Simulink Module Tool

April 2019

Contents

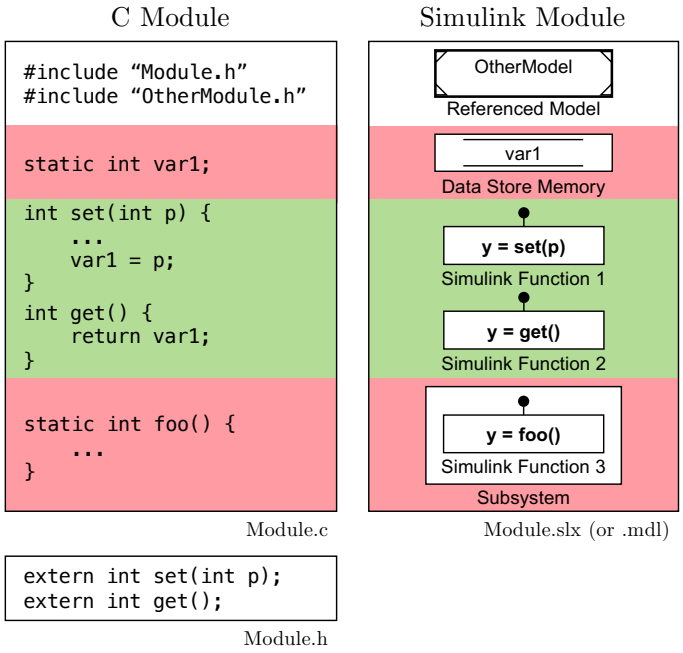


Figure 1: Simulink module structure based on C.

1 Introduction

This tool performs several functions in order to support modular development for Simulink models. A Simulink module structure is described in [1] and can be summarized by Figure 1. The following sections explain each of the tool functions.

1.1 Simulink Functions

In Simulink, a function can be defined via the Simulink Function block, and invoked graphically via Function Caller blocks. In the modular programming approach to software development, a system is decomposed into multiple modules. Each module must be able to hide information in its implementation, and only expose on the interface information what is to be shared with other modules. If a Simulink model is analogous to a module, one can use Simulink Function blocks to hide information in the implementation or expose information to the interface. However, a thorough understanding of the Simulink Function block is necessary in order to know how the *visibility parameter* and location in the model impacts its accessibility and name qualification.

The Simulink Module Tool assists developers with using Simulink Function blocks in implementing Simulink modules. The tool can automatically convert between the different scopes of Simulink Functions, thereby changing whether it

is exposed on the interface or local to the model. The tool also assists users in quickly calling Simulink Functions, with their appropriate qualifiers, from any location in a model or parent hierarchy, as well as creating Function Caller blocks with their *Function prototype*, *Input argument specifications*, and *Output argument specifications* parameters automatically populated (if possible).

1.2 Interface

A Simulink model's interface is commonly considered to be comprised of the Inports and Outports of the top-level system. An interface should make clear *all* the communication and dependencies of a module, and unfortunately Inports and Outports are not a complete representation of the possible data flow into and out of a model. The From File, From Spreadsheet, From Workspace blocks, and global Data Stores that are read are also inputs to the model. Elements that are outputs of the model can also include To Workspace, To File, and global Data Stores that are written to. Additionally, Simulink Function definitions can be exported from a model. A complete definition of a Simulink module interface is presented in [1].

The Simulink Module Tool automatically extracts the interface of a Simulink module and either represent it in the the root of the Simulink module, or prints the interface to the Command Window.

1.3 Dependencies

A Simulink model can have many dependencies in the form of Model References, Library linked blocks, and data dictionaries. These reside outside of the Simulink module, but are necessary in order to be able to simulate and code generate the model.

The Simulink Module Tool examines the model, determines which dependencies exist, and prints a list to the Command Window.

1.4 Guidelines

Several guidelines relating to Simulink Functions and interfaces are presented in [1]. Tables 1, 2, 3, and 4 outline the guidelines, and the tool automates the checking of these guidelines.

