5: University Of Washington controls development process

2.4.1 - Develop Overall HSC Requirements

The first step in developing the system requirements is to define what must be controlled. The controlled components in this case are the safety critical components, motors, engines, cooling pumps, contactors, etc. The second step is to define what data is required to control them; this data comes from a variety of locations, such as driver input and current vehicle status sensors. The third step is to define the interfaces between different control modules, sensors and supervisory modules (e.g. CAN bus interfaces) using the previously documented requirements.

2.4.2 - Control System Architecture Requirements

Once the controls system requirements had been defined, the control system architecture could be designed. The architecture links together the information presented by the requirements in a coherent manner. It must show how user input will be routed through control modules, and how information will be routed from the supervisory controller to the low level control modules. The final UW HSC architecture is shown in Figure 6.

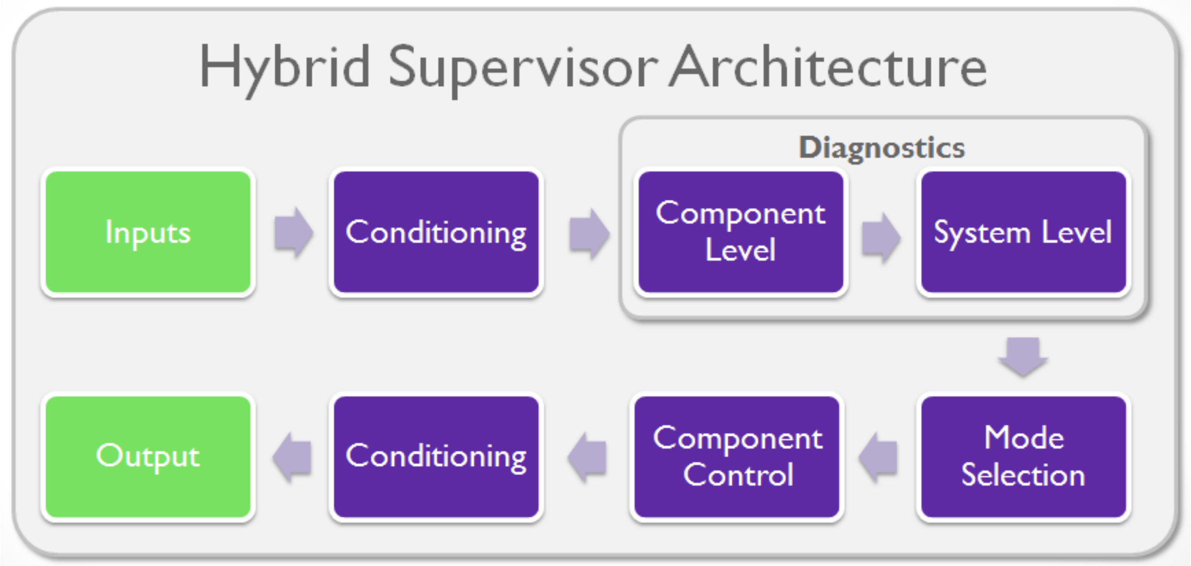


Figure 6: Final Hybrid Supervisory Controller Architecture

This architecture was designed for maximum possible partition of areas in the model, making it possible to easily identify and link plant model and softECU requirements to each subsystem. In addition, it facilitated smoother documentation of categorized requirements for the HSC functions in the controls master spreadsheet.

2.4.3 - Fault Action and Analysis

The fault action/analysis will be performed once the control system architecture has been developed. The fault analysis will consist of identifying which components and interfaces may fail. The fault action plan will define the required actions to be performed for each fault which may occur. The results of the fault/action analysis must be incorporated into the algorithm development process, which means that the plant model system needed to be capable of simulating each of the primary faults identified from this phase of the design.

The primary tools used to conduct this analysis were Design for Failure Mode Effects Analysis (DFMEA) and Fault Tree Analysis (FTA). Each approach has its benefits and drawbacks, but perhaps the most helpful one for developing initial plant model requirements was FTA. Using this analysis, components were investigated to determine all potential causes that could lead to a particular fault. This means that the plant model needed to be able to simulate each of those causes, which provided a direct source of plant model requirements. This analysis process was documented using the controls requirement master spreadsheet, and was updated throughout the competition.

2.4.4 - Develop and Implement Control Algorithms

After the control system requirements have been defined, the overall system architecture has been developed, fault analysis has been performed, and the fault action plan defined, algorithm development for the software controls may begin. The control algorithms will be the actual logic used to process data/signal inputs, set flags, operating modes, and calculate power/torque request for sub-system components. In order for the plant model to support this process, the physical systems and softECUs needed to be able to produce the outputs that would be read by the HSC. These additional requirements were recorded in the controls master spreadsheet.

2.4.5 - SIL/HIL Testing Process

During the course of Year 1 of the competition, SIL and HIL models were developed based on generic versions of component controllers. Since components had not been selected yet, most of the plant model requirements were based on being able to produce the general signal set needed by the HSC for normal operation, along with any safety-critical fault modes identified with DFMEA and FTA.

Once Year 2 began, the requirements for the plant model were updated to reflect the need for more accurate representation of signals produced by each softECU. Since at that point most of the controllers had been selected and ordered, there was a greater level of documentation available defining the signal sets and interfaces needed for each component and ECU. The plant model requirements for this stage of development mostly concerned adjusting the softECUs and associated plant models to produce the same set of signals that would be seen by the HSC on the real vehicle. This made the transition to in-vehicle testing significantly smoother, since both environments were designed to appear the same to the HSC.

2.4.6 - Hardware in Vehicle Testing

After all steps are completed, the controller was physically integrated into the vehicle with a pre-developed wiring harness for a fully functional vehicle roll out test. As mentioned previously, a great deal of care was taken to maintain backwards-compatibility with previous simulation platforms.

Chapter 3 - Supervisory Control System Development

One of the most challenging tracks in the EcoCAR 2 competition is the development and testing of the vehicle's hybrid control system. Over the course of the three year competition, the UW team has integrated in over a dozen powertrain and auxiliary components and their associated ECUs. The team has moved from purely software-based testing and development of primary hybrid vehicle modes to using HIL test bench simulations to final integration into the vehicle itself. The following sections document that process throughout the competition, with a focus on Year 3 optimization and vehicle testing [5].

3.1 - Control System Overview

Before elaborating on the control strategies developed by UW, it is first helpful to understand the system of hardware used and how the different modules and components interact with each other. The majority of the controls work throughout the years has been focused on developing programs that allow the vehicle's Hybrid Supervisory Controller (HSC) to interface with and control the different systems on the vehicle. The primary purpose of the HSC is to receive inputs from the driver and intelligently determine how to distribute those requests to each powertrain system in a manner that is not only efficient, but also safe and acceptable to the average consumer. In Figure 7 the team's MicroAutoBox II (MABx) from dSPACE functions as the HSC, distributing torque requests from the driver down to each of the individual powertrain systems.

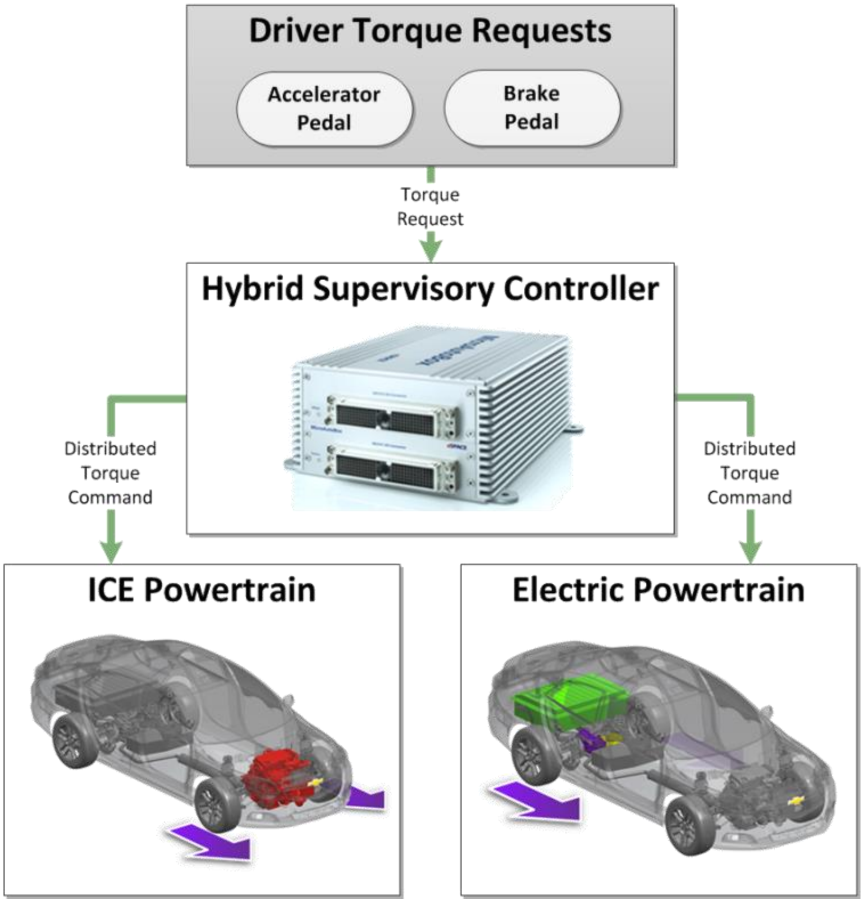


Figure 7. Torque Request Distribution Using HSC

As explained in previous sections, the UW hybrid architecture consists of two disconnected powertrains on the front and rear axle. The HSC must decide whether to send the entirety of the driver's torque request to the internal combustion engine (ICE) powertrain, the electric powertrain, or a combination of both. That request is distributed based on the capabilities of each required controller and component system that are needed to run the each powertrain system, and the HSC software is designed to assess these capabilities before selecting a primary hybrid mode for safe and functional vehicle operation.

3.1.1 - Electric Powertrain Hardware

Figure 8 displays the power transfer through the electric powertrain along with the controller interfaces that exist between the HSC and the various ECUs. In the figure, blue arrows represent the flow of analog, digital, and CAN signals between all the controllers. Electrical power transmission is shown in yellow lines, and mechanical torque is shown in black.

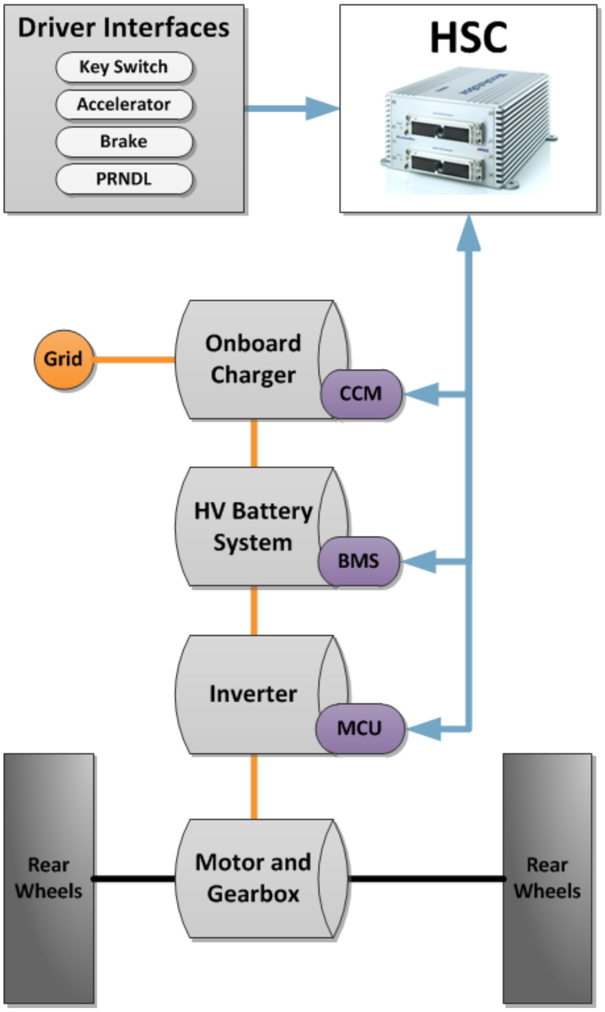


Figure 8. Electric Powertrain Interfaces

A123 Battery Management System (BMS): The battery system is managed by the BMS. It manages the charging, balancing and contactor operations of the ESS, and communicates with the MABx via CAN. The BMS has direct control over the contactors that allow current to flow to/from the batteries and the high voltage bus.

NLG5 Brusa Charging Control Module (CCM): This 3.3KWH charger communicates directly with the BMS through CAN. The charger only turns on when the AC plug is plugged into the vehicle, at which point it wakes up both the BMS and the MABx, allowing the batteries to begin charging and the coolant system to thermally manage the charger.

Rinehart Motor Control Unit (MCU) and Inverter: The inverter receives the torque request for the eSystem from the MABx via the CAN bus. It then calculates and supplies the required 3-phase waveform to be input to the motor to output the required torque.

Remy Electric Motor and GKN Gearbox: The electric motor receives 3-phase AC current from the inverter and sends motor position and temperature back via a low voltage harness. The only component that directly interacts with the motor is the MCU and Inverter, all of the information pertaining to the motor is relayed through the MCU to the CAN bus.

3.1.2 - ICE Powertrain Hardware

GM Engine Control Module (ECM) and Diesel Engine: The operation of the engine is managed primarily by the ECM. Once a torque request is sent to the ECM via two analog signals from the MABx, the ECM calculates the air fuel ratio, throttle opening, and any other parameters needed to meet the required request. Throttle request is sent to the ECM via signal wires that originally connected to the accelerator pedal sensors. This allows for the MABx to intervene on the driver's original throttle request, allowing it to intelligently split torque between the two powertrains.

