Introduction

CMBS - Collision mitigation brake system.

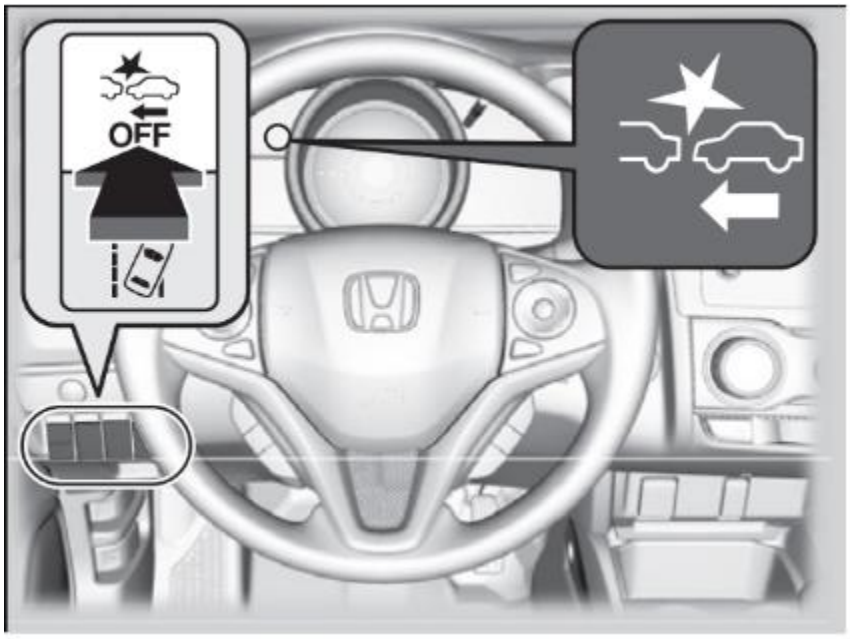
What exactly is CMBS?

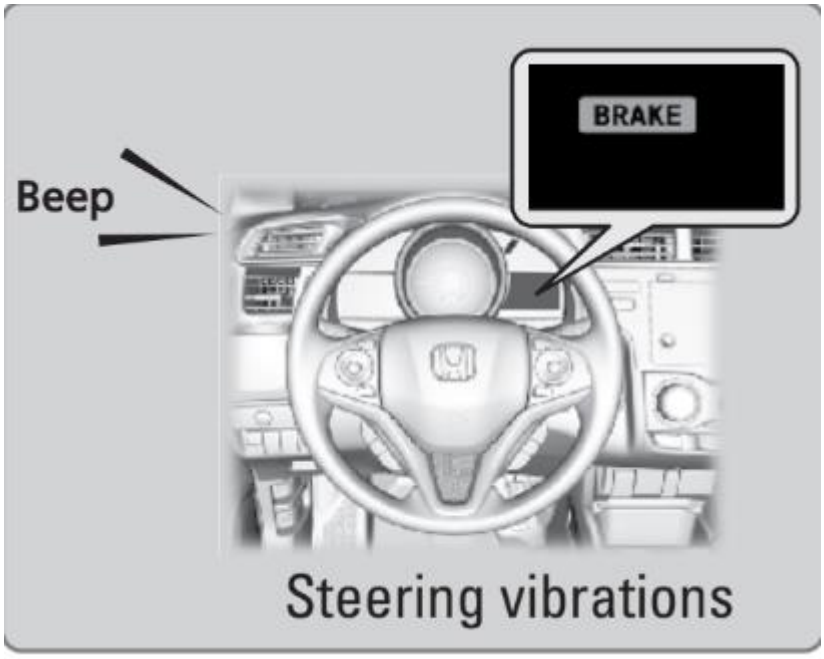
- Collision Mitigation Brake System (CMBS) is a radar-based autonomous emergency braking system. At speeds above 15km/h, moving and stationary vehicles are detected along a path some 100m ahead of the vehicle.

- When the system senses that the car is likely to hit one of these obstacles, a three-stage process is initiated. In the first, typically around 3 seconds before impact, the driver is alerted by visual and audible warnings. In the second stage, when the system senses that a collision is still likely (typically some 2 seconds before impact), three sharp tugs are given on the seat belt and the car automatically starts to apply some braking.

- Finally, when a collision is unavoidable, CMBS tightens the front seat occupants' seatbelts (using reversible tensioners different from the pyrotechnic devices used during the collision itself) and applies a high level of braking force. This braking can be supplemented by the driver up to the maximum that the car is capable of.

- All of the actions taken by CMBS are reversible: if an accident is averted (for example, if the vehicle moves out of the way at the last moment), the tension is removed from the seatbelts and the visual and audible warnings stop.





Benefits for CMBS -

CMBS is a system designed to help prevent rear-end collisions with vehicles which are stationary or travelling in the same direction. Several studies have shown that driver distraction or inattentiveness is a factor in the great majority of rear-end accidents. The system is aimed at alerting the driver to an imminent rear-end collision both at low speeds, typical of urban driving and at higher speeds typical of rural roads and highways. In such accidents, the most common sorts of injuries are to the cervical spine, the soft tissue of the thorax and to the knees.


TTC = d / V_rel
TTC = 3 sec.
TTC = 2 sec.
TTC = 1 sec.

Tools, Software and Keywords used

MATLAB, SIMULINK, MBD - Model-Based Development, Test Coverages, Auto-Code Generation, Model Checker and Advisor, CMBS.

Aim

Aim of the project is to design a Simulink model for one of the features of the Collision Mitigation Brake System. Purpose of the feature is to deactivate or not activate the Collision Mitigation Brake System(CMBS) while the vehicle is moving rearward or travelling in reverse gear. This Simulink model designed must evaluate the direction of the vehicle.