2 How to Use the Tool

This section describes what must be done to setup the tool, as well as how to use the tool.

2.1 Prerequisites

- For full use of the tool, please use MATLAB/Simulink R2017b or newer[1].
- To install the tool from a `.zip` file, ensure the unzipped contents are present in your MATLAB path. If you do not see the tool in the Context Menu, run sl_refresh_customizations. To install the tool from a `.mltbx` file, simply open MATLAB and double-click on the file. Your path should be automatically configured.
- Before using the tool, ensure your model is open and unlocked.

2.2 Getting Started

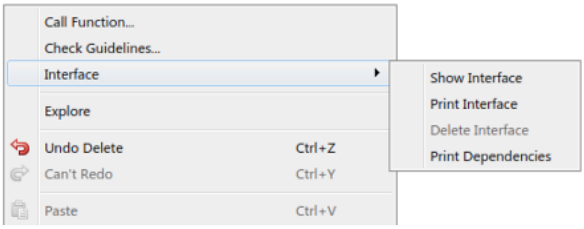
The tool can be used via the Simulink Context Menu, which can be viewed by right-clicking in a model. The following options can be available, depending on what is selected in the model, and which version of MATLAB/Simulink is used. Options available when no blocks are selected are (Figure 2a):

- *Call Function*
- *Check Guidelines*
- *Show Interface*
- *Print Interface*
- *Delete Interface*
- *Print Dependencies*

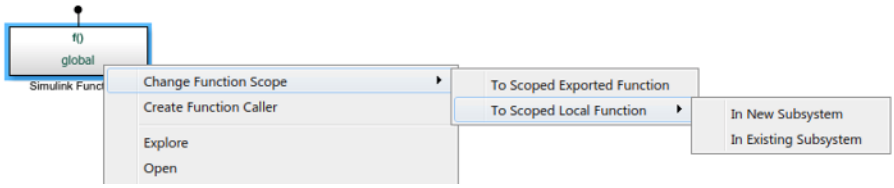
Options available when one or more Simulink Function blocks are selected are (Figure 2b):

- *Change Function Scope*
- *Create Function Caller*

[1] Simulink functions were introduced in R2014b. The *function visibility* parameter was introduced in R2017b.



(a) Menu options when no functions are selected.



(b) Menu options when one or more functions are selected.

Figure 2: Simulink Context Menu with context-dependant tool options visible.

2.3 Functionality

This section describes the tool functionality when it is used from the Simulink Context Menu (Figure 2).

2.3.1 Call Function

Note: This option is not available in versions prior to R2014b.

Right-clicking anywhere in the model and then selecting `Call Function` from the Context Menu will display the user interface shown in Figure 3. The listbox shows all Simulink Functions that can be called from that location in the model. Select a function, press OK, and a Function Caller will be created with its `Prototype`, `Input argument specifications`, and `Output argument specifications` parameters automatically populated.

2.3.2 Change Function Scope

Note: This option is not available in versions prior to R2017b.

Right-clicking directly on one or more Simulink Function blocks and then selecting `Change Function Scope` from the Context Menu, will display options for changing the scope of the functions. An example is shown in Figure 4.

The scope of a Simulink function, i.e., where it can be used, is determined by both its hierarchal *placement* in the model in which it is defined, and its *function visibility* parameter. The rules for determining the scope of a Simulink Function are not straightforward. When a Simulink Function is given global function visibility, it can be placed anywhere in a model, and will be available

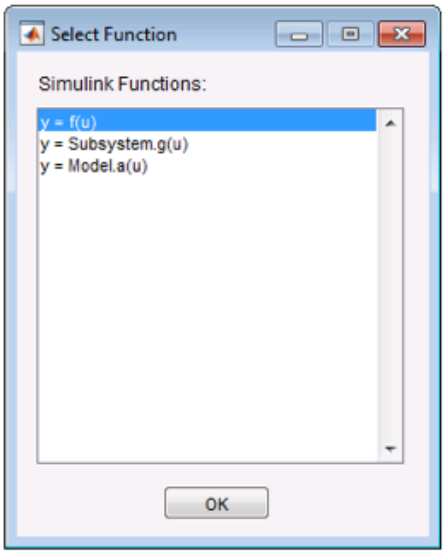


Figure 3: Select Function interface.

