

GT-SUITE

Controls Coupling and Real Time Manual

VERSION 2016



by

Gamma Technologies

Copyright 2015 © Gamma Technologies LLC. All rights reserved.
All information contained in this manual is confidential and cannot be reproduced or
transmitted in any form or by any means, electronic or mechanical, for any purpose, without
the express written permission of Gamma Technologies LLC.

GT SUPPORT

- TELEPHONE: (630) 325-5848
- FAX: (630) 325-5849
- E-MAIL: support@gtisoft.com
- Web Address: www.gtisoft.com
- Address: 601 Oakmont Lane, Suite 220
Westmont, IL60559
USA

Telephone Support Hours

8:00 A.M. to 5:30 P.M. Central Time Monday - Friday

TABLE OF CONTENTS

CHAPTER 1: INTEGRATED MODELING WITH SIMULINK	1
1.1 Running Coupled Simulations from GT-SUITE	1
1.1.1 Prerequisites.....	2
1.1.2 Building the desired Simulink model for the GT-SUITE Target.....	2
1.1.3 Tuning Simulink Parameters from GT-SUITE.....	4
1.2 Running Coupled Simulations from Simulink.....	11
1.2.1 Prerequisites.....	11
1.2.2 Preparing the Coupled Simulation.....	11
1.2.3 Running GT-SUITE and Simulink on Different Workstations/Operating Systems	17
1.2.4 Resolving performance issues	18
1.3 Building s-functions using GT-SUITE for use in Simulink	20
1.4 Using GT-SUITE Neural Networks in a Simulink Model.....	26
1.4.1 Prerequisites.....	26
1.4.2 Instructions	26
CHAPTER 2: GT-SUITE-RT - Real-Time Modeling	27
2.1 Licensing.....	27
2.2 HiLPlatforms	27
2.3 Download and Installation (for both PC and target machine):	28
2.4 GT-SUITE on Real-Time HiL (Hardware in Loop) Systems.....	29
2.5 Compiling Simulink Models with GT-SUITE-RT S-function as Standalone Executables	33
Prerequisites	33
Procedure.....	33
2.6 Common Errors.....	36
CHAPTER 3: Functional Mock-Up Interface (FMI) with GT-SUITE.....	38
3.1 Exporting an FMU (GT-SUITE as Slave)	38
3.1.1 Creating an 'FMUExport' Object	38
3.1.2 Exporting FMU.....	41
3.1.3 Checking Model Integrity as a Standalone Model.....	41
3.1.1 Troubleshooting GT-Made FMUs	42
3.2 Importing an FMU (GT-SUITE as Master)	42
3.2.1 Creating a 'FMUImport' Object	42
3.2.2 Updating Imported FMU	45
CHAPTER 4: Integrated Modeling with ASCET	47
CHAPTER 5: General Co-Simulation with GT-Master API.....	50
5.1 General Co-Simulation Principle.....	50
5.2 GT-Master API Functions.....	51
5.3 Using GT-Master API with General Model Written in C.....	52
5.3.1 Prerequisites.....	52
5.3.2 General Model Co-Simulation Setup Instructions	52

CHAPTER 1: INTEGRATED MODELING WITH SIMULINK

GT-SUITE applications may be run in conjunction with other modeling programs to simulate systems that are more complex than either program can simulate independently. One such modeling program that can be coupled with GT-SUITE is Simulink from The Mathworks. This is a common practice in the industry where engine, powertrain, and vehicle physics are modeled in GT-SUITE, and electronic controllers are modeled in Simulink. Any GT-SUITE control signal can be passed to or from Simulink via a 'SimulinkHarness' component. For example, vehicle speed can be sensed in GT-SUITE and then passed to Simulink, which responds by controlling the throttle angle in the GT-SUITE model. There are three methods in which this can be accomplished:

- 1) Compiling Simulink model(s) into .dll/.so files and importing them into a GT-SUITE model.
- 2) Running a GT-SUITE model from the Simulink interface using the GT s-function available in the Simulink block library
- 3) Creating a model specific s-functions using GT-SUITE for use in a Simulink

In only the first (2) cases, the MATLAB path must point to the directory that contains the GT-SUITE S-Function library. This depends on the operating system:

- PC: %GTIHOME%\v%GTI_VERS%\simulink.
- LINUX: \$GTIHOME/v\$GTI_VERS/GTsuite/bin/linux_x86

Where, the variable %GTIHOME% (for Linux \$GTIHOME) represents the GT-SUITE installation directory, and %GTI_VERS% (for Linux \$GTI_VERS) represents the currently installed version, v2016.

The MATLAB path can be modified once and saved for future MATLAB sessions. From the MATLAB environment, press “Set Path” and add the folder mentioned above. Then press “Save” (for more on this, please refer to the MATLAB documentation). Alternatively, a MATLAB script file named "addgtipath2016.m" is included with the GT-SUITE installation to automate this procedure but this will not save the path and you will need to set it at every time that you open MATLAB. This file can be found in the directory %GTIHOME%\v%GTI_VERS%\simulink.

Method 3 above does not require the Matlab path to include the aforementioned GT directory as all of the necessary files are packaged when the s-function is created.

1.1 Running Coupled Simulations from GT-SUITE

This feature allows users to create their Simulink models using the “GT-SUITE Model (Master)” S-function block, and use Simulink Coder (formerly Real-Time Workshop) to generate C-code from the model and compile it as a *.dll (or an *.so for Linux). This dll/so can then be pointed to in the ‘SimulinkHarness’, or 'CoSimInterface', in GT-SUITE. An infinite number of Simulink models can be loaded dynamically by a single GT-SUITE model with this method.

Using this method has several advantages over running the model from Simulink. As with non-coupled simulations, this allows the flexibility of using multiple cases. The GT-ISE DOE and direct optimizer tools can be utilized to analyze systems, which is not possible when running from Simulink. Additionally, Simulink models can now be simulated without having a Matlab or Simulink license – a GT-SUITE user needs only the .dll/.so file to run their coupled simulation through GT-SUITE.



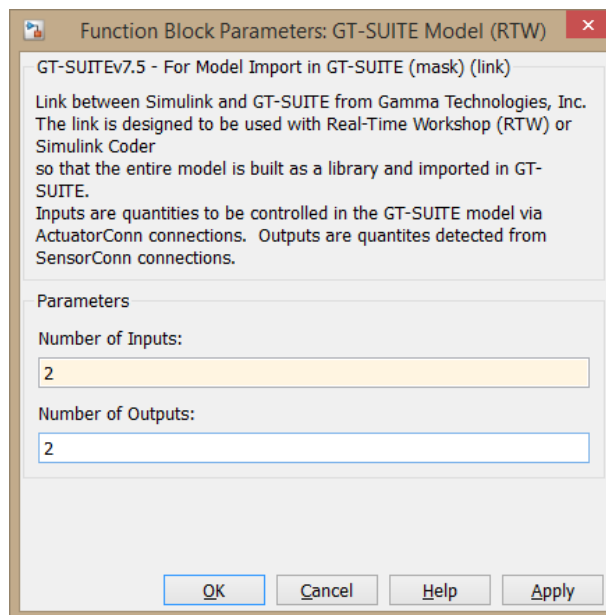
Please note that only one GT-SUITE Model (Master) block is allowed per Simulink Model.

1.1.1 Prerequisites

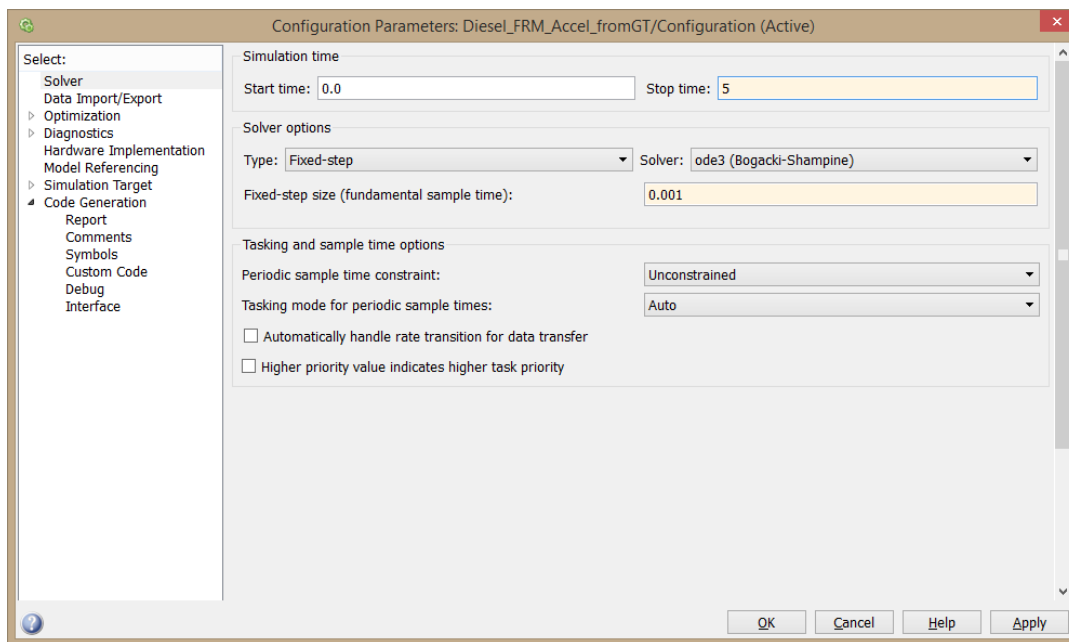
- GT-SUITE v7.1.0 or later
- Matlab R2006b or later with Simulink and Simulink Coder (formerly Real Time Workshop)
- Windows or Linux (32bit and 64bit supported for Windows, only 64bit supported for Linux)
- If using Windows: Microsoft Visual Studio 2005 or later, Microsoft SDK v7.1 or later
- If using Linux: Built-in gcc compiler is sufficient. Note that with Linux 64 bit, the gcc 32 bit libraries must be installed

1.1.2 Building the desired Simulink model for the GT-SUITE Target

1. Add the “GT-SUITE Model (Master)” S-function to the Simulink model. This block can be found in the Simulink Library Browser, or from the library file `gtlink2016.mdl`, located in: `%GTIHOME%\v%GTI_VERS%\simulink` for PC or `$GTIHOME\v$GTI_VERS\simulink` for Linux/Unix.
 - The “GT-SUITE Model (Master)” S-Function blocks possess one input port and one output port. Multiple input signals should be Muxed immediately upstream of the input port, and multiple output signals should be Demuxed immediately downstream of the output port. Note that the order of connection is important. Input #N of the Simulink Mux block connects to the output Signal #N of the GT-SUITE 'SimulinkHarness'. Similarly, Output #N of the Simulink Demux block connects to the input Signal #N of the GT-SUITE 'SimulinkHarness'.
 - Signals connected to the input port of the S-Function block are transmitted to the GT-SUITE simulation. Signals connected to the output port of the S-Function block are received from the GT-SUITE simulation.
2. Specify the parameters for the GT-SUITE Model (Master) S-Function Block



- **Number of Inputs/Outputs:** Specify the number of inputs and outputs to the S-function block. Note that values of “-1” are not appropriate with the "GT Master" block.
3. From the menu, choose Simulation -> Configuration Parameters -> Solver. Choose Fixed-step solver and specify the fixed-step size (this defines the communication interval between GT-SUITE and the Simulink Model). The variable-step solver is not available with this approach.




4. In the same window, select the Code Generation page (formerly Real Time Workshop). At the System target file entry, click Browse. Choose the gtsim_***.tlc file that corresponds to your platform:

Matlab 64bit/GT-SUITE 64bit on Windows/Linux:	gtsim2016_w64.tlc/gtsim2016_a64.tlc
Matlab 64bit/GT-SUITE 32bit on Windows:	gtsim2016_w64.tlc*
Matlab 32bit/GT-SUITE 32bit on Windows:	gtsim2016_w32.tlc

Press OK.

***Note:** The combination of 64bit Matlab and 32bit GT-SUITE on Windows machines requires the user to alter the name of a file in order to successfully create a *.dll. Navigate to %GTIHOME%\v%GTI_VERSION%\simulink. Change the name of 'use_to_create_w32_dll_64bit_matlab.m' to 'gtsim2016_w64_wrap_make_cmd_hook.m'. Proceed with these instructions to create your *.dll. Should the user no longer be working with the 64bit Matlab/32bit GT-SUITE combination, change the name of 'gtsim2016_w64_wrap_make_cmd_hook.m' back to 'use_to_create_w32_dll_64bit_matlab.m'.

5. The Template Makefile should automatically change to the corresponding gtsim_***.tmf file.
6. Click OK to return to the Simulink model block diagram.
7. Press the Build button (, or press Ctrl+B.



8. In the MATLAB command window, the code generation and building process should complete successfully. The <modelname>.dll/.so should now be in the working folder ready to be used from GT-SUITE.
9. Build the GT-SUITE Model as needed.
10. Link the 'SimulinkHarness' part to other parts in the GT-SUITE model. The 'SimulinkHarness' input and output signal numbers correspond to the output and input signal numbers (respectively) of the GT-SUITE (Master) block in the Simulink model (please see 'SimulinkHarness' in the reference manual, or in the online help, for more information). There are three ways that the links may be made:
 - "Wireless" links via the attributes in the 'SimulinkHarness'.
 - Wireless via named signals, using 'SendSignal' and 'ReceiveSignal' parts.
 - Direct links. When connecting to non-control parts, the 'SensorConn' and 'ActuatorConn' part are used. When linking to other control parts, the link can be directly made without any connection.
11. Set the **Simulation Type** attribute to "import_simulink_model" in the **Main** folder of the 'SimulinkHarness' part(s).
12. Specify the compiled .dll/.so for the **Simulink Model to Import (.dll/.so)** attribute. Multiple 'SimulinkHarness' parts may use the same or different .dll/.so libraries.
13. Run the GT-SUITE model in the same fashion as a non-coupled simulation.

GT-SUITE allows the user to view messages generated by the imported Simulink modules during the simulation. Specifically, if the solver graphical user interface is enabled, those messages are written to the <modelname>.con file located in the working directory of the GT-SUITE model. If the solver graphical user interface is disabled, those messages appear on the command prompt output window.

1.1.3 Tuning Simulink Parameters from GT-SUITE

With a few modifications to the Simulink model and the 'SimulinkHarness', the user can tune various parameters in a compiled *.dll from the GT-SUITE interface. At the present time, tuning parameters within a referenced Simulink model is not supported.

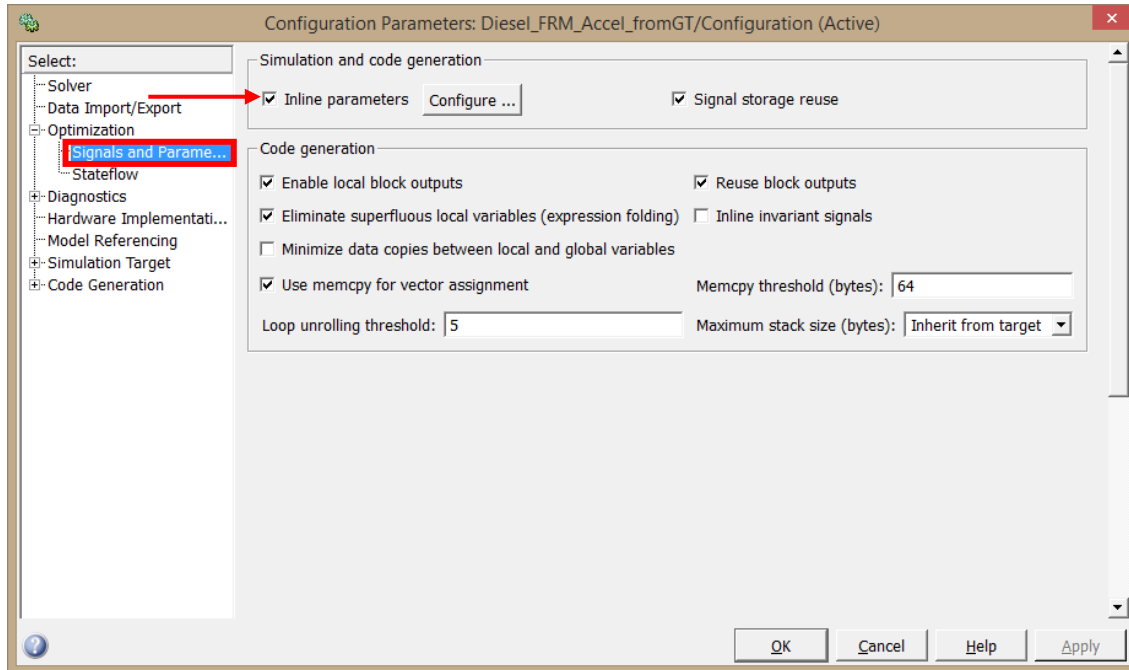
Note: Depending on your particular version of Matlab/Simulink, the layout of the 'Configuration Parameters' window pane may appear differently than what is shown below.

In the Simulink Model:

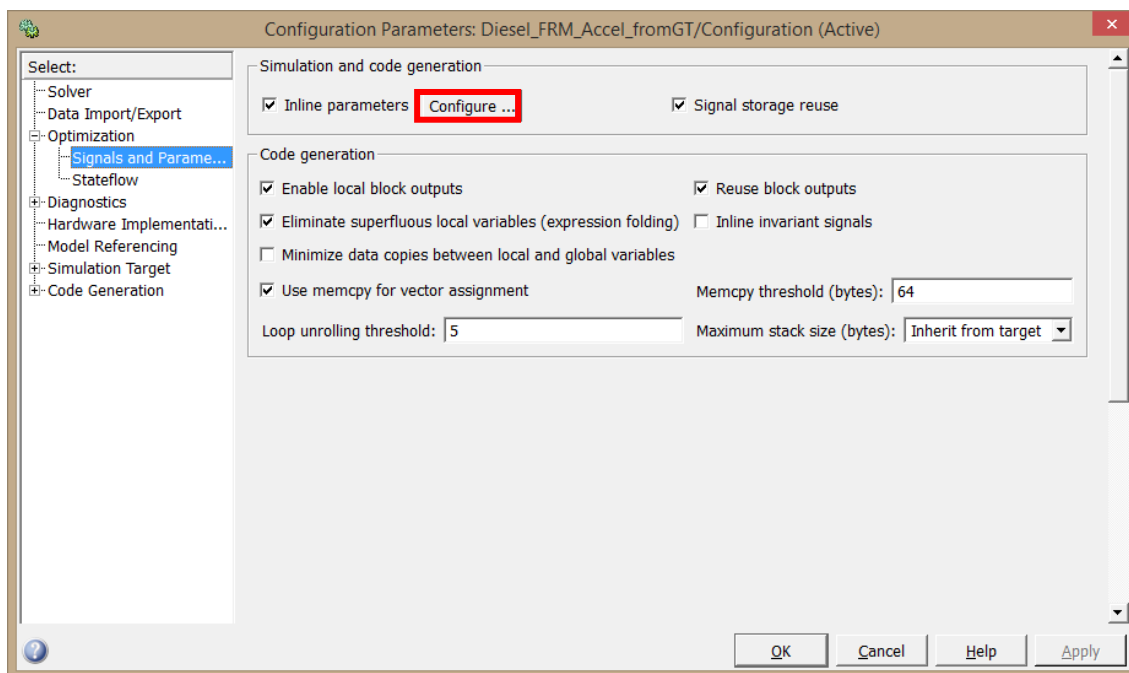
1. Open the 'Configuration Parameters' dialog box and select the Optimization category. Check the 'Inline parameters' option within the 'Simulation and code generation' pane.

