2022-2026 EcoCAR EV Challenge

Year 4 PCM Development Challenge

Propulsion Controls & Modeling (PCM) | September 2025

McMaster EcoCAR (MAC): McMaster University, Hamilton, ON, Canada

Table of Contents

1. Introduction

Modern electric vehicle propulsion controllers include an electronic transmission range selector (ETRS) feature, which is responsible for interpreting the driver's selected gear position (such as Park, Reverse, Neutral, or Drive) and coordinating it with the vehicle's operating conditions to ensure proper torque delivery and driveline engagement. In this challenge, you will have the opportunity to implement logical and temporal assessments of predetermined tests, based on the provided requirements, to track, test, and verify the functionality of this feature against a reference model developed by our team.

1.1. ETRS Feature Overview

The vehicle utilizes the Electronic Transmission Range Selection (ETRS) as the range select. Ranges are what the driver selects – Park, Reverse, Neutral, and Drive. Gears are the configuration of the transmission or drive unit. For example, 1st, 2nd, or 3rd gear in an automatic transmission. So, this section will discuss range changes rather than shifts. There is no mechanical connection between the shifter and the drive units, which is different than a "traditional" shift system where there is a physical cable connecting shifter to transmission. The system is further simplified in the vehicle because there are no mechanical differences between Drive, Neutral, and Reverse within the drive unit, as the motor can spin in either direction. So, the only physical actuation of the ETRS system occurs when shifting in and out of park.

Figure 1: Cadillac LYRIQ ETRS Shifter Stock

The vehicle is calibrated so that the parking brake will automatically engage upon driver range change to Park. The parking brake will also automatically disengage when the driver makes a range change to Drive, Reverse, or Neutral from Park. The system shall not send Park Position until the parking brake is engaged. In the time between the request for park from the vehicle/driver and the actual engagement of the parking brake the system shall send Neutral Position on **TransmissionPosition**. This also applies in the opposite direction. The system shall not send Drive Position or Reverse Position until the parking brake is disengaged, and the vehicle system is ready to provide torque. In the time between the range change request from the vehicle/driver and the disengagement of the parking brake, the system shall send Neutral Position. Additionally, all range state transitions shall pass through the neutral range state.

1.2. Provided Information

For this challenge, you will be provided with a couple of different resources that you need to use. First and foremost, you will be provided with all requirements that you are required to work with titled **DevChallengeETRSReqs.slreqx**. For those who are interested, these requirements have been extracted from the **"ETRS"** section in the document. Additionally, you will also be provided with a preconfigured model input file titled **ETRSInputs.mat** so that every test and corresponding requirement can be evaluated. Furthermore, a PDF document outlining how to use the required Simulink tools will also be provided, this can also be found in the repository you have access to.

You will be provided with a model that represents the ETRS feature, this model is called **DevChallengeETRS.slx.** You will also be provided with the Test Suite that organizes all the tests that you are responsible for executing titled **ETRSTests.mldatx**.

1.2.1. Provided ETRS Requirements

1. The PSC shall command "**Transmission Actual Range**" to replicate the highest priority range change request from the vehicle when the corresponding conditions are met
    a. If "**Electronic Transmission Range Primary**" indicates PARK or "**Charging System Shift to Park Request**" is True and "**EPB Application Status**" is True and "**EPB Availability Status**" is Available and **"Vehicle Speed"** is less than 0.5, "**Transmission Actual Range**" shall indicate PARK
        i. When "**Charging System Shift to Park Request**" is True and "**Transmission Actual Range**" is PARK, the PSC shall keep "**Transmission Actual Range**" in PARK
    b. If "**Electronic Transmission Range Primary**" indicates REVERSE and "**EPB Application Status**" is False and the Propulsion system is Active, "**Transmission Actual Range**" shall indicate REVERSE
    c. If "**Electronic Transmission Range Primary**" indicates NEUTRAL, "**Transmission Actual Range**" shall indicate NEUTRAL
    d. If "**Electronic Transmission Range Primary**" indicates DRIVE and "**EPB Application Status**" is False AND the Propulsion system is Active, "**Transmission Actual Range**" shall indicate DRIVE
2. The PSC shall indicate NEUTRAL via "**Transmission Actual Range**" when the parking brake (EPB) is not in the correct status for the requested gear
    a. If "**Electronic Transmission Range Primary**" indicates DRIVE/REVERSE and the EPB is not disengaged, the PSC shall indicate NEUTRAL
    b. If "**Electronic Transmission Range Primary**" indicates PARK or "**Charging System Shift to Park Request**" is True and the EPB is not engaged, the PSC shall indicate NEUTRAL
3. The PSC shall not allow unsafe gear changes while the vehicle is in motion
    a. The PSC shall only allow a shift to PARK if the vehicle speed is below 0.5 km/h
    b. When the vehicle is in motion above 0.5 km/h, the PSC shall not allow a shift to the propulsion mode which does not correspond to the vehicle's direction (i.e., it would not allow shifting to reverse while the vehicle is moving forward)
4. The PSC shall require the brake to be pressed for gear changes that could otherwise cause harm or significantly damage the vehicle
    a. The PSC shall not allow a shift out of PARK if the brake is not pressed