Methodology

Given -

Requirement Document: requirement_collision_mitigation_braking.docx

System Information File: Fit_Collision_Mitigation_Braking_System.pdf

Step 1 - Studying all the parts of the requirement document provided, also called as requirement analysis. Requirement Analysis is the first step in designing any model for a feature. Requirement sheet containing information regarding CMBS, function requirements and feature description is provided. Requirement sheet itself explains in detail about how the CMBS works.

1. Requirement document analysis -
    1. CMBS starts working once the vehicle reaches the speed of 15 kmph.

    2. The brake system on vehicles equipped with CMBS AND a manual transmission shall communicate wheel direction for at least one wheel via the CAN signal, *Whl_Dir_xx*.

    3. If a fault prevents accurate detection of vehicle direction, Whl_Dir_xx shall be set to 'Failed'. In case of a normal transient condition, it shall not report “Failed” but may report “Unknown”. (i.e. around 0 kph if the direction detection is temporary, not accurate enough).

    4. If a directional wheel speed sensor is not installed on a given wheel, Whl_Dir_xx shall be set to 'unknown'.

    5. Vehicle speed is compared with calibration data which says wheel Direction detection is not accurate. If the vehicle speed is greater than calibration data, a

FAULT is detected at the wheel direction detection, else it says data inaccurate to determine vehicle direction detection.

6. Fault data of vehicle wheel sensor (from CAN) is compared against the Vehicle Wheel sensor data & error data which compares wheel direction. When no fault is detected, it takes sensor data, else it takes the error data calculated. This calculation remains the same across all wheels.

7. The output of the system is information regarding the direction of each wheel.

2. Modelling and Logic development -

1. After the successful requirement analysis with the help of document provided, there is the time to build the required Simulink model. In the first stage, the base model is developed. It consists of the subsystem including all the input and outputs.

2. The inputs for the system according to the requirement sheet document provided are -
    1. Vehicle speed data - EstVehOvrGnd

    2. Front Left Wheel direction sensor data - Whl_Dir_FL_Snsr_Data

    3. Front Right Wheel direction sensor data- Whl_Dir_RL_Snsr_Data

    4. Rear Left Wheel direction sensor data - Whl_Dir_RL_Snsr_Data

    5. Rear Right Wheel direction sensor - Whl_Dir_RR_Snsr_Data

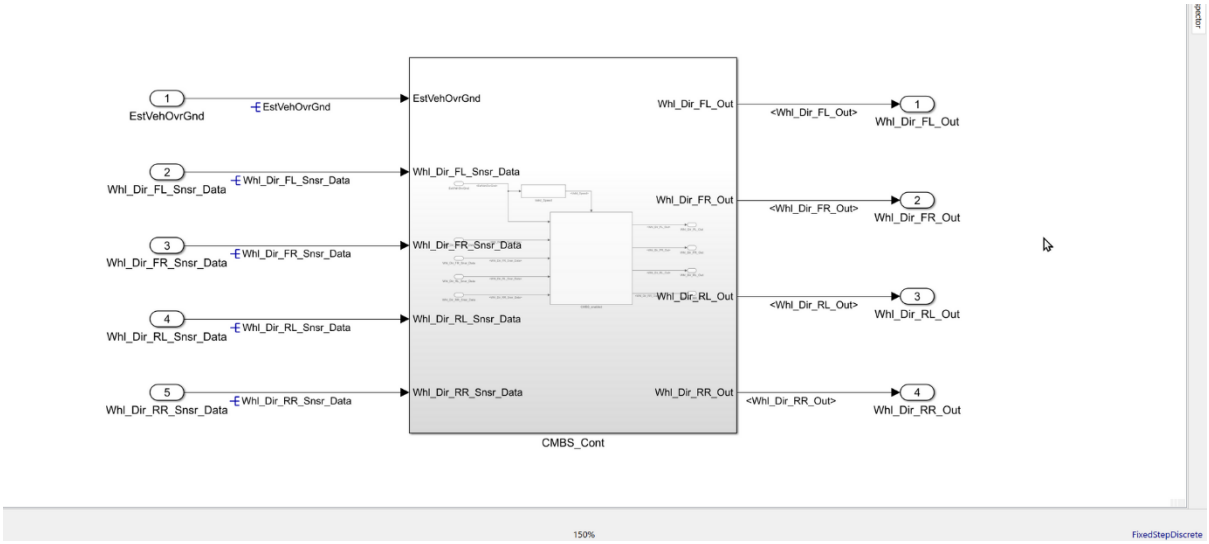
3. The outputs for the mentioned subsystem are -
    1. Front Left Wheel Direction data - Whl_Dir_FL_Out

    2. Front Right Wheel Direction data - Whl_Dir_FR_Out

    3. Rear Left Wheel Direction data - Whl_Dir_RL_Out

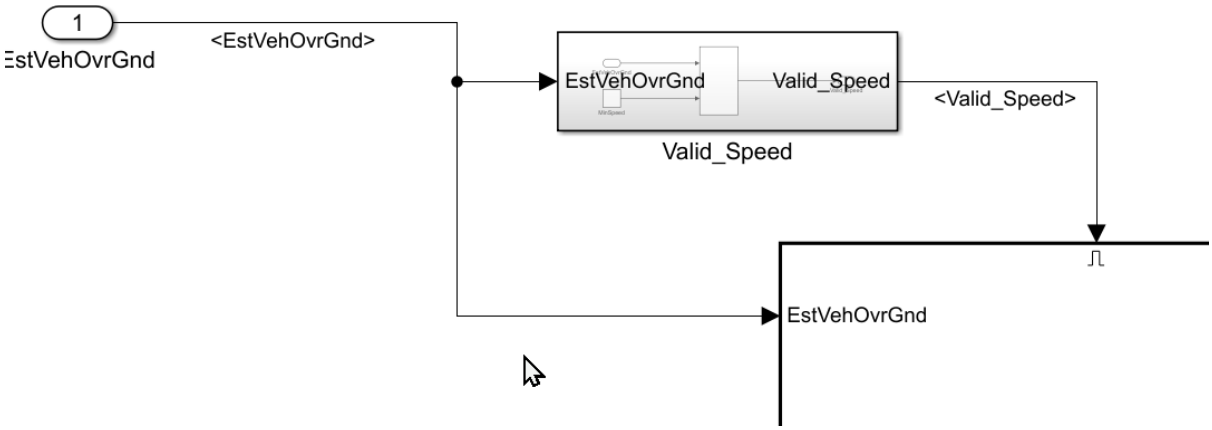
    4. Rear Right Wheel Direction data - Whl_Dir_RR_Out