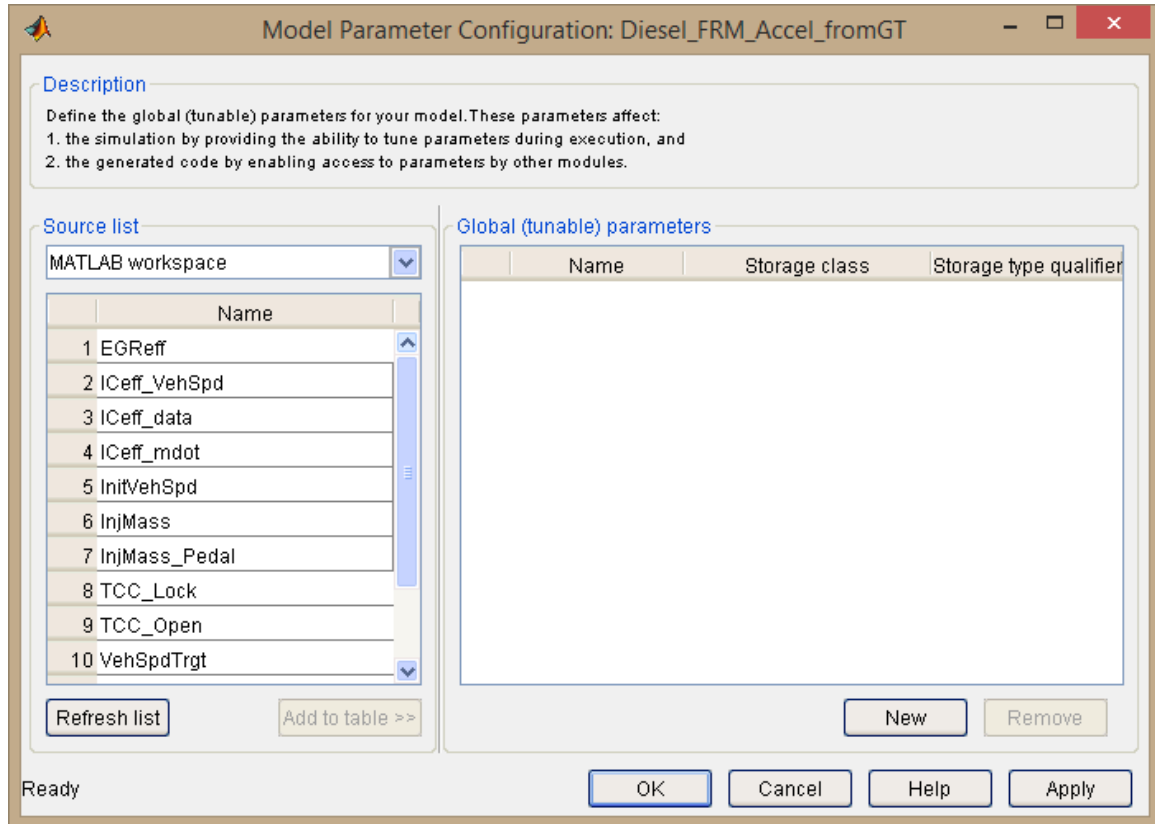
Note: For Matlab 2012a and beyond, you will find the 'Inline Parameters' option in the 'Simulation and code generation' pane of the 'Signals and Parameters' section under the 'Optimization' category.





2. With the 'Inline parameters' options checked, the 'Configure...' button will become active. Click this button to bring up the 'Model Parameter Configuration' dialog.

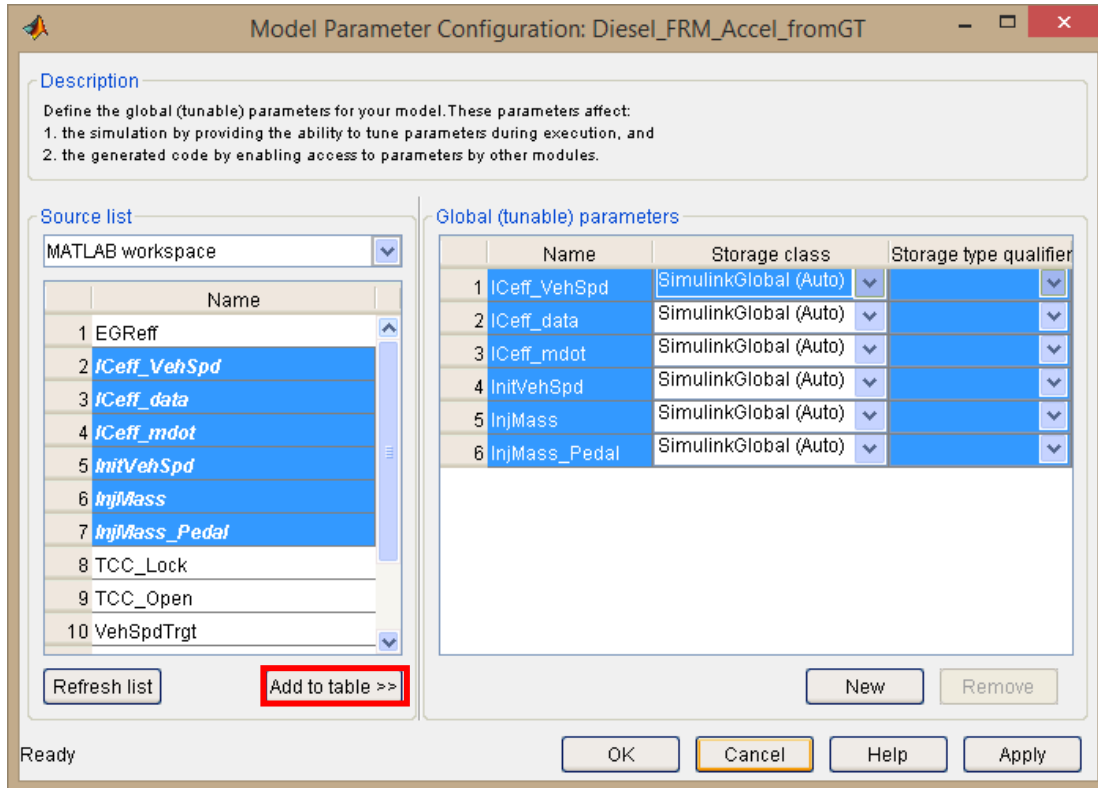




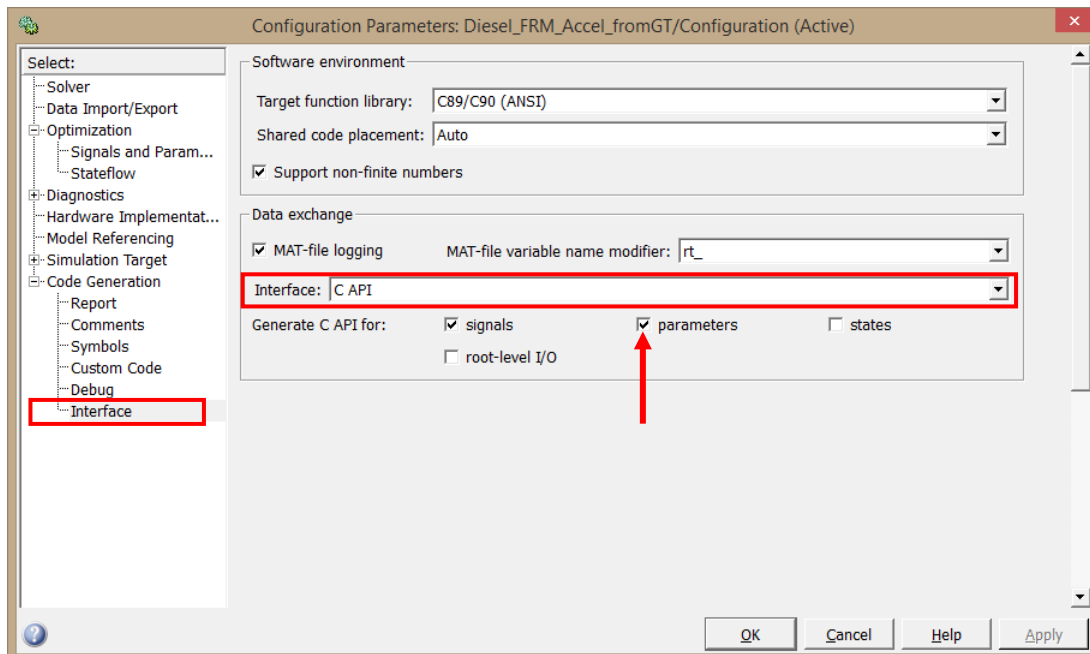
3. In the left pane of the Model Parameter Configuration dialog you will see a list of all of the parameters that are available to be set as Global (tunable). GT-SUITE supports the following formats:
 - Single numeric parameters
 - XYMap (1-D Lookup Table)
 - XYZMap (2-D Lookup Table)
 - Arrays

Select the values that are to be tunable and click 'Add to table >>'. Make sure the 'Storage Class' is set to 'SimulinkGlobal (Auto)'. Click OK when finished.





4. Select the 'Interface' category in the 'Configuration Parameters' dialog box. In the 'Data exchange' pane, change the value of the 'Interface to C API' and verify the 'parameters' option is checked. Click OK when finished.

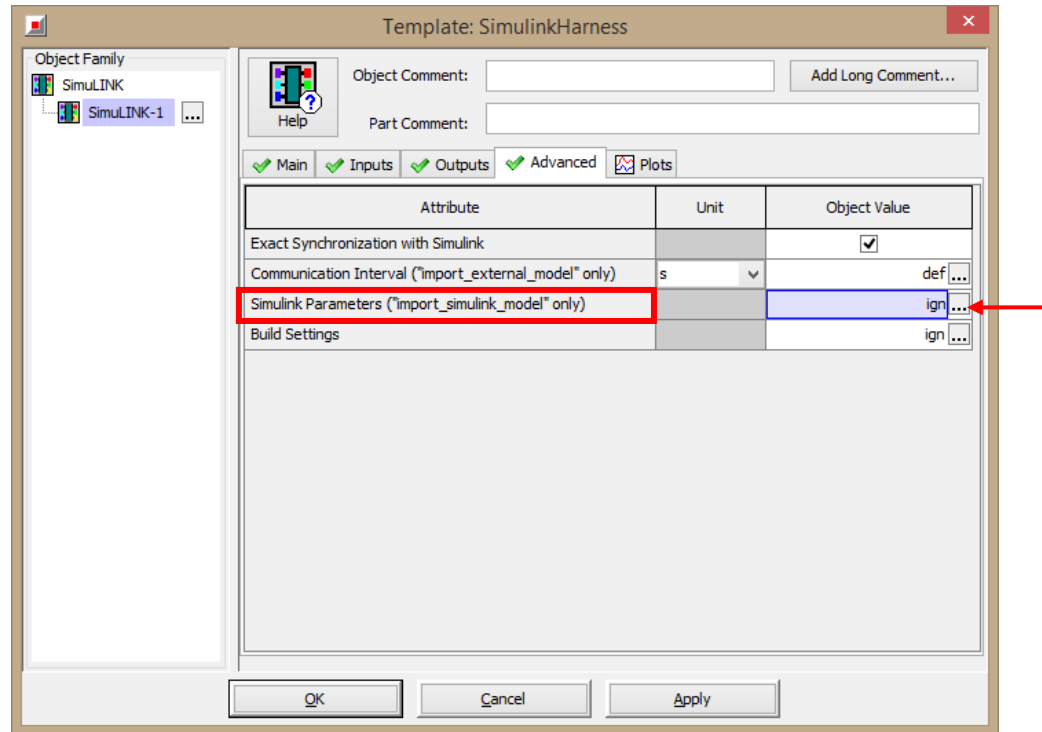


5. Construct the desired Simulink model and build per usual.



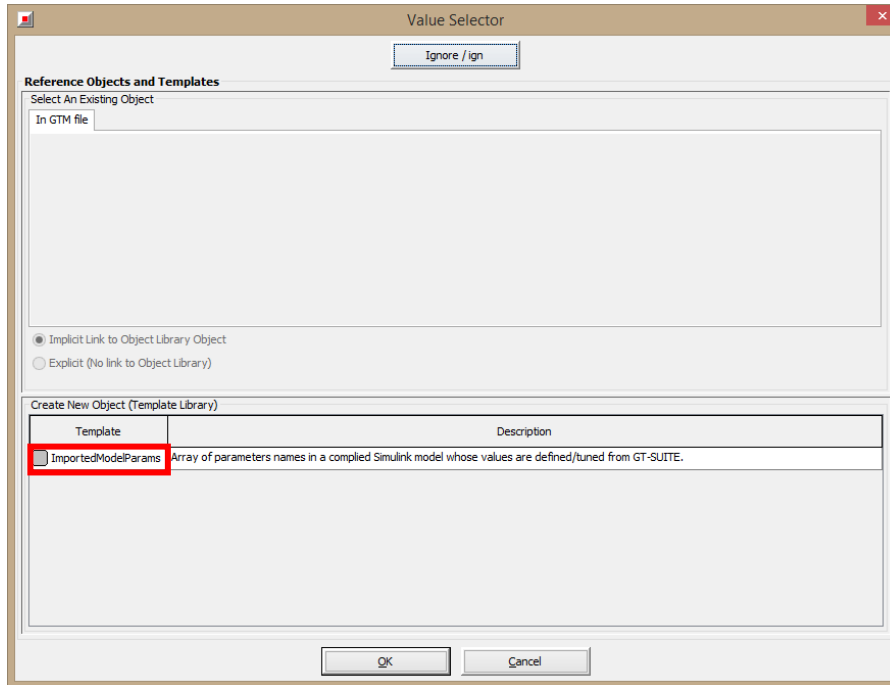
In the GT-SUITE model:

1. In the 'SimulinkHarness', click on the 'Advanced' folder. Using the 'Simulink Parameters' attribute, a reference object can be created that defines the corresponding parameters declared as 'Global' in the Simulink model, as well as, the values for them. Click the Value selector in the 'Object Value' field for this attribute.

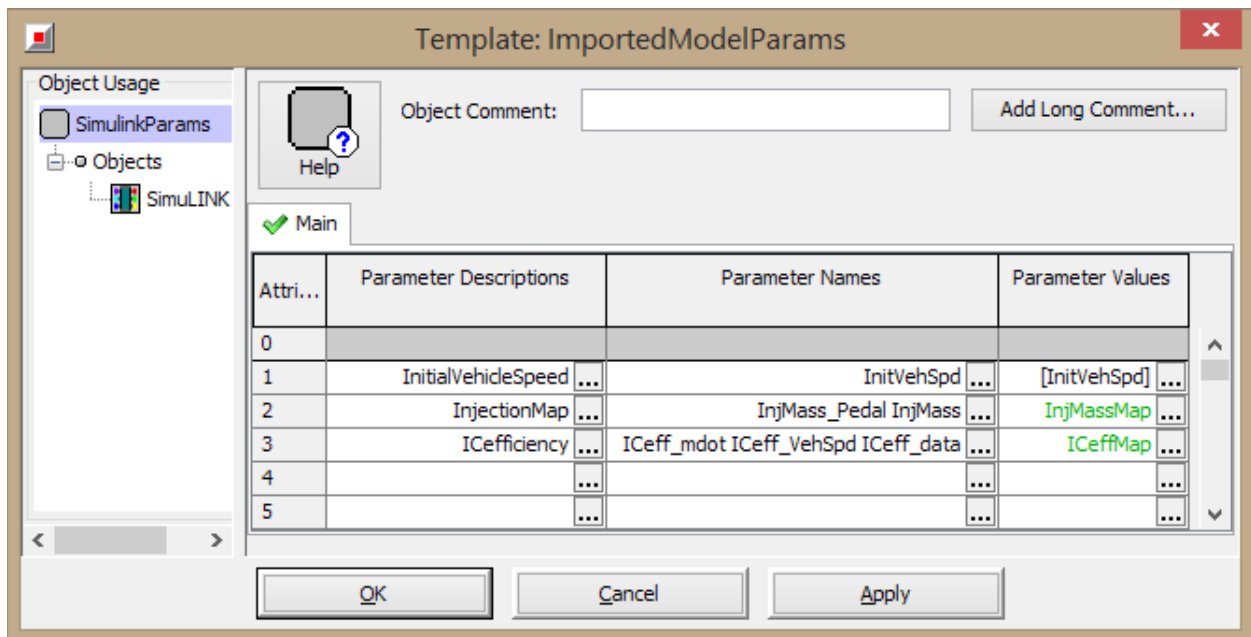


2. In the Value Selector dialog box, either select a previously created 'ImportedModelParams' reference object, or double-click the template link to create a new one. For more information regarding this reference object, consult the help documentation for this template.





3. Create a reference object reflective of the parameters you want to tune.

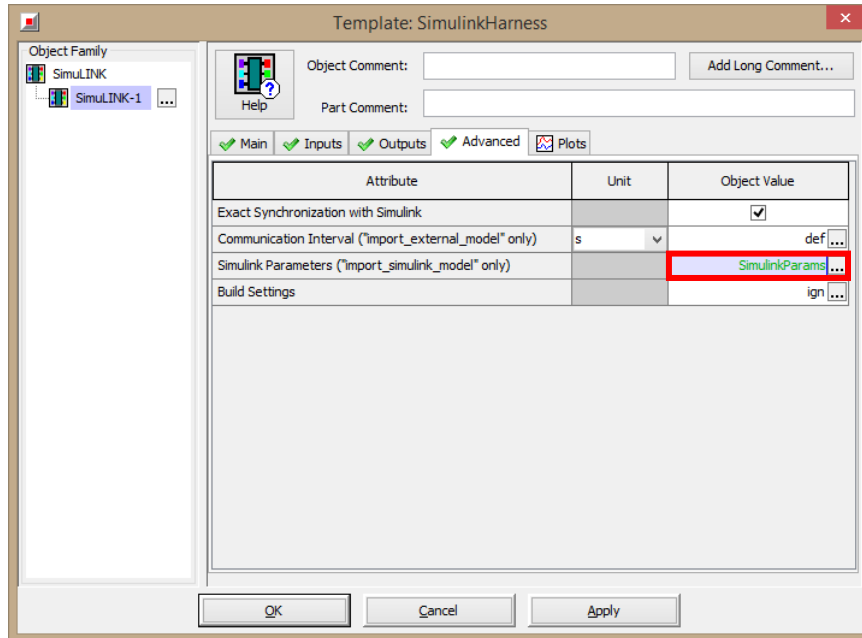


Enter the names of the Simulink Parameters in the 'Parameter Names' column. These must appear exactly as they appear when declared in Simulink. The 'Parameter Values' field contains the value, or table, in GT-SUITE that will be assigned to the corresponding parameter in Simulink. This can be a number, a parameter selected from Case Setup, or even a reference object such as an XYTable, or XYZMap.



Note: It is imperative that the size of the maps in GT-SUITE match exactly the size of their counterparts in Simulink.

Click OK and you will now see the reference object defined in the Simulink Parameters field.



Click OK.

4. Notice within the reference object that the parameter values are explicitly defined in the part, and there are parameters promoted to the Case Setup. Define the values in Case Setup and the model is ready to run.



1.2 Running Coupled Simulations from Simulink

GT-SUITE models can be executed from the Simulink environment. This is a common practice in the industry where engine, powertrain, and vehicle physics are modeled in GT-SUITE, and electronic controllers are modeled in Simulink. Any GT-SUITE control signal can be passed to or from Simulink via a 'SimulinkHarness' component. For example, vehicle speed can be sensed in GT-SUITE and then passed to Simulink, which responds by controlling the throttle angle in the GT-SUITE model. There are two methods in which this can be accomplished:

- 1) Co-simulation using the GT-SUITE s-function block
- 2) Using GT-SUITE to generate a model specific s-function and Matlab executable

Each method carries the restriction that only one 'SimulinkHarness' part may be used in the GT-SUITE model. The first method benefits those users with an installation of GT-SUITE available for the co-simulation. Those users that wish to share the GT-SUITE model as a pure "black box" with a Simulink end user should employ the second method. This approach requires the end user only having access to a GT license and does not require a full installation of GT-SUITE.

1.2.1 Prerequisites

- 1) Simulink coupling is supported from Matlab 2006b to Matlab 2015b.
- 2) For 64bit systems:
 - a) PC: `gtsuitesl2016_dp.mexw64` must be present in the directory `%GTIHOME%\v%GTI_VERS%\simulink`
 - b) LINUX users must have the proper files, and must make the necessary modifications to environment variables. Note: Matlab/Simulink may not support all platforms supported by GT-SUITE, and GT-SUITE may not support all platforms supported by Matlab/Simulink.