1.2.2. Preconfigured Model Input File Details

The test suite that is provided titled **ETRSTests.mldatx** will contain tests that are based on all requirements that are also outlined in the file titled **DevChallengeETRSReqs.slreqx**. Another file that was mentioned previously, the **ETRSInputs.mat** file is a MATLAB inputs file that contains a mapping of all the inputs of the ETRS model so that you can drive each input correctly.

1.2.3. Provided Model

Below is the model that you will be interfacing with while doing this challenge. As shown, there is an input layer to the left which is what your Model Input file should be mapped to. In the middle is the control logic, this is the section of the model that contains the functional logic of the system and is where you will be implementing your requirements. To the right is the output layer, this section simply contains the outputs generated from the control logic upon input reception.

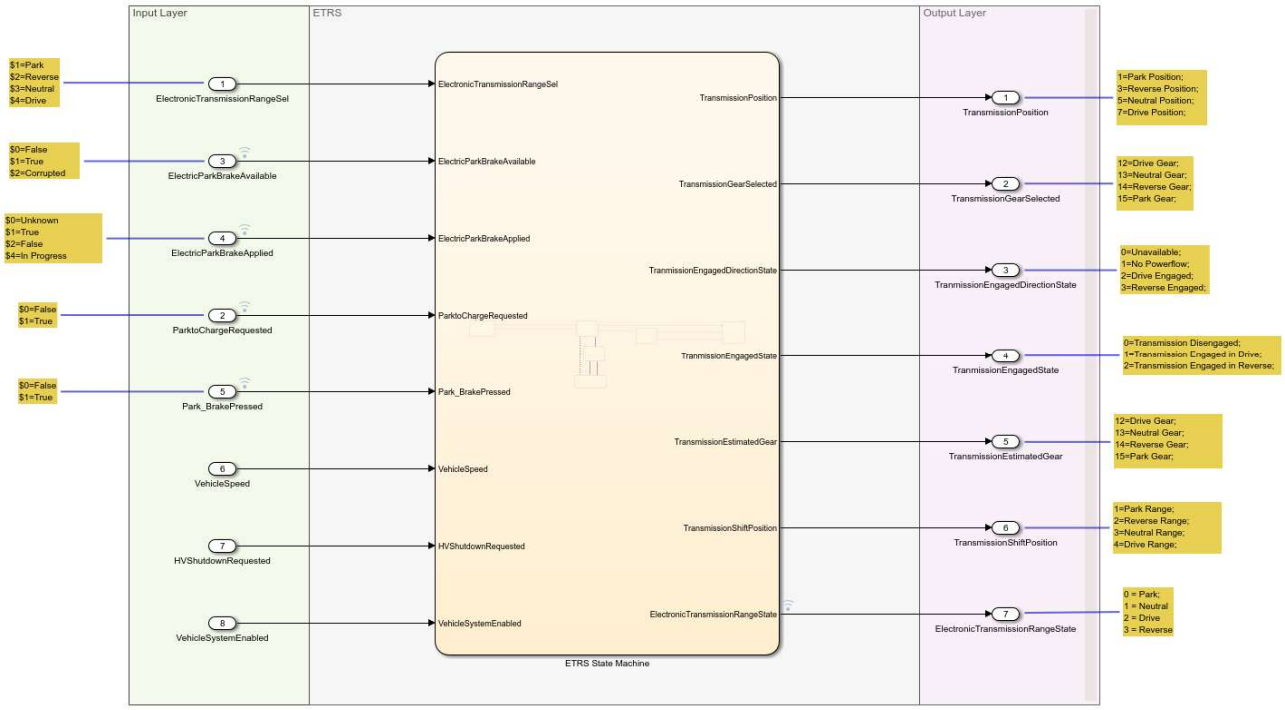

Figure 2: Simulink Model Overview

1.2.4. Understanding the Stateflow

Stateflow is used to define operational states of a system. In a physical system like a Range Selector in a vehicle (the gear shifter), each drive mode is considered a *state*. Park, Reverse, Neutral, and Drive are all operating states, which may or may not change what value is output for each output signal from the Stateflow. A list of all output values is shown in Table 1: Stateflow Output Variables and Their Values.

Table 1: Stateflow Output Variables and Their Values

State	Output Actions
Park	TransmissionPosition = 1; TransmissionShiftPosition = 1; TransmissionEstimatedGear = 15; TransmissionGearSelected = 15; TranmissionEngagedState = 0; TranmissionEngagedDirectionState = 1; ElectronicTransmissionRangeState = 0;
