ASM Diesel Engine

Model Description

Release 2021-A – May 2021

dSPACE

How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	+49 5251 1638-0
Fax:	+49 5251 16198-0
E-mail:	info@dspace.de
Web:	http://www.dspace.com

How to Contact dSPACE Support

If you encounter a problem when using dSPACE products, contact your local dSPACE representative:

- Local dSPACE companies and distributors: http://www.dspace.com/go/locations
- For countries not listed, contact dSPACE GmbH in Paderborn, Germany.
  Tel.: +49 5251 1638-941 or e-mail: support@dspace.de

You can also use the support request form: http://www.dspace.com/go/supportrequest. If you are logged on to mydSPACE, you are automatically identified and do not need to add your contact details manually.

If possible, always provide the relevant dSPACE License ID or the serial number of the CmContainer in your support request.

Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit http://www.dspace.com/go/patches for software updates and patches.

Important Notice

Contents

About This Document

Contents

This document shows you how to work with the ASM Diesel Engine Model.

Symbols

dSPACE user documentation uses the following symbols:

Symbol	Description
⚠ DANGER	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
⚠ WARNING	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
⚠ CAUTION	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a hazard that, if not avoided, could result in property damage.
Note	Indicates important information that you should take into account to avoid malfunctions.
Tip	Indicates tips that can make your work easier.
?	Indicates a link that refers to a definition in the glossary, which you can find at the end of the document unless stated otherwise.
📖	Precedes the document title in a link that refers to another document.

Naming conventions

dSPACE user documentation uses the following naming conventions:

%name% Names enclosed in percent signs refer to environment variables for file and path names.

< > Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

Special folders

Some software products use the following special folders:

Common Program Data folder A standard folder for application-specific configuration data that is used by all users.

`%PROGRAMDATA%\dSPACE\<InstallationGUID>\<ProductName>`

or

`%PROGRAMDATA%\dSPACE\<ProductName>\<VersionNumber>`

Documents folder A standard folder for user-specific documents.

`%USERPROFILE%\Documents\dSPACE\<ProductName>\`
`<VersionNumber>`

Local Program Data folder A standard folder for application-specific configuration data that is used by the current, non-roaming user.

`%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\`
`<ProductName>`

Accessing dSPACE Help and PDF Files

After you install and decrypt dSPACE software, the documentation for the installed products is available in dSPACE Help and as PDF files.

dSPACE Help (local) You can open your local installation of dSPACE Help:

- On its home page via Windows Start Menu
- On specific content using context-sensitive help via **F1**

dSPACE Help (Web) You can access the Web version of dSPACE Help at www.dspace.com/go/help.

To access the Web version, you must have a *mydSPACE* account.

PDF files You can access PDF files via the icon in dSPACE Help. The PDF opens on the first page.

Introduction

Where to go from here

Information in this section

Demo Information

Demo model

This model simulates turbocharged diesel engines extended by longitudinal models for drivetrain, vehicle and driver.

The engine model consists of components for the complete air path including turbocharger, the fuel system, the combustion and the exhaust system. Generally it is modeled as a mean value engine model with additional combustion torque modulation. The engine is controlled by a soft ECU.

The drivetrain has two options for the transmission system: a manual transmission and an automatic transmission. Instead of a drivetrain the engine can also be simulated on an engine test bench.

The longitudinal vehicle model takes into account slope and air, rolling and braking resistance. The driver model is used to follow reference velocities given by standard driving cycles like FTP-75 (Federal Test Procedure), NEUDC (New European Driving Cycle), Japanese 10-15, …

The initial default parameterization of the model simulates a:

- 4 cylinder 2.0 l turbocharged diesel engine
- Common-rail fuel system

- Manual 5-Gear transmission
- Mid-size car with a vehicle mass of 1000 kg
- No slope, standard ambient parameters

The initial maneuver defines that the driver should follow a reference velocity given by the standard driving cycle WLTC_Class3.

An alternative default parameterization of the model simulates a:

- 6 cylinder 12.0 l turbocharged diesel engine
- Common-rail fuel system or Unit Injector system
- Manual 12-Gear transmission
- Truck with a vehicle mass of 40000 kg
- No slope, standard ambient parameters

To call this alternative variant in MATLAB, call the following command after opening the model.

```
go('engine', 'TruckDiesel6Cyl12p0l')
```

For detailed information, refer to Model Initialization with go.m File (ASM User Guide 🕮).

For this parameterization, the engine can also be simulated on an engine test bench running cycles such as the European Stationary Cycle (ESC) and European Transient Cycle (ETC). Refer to How to Select a Test Cycle on page 24.

Note

Two library versions are available:

- The ASM Developer library: The Developer library is a standard ASM library. You can view the implementation of the models down to the last detail. You can use the models from the Developer library to generate code for dSPACE platforms.
- The ASM Operator library: On the block level, the Operator library consists of S-functions. The implementation of the ASM blocks is hidden. You cannot build models for dSPACE platforms. However, the two libraries look the same up to the ASM block level.

Related topics

Basics

Model Initialization with go.m File (ASM User Guide 🕮)

Model Preparation for Real-Time Simulation (PHS-Bus-Based Platforms)

Model preparation

Before you use the model for real-time simulation, you have to prepare the model. To avoid the real-time process from being stopped by task overruns during the initialization phase of the model, it is recommended to allow a number of queued task calls before the simulation is stopped.

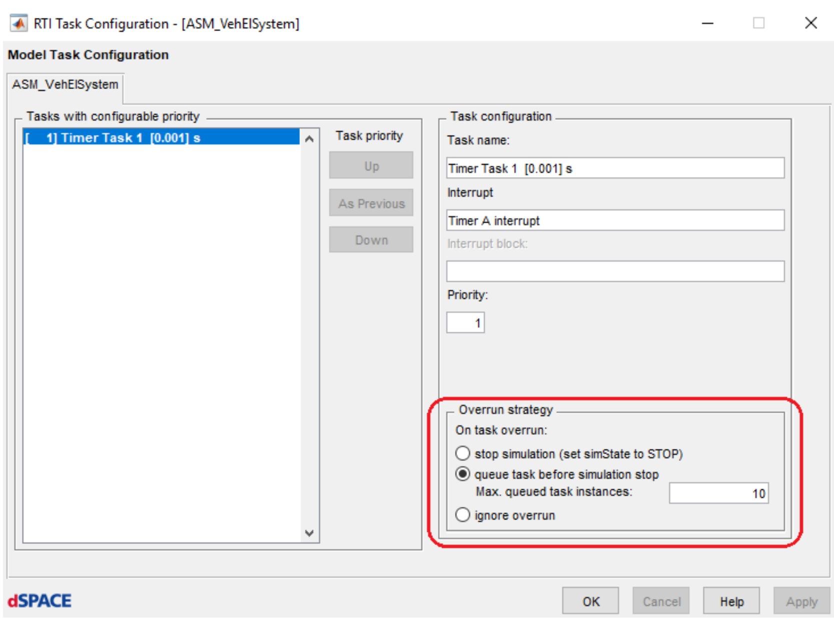
> **Note**
>
> If you use the Operator version of the ASM, you cannot generate code for real-time simulation.

Method

To avoid stopping the real-time process on a singular task overrun

1. In Simulink, select Code – C/C++ Code – Code Generation Options to open the Configuration Parameters dialog.
2. In the Configuration Parameters dialog, select RTI simulation options from the Select tree.
3. Click Task Configuration.

   After a few seconds the RTI Task Configuration dialog opens.



4. In this dialog, select queue task before simulation stop as the overrun strategy.
5. In Max. queued task instances, enter `10` and click OK.

   This setting has no appreciable effect on your normal task overrun behavior, because only a few singular task overruns (for example, 10) are accepted.

Tutorials

Where to go from here

Information in this section

Handling the Model

Where to go from here

Information in this section

How to Open an ASM Library

Objective

After successfully installing ASM on your PC, you can now easily open an ASM library.

This lesson explains how to open the Gasoline Engine library. You can start all ASM products in a similar way. The MATLAB Command Window shows you all the products that are installed and covered by your license.

Method

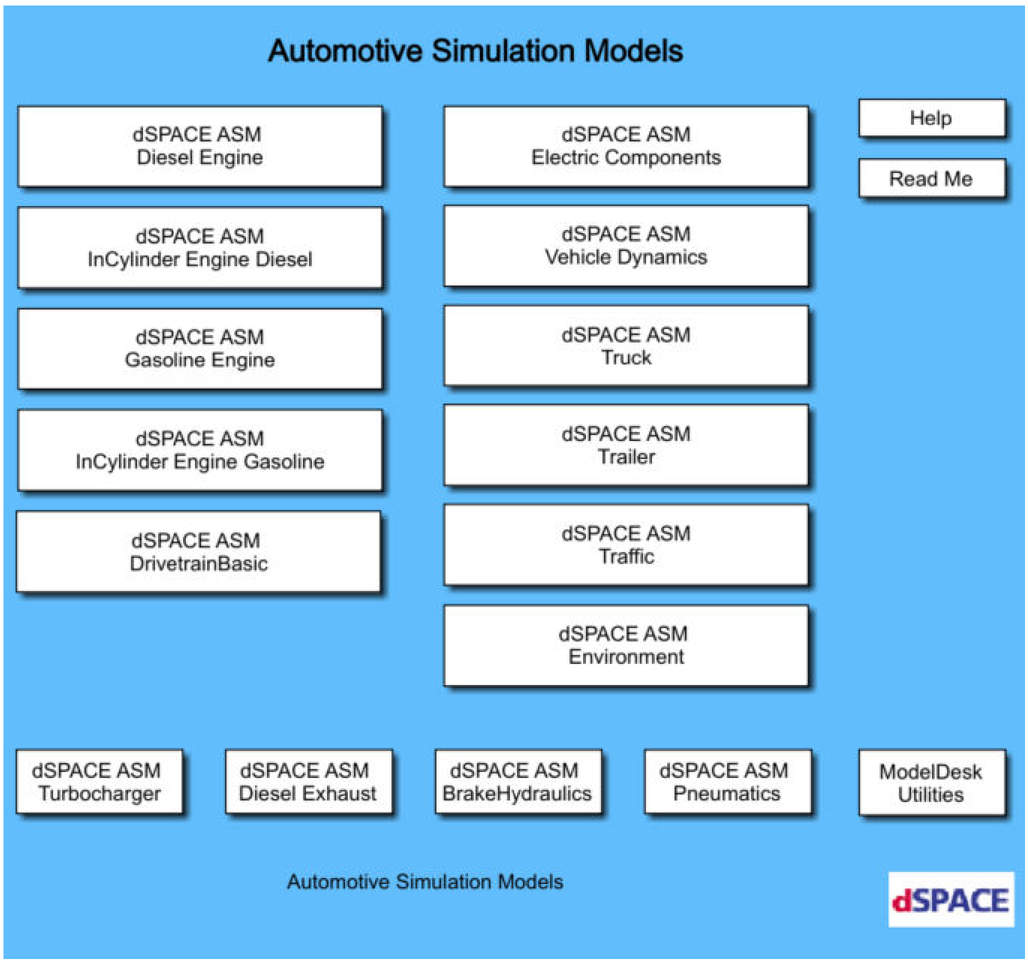
To open an ASM library

1. Start a MATLAB session.

   When MATLAB starts, the MATLAB Command Window lists all the ASM blocksets that are installed on the PC.

2. Type `asm` in your MATLAB Command Window.

   The central ASM library opens.



   It provides an overview of all the accessible major ASM libraries:

   - ASM Diesel Engine
   - ASM InCylinder Engine Diesel
   - ASM InCylinder Engine Gasoline
   - ASM Gasoline Engine
   - ASM Drivetrain Basic
   - ASM Vehicle Dynamics

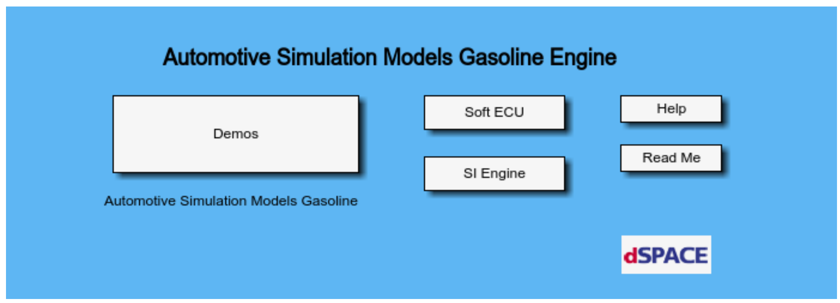
- ASM Electric Components
- ASM Trailer
- ASM Traffic
- ASM Truck
- ASM Environment

Depending on your software license, some of the libraries are grayed out. If a library is grayed, you do not have a valid license for it.

3 Double-click the **dSPACE ASM Gasoline Engine** subsystem.

The subsystem opens. Depending on the installed license, you can open the developer or operator library. For details, refer to Operator Version (ASM User Guide 📖).

4 Double-click the **Developer** or **Operator** subsystem to open the library.



Result

The library opens. It is subdivided into three sections.

- On the left is a subsystem with all the demo models belonging to the library.
- The middle section contains the subsystems that are part of the library and which are contained in the example model as links.
- On the right are links to the documentation and to the last-minute information in the readme section.

Related topics

HowTos

How to Start with an ASM Demo Model via MATLAB.. 14

How to Start with an ASM Demo Model via MATLAB

Objective

After successfully installing ASM on your PC, you can now easily start a demo model of your ASM installation via MATLAB.

This lesson explains how to start the Gasoline Engine model via MATLAB. You can start all ASM products in a similar way. The MATLAB Command Window shows you all the products that are installed and covered by your license.

Methods to start with an ASM demo model

You can start an ASM demo model with the following methods:

- Start an ASM demo model via MATLAB, see below.
- Generate an ASM project by using ModelDesk or by using MATLAB/Simulink. Refer to How to Create a Project Based on an ASM Demo (ModelDesk Project and Experiment Management 📖).

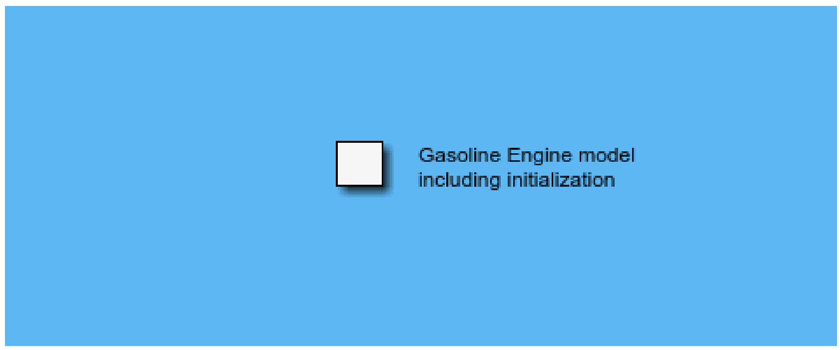
Preconditions

The ASM library is open. Refer to How to Open an ASM Library on page 12.

Method

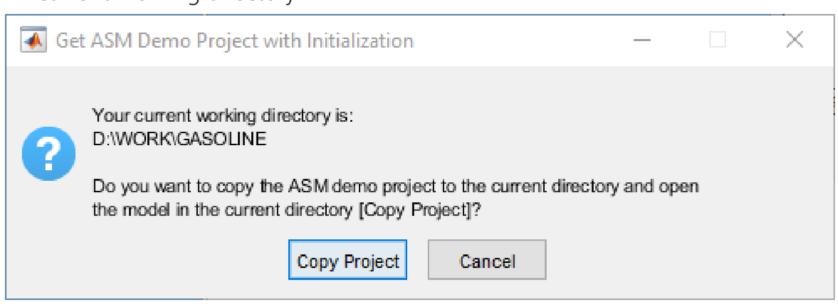
To start with an ASM demo model via MATLAB

1 Double-click the Demos subsystem to open a new window which presents all the demo models that are part of the library.



2 Double-click the demo model to start it.

This opens a dialog that offers you the possibility to copy the project to your current working directory.



3 Click Copy Project to copy the current demo folder to your current MATLAB work folder.

All the files related to the demo model are copied to the current MATLAB work folder. This includes the entire parameterization and the ControlDesk example layouts.

Result

Now the demo model opens. All the initialization files are executed and you can start simulation.

Related topics

HowTos

How to Create a Project Based on an ASM Demo (ModelDesk Project and Experiment Management 📖)

How to Start with an ASM Demo Model Using ModelDesk

Objective

After successfully installing ASM on your PC, you can now easily start a demo model of your ASM installation using ModelDesk.

Methods to start with an ASM demo model

You can start an ASM demo model with the following methods:

- Generate an ASM project by using ModelDesk or by using MATLAB/Simulink, see below.
- Start an ASM demo model via MATLAB. Refer to How to Start with an ASM Demo Model via MATLAB on page 14.