 Additionally, `gtsuitesl2016_dp.mexa64` must be present in the directory `$GTIHOME/v$GTI_VERS/GTsuite/bin/linux_x86`. The environment variable `LD_LIBRARY_PATH` must be modified to include `$GTIHOME/v$GTI_VERS/GTsuite/bin/linux_x86_64`
- 3) For 32bit systems:
 - a) PC users, `gtsuitesl2016_dp.mexw32` must be included in the directory `%GTIHOME%\v%GTI_VERS%\simulink`.
 - b) LINUX x86: the S-Function library file `gtsuitesl2016_dp.mexglx` should be present in the directory `$GTIHOME/v$GTI_VERS/GTsuite/bin/linux_x86`. The environment variable `LD_LIBRARY_PATH` must be modified to include `$GTIHOME/v$GTI_VERS/GTsuite/bin/linux_x86`

1.2.2 Preparing the Coupled Simulation

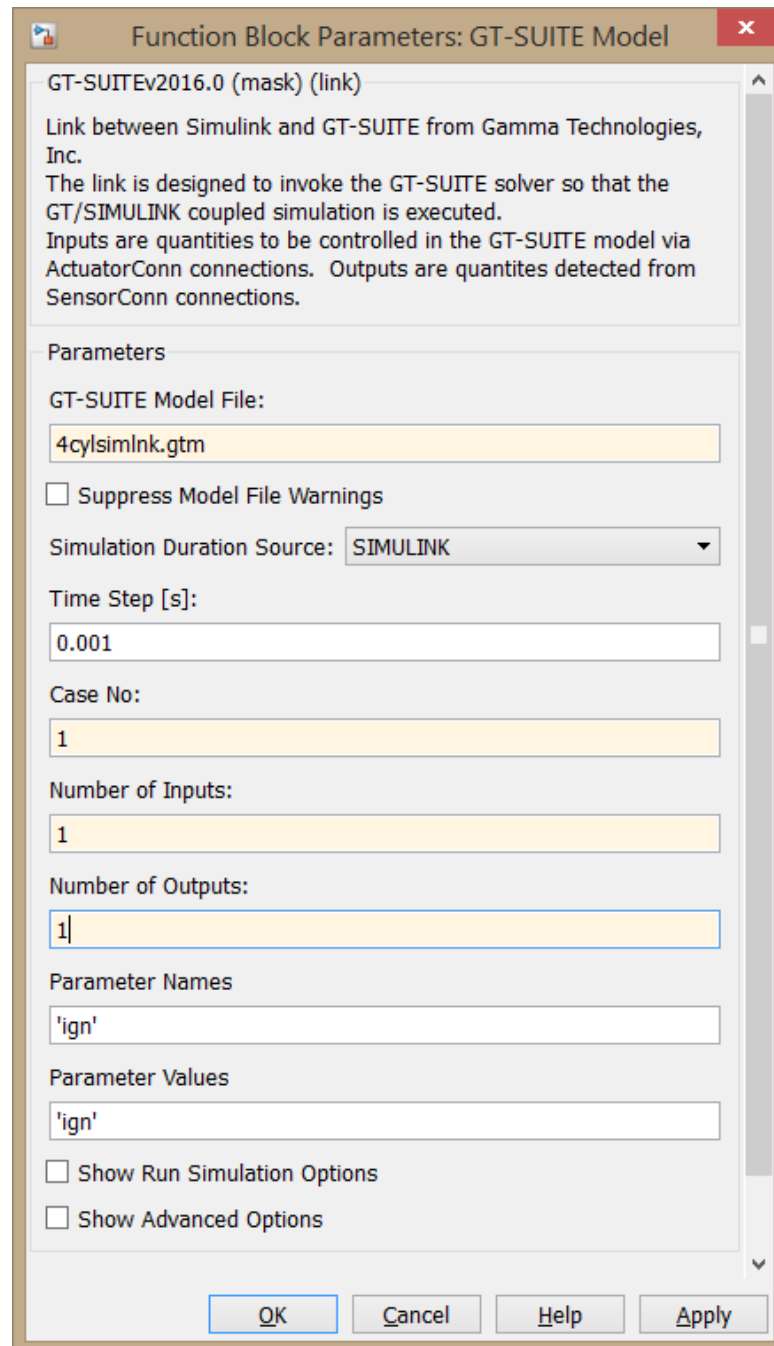
1. Build the complete system model in GT-SUITE.
2. Include a 'SimulinkHarness' part in the model.
3. Link the 'SimulinkHarness' part to other parts in the GT-SUITE model. The 'SimulinkHarness' input and output signal numbers correspond to the output and input signal numbers (respectively) of the GT-SUITE S-Function Block in the Simulink model (please see 'SimulinkHarness' in the reference



manual, or in the online help, for more information). There are three ways that the links may be made:

- "Wireless" links via the attributes in the 'SimulinkHarness'.
 - Wireless via named signals, using 'SendSignal' and 'ReceiveSignal' parts.
 - Direct links. When connecting to non-control parts, the 'SensorConn' and 'ActuatorConn' part are used. When linking to other control parts, the link can be directly made without any connection.
4. In the 'SimulinkHarness' part, ensure that the **Simulation Type** attribute in the **Main** folder to "run_from_simulink" and the **Simulink Model to Import (.dll/.so)** is set to "ign".
 5. Ensure the Direct Optimizer found under the 'Optimization' icon is set to "off". Running a coupled simulation from Simulink does not support the use of the direct optimizer.
 6. It is no longer required to generate a *.dat file from GT-SUITE, but pressing play to verify the GT model runs for a few moments without any errors is a recommended practice. Please note that if your model has an encrypted external subassembly, the *.sim file would have to be copied into the directory containing the Simulink model as well.
 7. Build the desired Simulink model, adding the "GT-SUITE Model" S-Function block. This calls the standard double precision block. Single precision is not supported. This block can be found in the Simulink Library Browser, or from the library file gtlink2016.mdl, located in: %GTIHOME%\v%GTI_VERS%\simulink for PC or \$GTIHOME\v\$GTI_VERS\simulink for Linux/Unix.
 8. Double-click on the "GT-SUITE Model" S-Function block to specify the proper values in the Block Parameters dialog box:





Function Block Parameters: GT-SUITE Model

GT-SUITEv2016.0 (mask) (link)

Link between Simulink and GT-SUITE from Gamma Technologies, Inc.
The link is designed to invoke the GT-SUITE solver so that the GT/SIMULINK coupled simulation is executed.
Inputs are quantities to be controlled in the GT-SUITE model via ActuatorConn connections. Outputs are quantities detected from SensorConn connections.

Parameters

GT-SUITE Model File:
4cylsimlnk.gtm

☐ Suppress Model File Warnings

Simulation Duration Source: SIMULINK

Time Step [s]:
0.001

Case No:
1

Number of Inputs:
1

Number of Outputs:
1

Parameter Names
'ign'

Parameter Values
'ign'

☐ Show Run Simulation Options
☐ Show Advanced Options

OK Cancel Help Apply

- **GT-SUITE Model file:** Enter the name of the GT-SUITE model file. This entry must not be enclosed in quotes. There are two options to reference the model here. First, the name of entire GT-SUITE model can be entered with the '.gtm' extension included. In this case, the *.dat file will be created at run time and changes can be made in the GT-SUITE model without having to manually re-generate the *.dat file. Secondly, the name of the GT model can be entered here without adding the any extension in order to use the *.dat file. This case requires the *.dat file be re-generated should any changes be made in the GT-SUITE model. Otherwise, the changes will not be reflected in the simulation.



- **Suppress Model File Warnings:** When this is left unselected, a user utilizing a *.dat file will be warned via a dialog box, as well as, a message in the command line that it is recommended a GT-SUITE (*.gtm) model be referenced instead. By selecting this option, this warning will no longer appear.
- **Simulation Duration Source:** Choose whether the simulation duration will be controlled by Simulink or GT-SUITE. Please refer to Item 9 below for more details.
- **Time step [s]:** This value dictates the communication interval between Simulink and the GT-SUITE model. For Simulink models with a "Fixed-step", a value of "-1" can be used in this attribute so the interval will be automatically set to the Simulink master timestep.
- **Case No.:** If a positive integer is entered, this value dictates the case number that will be run in the GT-SUITE model (even if this particular case was not checked in GT-SUITE). Alternatively, the user may enter a value of "-1". With this approach (default), the *first* CHECKED case in GT-SUITE is selected.
- **Number of inputs/outputs:** Specify the number of inputs and outputs to the S-Function block. The user may enter the exact number of inputs and outputs, in which case Simulink checks to ensure that the actual number of inputs and outputs match these values. Alternatively, the user may enter a value of "-1" for Number of inputs, Number of outputs, or both. With this approach, Simulink automatically accepts all of the input and/or output signals connected to the S-Function block. Note that, internally, consistency between the number of I/O signals of the S-function block and those of the 'SimulinkHarness' is being checked and the appropriate warning messages are output in the MATLAB command window (note: the use of the 'SendSignal' feature in conjunction with "-1" reduces the accuracy of this consistency check)
- **Parameter Names/Values:** If there are parameters in the GT-SUITE model file, the values of these can be modified from the S-Function block. The names of the parameters must be entered in the "Parameter Names" attribute as a string containing the space-delimited names of the parameters (corresponding to the names in Case Setup of GT-SUITE). The values corresponding to the parameters specified under "Parameter Names" must also be entered as a string containing space-delimited values. The default value for these fields is "ign", and none of the GT-Suite parameters will be modified. Parameter values must be passed in the same unit that appears in the GT model's Case Setup.

It is possible to pass anything that would be available in case setup through the S-function (i.e. numbers, reference objects, file names etc.). While passing a reference object as a parameter from Simulink, it is not sufficient that the reference object exists in the GT-SUITE model, it is also necessary that the reference object be used in the model. If the desired reference objects are not used in the GT model, it is recommended to include them in an unused case. Otherwise, the GT-SUITE model strips off the unused reference objects during dat file creation, and when this object is called from Simulink, a fatal error will occur.

Example: If there are 3 parameters *rpm*, *throttle* and *profile* defined in the GT-Suite model with their values defined in Case Setup, their values can be modified from the GT S-function by entering the following values in the Parameter Names/Values:

Parameter Names: '*rpm throttle profile*'

Parameter Values: '*3000 45 myprofile*'

It is important to note that if there are any GT-SUITE parameters in Case Setup that depend on a formula that contains other Case Setup parameters that are also being modified via Simulink,



these dependent values will only be updated if the "GT-SUITE Model File" is specified with the '.gtm' extension.

- **Show Run Simulation Options:** Selecting this allows the user to change the following runtime options:
 - **Number of Cores Used:** Chose number of cores used for the simulation. Should a number greater than 1 be entered, GT-SUITE will invoke the parallel solver and will assign the selected number of cores. This is most advantageous for flow, thermal and lubrication models.
 - **Runtime Interface:** Select which user interface will display during the simulation, or select to show no interface during the run. Note: when the 'console window' option is selected using Linux, this is referring to the mode when the user is executing Matlab from the Linux console window command line and not the typical Matlab user interface.
 - **Runtime Monitors:** Allows the user to disable, or enable, all runtime monitors in the GT-SUITE model.
 - **Solver Architecture:** Select between 32bit and 64 bit architectures.
 - **Solver Archive:** Should a different GT-SUITE solver archive be available, the number can be entered here for use during the simulation.
- **Show Advanced Options:** With this option checked, the following options are revealed for use:
 - **When Simulink Stops, Halt GT Model Immediately:** With the 'Simulation Duration Source' set to 'Simulink', check this option to ensure that the GT model will halt when Simulink determines the simulation must end. Without this checked on, the GT solver will continue to run until a final cycle completes even if Simulink has stopped.
 - **Delayed Initialization:** Should the GT s-function block reside in an enabled, or triggered, subsystem, check this box to ensure the GT-SUITE solver will only execute when the subsystem is actually enabled.
 - **Results will be Combined with other Cases:** By default, regardless of what case is specified in the 'Case No.' field, the results will show in GT-POST as 'Case 1'. By selecting this option, the results will be stored in such a way where the actual case number is associated when the particular run. This allows the user to select different cases for different simulations and then combine the results into one GT-POST file. Contact support@gtissoft.com for further information.
 - **Output I/O List in Matlab Command Window:** When checked, the input and output signal descriptions as entered in the 'SimulinkHarness' part will be displayed in the Matlab command window.
 - **Direct Feedthrough:** Check this option on to specify the GT s-function as being a direct feed through block. For more information, please consult the Simulink help documentation.

9. The simulation will be controlled from Simulink, and can be run as any other Simulink simulation would be run. The user has the option to choose whether the simulation duration is controlled by Simulink or GT-SUITE:



- If the **Simulation Duration Source** is set to "SIMULINK" the GT-SUITE simulation duration will automatically be updated to match the Simulink duration.
 - If the **Simulation Duration Source** is set to "GT-SUITE", then the user must set the Simulink Stop Time to a value larger than the GT-SUITE simulation duration. Once GT-SUITE completes the run, it commands Simulink to stop. For example, if the GT-SUITE model is set to run for 5 seconds, then the Simulink model must be set to run longer (e.g. 100 seconds). Once the GT-SUITE model reaches 5 seconds and completes the final cycle, it forces Simulink to stop simultaneously. This feature is very useful for cases where the user wants to stop the co-simulation at a specific number of cycles or when a convergence criterion is met. In addition, it guarantees valuable and accessible results even for the last cycle; since the Simulink signals are not held constant (the signals are not frozen at specific values and are being updated as the simulation continues).
 - If the user stops the simulation prematurely by clicking on the "Stop" button in the Simulink model window, the GT-SUITE simulation will also stop, after completing the cycle that it was running, plus one more cycle in order to produce the output file (.gdx) - provided the "When Simulink Stops, Halt GT Model Immediately" option is not checked. During the last cycle all inputs from Simulink to GT-SUITE will be held at their last calculated values.
10. When a GT-SUITE simulation is launched from GT-ISE, a command prompt window, or GUI window, displays messages and information regarding the simulation status. When the simulation is launched from Simulink, and not on a PC platform, the command prompt window is not available, and all the status information is instead written to a file named *.out, where "*" denotes the filename of the GT-SUITE model. If the MATLAB command window indicates that the GT-SUITE simulation has stopped due to an error, the *.out file can be reviewed for error messages. On some UNIX platforms, the simulation status information is written to the MATLAB command window instead of the *.out file.
 11. If the GT-SUITE simulation stops due to an error, a message will be displayed in the MATLAB command window and the Simulink simulation will also stop.
 12. When opening existing Simulink models, the user must make sure to change to the correct directory containing the GT-SUITE model file (.dat or .gtm). This could be done either by using the MATLAB Path Browser or the MATLAB "cd" command. If the user starts Simulink and loads the Simulink model using the File/Open option in a Simulink model window, MATLAB will not know the correct location of the GT-SUITE model file. This will result in an error.
 13. GT-SUITE models which contain 'SimulinkHarness' components can also be run from GT-ISE. Any 'ActuatorConn' connections will be ignored and the values to be actuated will be kept constant at their initial values. This is useful for checking models for errors before running a combined GT-SUITE – Simulink simulation.

Two or more GT-SUITE models can be coupled with a single Simulink model. A typical example is where the engine and driveline are modeled in GT-SUITE, and the electronic controllers are modeled in Simulink. Any GT-SUITE control signal of each GT-SUITE model can be passed to or from Simulink via an individual 'SimulinkHarness' component. For example, engine speed can be sensed in GT-SUITE and then passed to Simulink, vehicle speed can be sensed in GT-SUITE and then passed to Simulink, and Simulink responds by controlling the gear number in the GT-SUITE model.

Note: Some antivirus and firewall solutions monitor the communication between Simulink and GT-SUITE and may result in significant performance degradation on this communication link. If you experience slow communication between Simulink and GT-SUITE, please try to create special firewall/antivirus configuration settings that exclude this traffic from being monitored, or temporarily



disable your firewall/antivirus solution or, as a last resort, consider uninstalling it. Please refer to chapter 1.2.4 for more details.

1.2.3 Running GT-SUITE and Simulink on Different Workstations/Operating Systems

If a coupled simulation designed to run from Simulink is to be run on different platforms, then the user must perform the following steps (it is assumed that "client" is the computer that will run Simulink and "server" is the computer that will run GT-SUITE). The following steps do not apply when choosing to compile the Simulink model and run from GT-SUITE. Instead, standard remote/distributed run procedure applies.

- (1) Create a new file, `gtlink.prm`, in the Simulink working directory, and specify the server's hostname (i.e. the computer on which the GT-SUITE solver will run) and a port number. For example the `gtlink.prm` file may look like,

```
HOST=MyPC
PORT=5556
```

- (2) Start the server manually by typing the following on a shell or command line:

```
gtlink -s:<port>
```

where `<port>` is the port number, e.g. 5556. Note the brackets are not included.

- (3) Start the coupled simulation from the client machine.

Note that during the simulation, two log files are created providing diagnostic information, **gtlinkclient.out** and **gtlinkserver.out**. In addition, environment variables, **GTIDEBUG** and **GTIVERBOSE**, may be set to 1 for additional output information during debugging. The variable **GTIVERBOSE** provides more detailed information than does **GTIDEBUG**. Please note that these two environment variables should not be set unless problems occur as the writing of diagnostic information slows down the simulation.



1.2.4 Resolving performance issues

Some antivirus and firewall solutions monitor the communication between Simulink and GT-SUITE, which may result in significant performance degradation on this communication link. This chapter describes the problem and proposes ways to improve performance.

Symptoms and Cause

Simulation execution speed is significantly lower compared to when running the model standalone, assuming identical timestepping. Antivirus and firewall solutions monitor processes in real-time to ensure that they do not pose a risk for the system. Depending on their configuration and their monitoring algorithms, they may be adding considerable overhead to the messages exchanged between Simulink and GT-SUITE during a coupled simulation. Please note that this is more visible with fast GT-SUITE models, i.e. when the communication overhead is relatively significant compared to the overall simulation processing.

Solution

Due to the nature of the issue, there is no universal solution that works for all antivirus/firewall products. A solution has to be found on a case-by-case basis. Gamma Technologies provides some general guidelines applicable to all products and specific instructions for products for which a solution has already been investigated and found.

For all performance degradation problems experienced with simulation coupling, you are encouraged to contact GT Support, so that we work with your antivirus/firewall solution vendor and resolve the issue.

The following guidelines may be helpful:

- a. Create special antivirus/firewall configuration settings that exclude this traffic from being monitored
- b. Temporarily disable the real-time monitoring process of your antivirus/firewall solution
- c. Temporarily disable your antivirus/firewall solution
- d. Temporarily disable your backup service that sometimes comes together with antivirus/firewall solutions
- e. Consider uninstalling your antivirus/firewall solution

For Symantec Endpoint Protection (SEP), Gamma Technologies provides a small application (script) that configures SEP for high simulation-coupling performance. Please note that these configuration settings are not available through the SEP User Interface. Please contact GT Support and ask for this special script.

For Microsoft Security Essentials, please follow the steps below:

- a. Open Microsoft Security Essentials panel
- b. Go to Settings tab
- c. Click on the 'real-time protection' option
- d. Uncheck the 'Turn on real-time protection' checkbox



For Trend Micro OfficeScan, please follow the steps below, either locally through the ‘OfficeScan client’ application (if necessary permissions are granted) or remotely through the OfficeScan web server console:


- a. Disable firewall, either locally through the ‘OfficeScan client’ application (if required permissions are granted) or remotely through the OfficeScan web server console
- b. Disable TMACTMON service by changing
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\tmactmon\Start value to 4
- c. Disable TMEVTMGR service by changing
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\tmeytmgr\Start value to 4
- d. Disable real-time monitoring, either locally through the ‘OfficeScan client’ application (if required permissions are granted) or remotely through the OfficeScan web server console

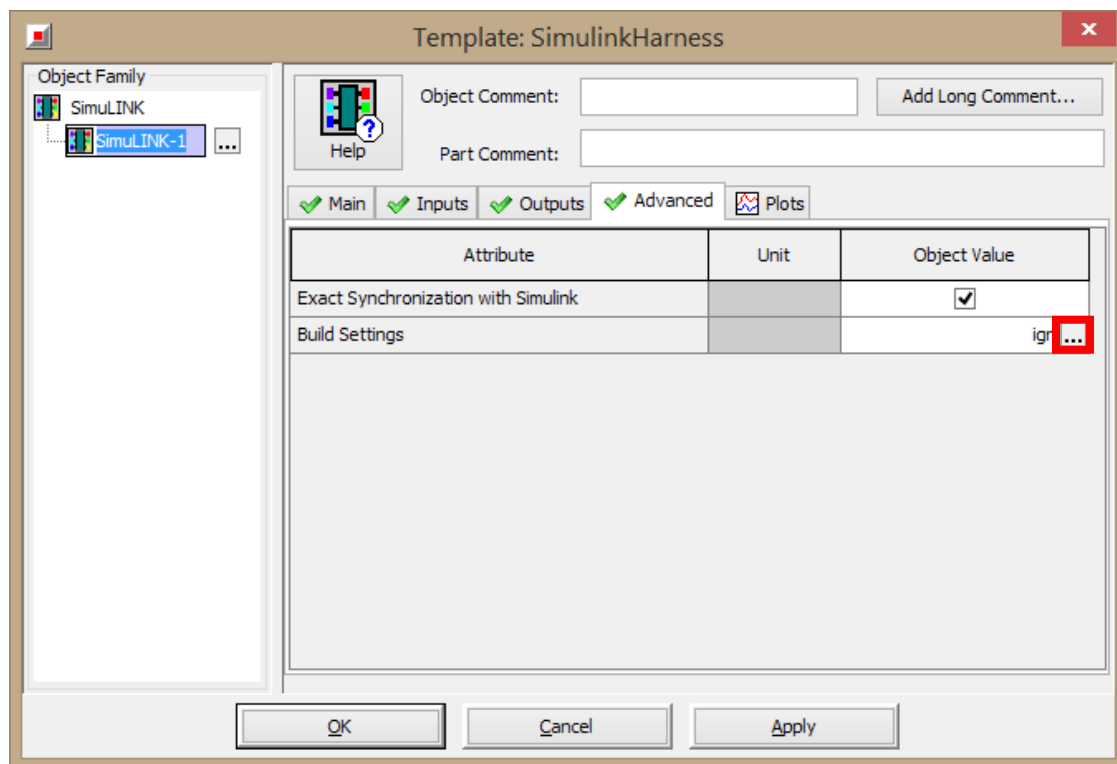


1.3 Building s-functions using GT-SUITE for use in Simulink

Should the user have a compiler installed, they can generate a model specific s-function for use in Simulink. A reference object called 'BuildForSimulink' can be used in the 'Build Settings' field of the SimulinkHarness template that, when invoked, will generate code, as well as, a simple reference model that contains an s-function of the GT-SUITE model.