Neutral	TransmissionPosition = 5; TransmissionShiftPosition = 3; TransmissionEstimatedGear = 13; TransmissionGearSelected = 13; TranmissionEngagedState = 0; TranmissionEngagedDirectionState = 1; ElectronicTransmissionRangeState = 1;
Drive	TransmissionPosition = 7; TransmissionShiftPosition = 4; TransmissionEstimatedGear = 12; TransmissionGearSelected = 12; TranmissionEngagedState = 1; TranmissionEngagedDirectionState = 2; ElectronicTransmissionRangeState = 2;
Neutral_Intermediate_Drive	TransmissionPosition = 5; TransmissionShiftPosition = 3; TransmissionEstimatedGear = 13; TransmissionGearSelected = 13; TranmissionEngagedState = 0; TranmissionEngagedDirectionState = 1; ElectronicTransmissionRangeState = 4
Neutral_Intermediate_Reverse	TransmissionPosition = 5; TransmissionShiftPosition = 3; TransmissionEstimatedGear = 13; TransmissionGearSelected = 13; TranmissionEngagedState = 0; TranmissionEngagedDirectionState = 1; ElectronicTransmissionRangeState = 5;
Reverse	TransmissionPosition = 3; TransmissionShiftPosition = 2; TransmissionEstimatedGear = 14; TransmissionGearSelected = 14; TranmissionEngagedState = 2; TranmissionEngagedDirectionState = 3; ElectronicTransmissionRangeState = 3;

In order to change between states, specific conditions must be met. These conditions act as a set of logical checks to ensure that the incorrect state is not entered by mistake. In terms of the ETRS, it ensures that the vehicle is in *PARK, REVERSE, NEUTRAL,* or *DRIVE* only when it is safe to be. Figure 3: Stateflow Chart with Highlighted Transition Condition shows the ETRS Stateflow, with one transition condition highlighted for the transition between the *NEUTRAL* state and the *REVERSE* state.