for use anywhere in the model hierarchy (i.e., in any subsystems or in the parent model hierarchy). When a Simulink Function is given scoped function visibility, its placement in the model affects its accessibility. If placed at root level, it is accessible in the model hierarchy. The difference between a global Simulink Function and a scoped Simulink Function placed at the root is in the way it is called. In the latter, the function name must be qualified with the model reference block name. If the scoped Simulink Function is placed in a subsystem S (i.e., not at the root), it is no longer available outside of the model. Instead, it is available in its parent subsystem and any descendants.

2.3.3 Create Function Caller

Note: This option is not available in versions prior to R2014b.

Right-clicking on one or more Simulink Function blocks and then selecting `Create Function Caller` from the Context Menu, will automatically create corresponding Function Caller blocks for the selected functions. The `Prototype`, `Input argument specifications`, and `Output argument specifications` parameters of the new Function Caller are automatically populated. An example is shown in Figure 5a.

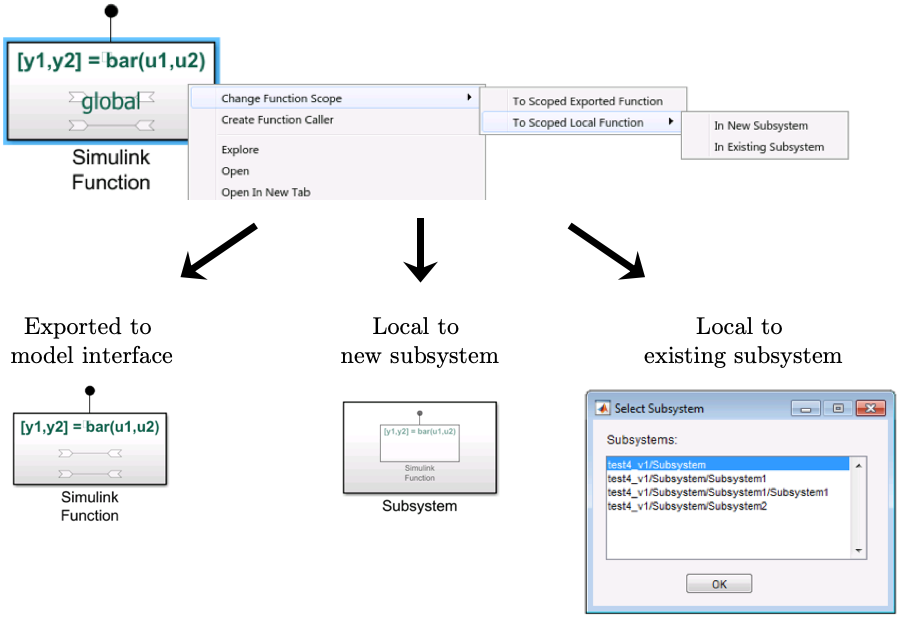


Figure 4: Changing Simulink Function scopes.

2.3.4 Check Guidelines

Right-clicking anywhere in the model and then selecting `Check Guidelines` from the Context Menu will display the user interface shown in Figure 6. The window shows all the guidelines that can be enabled by the user, and hovering over their names with the cursor will display a brief description of the guideline. The guidelines are detailed in Tables 1, 2, 3, and 4. Select one or more guidelines to check, press OK, and the Command Window will display any violations.

2.3.5 Show Interface and Print Interface

Note: This option is not available in versions prior to R2016a.

Right-clicking anywhere in the model and then hovering over `Show Interface` or `Print Interface` from the Context Menu will give the user options to insert the interface representation in the model (e.g., Figure 7a), or print the interface description to the Command Window (e.g., Figure 7b).