ASM demos

The ASM installation contains several ASM demos. The ASM demos include all the necessary files for the simulation, for example:

- Simulation model based on the ASM blocks
- Simulation applications for the simulation platforms
- ModelDesk project for parameterizing the model
- ControlDesk project for experimenting with the model
- MotionDesk project for animation (if useful)

Preconditions

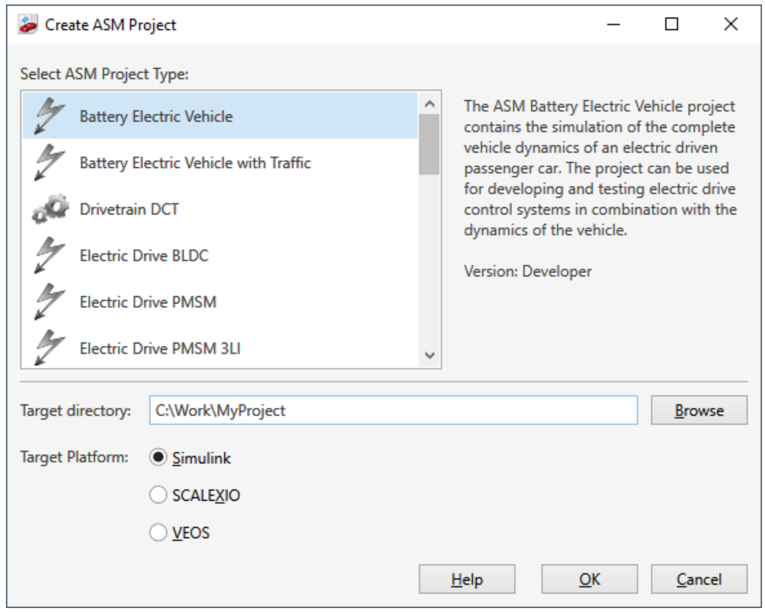
- No other project must be open.
- The license of the ASM library that is used by the ASM demos must be available.
- The ASM library must be decrypted.

Method

To to start with an ASM demo model using ModelDesk

1 On the File ribbon, click New – ASM Project.

The Define a Project dialog opens.



2 Select an ASM project type.

3 Specify the target directory. Select an empty directory or specify a new directory. You must have write permission to the directory.

4 Select the target platform to be activated in the experiment.

5 Click OK.

Result

ModelDesk copies all files of the selected ASM demo to the specified target folder and opens the project.

How to Open a Parameterization Project

Objective

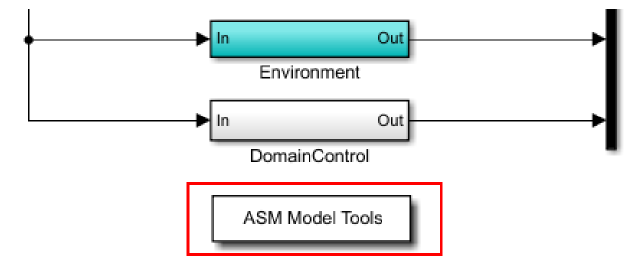
You can open a parameterization project from the model.

The parameterization project for the current model is accessible via the Open ModelDesk Project & Experiment button.

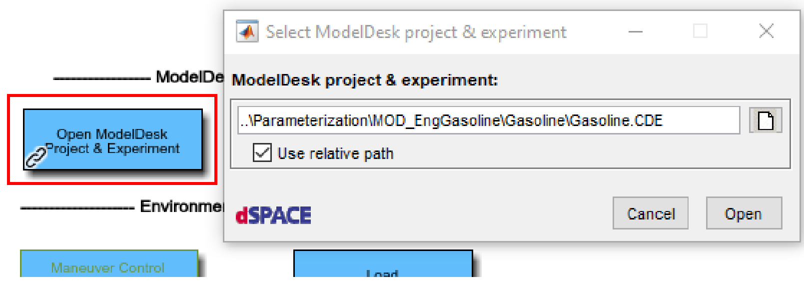
Method

To open a parameterization project

1 In the ASM model, double-click the ASM Model Tools button below the modules.



2 Double-click the Open ModelDesk Project & Experiment button.



3 In the dialog, click Open to open the selected ModelDesk project.

Result

ModelDesk is started and the selected project and experiment is loaded.

How to Parameterize ASM via ModelDesk

Objective

The `<Projectfolder>\Parameterization` folder includes a complete ModelDesk project. You can perform the complete parameterization of the ASM model with ModelDesk.

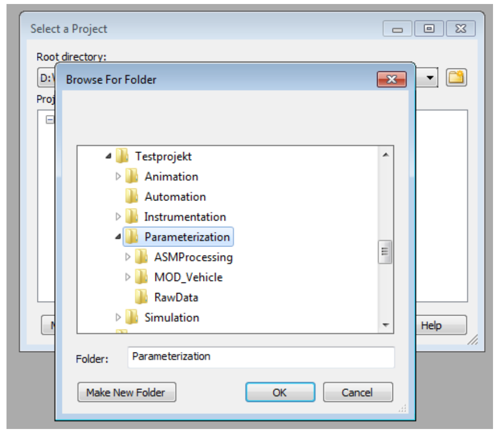
In the following method, an ASM Electric Components demo is used as an example.

Method

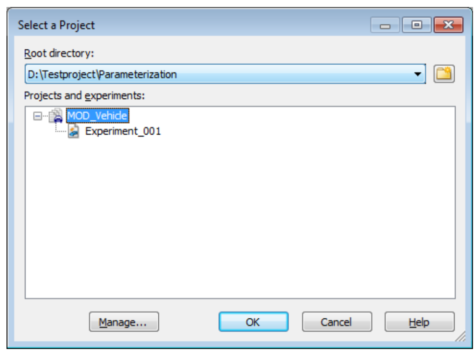
To parameterize ASM via ModelDesk

1 Start ModelDesk and open the project from `<Projectfolder>\Parameterization`.

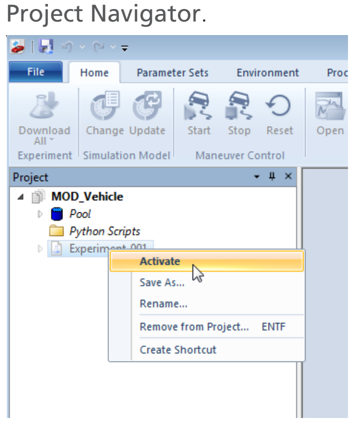
1. Click **File – Open – Project + Experiment**.
2. Define the root directory in `<Projectfolder>\Parameterization`.



3. Select the project.

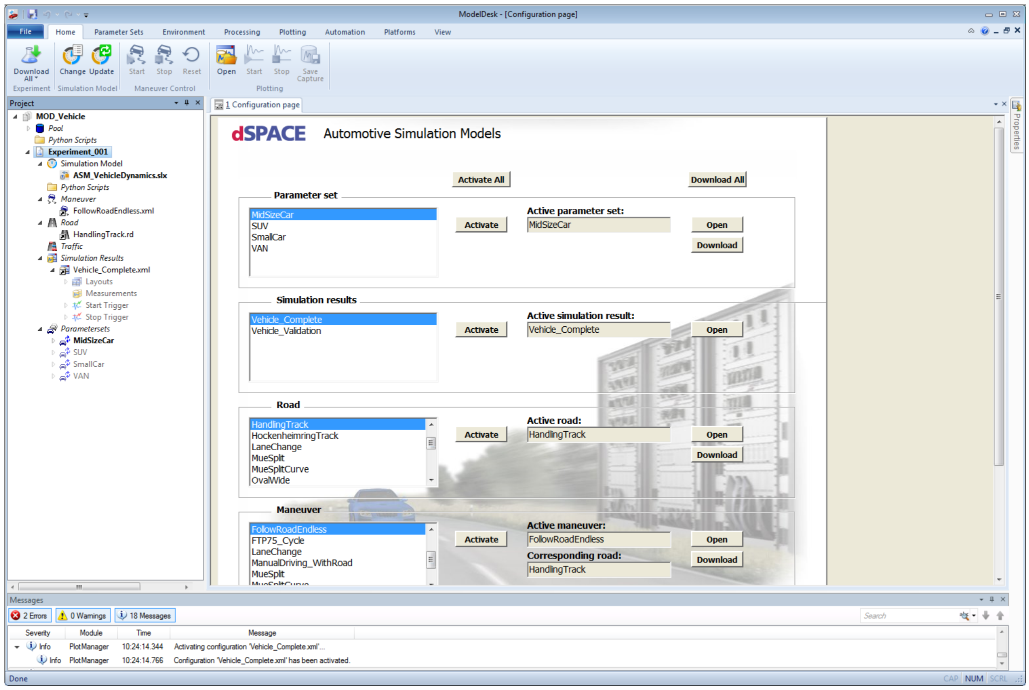


2 Activate the demo experiment via context menu on the experiment in the **Project Navigator**.



The project structure is available in the **Project Navigator** and all parameters are available for configuration.

3 Switch to the experiment's **Configuration** page.



The **Configuration** page of the experiment lets you select parameter sets for activating and downloading them to the simulation model or for activating and editing (**Open** button) them.

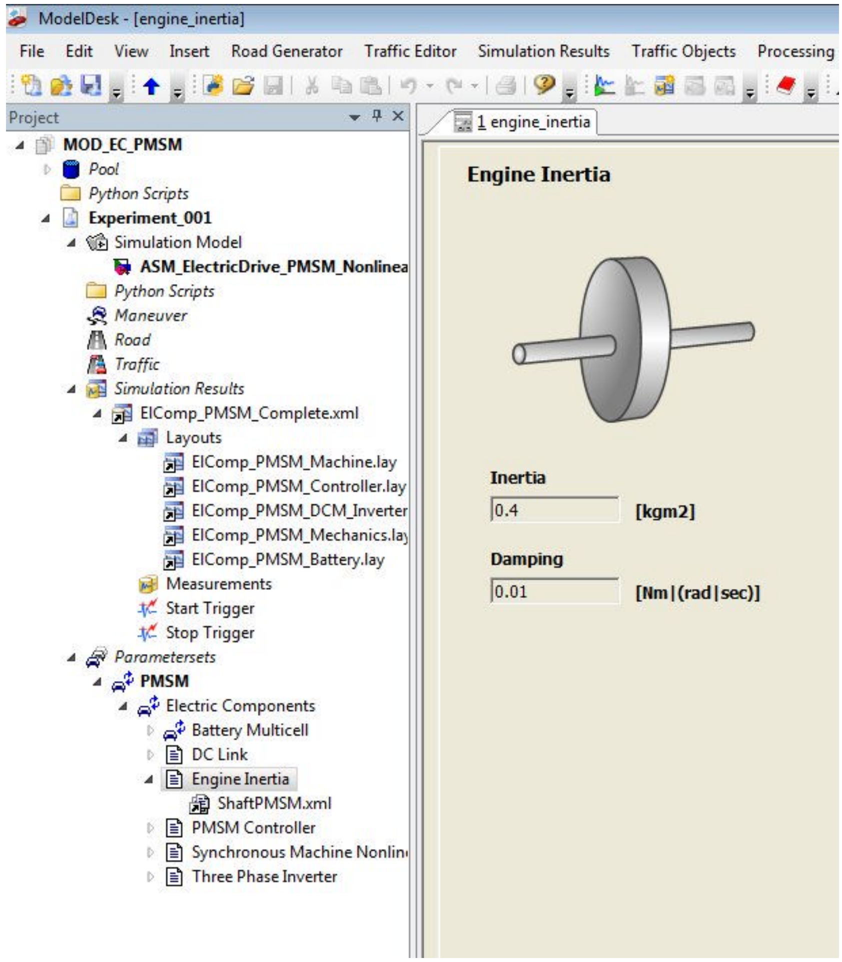
You can also activate plotting sets and open them to obtain simulation results in ModelDesk via the ModelDesk Plotting feature.

4 Click **Activate** and then **Download** to activate and download the parameter set.

5 Start a simulation (if possible in accelerator mode) and observe the results in the Simulink plotters, and/or ModelDesk plotting.

6 Modify, for example, the engine inertia in ModelDesk.

In the Project Navigator, select Parametersets – <ParameterSet name> – ElectricComponents – Engine Inertia.



7 Use the context menu of the current page to save and download the new setting or switch back to the main experiment page and download the current parameter setting.

8 Start a simulation (if possible in accelerator mode) and observe the results in the Simulink plotters, ModelDesk plotting, and/or MotionDesk.

9 Switch to the MATLAB workspace and look at the value of the engine inertia, `MDL.ElectricComponents.Mechanics.EngineInertia.Const_Inertia`. If you are working with the Simulink simulation, all parameters are available in the structure `MDL` in the MATLAB workspace and ModelDesk can download new values to the `MDL` structure.

```
>> MDL.ElectricComponents.Mechanics.EngineInertia.Const_Inertia
ans =
```

```
     Comment: 'Inertia'
      Author: '(null)'
      Origin: ' '
     Version: '2'
LastModified: '16.01.2015 11:21'
       vName: 'Const_Inertia'
       vUnit: '[kgm2]'
           v: [0.4000]
```

Result

You parameterized the model in ModelDesk.

Related topics

Basics

Working with Parameter Sets (ModelDesk Parameterizing 📖)

References

Settings (ModelDesk Road Creation 📖)

How to Select Maneuver Conditions

Objective

You can select different maneuver conditions.

Maneuver type

The following table describes the options for the maneuver type:

Maneuver Type	Description
Manual	To simulate the model manually in ControlDesk on a dSPACE platform or in Simulink on a standard PC.
Stimulus	To define a time-dependent stimulus maneuver to test the behavior of the model.
Test cycle	Contains the longitudinal driving and engine cycles, which can be defined by a MATLAB function that is part of your installation.

Maneuver state

Different states are possible for the Test cycle and Stimulus maneuver types.

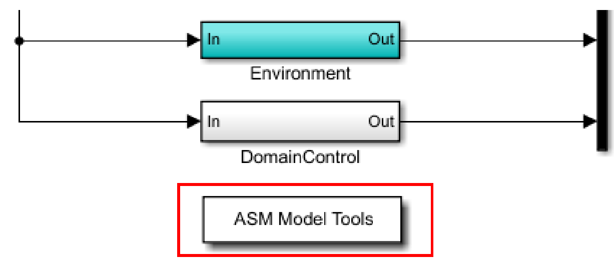
The following table describes the options for the maneuver state:

Maneuver State	Description
Stop	Maneuver stopped.
Start	Maneuver started.

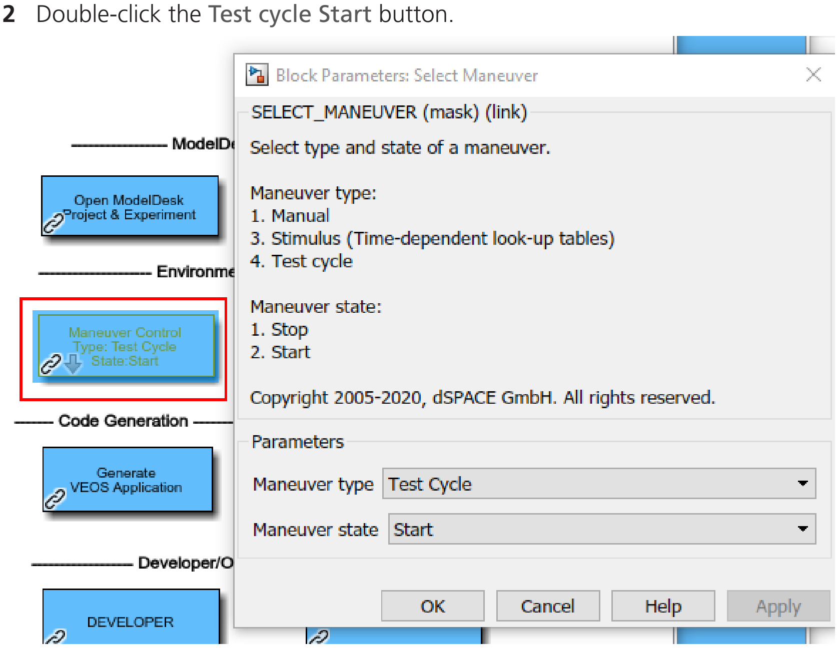
Method

To select maneuver conditions

1 In the ASM model, double-click the ASM Model Tools button below the modules.



2 Double-click the Test cycle Start button.



3 In the Simulink dialog, adjust the maneuver type and the maneuver state and click OK.

Result

You selected maneuver conditions.

Related topics

References

Select Maneuver (ASM User Guide 📖)

How to Select a Test Cycle

Objective

You can select different test cycles.