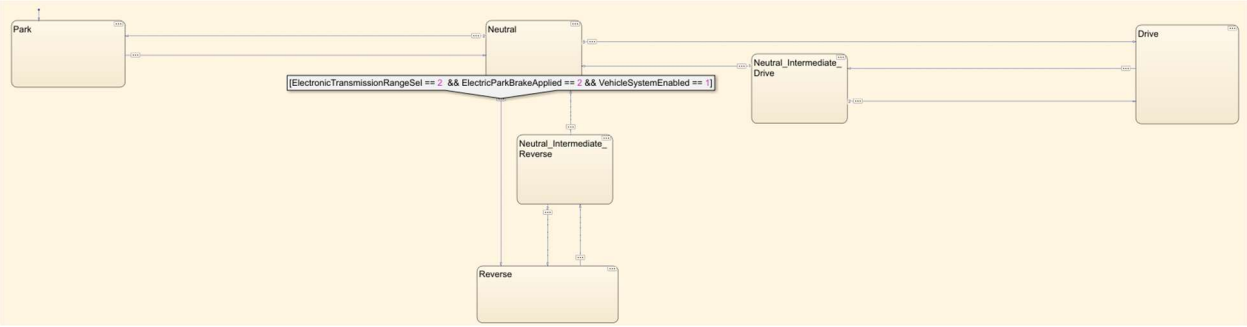

Figure 3: Stateflow Chart with Highlighted Transition Condition

Any time that a transition condition is wrapped in square brackets '[]', this can be read as an 'if' statement. When analyzing the condition in Figure 3, it would be read as "If **ElectronicTransmissionRangeSel** is equal to 2 AND **ElectricParkBrakeApplies** is equal to 2 AND **VehicleSystemEnabled** is equal to 1, transition from *NEUTRAL* to *REVERSE*. A full list of transition conditions for each state is shown in

Table 2: ETRS Stateflow States, Transition Conditions, and Destinations

Source	Destination	Condition
Neutral	Park	[(ElectronicTransmissionRangeSel == 1 || ParktoChargeRequested == 1) && ElectricParkBrakeAvailable == 1 && VehicleSpeed < 0.5 && ElectricParkBrakeApplied == 1]
Neutral	Drive	[ElectronicTransmissionRangeSel == 4 && ElectricParkBrakeApplied == 2 && VehicleSystemEnabled == 1]
Park	Neutral	[((ElectronicTransmissionRangeSel ~= 1 && ElectronicTransmissionRangeSel ~= 0) || (ElectricParkBrakeApplied == 2 || ElectricParkBrakeApplied == 4)) && ParktoChargeRequested == 0 && VehicleSpeed < 0.5 && Park_BrakePressed == 1]
Neutral_Intermediate_Drive	Neutral	[VehicleSpeed < 0.5]
Drive	Neutral_Intermediate_Drive	[((ElectronicTransmissionRangeSel ~= 4 || ParktoChargeRequested == 1 || ElectricParkBrakeApplied == 1) || HVShutdownRequested == 1)]
Neutral	Reverse	[ElectronicTransmissionRangeSel == 2 && ElectricParkBrakeApplied == 2 && VehicleSystemEnabled == 1]
Neutral_Intermediate_Reverse	Neutral	[VehicleSpeed < 0.5]
Neutral_Intermediate_Drive	Drive	[ElectronicTransmissionRangeSel == 4 && ParktoChargeRequested == 0 && ElectricParkBrakeApplied == 2 && VehicleSystemEnabled == 1]
Neutral_Intermediate_Reverse	Reverse	[ElectronicTransmissionRangeSel == 2 && ParktoChargeRequested == 0 && ElectricParkBrakeApplied == 2 && VehicleSystemEnabled == 1]
Reverse	Neutral_Intermediate_Reverse	[(ElectronicTransmissionRangeSel ~= 2 || ParktoChargeRequested == 1 || ElectricParkBrakeApplied == 1) || HVShutdownRequested == 1]

2. Development Challenge Task

2.1. Getting Started

To complete the development challenge, you will need to install **MATLAB R2023a** with **ALL ADD-ONS.** Be sure that your add-on list includes Simulink Test and Requirements Toolbox, which is where you will develop test cases for the feature and incorporate your requirements within the test suite. A detailed explanation of the Simulink Test Manager and the Requirements Toolbox used for this challenge can be found in the included guide **Using the Requirements and Simulink Test Toolboxes.pdf**.

2.1.1. Loading the MATLAB Project

To ensure that all files are loaded correctly, it is crucial that you load the file **DevChallenge2025.prj** by double clicking on the file within the MATLAB navigator. This is a MATLAB project file, which is used to add all the necessary files to your path. Failing to do so will result in your tests failing to work correctly. When loaded correctly, your MATLAB window should look similar to the image below.