GM Transmission Control Module (TCM) and MH8 Transmission: The TCM shifts the gears of the transmission in the diesel powertrain depending on the request of the MABx, which intercepts the driver's PRNDL commanded gear. This allows the transmission to be in neutral even when the gear shifter is in "drive" during electric only mode.

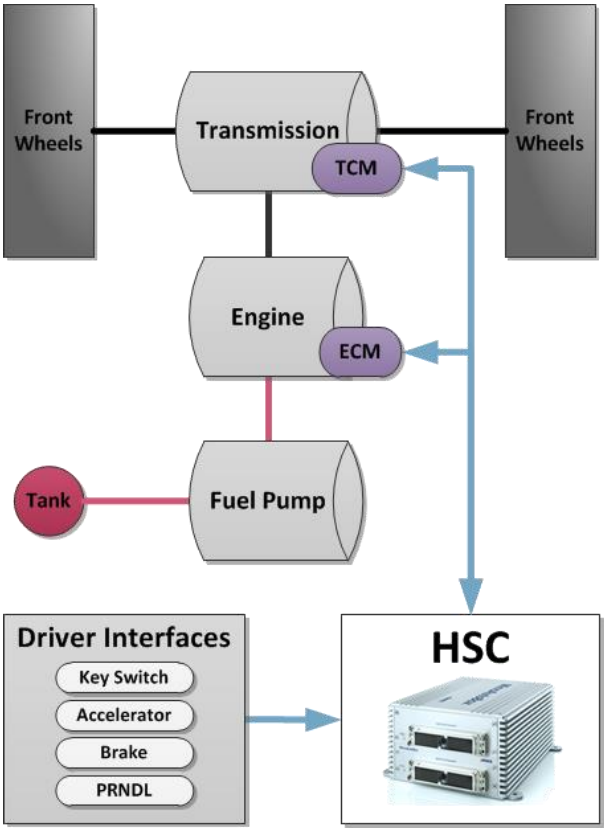


Figure 9. ICE Powertrain Interfaces

3.1.3 - Emissions and Auxiliary Hardware

Cooper Bussmann Multiplexed Vehicle Electrical Center (MVEC): Because the MABx has relatively low-current digital outputs, two CAN-controlled relay and fuse blocks known as

MVECs are used to distribute fused power to new powertrain components, turn components on and off via wake and power line relays, drive LEDs, and deliver other higher power requirements. Each MVEC is woken up by the MABx's digital outputs, and communicate with the MABx via CAN.

Bosch Dosing Control Unit (DCU): This component controls the urea dosing to eliminate NOx in the emissions. The MABx determines how much urea must be dosed to eliminate the NOx in the emissions, using information supplied by two NOx sensors and an ammonia sensor. Communication is done via CAN with power supplied by the MVECs.

GM Accessory Power Module (APM): During electric only mode, the alternator on the engine is unable to supply power to the 12 volt battery, so the APM converts 350VDC from the ESS to 12VDC for the low voltage battery. Again, communication with the MABx is done via CAN.

3.2 - Supervisory Controller Software Architecture

The HSC layout was designed to import a variety of inputs from each component's ECU and diagnose the current status of each drive system. Figure 10 displays the primary HSC model areas that enable this signal import and drive system availability assessment.

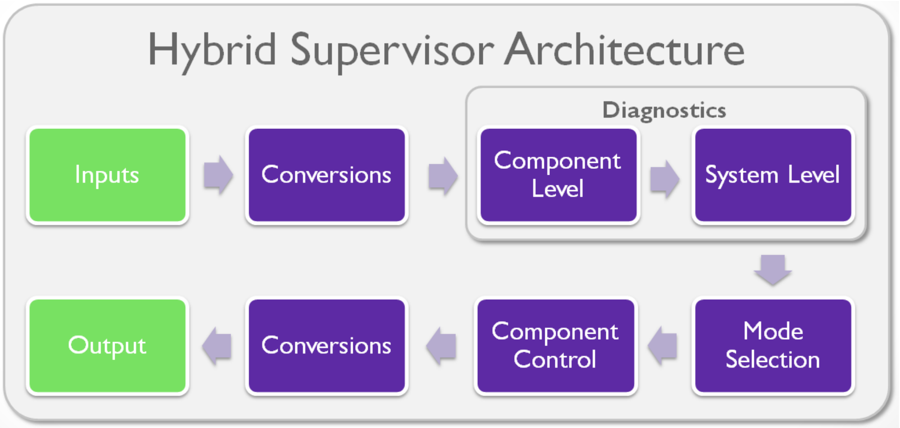


Figure 10. HSC Software Architecture and Primary Subsystems

3.2.1 - Signal Conversions and Override Switches

The code begins by taking in input signals via CAN, analog, and digital I/O from each component on the vehicle. These inputs are then passed through the conditioning block, where operations such as unit conversions, pedal mapping, and variable type assignment convert all the signals into a usable form [1].

For example, the high voltage battery pack used on the vehicle was comprised of a set of A123 Lithium Iron Phosphate modules which were monitored by their integrated Battery Management System (BMS). The BMS receives commands from the CAN bus-connected HSC to power on, close contactors, and perform other actions. It also reports the status of the ESS over CAN. In the signal conditioning area, the HSC converts the signals from the BMS into units and variable types that match those used in the HSC model.

In addition to this conversion, each signal also has an override switch block added that allows a developer to override the incoming signal from the BMS or other components. These override switches can be used in the testing and validation process to simulate faults and ensure that the HSC detects and acts on these failures effectively. The detection of failures and other component health checks is programmed into the component level diagnostics area of the model.

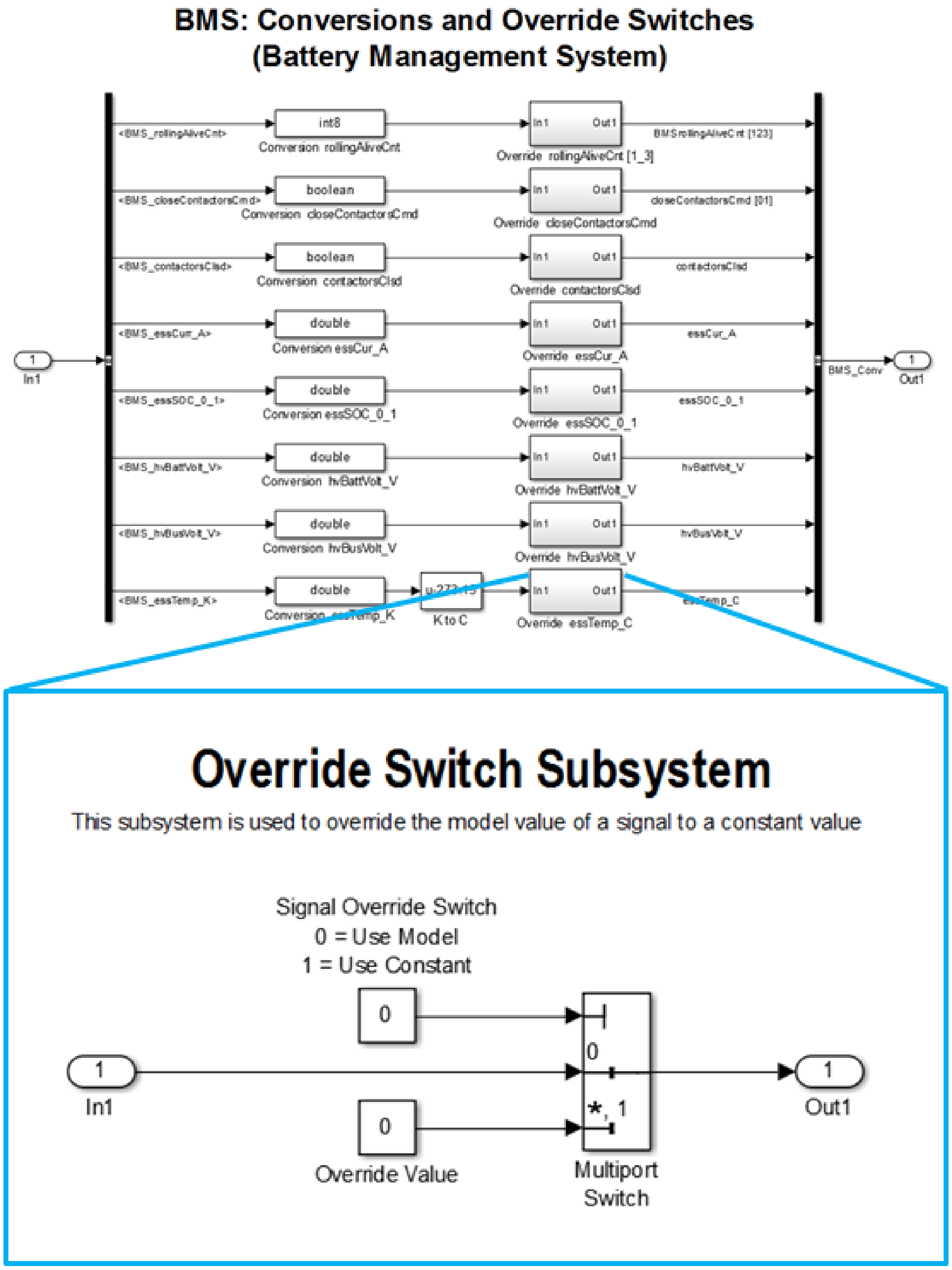


Figure 11. Conversions and override switches subsystem

3.2.2 - Component Level Diagnostics Structure

In the Component Level Diagnostics area, all of the signals from a particular component and its ECU are analyzed to determine the state of health of that particular component. [1]

This evaluation is based on the approach that a component's ECU has been pre-programmed by outside entities to disable the component operation in the case of events which would cause harm to the component or vehicle occupants. As such, it is necessary to perform two sets of checks on each component. First, the code checks the component signals to determine if the component's ECU has brought the component offline unexpectedly due to the presence of various manufacturer-defined trouble codes. For example, there are a number of instances where the A123 BMS will command the ESS to disconnect itself from the high voltage bus by opening contactors, preventing the use of the ESS for the remainder of the drive.

Second, the code checks to see if any situations are observable that may soon lead to the component's ECU bringing it offline, such as a high temperature in the ESS coolant that would cause a BMS-commanded power off if it was allowed to continue rising. The code also checks for any additional failures that are not checked for by the component's ECU, but have been identified by the UW team's failure mode effects analysis as a high-risk event. In this way, the state of health for each component is evaluated to be Online if no current or impending failures are detected, Offline if a failure has been determined to have occurred, or Limited if it has been deemed necessary to limit the subsequent component operation to avoid a failure. [1]

The HSC subsystem for Component Diagnostics is pictured in Figure 12. It is important to note that each of the blocks pictured in the figure are linked library blocks. These blocks contain the diagnostics mentioned in the previous paragraph for determining the status of each individual component. By utilizing a highly partitioned model structure and separate libraries for each component, every time a new diagnostic was added the only file that needed to be committed was the component's library model. This means that, as long as only one member of the team was developing code for a particular component at any point in time, they would only be updating the library of that component and would not create changes that conflict with changes made to a different component's diagnostics by another member of the team [4].

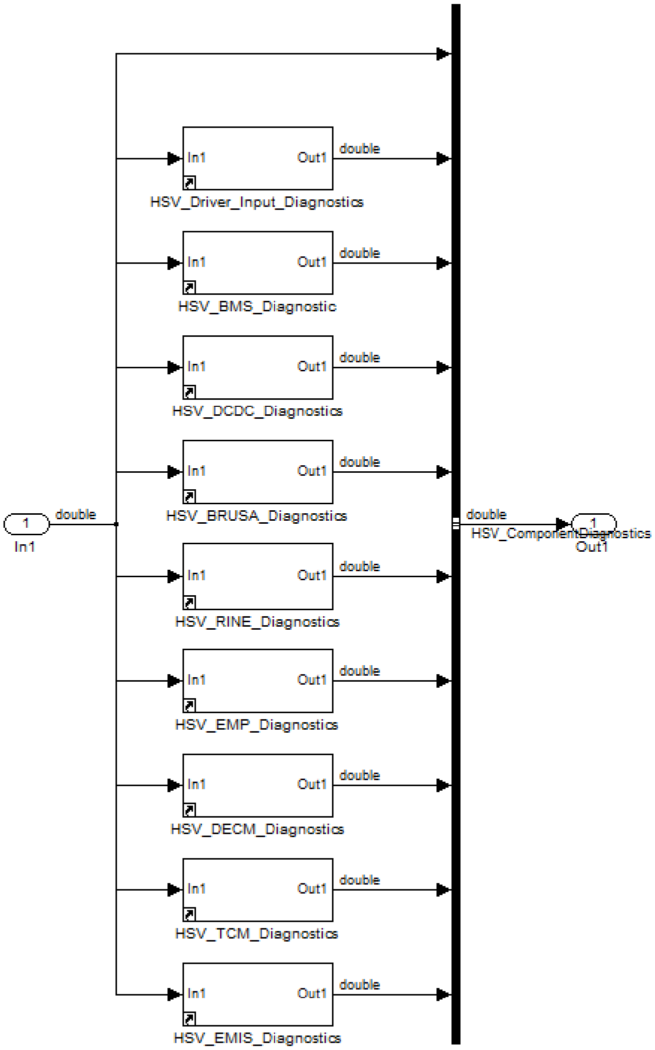


Figure 12. Component level diagnostics subsystem layout

Once the statuses of each component were determined, they are passed into the next area of the model, system level diagnostics [4].

