

# **Release Notes for Simulink<sup>®</sup> Code Inspector<sup>™</sup>**

---

## How to Contact MathWorks



[www.mathworks.com](http://www.mathworks.com) Web  
[comp.soft-sys.matlab](mailto:comp.soft-sys.matlab) Newsgroup  
[www.mathworks.com/contact\\_TS.html](http://www.mathworks.com/contact_TS.html) Technical Support



[suggest@mathworks.com](mailto:suggest@mathworks.com) Product enhancement suggestions  
[bugs@mathworks.com](mailto:bugs@mathworks.com) Bug reports  
[doc@mathworks.com](mailto:doc@mathworks.com) Documentation error reports  
[service@mathworks.com](mailto:service@mathworks.com) Order status, license renewals, passcodes  
[info@mathworks.com](mailto:info@mathworks.com) Sales, pricing, and general information



508-647-7000 (Phone)



508-647-7001 (Fax)



The MathWorks, Inc.  
3 Apple Hill Drive  
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

*Release Notes for Simulink® Code Inspector™*

© COPYRIGHT 2011–2013 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

### Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [www.mathworks.com/trademarks](http://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

### Patents

MathWorks products are protected by one or more U.S. patents. Please see [www.mathworks.com/patents](http://www.mathworks.com/patents) for more information.

## R2013a

Support for Stateflow data and transitions .....	2
Support for subsystems and models with active design variants .....	3
Support for models that generate nonreusable or nonreentrant code .....	4
Support for Width, Dynamic Dead Zone, and Dynamic Saturation blocks .....	5
Additional compatibility checks .....	6

## R2012b

Support for Simulink subsystems: If Action Subsystem, Function-Call Subsystem, and Triggered Subsystem blocks .....	8
Support for Simulink blocks: Discrete Time Integrator, Bitwise Operator, Shift Arithmetic, Sqrt, and others ...	9
Support for 2D matrix data type and tunable parameter ..	10
Support for C99 and code replacement math library .....	11
Enhanced code inspection .....	12
DO-178C qualification support (requires DO Qualification Kit) .....	13
Additional compatibility checks .....	14

## R2012a

Support for Additional Simulink Blocks .....	16
Enhanced Sum, MinMax, and Logical Operator Block Support .....	17
Support for Enumerated Types .....	18
Support for Nonencapsulated C++ Code .....	19
Traceability Matrix Generation .....	20

DO-178 Qualification Support (Requires DO Qualification Kit) .....	21
Enhanced Support for Code Optimizations .....	22

## **R2011b**

Introducing Simulink Code Inspector .....	24
Features .....	25
Introductory Example .....	26

# R2013a

---

Version: 1.3  
New Features: Yes  
Bug Fixes: Yes

## **Support for Stateflow data and transitions**

### **Stateflow data**

Code inspection is now supported for Stateflow® charts that use:

- Data stores
- Constants
- Parameters
- Local data
- Nonscalar data

Previously, only Stateflow charts with scalar inputs and outputs were supported for code inspection. For detailed support information, see “Stateflow”.

### **Stateflow transition code inspection status in report**

Simulink® Code Inspector™ reports the traceability and verification status of Stateflow charts and transitions. Previously, code inspection reported only the status at the chart level.

For more information, see “Code Inspection Reports”.

## **Support for subsystems and models with active design variants**

Code inspection now supports models and subsystem blocks with active design variants. Previously, if your model or subsystem block had design variants, the model or subsystem was not compatible with code inspection. For detailed support information, see the code inspection “Model” and “Subsystems” constraints.

## **Support for models that generate nonreusable or nonreentrant code**

Code inspection now supports models that generate nonreusable and nonreentrant code. Previously, compatibility with code inspection, models had to generate reusable and reentrant code.



## **Support for Width, Dynamic Dead Zone, and Dynamic Saturation blocks**

Code inspection is now supported for the following Simulink blocks:

- “Width”
- Dynamic Dead Zone
- Dynamic Saturation

The Dynamic Dead Zone and Dynamic Saturation blocks are mask blocks. For a complete list of mask blocks supported for code inspection, see “Supported Mask Blocks”. Blocks that are supported for code inspection are available in the block library `slcilib`. You can open this library by entering `slcilib` in the MATLAB® Command Window.

## **Additional compatibility checks**

For further verification that your model is compatible with Simulink Code Inspector, there are additional compatibility checks:

- “Check for Terminator blocks connected to Model Reference block outports”
- “Check for root Outport blocks being testpointed”

# R2012b

---

Version: 1.2  
New Features: Yes  
Bug Fixes: Yes

## **Support for Simulink subsystems: If Action Subsystem, Function-Call Subsystem, and Triggered Subsystem blocks**

Code inspection is now supported for the following Simulink subsystem blocks:

- If Action Subsystem
- Function-Call Subsystem
- Triggered Subsystem

For detailed block support information, see Block Constraints.

## Support for Simulink blocks: Discrete Time Integrator, Bitwise Operator, Shift Arithmetic, Sqrt, and others

Code inspection is now supported for the following Simulink blocks.