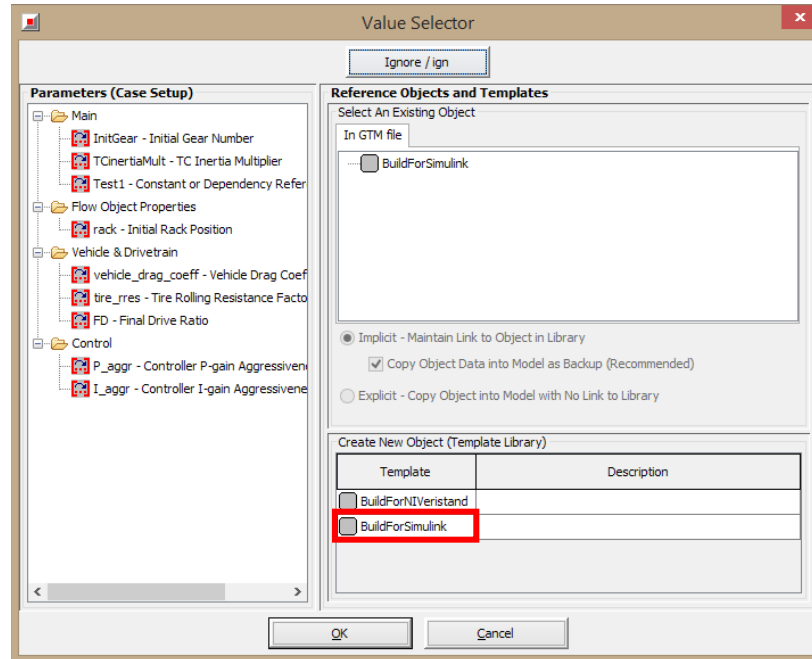
Note: Access to a GT-SUITE DL, or solver, license is required when running from Simulink in this fashion.


1. Build the model in GT-SUITE and add a SimulinkHarness template. Select 'run_from_Simulink' and configure the inputs and outputs as you normally would.
2. Under the 'Advanced' folder, click the Value Selector, , in the 'Build Settings' field.

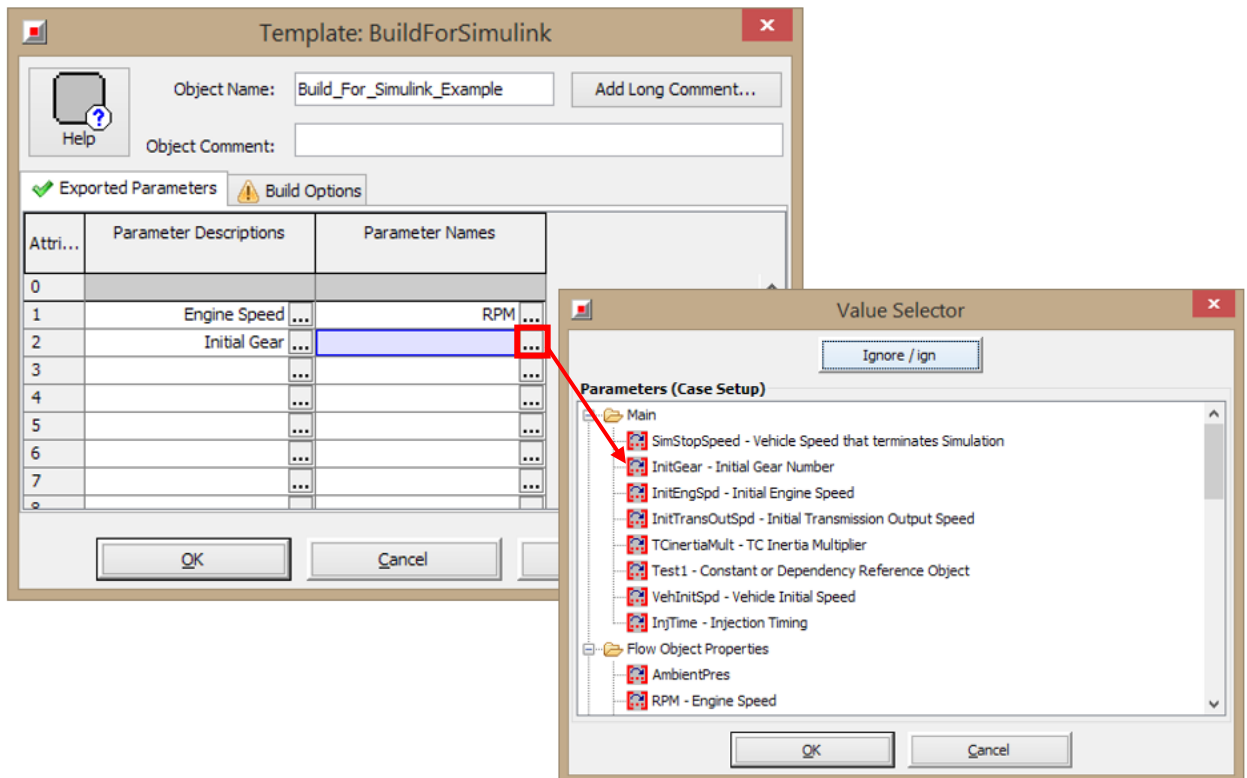


3. The 'Value Selector' dialog box appears and lists the available reference objects for this field. Select to create a new 'BuildForSimulink' reference object.

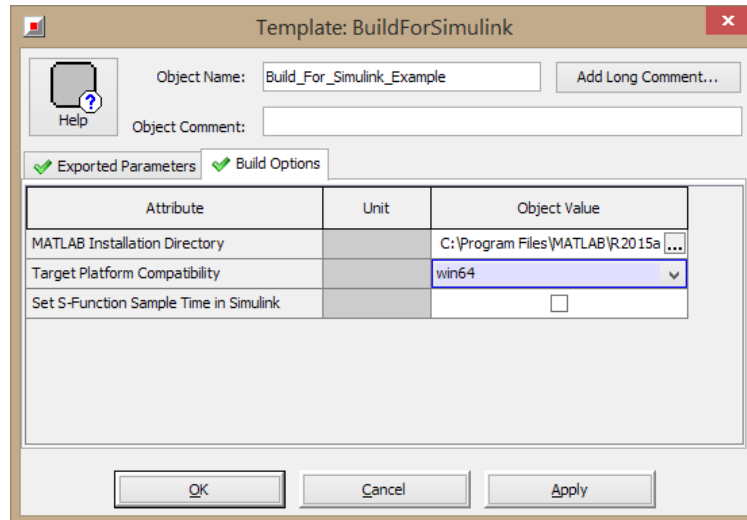




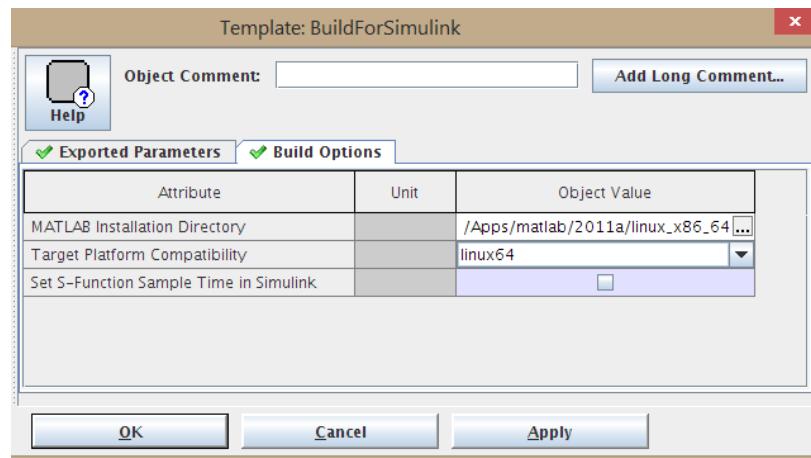
- The 'Create Object' dialog box appears. Here enter the description and parameter name for each of the attributes from Case Setup you wish to tune from Simulink. It is highly recommended that the Value Selector, , be used in order to select the parameter names.



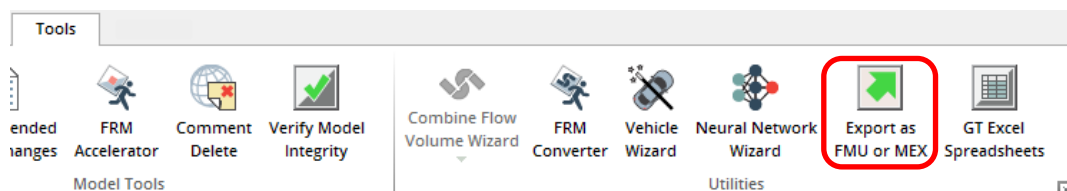
- Next, click on the 'Build Options' folder. Indicate the location of the desired Matlab installation and which platform will be used. For example, the figure below indicates a 64 bit version of Matlab 2015a will be used.



Note: Linux users must ensure the 'linux_x86_64' directory is included in the path. For example:



- Click 'OK'.
- Verify the GT-SUITE model runs without error.
- Click Tools\Export Model as FMU or MEX

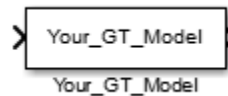


During the build process Matlab is launched and closed. Once the process is complete, the GT-SUITE scoreboard indicates a successful build.

```
INFO Generating code for Simulink S-Function
INFO Building S-Function...
INFO Model generated successfully.
```

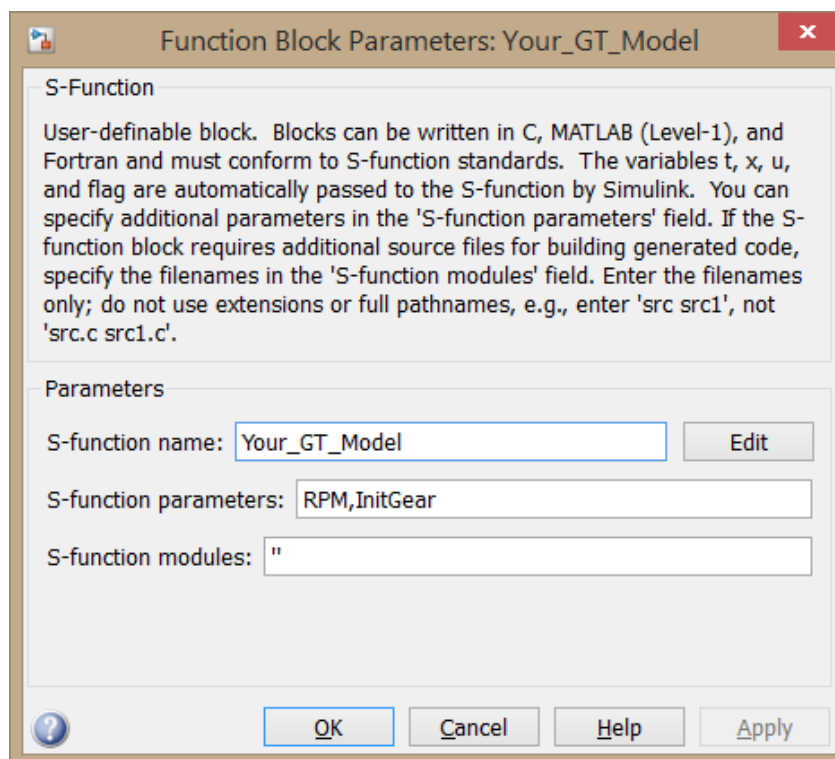
Note: It is recommended that the user first verify the appropriate compiler is selected in Matlab by typing 'mex -setup' in the command window.

9. Open Matlab and navigate to the directory that contained the GT-SUITE model. Notice there is a 'Your_GT_Model_smlk.mdl' file as well as a "Your_GT_Model _resrc' directory and 'Your_GT_Model.mexw64' Matlab executable. These are the build products from the 'Export as FMU or MEX' function in GT-SUITE. Open the 'Your_GT_Model_smlk.mdl' file. The model contains a single block that is an s-function.



Double-click on this block to see the parameters of the s-function. Note they are identical to the parameter names entered in the 'BuildForSimulink' reference object.





10. Place this s-function block into your working Simulink model and connect as required. When the simulation is executed, this s-function block calls the GT-SUITE model and solver.

Additional Requirements for Air Conditioning Applications:

Running GT-SUITE models with Air Conditioning components using refrigerants require additional files to be placed in the *_resrc directory:

For pure refrigerants:

1. All the .FLD files that correspond to refrigerants in the model. These files are located in GTIHOME/resrc/fluids with standard GT-SUITE installations. For example, if R134a is used in the model, R134a.FLD and its respective R134a.gtixxxx.FLD file in GTIHOME/v7.4
2. /resrc/fluids must be copied to the RT machine's working directory.
3. HMX.BNC from GTIHOME/v2016/resrc/fluids.
4. These additional 6 .FLD files found in GTIHOME/v7.4
5. /resrc/fluids (many refrigerants depend on property models from these refrigerants):
 - Nitrogen.FLD
 - Propane.FLD
 - R113.FLD
 - R12.FLD
 - R124.FLD
 - R134A.FLD

For mixtures:

1. All the .MIX files (including the *.gtixxxx.MIX files) that correspond to mixtures in the model. These files are located in GTIHOME/v7.4
2. .0/resrc/mixtures with standard installations.



3. All .FLD files corresponding to refrigerants that are contained within the mixtures in the model. These can be found by opening the .MIX files in a text editor.
4. Items 2-3 from the instructions for pure refrigerants above.



1.4 Using GT-SUITE Neural Networks in a Simulink Model

It is possible to utilize GT-SUITE generated Neural Networks in any Simulink model without coupling to GT-SUITE (i.e. no license is required). To accomplish this, a special S-function block is utilized. Please note that this block will only execute the 3-layer feedforward neural network architecture. The files needed for this feature are not available in a default installation of GT-SUITE, please contact GT if interested in this feature.

1.4.1 Prerequisites

- Operating System: Windows or Linux 32/64 bit
- GT-SUITE V71 B3 or later
- Matlab 2006b or later
- Simulink Coder (formerly Real Time Workshop) (if desiring to compile model into executable)
- 3-layer feed forward neural network output (.nno) file(s) generated with GT-SUITE
- Files provided by GT have been placed in safe directory, for example
GTIHOME\2016\NeuralNets

1.4.2 Instructions

The following instructions apply when running a Simulink model from Matlab:

- a. Add the folder containing the GT supplied Neural Network files to the Matlab path (either by using the “addpath” command or File -> Set Path).
- b. Add the Neural Net S-function block to a Simulink model by selecting the “GT-NeuralNet2016” library from the Simulink Library Browser.
- c. Use a Mux block to vectorize the signals that will serve as inputs to the GT-SUITE Neural Net block.
- d. Connect the Mux block with GT-SUITE Neural Net block. Note that the number of signals into the block should coincide with the number of input signals that the trained neural network expects.
- e. Connect the output of the GT-SUITE Neural Net block with the rest of the model. Note that the output is always single-valued.
- f. Double-click on the GT-SUITE Neural Net block. There is only one parameter to be filled; a vector of neural network parameters is expected. To this end, open the .nno file and copy the neural network parameters into the block. Specifically, mark and copy all the values written under the following section:

```
*****
W**** THE DATA BELOW SHOULD NOT BE MANUALLY EDITED
*****
```

The set of values that will be copied must be included in brackets, e.g. [10 7 1 ...] so that they form a MATLAB vector. Alternatively, declare a vector variable in MATLAB, copy the parameters to this variable and specify this variable inside the GT-SUITE Neural Net block.

The source code for this S-function block is provided, so integration with Real Time Workshop is supported. In order to compile a model into an executable, the following setting must be adjusted in the configuration settings of RTW:

- In Simulink, navigate to Simulation -> Configuration Parameters->Real-Time Workshop -> Custom Code
- In the “Include custom C code in generated”, add “gtneuralnet2016.c” to the list
- Compile mode



CHAPTER 2: GT-SUITE-RT - Real-Time Modeling

GT-SUITE-RT is a special license which allows the user to utilize a speed optimized solver which will run the model 30-50% faster than the standard solver and with less computational fluctuations. The GT-SUITE-RT license and its related runtime libraries are designed for use only for applications that are to be run on Hardware-in-the-Loop (HiL) systems, with hard real time requirements. The model generated by the license can be used on the desktop as part of the validation of the HiL workflow and process. To this end, the GT-SUITE-RT solver is provided as an executable, dynamic library and static library so that it can be used from GT-ISE, Simulink, & HiL applications. It can run models locally on a PC and on Hardware-in-the-Loop (HiL) platforms such as National Instrument's Veristand, dSPACE ds1006, and dSPACE SCALEXIO systems (see below for all platforms).

From a high-level standpoint, the principle is that the user downloads the GTSUITE-RT package, builds the real-time application using the source code, libraries, scripts and makefiles of the appropriate 'make' folder, downloads it to the target machine and then runs the HiL experiment using the applications and scripts of the appropriate 'bin' folder.

2.1 Licensing

The GT-SUITE-RT licensing differs from standard licensing because licensing checks in the middle of a HiL application are unacceptable. As such the GT-SUITE-RT license governs the creation of special encrypted *.dat files from GT-ISE. Additionally, the GT-SUITE-RT license is not "cheeked out" from the server by a given user, but the mere presence of the license on the server gives anyone who is running GT-ISE, the ability to generate the RT dat file. This dat file is the input file to the solver and specifies the model properties. The RT model may be run indefinitely without further licensing checks, as long as the model is not modified in any way, not even the filename. However, the executable which is to be run from GT-ISE in the "stand alone validation" tests below will check for a GT-SUITE-RT license at run time.

2.2 HiLPlatforms

GT-SUITE-RT is very easy for a vendor to implement and as such Gamma has not been directly involved with all implementations and subsequent testing. GT-SUITE-RT is known to have worked on the following platforms: AandD Procyon, Cosateq Scale-RT, Cybermetrix CyFlex, dSPACE DS1006 quadcore, dSPACE SCALEXiO, ETAS LABCAR, Mathworks xPCTarget, National Instruments Veristand, Realtime Technologies SimCreator.

The paragraphs below will detail platforms where GT has worked closely with the vendor to implement GT-SUITE-RT. We will try to provide a bit of commentary on each system, as we are commonly asked for our observations about the various options. For those platforms which are not specifically mentioned below, we do not have information on the quality of the services/tools of those vendors since the integration was done completely on the vendor's side and therefore we have not worked with their tools.

dSpace

If you have a **DS1006 quad core** platform the dSpace Advanced Support Package is required which is basically the software that enables the loading of a different operating system on the 4th core of the system. This allows GT-SUITE-RT to run on 4th core of the board, while the rest of the Simulink model



lies on other cores. The solution is straightforward and user friendly. In addition the **SCALEXiO** system is supported and runs natively on any core specified by the user.

ETAS

ETAS provides a flexible system (ETAS LABCAR). Integration with this system does not require extra hardware. ETAS uses off-the-shelf x86-based hardware and software components thus eliminating compatibility issues. There is no additional board needed to run GT-SUITE-RT and our solver is simply linked in the model as one more runtime library. One of the advantages of ETAS LABCAR is that the user is free to choose the CPU of the system and thus to control the balance between cost and execution speed.

National Instruments

We also allow straightforward integration with National Instruments "NI Veristand" tool for HiL simulations. It is worth noting that the generation of the NI Veristand compatible model is done automatically from the GT-SUITE environment. More information on this feature can be found in the Template Reference Manual.