3.2.3 - System Level Diagnostics Structure

The results of the checks for all the components are used to evaluate both the B20 internal combustion engine (ICE) and rear traction motor powertrain systems as a whole to be Online, Offline, or Limited based on the status of all of their critical components. If any one of a powertrain system's critical components is offline, the system is considered to be Offline and all modes that use that system for propulsion are disabled. The system level diagnostics subsystem is pictured in Figure 13.

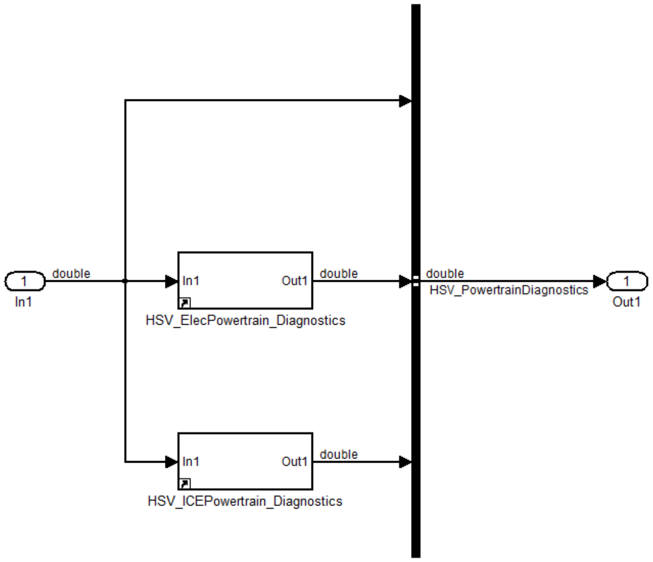


Figure 13. System level diagnostics subsystem layout

Note that these subsystems are partitioned and in libraries as well. A lower level look at the electric powertrain subsystem can be found in Figure 6.

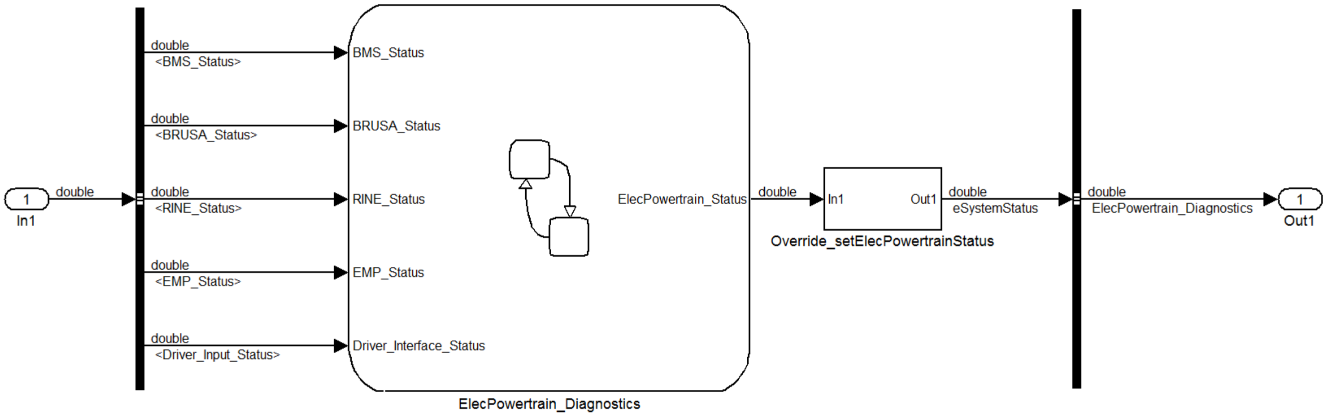


Figure 14. Electric drive system diagnostics model layout

These converted signals are then passed into a Component Level diagnostics subsystem, where the health and capabilities of each individual controller are assessed. Each component is given an assessment of Online, Offline, or Limited operation based on everything from temperature levels to rolling alive counters. The assessment of each component is then used in the System Level assessment to determine the overall ability of the ICE and electric drive systems to output torque and provide other required functionality. For example, if a critical component of the electric drive system like the motor inverter is not functioning, that would cause the HSC to assess the electric drive system to be Offline and unable to provide torque.

The system level diagnostics are passed into the mode selection area, where the HSC selects primary modes of operation based on ESS state of charge, powertrain system availability, and several other factors. Modes that use a particular powertrain are only allowed if that powertrain has the capability to deliver driver demands. If the electric powertrain is Offline for example, this would force the vehicle into an ICE Only mode to ensure that the vehicle only requests torque from powertrains that can deliver it safely. This Mode Selection Process and the primary modes are described in later sections. Once a proper mode is selected and the torque and other requests are effectively distributed, the Component Control area translates these general requests into specific commands for each component ECU. The translated commands are then converted to raw signal values for transmission to the various hardware components. These components and ECUs are defined in the next sections along with how the different components interact with each other.

3.2.4 - Mode Selection Logic Structure

The results of the System Level evaluation are then passed into the Mode Selection block. This block contains the main set of strategies for controlling the different powertrain systems. The UW team elected to use a set of discrete modes of operation in order to determine how to coordinate the torque requested from the two powertrain systems. Transitions between operation modes governed by a rule-based decision making process. While there are several more complex and possibly more efficient strategies available, it was decided by the team that a conservative rule-based approach would better align with the team goals. By using rule based control, the team could focus on rapidly developing a reliable and functional control system and spend more time on the testing and integration process. The next section explains these strategies in detail.

3.3 - Control Strategy Goals and Modes

As described above, the Mode Selection area is responsible for selecting the correct mode of operation for the current state of the vehicle. The following sections outline the primary vehicle modes used in this selection process and the goals accomplished by each one.

During the Hybrid Mode Development process, the UW team designed hybrid operation modes to accomplish a number of primary goals. From a consumer-acceptability standpoint, the team wanted the vehicle to feel as responsive as possible and have a similar acceleration profile in every primary mode. In order to ensure success in competition dynamic events, the vehicle should minimize overall energy consumption and criteria emissions as much as possible while still maintaining driveability. Finally, the vehicle should have additional modes that allow the vehicle to be driven in high performance mode if desired, and limp home modes if a fault brings either powertrain system offline. The resulting modes that accomplish each of these goals are outlined below.

3.3.1 - Charge Depleting Mode

In order to accomplish the goal of minimizing fuel consumption and emissions as much as possible, the vehicle operates in a Charge Depleting (CD) mode whenever the SOC is over a certain threshold. In this mode, the ICE is turned off and all torque requests are delivered by the eSystem, as shown in Figure 15.

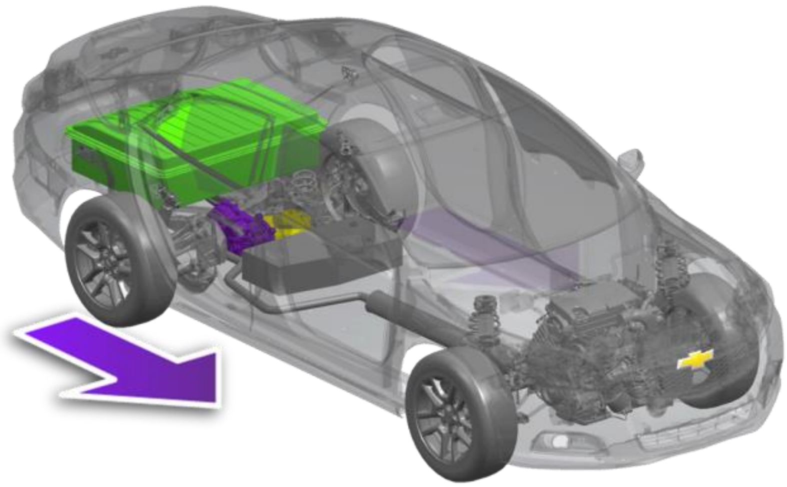


Figure 15. CD Mode Uses ePowertrain to Deliver Torque

This mode also achieves a high degree of smooth driveability and vehicle responsiveness due to the motor’s near instant response time and lack of gear shifts. Regenerative braking is also allowed in this mode, and has been calibrated to ensure smooth transitions between propulsion, braking, and slowing to a stop.

3.3.2 - Charge Sustaining Mode

Once the vehicle drops below a lower limit to SOC, the HSC triggers a shift to Charge Sustaining (CS) mode. This mode is the most complex of any of the modes in order to precisely control startup and operation points of the ICE to minimize both emissions and fuel economy. The team is still optimizing the ICE warm-up strategy by investigating the effect of loading the engine to various degrees to balance fast exhaust catalyst heating with the various criteria emissions rates caused by loading the engine during heating process. Once the ICE is sufficiently warm, the CS mode operates in a few general stages. When the vehicle is at a stop, the ICE is turned off and the vehicle launches by using the eSystem to deliver the full driver torque request, as shown in Figure 16.

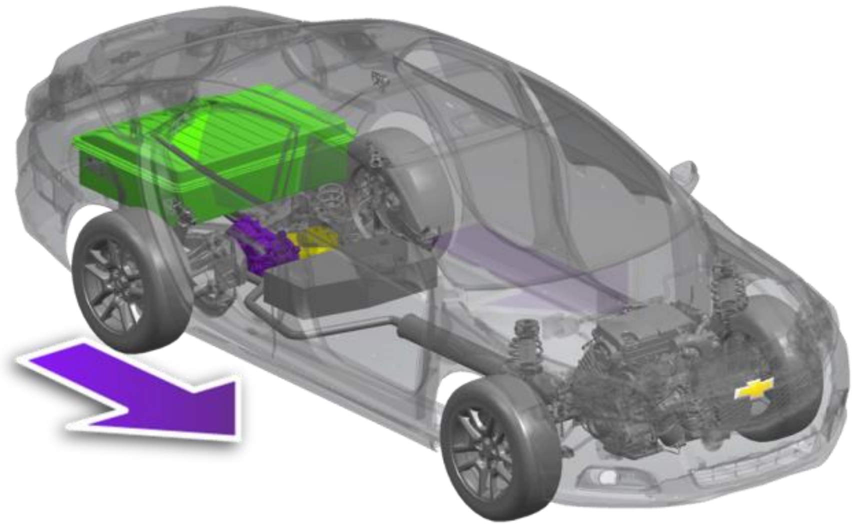


Figure 16. CS Electric Launch Phase

This electric launch helps ensure that vehicle has the same smooth responsiveness in both CD and CS modes at lower speeds. It also helps avoid using the ICE excessively before the torque converter’s lockup clutch is engaged, which would decrease torque path efficiency through the ICE powertrain system. Once a certain speed threshold is reached, the engine is started and torque from the iceSystem is slowly blended in as speed increases. The overall vehicle torque request from the driver is still achieved by reducing the eSystem torque at the same rate the iceSystem increases. This blending process is shown in Figure 17.

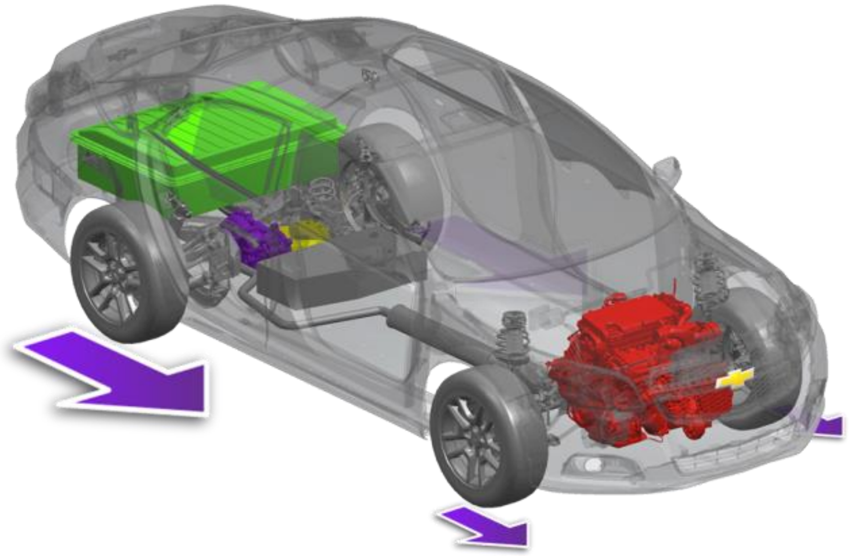


Figure 17. CS ICE Transition Phase

This blending is calibrated to smoothly transition to using the iceSystem to deliver the majority of torque requests in a way that is enjoyable and safe for the driver. Once the iceSystem is successfully blended in, the ICE Propulsion with Load Shifting phase begins. In this phase, the HSC continually calculates an ICE target torque based on a number of factors. The team experimentally derived efficiency and criteria emissions maps of the engine operation points based on torque and speed. Each of these maps was given a weighting based on their impact on scoring in the Emissions and Energy Consumption (E&EC) event, and these weights were used to create a combined engine operation map that can be used to find the best torque for a particular engine RPM. This is the torque request sent to the iceSystem, and the difference between iceSystem torque at the wheels and the driver's vehicle torque request is delivered by the eSystem. In this way, the eSystem provides the positive or negative torque required to load shift the engine into the desired operation region while still allowing for the correct overall vehicle torque, as shown in Figure 18 below.