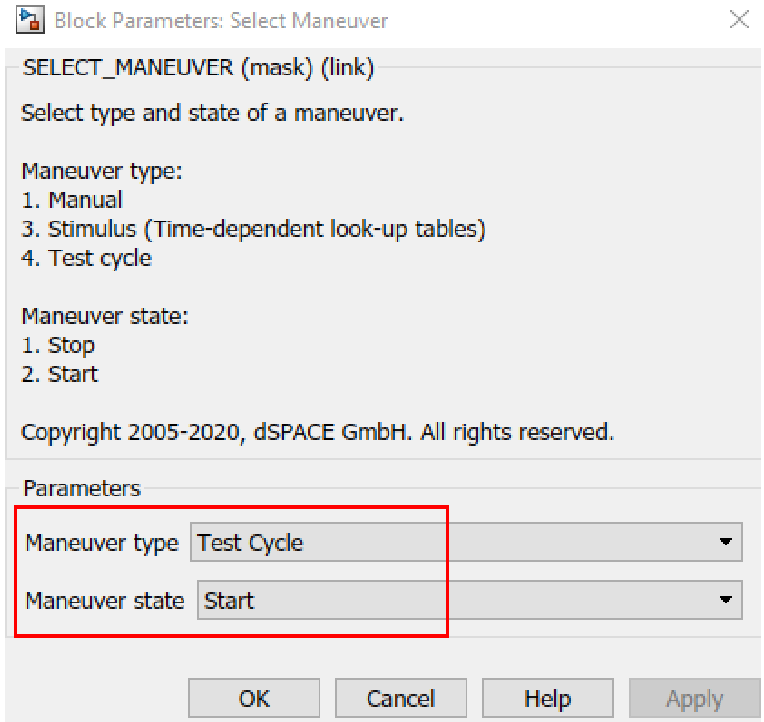
Description

The test cycles are stored in the ASM project at `<Projectfolder>\Simulation\IniFiles\DrivingCycles`. You can select a test cycle via a Simulink dialog.

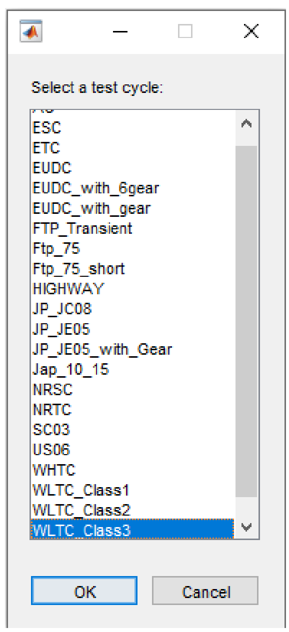
Method

To select a test cycle

1 In the ASM Tools, set the maneuver type to *Test Cycle* and the maneuver state to *Start*. Refer to How to Select Maneuver Conditions on page 22.



2 In the ASM Tools, double-click the Load Test Cycles button to open the Simulink dialog which lists all the driving cycles.



3 In the Simulink dialog, select a new test cycle and click OK.
4 Restart the simulation with the new test cycle.

Result

You selected the test cycle.

Related topics

References

Test Cycle (ASM User Guide 📖)

How to Plot Simulation Results in ModelDesk

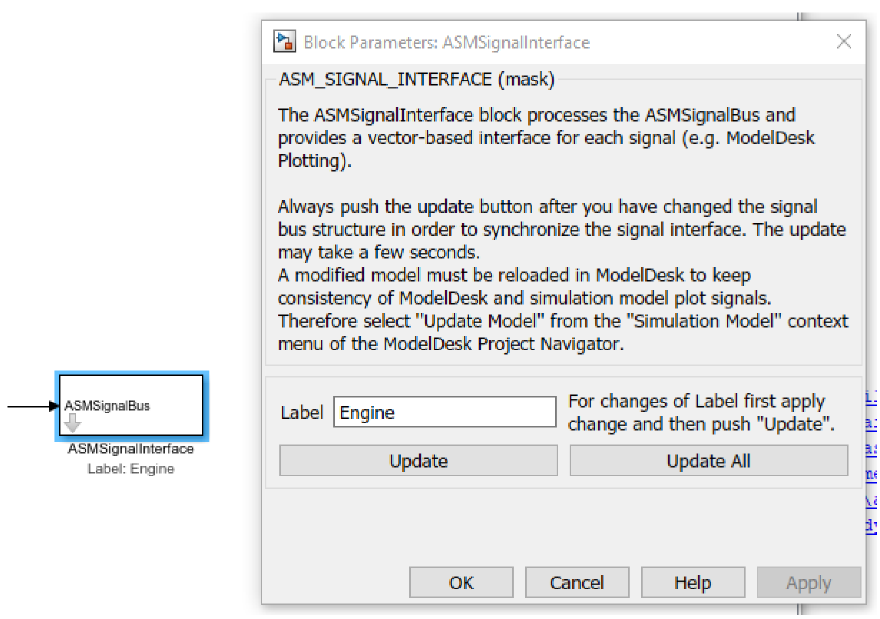
Objective

You can plot simulation results in ModelDesk via the SignalInterfaces.

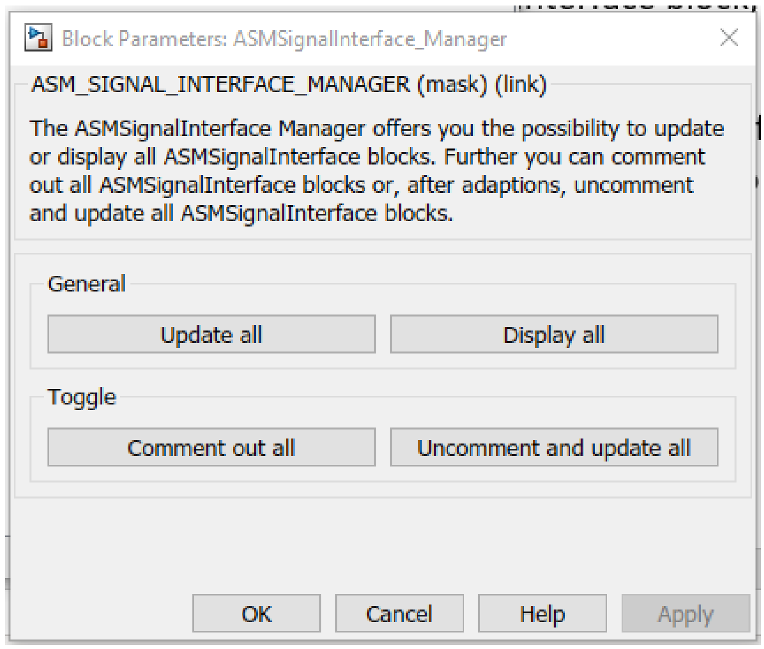
Description

You can use the ASMSignalInterface block and the ASMSignalInterface Manager to toggle the plotting.

ASMSignalInterface block Simulation data is transferred from the ASM model to ModelDesk via the ASMSignalInterface block. There is one block inside each ASM module. Refer to ASMSignalInterface (ASM User Guide 📖).



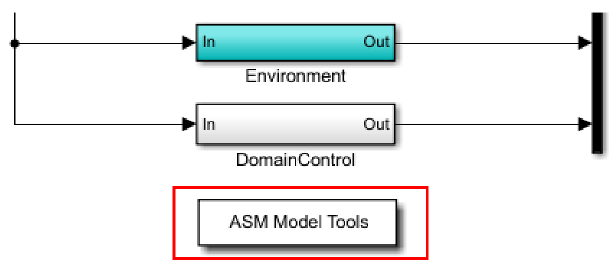
ASMSignal Interface Manager The ASMSignalInterface Manager collects the signals of all modules of your model. You can open it via the ASM Tools. Refer to ASMSignalInterface Manager (ASM User Guide 📖).



Method

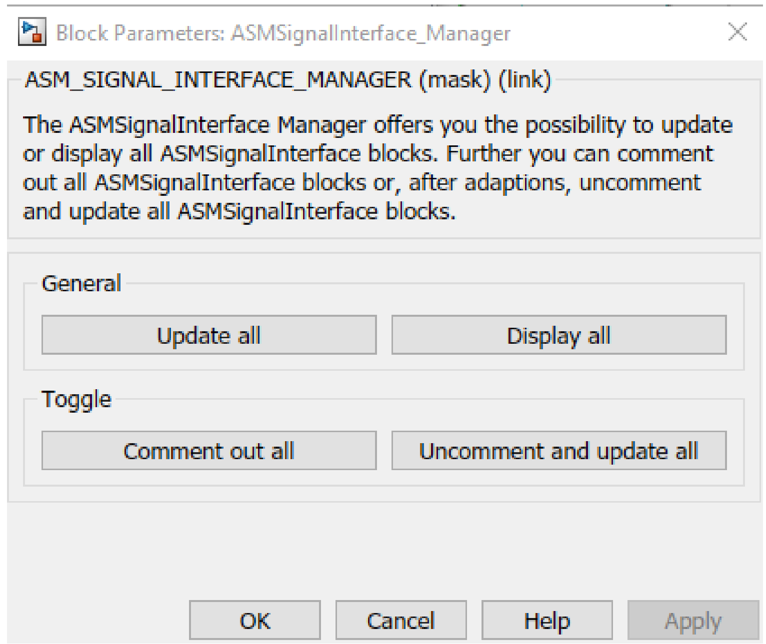
To plot simulation results in ModelDesk

1 In the ASM model, double-click the ASM Model Tools button below the modules.



2 Double-click the ASMSignalInterface Manager button and press one of the two buttons:

- Uncomment and update all: for plotting in ModelDesk.
- Comment out all: if you do not need plotting in ModelDesk. Commenting out the blocks will improve the simulation performance.



Result

You activated or deactivated plotting in ModelDesk.

Related topics

HowTos

How to Configure a Plotter in ModelDesk for a Simulink Simulation.. 39
How to Update a Plot Configuration for ModelDesk for a Simulink Simulation......................... 38

How to Switch the Library Type

Objective

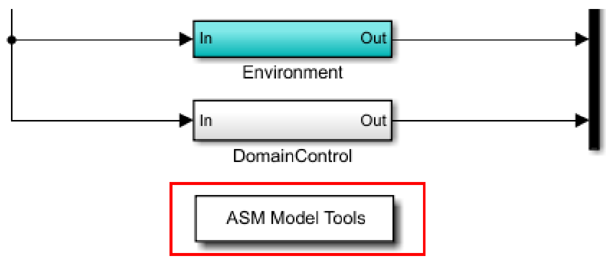
You can switch between the developer library and the operator library.

The developer library is a Simulink library containing the open-source ASM blocks. In the operator library, the ASM blocks are Simulink S-functions for offline simulation.

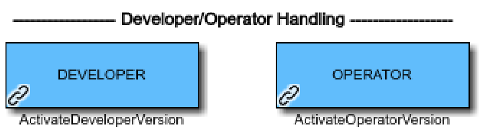
Method

To switch the library type

1 In the ASM model, double-click the ASM Model Tools button below the modules.



2 Depending on the library type, you want to switch to, double-click the DEVELOPER button or the OPERATOR button.



Result

You switched between the developer library and the operator library.

Related topics

References

Activate Developer Version (ASM User Guide 🕮)
Activate Operator Version (ASM User Guide 🕮)

How to Handle Multi-Instances

Objective

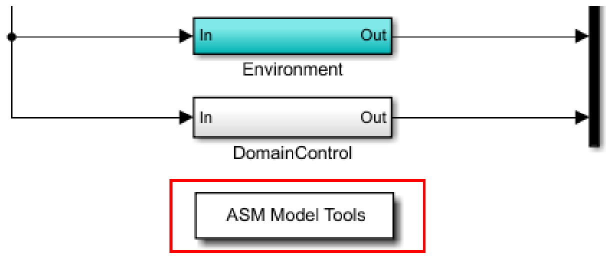
The Multi-Instance Overview button opens a dialog to get an overview of all multi-instance blocks in the model and to set the multi-instance parameters.

For more information on multi-instances, refer to Multi-Instance (ASM User Guide 🕮).

Method

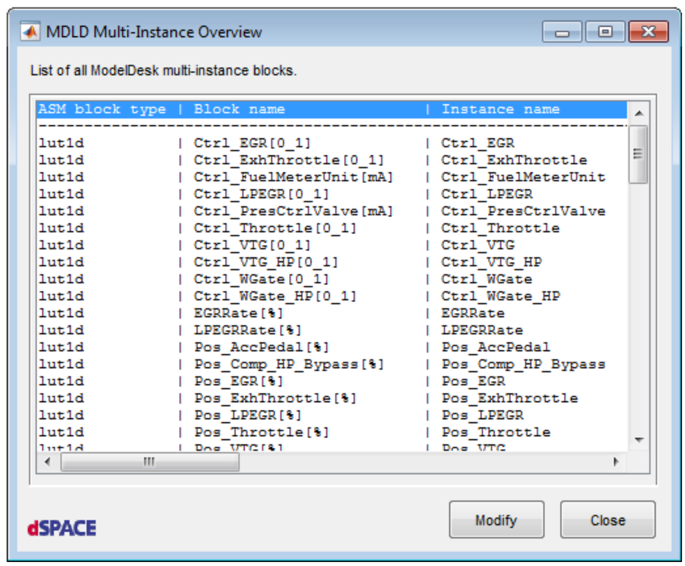
To handle multi-instances

1 In the ASM model, double-click the ASM Model Tools button below the modules.



2 Double-click the Multi-Instance Overview button.

A dialog opens that displays the multi-instances in your model.



3 To modify a multi-instance, select it in the dialog and click Modify.

Result

You modified multi-instances.

Related topics

Basics

Multi-Instance (ASM User Guide 📖)

How to Display the ModelDesk Parameter Group

Objective

You can modify the grouping in the parameter set that is displayed in the project tree of your ModelDesk experiment.

For more information, refer to Basics on Grouping the View of a Parameter Set (ModelDesk Parameterizing 📖).

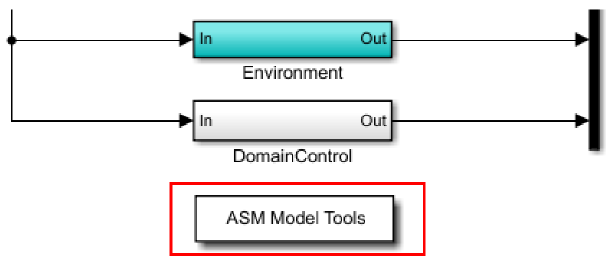
The ModelDesk Parameter Group Overview block in the ASM Tools opens an overview of the ModelDesk parameter group blocks that are contained in the simulation model.

For more information, refer to ModelDesk Parameter Group (ASM User Guide 📖).

Method

To display the ModelDesk parameter group

1 In the ASM model, double-click the ASM Model Tools button below the modules.



2 Double-click the ModelDesk Parameter Group Overview button.

Result

The MDLD Parameter Group Overview dialog opens. It displays all ModelDesk parameter groups.

Select a group in the list and click Open System to open the group in your model.

Related topics

Basics

Basics on Grouping the View of a Parameter Set (ModelDesk Parameterizing 📖)
ModelDesk Parameter Group (ASM User Guide 📖)

Simulating in Simulink

Where to go from here

Information in this section

How to Use the Simulink Accelerator Mode

Objective

You can use the Simulink accelerator mode to speed up the simulation (Pentium 4 Processor with 2 GB RAM: up to 2.5 times faster than real time).

Note

You can perform all steps in this tutorial by using the accelerator mode.

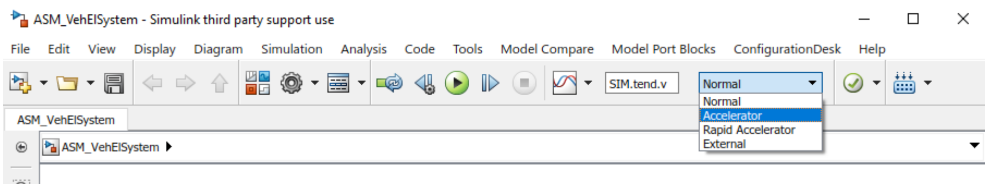
Prerequisite

You need a license for the Simulink Performance Tools.

Method

To use the Simulink Accelerator mode

1 In Simulink, activate the accelerator mode for the ASM demo model on the menu bar.



2 Start the simulation and observe the results on the Simulink plotters.

Result

The model is simulated in accelerator mode.

First, the build process for the accelerator mode is performed, which takes some time.

Whenever you perform an accelerator mode simulation and the model structure has not changed, no new code is generated, even if new parameters for the model are loaded, for example, from ModelDesk.

How to Simulate the Model Manually in Simulink

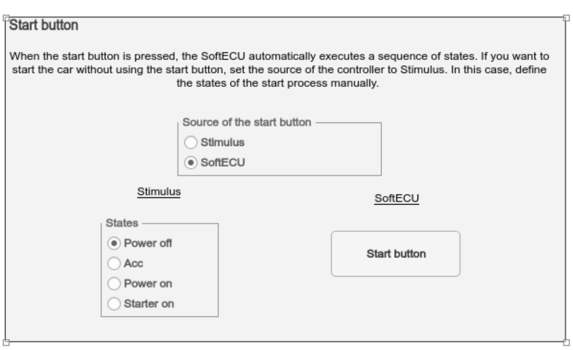
Objective

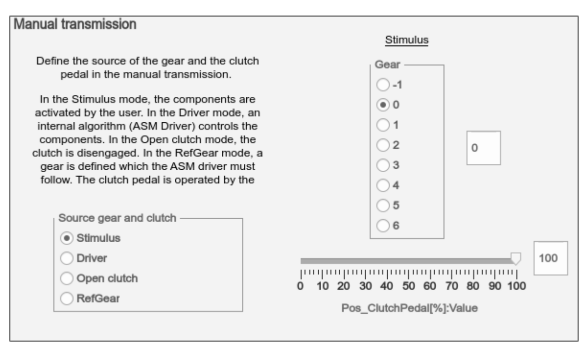
You can manually simulate your model in Simulink with prepared dashboard instruments that come with your ASM model.