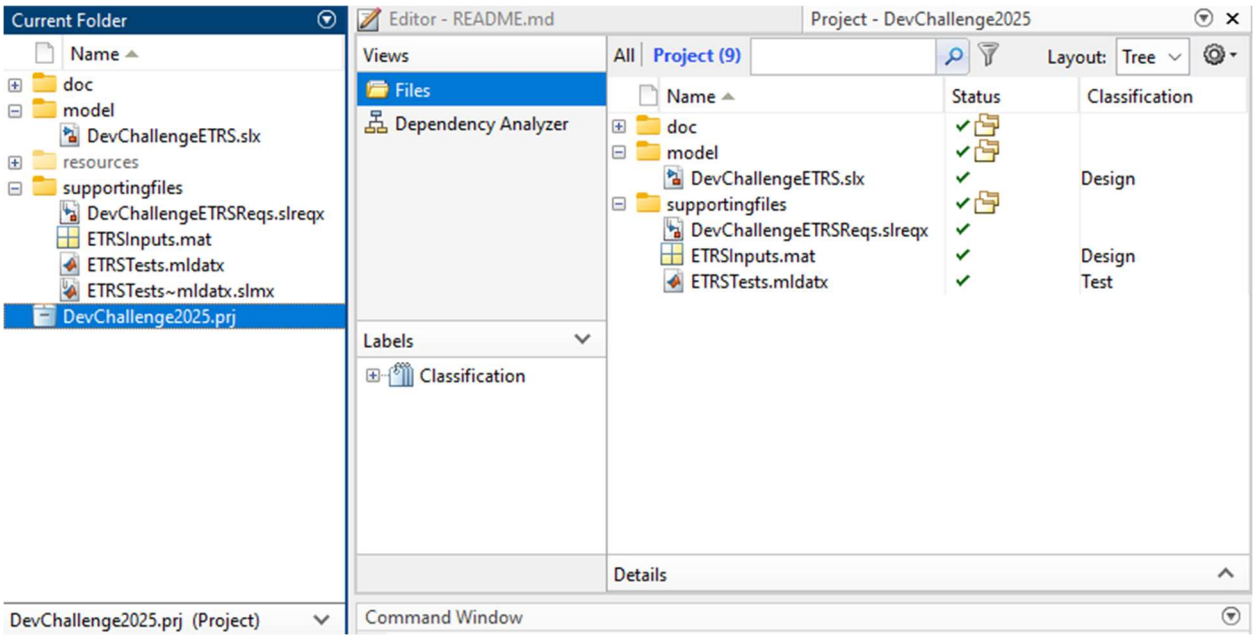

Figure 4: MATLAB Project Window View

Once loaded, the test suite file titled **ETRSTests.mldatx** can also be loaded (by double clicking on it within MATLAB). This is the primary location where the dev-challenge task work will take place.

2.2. Model Review

To successfully modify the provided test suite to implement the logical and temporal assessments it is crucial to understand the provided model first. The model is split into 3 areas, the *Input Layer*, the *ETRS* and the *Output Layer*. The **ETRS Feature Overview** and **Provided Information** sections outline the logic of the ETRS feature; understanding the logic within the Stateflow block is crucial to successfully implementing the logical and temporal assessments for this assignment.

2.3. Requirements Suite Usage

As previously mentioned, a MATLAB requirements file titled **DevChallengeETRSReqs.slreqx** is provided. This file is to be used as a reference point for your development of the logical and temporal assessments.

2.3.1. System Under Test Settings

Your system under test settings should be configured to use the **DevChallengeETRS.slx** model and should have a stop time of 35 seconds. Please refer to the included **Using the Requirements and Simulink Test Toolboxes.pdf** guide which outlines how to do this.

2.3.2. Test Case Inputs

Within the "Inputs" dropdown in the requirements tests settings, ensure to add the provided file as an external input so that preconfigured inputs mentioned above are mapped to your model. Without doing this, you will not be able to run any of your tests.

To do this correctly, hit the **"+add"** button below the external inputs area and navigate to the provided file on your directory. Once the provided file is selected hit the **"Map Inputs"** button when in the **"Block Name"** mapping mode. Beneath are a couple of screenshots that visualize this process in order. Note that in this example the file name is **ModelInputs.mat**, this is not the naming convention that is being used for this year's challenge, note the difference when looking for the correct file.