Action Port	Bitwise Operator	Data Type Propagation	Discrete-Time Integrator
Function-Call Generator	If	Merge	Probe
Reshape	Shift Arithmetic	Sign	Signal Specification
Sqrt	Switch Case	Trigger	Vector Concatenate

For detailed block support information, see Block Constraints.

## **Support for 2D matrix data type and tunable parameter**

Code inspection is enhanced for blocks with 2D matrix data types.

Code inspection no longer requires you to select **Optimization > Signals and Parameters > Inline parameters** on the Configuration Parameters dialog box. In addition, you can now use symbolic names or inlined numerical values for tunable model parameters in generated code. Previously, tunable parameters were constrained to inlined numerical values.

## **Support for C99 and code replacement math library**

You can now inspect code that uses C99 libraries and Supported Functions and Operations in Code Replacement Libraries.

## Enhanced code inspection

Code inspection is enhanced for:

- Trigonometric Function blocks. Code inspection now supports the `sincos` function.
- Product and Gain blocks. Code inspection now supports matrix multiplication.
- Blocks with function-calls at root of model.
- Switch Case Action Subsystem.

For detailed block support information, see Block Constraints.



## **DO-178C qualification support (requires DO Qualification Kit)**

The DO Qualification Kit product now provides documents, templates, test cases, and test procedures that you can use to qualify the Simulink Code Inspector tool for DO-178C certification.

## **Additional compatibility checks**

There are additional checks to verify that your model is compatible with Simulink Code Inspector:

- Check storage class for workspace variables
- Check for usage of synthesized local data stores
- Check loop unrolling threshold setting
- Check destinations of If and Switchcase blocks

For detailed information on using the checks, see [Model Compatibility](#).

# R2012a

---

Version: 1.1  
New Features: Yes  
Bug Fixes: Yes

## Support for Additional Simulink Blocks

Code inspection is now supported for the following Simulink blocks:

- Atomic Subsystem (inlined)
- Enabled Subsystem (inlined)
- Enable Port
- 1-D Lookup Table
- 2-D Lookup Table
- n-D Lookup Table (1 or 2-D only)
- Rounding Function
- Ground

For detailed block support information, see Block Constraints.

## **Enhanced Sum, MinMax, and Logical Operator Block Support**

Code inspection now supports greater than two inputs for Sum, MinMax, and Logical Operator blocks.

## **Support for Enumerated Types**

Code inspection now supports enumerated types used in models.

## **Support for Nonencapsulated C++ Code**

Code inspection now supports models for which the selected target language is C++, as well as C. The target language C++ (Encapsulated) remains unsupported.

## Traceability Matrix Generation

On Windows® systems, R2012a allows you to generate a *traceability matrix* for your model. For a given model, a generated traceability matrix provides information about traceability of model objects between the model and generated code. The traceability matrix is a Microsoft® Excel® file that contains **Model Information**, **Code Interface**, **Code Files**, and **Report** worksheets.

After generating code and inspecting a model, you can generate a traceability matrix using the `slci.ExportTraceReport` function from the MATLAB Command Window. For example:

```
>>  
slci.ExportTraceReport('slcidemo_roll','slcidemo_roll_tracereport')
```

For more information, see Traceability Matrices.



## **DO-178 Qualification Support (Requires DO Qualification Kit)**

The DO Qualification Kit product now provides documents, templates, test cases, and test procedures that you can use to qualify the Simulink Code Inspector tool for DO-178B certification.

## Enhanced Support for Code Optimizations

Code inspection now supports any setting for the following code optimizations, which are located on the **Optimization** and **Optimization > Signals and Parameters** panes of the Configuration Parameters dialog box. Previously, their values were constrained to either on or off.

- **Remove root level I/O zero initialization**  
(ZeroExternalMemoryAtStartup)
- **Remove internal data zero initialization**  
(ZeroInternalMemoryAtStartup)
- **Use memset to initialize floats and doubles to 0.0**  
(InitFltsAndDblsToZero)
- **Use memcpy for vector assignment** (EnableMemcpy)

# R2011b

---

Version: 1.0  
New Features: Yes  
Bug Fixes: No

## **Introducing Simulink Code Inspector**

Simulink Code Inspector automatically compares generated code with its source model to satisfy code-review objectives in DO-178B 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.

## Features

Key features of Simulink Code Inspector Version 1.0 include:

- 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

Use Simulink Code Inspector tooling to:

- Prepare for code inspection during model development.
- Run inspections on code generated from models and review reported results.
- Automatically generate code verification reports to support software certification.

## Introductory Example

The Simulink Code Inspector product provides the following introductory example.

Example	Shows How You Can...
slcidemo_intro	Use MATLAB commands to: <ul style="list-style-type: none"><li>• Prepare a model hierarchy for code generation and code inspection.</li><li>• Automatically generate code for the model hierarchy.</li><li>• Verify the generated code independently of the code generation tool.</li><li>• Purposely introduce an error into the generated code and inspect for failure.</li></ul>

---

**Note** Inspect Generated Code for a Sample Model in the Simulink Code Inspector documentation provides an equivalent example using the Simulink Code Inspector dialog box to control the code inspection workflow.

---