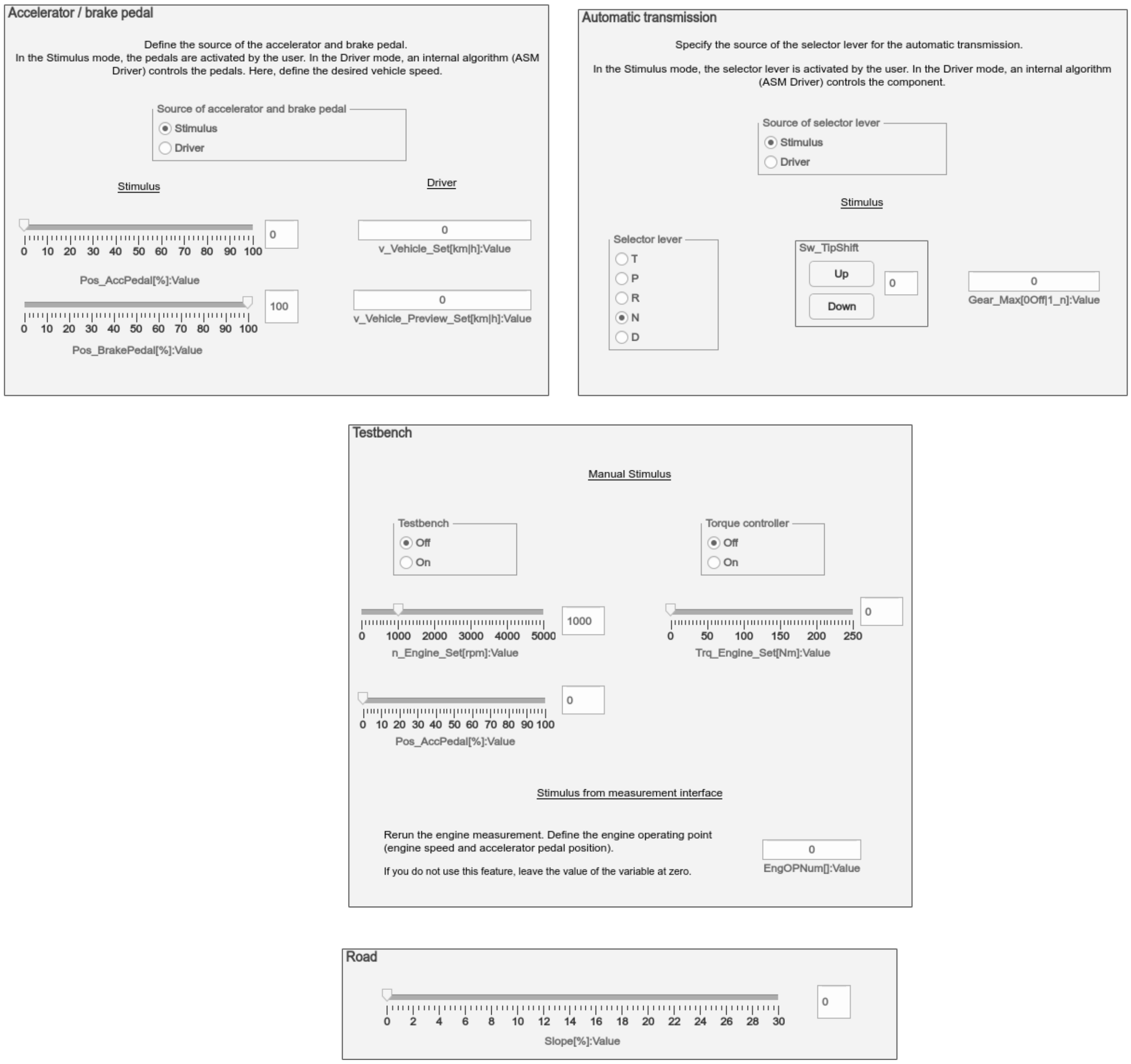
Dashboard instruments

You can find the dashboard instruments in: `/Environment/Plant/UserInterface/PAR_Plant/Manual_Controller`.

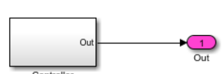
There are the following instruments:

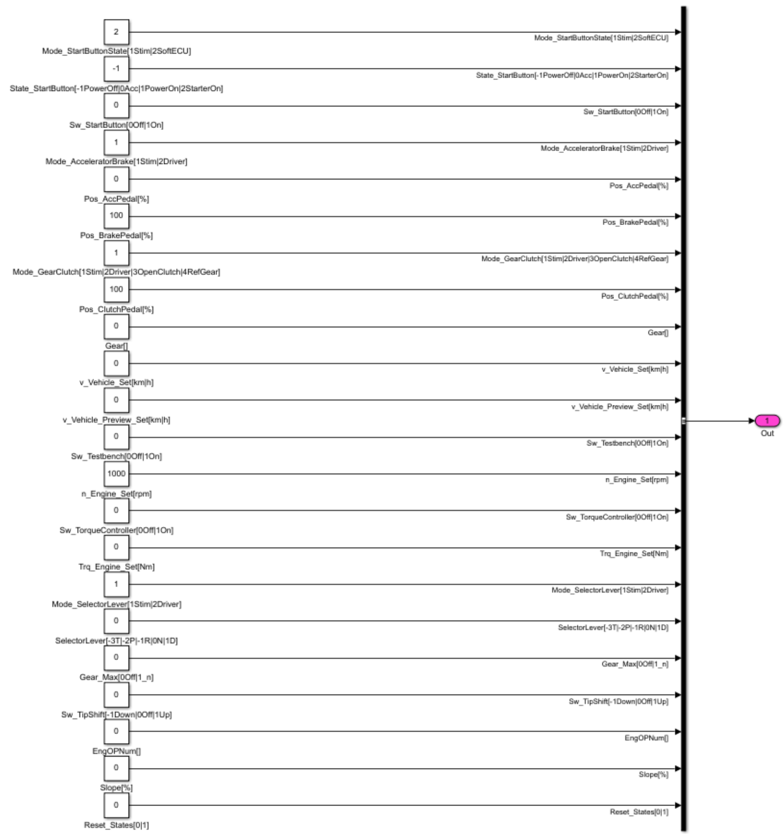






With these instruments, you can change the settings of the model controllers in `/Environment/Plant/UserInterface/PAR_Plant/Manual_Controller/Controller`.





Method

To simulate the model manually in Simulink

1. In the ASM Model Tools, set the maneuver type to *Manual*. This lets you run the model manually.
2. Start the simulation in Simulink.
3. Use the instruments on the dashboard to manipulate parameters in the model.

Result

You can now observe the results of the simulation in Simulink. Refer to How to Simulate the Model and Observe the Results in Simulink on page 36.

Related topics

HowTos

How to Simulate the Model and Observe the Results in Simulink

Starting simulation

The demo model is prepared for a Simulink simulation. When you start simulation after you opened the demo model, a standard vehicle dynamics maneuver or engine driving cycle starts.

Simulation results

You can observe the simulation results in Simulink using Simulink scopes. The scopes are in the UserInterface/DISP_xyz subsystem of an ASM module. For example, you can find a scope in the Engine/Control/UserInterface/DISP_Control subsystem.

You can also watch the simulation results in oversampled model parts. For more information, refer to Watching the Simulation Results in Oversampled Model Parts (ASM Gasoline Engine InCylinder Model Description 📖) or Watching the Simulation Results in Oversampled Model Parts (ASM Diesel Engine InCylinder Model Description 📖).

Method

To simulate the model and observe the results in Simulink

1. In MATLAB, switch to `<Projectfolder>\Simulation` in the project and call `go` to open the model.
2. Start the simulation.
3. Open one of the scopes from the UserInterface subsystem, for example, in Engine/Control/UserInterface/DISP_Control. After a few seconds, look at the scope signals.

Result

You simulated the model and observed the results in Simulink.

How to Handle Simulink Plotters

Objective

You can handle the Simulink plotters in your simulation model.

Handling Simulink Plotters

The SCOPE HANDLING GUI button opens a dialog that lets you perform the following actions:

- Disable: Comments out all Simulink plotters, displays, and xy-graphs in the model.
- Restore: Uncomments all Simulink plotters, displays, and xy-graphs in the model.
- Show: Shows a list with all the Simulink plotters, displays, and xy-graphs in the model.

The block makes directly use of the *commented* block property in Simulink. Disabled blocks are excluded from simulation and signals are terminated and grounded.

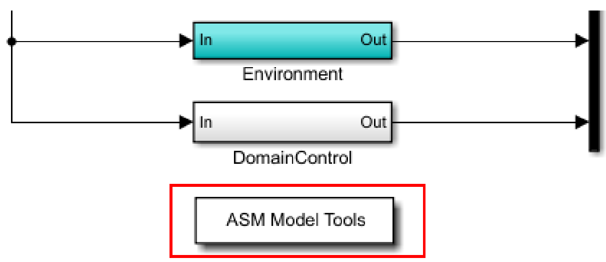
For ASM Engine Models: If you want to simulate your model with `asm_eng_testbench.m`, it is sufficient to store the simulation results in the variables under TEST_CONTROL or TEST_PLANT. No plotters are necessary.

The testbench script contains a postprocessing.

Method

To handle Simulink plotters

1 In the ASM model, double-click the ASM Model Tools button below the modules.



2 Double-click the SCOPE HANDLING GUI button.

The ASM Scope Handling GUI dialog opens.

3 In the ASM Scope Handling GUI dialog, click one of the buttons to handle Simulink plotters.

For information on the options, refer to Handling Simulink Plotters on page 36 above.

Result

You handled the Simulink plotters in your simulation model.

Related topics

References

Scope Handling User Interface (ASM User Guide 📖)

How to Update a Plot Configuration for ModelDesk for a Simulink Simulation

Objective

The ASM package includes a signal interface block for ModelDesk so that signals of the ASMSignalBus can be plotted. This block must be updated manually if the ASMSignalBus is changed.

If the respective simulation model is an operator version, it is mandatory to perform a manual update of the plotting block and plotting configuration, before using the plotting feature.

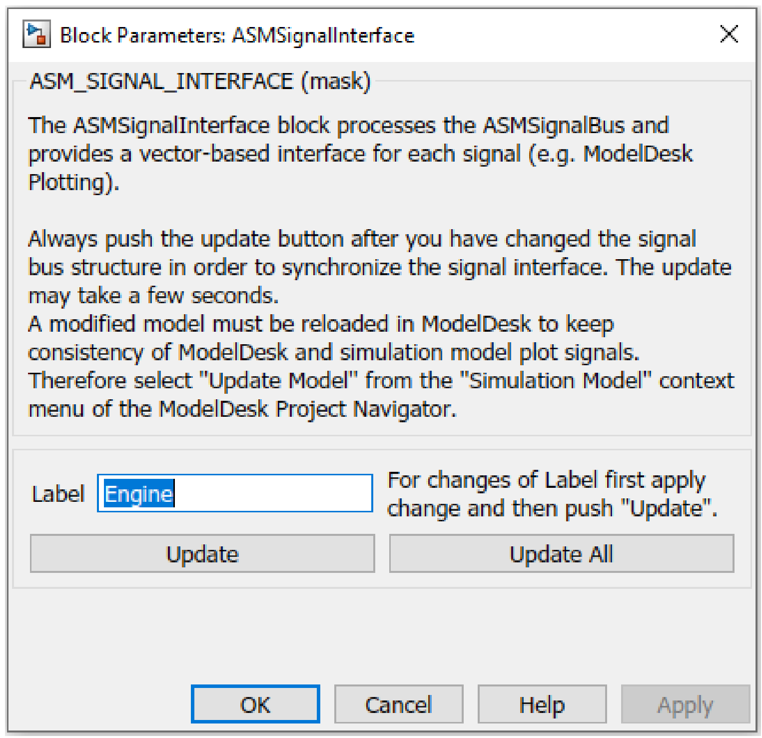
Method

To update a plot configuration for ModelDesk for a Simulink simulation

1 Inside an ASM module, open the SignalInterface subsystem.

2 Double-click the ASMSignalInterface block.

The following dialog appears with varying labels, depending on your model.

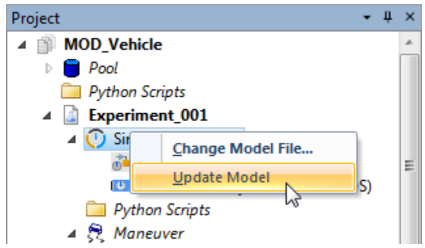


3 Click:

- Update: To update only the current block.
- Update all: To update all blocks in the model that are currently not commented out.

All the ASMSignalInterface blocks are updated. This also changes the model, which you must update in ModelDesk.

4 In ModelDesk's Project Manager, right-click the Simulation model element and select Update Model.



Result

The plot configuration is updated.

Related topics

Basics

Collecting Signals for Plotting in the ASM Model (ModelDesk Plotting 📖)

References

Signal Selector (ModelDesk Plotting 📖)

How to Configure a Plotter in ModelDesk for a Simulink Simulation

Objective

To plot a signal from the ASMSignalBus, you have to configure a project and an experiment in ModelDesk.

The `<Projectfolder>\Parameterization\MOD_<Blockset>` folder includes a complete ModelDesk project. You can completely parameterize the engine in ModelDesk. The demo project includes examples of typical plotter signals.

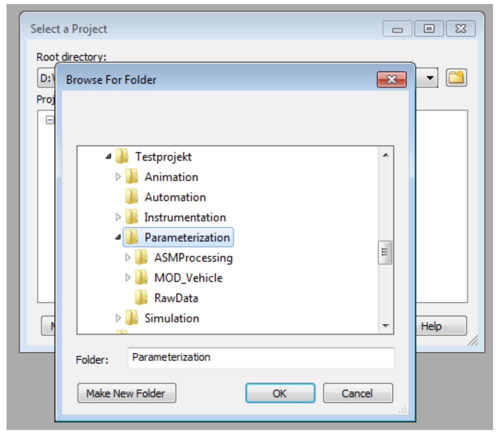
Method

To configure a plotter in ModelDesk for a Simulink simulation

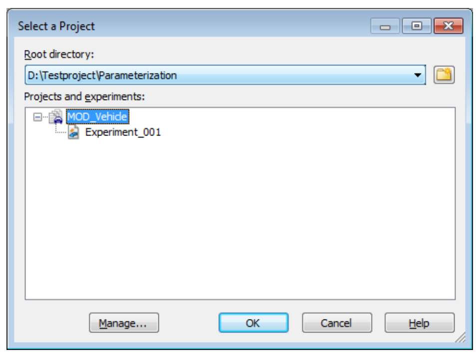
1 Start ModelDesk and open the provided ModelDesk project from `<Projectfolder>\Parameterization`.

1. Click File – Open - Project + Experiment.

2. Select the `<Projectfolder>/Parameterization` folder.

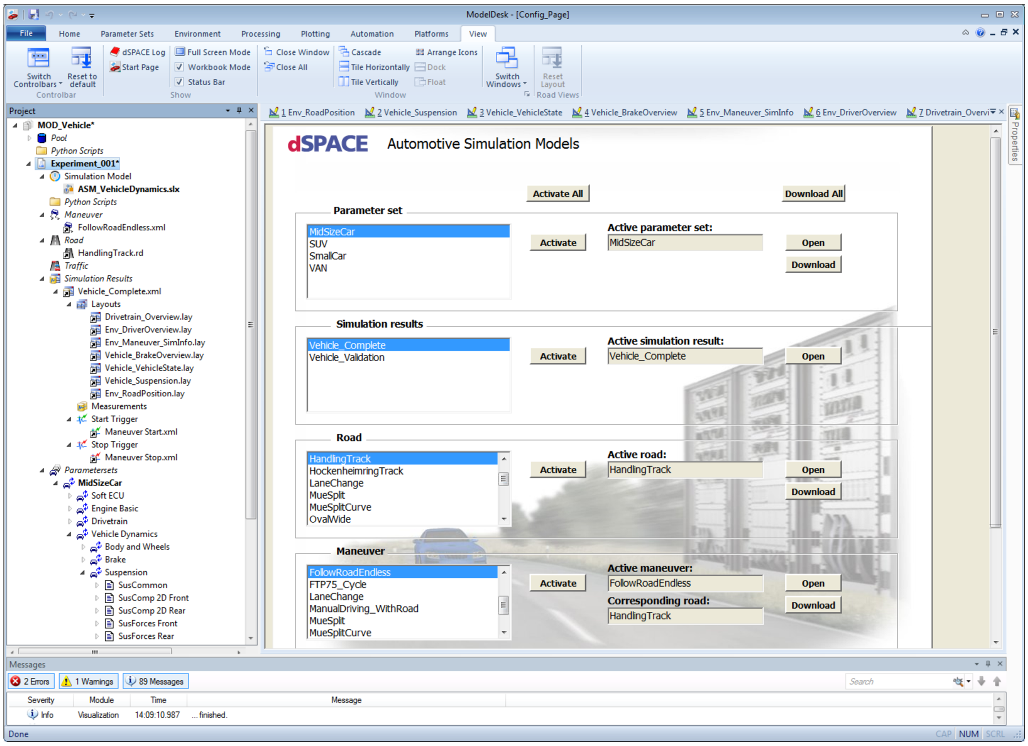


3. Select an experiment from your ModelDesk folder.

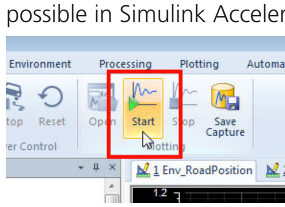


2 Switch to the experiment's Configuration page.

3 Select a plot configuration from the **Simulation results** list, click **Activate** and **Open**.



4 To start a simulation, go to the **Home** ribbon and click **Plotting – Start** (if possible in Simulink Accelerator mode).