Warnings

1. This feature is only available for versions R2017b and newer. The Context Menu will have this option disabled for earlier versions. This is because the *function visibility* parameter (i.e., scoping) was not introduced for Simulink Functions until R2017b.

Table 1: Guideline on Simulink Function placement.

ID: Title	1: Simulink Function Placement
Priority	Strongly Recommended
MATLAB Version	R2014b and later
Description	Position the Simulink Function block in the lowest common parent of its corresponding Function Caller blocks. Do not position the Simulink Function in the top layer for no reason. Avoid placing Simulink Function blocks below their corresponding Function Caller blocks.
Rationale	☑ Readability ☑ Verification and Validation ☑ Workflow ☑ Code Generation ☐ Simulation

Table 2: Guideline on Simulink Function scope.

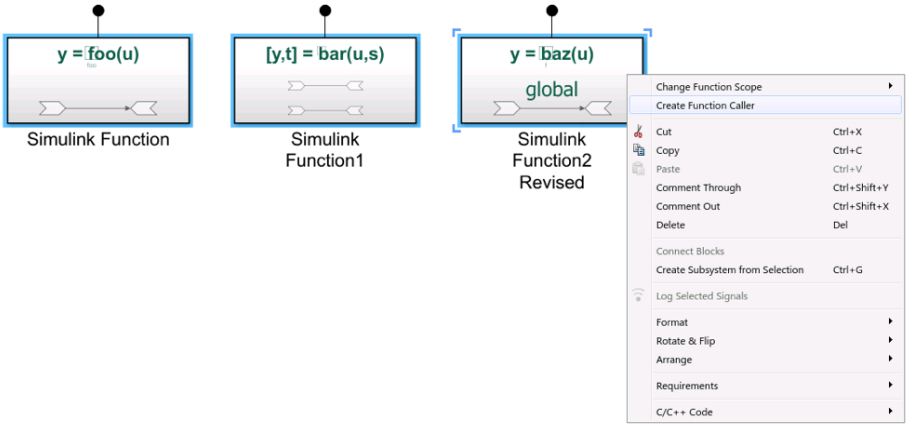
ID: Title	2: Simulink Function Scope
Priority	Strongly Recommended
MATLAB Version	R2017b and later
Description	Limit the Function Visibility parameter of the Simulink Function block's trigger port to scoped if possible. 
Rationale	☑ Readability ☑ Verification and Validation ☑ Workflow ☑ Code Generation ☐ Simulation

Table 3: Guideline on Simulink Function shadowing.

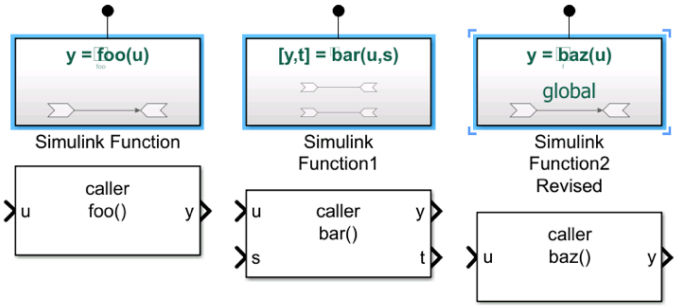
ID: Title	3: Simulink Function Shadowing
Priority	Strongly Recommended
MATLAB Version	R2014b and later
Description	Do not place Simulink Functions with the same name and input/output arguments within each other's scope.
Rationale	☑ Readability ☑ Verification and Validation ☑ Workflow ☐ Code Generation ☐ Simulation

Table 4: Guideline on using the base workspace.