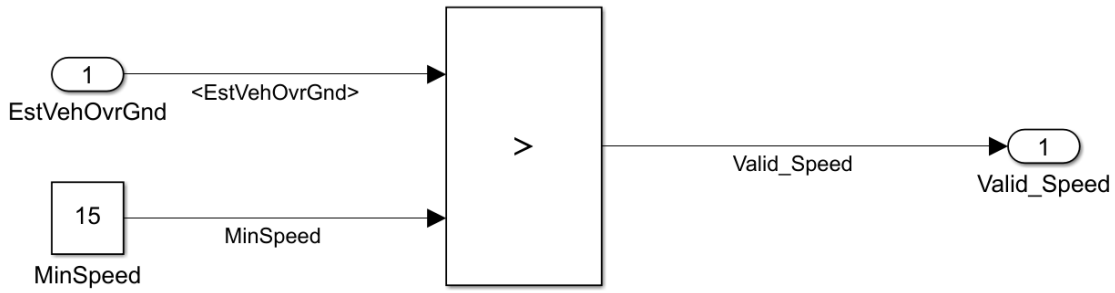
4. Inside the base subsystem, another subsystem is developed. This subsystem is enabled based on the comparison made based on the speed conditions. As per the requirement document, the threshold speed for turning the CMBS system is 15 Kmph. Input speed - EstVehOvrGnd - is compared with a constant block with value 15 Kmph. Inside the enabled subsystem, there are separate subsystems for 4 wheels. Each subsystem gives wheel direction as an output. Each subsystem receives its input as vehicle speed which is - EstVehOvrGnd - and respective wheel sensor data.

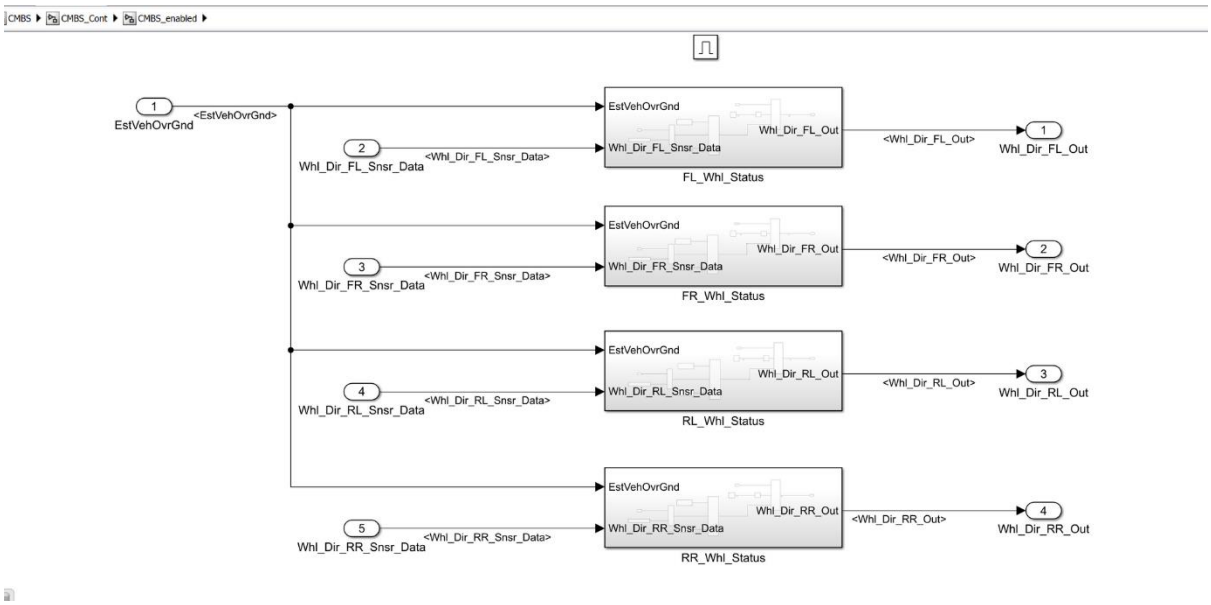


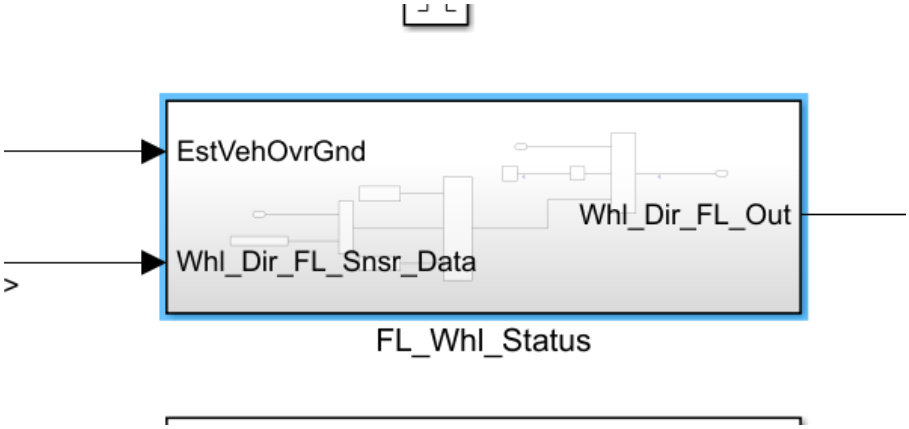
Inside the main subsystem - CMBS_Cont, another two subsystems are created for speed comparison which is Valid_Speed and another enabled subsystem for logic developed for individual tyre speed control.