Result

ModelDesk starts the simulation and plots the selected signals. You can observe the results in the ModelDesk plotters.

Related topics

Basics

Plotting Signals (ModelDesk Plotting 📖)

Simulating on dSPACE Platforms

Where to go from here

Information in this section

How to Generate an OSA File for VEOS

Objective

You can generate an Offline Simulation Application (OSA) for VEOS either manually or automatically.

Model preparation for VEOS

If the model has an oversampled engine part, it must be implemented as a Simulink For-Iterator subsystem for VEOS simulation. This 'offline' mode is automatically inserted if the model is opened with `go ('simmode', 'PC')`. For more information on the different oversampling strategies, refer to Basic Model Concept (ASM Gasoline Engine InCylinder Model Description 🕮).

Possible methods

There are two ways to generate code for VEOS:

- Manually. Refer to Method 1 on page 43.
- Automatically. Refer to Method 2 on page 43.

Method 1

To generate code for VEOS manually

1. Open the model with `go ('simmode','CPT','platform','VEOS')`. This selects the system target `dsrt.tlc` file provided by the Model Interface Package for Simulink.
2. Generate code for the ASM model.
   The Model Interface Package for Simulink generates a Simulink implementation container (SIC) file containing ASM model code.
   For instructions, refer to Generating Simulink Implementation Containers (Model Interface Package for Simulink - Modeling Guide 📖).
3. Create a new OSA file:
   1. Start the VEOS Player.
   2. On the **Home** ribbon, select **New** to create an empty OSA file.

   For instructions, refer to Basics on the VEOS Player (VEOS Manual 📖).
4. Import the SIC file to the VEOS Player to integrate the model in an Simulink simulation application for VEOS.
   The VEOS Player builds the OSA for simulation on VEOS.

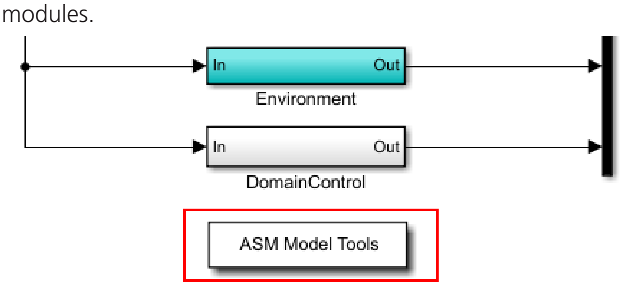
   > **Note**
   >
   > ASM does not support a 32 bit version of VEOS. Make sure to select **HostPC64** as simulation target on the **Build Options** tab during import to the VEOS Player.
   > You can also set **HostPC64** as default simulation target via the Simulation Target Manager in VEOS.

   For instructions, refer to How to Import Simulink Implementations (VEOS Manual 📖).

Method 2

To automate code generation for VEOS

1. Open the model in MATLAB.
2. In the ASM model, double-click the **ASM Model Tools** button below the modules.



3. Double-click the **Generate VEOS Application** button.
4. A **Code Generation** confirmation prompt opens. Click **Yes** to start code generation.

Result

You have generated code for VEOS.

Related topics

References

Generate VEOS (ASM User Guide 📖)

Generating a Real-Time Application for PHS-Bus-Based Platforms

Introduction

If you have a dSPACE Real-Time Interface (RTI) license, you can generate a new real-time application for the ASM. This is needed if you want to modify the model, for example, by adding I/O blocks for HIL simulation, or to have different initial settings for the model (engine settings).

Note

RTI code generation of ASM models is currently not supported if the *Accelerator* simulation mode is selected.

For real-time simulation, the model must be adapted. The model can be switched to online (real-time) mode by opening it with the following:

```
go('simmode','HIL', 'platform','RTI')
```

Note

Make sure that the MATLAB current folder is in the Simulation folder of your project, `<Project>\Simulation`

If the online mode is activated, code generation can start. This can be done via the **Simulink Preferences** or with **Ctrl+B** when clicking a Simulink model.

Code is always generated in the current folder.

To generate real-time code to another folder, change the MATLAB current folder after running the **go** file.

Generating code for PHS-bus-based platforms

For information on how to generate code, refer to Building and Downloading the Model (RTI and RTI-MP Implementation Guide 📖).

Generating the Real-Time Application for SCALEXIO

Generating code for SCALEXIO

Generation of the real-time application has to be started from ConfigurationDesk. For information on how to set up a ConfigurationDesk project and start code generation, refer to Building Real-Time Applications (ConfigurationDesk Real-Time Implementation Guide 📖).

How to Observe Results in ControlDesk

Objective

When simulating the model, you can observe the results in ControlDesk.

The ASM demo model contains preconfiguered real-time application files for the SCALEXIO and VEOS platforms. You can use these files to observe simulation results with the preconfigured ControlDesk experiment.

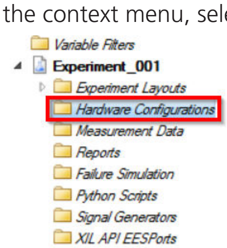
Method

To observe results in ControlDesk

1 Start ControlDesk.

On the ribbon, click File – Open – Open Project and Experiment to open the preconfigured ControlDesk experiment. Select <Projectfolder>\Instrumentation\CD_<Blockset>\CD_<Blockset>.CDP.

2 In ControlDesk, right-click Hardware Configurations in the project tree. On the context menu, select Add Platform/Device to add the platform.



Select your simulation platform and click Next. Import the prebuilt real-time application from <Projectfolder>\Simulation\RealTimeObjects\<platform>\ASM_<Blockset>.sdf.

3 On the Platforms ribbon, select Register Platforms to register your platform.

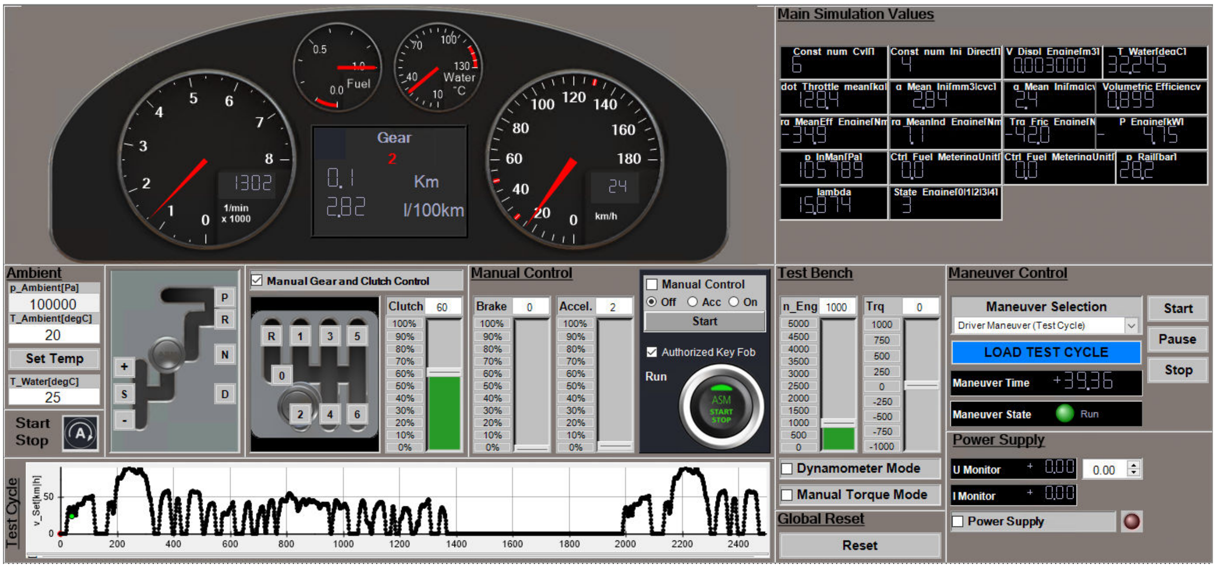
4 Open the platform's context menu.

Open the predefined layout in the Experiment Layouts folder in the project tree of the ControlDesk experiment.

5 Save the experiment.

On the Home ribbon, select Go Online to start online calibration.

The application runs on the selected platform. You can observe the Simulation results on the available layouts.



6 Use the instruments in the dashboard layout to control the maneuver, to start and stop the engine and to observe variables such as vehicle speed and engine speed.

You can also select whether you would like to drive with the ASM driver model, with stimulated input signals, or with a manual control, by changing the maneuver selection. In real-time simulation, you have to set the pedal positions and the gear switch yourself, for example, by using the instruments of the Manual Control area in the dashboard layout.

Depending on the ASM used, you can use further layouts to observe engine or vehicle dynamics signals.

Result

You observed results in ControlDesk.

Note

If you want to update the real-time application, note the following points: For RTI and VEOS, the real-time code is generated and saved in the MATLAB working directory, which is typically the `Simulation` folder. When you generate the code repeatedly in the same directory, ControlDesk detects the update and prompts you to reload the real-time application when going online.

How to Update a Plot Configuration for ModelDesk for a dSPACE Platform

Objective

The ASM package includes a signal interface block for ModelDesk so that signals of the ASMSignalBus can be plotted. This block must be updated manually if the ASMSignalBus is changed.

If the respective simulation model is an operator version, it is mandatory to perform a manual update of the plotting block and plotting configuration, before using the plotting feature.

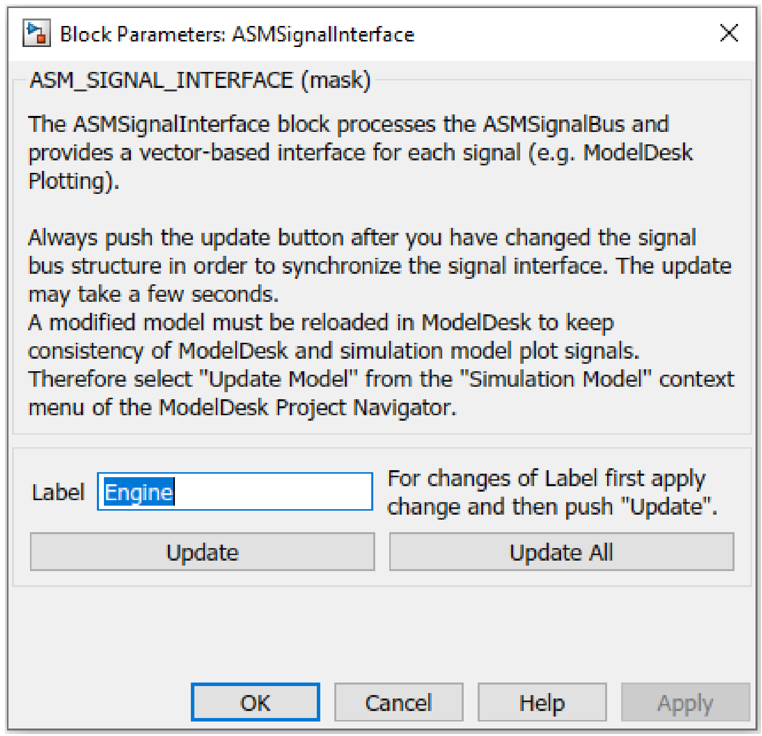
Method

To update a plot configuration for ModelDesk for a real-time simulation

1 Inside an ASM module, open the SignalInterface subsystem.

2 Double-click the ASMSignalInterface block.

The following dialog appears with varying labels, depending on your model.



3 Click:

- Update: To update only the current block.
- Update all: To update all blocks in the model that are currently not commented out.

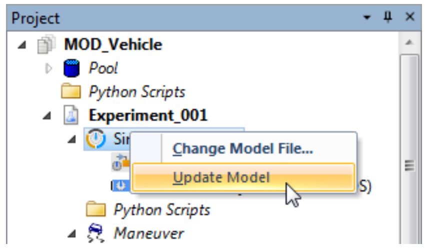
All the ASMSignalInterface blocks are updated. This also changes the model, which you must update in ModelDesk.

4 Start a new code generation process.

You must update the simulation model in ModelDesk to make all changes in the ASMSignalBus visible to ModelDesk.

Make sure your experiment is linked with the correct real-time model SDF file. ModelDesk shows the active model in bold.

5 In ModelDesk's Project Manager, right-click the Simulation model element and select Update Model.



Result

The plot configuration is updated.

Related topics

Basics

Collecting Signals for Plotting in the ASM Model (ModelDesk Plotting 📖)

References

Signal Selector (ModelDesk Plotting 📖)

Changing the Simulation Platform

Introduction

You can change the simulation platform and reuse ControlDesk experiments. For how to change from VEOS to SCALEXIO, for example, refer to Switching the Simulation Platform and Reusing Experiment Parts (ControlDesk Platform Management 📖).

Changing between other platforms works similarly.

Adapting the Model to Special Needs

Where to go from here

Information in this section

Introduction to Different Model Variants

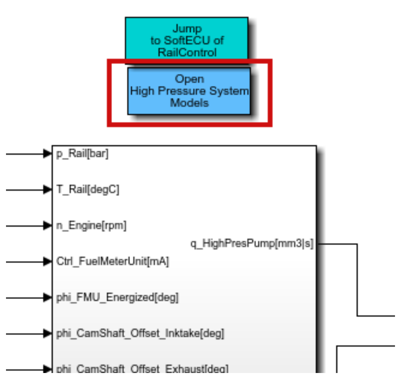
Basics

In the ASM engine models, some components come with different variants, for example, the high-pressure pump of the common rail system.

There are two variants of the high-pressure pump model:

- A current-based high-pressure pump: Here, the fuel flow is controlled by the amperage.
- A crank-angle-based high-pressure pump: Here, the fuel flow is controlled by the timing of the energization of the fuel metering unit.

You can select these variants in the ASM Diesel Engine Model via **Open High-Pressure System Models** in the **FuelSystem** subsystem.



Another example of model variants in the ASM Diesel Engine Model is the Low Pressure EGR.