Mathworks

The Mathworks "xPC Target" HiL simulator is also supported. Integration with this tool requires no additional software or hardware and follows the typical workflow suggested by the Mathworks.

2.3 Download and Installation (for both PC and target machine):

The RT packages are not shipped with the standard installation and must be downloaded from our web site. For users with a valid GT-SUITE-RT license, the files are downloadable from our website in the same manner as downloading an update. If you do not have download rights, then your IT administrator who is responsible for downloading the official installation should be entitled to them. Specifically, first visit <http://www.gtisoft.com> and log in using your credentials (or register if you have not done so yet). Next, from the Support menu, select Downloads -> Updates and choose "2016 Realtime Packages". Click on the Build number of your choice and the list of various integration packages will appear. Select the configuration(s) that apply to your host and target machine.

Once the zip file is obtained, the following steps need to be completed (one time only):

1. **On Target Machine** Extract the zip file to the desired location on the machine (referred to later as *RTHOME*). Please note that a full GT-SUITE installation is NOT required for this step. Only the files contained in the supplied zip file should be necessary to perform a coupled simulation using the RT libraries. However, if the machine that already has a full GT-SUITE installation, the RealTime folder will already exist as \$GTIHOME/v2016
2. /simulink/RealTime. Please copy-paste the downloaded folders into this location overriding any existing files that have the same name.
3. **On PC**, Modify the "**Path**" environment variable to include the following three directories:
 - a. *RTHOME*\GTSuiteRT
 - b. *RTHOME*\sfunction
 - c. *RTHOME*\src



4. **On Linux**, Modify the "**LD_LIBRARY_PATH**" environment variable to include the following three directories:
 - a. *RTHOME/GTSuiteRT*
 - b. *RTHOME/sfunction*
 - c. *RTHOME/src*

5. Open MATLAB and add *RTHOME/sfunction*, *RTHOME/src* and *RTHOME/GTSuiteRT* to the Matlab Path. This can be done via the File menu in Matlab (File->Set Path.. ->Add Folder). Once you have saved and closed this dialog, you should see "GT-Suite2016RT" in the Library Browser of Simulink. For MATLAB versions R2014b and higher, the "GT-Suite2016RT" category may not appear in the Library Browser. In this case, an underlined 'Fix' should appear on the top of the Simulink Library window. Selecting it and resaving as SLX format allows the "GT-Suite2016RT" category to appear.

Additional Requirements for Air Conditioning Real Time Applications:

Running GT-SUITE models with Air Conditioning components using refrigerants require additional files to be placed in the **working directory** of the RT machine. The following files must be placed in the working directory of the RT machine:

For pure refrigerants:

- All the .FLD files that correspond to refrigerants in the model. These files are located in GTIHOME/resrc/fluids with standard GT-SUITE installations. For example, if R134a is used in the model, R134a.FLD and its respective R134a.gtixxxx.FLD file in GTIHOME/v2016
- /resrc/fluids must be copied to the RT machine's working directory.
- HMX.BNC from GTIHOME/v2016/resrc/fluids.
- These additional 6 .FLD files found in GTIHOME/v2016
- /resrc/fluids (many refrigerants depend on property models from these refrigerants):
 - Nitrogen.FLD
 - Propane.FLD
 - R113.FLD
 - R12.FLD
 - R124.FLD
 - R134A.FLD
 -

For mixtures:

- All the .MIX files (including the *.gtixxxx.MIX files) that correspond to mixtures in the model. These files are located in GTIHOME/v2016
- /resrc/mixtures with standard installations.
- All .FLD files corresponding to refrigerants that are contained within the mixtures in the model. These can be found by opening the .MIX files in a text editor.
- Items 2-3 from the instructions for pure refrigerants above.

2.4 GT-SUITE on Real-Time HiL (Hardware in Loop) Systems

It is possible to run a GT-SUITE simulation at real time (RT), assuming that RT hardware and software is used, and that specific modeling practices are followed. RT simulation is primarily required for



Hardware In the Loop (HIL) testing, where a GT-SUITE model is used to represent the "plant" for engine or vehicle controls development. The following section includes a description of the real time capabilities and requirements of GT-SUITE as well as process used to run RT.

GT-SUITE Real Time Capabilities and Requirements:

To run a GT-SUITE simulation at real time, the simulation process must first be capable of running faster than clock speed at all times during the simulation. Then, specific hardware/software is used to slow the simulation down so that it runs at exactly real time. Most GT-SUITE simulations today run slower than real time, sometimes significantly so. For example, a typical GT-POWER engine model may run somewhere between 100-200 times slower than real time. This implies that only specific models will be capable of achieving real time. For engine models (the most typical use for RT simulation), it is possible to convert a detailed model to a mean value model. In a mean value model, the high frequency resolution is sacrificed to achieve larger calculation time steps while lower frequency accuracy is maintained at an acceptable level. This model reduction process is covered step by step in a GT-POWER tutorial.

Once an RT capable model is available, it is run using a special GT-SUITE library compiled specifically for RT operation. This library file contains all of the elements of the standard GT-SUITE solver that are needed for typical RT simulations, but the code has been optimized for speed. This library file is supplied by GT upon request once a GT-SUITE-RT license has been purchased. The simulation is done via a coupled Simulink run and must be run on special RT hardware/software. The RT hardware/software solution can be provided by several suppliers. Please contact GT for the most recent list of supported RT solution providers). The following are the requirements to run RT:

1. Real Time Hardware/Software solution provided by a third party supplier
2. GT-SUITE v2016
3. on a separate "regular" machine (for building models to be run on the RT machine)
4. GT-SUITE-RT License accessible by the model building machine (not the RT machine)
5. RT capable GT-SUITE model file with the project type set to GT-SUITE-RT
6. RT library zip file provided by GT (supported platforms are Windows 32 bit and 64 bit, and Linux_x86 32 and 64 bit) See Installation section for details.

The following sections detail the required steps to run a real time GT-SUITE simulation.

1. Create a Real Time capable GT-SUITE model:

Before an RT simulation can be run on the target RT machine, the GT-SUITE model must be created on a standard machine with GT-SUITE installed.

On the model building machine, modify the GT-SUITE model as necessary to minimize CPU time while maintaining the desired level of accuracy. For engine models, a specific process has been developed that can be used to reduce a detailed model to a mean value model. Please refer to the GT-POWER Tutorials for the step by step model reduction process.

Convert the model to use a GT-SUITE-RT license if it is not already configured to do so. This can be done through File > Change License Used... in GT-ISE. The conversion of project type may generate errors if unsupported templates exist in the model. All of these errors must be handled. The following is a general list of templates (or categories of templates) available in the standard GT-SUITE version that are not supported for GT-SUITE-RT:

- a. All Acoustics templates
- b. 'RLTSensor' and 'RLTCreator' templates
- c. 'SampledOutput' template



- d. All Monitor templates ('MonitorSignal', 'MonitorXY', 'MonitorCaseRLT')
- e. 'ActiveDial' template
- f. All Hydraulic templates
- g. All Valvetrain templates
- h. All mechanical templates that cannot be expected to achieve RT with reasonable results due to the need for very small timesteps (i.e. 'TireConn').

If RLTDpendence objects are used in the model, the RLTs used by these objects have to be created using the RLTCreatorRT template. No RLTs are stored in GT-SUITE-RT without creating them using RLTCreatorRT.

Once the model has been modified as needed so that it can be converted to the GT-SUITE-RT project type, it is possible to set up controls components to pass any desired output from the GT-SUITE model to the Simulink model. This could be useful when the GT-SUITE model is integrated in the Real Time hardware.

Any instantaneous output of interest will then be sensed with a 'SensorConn' and passed out of the GT-SUITE model using the 'SimulinkHarness' component.

Any RLT output of interest will have to be created by first using a 'SensorConn' to sense an instantaneous quantity. When not running in standalone, all result storage usually takes place in an external application, so the signal must be passed out of the GT-SUITE model using the 'SimulinkHarness' component. Other controls components can be used as necessary, either in the GT-SUITE model or in the co-simulation model, to modify the sensed signal to calculate the equivalent of the RLT of interest (for example to take an average over a cycle).

In the following step, the model will be run as a stand-alone model without coupling to Simulink. Therefore, it may be desirable to set initial output values in 'SimulinkHarness' (inputs to the GT-SUITE model from the Simulink model) that will be kept during the entire simulation, so that reasonable values are passed to the GT-SUITE model. Note that the 'SimulinkHarness' will not cause any problems in a stand-alone model without defined initial outputs, as it serves as a simple signal terminator.

2. Verify that the GT-SUITE model runs "stand-alone" without errors:

Once the RT model has been created, it is desirable to verify that the model will run without errors using the RT solver. The model can be run from GT-ISE as any other GT-SUITE model would be run (using the Run Simulation button). Because the project type has been set to GT-SUITE-RT, the RT version of the solver executable will be used. This executable is identical to the RT library file that will eventually be used on the RT machine, but has been compiled as an executable version to facilitate running "stand-alone" simulations from GT-ISE without requiring coupling to Simulink.

Running an RT simulation using the executable version requires an RT license. This license is checked out during the simulation and checked back in at the end of simulation in the same way that standard GT-SUITE licenses are used.

The main purpose of running this simulation is to verify that no fatal errors are generated when running the model. However, when the good inputs are provided thanks to signal generators, consistency of results can easily be verified in GT-POST. A secondary purpose for running this stand-alone simulation is to get an approximate idea of the ratio of CPU time to clock time. If this ratio is significantly less than 1, then it is likely that the model will run at real time on the RT machine. If the ratio is 1 or larger, then it is likely that further modifications to the model are required to achieve real time. This CPU time test is




not a perfect test because the RT machine may have a faster or slower CPU than the model building machine and in some cases may be a different platform. In addition, there is some factor of safety required so that the RT machine can accomplish other tasks including communication between Simulink and GT-SUITE, while still maintaining RT operation. However, it is still a good "sanity" check of model performance using the RT solver. With experience, it should be possible to develop a CPU:clock time ratio requirement for the specific build machine and RT machine hardware used.

3. Verify that the coupled Simulink/GT-SUITE simulation runs

Once the model has been successfully tested using the executable version of the RT solver, the next step in the process is to verify that a coupled simulation using both the GT-SUITE model and the Simulink model runs without errors and produces the desired output.

This step cannot be completed without obtaining the necessary GT-SUITE-RT library files from GT, as these files are not supplied on the installation CD. See the Installation section for details.

Once the above steps have been successfully completed, the remaining steps to run the coupled simulation are very similar to those of a standard GT-SUITE/Simulink co-simulation:

1. The GT-SUITE model should be configured so that the outputs of the Simulink model will be properly passed to the GT-SUITE model.
2. Generate a *modelname.dat* file for the GT-SUITE model (In GTISE; Run -> Run-Simulation Utilities -> Create .dat File...). Note that the project type **MUST** be set to GT-SUITE-RT for the .dat file to be created properly for the RT solver, and that an RT license is required for this step.
3. **When using a unique S-function**, the Simulink model should contain the GT-SUITE V2016
4. RT S-Function block, with all inputs filled in (pointing to the .dat file created in the previous step). This S-Function should be available in the Simulink library browser assuming the Matlab Path was modified properly in the previous steps.
5. **In case of the calculation of the In-Cylinder Pressure by a separate S-function**, an additional block "In-Cylinder Pressure" is available. The input port of this In-Cylinder Pressure S-function expects to be connected directly with one of the output ports of GT-SUITEv2016
6. RT (master) S-function. The master S-function provides useful quantities calculated at the sampling rate of the main engine model. The In-Cylinder Pressure S-function, uses these inputs to calculate In-Cylinder quantities. The output of the S-function contains 10 quantities: pressure (bar), temperature (K), volume (L), heat transfer coefficient (W/m²-K), cp(J/kg-K), gamma, sek, burned fuel fraction, burn rate (1/CA), trapped mass (g)
7. Start the co-simulation from within the Simulink environment by pressing the "Run"  button.

It is necessary to point out that a co-simulation run in this manner will not run at exactly real time, as there is nothing limiting the calculations to run at exactly real time. The purpose of this step is to verify that the co-simulation runs without error and that the desired outputs are obtained within the Simulink model. It also provides another opportunity to verify that the CPU:clock time ratio is has an acceptable margin prior to attempting a true RT simulation on the RT machine.

4. Set Up and Run the simulation on the RT machine

Once the co-simulation has been successfully tested in the previous steps, it should be possible to then run a real time simulation on dedicated RT hardware/software. The process of running on an RT machine is very different from that described in the previous step, and depends heavily on the third part supplier of the RT solution. For this reason, only a very generic description of the steps will be provided.



When running on an RT machine, the GT-SUITE-RT libraries and S-function block used in previous steps are not used directly. Instead, Simulink is used to generate C-code from the model. This C-code is then downloaded to the target (RT machine) and compiled into an executable using the supplier-specific libraries that handle real-time issues like synchronization. This executable is then run on the target RT machine. All of the files required for these steps are available in the provided zip file (*RTHOME* in previous steps). Please refer to the documentation of the specific RT solution supplier for more exact details.

2.5 Compiling Simulink Models with GT-SUITE-RT S-function as Standalone Executables

Models that contain the GT-SUITE RT S-function may be compiled on a host machine into a standalone executable with the Rapid Simulation Target in Real Time Workshop. This does not require a HiL setup; the executable may be run on any machine with a similar architecture as the compiling machine. Please read the previous section regarding the use of the GT-SUITE-RT S-function, and ensure that the model runs properly standalone before attempting to compile.

Prerequisites

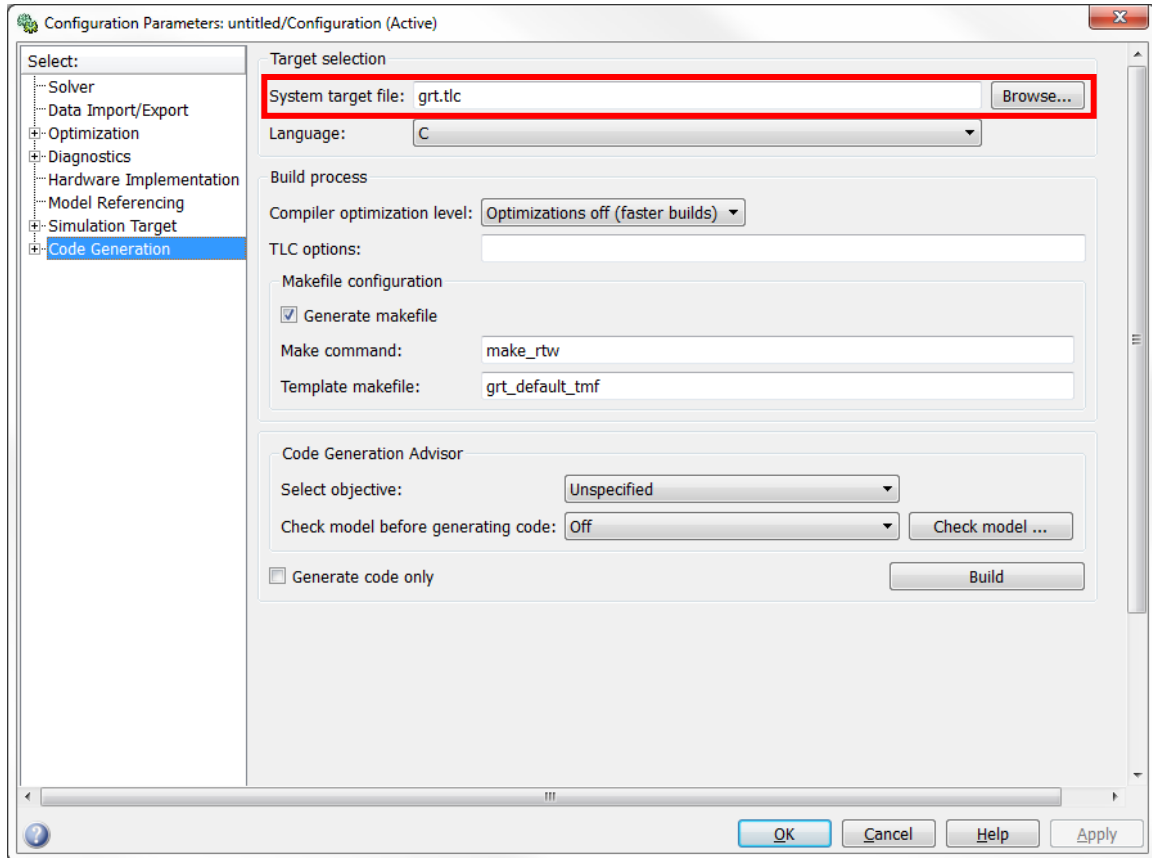
- 1) Matlab 2006b or later with Simulink Coder (formerly Real Time Workshop)
- 2) Compatible compiler (Windows: Visual Studio 2005, Linux: gcc)
- 3) Windows or Linux 32/64 bit
- 4) User has created a GT-SUITE-RT model which runs successfully standalone (see previous section for details)

Procedure

- 1) From the Simulink model menu, navigate to Simulation -> Configuration Parameters -> Code Generation and select the desired system target file (rsim, xpctarget, etc.)

Note: For Matlab 2012a and beyond the system target file is selected with the 'Target selection' pane under the Code Generation category. For versions of Matlab prior to 2011b, the system target file selection is found in the Real-Time Workshop category.

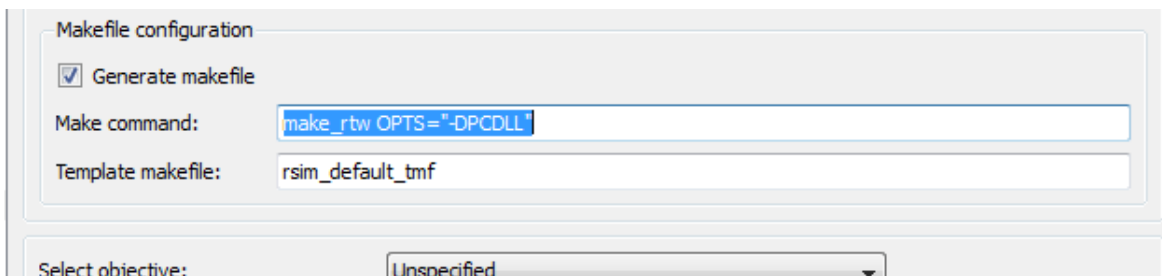




2) Add the following options to the 'make command' as shown below:

Do not copy and paste this option from manual.