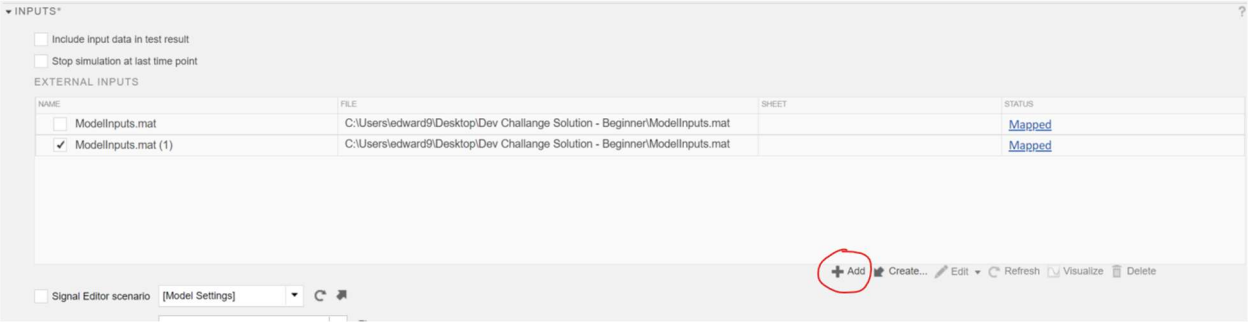

Figure 5: Simulink Test Manager - Inputs Dropdown

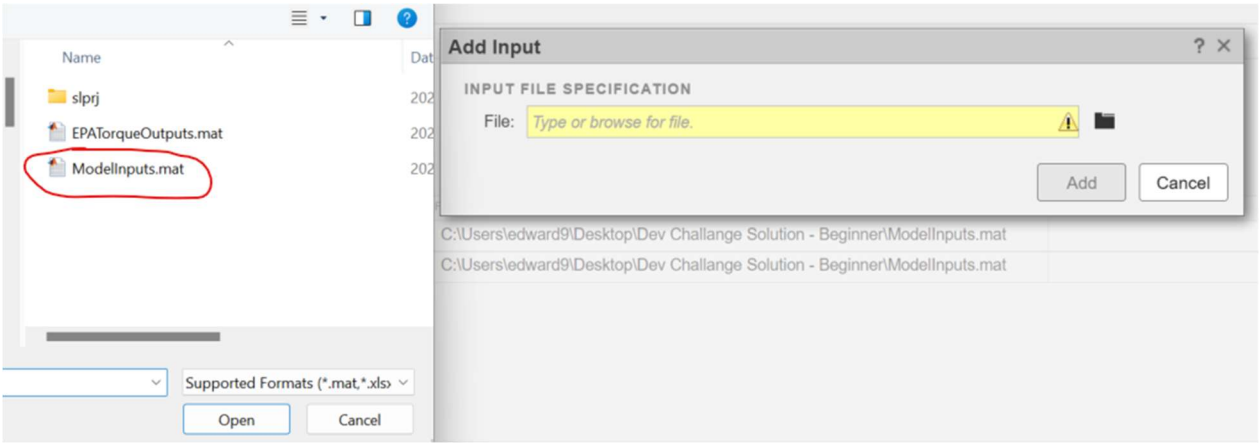

Figure 6: Simulink Test Manager - Adding Input Sets

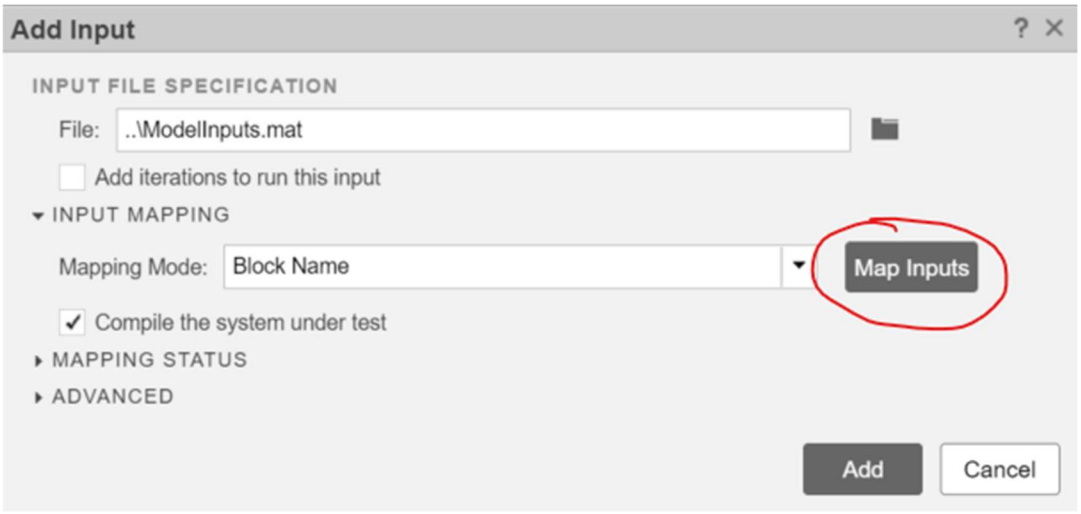

Figure 7: Simulink Test Manager - Mapping Input Sets

2.3.3. Test Suite Baseline Criteria

For your requirements tests, you can skip adding baseline data. It will not be necessary for the development challenge.

2.3.4. Logical and Temporal Assessments for your Tests

This is where the bulk of the work is to be done, you are responsible for editing the signals that are being driven in the input (.mat) file to test each of the requirements that are outlined, so if a requirement outlines how the ETRS Stateflow should enter the "*Drive*" state, you must open the Stateflow and after understanding the logic involved with how the "Drive" state is entered, drive the inputs in the .mat file to trigger the Stateflow to execute the behavior required to enter the "*Drive*" state. In summary, your goal is to prove that each of the requirements align with the behavior that is implemented in the provided model of the ETRS feature. You edit the .mat file by clicking on the "*Edit*" button that is beside the "*Create*" button inside the Inputs dropdown menu where you would have originally added your .mat file as an external input.

In layman's terms, we are handing you code that we know works and asking that you create a "catch" system that constantly checks if certain behaviours that we have implemented in the code occur. An example of this style of logic would be, if variable X is equal to 3 and variable Y is equal to 7, then I expect variable Z to be equal to 5 if the first two conditions are met at the same time. Be sure to allow a buffer of around 10-20 milliseconds delay for each logical and temporal assessment.

3. Development Challenge Advanced Task

For the advanced version of this challenge, the base version must be completed. You are then required to prove that the control logic will not exhibit incorrect behavior if given bad input data. For example, if only 2 out of 3 conditions are met to move to a given state within the Stateflow chart, your test should validate that the state in question is never entered.

Copy the existing input set and give the file a unique name. Create a new test within the test suite for each new test being conducted. You must develop these *"failure mode"* test cases within the logical and temporal assessment section of the Test Manager for at least 5 requirements. You will need to modify the

copied input set to induce scenarios in which not all conditions are satisfied. Please feel free to reach out to our team if you are unsure of how to proceed with the input modification.

3.1.1. Final Submission/Presentation

Each development challenge team is required to prepare a final submission and presentation. The final submission will be submitted after the dev challenge has concluded. All teams will be required to attend the development challenge day at McMaster Automotive Resource Centre (MARC), where you will present a science fair-style presentation of your work and results. This is an informal presentation, where you will walk the evaluators through your development process, results, challenges, etc.

4. Final Submission

All teams are required to create a test report from the results of your testing. To do this correctly, you will need to run the entire test suite, as opposed to running each individual test case one at a time. To do so, select the test suite from the test browser and press the run button in the top banner.