- FL_Whl_Status
- FR_Whl_Status
- RL_Whl_Status
- RR_Whl_Status



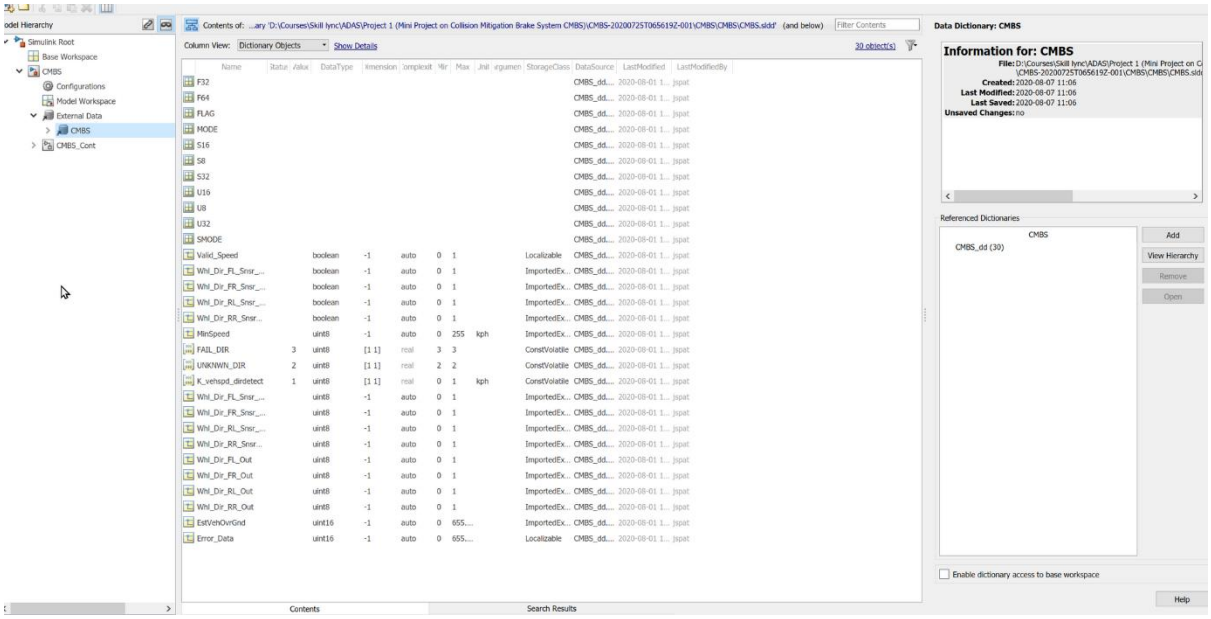

1
EstVehOvrGnd
<EstVehOvrGnd>
15
MinSpeed
MinSpeed
>
Valid_Speed
1
Valid_Speed


CMBS
CMBS_Cont
CMBS_enabled
1
EstVehOvrGnd
<EstVehOvrGnd>
2
Whl_Dir_FL_Snsr_Data
<Whl_Dir_FL_Snsr_Data>
EstVehOvrGnd
Whl_Dir_FL_Snsr_Data
Whl_Dir_FL_Out
FL_Whl_Status
<Whl_Dir_FL_Out>
1
Whl_Dir_FL_Out
3
Whl_Dir_FR_Snsr_Data
<Whl_Dir_FR_Snsr_Data>
EstVehOvrGnd
Whl_Dir_FR_Snsr_Data
Whl_Dir_FR_Out
FR_Whl_Status
<Whl_Dir_FR_Out>
2
Whl_Dir_FR_Out
4
Whl_Dir_RL_Snsr_Data
<Whl_Dir_RL_Snsr_Data>
EstVehOvrGnd
Whl_Dir_RL_Snsr_Data
Whl_Dir_RL_Out
RL_Whl_Status
<Whl_Dir_RL_Out>
3
Whl_Dir_RL_Out
5
Whl_Dir_RR_Snsr_Data
<Whl_Dir_RR_Snsr_Data>
EstVehOvrGnd
Whl_Dir_RR_Snsr_Data
Whl_Dir_RR_Out
RR_Whl_Status
<Whl_Dir_RR_Out>
4
Whl_Dir_RR_Out



As per the requirement document, we need to compare vehicle speed and calibration value - K_vehspd_dirdetect. Comparison is done using a relational operator block. The output of comparison is given to a switch. If the result of the comparison is 1 that is Vehicle speed is greater than calibration value, K_vehspd_dirdetect is Set to "Failed". Hence FAIL_DIR values go further, else UNKNWN_DIR will go. The output of the switch is again given to port of another switch. This switch is triggered by the detection of a sensor fault. If the sensor is faulty, the error data generated from the previous switch goes further. If the sensor is not faulty, wheel sensor data goes out of switch as wheel direction output signal.

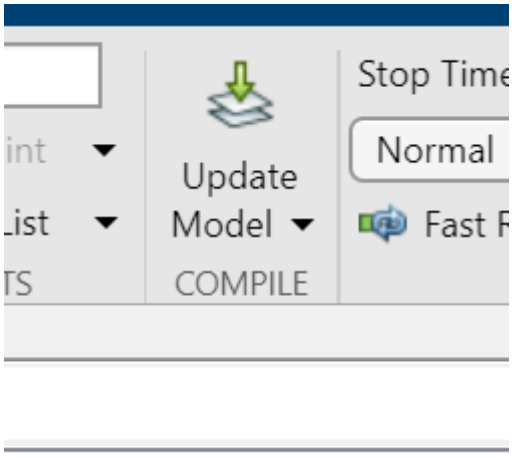
3. Next step is to create an SLDD library and corresponding signal resolution for inputs and outputs for every subsystem in the model.