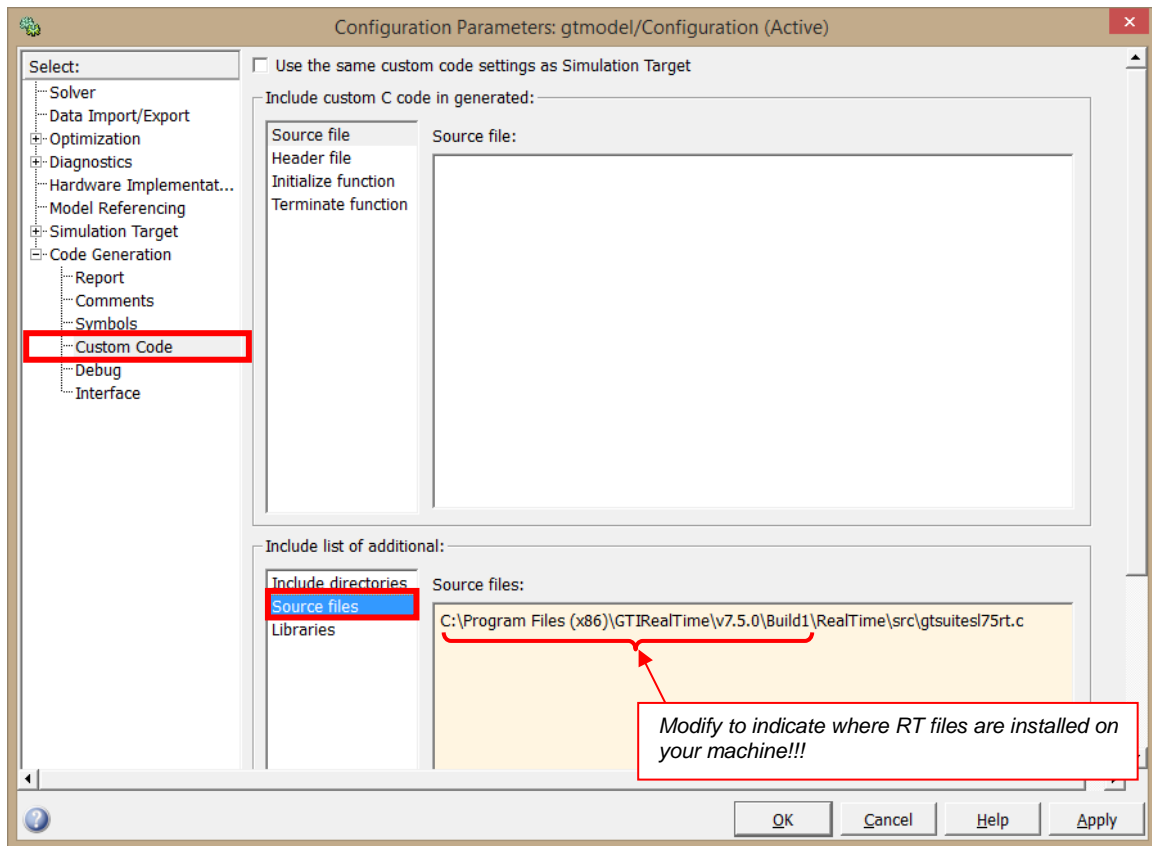
- Windows 32bit: OPTS="-DPCDLL"
- Windows 64bit: OPTS="-DWIN64"
- Linux: No additional options required



3) In the same window on the left pane, expand the Code Generation category and select 'Custom Code'. Choose the Source Files option, adding (full path may be required - RTHOME is where the RT files are installed):

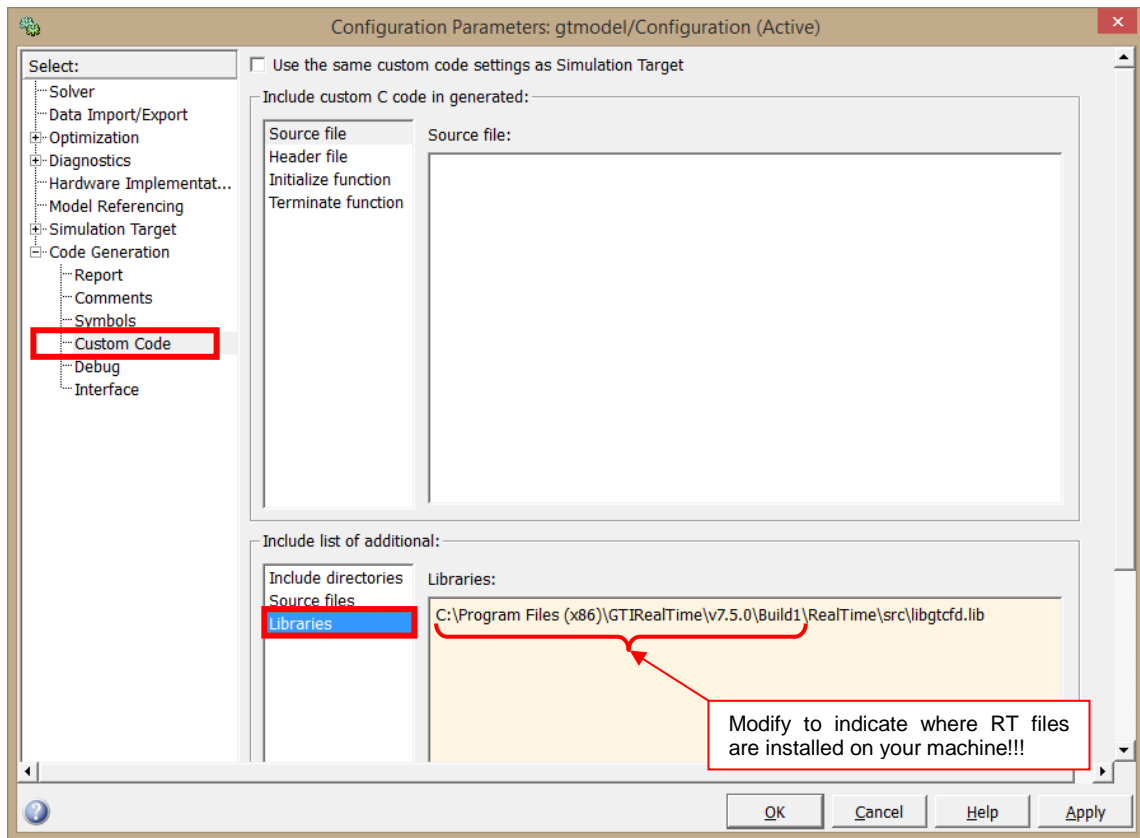
- RTHOME\src\gtsuitesl2016rt.c





- 4) Scroll down to the "Include list of additional pane" and select 'Libraries'. Add (full path may be required):
- Windows 32bit: RTHOME\src\libgtcfd.lib
 - Windows 64bit: First rename RTHOME\src\libgtcfd_64.lib to libgtcfd.lib, then specify RTHOME\src\libgtcfd.lib
 - Linux: RTHOME\GTSuiteRT\libgtcfd.so





- 5) Save changes and build Simulink model.
- 6) When running, the executable will still require RTHOME\src and RTHOME\GTSuiteRT to be included on the machine and in the LD_LIBRARY_PATH (Linux) or PATH (Windows) environment variable.

2.6 Common Errors

“The specified module could not be found”

This error is an indication that the setup instructions were not properly followed. Please add the folder RealTime\GTSuiteRT, which contains our solver libraries, in the PATH environment variable or copy the solver libraries into the working folder. Alternatively, if the user does not have administrator rights, they can use the following command in MATLAB, before they start the simulation:

```
setenv('PATH',strcat(getenv('PATH'),';C:\FFF\RealTime\GTSuiteRT'))
```

where FFF is the folder where they unzipped the RT package.

This temporarily adds the folder to the PATH environment variable without using admin rights.

Simulation dies immediately and is missing the *.out file

The most common cause of this problem is that the model file (i.e. .dat file) has not been placed in the working directory. It is very important that the dat file as well as all auxiliary files be placed in the



working folder of the machine that the simulation is running. For PC Simulink simulations, the working directory is the “Current Directory” in MATLAB. For HiL applications, the working folder varies, but details are mentioned in the instructions document per HiL system.

Simulation dies immediately and has a trivial *.out file

Change the “Messaging Level” in the GT-SUITE-RT S-function block to “All Messages” and re-run. The out file must be more verbose. If there is no “FAIL” message, please send the model to GT.

The results are not what was expected

Try to isolate the problem.

- If the model runs from Simulink, impose the same inputs to the model with SignalGenerators and run it from GT-ISE. If the problem goes away, it means that something may not be set up well in Simulink. Change the “Messaging Level” in the GT-SUITE-RT S-function block to “All Messages”, re-run and observe the .out file for unusual messages, warnings etc.
- If the problem persists, then switch to GT-SUITE license and rerun. If the problem goes away, then there is something wrong with the GT-SUITE-RT simplifications. One possible reason is the ODE solver which by default is the Euler in GT-SUITE-RT. Open RunSetup->ODEControl and switch the ODE solver to Explicit-Runge-Kutta. Also several other simplifications are located in Advanced Setup->RTSetup. The user may want to ensure that these simplifications do not hurt his results.
- If the problem persists, then it is a standard GT-SUITE problem and should be forwarded to standard support.



CHAPTER 3: Functional Mock-Up Interface (FMI) with GT-SUITE

The Functional Mock-Up Interface (FMI) is a standardized tool, originally initiated, organized, and headed by Daimler AG, which supports both model exchange and co-simulation of dynamic models created in different modeling environments. Starting in 7.4 Build 1, GT-SUITE supports the FMI standard.

GT-SUITE supports the FMI standard by importing and exporting Functional Mockup Units (FMU) into and out of the GT-SUITE modeling environment. According to the FMI standard, a FMU is a component which implements the Functional Mock-Up Interface. A FMU file is a zipped file (*.fmu) containing an XML file that describes the interface data and C code (or binary) that defines the functionality of the FMU.

The FMI standard supports two options for coupling dynamic models: FMI for Co-Simulation and FMI for Model-Exchange. The following table describes what portions of the FMI standard that GT-SUITE currently supports, and the earliest version of GT-SUITE that supports each.

Platform	↔ Model-Exchange		⌘ Co-Simulation	
	<i>Export</i>	<i>Import</i>	<i>Slave</i>	<i>Master</i>
win32	Not Available	V7.5 Build 1	V7.4 Build 2	V7.4 Build 1
win64	Not Available	V7.5 Build 1	V7.4 Build 2	V7.4 Build 1
linux64	Not Available	V7.5 Build 1	V2016 Build 1	V7.4 Build 2

3.1 Exporting an FMU (GT-SUITE as Slave)

The GT-ISE template named 'FMUExport' allows users to export models as an FMU, which will follow the FMI standards for Co-Simulation. In order to create an FMU, two things are needed

1. A model that runs in GT-SUITE with no failures or integrity errors as a standalone model (please see the section named "Checking Model Integrity as a Standalone Model" for details on how to test this) **AND**
2. Either Microsoft Visual Studio 2010 **OR** Microsoft Software Development Kit (SDK) 7.1 installed.

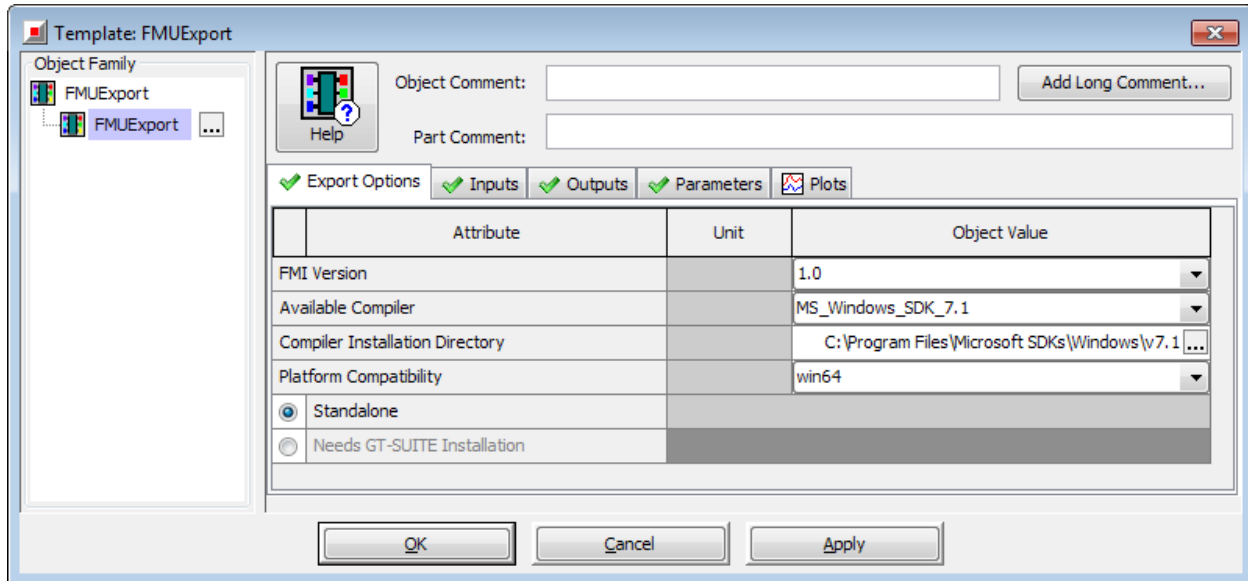
3.1.1 Creating an 'FMUExport' Object

When creating an object derived from 'FMUExport' the "Export Options" folder will define a number of things about the FMU including: choice of FMI version, compiler options, and choosing what platform the FMU will be created for (win32, linux64, etc.).

GT-SUITE is able to export FMUs that follow the FMI 1.0 or FMI 2.0 versions. In 32 or 64 bit Windows platforms, GT-SUITE can use the Microsoft Windows SDK 7.1 or Microsoft Visual Studio 2010 compilers to build FMUs. In a 64-bit Linux platform, GT-SUITE can use the GCC compiler to build FMUs.

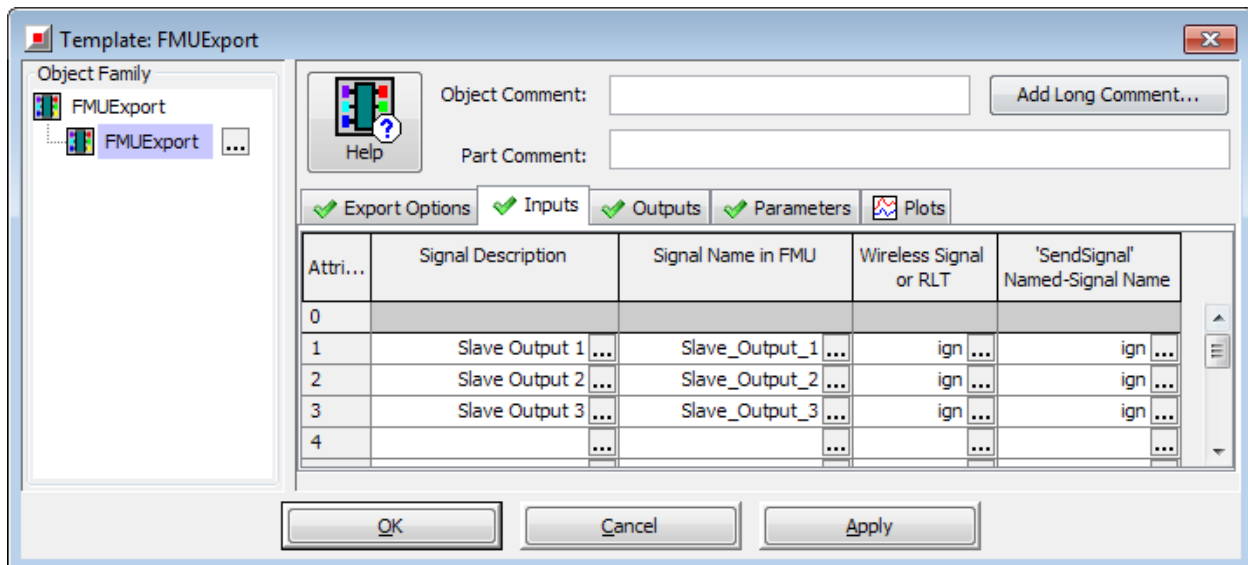
Additionally, GT-SUITE can export FMUs that follow the "Co-Simulation Standalone" or the "Co-Simulation Tool" standards defined by the FMI standard.





The "Inputs" folder will define the inputs into the 'FMUExport' part, which means that the signals defined in the "Inputs" folder will be FMU Outputs. The "Signal Description" and "Signal Name in FMU" will define the "name" and "description" attributes in the imbedded XML file in the FMU, respectively.

The "Signal Name or RLT Name" and "'SendSignal' Named-Signal Name" attributes are advanced options that allow the FMUExport part to use wireless connections, which may be useful to keep the GT-ISE map organized.

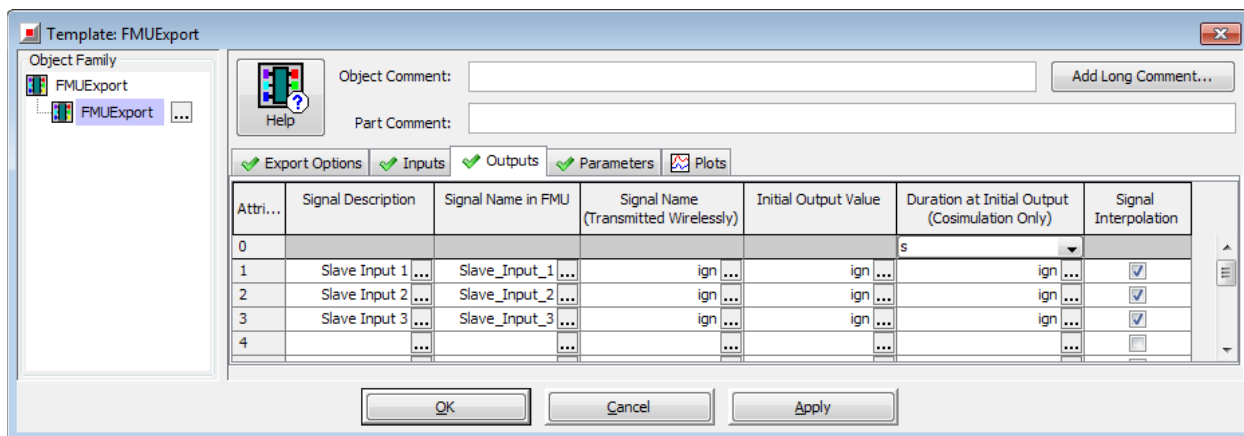


The Outputs folder is very similar to the Inputs folder. The "Outputs" folder defines outputs from the 'FMUExport' part, which means that the signals defined in the "Outputs" folder will be FMU Inputs. The "Signal Description" and "Signal Name in FMU" will define the "name" and "description" attributes in the imbedded XML file in the FMU, respectively.



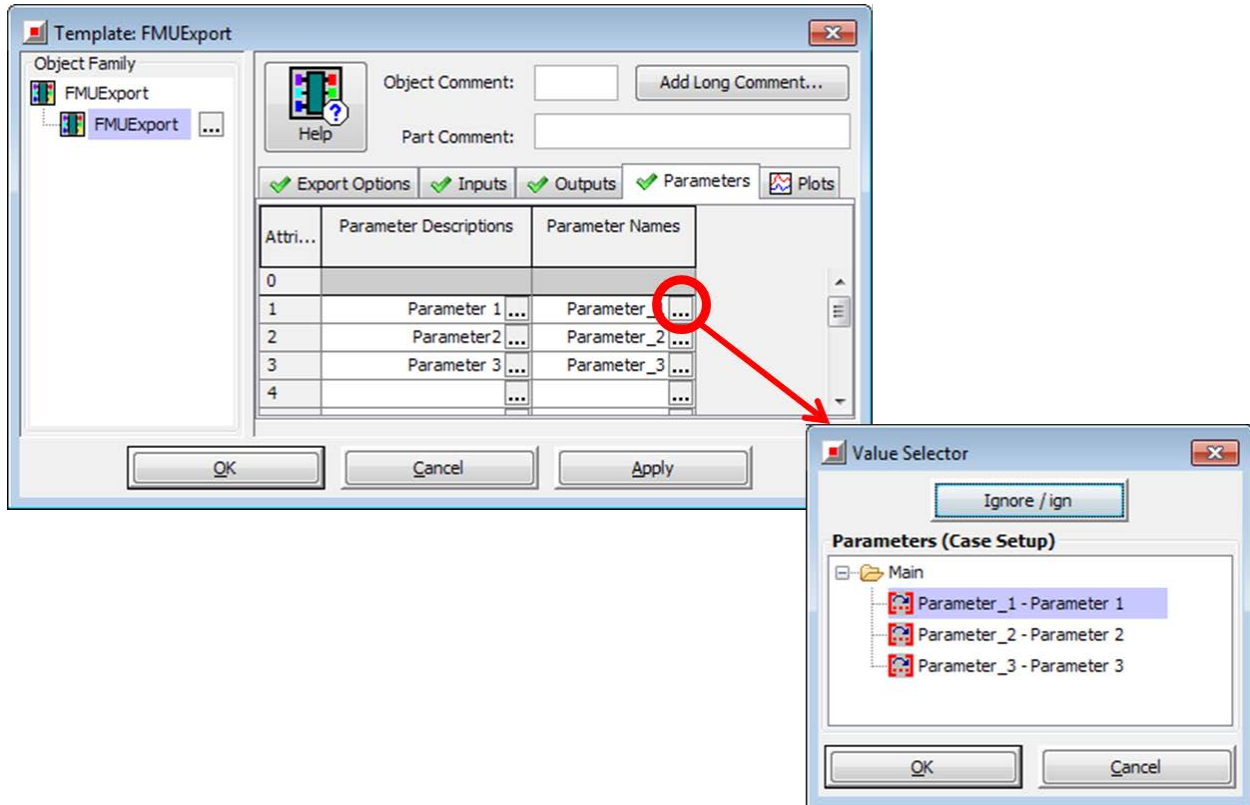
The "Signal Name (Transmitted Wirelessly)" attribute allows for wireless connections inside GT-ISE, which may be useful to keep the GT-ISE map organized.

The next attributes, initial output value and duration at initial output, are used to define the initialization states for each output signal for the co-simulation event. The signal interpolation check box indicates whether the output signal is interpolated. If unchecked, the signals coming into the FMU remain constant until they get updated at the next communication interval. This may cause small "stair-step" discontinuities and therefore, it is recommended that this check box is checked for most use cases.