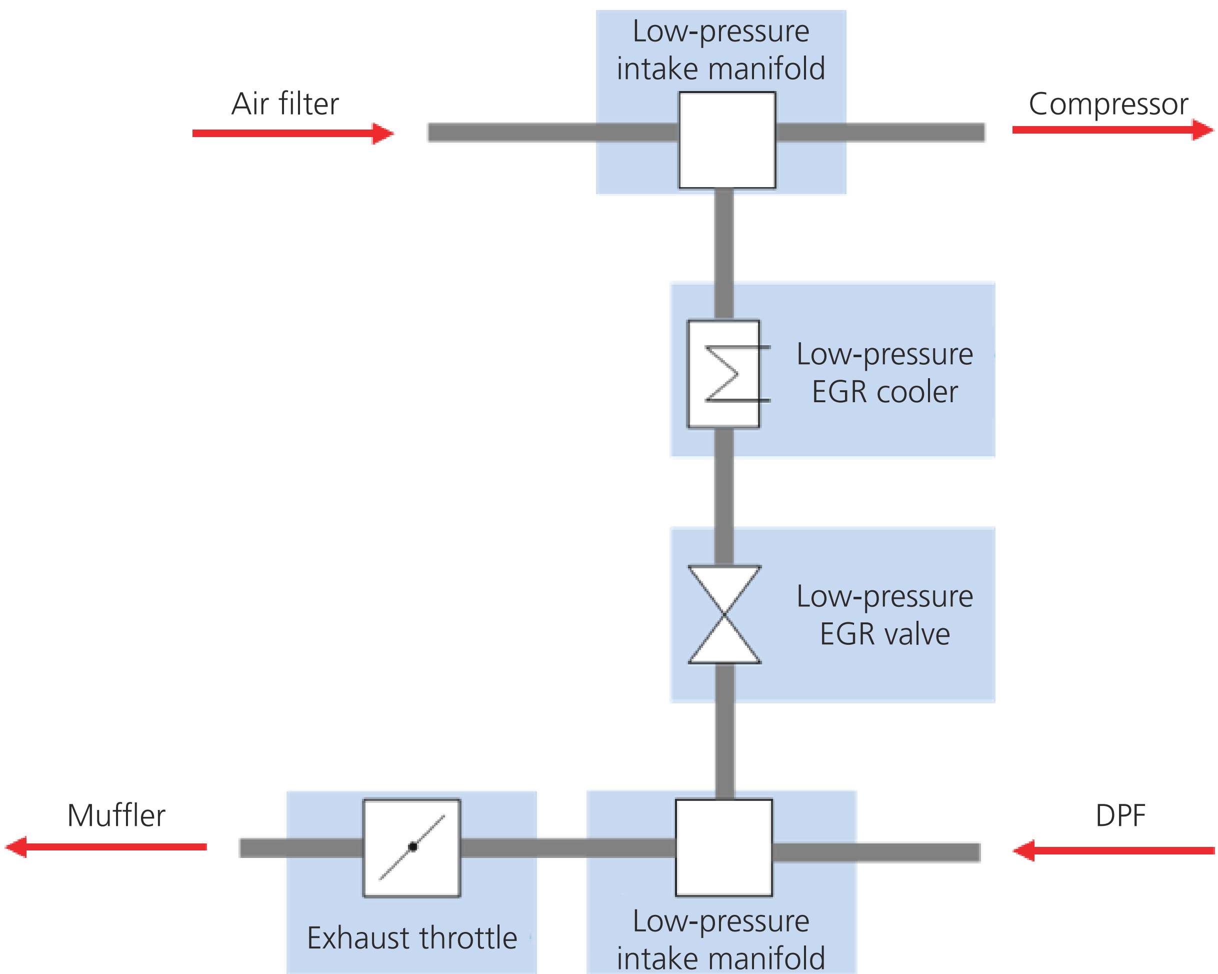
How to Integrate the Low-Pressure EGR Demo Model

Objective

The ASM Engine Diesel library contains a model of a low-pressure EGR system. The model is not a part of the ASM Engine Diesel demo model but it can be easily integrated into the model.

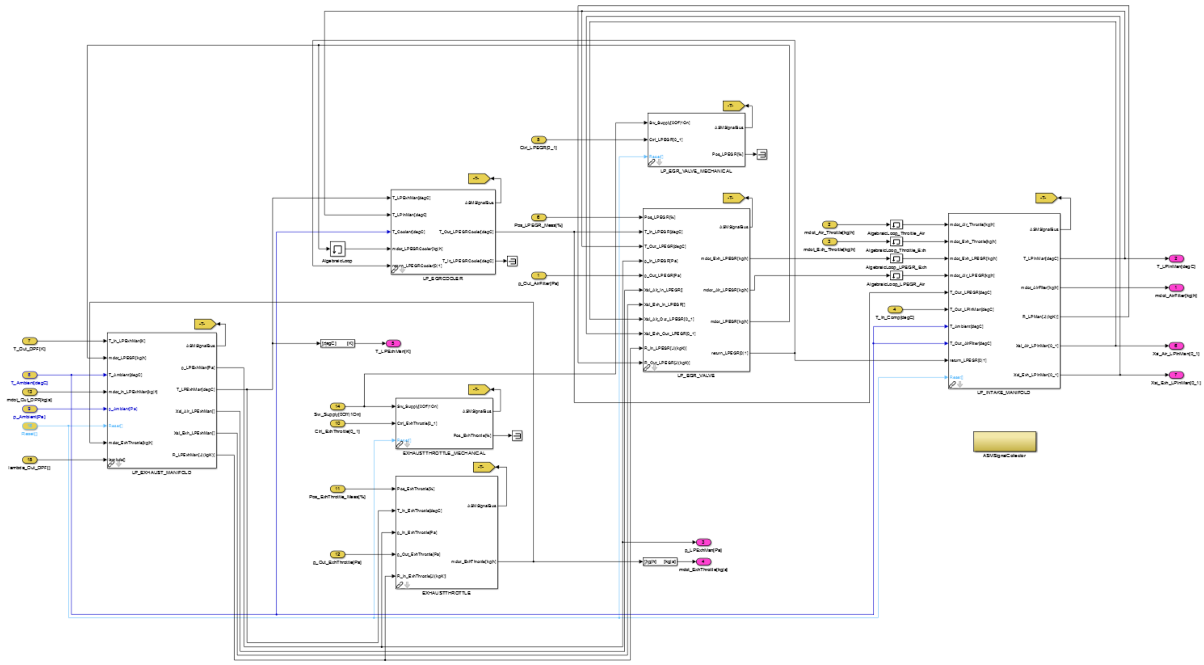
Description

The following illustration displays a schematic of the low-pressure EGR model.



The exhaust gas for the low-pressure EGR is taken after the DPF, cooled and mixed with the fresh air. This gas is then compressed, cooled and fed to the engine. To increase the rate of the low-pressure EGR, an exhaust throttle is used.

The following illustration shows the Simulink implementation of the low-pressure EGR demo model of ASM Diesel Engine.



The low-pressure EGR demo model consists of the following:

- Low-pressure EGR cooler
- Low-pressure EGR valve
- Mechanical low-pressure EGR valve
- Low-pressure EGR intake manifold
- Low-pressure EGR exhaust manifold
- Exhaust throttle
- Mechanical exhaust manifold

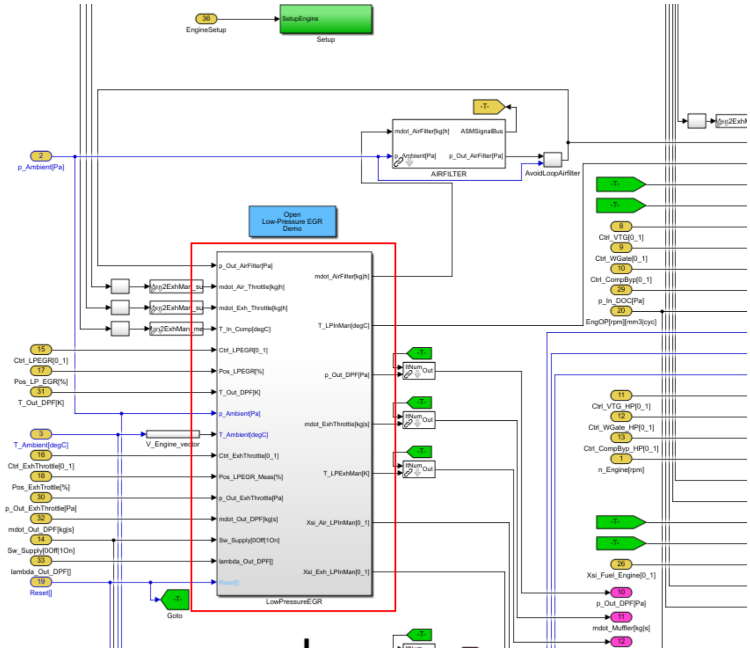
Method

To integrate a low-pressure EGR system into the ASM Diesel Engine model

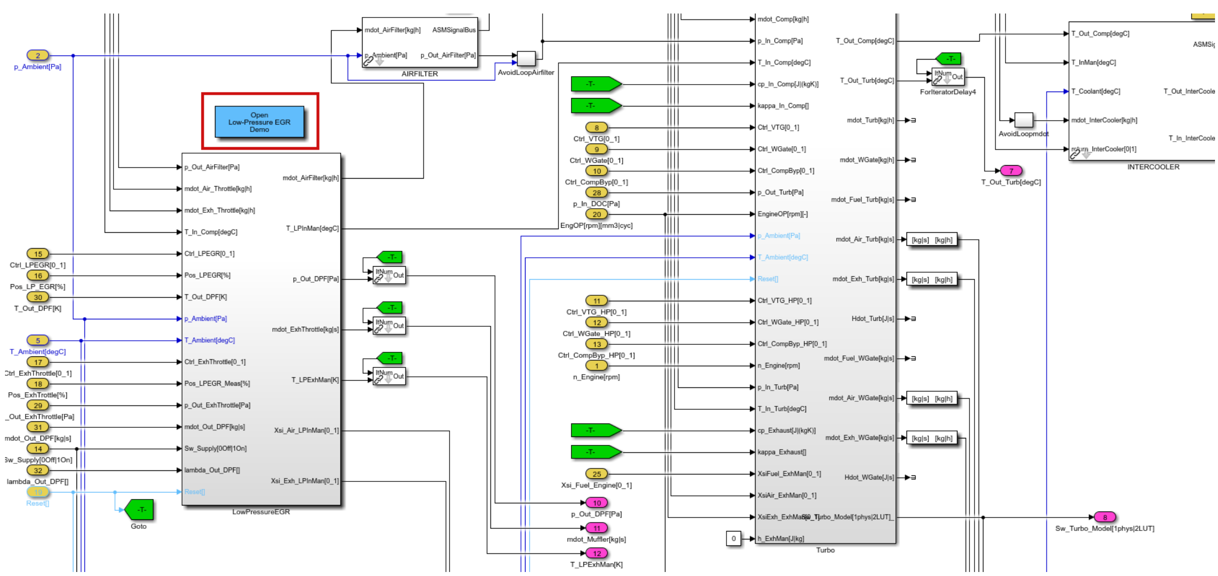
1 Open the ASM Diesel Engine demo model.

2 In the demo model, go to Engine/Plant/EngineDiesel/AirPath.

The model already contains a placeholder for the low-pressure EGR model.



3 Double-click the Open Low-Pressure EGR Demo button above the LowPressureEGR block.



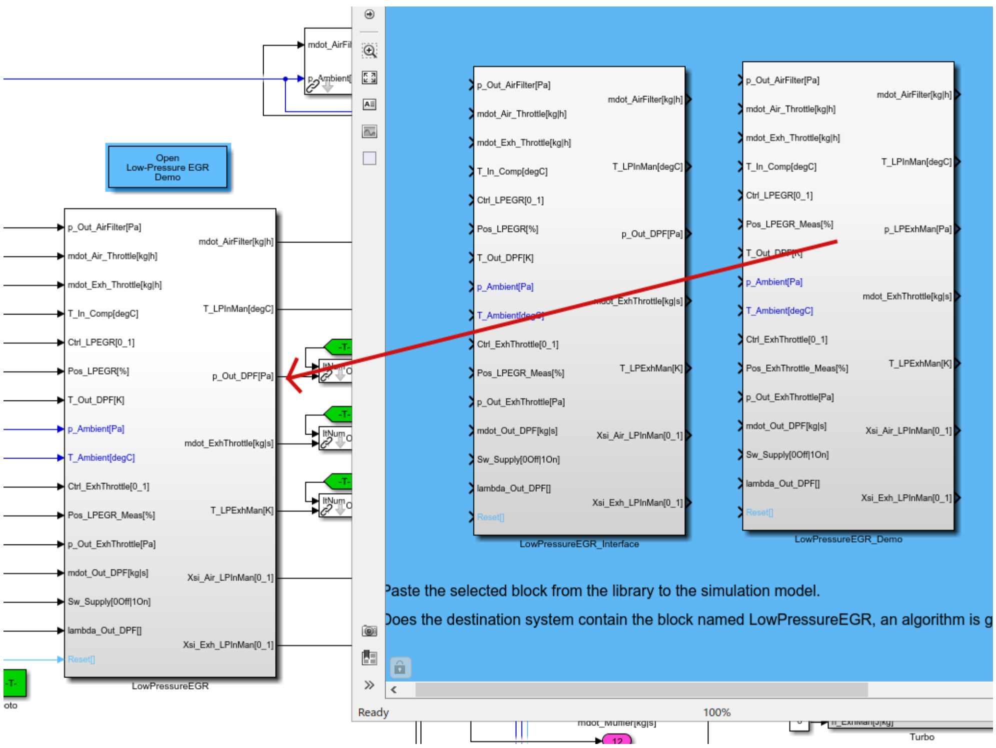
The LP_EGR_Models library opens. It includes two subsystems:

- LowPressureEGR_Interface: placeholder for the low-pressure EGR
- LowPressureEGR_Demo: low-pressure EGR demo model

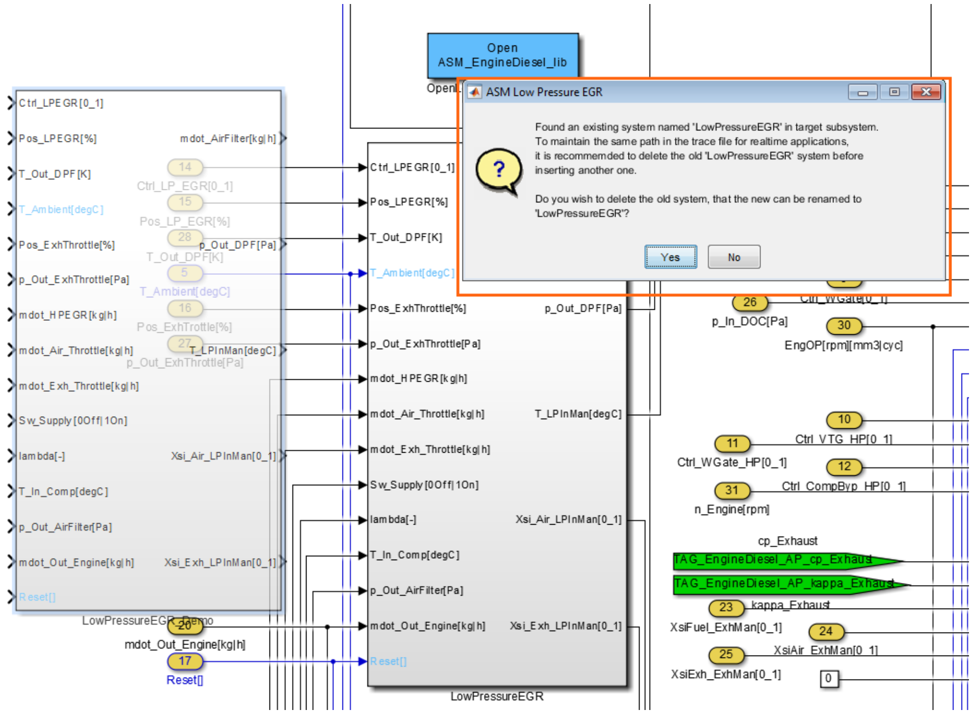
Note

The EGR low-pressure system models are also available via the ASM Diesel Engine Library:
ASM_EngineDiesel_lib/DieselEngine/AirPath/LP_EGR/LP_EGR_Models

4 Use drag & drop to add the LowPressureEGR_Demo block from the library to the demo model.



5 A confirmation prompt opens.



Click **Yes** to replace the **LowPressureEGR** block.

Result

The existing **LowPressureEGR** block is replaced by the new block. The new block is renamed.

The ASM project initially contains no parameters for the new components.

To get the parameters:

1. Save the model
2. Open your ModelDesk experiment
3. Update your model in the ModelDesk experiment

When ModelDesk parses the model, it finds the new components in the model and creates the corresponding parameter pages with the default values. Adjust these parameter values to your engine.

Then you have the following options in ModelDesk:

- To download the parameters to your simulation platform.
- To generate initialization files.