4. After the SLDD file is created and all the signal have been resolved, we proceed further with the auto code generation with the help of Simulink code generation facility.

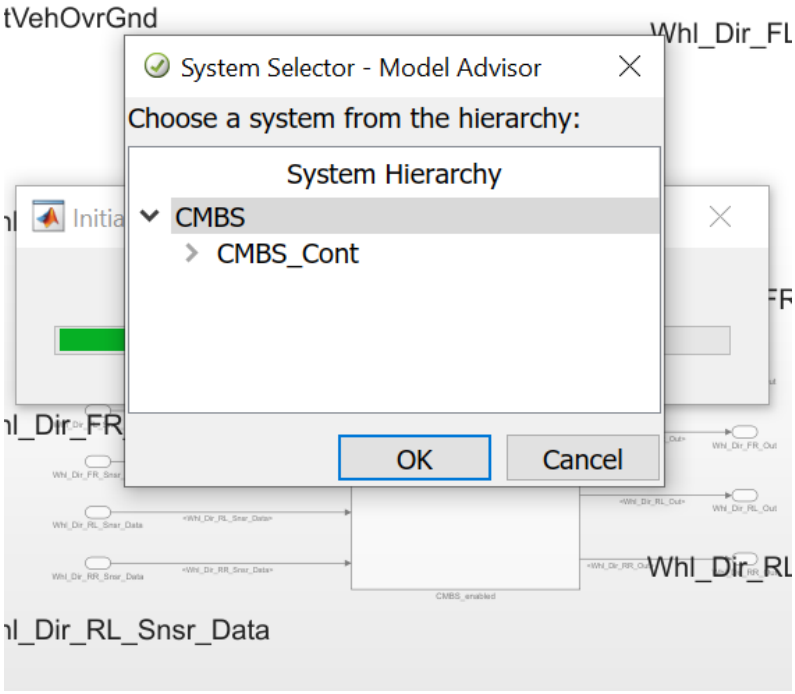
But before that the model and updated and then checked with the help of model advisor report.

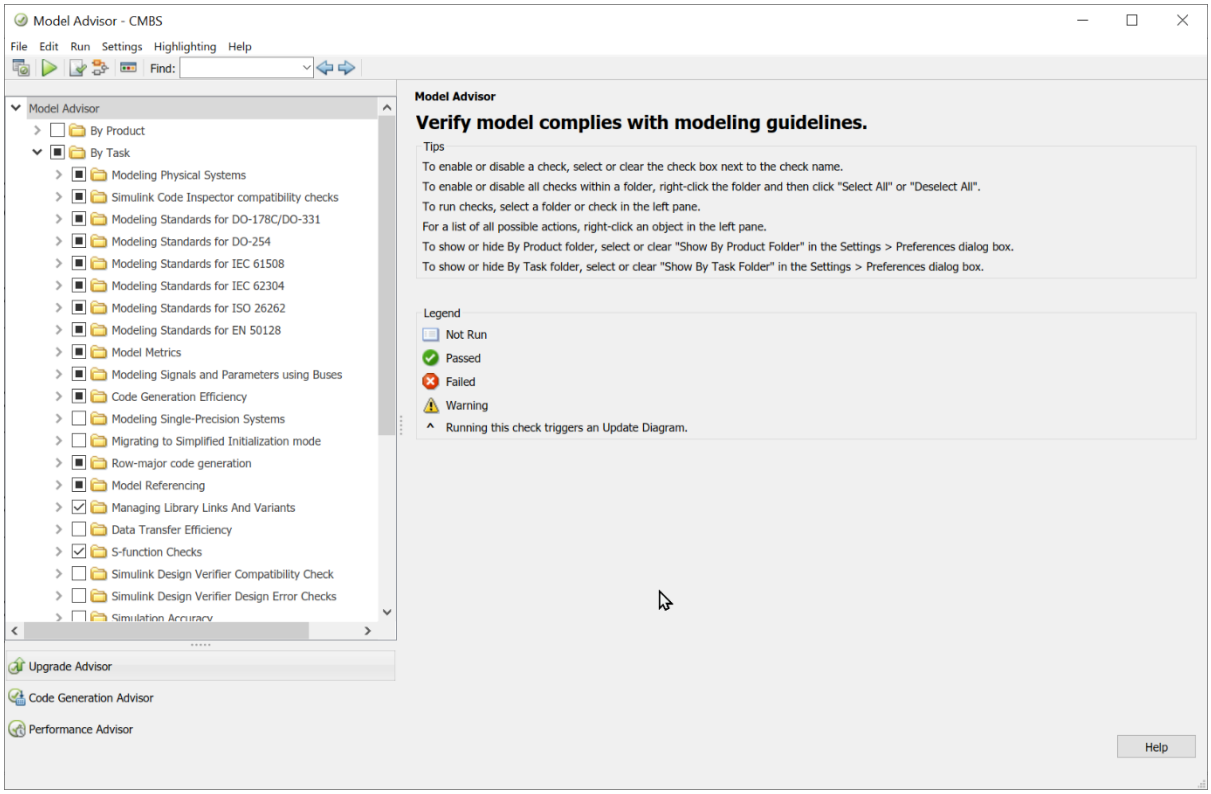
After resolving all input and output signals, the model is updated using the update option in modelling tab. No Errors are found in model design. It means there wouldn't be any error while the run of the model and code generation. It is also done using shortcut key ctrl+D.

