Simulink Code Inspector

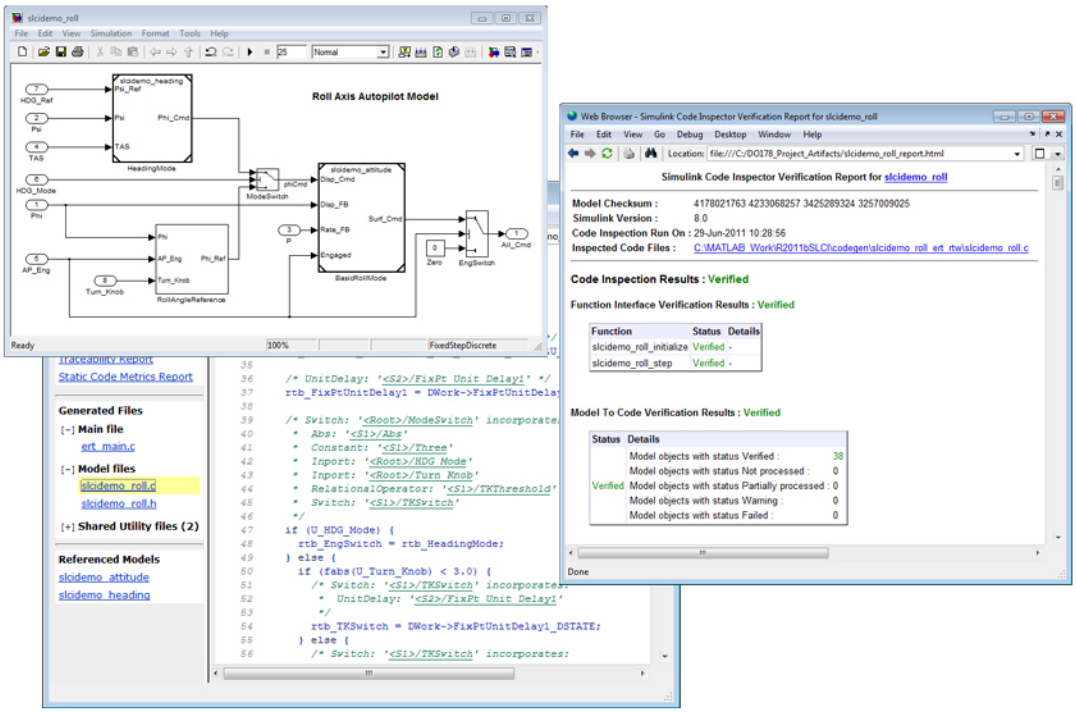
Automate source code reviews for safety standards

Overview

Simulink Code Inspector™ automatically compares generated code with its source model to satisfy code-review objectives in DO-178 and other high-integrity standards. The code inspector systematically examines blocks, parameters, and settings in a model to determine whether they are structurally equivalent to operations, operators, and data in the generated code. Simulink Code Inspector provides detailed model-to-code and code-to-model traceability analysis. It generates structural equivalence and traceability reports that you can submit to certification authorities to satisfy DO-178 software coding verification objectives.

Key Features

- Structural equivalence analysis and reports
- Bidirectional traceability analysis and reports
- Compatibility checker to restrict model, block, and coder usage to operations typically used in high-integrity applications
- Tool independence from Simulink® code generators