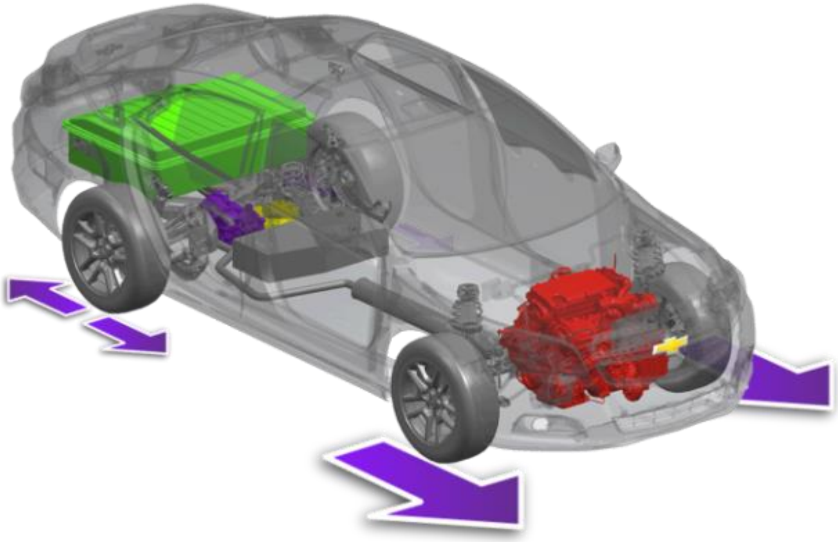


Figure 18. CS ICE Propulsion with Load Shifting

3.3.3 - Additional Modes

The final two modes of the vehicle are used in very specific situations. Performance mode is used to enhance consumer acceptability in situations that require a high performance vehicle. To enter this mode the driver must press both the brake and accelerator pedal for two seconds, which causes the HSC to start the engine if it was off and begin evenly splitting torque between the two powertrains. In addition, the pedal mapping is adjusted to reflect this new distribution, with 100% pedal corresponding to full torque output from both powertrains.

In addition, the HSC has been programmed with an ICE Only mode, which is entered if the eSystem is ever evaluated to be offline. In this mode, all torque requests are sent to the iceSystem and regenerative braking is disabled. In a similar manner, if the iceSystem faults out and is evaluated to be Offline in the System Level diagnostics, the vehicle will switch to CD mode. Both of these modes help the vehicle return home safely in the case of faults, enhancing vehicle reliability and consumer acceptability.

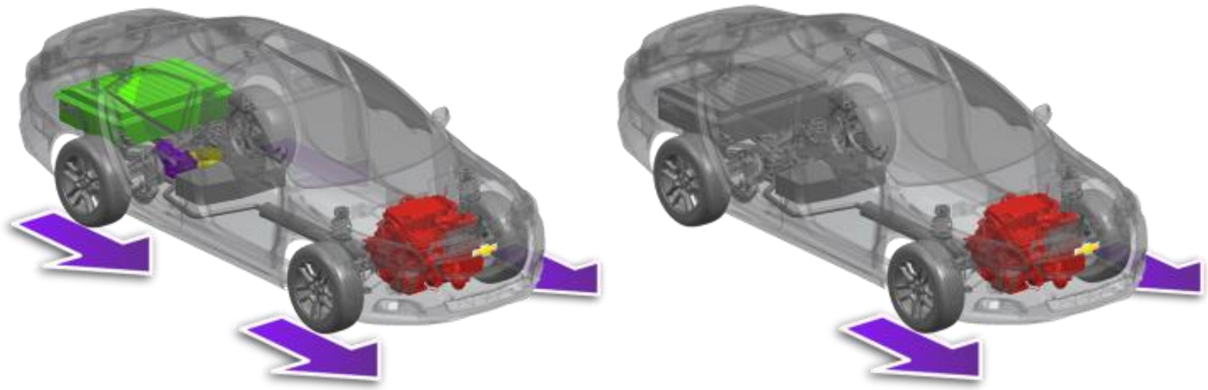


Figure 19. Performance and ICE Only Modes

Chapter 4 - Plant Model Development and Validation

In order to test the control model throughout the 3-year development process, it was necessary to develop a series of Simulink models that simulate the behavior of the primary drivetrain components and overall vehicle dynamics. This vehicle plant model was used in conjunction with the control model to refine basic control algorithms, validate diagnostics, and perform extensive testing of all controller functions before the code was used on the actual vehicle during dynamic testing. It was important to use models that provided an accurate simulation of the real components so that informed decisions and refinements could be made before testing the control code on the actual vehicle. The following sections outline the development of the primary drive system component models, along with the real-world validation testing conducted to ensure the models were adequately accurate.

4.1 - Plant Model Development Process Overview

The key to developing useable plant models was not starting from scratch. As a result, the UW team used pre-built models from Autonomie as a development starting point in an effort to minimize complexity as much as possible. The testing procedures were more interchangeable across a wide variety of plants in effect.

4.1.1 - ESS Model Development

The UW team evaluated a variety of potential models types to simulate the behavior of the ESS. Because battery modeling is a fairly new area of research, few tests have been done to evaluate the accuracy of the previously mentioned battery models. Fortunately, the Beijing Institute of Technology (BIT) has performed the analysis required to evaluate the accuracy of several battery models [6]. The results of their testing are shown in the table below.

Table 3: Beijing Institute of Technology Results [6]

Model	Maximum Error (V)	Mean Error (V)	Variance (V^2)	Error Rate (%)
Simple Internal Resistance Model	1.62	0.39	0.076	2.81
PNGV Model	0.57	0.087	0.024	1
DP Model	0.21	0.043	0.0021	0.38

Based on the results from the BIT study, the team selected the DP model as the most accurate model for simulating behavior of the team's ESS. A diagram of this model is shown in the figure below.

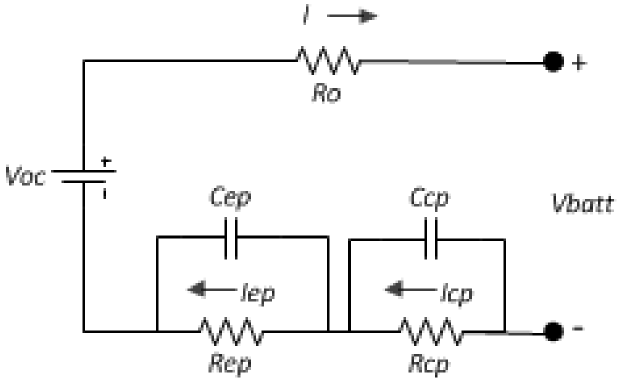


Figure 20. Dual Polarization model used [6]

In order to parameterize the DP model to simulate the team's ESS, a procedure developed in the The *Battery Test Manual* published by the Idaho National Energy and Environment Laboratory in 2001 was used. This manual outlines a procedure for deriving battery parameters from experimental HPPC data. Given the open circuit voltage across the battery terminals, the actual battery voltage, the current load as a function of time, and SOC, numerical solutions can be calculated for the two polarization currents [7]. Using proprietary HPPC data from the battery module manufacturer, the derived parameters were packaged in the dual polarization model developed by ANL and integrated in the vehicle system. To represent the temperature of the battery modules, it is assumed the power loss due to voltage sag under a current load goes into heating the modules. A heat capacity is assumed and the mass of the battery modules is known, thus an approximate temperature is calculated. This model development process fulfilled both the requirements for model fidelity and signal generation to simulate fault scenarios in the ESS system.

4.1.2 - RTM Model Development

The RTM model structure was borrowed from Autonomie models and modified for integration in the TTR vehicle system. The model can be summarized as a power balance with the introduction of an efficiency value based on efficiency charts provided by the manufacturer. Also simulated is the decay of the effective RTM torque from the peak torque to the continuous torque as a function of time and current speed.

The inputs to the model are the RTM torque request issued by the supervisory controller, current motor speed determined by vehicle dynamics, and ESS voltage which is calculated in the aforementioned battery model. Using peak and continuous torque curves provided by the manufacturer, the model either grants the torque request or provides the maximum available torque (effective torque). The decay from peak to continuous torque is governed by a time constant that represents the heating factor of the motor. Next, the motor efficiency is calculated using efficiency charts provided by the manufacturer which plot the motor efficiency as a function of both torque and speed. Finally, the current draw is calculated and sent the battery model as the value of the load current.

4.1.3 - ICE Powertrain Plant Model Development

The UW team utilized an ICE powertrain model that was donated by dSPACE as part of their Automotive Simulation Models. The ICE model was parameterized specifically to match the B20 engine used by the team, and was thus left completely unchanged. The transmission and transmission controller were also used from dSPACE. The available gear ratios were confirmed to be the expected gear ratios.

4.2 - Validation of Plant Models and Vehicle Testing

Over the course of Year 3, a great deal of effort was put into validating the plant models used throughout the competition. Many of the more sophisticated CS strategies rely heavily on accurate maps of engine emissions and fuel consumption, so a particular emphasis was placed upon validating those maps during the official Emissions Testing Event at ANL. The fuel consumption values taken from the dynamometer test of the vehicle was compared to the fuel consumption values extrapolated from the engine efficiency map used in the model. These two plots were compared with one another to validate the fuel consumption map used in the model.

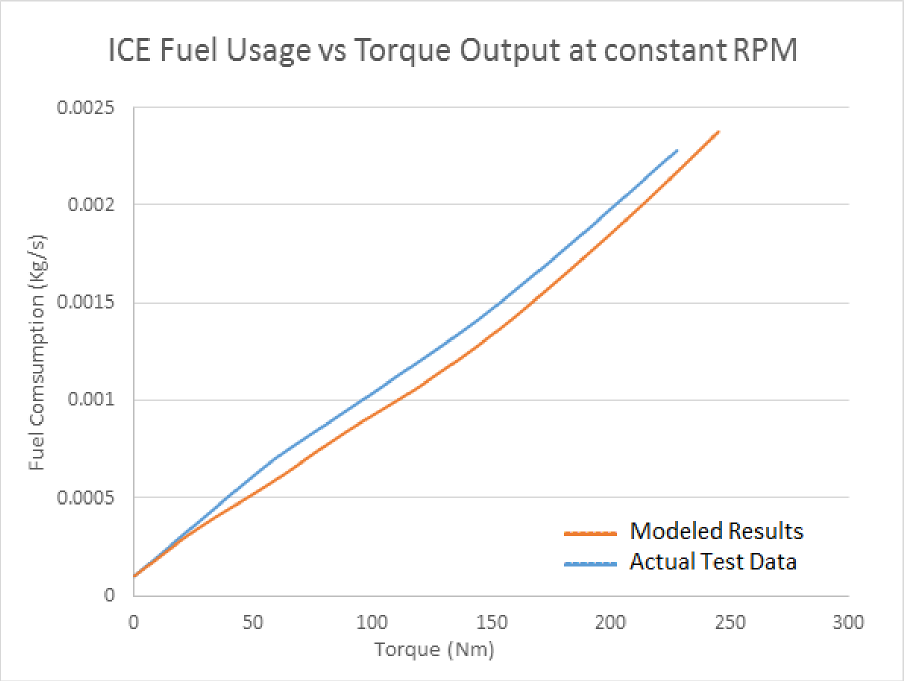


Figure 21: ICE Fuel Consumption Model Validation

4.2.1 - ESS Plant Model Validation

ESS current and voltage data from dynamic drive testing was used as the primary validation for the ESS plant model. To conduct this validation testing, current data from a dynamic drive was fed into the parameterized Simulink DP model. The model’s calculation of the resulting voltage was then compared to the real-world values obtained during testing. The results of this validation testing can be found in Figure 22, which plots the ESS Voltage for both the Model and real world Vehicle results over a dynamic driving scenario.

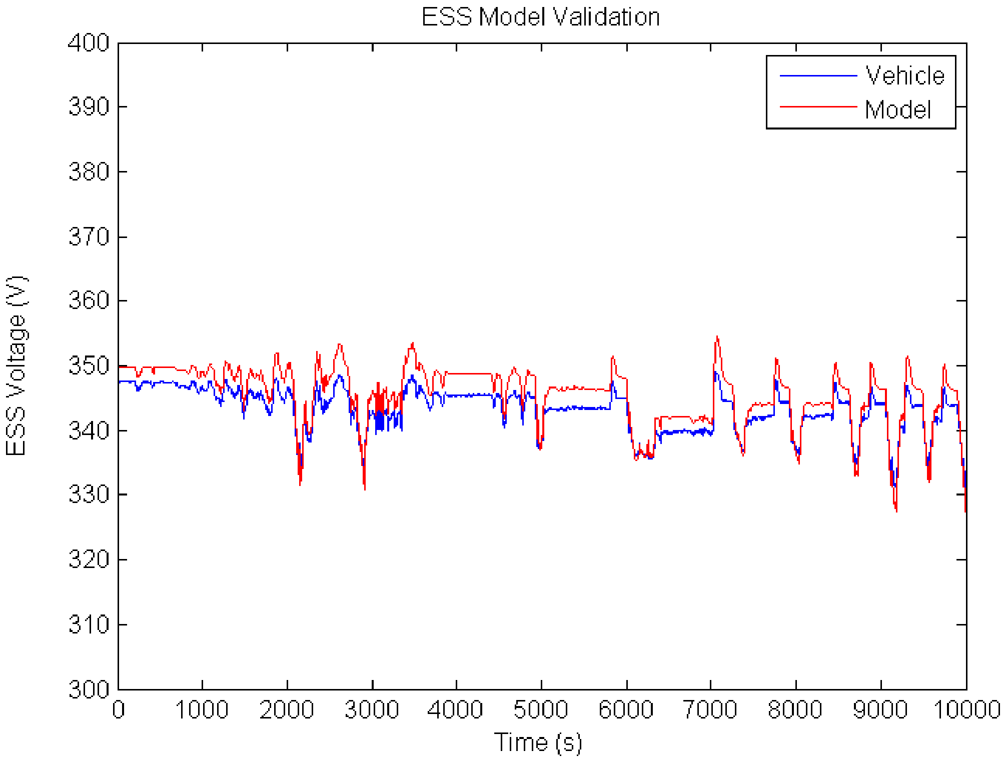


Figure 22: Isolated ESS Model Validation Results

It can be seen from the testing results that the DP model resulted in highly accurate simulation of the ESS behavior on the drive. The model's ESS voltage is slightly offset by approximately 3 V, which has since been corrected with an adjustment to one of the parameters. It can also be seen that the vehicle ESS experienced smaller variations in voltage over the test cycle. This could potentially be due to the model not including the additional bus capacitance provided by the inverter and other components.