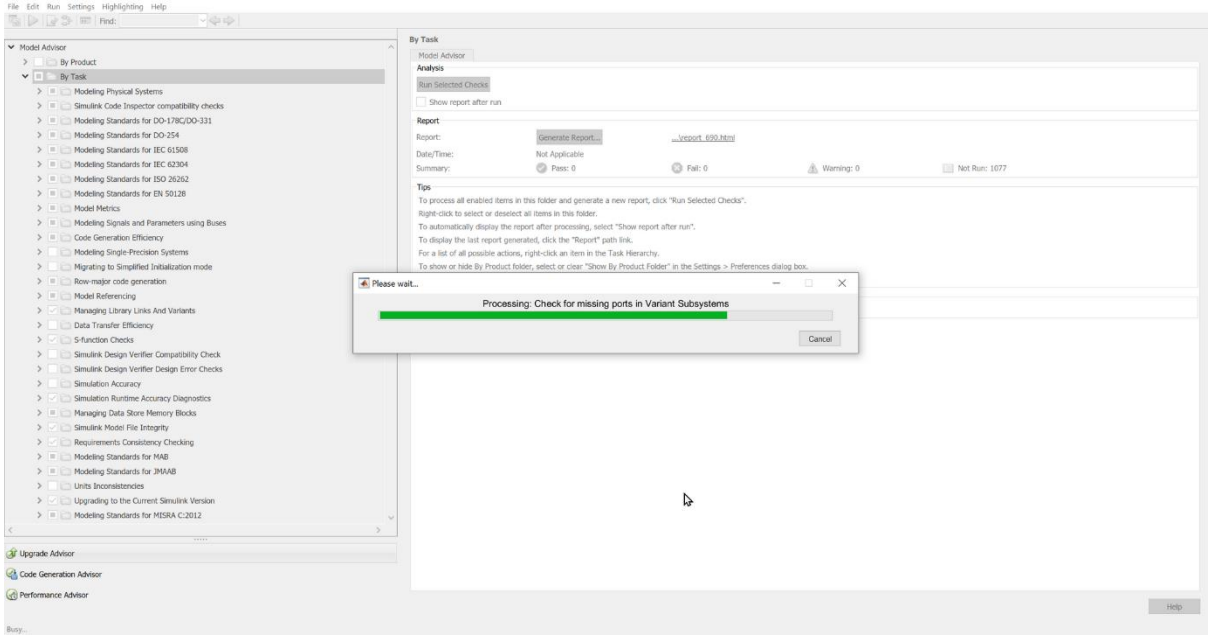


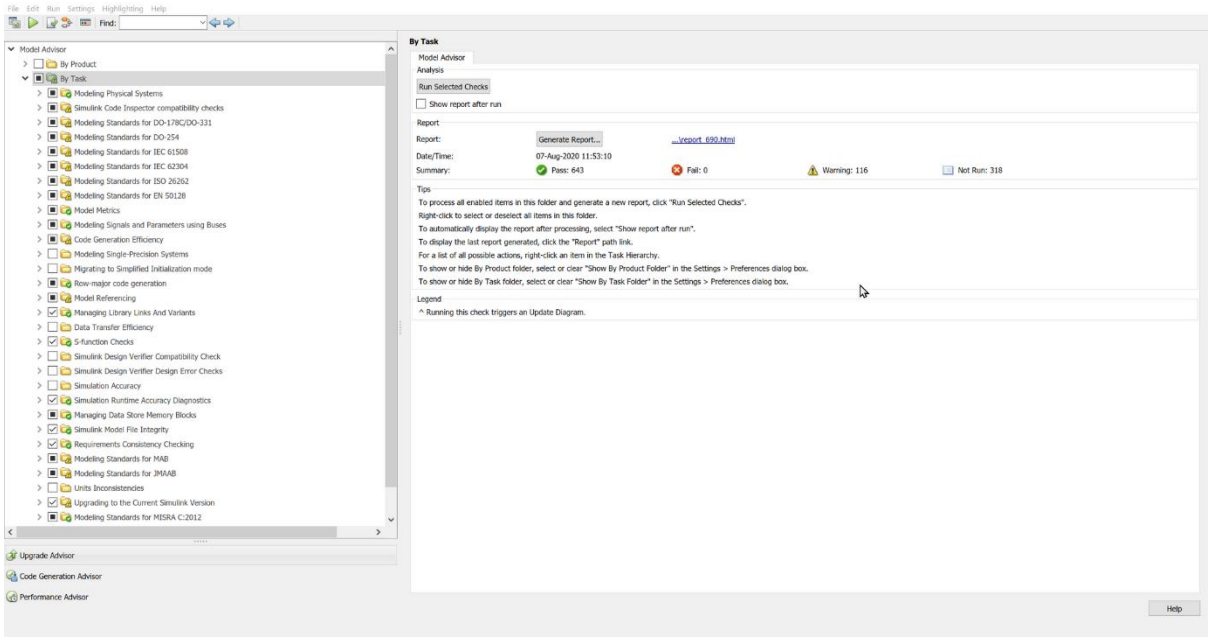
Configuration Parameter Update: Once it is sure that the model doesn't have any error, the model set can be set appropriately. Model Setting option is available under Modelling Tab. All Configuration parameter is set and its images are saved in ppt. The same ppt is attached to this project.

Model Advisor Check: Model Advisor Check option is available in Modelling Tab. As the scope of the project is small, the Advisor check is carried out for a few parameters only. Warnings got after running the check are resolved and the final report is generated and saved. The same report is also attached to the project. It is not necessary to resolve all the errors, so important errors are resolved which may hamper the code generation process.