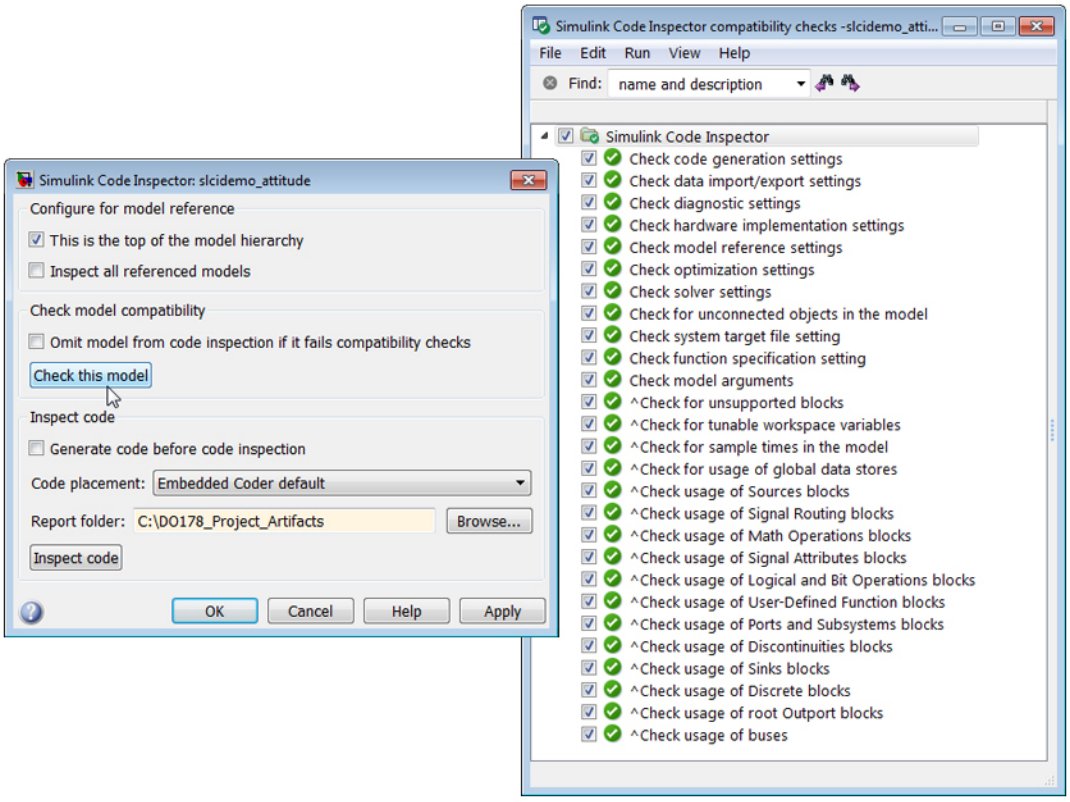


Model (left), generated code (bottom), and code inspection report (right) from Simulink Code Inspector.

Model Preparation

Simulink Code Inspector supports a constrained set of modeling semantics and code optimizations often used for high-integrity system models. A compatibility checker provided by Simulink Code Inspector determines whether your model complies with the constrained set. You can invoke compatibility checking interactively from the Simulink Code Inspector user interface or programmatically using MATLAB® commands. You can place blocks

and other portions of a model identified as incompatible within a referenced model, and then configure Simulink Code Inspector to omit the incompatible referenced model during code inspection. You can then perform a manual code inspection for models not automatically reviewed.



Simulink Code Inspector user interface and model compatibility check results.

Code Generation and Inspection

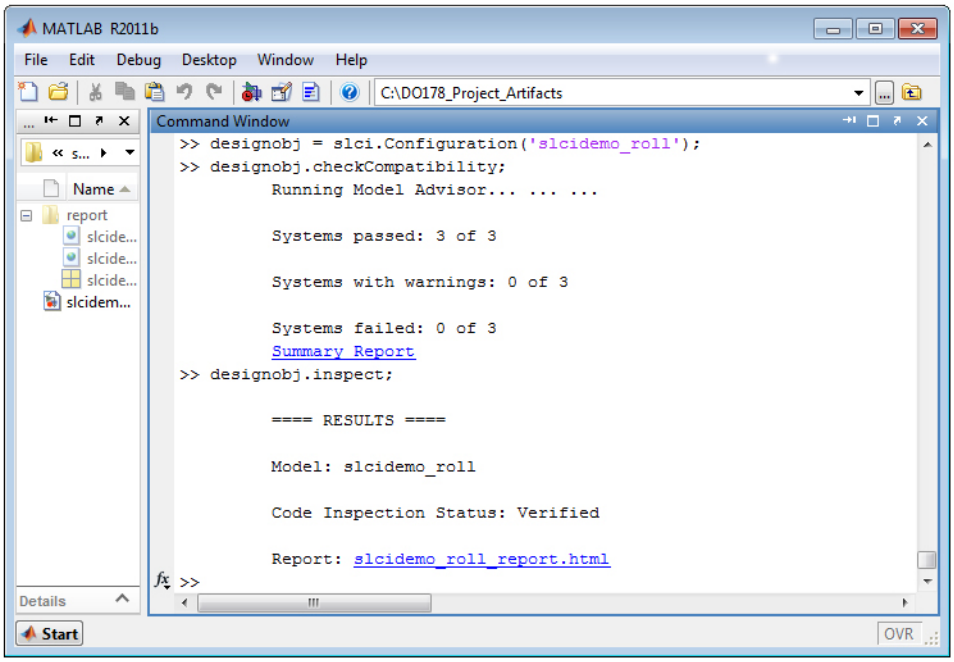
Code inspection is independent of code generation. By default, Simulink Code Inspector assumes that the code has already been generated. However, you can configure the inspector to generate code as part of the code inspection process. You can also specify the locations of the generated source code files, such as those used by your production build processes. Invoking the inspection is done via the user interface or the command line.