Related topics

References

Air Path (ASM Diesel Engine Reference 🕮)

How to Integrate the High-Pressure Pump Model

Objective

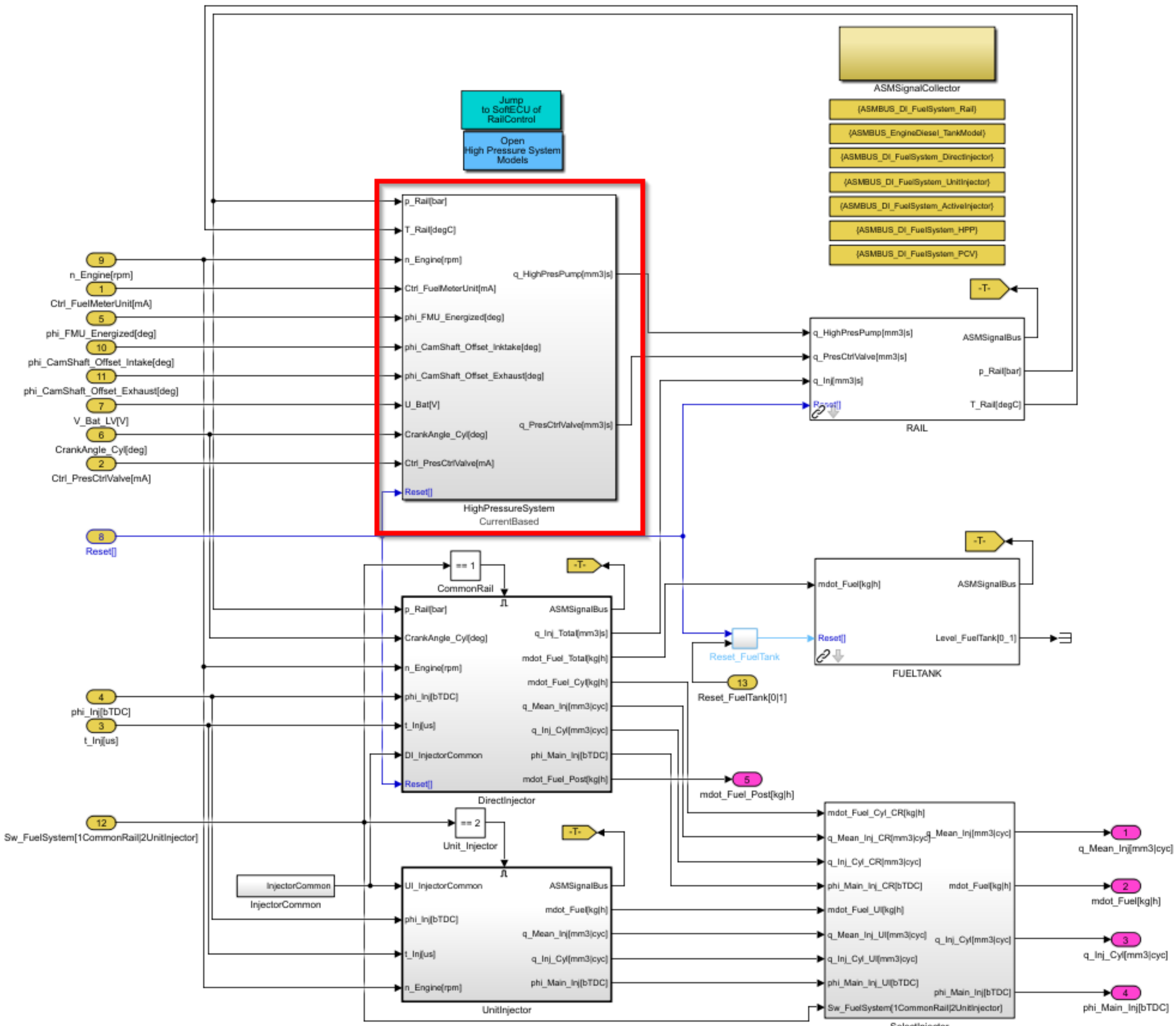
The ASM Engine Models contain two different models of the high-pressure pump. The models differ in the way how they are controlled.

- One type is the current-based high-pressure pump. Refer to High-Pressure Pump (ASM Diesel Engine Reference 🕮). Here, the amount of electricity (current) controls the flow of fuel.
- The second pump is a crank angle-based high-pressure pump. Refer to High-Pressure Pump (Crank-Based) (ASM Diesel Engine Reference 🕮). Here, the timing of energization defines the flow rate of the high-pressure pump.

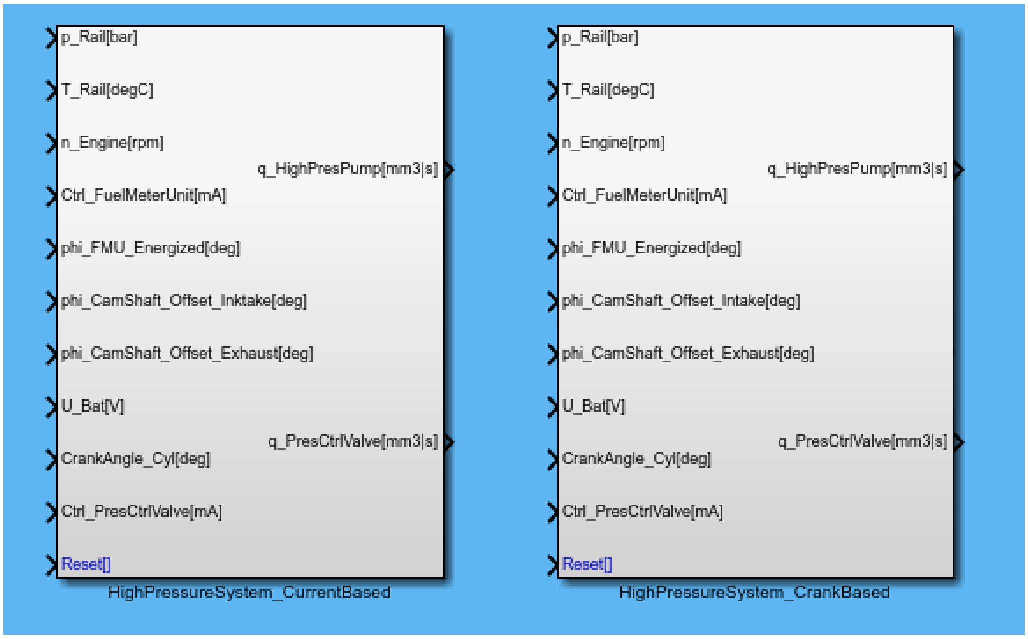
Method

To integrate the high-pressure pump model

1 In your ASM Diesel Engine Demo Model, open the Fuel System in Engine/Plant/EngineDiesel/FuelSystem.



2 Double-click Open High-Pressure System Models above the HighPressureSystem block.



The library containing the HighPressureSystems opens.

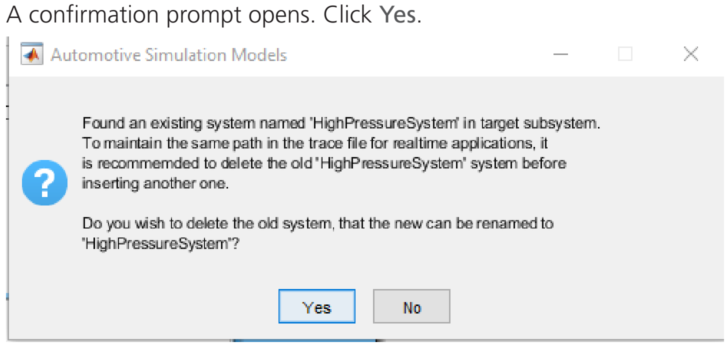
- The HighPressureSystem_CurrentBased model consists of the HIGH_PRESSURE_PUMP and PRESSURE_CONTROL_VALVE blocks.
- The HighPressureSystem_CrankBased model consists of the HPP_CRANKBASED block.

Tip

The HighPressureSystem models are also available via the ASM Diesel Engine Library: ASM_EngineDiesel_lib/DieselEngine/FuelSystem/HighPressurePump/HighPressurePump_Models.

3 Use drag & drop to add the required high-pressure pump model from the library to the demo model.

4 A confirmation prompt opens. Click Yes.



Result

The high-pressure pump is replaced. The new block is automatically connected to the model.

The ASM project initially contains no parameters for the new components.

To get the parameters:
1. Save the model
2. Open your ModelDesk experiment
3. Update your model in the ModelDesk experiment

When ModelDesk parses the model, it finds the new components in the model and creates the corresponding parameter pages with the default values. Adjust these parameter values to your engine.

Then you have the following options in ModelDesk:
- To download the parameters to your simulation platform.
- To generate initialization files.

Related topics

References

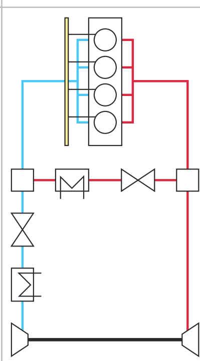
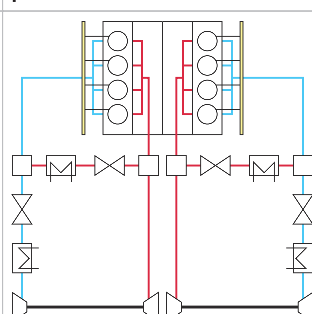
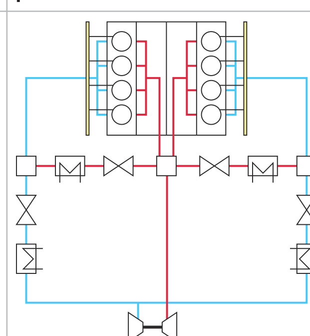
High-Pressure Pump (ASM Diesel Engine Reference 🕮)
High-Pressure Pump (Crank-Based) (ASM Diesel Engine Reference 🕮)

Working with Different Engine Configurations

Configuration overview

The ASM model lets you define engine configurations. The installation version contains three default configurations. Each configuration is stored in a parameter XML file for the Engine Setup

The following table shows three different configurations:

Configuration	Row engine	V engine with 2 air paths and 2 exhaust paths	V engine with 2 air paths and 1 exhaust path
Schematic			
Fuel systems	1	2	2

Configuration	Row engine	V engine with 2 air paths and 2 exhaust paths	V engine with 2 air paths and 1 exhaust path
Air paths	1	2	2
Exhaust paths	1	2	1
Turbochargers	1	2	1

The row engine is the default configuration and is stored in the `<Model>.xml` file.

The V-engine with two air paths and two exhaust paths is stored in `<Model>_VEngine_2In_2Exh.xml`.

The V-engine with two air paths and one exhaust path is stored in `<Model>_VEngine_2In_1Exh.xml`.

Note

If you use the operator version, it is not possible to use an advanced or two-stage turbocharger model with more than one exhaust manifold/turbocharger/exhaust system.

Measurement Interface

Where to go from here

Information in this section

Basics on Measurement Interface

Basics on Measurement Interface

Basics

The Measurement Interface allows easy evaluation of the simulation results by comparing them to the results of a steady state measurement (for example, engine speed, injection quantity, manifold pressure, mean effective torque). In case of InCylinder models also crank angle resolved data (for example, in cylinder pressure) will be evaluated. It is therefore important to provide the same conditions to the model, which have existed during the test bench run of the real engine. This is done by using some of the measurement signals (ambient pressure and temperature, engine speed, accelerator pedal position or injection signals) as stimulus signals for the engine model. Each measurement of the steady state measurement is an operating point. They are numbered from one to the total number of operating points.

The influence of (optimized) parameters can be easily investigated by modifying the parameters and observing the change in the simulation results in comparison to the measurement signals.

The following topics describe the required settings in the parameterization project and the related parts in the Simulink model.

Note

The Measurement Interface is designed to work on real-time hardware. If you want to test the model parameterization in a Simulink simulation, it is recommended to use the ASM Engine Testbench. Refer to ASM Engine Testbench (ASM User Guide 📖).

Steady-State Measurement

Where to go from here

Information in this section

Basics on Steady State Measurement

Basics

ModelDesk allows generating the steady state measurement interface subsystem according to the available measurement variables from a Measurement Type. This steady state interface subsystem is created with LUT1D records which can be parametrized by processing functions. In ASM InCylinder demo projects, in addition, crank angle resolved data can be provided with the LUT2D block.

Steady-State Measurement

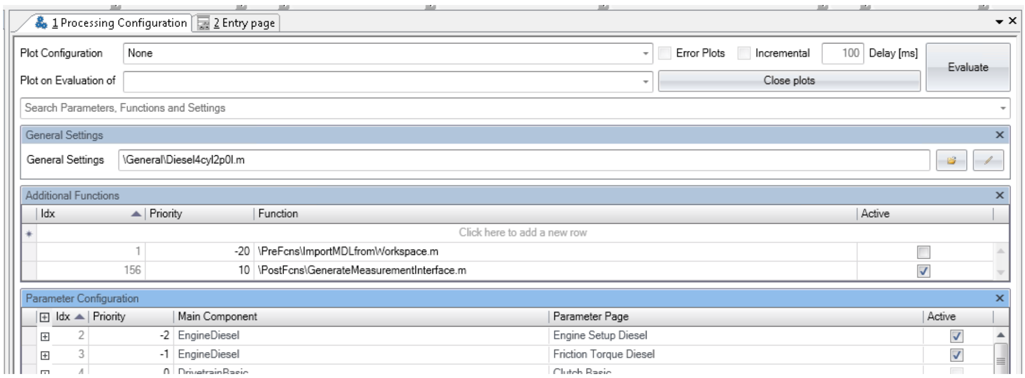
Introduction

There is a file to control the generation of the steady-state measurement interface. You must select the file in ModelDesk. You can generate a measurement interface for each measurement type.

Settings in ModelDesk

The following illustration shows the Processing Configuration pane in ModelDesk with the Additional Functions section.

Select the `GenerateMeasurementInterface.m` function in the Additional Functions section via the corresponding checkbox.



Function

The generation of the steady state measurement interface is controlled by the `GeneralSettings` file. A measurement interface can be generated for each measurement type.

```
Settings.GenerateMeasurenmenInterface(1).Active = true
Settings.GenerateMeasurenmenInterface(1).MeasurementTypeName = 'Engine';
Settings.GenerateMeasurenmenInterface(1).ModelName = 'Engine_MeasurementInterface';
Settings.GenerateMeasurenmenInterface(1).ModelPath = fileparts(mfilename('fullpath'));
Settings.GenerateMeasurenmenInterface(1).Calculated = {...% activation, result signal, operation (+ or *) ,
first signal, second signal or factor
    1,'p_In_Throttle[Pa]','+','p_Ambient[Pa]','p_In_Throttle_Rel[Pa]';
    1,'p_InMan[Pa]', '+','p_Ambient[Pa]','p_InMan_Rel[Pa]';
    1,'p_ExhMan[Pa]', '+','p_Ambient[Pa]','p_ExhMan_Rel[Pa]'
    };
```

The struct fields have the following meaning.

Active Boolean to activate the generation of the measurement interface to the specific type.

MeasurementTypeName Name of the measurement type for which the interface is generated.

ModelName Name of the generated Simulink model.

ModelPath Folder where the Simulink model is created. The folder of the general settings file is the default folder.

Calculated Cell array which defines additional variables. These can be calculated by combining existing variables or scaling them. This can be used to convert relative to absolute pressure or scale units.

Each row defines one calculated variable with the following cells:

- Activation flag
- Name of the calculated variable
- Operation how to combine signals. Either '+' or '*'
- Name of first signal
- Name of second signal or factor

A **SteadyState** subsystem is generated within this model which includes **LUT1D** blocks for all measurement variables in the measurement type. The mask parameter of the **LUT1D** subcomponent blocks are set to `MDL.MeasurementInterface.SteadyState.<MeasurementTypeName>.<VariableName>`. For details on the **LUT1D** block, refer to Look-Up Table 1-D (ASM Drivetrain Basic Reference 📖).

The instance names are also set to the variable names. The input is connected to the engine operating point number. Behind the **SteadyState** subsystem, a calculation subsystem is generated in order to build up the variables as specified. Further, all variables are routed through a subsystem with the variable name. In this way,they can easily be accessed by ControlDesk. This subsystem has to be copied to the Simulink model that is used in the project. After this, the model has to be updated in ModelDesk for the new blocks to be added to the parameter set. For details, refer to Simulink Model on page 69.