Model Advisor - CMBS
File Edit Run Settings Highlighting Help
Find:
Model Advisor
By Product
By Task
Modeling Physical Systems
Simulink Code Inspector compatibility checks
Modeling Standards for DO-178C/DO-331
Modeling Standards for DO-254
Modeling Standards for IEC 61508
Modeling Standards for IEC 62304
Modeling Standards for ISO 26262
Modeling Standards for EN 50128
Model Metrics
Modeling Signals and Parameters using Buses
Code Generation Efficiency
Modeling Single-Precision Systems
Migrating to Simplified Initialization mode
Row-major code generation
Model Referencing
Managing Library Links And Variants
Data Transfer Efficiency
S-function Checks
Simulink Design Verifier Compatibility Check
Simulink Design Verifier Design Error Checks
Simulation Accuracy
Upgrade Advisor
Code Generation Advisor
Performance Advisor
Model Advisor
Verify model complies with modeling guidelines.
Tips
To enable or disable a check, select or clear the check box next to the check name.
To enable or disable all checks within a folder, right-click the folder and then click "Select All" or "Deselect All".
To run checks, select a folder or check in the left pane.
For a list of all possible actions, right-click an object in the left pane.
To show or hide By Product folder, select or clear "Show By Product Folder" in the Settings > Preferences dialog box.
To show or hide By Task folder, select or clear "Show By Task Folder" in the Settings > Preferences dialog box.
Legend
Not Run
Passed
Failed
Warning
Running this check triggers an Update Diagram.
Help


Please wait...
Processing: Check for missing ports in Variant Subsystems
Cancel

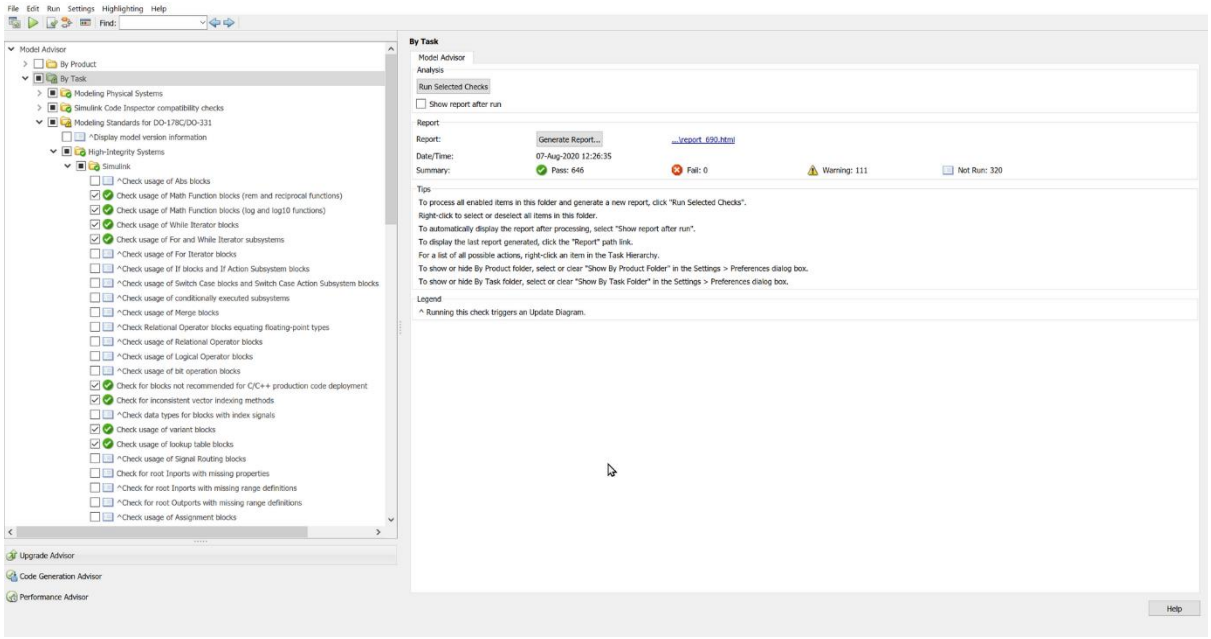


After the model advisor check report, the result was as follows -

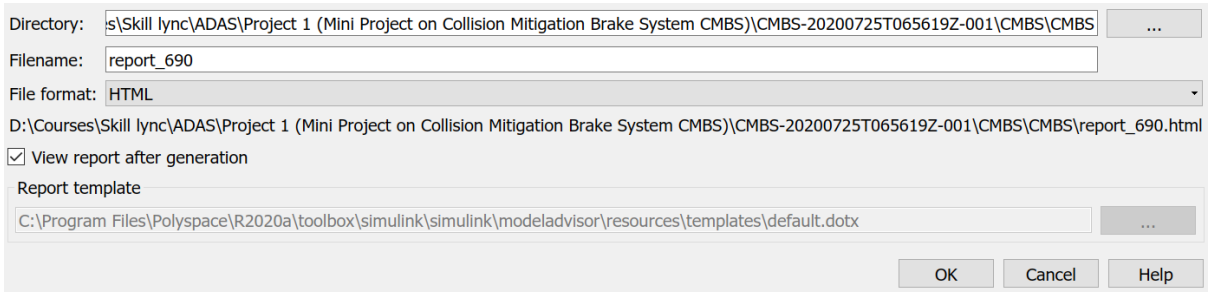
- Pass - 643
- Failed - 0
- Warnings - 116