Simulink Code Inspector examines the following general categories during code inspection:

- Model interfaces
- Block behavior
- Block connectivity and execution order
- Data and file packaging
- Local variables and functional model elements

The specific analyses performed range from high-level interface assessment, such as checking whether the initialize and step functions were generated, to a detailed analysis of block execution order to determine if the data dependency between two block components is preserved in the generated code.

You can review the inspection status message directly in MATLAB or examine detailed reports, as described in the next section, which include a fine-grained traceability analysis with interactive links to the design objects.



A successful (verified) code inspection using MATLAB commands.

Report Generation

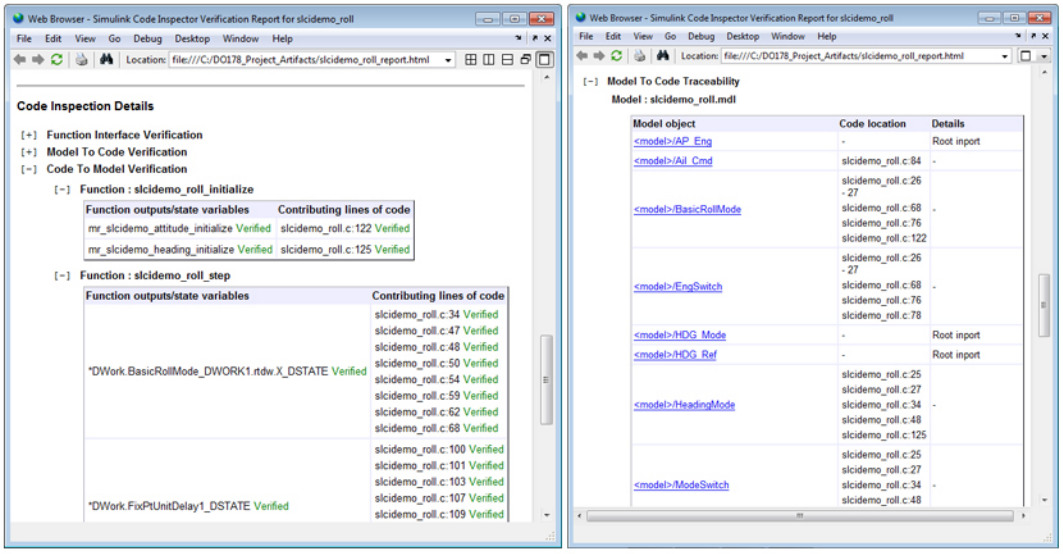
With Simulink Code Inspector, you can generate reports detailing the code inspection results. You specify the location where the reports are stored. This makes it easy to archive and include the reports in your complete certification package. The inspection report begins by identifying the exact model and source code examined, using model checksum, path names, and date/time stamp. It then provides a summary of the verifications performed and their status (for example, verified, partially verified, or failed). A detailed section is also provided describing verification results for every file, function, and line of the generated code.

Code inspection details include:

- Function interface verification
- Model-to-code verification
- Code-to-model verification
- Temporary variable usage

Traceability details include:

- Model-to-code traceability
- Code-to-model traceability



Simulink Code Inspector detail report showing code-to-model verification (left) and model-to-code traceability (right).

DO-178 Objectives

The standard RTCA/DO-178B *Software Considerations in Airborne Systems and Equipment Certification* specifies objectives for source code verification. The automated code inspection Simulink Code Inspector provides significantly reduces the time required for satisfying DO-178 source code verification objectives, as shown in the table below. Additionally, Objective (4) *Source code conforms to standards,* can be satisfied using tools such as the MISRA-C® analyzer provided by Polyspace® code verifiers.

DO-178B Objectives Compatible with Simulink Code Inspector

Annex A Table	Objective	DO-178B Reference	Software Levels
Table A-5	(1) Source code complies with low-level requirements	Section 6.3.4a	A, B, C
Table A-5	(2) Source code complies with software architecture	Section 6.3.4b	A, B, C
Table A-5	(3) Source code is verifiable	Section 6.3.4c	A, B
Table A-5	(5) Source code is traceable to low-level requirements	Section 6.3.4e	A, B, C
Table A-5	(6) Source code is accurate and consistent	Section 6.3.4f	A, B, C

Resources

Product Details, Demos, and System Requirements
www.mathworks.com/products/simulink-code-inspector

Trial Software
www.mathworks.com/trialrequest

Sales
www.mathworks.com/contactsales

Technical Support
www.mathworks.com/support

Online User Community
www.mathworks.com/matlabcentral

Training Services
www.mathworks.com/training

Third-Party Products and Services
www.mathworks.com/connections

Worldwide Contacts
www.mathworks.com/contact