ID: Title	4: Use of the Base Workspace
Priority	Recommended
MATLAB Version	R2006a and later
Description	Do not use the base workspace for storing, reading, or writing data that a model is dependant on. Instead, place such data in either the model workspace, if it is to be used in a single model, or a data dictionary if it is to be shared across models. This means avoiding the use of the sources, • From File • From Workspace • From Spreadsheet and sinks, • To File • To Workspace as well as avoiding defining signals, types, etc. in the base workspace.
Rationale	☑Readability ☑Verification and Validation ☑Workflow ☑Code Generation ☑Simulation



(a) Select Create Function Caller from the Context Menu.



(b) New function callers are added.

Figure 5: Example of Simulink Function Caller creation.

2. If a Simulink Function is moved and one or more Function Callers with the same prototype exist within its previous scope, a warning will appear in the Command Window. It is recommended that the user ensure all Function Callers still correctly call the Simulink Function. Automatic updating of Function Caller blocks is planned for a future version of this tool.

2.3.6 Delete Interface

Right-clicking anywhere in the model and then selecting `Delete Interface` will delete the interface that was created using the `Show Interface` option.

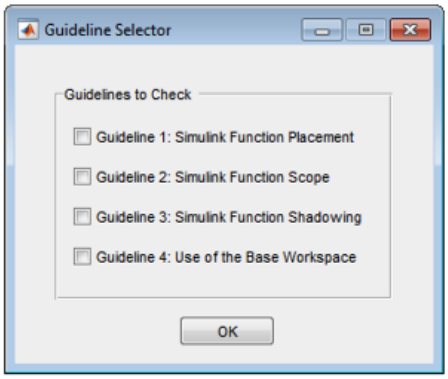


Figure 6: Select guidelines GUI.

2.3.7 Print Dependencies

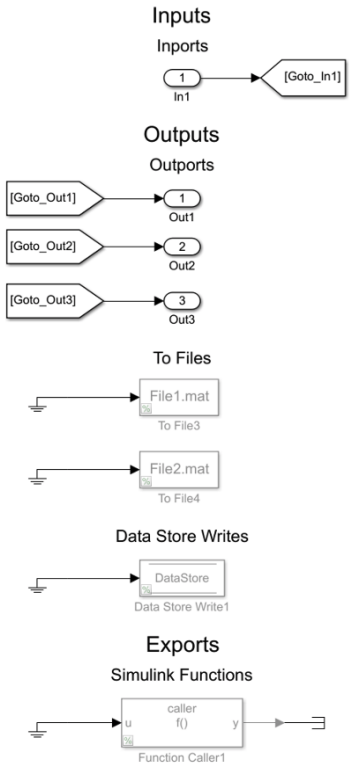
Right-clicking anywhere in the model and then selecting `Print Dependencies` from the Context Menus will print the dependencies that the model relies on in the Command Window. Dependencies include: Model Reference blocks, Library blocks, and data dictionaries. An example of the output is shown in Figure 8.

2.4 Errors and Warnings

Any errors or warnings during tool use will be visible in the MATLAB Command Window.

References

[1] Author One, Author Two, Author Three, and Author Four, "Supporting Modularity in Simulink Models", 2019, *(Manuscript submitted for publication to MODELS).*



(a) Visual representation.

```
Inputs
------
Inports:
  Demo/In1, uint16, 1, 1

Outputs
-------
Outports:
  Demo/Out1, Inherit: auto, 1, 1
  Demo/Out2, Inherit: auto, -1, 1
  Demo/Out3, Inherit: auto, -1, Inf
To Files:
  Demo/To File1, Timeseries, N/A, 1
  Demo/To File2, Timeseries, N/A, 1
Data Store Writes:
  Demo/Data Store Write, uint16, 1, 1

Exports
-------
Simulink Functions:
  Demo/Simulink Function,
    In:  uint16, 1, -1
    Out: uint16, 1, -1
```

(b) Textual description.

Figure 7: Simulink module interface.

```
Model References
------
Demo2/ControlModel
Demo2/Subsystem/EstimationModel

Library Links
------
Demo2/Subsystem2/CustomTable

Data Dictionaries
------
definitions.sldd
```

Figure 8: List of dependencies.