4.3 - Performance Testing, Validation, and Results

4.3.1 - Vehicle Validation Plan

The team's vehicle validation plan involved two primary validation areas, the first ensuring that the vehicle supervisory controller passes all of the requirements described in the team's control requirement spreadsheet. This spreadsheet includes all of the faults and modes for each component and how the supervisory controller should react in response to different situations. This step in the validation plan required extensive SIL and HIL testing since it was often necessary to induce simulated fault modes to adequately test that these requirements were being met. This HSC and plant model validation process was described briefly in the previous sections, and in more detail in the team's recently released Modeling and Simulation White Paper. The HSC and Plant Model Validation Plan Table from that paper is displayed in the Appendices [8].

The next phase in the validation plan involved developing test procedures that would validate the overall vehicle's ability to meet the VTS targets set by the team. Not only would this testing process validate that the vehicle reaches the level of performance that was modeled, but it also would demonstrate that the vehicle is able to complete and excel in each of the dynamic events during the Year 3 competition. The majority of this initial validation testing was planned for execution during ETE at ANL, where the team had access to state-of-the-art dynamometer testing facilities that allowed for highly controlled, repeatable test scenarios. In addition, many of the VTS line item validation tests were developed to closely match the procedure and conditions that would be present during Year 3 Competition. Static line items like cargo space, ground clearance, and vehicle mass were tested using the same tools that would be used for competition. While at ETE, the UW team confirmed that the vehicle was able to meet cargo space and ground clearance requirements without penalties, and was also very close to the estimated vehicle mass VTS.

For VTS line items that required dynamic testing, the team developed a variety of procedures that could be executed during the eight hours of allotted dynamometer testing time. The next sections detail the test procedure used and results of testing to validate acceleration VTS targets, in addition to an overview of testing used to validate energy consumption VTS goals.

4.3.2 - Acceleration Test Procedure

While at Argonne National Labs (ANL), both of the acceleration VTS line items where tested. This was done by setting the dynamometer to torque mode with dynamometer road load coefficients matching those determined by ANL using the stock Malibu Eco. While these values will differ slightly from the UW Malibu due to the addition of the rear powertrain and extra vehicle mass, it was decided that these values would be adequate for initial validation testing since the performance mode has yet to be precisely calibrated.

The test was conducted after a series of E&EC warm-up cycles were run to test various engine warm-up strategies. In addition to providing valuable data on ICE startup behavior, the E&EC cycles served to warm both the tires and engine to operating temperature before conducting the acceleration test, similar to the warm-up laps that will be conducted at Year 3

Acceleration and Braking testing. At this point, the HSC was manually switched into Performance Mode, enabling full torque requests to be sent to both eSystem and iceSystem powertrains.

While logging the vehicle data both from the dynamometer side and the vehicles internal CAN networks, the driver started from a stop and accelerated to 65 mph, then reduced speed to 45 mph before and accelerating up to 75 mph. This procedure is similar to how the acceleration event is completed during year end competition. There are some slight differences in the test procedure used and dynamometer environment compared to the on-road competition event. Most notable are the differing road-load amounts and the slicker surface of the dynamometer wheel drums. In addition, the driver at ETE did not start with his foot on the brake pedal to allow the engine to reach optimum RPM levels before launching. As such, it is predicted that the acceleration results from this testing are likely slightly slower than those that will be seen at Year 3 Competition.

4.3.3 - Acceleration Testing Results

Using the vehicle's speedometer that was broadcast over the communication network (shown in black) and the driver's accelerator pedal position signal (shown in grey), the following two graphs where created. The first displays the 0-60 test velocity profile, and the second displays the 50-70 test velocity profile.

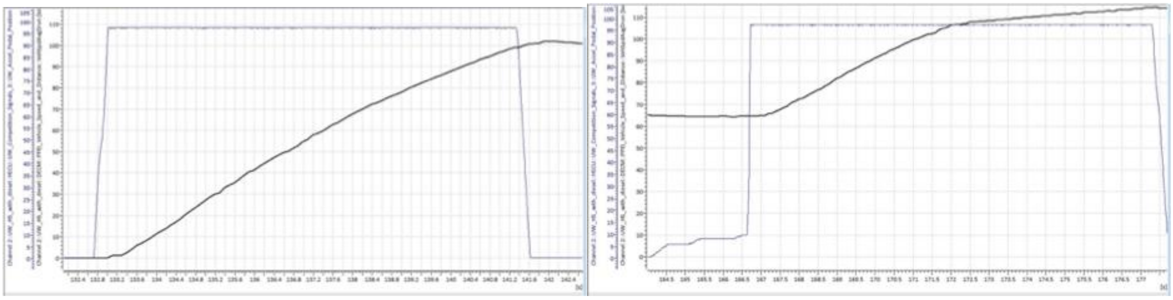


Figure 23. 0-60 and 50-70 Velocity Profiles

For the first test the team calculated the time difference between initial vehicle movement (IVM) and the vehicle reaching 60 mph. IVM was considered to be the first data point with a greater than zero value. For the 50-70 mph test the team subtracted the time at 70 mph from the time at 50 mph. The results of this testing are given in the table below, with the target VTS values from Year 3's Progress Report 2 and the finalized values from Year 3's PR3 that were selected based on the ETE results.

Table 4. ETE Acceleration Testing Results

Test Specification	PR2 Target VTS	ETE Test Result	PR3 Finalized VTS
Acceleration 0-60 mph	6.8s	7.4s	7.0s
Acceleration 50-70 mph	4.5s	3.8s	3.4s

As the table indicates, the ETE testing revealed that the UW vehicle would likely not be able to meet the original VTS target for 0-60 mph time. This was primarily caused by the discovery that the inverter was limiting the torque output of the motor to 250 Nm rather than the modeled 311 Nm. While this discrepancy is unfortunate, after several conversations with the component manufacturer it was decided that the internal inverter values would be left stock to ensure that the system would continue to work reliably through Year 3 competition. As such, the decision was made to increase the final VTS target for 0-60 to 7.0s. This value was selected to be lower than the ETE result due to the excessive amount of front tire slip experienced during the test. By conducting a similar test on actual pavement and adding in brake-launching by the driver, the team is hoping to meet the stated 7.0s 0-60mph time within the 5% error bounds required to avoid a VTS penalty. This bound allows for a maximum of 7.35s at Year 3 competition. For the 50-70 time, the finalized VTS was selected to compensate for an error in the internal programming of the inverter. The inverter was incorrectly set to decrease torque output to zero once a certain speed threshold was reached, but unfortunately for ETE that value was set to approximately 65mph. That value has now been modified so that the vehicle will be able to obtain the 3.4s VTS value at competition.

In addition to the acceleration testing, the team also conducted a number of tests in order to validate VTS items focusing on fuel and energy consumption. These investigations are not presented in detail in this report due to both their more complex nature and lack of completely reliable results. However, the results were used in the final selection of VTS numbers, so a cursory explanation is presented as follows.

Since most of the VTS items are tested during the E&EC event at competition, this testing was conducted using the same drive cycles that would be used during event itself. In addition, the dynamometer road load coefficients were adjusted to reflect the addition of the emissions measurement trailer load. For the test itself, the vehicle was driven over the E&EC drive cycle in both CD and CS modes. Since the CS mode had not yet been refined at ETE, a small parameter sweep was used to investigate the effect of varying the degree of load shifting used during CS. The best results from these tests were used to generate the values for range, CD and CS energy consumption values. The following chapter outlines the process used to select control system parameters and strategies to best ensure success during the E&EC event.

Chapter 5 - Emissions and Energy Consumption Testing Analysis

5.1 - E&EC On-Road Test Description

The single more impactful event during the EcoCAR 2 competition is Emissions and Energy Consumption (E&EC), where on-road testing is conducted over a pre-set drive cycle to determine the vehicle's total fuel and grid electricity consumption along with the tailpipe emissions generated over an approximately 103 mile drive. This section outlines the event description found in the EcoCAR 2 Year 3 rules [9] so that the engine start-stop analysis and SOC bound selection sections make more sense in the context of the testing procedure used.

5.1.1 - Event Preparation

The 103 mile E&EC Event consists of a number of driving and charging portions. Before entering the event, vehicles must be fully charged and filled with fuel, at which point the fuel tank is weighed in order to calculate the change in fuel mass consumed over the drive. Once the vehicle is fully charged and fueled, a trailer is installed which carries a portable emissions gas analyzer. At this point, the vehicle is pushed to the starting line and the driving portion can begin.

5.1.2 - E&EC On-Road Drive Testing

The E&EC driving portion consists of driving the vehicles around a circular test track with designated velocities at different locations on the track. Drivers execute the following procedure for the event:

1. Drive to circle track (Figure 24)
2. 3 City/Highway (C/H) Cycle repetitions around circle track (Figure 25)
3. 20 min break with key in Off position
4. 4 C/H repetitions around circle track
5. Drive back to garage (Figure 27) and begin grid charging

For the To Track and From Track portions of the event, it should be noted that the actual measured trace may vary from the target traces shown below, since the roads to and from the circle track are trafficked by a variety of test vehicles. However, these portions together account for approximately only 4.1% of the overall distance travelled, so the majority of the test occurs on the more controllable C/H portion which takes place on the circle track itself.

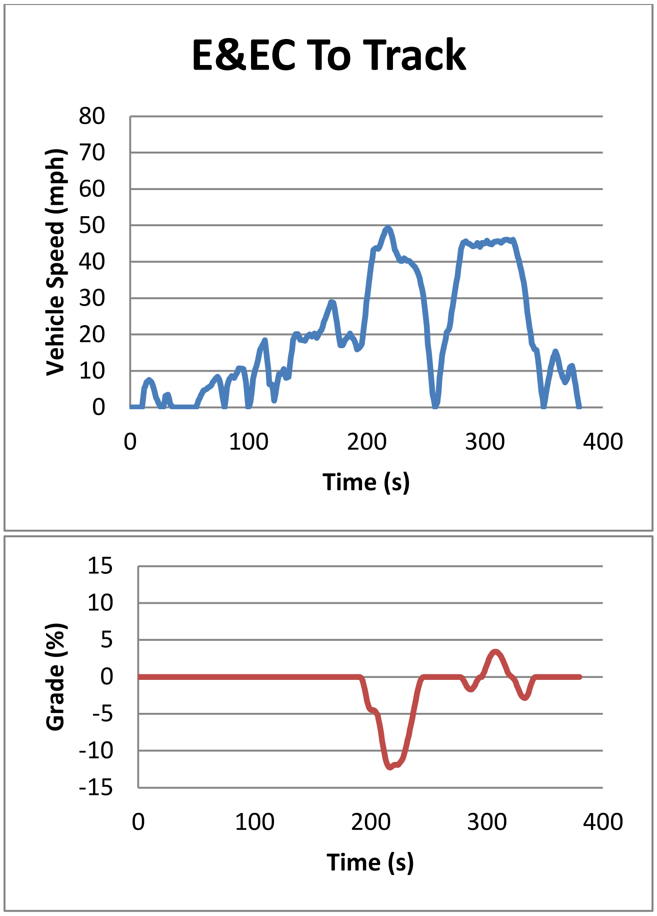


Figure 24. E&EC speed and grade traces for To Track portion

Upon inspecting the To Track segment, it can be seen that there is a significant and extended downhill grade portion, reaching over -12% grade at near 50 mph. Since this downhill grade occurs near the beginning of the cycle when the SOC is high, special care must be taken in developing the regenerative braking strategy to not overcharge the ESS.

Once the vehicle arrives at the circle track, the driver begins to follow the C/H speed trace around the circle track. The trace for this cycle is shown in Figure 25. The circle track is designed to maintain a 0% grade for the entirety of the track.