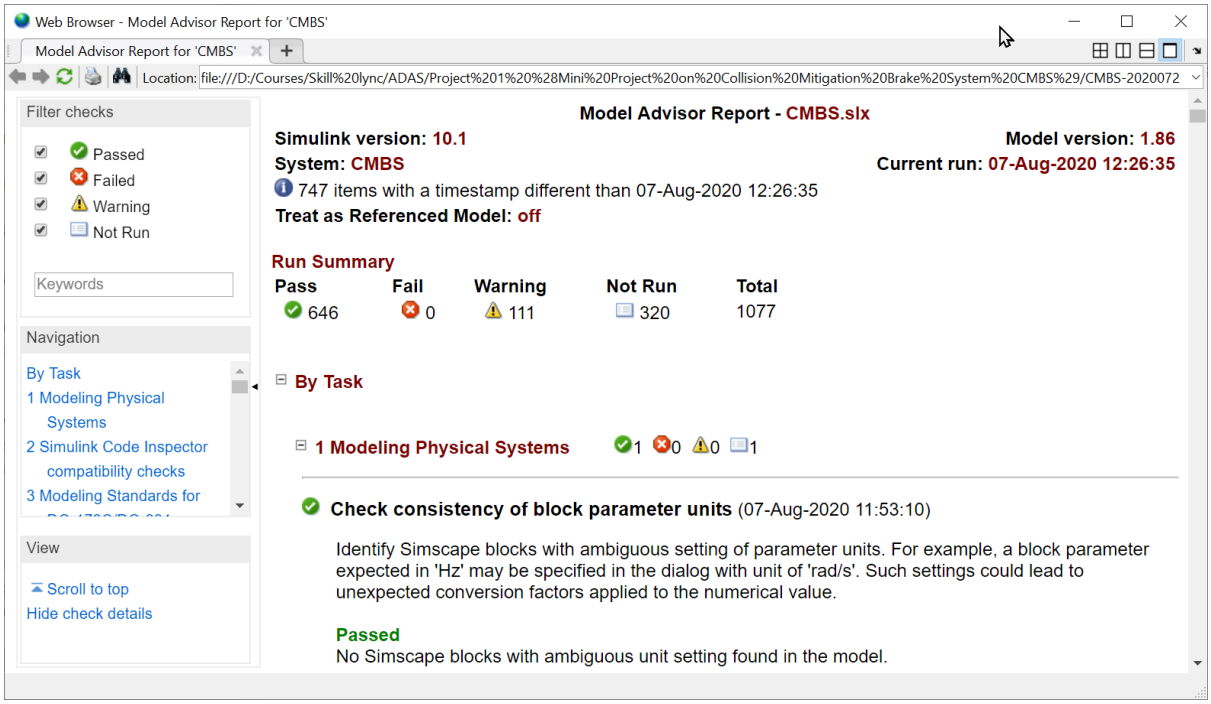
Furthermore, correct some of the error suggested by the model advisor, the model was rechecked, this time the number of warnings got reduced and the number of modules that were passed got increased.

- Pass - 646
- Failed - 0
- Warnings - 111



The report is then generated for the given Simulink model.





5. Next part we move on to the code generation. Last Part of the model design is code generation. Once all necessary changes been done as per model advisor, code generation can be started. ctrl+B is a shortcut key to start code generation. After successful completion of the code generation process, c-code is generated for the model created in Simulink.

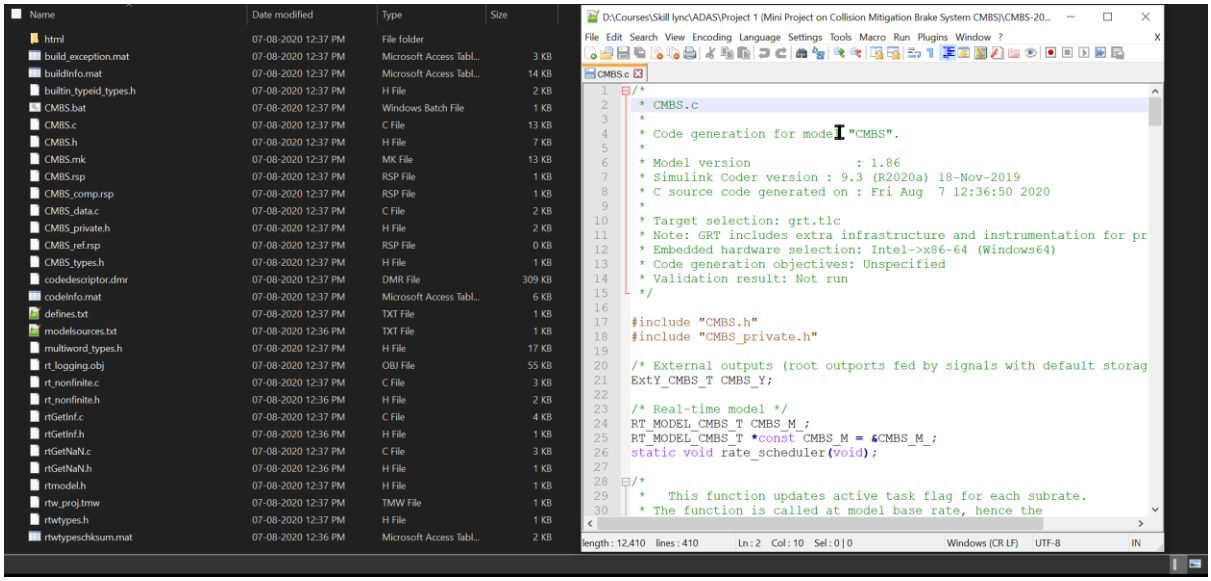
Analysis and Results

Requirement Document: requirement_collision_mitigation_braking.docx

System Information File: Fit_Collision_Mitigation_Braking_System.pdf

```c
/*
 * CMBS.c
 *
 * Code generation for model "CMBS".
 *
 * Model version              : 1.86
 * Simulink Coder version : 9.3 (R2020a) 18-Nov-2019
 * C source code generated on : Fri Aug  7 12:36:50 2020
 *
 * Target selection: grt.tlc
 * Note: GRT includes extra infrastructure and instrumentation for prototyping
 * Embedded hardware selection: Intel->x86-64 (Windows64)
 * Code generation objectives: Unspecified
 * Validation result: Not run
 */

#include "CMBS.h"
#include "CMBS_private.h"
```



Conclusion

THE FOLLOWING FILES ARE CREATED DURING THIS PROJECT.

Model File: CMBS.slx

Model Advisor Report: Model_Advisor_Report.pdf

Simulink Data Dictionary: CMBS.sldd

C- Code : CMBS.c

Header File: CMBS.h

Model Setting: Configuration Parameter Settings.pptx