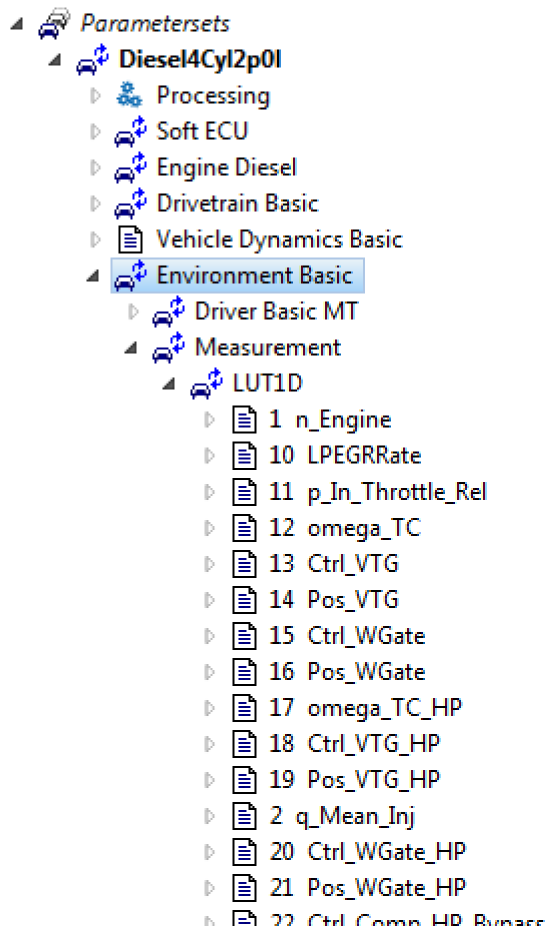
LUT1D records

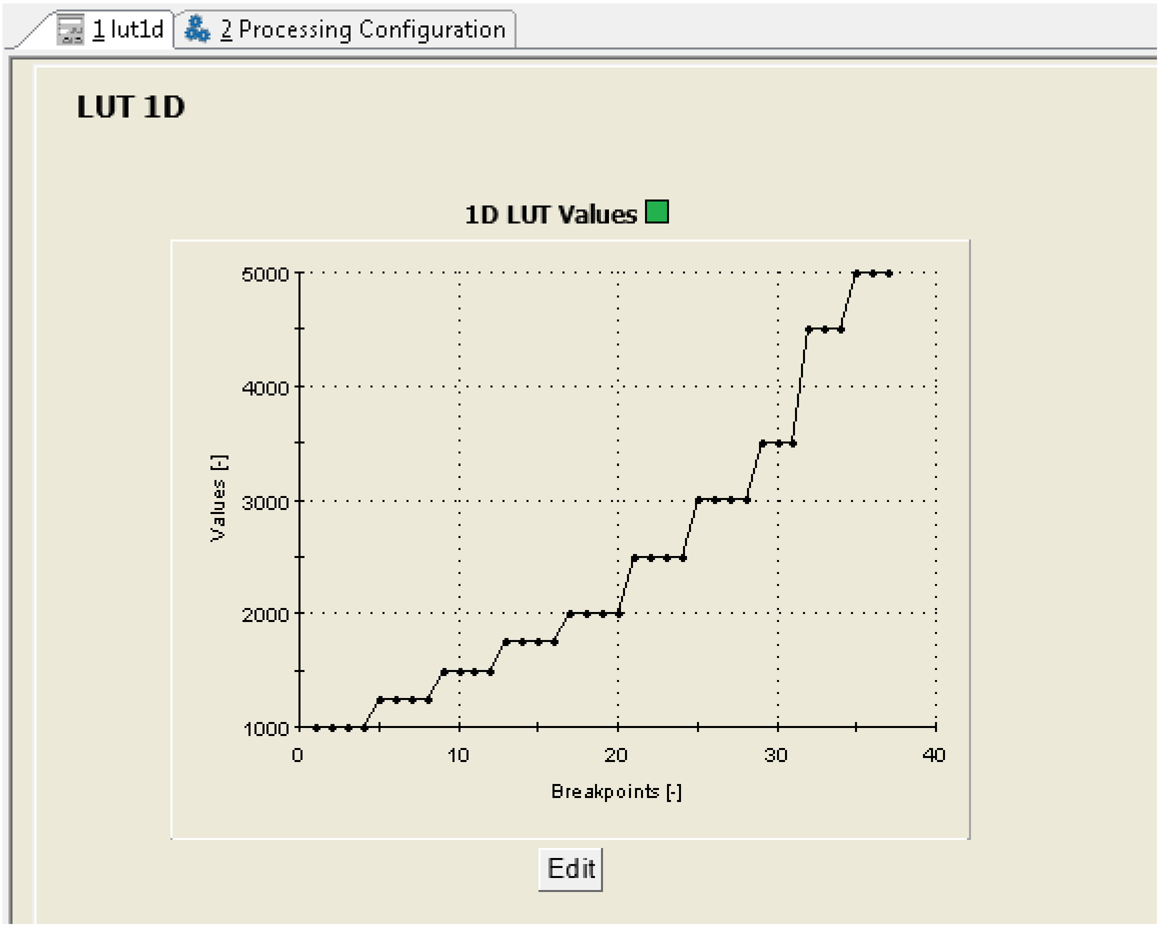
The following illustrations show the **LUT1D** records that are used in the generated measurement interface model. To show these tables it is necessary to update the Simulink model in ModelDesk by right-clicking **Simulation Model** in the project tree and select **Update Model** from the context menu.

For each measurement variable, a **LUT1D** record with a 1-D look-up table is provided. By default, the `\Environment\LUT_1D\LUT_1D_CopyMeasurement.m` processing function is connected. The measurement type and measurement variable name are determined by the place the variable has to be copied to. This is done by the mask variable.

Note

After updating the model new instances of the **LUT1D** record are linked to the `LUT1D.xml` parameter file by default. Make sure that each instance refers to an independent parameter file to ensure the correct measurement variable is stored. This can easily be done with the `CreateIndependentParameterxml.py` file which is available in the **Python Script** folder of the project.
Keep in mind that running the script affects all records, not only the **LUT1D**.


Parametersets
Diesel4Cyl2p0l
Processing
Soft ECU
Engine Diesel
Drivetrain Basic
Vehicle Dynamics Basic
Environment Basic
Driver Basic MT
Measurement
LUT1D
1 n_Engine
10 LPEGRRate
11 p_In_Throttle_Rel
12 omega_TC
13 Ctrl_VTG
14 Pos_VTG
15 Ctrl_WGate
16 Pos_WGate
17 omega_TC_HP
18 Ctrl_VTG_HP
19 Pos_VTG_HP
2 q_Mean_Inj
20 Ctrl_WGate_HP
21 Pos_WGate_HP



Initialization file When generating initialization files for EnvironmentBasic, the measurement data information is written to the MDL structure. The following example shows the created MDL structure `MDL.MeasurementInterface.SteadyState` for the engine speed:

```
...SteadyState.Engine.n_Engine.Comment = '1D LUT Values';
...SteadyState.Engine.n_Engine.Author = '(null)';
...SteadyState.Engine.n_Engine.Origin = ' ';
...SteadyState.Engine.n_Engine.Version = '3';
...SteadyState.Engine.n_Engine.LastModified = '02.02.2015';
...SteadyState.Engine.n_Engine.xName = 'Breakpoints';
...SteadyState.Engine.n_Engine.xUnit = '[-]';
...SteadyState.Engine.n_Engine.vName = 'Values';
...SteadyState.Engine.n_Engine.vUnit = '[-]';
table = [
1 1000.01
2 1000.01
…
208.31 4999.99
];
...SteadyState.Engine.n_Engine.x = table(:,1);
...SteadyState.Engine.n_Engine.v = table(:,2);
```

The <x> struct field is always the engine operating point number which is just counted from 1 to the total number of provided engine operating points.

The <v> field contains the related data.

Related topics

Basics

Additional Functions (ASM User Guide 🕮)

References

Look-Up Table 1-D (ASM Drivetrain Basic Reference 🕮)
Processing Configuration (ModelDesk Processing 🕮)

Simulink Model

Where to go from here

Information in this section

Overview of the Simulink Model

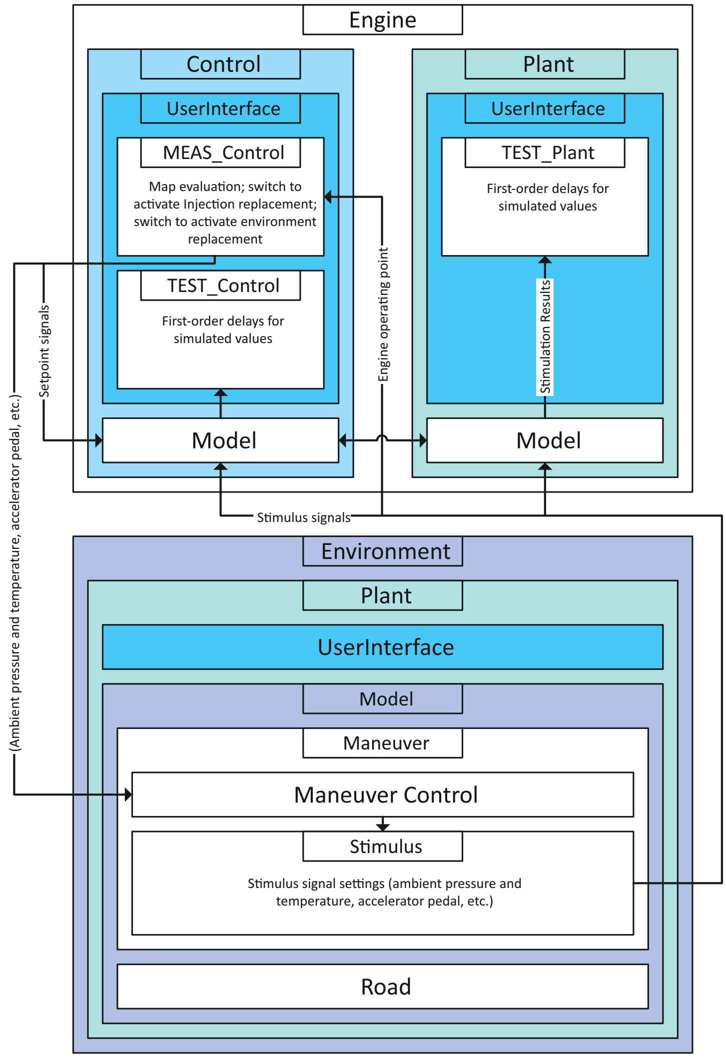
Introduction

The following illustration gives an overview of the parts of the Simulink model which are important for the measurement interface.

Model overview

The emphasis of the model overview is in the signal flow which can be done by real line connections.

The model structure itself is reduced as much as possible and a lot of parts with lower importance are skipped.



Environment Maneuver

Description

If the engine operating point which is selected in the Measurement Interface is not zero, inside the Environment/Maneuver subsystem, the Pos_AccPedal, n_Engine_Set, and Trq_Engine_Set signals are replaced by the corresponding values from the Measurement Interface.

Also, you can activate the ASM Engine Testbench by setting Sw_ModelTestCaptureEnable in Engine/PAR_Control to *one*.

If the flag to replace the environment signals (Sw_Replace_Env[0|1]) is active, the p_Ambient, T_Ambient, and T_Water signals are additionally replaced by their corresponding values from the measurement interface.

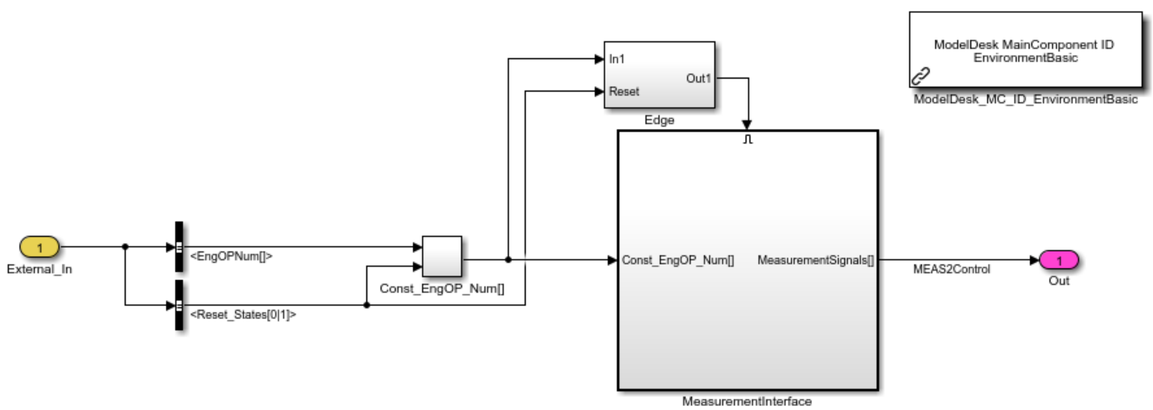
Measurement Control

Basics

You can use steady-state measurement signals in the simulation model. You can use the measurement as reference data in the control algorithm of the soft ECU of the model, for example. The signals of the steady-state measurement are located in the Engine/Control/UserInterface/MEAS_Control system.

Main task measurement

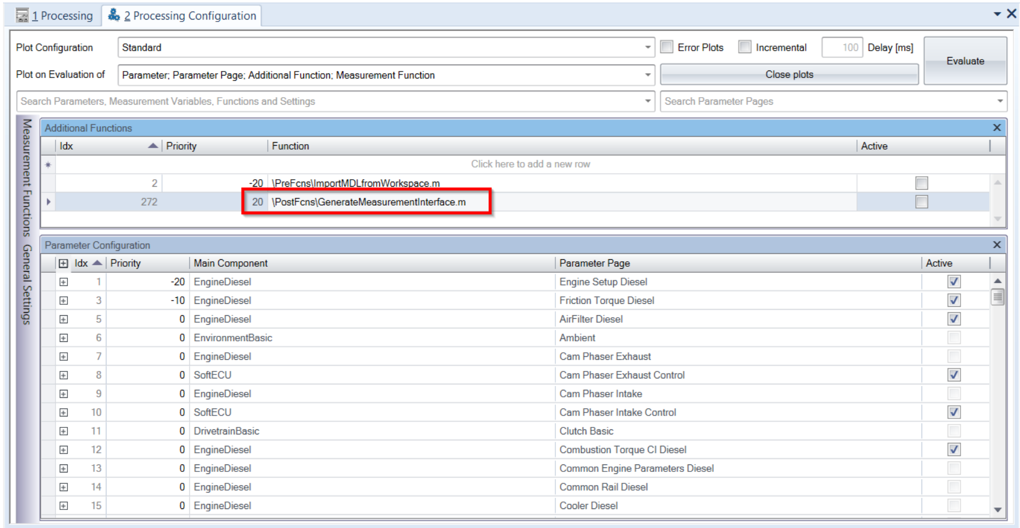
The following illustration shows the contents of the subsystem:



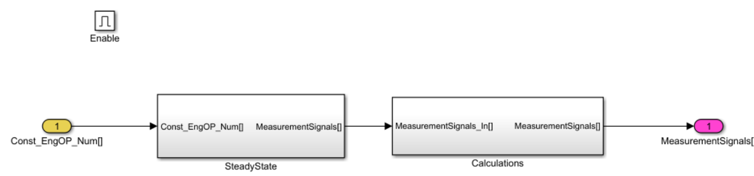
The engine operating number is selected from the Environment module. Only if the engine operating number is changed, the MeasurementInterface block is executed. The output bus contains the measurement signals of the current operating point.

To change the signals of the block, use the Additional Functions feature of ModelDesk Processing. Refer to Additional Functions (ASM User Guide 📖).

When you execute the `GenerateMeasurementInterface.m` function, a new Simulink system measurement interface is created. The MATLAB function analyses the measurement associated with the parameter set active in ModelDesk and generates a Simulink look-up table for each measurement variable. Refer to Steady-State Measurement on page 63.



The MeasurementInterface block contains two blocks:



The output of the SteadyState block is a Simulink bus containing all the signals of the measurement at the current operating point. The engine operating point number is included as EngOp_Num[] parameter.

The Calculations block has two tasks.:

- It calculates additional signals. For example, it calculates the absolute intake manifold pressure from the ambient and the relative intake manifold pressure. This signal can then be easily compared to the simulated intake manifold pressure, which is always an absolute value. This subsystem will also be generated by ModelDesk Processing.

  Two signals are used as flags.

  - Sw_Replace_Env[0|1]:

    Depending on the Sw_Replace_Env[0|1] flag, the environment pressure and temperature is replaced by the values from the measurement.

  - Sw_Replace_IOSignals[0|1]:

    Use this flag to replace the IO signals by the measurement data.
- It provides easy access to the measurement signals from ControlDesk.

Related topics

Basics

Additional Functions (ASM User Guide 📖)
Steady-State Measurement.. 63

Working with the Measurement Interface

How to Work with the Measurement Interface

Objective

You can compare the simulation result with the measurement data.

> **Note**
>
> The Measurement Interface data functionality is currently not supported by ModelDesk. Hence, if you want to use ModelDesk to generate the initialization files for the model, the Measurement Interface blocks must be removed. In this case, you can generate dummy measurement interface subsystems, which can be inserted in the model. This is done by selecting *-2* for the Sw-Selection flag of all variables in the measurement interface settings file. Refer to Steady-State Measurement on page 63.

Method

To work with the measurement interface

1 To replace the environment signals, set the value of the Sw_Replace_Env[0|1] in `Engine/Control/UserInterface/MEAS_Control/MeasurementInterface/Calculations` to `1`.

> **Note**
>
> A change of this parameter only has an effect on the simulation results, when the engine operating point is once changed afterwards.

2 Set the desired engine operating point number.

This evaluates all look-up tables with measurement data once. Furthermore, it is switched to the test bench mode using the accelerator pedal position and the set engine speed from the measurement.

3 Wait until the simulation results reach their steady state. Depending on the observed signals this can be after a few seconds (for example, pressures) or take several minutes (for example, temperatures of exhaust system).

4 Compare the simulation results.

5 You can now change to another operating point or try to change parameters to improve simulation results.

6 To quit the measurement comparison, set the engine operating point to *zero*.

This stops the replacement of environment and sets the test bench to the previous state. If external setpoints for the SoftECU controller have been activated, they must be set back to Map or Calculated setpoint.

Result	You used the Measurement Interface to compare simulation results.
Related topics	Basics

Troubleshooting

Troubleshooting with Rising Turnaround Times at Steady Operating Point

Description

Simulating the model on a real-time processor board at a steady state, might lead to a significant increase in turnaround time after several seconds of simulating this operating point. When changing to another operating point, the turnaround time decreases to the original lower value. For example, in case of an engine model this behavior might occur when you drag the engine to test bench mode at a fixed operating point.

For information on possible reasons and workarounds, refer to Troubleshooting with Rising Turnaround Times at Steady Operating Points (ASM User Guide 🕮).