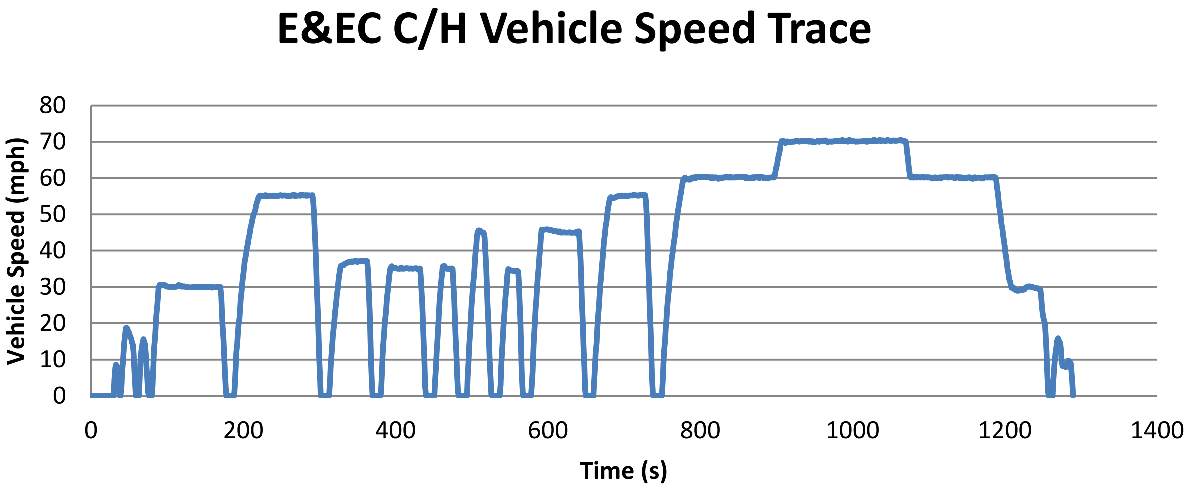


Figure 25. E&EC C/H speed trace

There are a number of interesting items to note on the C/H cycle. The first is that the cycle in general consists of repeated acceleration portions of varying intensities, followed by steady-speed portions at various speeds. Each of these steady-speed portions is followed by a braking event that brings the vehicle to a stop for ten seconds. The average deceleration rate for each of the braking events is shown at the bottom of Figure 26. These values were calculated starting with the first event greater than -0.3 mph/sec and averaging the deceleration rate until the vehicle stops. It can be seen that most of the post steady-speed braking events in the middle of the cycle are between 4.3 and 5.5 mph/sec. This repeatability of braking to a ten second stop from steady speeds enables the start-stop fuel consumption analysis in the following sections

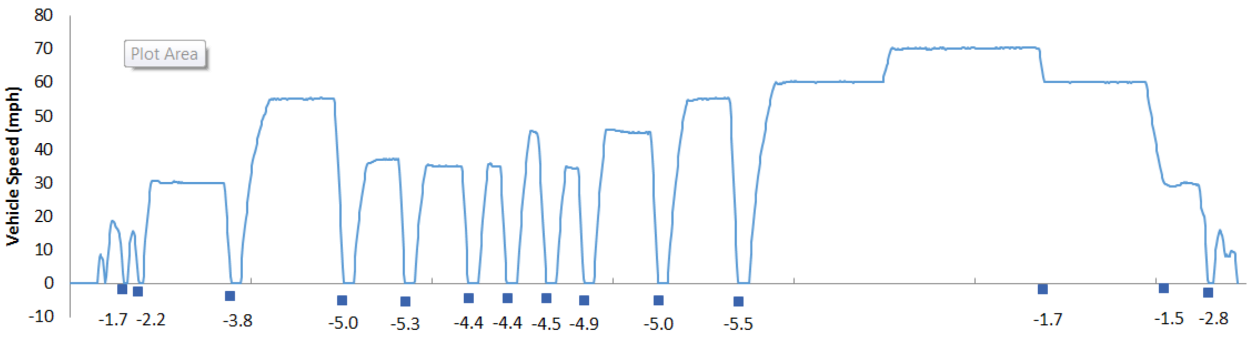
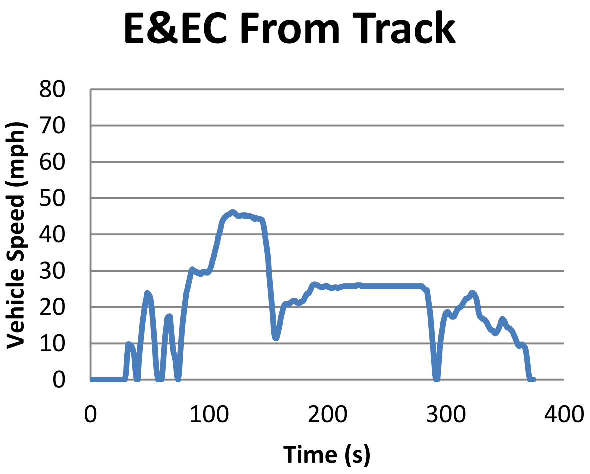


Figure 26. E&EC braking intensities (values in mph/sec)

One final item of note is that the cycle only significantly exceeds 60 mph in one segment of the cycle. This 70 mph portion is followed by a steady-speed 60 mph portion, after which the driver slows to 30 mph, exits the circle track, and drives to the inner portion where the vehicle idles until the cycle repeats or the From Track portion begins (if all C/H sections have been completed).

The speed and grade traces for the final portion of the E&EC event are shown in Figure 24. It is worthwhile to note that there are some significant uphill grades encountered on the From Track segment, meaning that it is important for the vehicle's control strategy to maintain an adequate SOC level to make it back from the event.



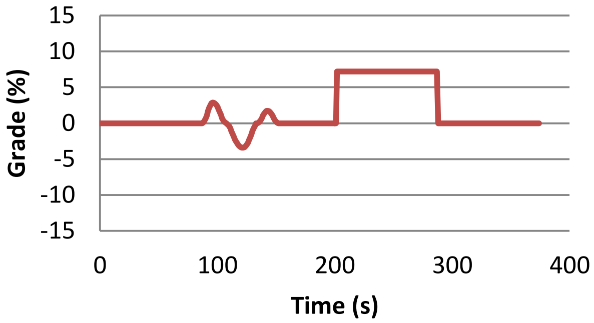


Figure 27. E&EC speed and grade traces for From Track portion

The following sections analyze the effectiveness of various control system strategies and parameters on the E&EC on-road testing results.

5.2 - Setting SOC Limits

Setting the lower threshold SOC for transitioning to CS mode was a key element of the team's success in Year 3 competition due to the scoring breakdown for the E&EC event. In the context of energy consumption and criteria emissions, it was most beneficial to obtain the largest CD range possible with the implemented hardware while still ensuring that the vehicle could successfully complete both the E&EC event and all other dynamic testing events [10]. To find out the CD range of the vehicle with various SOC lower limits, dynamometer testing at ANL was performed to approximate the EV range of the vehicle at various lower SOC limits.

5.2.1 - EV Range Testing on E&EC

For this testing, the vehicle was driven over the To Track and first portion of the E&EC C/H in CD mode on the dynamometer with road load coefficients set to emulate the load of both the vehicle and emissions collection trailer. The original test procedure called for two repetitions of the C/H in order to average the ΔSOC over the two cycles. However, due to a malfunctioning cell in the ESS of the test vehicle the pack could not deliver the expected current capabilities ESS system and would flag a fault under heavy loading below 65% SOC. This problem was rectified by replacing the cell before competition, but unfortunately for the test results this meant that only a portion of the To Track and C/H could be completed. This data was still quite valuable for estimating the vehicle EV range and ESS currents for the testing however. The results of this testing is shown in the figure below.

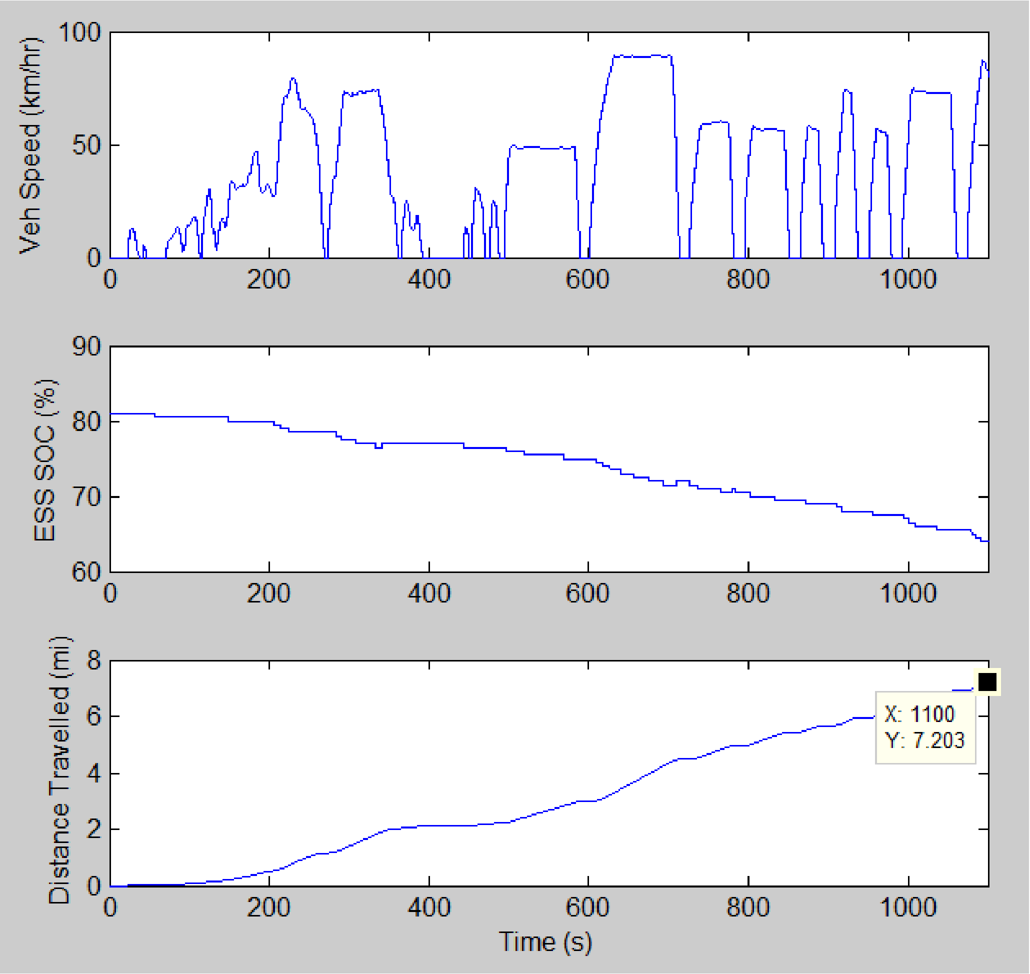


Figure 28: E&EC To Track and C/H EV Range Test Results

Using this data, it was possible to estimate the vehicle's EV range by calculating the overall electrical energy consumption per mile experienced during the test. The data from the E&EC portions tested showed that the UW EcoCAR 2 travelled 7.2 miles with a ΔSOC of 17%, leading to an electrical energy consumption of 2.4% SOC/mi. Since the vehicle used an 18.9 kWh ESS, this is approximately equivalent to 0.45 kWh/mi of battery electrical energy consumption. This value is lower than the grid electrical energy consumption results that are collected during E&EC for scoring purposes, but can be used to find the vehicle's EV range as a function of the CS transition %SOC. For example, if the CS transition threshold was to be set at 10%, the vehicle would be able to travel approximately 38.1 miles in EV mode before turning on the engine and transitioning to CS mode. A list of these predicted EV range values is given in Table 5.

Table 5: Vehicle Range for Charge Sustaining Transition Points

CS Transition (%)	EV Range (mi)
0	42.4
5	40.3
10	38.1
15	36.0
20	33.9

One very important item to note is that the 20 minute key-off break occurs after the completion of the To Track portion followed by 3 repetitions of the C/H. This portion of the drive is approximately 44.3 miles, meaning that it is not possible to complete the first segment of driving and reach the key-off event without turning on the engine. It is possible that optimization of control logic behavior could potentially result in some increases in EV range, and in fact during Year 3 competition the more refined controller was able to achieve a vehicle range of 39 miles using a CS Transition value of 10%. This value was set in response to specifications from the manufacturer that begin to limit the current output of the ESS below that threshold. In order to validate that the ESS SOC could drop below 10% without causing an ESS fault condition, the currents out of the ESS were analyzed over the course of the drive to find out whether the 10 sec and 60 sec rolling average current values would ever exceed the limits at low SOC. These plots of the To Track and the first portion of the C/H testing are shown in the following figures.