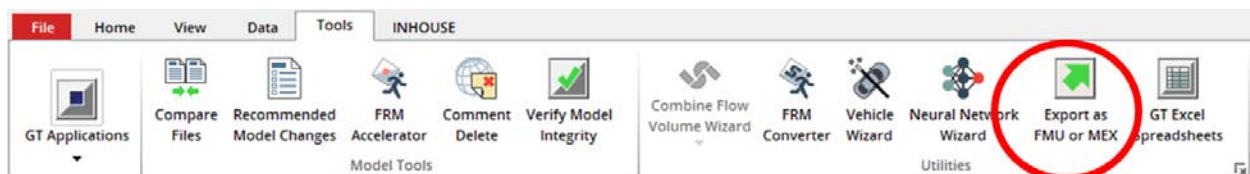
The "Parameters" folder defines the FMU parameters. The "Parameter Descriptions" can be any string; however, the "Parameter Names" must point to a GT-ISE parameter (an attribute that has been promoted to *Case Setup*). To select a Parameter to be used as the value for the "Parameter Names" attribute, click on the value selector (the ... button) and select a parameter, as shown in the figure below.





3.1.2 Exporting FMU

After a 'FMUExport' object has been created, a derived part must be placed on the map and connected using standard GT-ISE techniques. After the part has been connected and the model is error-free, GT-SUITE can export the model as a FMU. To do this, navigate to the "Tools" tab in the toolbar and find the "Export Model as FMU or MEX" button. Please see the figure below.



If the model is error-free and the 'FMUExport' part was correctly made and connected, the SolverUI will start. When the FMU is finished building, the FMU will be placed inside the model directory and is ready to be used by any other FMI standard master software.

3.1.3 Checking Model Integrity as a Standalone Model

'FMUExport' is a tool that can be used to create FMUs with GT-ISE when the "Export Model as FMU or MEX" button is chosen. However, the usage of 'FMUExport' changes when a simulation is run by conventional GT-ISE methods (by going to Run → Start Simulation or clicking the green triangle in the toolbar). If a simulation is run with a 'FMUExport' part, the 'FMUExport' part acts as a combination of numerous 'SignalGenerators' and 'SignalTerminators.' The virtual 'SignalGenerators' are used to output



the "Initial Output Value" of each of the FMUExport's output signals, and the virtual 'SignalTerminators' are used to terminate each of the FMUExport's input signals.

To check that the FMU will run without any failures or integrity errors, it is a good practice to set up each of the FMUExport's output signals with Initial Output Values and test to make sure that the model runs without failures.

3.1.1 Troubleshooting GT-Made FMUs

If a GT-Made FMU is created and used in a 3rd party software tool, there is always the possibility that the GT solver may fail. Most likely, the 3rd party software master tool will not output the *FAIL message for the user to read. This *FAIL message, however, will be output from the GT solver to the .out file. To read the .out file from a GT-made FMU, follow these directions:

On Windows machines:

- 1) Open a new Windows Explorer window and type "%TEMP%" into the directory path field. This will open a temporary directory that many programs use to write temporary files on windows machines.
- 2) Navigate to the folder named "fmu" and open the folder with the same name as the GT-made FMU.
- 3) Navigate to the "resources" directory and open the file with the ".out" file extension. If the GT-made FMU failed, the *FAIL message will be stored in this file.

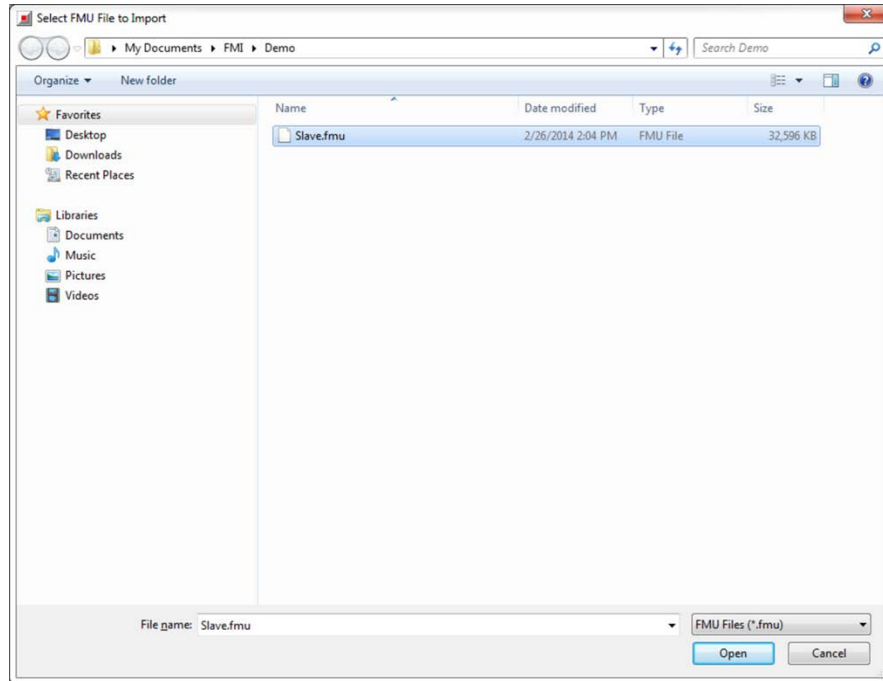
3.2 Importing an FMU (GT-SUITE as Master)

The GT-ISE template named 'FMUImport' allows users to import and call FMUs created by various modeling environments.

3.2.1 Creating a 'FMUImport' Object

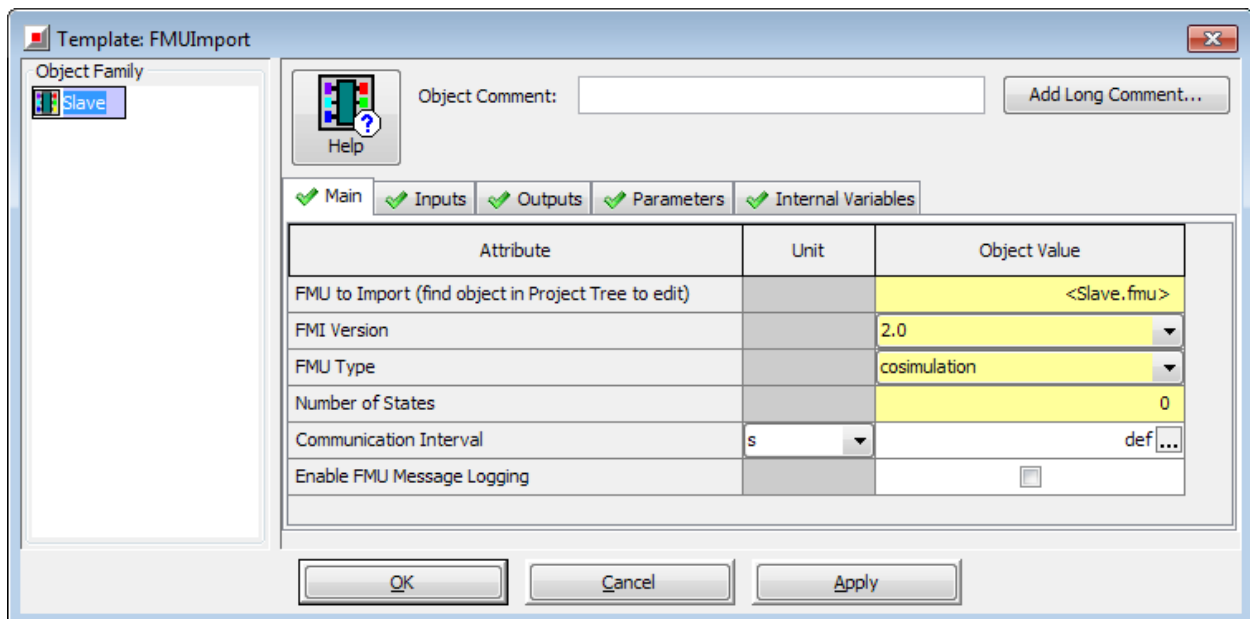
To create a new 'FMUImport' object, find the 'FMUImport' template in the project library (it may need to be dragged into the project library from the template library), then double-click on it to create a new object. After double-clicking on the template to create a new object, instead of the regular "Create Object" window, the user will be prompted with a dialog box that asks to select a FMU File to Import. This will look similar to the File → Open dialog box. See the figure below for an example:





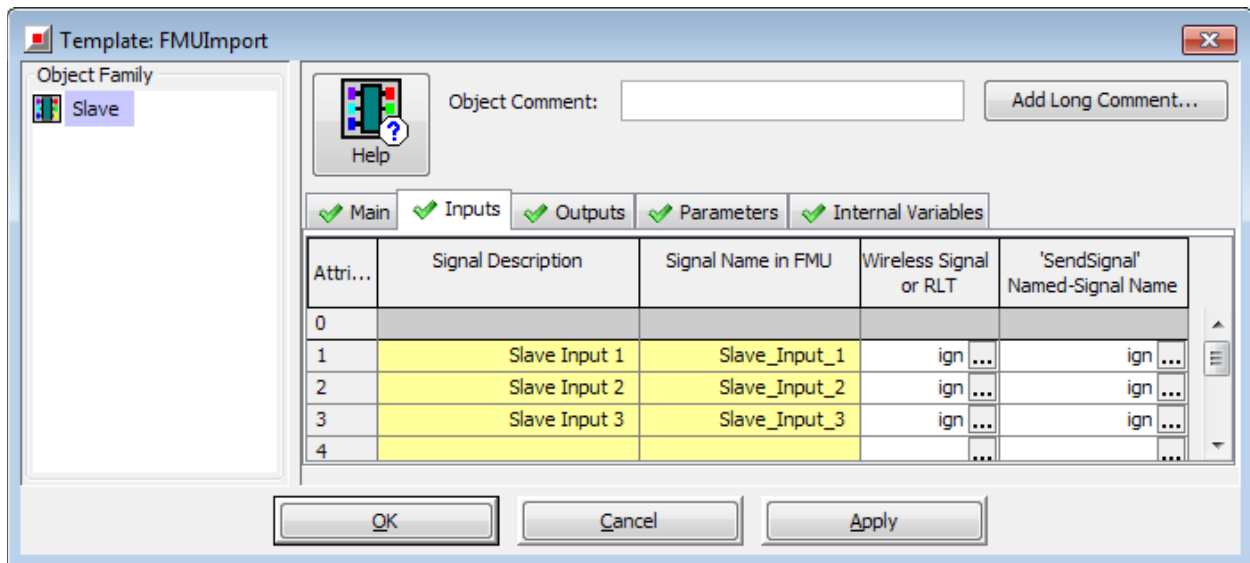
After a FMU is chosen, an object will be created. There are many attributes that will be shaded yellow. This means that this attribute is not editable. The input, output, and parameter descriptions and names will be automatically imported into the 'FMUImport' object. GT-ISE is able to parse this information from the imbedded XML document in the FMU. GT-ISE automatically completes these attribute values and does not allow them to be edited because if the names do not match the FMU exactly, the model will not run as desired.

In the Main folder, the location and name of the FMU that will be used is defined. To change the location or name of the FMU that it points to, please see the section titled "Updating Imported FMU."



The "Inputs" folder will define the inputs into the FMU. In the Inputs folder, the input signals descriptions and names can be viewed (but not edited).

The "Signal Name (Transmitted Wirelessly)" attribute allows for wireless connections inside GT-ISE, which may be useful to keep the GT-ISE map organized.

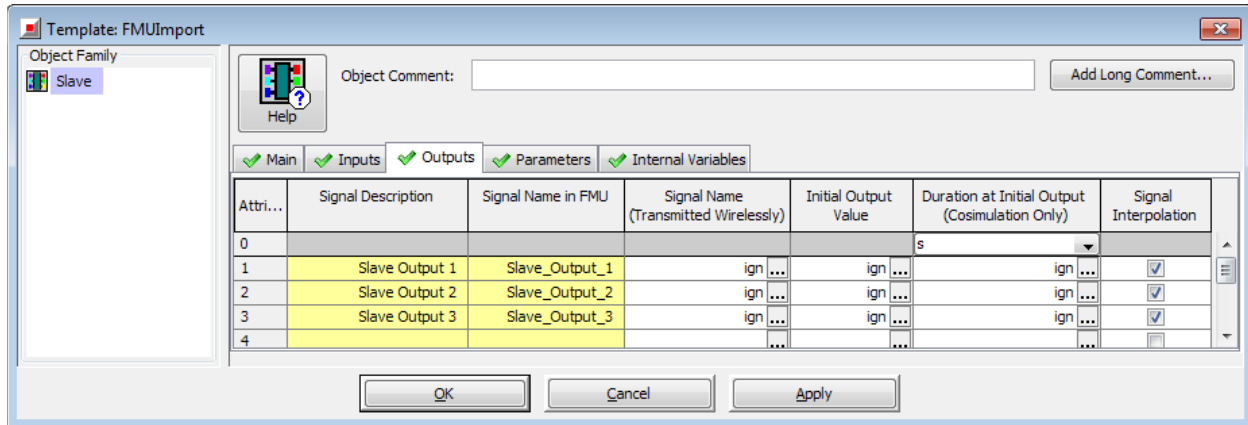


The "Outputs" folder will define the output coming from the FMU. In the Outputs folder, the output signals descriptions and names can be viewed (but not edited).

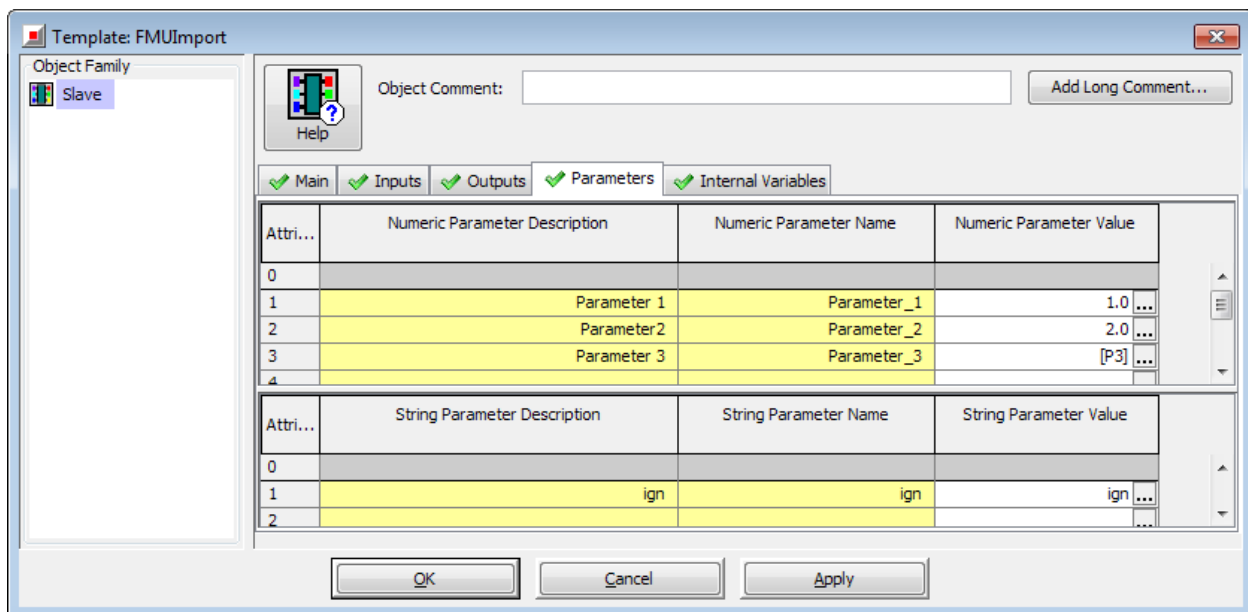
The "Signal Name (Transmitted Wirelessly)" attribute allows for wireless connections inside GT-ISE, which may be useful to keep the GT-ISE map organized.

The next attributes, initial output value and duration at initial output, are used to define the initialization states for each output signal for the co-simulation event. The signal interpolation check box indicates whether the output signal is interpolated. If unchecked, the signals coming into the FMU remain constant until they get updated at the next communication interval. This may cause small "stair-step" discontinuities and therefore, it is recommended that this check box is checked for most use cases.





The parameters folder allows the user to view (but not edit) the FMU's parameter descriptions and names. The "Parameter Values" attribute allows the user to define, change, or edit the value of the FMU's parameters (these values can even be promoted to Case Setup, see Parameter 3 in the figure below).



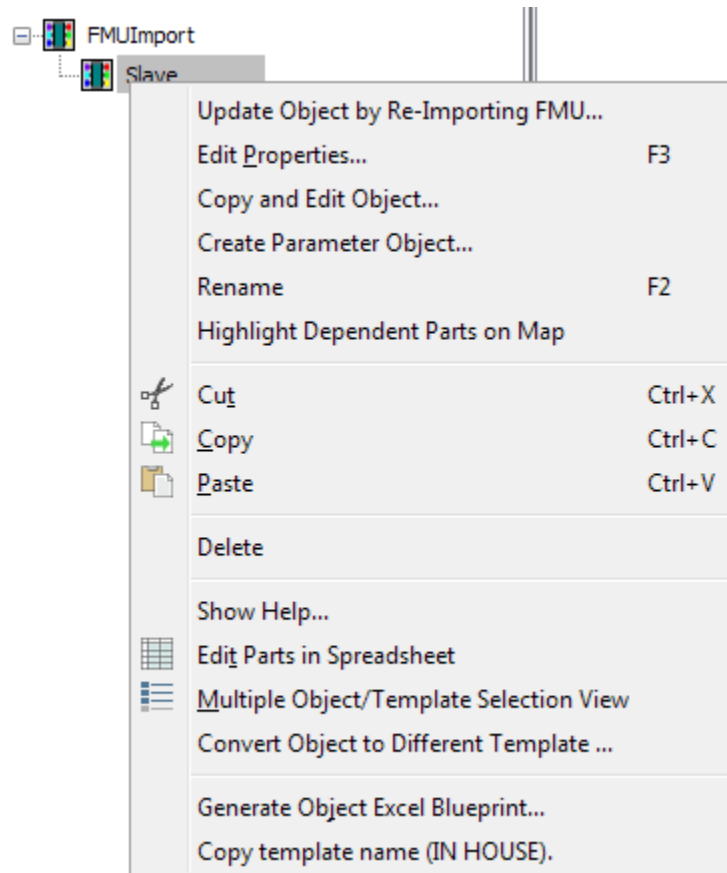
After a 'FMUImport' object has been created, a derived part can be placed on the map and connected using standard GT-ISE techniques. To run the model, go to Run → Start Simulation or click the green triangle in the toolbar.

3.2.2 Updating Imported FMU

When GT-ISE imports a FMU, it uses the imbedded XML document to define the various signal descriptions and names. This, however, is not dynamically changed when the FMU is changed. For instance, if a 'FMUImport' object is created by importing a FMU, then an input signal is added to the FMU, the 'FMUImport' will not include the input signal.



In order to update the imported FMU, find the FMUImport object in the project tree, right click on it, and select "Update Object by Re-Importing FMU..." as shown in the figure below.



After this is done, the input, output, and parameter names and descriptions will be updated with the changes that have occurred to the FMU.



CHAPTER 4: Integrated Modeling with ASCET

GT-SUITE also supports integration with ASCET, created by ETAS.

Notes:

1. ASCET / GT-SUITE co-simulations are currently supported for Win32 platforms only.
2. ASCET / GT-SUITE co-simulations are officially supported for ASCET v6.0 or later.

Instructions:

1. Build the complete system model in GT-SUITE.
2. Include a 'SimulinkHarness' part in the model.
3. Link the 'SimulinkHarness' part to other parts in the GT-SUITE model. The 'SimulinkHarness' input and output signal numbers correspond to the output and input signal numbers (respectively) of the GT-SUITE S-Function Block in the Simulink model (please see 'SimulinkHarness' in the reference manual, or in the online help, for more information). There are three ways that the links may be made:
 - a. "Wireless" links via the attributes in the 'SimulinkHarness'.
 - b. Wireless via named signals, using 'SendSignal' and 'ReceivedSignal' parts.
 - c. Direct links. When connecting to non-control parts, the 'SensorConn' and 'ActuatorConn' part are used. When linking to other control parts, the link can be directly made without any connection.
4. In the 'SimulinkHarness' part, ensure that the **Simulation Type** attribute in the **Main** folder to "run_from_simulink" and the **Simulink Model to Import (.dll/.so)** is set to "ign".
5. Ensure the Optimization Type in the Optimization folder of the Optimizer (available from the Optimization Menu) is set to "off". Running a coupled simulation from ASCET does not support using the direct optimizer.
6. Generate a .dat file (Ctrl-D) and place this file in the ASCET's working directory. For ASCET v6.0 this directory is by default the %ETASHOME%/ASCET6.0. Please note that if your model has an encrypted external subassembly, the *.sim file would have to be copied into the same directory.
7. Verify that you have a folder called: %GTIHOME%\v2016\Gtsuite\bin\win32\coupling
8. Add this folder to the PATH environment variable.
9. Copy the file %GTIHOME%\v2016\ascet\GTSUITEv2016.exp to a directory where you can edit it. It is recommended that you avoid editing directly the file in the GT installation directory.
10. Open ASCET, open an existing database (or a new one), and create your ASCET model. Click on File->Import... and choose the file GTSUITEv2016.exp.
11. The folder with the GTSUITEv2016 interface should now be in your model.
12. Open the GTSUITEv2016 block and modify the GT Editable Parameters in the Header as described below.
13. Compile the model and run the simulation. A command window indicating that GT-SUITE is running should open immediately. This window displays important information on the progress of the simulation.
14. If the GT-SUITE simulation stops due to an error, a message will be displayed in the ASCET MONITOR and the coupled simulation will stop.