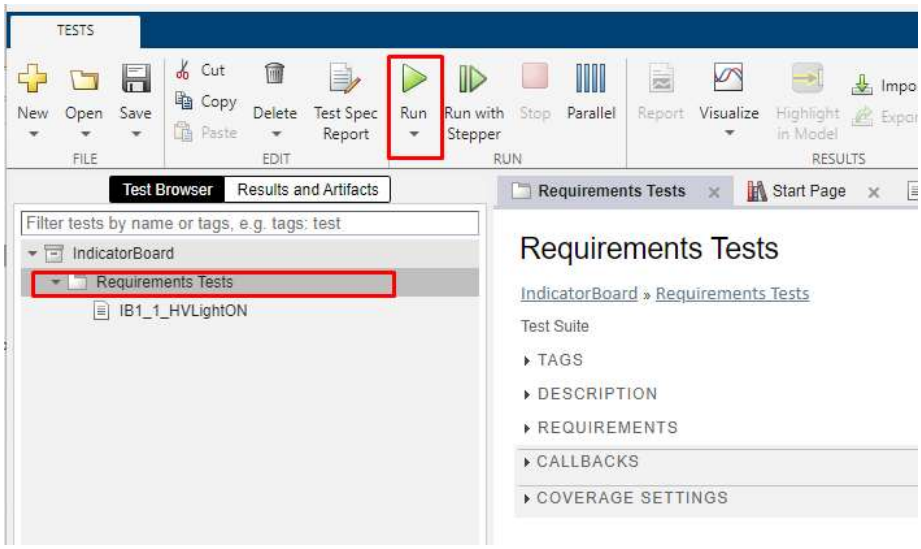

Figure 8: Simulink Test Manager - Running The Test Suite

Once all tests have completed, navigate to the Results and Artifacts tab of the left navigation panel. Right-click the Result and select the '*Create Report*' option, shown below:

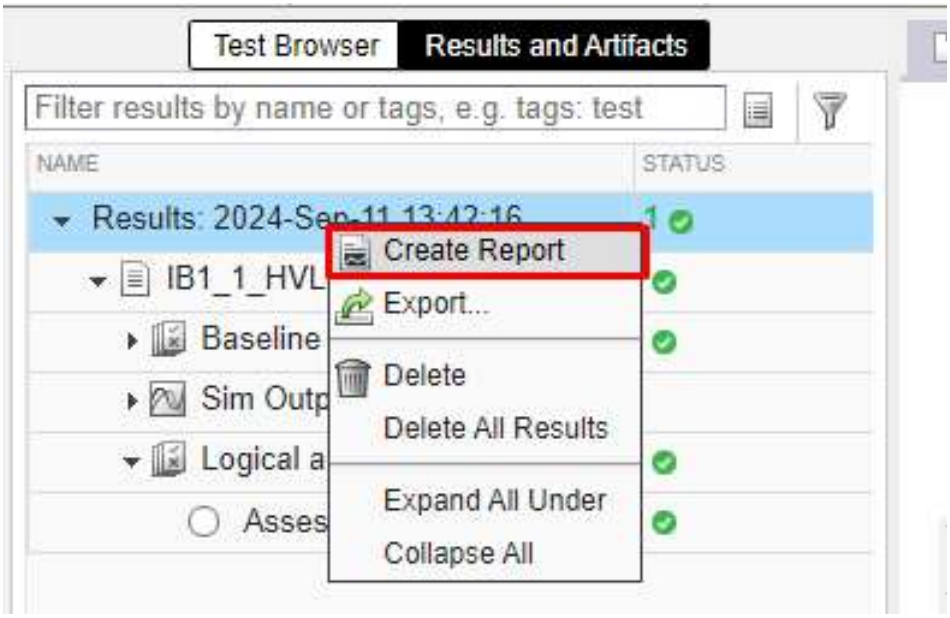

Figure 9: Simulink Test Manager - Creating a Test Report

This will open a pop-up window. Ensure your settings match the image below, with the Author section being filled with all team-member macids, and the File Name being set correctly. Your file should be named <Team#_Tests> (e.g. Team1_Tests).

4.1.1.1. Dev Challenge Presentation

Each team must create a presentation for the final day of the development challenge. This presentation can be made using PowerPoint or Prezi and must also include a quick walk-through of your test suite live in Simulink. Note that teams will be required to bring their own laptop for their presentation. Your presentation should outline the task, your processes to complete the task, results, challenges faced, and any other information you feel is relevant. Be prepared to answer questions at the end of your presentation.

5. Contacts for Support

If at any time you are stuck while completing the development challenge, please do not hesitate to contact any of the people below via teams!

Contact Name	Contact MacID
George Wahba	Wahbag
Thomas Stickland	Sticklat
Luc Rajotte	Rajottel
Daniel Edward	Edward9
Henry Zhao	Zhaoh52