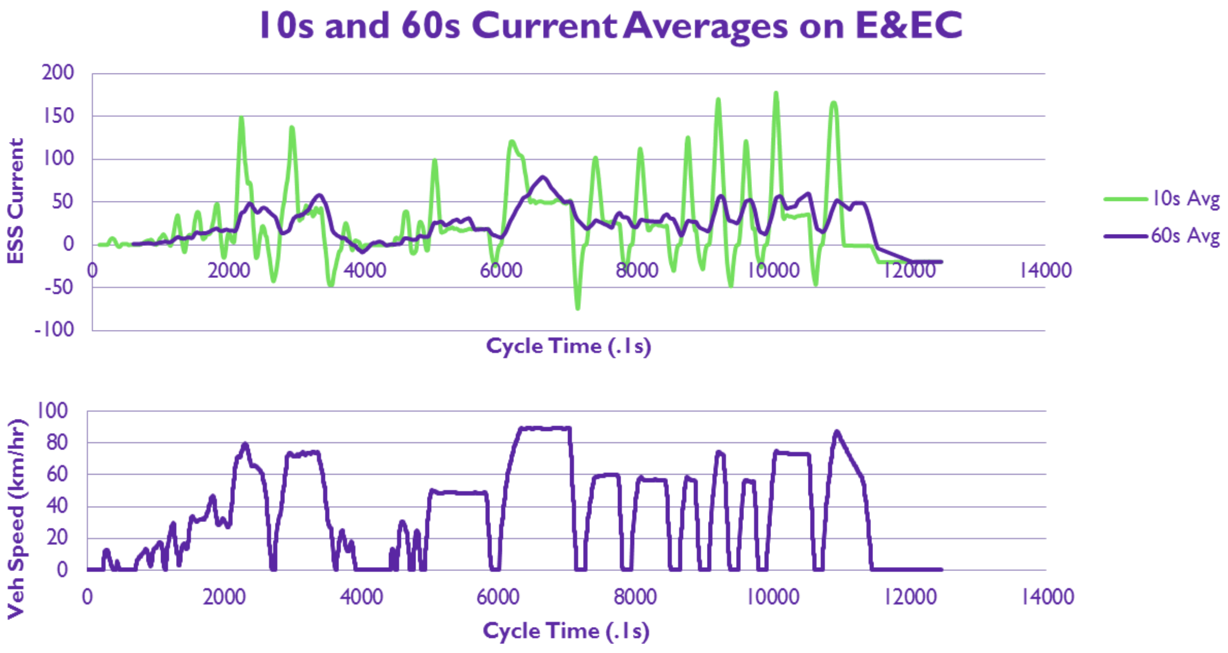


Figure 29. Plots showing rolling average ESS current over To Track and C/H cycles

Since the maximum ESS current seen over the course of the cycle was less than half that of the limit at 10%, it was estimated that the CS transition of 10% could be used safely. Based purely on the results of the test, it may even be possible to select a lower transition %. However, since the malfunctioning cell in the ESS prevented the vehicle from completing the full highway portion of the C/H cycle, it was decided that the 10% transition would be used. This allowed adequate time for the engine to warm up before the 20 minute key-off, preventing a full engine cold start after the 20 minute break while at the same time providing a large EV range.

5.3 - Deceleration and Engine Idle Stop Analysis

One of the more prevalent techniques for reducing fuel consumption in production vehicles is to stop the engine when the vehicle is idling and restart it again once the driver is ready to accelerate away from the stop. Idle stop is often used in hybrid vehicle applications, though some conventional models with 12V starters have begun utilizing the strategy as well. A study from Argonne National Labs quantified the benefits of start stop strategies using four test vehicles, running both US and European certification cycles on a vehicle dynamometer with stop start enabled then and disabled for each vehicle. Their testing showed a significantly greater fuel consumption impact from stop start on the European NEDC cycle versus the US UDDS cycle, with an average start stop fuel consumption improvement of 4% on the UDDS and 10% on the NEDC for all the test vehicles. This difference is caused by the significantly higher time spent at a stop in the NEDC cycle, with 30.6% vehicle stop versus 17.6% on the UDDS [11].

This result means that the choice of whether to implement stop start is highly dependent on the test cycle used to validate the vehicle. The following sections attempt to quantify the fuel savings that could be achieved on the E&EC cycle using engine stop start. There are a few important points to note before beginning the analysis. The first is that the UW EcoCAR 2 uses a B20 engine with a standard 12V starter motor. Many hybrid applications use a high voltage starter and alternator motor in order to increase the efficiency of each stop start operation.

The next consideration is that many newer vehicles are equipped with deceleration fuel cutoff in order to cut fuel to the engine while going down a hill or coasting to a stop. The following analysis investigates the both the fuel consumption used by the UW vehicle's engine during a hot start and the effect of deceleration fuel cutoff on the E&EC cycle. In order to most accurately quantify the effects of implementing the stop start strategy outlined, it would be advisable to run the vehicle on a dynamometer over the same C/H with stop start enabled and then disabled over the course of multiple test runs. However, at the time of the ANL dynamometer testing the vehicle's stop start strategy had not been implemented and refined. This analysis was thus performed in an attempt to estimate the improvement that would be generated by implementing a stop start strategy by analyzing test data collected over several E&EC C/H tests. In addition, the analysis was performed in order to better understand the behavior and consumption of the particular engine used in the UW EcoCAR 2, since the configuration and ECU programming of the ICE system was significantly different from the diesel engine's stock vehicle application.

5.3.1 - Deceleration Fuel Cutoff Effects

The first set of testing was conducted during the aforementioned vehicle dynamometer testing at ANL to first determine if the engine possessed fuel cutoff on deceleration and then quantify the effects of that fuel cutoff. Depending on both the presence of and the effectiveness of deceleration fuel cutoff, this may affect the ideal stop start behavior; because the E&EC cycle features many extended decelerations to a stop, it is possible that a significant fuel savings can be made by recognizing these deceleration events and proactively shutting down the engine and shifting the transmission to neutral. Note that this engine shutoff on deceleration would only be possible because the rear traction motor is powerful enough to drive the vehicle without the engine on, so that the vehicle can respond quickly if the engine shuts off in anticipation of a stop but the driver then requires a positive acceleration maneuver.

The first portion of the analysis involved determining if deceleration fuel cutoff actually exists. To test this capability, the vehicle was driven on a dynamometer up to 55 mph, then the vehicle was slowed down at a constant rate to a stop. Data collected for vehicle speed, instantaneous fuel consumption, and brake pedal position during this maneuver is shown in Figure 30.

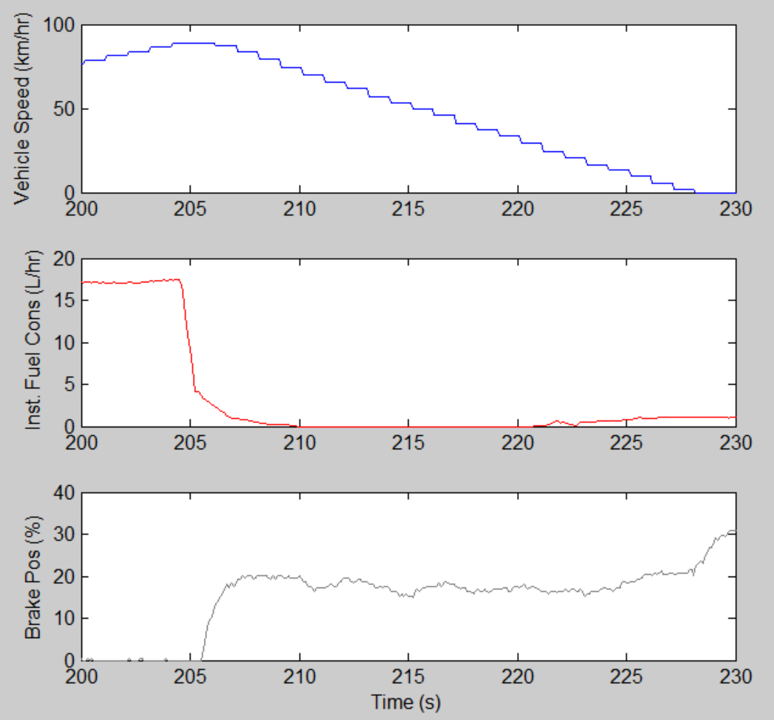


Figure 30: Engine Fuel Cutoff during deceleration

The data collected confirms that the engine does indeed possess deceleration fuel cutoff. At around the 208 second mark, the engine fuel usage begins reducing to zero at the 210 second mark. The engine then uses no fuel until around 220 seconds or about 15 mph, at which point the tires are no longer spinning at a sufficient rate to keep the engine from stalling and fuel injection begins again.

It is important to realize that the fuel cutoff behavior was identified during a very extended braking maneuver. The E&EC C/H braking events consist of significantly more aggressive decelerations. Therefore, the vehicle was also tested on the dynamometer over the E&EC C/H with a test mass and vehicle dynamometer coefficients matching the values seen during testing with the emissions collection trailer. This data from this test for vehicle speed and instantaneous fuel consumption is summarized in the following figure, with each braking to a stop event highlighted by an arrow.

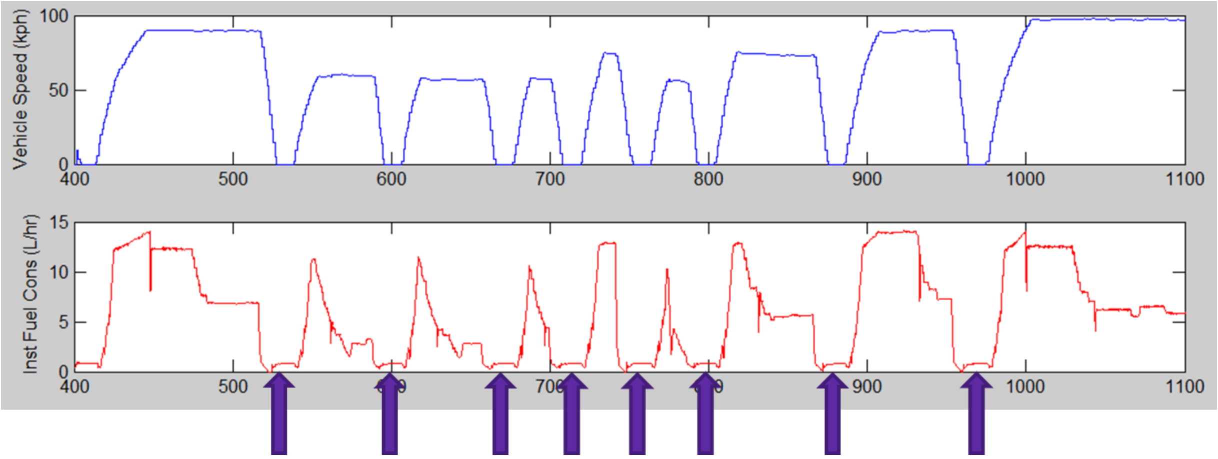


Figure 31: Braking Events Fuel Cutoff during E&EC Cycle

It can be seen in the figure that the E&EC braking events are often too short for fuel consumption to reach zero. Even the ones that exhibit the cutoff behavior do not activate soon enough to be at zero fuel for more than 1-2 seconds. This means that there could be significant fuel savings introduced by having the supervisory controller turn off the engine once an extended braking event is detected, and waiting until a threshold velocity is reached to turn back on the engine. In the meantime, torque requests would be achieved using the RTM.

5.3.2 - Calculating Fuel Used During Brake to a Stop and Idle

In order to determine if this strategy would actually be worthwhile, the next step in the analysis is to find the fuel used during each braking to a stop event using the non stop start strategy versus turning off the engine for the deceleration and turning it back on after a threshold velocity is reached. To do this, the data from E&EC dynamometer testing was analyzed to isolate each braking to a 10 second stop event on the C/H and determine the amount of fuel that was used. Figure 32 displays the data from one of these braking to a stop events.

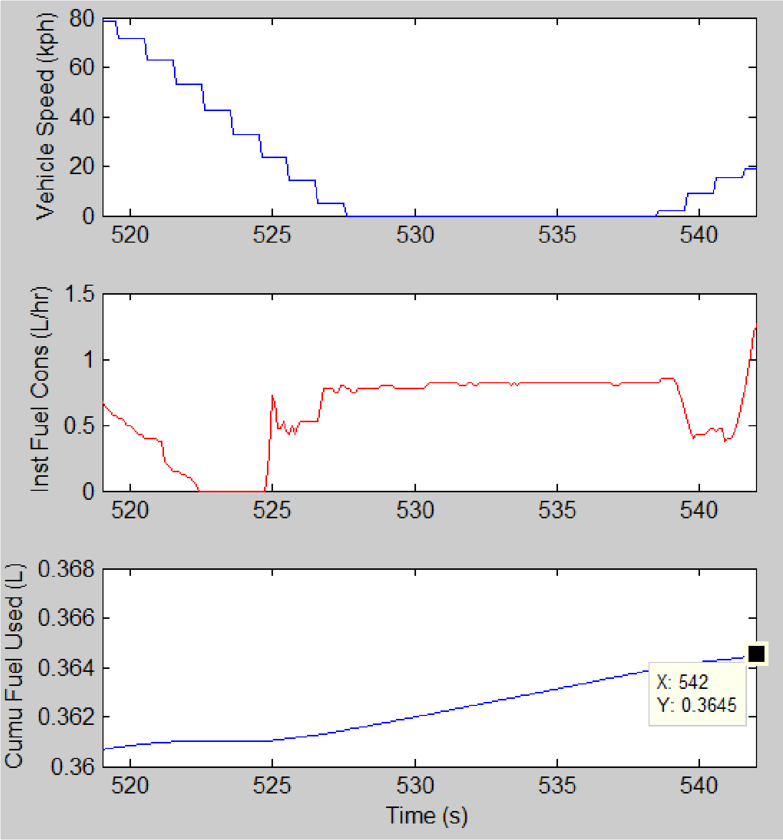


Figure 32: Effects of Normal Braking to Engine Idle Stop

During each event, the engine on average consumed .0038 L of B20. This value was calculated by integrating the fuel used from 2 seconds after negative acceleration began up to the point the vehicle speed increased above zero. This means that the total fuel used during the C/H during the nine braking to a stop and ten second idle events was .0342 L. The nine braking to a stop events that were analyzed are shown in the following figure.