Information:

GTSUITEv2016 is a "Continuous Time Block" as defined in the ASCET manual. It has been created using C. It enables a combined ASCET - GT-SUITE simulation. The model has an input array called "inputs" and an output array called "outputs". Each element of the input and output array corresponds to the output and input port number of the 'SimulinkHarness' part in the GT-SUITE model respectively. In order to run the model the user has to edit the following part of C code located inside the Header of the CT Block.

```

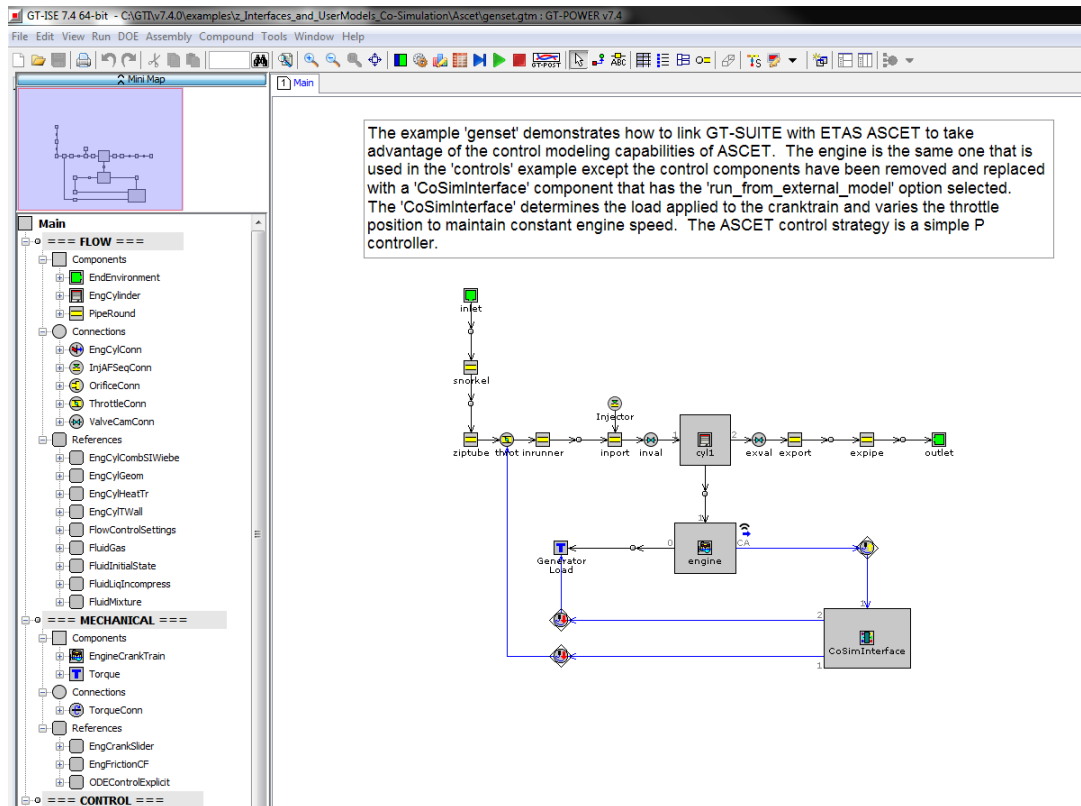
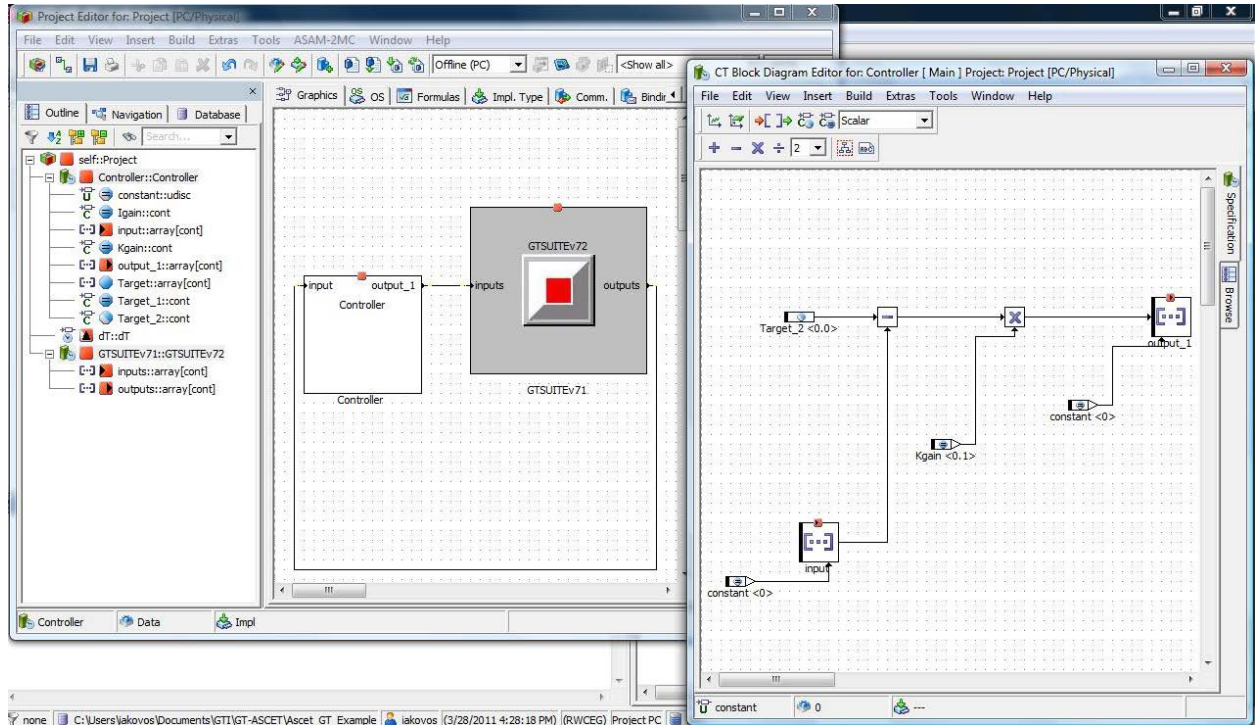
/*****GT EDITABLE PARAMETERS *****/
char *gti_fname = "genset";          /*model name*/
int   gti_client = 1;                /*Reserved for future use*/
int   gti_icase  = -1;               /*Case No to run*/
int   gti_ni     = 2;                /*Number of Inputs*/
int   gti_no     = 2;                /*Number of Outputs*/
/*****END GT EDITABLE PARAMETERS *****/

```

- **GT-SUITE data file (gti_fname):** The name of the GT-SUITE data file to be used in the simulation. Do not include the ".DAT" extension and do not enclose the filename in quotes. Note that the data file must have been generated from a GT-SUITE v2016 model.
- **Simulation Duration Source (gti_client):** This variable is reserved for future use.
- **Case No (gti_icase):** This is the case number in CaseSetup that is to be used for the simulation. If this value is set to -1, the first CHECKED case in the GT-SUITE model will be used for the simulation.
- **Number of inputs (gti_ni):** Number of input signals into the GT-SUITE model (maximum 128).
- **Number of outputs (gti_no):** Number of output signals from the GT- SUITE model (maximum 384).

The figures below show how a example model in ASCET and GT-SUITE looks like. In this example, a 1-cylinder engine model in GT-SUITE is controlled (through the throttle angle) by a controller in ASCET so that a desired brake torque is achieved.





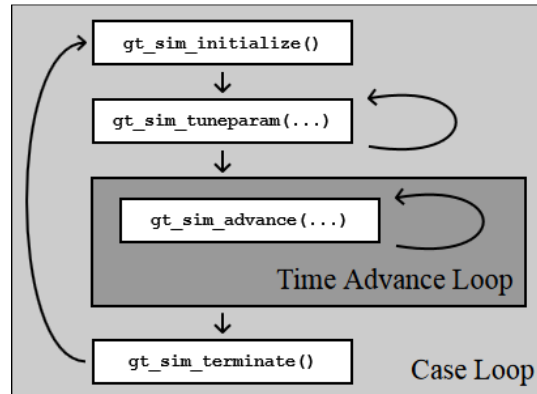
CHAPTER 5: General Co-Simulation with GT-Master API

GT-SUITE simulations may be run in conjunction with other modeling applications or with hand-written C-code to simulate complex systems. GT-Master API is a general interface that allows GT-SUITE to co-simulate with any program that is able to comply with the specification described below.

GT-SUITE solver controls the simulation and acts as a "master". The external model, which is compiled as a dynamic library (.dll or .so), is called upon to perform its tasks by GT-SUITE and accepts a "slave" role. GT-SUITE provides the external model with input and then calls external model's functions whose task is to update outputs. One example of such a model is custom user code written in C language that is compiled as a loadable library.

5.1 General Co-Simulation Principle

Following diagram shows the execution order and execution loops of external model calls in the GT-Master API.



Following paragraphs describe the co-simulation process as performed by GT-SUITE in more detail.

1. GT-SUITE dynamically loads the library from the file that is specified in the 'External Model to Import' attribute of the CoSimInterface object. The 'Simulation Type' drop-down attribute must be set to 'import_external_model'.
2. At the beginning of simulation GT-SUITE loads all the functions that are exported in the library. Namely `gt_sim_initialize()`, `gt_sim_tuneparam()`, `gt_sim_advance()` and `gt_sim_terminate()`.
3. GT-SUITE asks the external model to initialize by calling `gt_sim_initialize()`.
4. GT-SUITE optionally overrides the parameters of the external model through `gt_sim_tuneparam()` function. The override function is called once for every parameter defined in 'External Model Parameters' attribute located on 'Advanced' folder of CoSimInterface template.
5. GT-SUITE repetitively executes the code of `gt_sim_advance()` and advances time by a specific amount. This amount of time is defined by the external model itself and is passed back to GT-SUITE. This way GT-SUITE solver ensures that simulation remains stable and executes the code after its internal simulation time has caught up with the external model simulation time.
6. At the end of simulation GT-SUITE asks the model to perform termination operations by calling `gt_sim_terminate()`.



5.2 GT-Master API Functions

Following section summarizes all functions available in GT-Master API. To run the simulation all of the functions must be compiled in the library binary file. At run time the GT-SUITE solver tries to load all functions from the library and if the loading fails the solver issues an error message and stops.

Note that due to the risk of sharing global and static variables it isn't recommended to have two or more CoSimInterface parts loading the same external model library file. If two parts should have the same functionality the respective library files need to be copied to two different locations. Each of the objects must reference its unique dynamic library file.

```
int gt_sim_initialize(void)
```

Perform initialization operations on model. GT-SUITE calls this function during its own initialization and allows the external model to initialize as well. Typical initialization operations are memory allocation or opening of output/input files.

The function doesn't accept any arguments.

```
int gt_sim_tuneparam (char *param_name, double *param_value, size_t param_name_length)
```

Set external model parameters based on values specified in GT-SUITE. GT-SUITE calls this function immediately after `gt_sim_initialize()` and the call is made once for every 'External Model Parameter' defined in CoSimInterface 'Advanced' folder.

Function arguments list:

- `param_name` Name of the parameter that is being processed.
- `param_value` Value of the processed parameter.
- `param_name_length` Length of `param_name` string.

```
int gt_sim_advance (const double *gt2sl, int gt2sl_count, double *sl2gt, int sl2gt_count, double *sl_tstep)
```

Advance the simulation time-step. Size of `gt2sl_count` and `sl2_count` are derived from the number of inputs and outputs of the corresponding CoSimInterface part. The desired time-step size of the external model `sl_tstep` is used so GT-SUITE solver can call the function again, when it has caught up with the external timestep size. Based on the value of "Exact Synchronization with External Tool" check box, when necessary, GT-SUITE will take smaller time-steps than it's calculated ones in order to synchronize with the timestep of the external tool. In a case where GT-SUITE timestep is larger than the external tool's timestep, GT-SUITE will be forced to use the timestep of the external model independently of the check box value.

Function argument list:

- `gt2sl` Pointer to array of CoSimInterface input values going in the direction from GT-SUITE to the external model.
- `gt2sl_count` Number of `gt2sl` array members.
- `sl2gt` Pointer to array of CoSimInterface output values going in the direction from the external model to GT-SUITE.
- `sl2gt_count` Number of `sl2gt` array members.
- `sl_tstep` External model time-step size.

```
int gt_sim_terminate (void)
```



Perform termination operations on model. GT-SUITE calls this function during its own shut-down and allows the external model to terminate as well. Typical termination operations are memory de-allocation or closing of output/input files.

Function doesn't accept any arguments.

5.3 Using GT-Master API with General Model Written in C

The following section assumes that the user has basic knowledge of the GT-SUITE and has a good understanding of the coupling principle as it is described in earlier sections of this manual. Beside that it is assumed that the user has a good knowledge of C language and knows how to compile and link C language programs and libraries.

Good starting point worth looking at is a very simple proportional controller example model located in \$GTIHOME/v\$GTIVERS/examples/z_Interfaces_and_UserModels_Co-Simulation/GTI_API/genset_pcontroller.gtm. The model contains the libraries compiled for all architectures where GT-SUITE is supported as well as the source code.

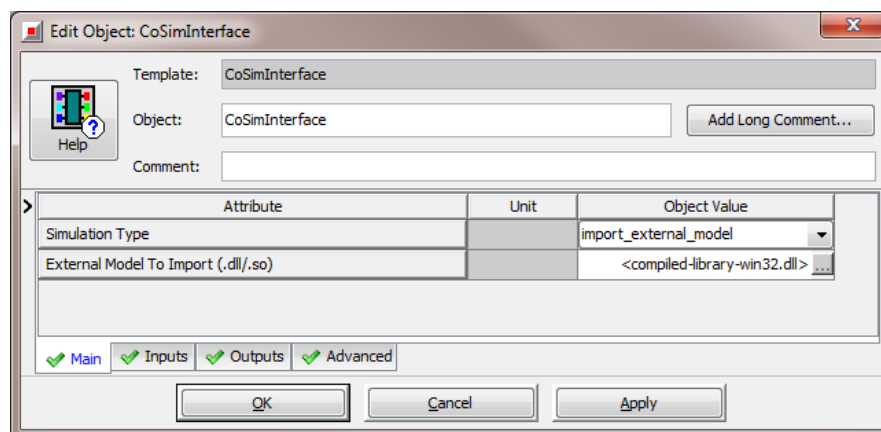
5.3.1 Prerequisites

- Operating system - Windows or Linux 32/64 bit.
- GT-SUITE - 7.4 or later.
- C language compiler
 - Microsoft Visual Studio 2010 or later (Express edition is free of charge.)
 - Microsoft SDK 7 (Later versions of the SDK doesn't contain compiler.)
 - GCC 4.1 or later (Free software)

5.3.2 General Model Co-Simulation Setup Instructions

Should the user workstation meet all the above prerequisites, it is possible to load and run external co-simulation model in GT-SUITE. Following steps are necessary to achieve that:

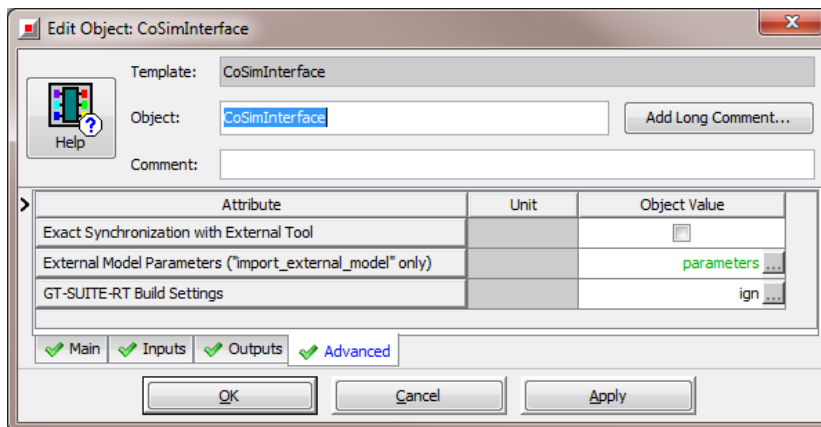
1. Build the model in GT-SUITE and add a CoSimInterface template to the map.
2. Setup the CoSimInterface object 'Main' folder according to the picture below.



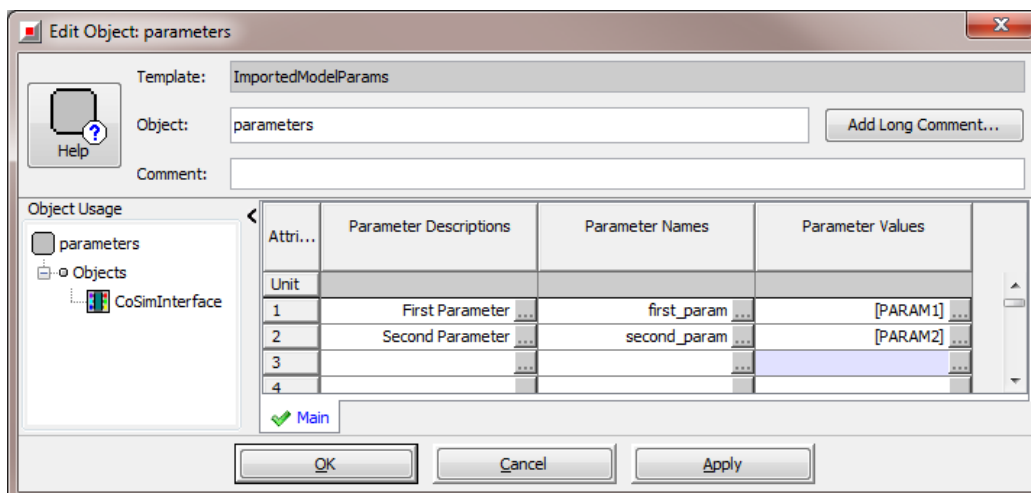
- Select 'import_external_model' in 'Simulation Type' drop-down attribute



- Specify the compiled external model library file in the 'External Model to Import' attribute. The compiled library must export all the functions defined in section [GT-Master API Functions](#).
 - Please note, that the solver architecture must be the same as the library architecture. For example 32-bit solver can only load 32-bit library. Default solver architecture can be specified under Tools→Options→Run.
3. Connect and configure input and output ports under 'Inputs' and 'Outputs' folder. Note that, the general co-simulation code compiled in the external library may run only when certain number of input and output ports is connected.
 4. Under the 'Advanced' folder set the attribute values according to the picture below.



- If the 'Exact Synchronization with External Tool' flag is unchecked, GT-SUITE will speed up the simulation at the expense of accuracy. However in a case where GT-SUITE time step is larger than the external model's time step, GT-SUITE will be forced to use the time step independently of the checkbox value.
 - Set GT-SUITE-RT Build settings to 'ign' since this attribute doesn't have any effect on general external model co-simulation.
5. Optionally define 'External Model Parameters' reference object in a similar fashion as is shown on the image below.



- Set the parameters names and values that are used in the external model. Note that the external model library may require certain parameters to be present and to have correct value for the model to work correctly.
 - At model runtime, function `gt_sim_tuneparam()` is called once for every parameter defined here.
6. Run the simulation.



INDEX**A**

ASCET, 47

C

Common Errors, 36

controls modeling

 with integrated Simulink, 11

G

GTsuiteRT, 29

GTsuiteRT, 27

H

Hardware In the Loop (HIL), 29

R

real-time simulation, 27, 29

S

SiL, 29

simulation dies immediately w/ out file, 36

simulation dies immediately w/o out file, 37

Simulink, 11

specified module could not be found, 36

Standalone Executable, 33

T

troubleshoot unexpected results, 37