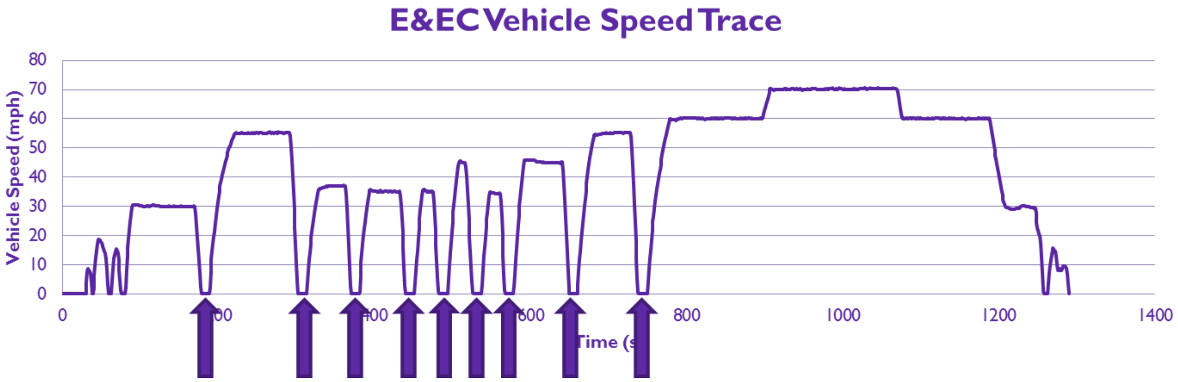


Figure 33: Deceleration to a stop instances on the E&EC Cycle

This amount does not yet account for fuel used in other portions of the C/H. In addition, each time the engine is started it is cranked using the 12V starter motor, using a significant amount of energy in the process. In order to analyze the effectiveness of the stop start strategy, it is also necessary to quantify the fuel used to drive the alternator at a higher duty cycle and compensate for this drop in 12V battery SOC. To do this, the following section looks at the additional fuel used during an engine start compared to normal vehicle idling.

5.3.3 - Calculating Additional Fuel Used for Engine Start

In order to quantify the fuel consumed during an engine start versus normal idling, several engine hot starts were investigated which occurred over the course of the ANL dynamometer testing. One of these starts is shown in the figure below, which displays plots of data collected for engine speed, instantaneous fuel consumption, and cumulative fuel used during an engine stop start event.

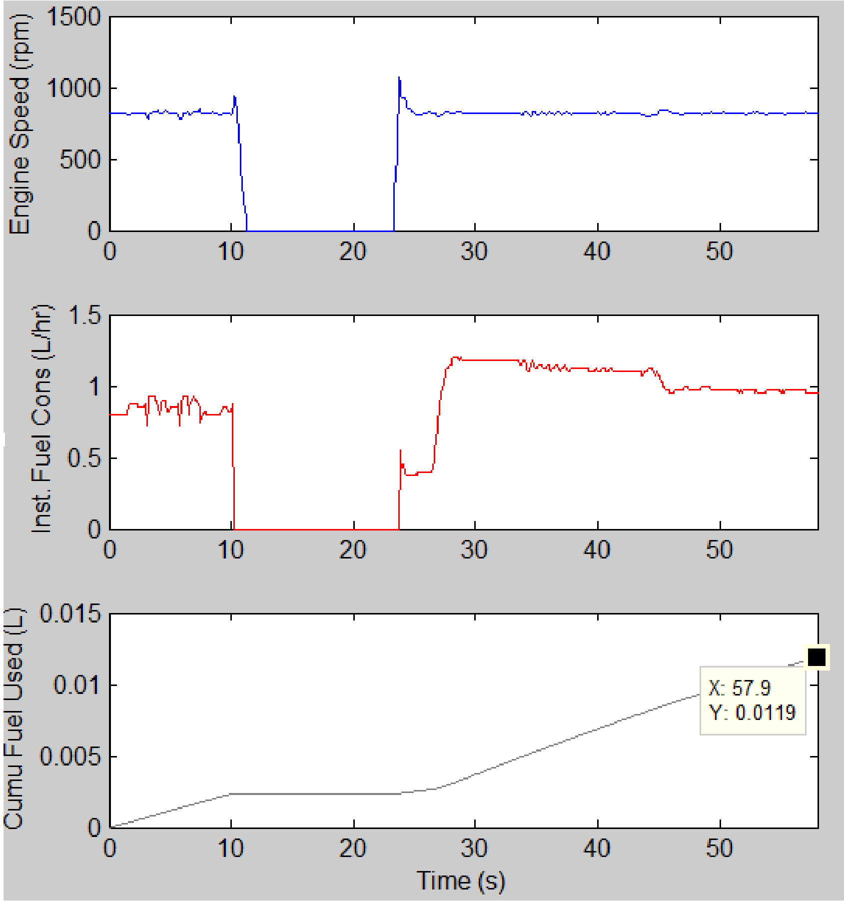


Figure 34: Fuel Usage for Engine Start

It can be seen in the red plot on Figure 34 that the fuel usage during normal idle (seconds 0-10), is lower than that used directly after the engine is restarted (seconds 26-45). The average idle fuel consumption was sampled at various times during the dyno testing, resulting in an average idle fuel consumption of approximately 0.9 L/hr. This value was then used to find the total extra fuel used in the engine start process, or EngStartDeltaFuel, when compared to standard idling. This was done according to the equation below, by using the trapz function in MATLAB to integrate the instantaneous fuel consumption from the second plot in Figure 34 over the time period from 24 to 47 seconds, when the fuel consumption had returned to normal idle levels. This value is notated as EngStartCumuFuel the following equation. The average consumption value from earlier was then multiplied by the delta time used in the trapz function to find the volume of fuel that would have been used if the engine had been idling for the entire time of the engine start. The difference between these two values is shown in the following equation as EngStartDeltaFuel, representing the increase in volume of fuel used as a result of starting the engine rather than idling it for the same period of time.

$$\boldsymbol{EngStartDeltaFuel = EngStartCumuFuel - AvgIdleCons * \Delta t} \qquad \textbf{(Equation 1)}$$

This calculation was performed over two engine hot starts during the ANL dyno testing, resulting in an engine start penalty of .0011 L of extra fuel used by the engine. Note that methodology in the following calculations assumes that the engine start penalty when the engine is started and allowed to idle is the same as when the engine is quickly ramped up after the starting procedure is complete. It would be advantageous to conduct E&EC C/H dynamometer testing with stop start enabled and then disabled to more accurately assess the impact of the strategy once it is successfully implemented on the test vehicle.

To find the total fuel savings that stop start could potentially provide, the total fuel used over each of the nine braking to a stop events in the C/H was calculated from the test data, resulting a total fuel volume of .0342 L. When the nine engine start penalties are subtracted from this volume, the resulting total potential fuel savings due to implementing a stop start scheme was .0246 L. Since the total fuel used over the C/H was 1.883 L, the implementation of engine stop start within 2 seconds of deceleration would result in a 1.30% fuel savings on each cycle. While this value is not insignificant, it is very important to note that engine stop start also impacts emissions in addition to fuel consumption. Since 9 stop starts are required to achieve this 1.3% fuel savings, it is possible that implementing a stop start scheme specifically for the nine braking to a stop events is not worth the savings in fuel when factors such as emissions, drivability, reliability, and noise, vibration, and harshness are taken into account. However, the preceding analysis did not take into account the beginning and ending portions of the E&EC cycle highlighted in green in Figure 35.

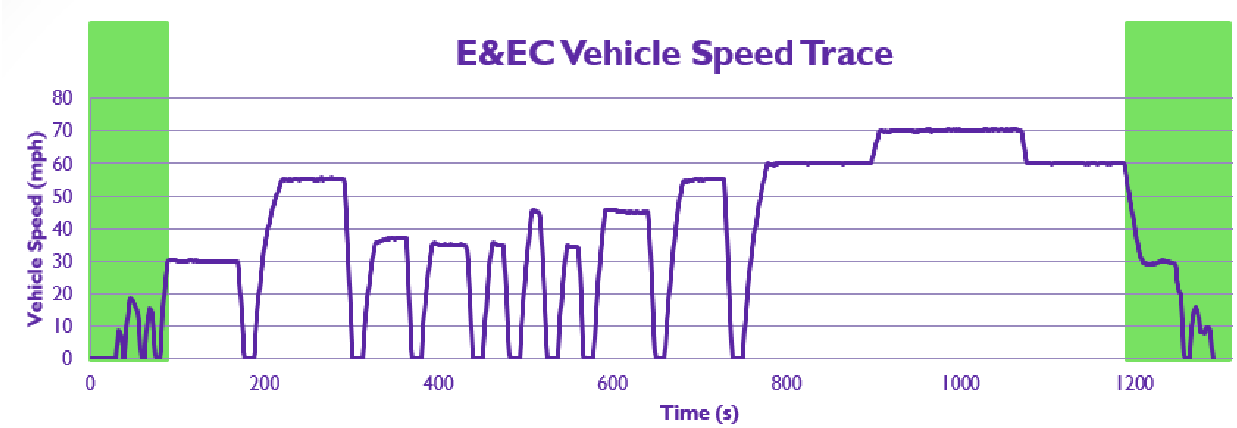


Figure 35: Opportunity for stop start at beginning and end of E&EC cycle

It can be seen in the figure that these sections feature an extended period of time where the engine would not be used and the RTM would be delivering all of the torque request. In the control strategy used on the test vehicle, the engine does not deliver any of the driver torque request until the vehicle exceeds 25 mph, unless the SOC drops to very low levels. This means that the engine could potentially be shut off 2 seconds after the extended deceleration begins at around 1200 seconds, and left off until approximately 100 seconds when the vehicle exceeds 25 mph again. This would result in only one engine start penalty for a total idle time of 190 seconds, which means that with only one engine start a total of .0464 L of fuel could be saved for a savings of 2.47% over one C/H.

To implement this type of strategy, it would be necessary to monitor the vehicle speed and trigger an engine stop event each time the driver applies the brake pedal for greater than two seconds after the vehicle has been traveling near 60 mph for more than 30 seconds. While it is unlikely that this exact scenario would occur in real-world testing by consumers, similar behavior could potentially be seen when a driver exits the freeway and navigates a neighborhood back to their residence. In addition, by using the RTM as the sole source of proportion for the final part of the drive home, it would be possible to use up any charge needed to reach a desirably low SOC before plugging in the vehicle for the night. Finally, this strategy would reduce emissions and traffic noise in neighborhood areas.

One caveat on this strategy is that the vehicle must shut off its engine with enough SOC to reach the primary residence successfully without over-depleting the ESS. Since this varies be customer, some sort of learning algorithm would need to be applied that tracks driver behavior with location data in an attempt to trigger this returning home mode in the most efficient and reliable way possible. In addition, if the SOC does deplete too much and an additional engine start is needed, the engine and exhaust may have cooled for long enough that any catalytic converters or other emissions control components may have dropped below operating temperature. This would result in unfavorable emissions for the final portion of the drive, and as such should be avoided if at all possible.

Conclusions

Over the course of the controller development and dynamic vehicle testing process, several points and conclusions have emerged.

It is highly important to structure the architecture of any Simulink-based supervisory controller such that parallel code development can be successfully performed. This also aids in implementing version control.

Correct parameterization of component models can result in increased efficiency of HSC algorithm development. These parameters should be tested as soon as possible using the actual test vehicle in order to validate the assumptions used in the creation of each model.

Selecting the CS transition SOC can have an immense impact on the overall vehicle energy consumption depending on the test procedure used. This selection is not a simple process, and relies on many assumptions which should also be validated in real-world testing.

Engine stop start can have a significant impact on CS mode fuel consumption for diesel engines, even with a 12V starter instead of a high voltage BAS or other starting system. In the case of the E&EC C/H cycle, stop start has the potential to reduce fuel consumption by approximately 3.8% depending on the implementation and strategy used. It is important however to also consider the emissions impact of each implementation, since longer idle times would result in greater consumption gains but may cause exhaust system cooling to the point where catalytic converters and other emissions control systems cannot function.

References

[1] S. J. Boyd, "Hybrid Vehicle Control Strategy Based on Power Loss Calculations," Blacksburg, Virginia, 2006.

[2] D. K. Mehr, M. Michalak, S. Erlien and G. R. Bower, "Optimization and Testing of a Through the Road Parallel, Hybrid-Electric, Crossover Sports Utility Vehicle," Detroit, MI, 2009.

[3] J. C. Mathews, K. J. Walp and G. M. Molen, "Development and Implementation of a Control System for a Parallel Hybrid Powertrain," in *VPPC*, United States, 2006.

[4] T. Crain, T. Fayer, B. Fabien and P. Reinhall, "Structuring a Hybrid Vehicle Supervisory Control System Simulink Model for Simpler Version Control with Multiple Software Developers," in *SAE Technical Paper 2014-01-1923*, 2014.

[5] T. Crain, J. Wilke, B. Boyer and T. Fayer, "Powertrain Integration and Controls Development Process for a Parallel Through the Road Plug-in Hybrid Electric Vehicle," in *SAE Technical Paper 2017-01-1917*, 2014.

[6] H. He, R. Xiong and J. Fan, "Evaluation of Lithium-Ion Battery Equivalent Circuit Models for State of Charge Estimation by an Experimental Approach," *National Laboratory for Electric Vehicles, Beijing Institute of Technology,* pp. 582-598, 2011.

[7] Idaho National Engineering and Environmental Laboratory, "PNGV Battery Test Manual," Department of Energy, 2001.

[8] T. Crain and R. Mallory, "University of Washington Modeling and Simulation White Paper," 2014.

[9] "EcoCAR 2 Year 3 Rules," 2014.

[10] T. Fayer, T. Crain, B. Fabien and P. and Reinhall, "The Importance of Maximizing Grid Electricity Usage in the Component Selection and Design of a Midsize PHEV," in *SAE Technical Paper 2013-01-0548*, 2013.

[11] H. Lohse-Busch, "APRF/AVTA Idle Stop Vehicle Testing," August 2011. [Online]. Available: http://www.transportation.anl.gov/D3/reports/ANL_APRF_IdleStopvehicleStudy_August2011.pdf.