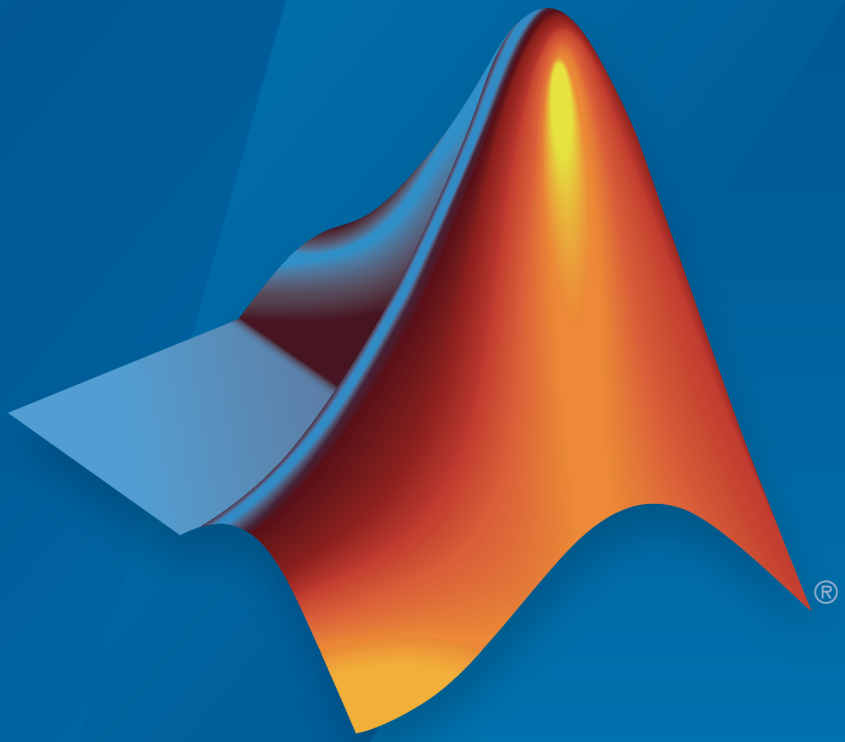
# Simulink® Design Verifier™ Release Notes



MATLAB® & SIMULINK®

MathWorks®

# How to Contact MathWorks

Latest news: www.mathworks.com

Sales and services: www.mathworks.com/sales_and_services

User community: www.mathworks.com/matlabcentral

Technical support: www.mathworks.com/support/contact_us

Phone: 508-647-7000

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

*Simulink® Design Verifier™ Release Notes*

© COPYRIGHT 2007–2016 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 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 for more information.

# Contents

## R2016b


**Model Slicer for Stateflow: Highlight active states and transitions for specified simulation time window** . . . . . 1-2

**Model Slicer: Highlight unexpected behavior in test harnesses created by Simulink Test** . . . . . 1-2

**MCDC Test Generation: Generate tests from MCDC coverage for cascaded Simulink logic blocks** . . . . . 1-2

**Incremental Test Generation: Generate tests to increase coverage for model objects in a test harness** . . . . . 1-2

**Dead Logic Detection: Precisely detect dead logic without rational approximations** . . . . . 1-2

**Detect Run-Time Errors in Supported S-Functions** . . . . . 1-2

**Generate Tests for Relational Boundary Code Coverage for Supported S-Functions** . . . . . 1-3

**Analyze Models that Contain Cell Arrays** . . . . . 1-3

**Detect Dead Logic Only without Floating-Point to Rational Number Conversion Approximation or While Loop Approximation** . . . . . 1-3

## R2016a

**Test Generation: Automatically generate tests for C/C++ S-Functions** . . . . . 2-2

**Model Slicer: Time window adjustment without the need to rerun simulations** . . . . . 2-2

**Variant Reducer: Create sliced models based on active variant configurations** . . . . . 2-2

**Overflow Detection: Automatically find overflow errors for fixed-point types with nonstandard word length** . . . . . 2-2

**Simulink Functions: Perform verification of models that contain Simulink Functions** . . . . . 2-2

**Report Generation: Generate analysis reports in PDF format** . . . . . 2-2

**Model Slicer: Slice by using nonzero start time for simulation time window** . . . . . 2-3

**Model Slicer: Support for root-level inports as starting points for downstream highlighting** . . . . . 2-3

## R2015aSP1

**Bug Fixes**

## R2015b

**Analysis of C S-functions** . . . . . 4-2

**Model Slicer API** . . . . . 4-2

**Analyze minimum and maximum ranges specified for bus elements** . . . . . 4-2

**Model Advisor checks for design error detection** . . . . . 4-2

**Test Generation Advisor improvements** . . . . . 4-2

**Generate test inputs and export them to test cases in Simulink Test** . . . . . 4-2

**Support for Discrete Filter Blocks and Discrete Transfer Fcn Blocks** . . . . . 4-3

## R2015a

**Isolate important model content and reduce model complexity based on design interests with Model Slicer** . 5-2

**Load results from previous Test Generation Advisor analysis** . . . . . 5-2

**“Parameter table” replaces “Parameter configuration table”** . . . . . 5-2

## R2014b

**Test generation for relational boundary values** . . . . . 6-2

**Fast dead logic detection and Model Advisor check** . . . . . 6-2

**Analysis for arrays of buses, For Each block, and For Each Subsystem block** . . . . . 6-3

**Test Generation Advisor to guide component analysis** . . . . . **6-3**

**Improved test generation performance for lookup tables and timers** . . . . . **6-4**

## R2014a

**Parameter Configuration Table for constraint specification and management** . . . . . **7-2**

**Compatibility check integrated with Model Advisor** . . . . . **7-2**

**Condition coverage test generation for Relational Operator blocks** . . . . . **7-2**

**For Each Subsystem block analysis** . . . . . **7-2**

**Parameter handling for Simulink data dictionary** . . . . . **7-3**

**Japanese language localization support** . . . . . **7-3**

## R2013b

**Highlighting of partial results in model during analysis for visualizing progress** . . . . . **8-2**

**Summarizing and highlighting of prior analysis results** . . . **8-2**

**Performance improvements for test generation with input constraints** . . . . . **8-2**

**Analysis time information for objectives in results window, report, and data file** . . . . . **8-2**

**Mac i64 support** . . . . . **8-2**

Improved `while` loop bound detection . . . . . 8-3

Internationalization support for Simulink Design Verifier Options pane . . . . . 8-3

Additional status information for undecided objectives . . . . 8-4

Example of property proving using Truth Table . . . . . 8-4

Continuous state-space block family not stubbable . . . . . 8-4

## R2013a

Detection of out-of-bound array access design errors . . . . . 9-2

## R2012b

Support for Discrete Filter Blocks and Discrete Transfer Fcn Blocks . . . . . 10-2

## R2012a

Design Error Detection For Dead Logic . . . . . 11-2

Filtering Model Objects From Model Coverage . . . . . 11-2

Improved Property Proving For Look-Up Tables . . . . . 11-2

## R2011b+

sldvtimer Function Available For Generating Test Cases . 12-2

## R2011b

Checking Specified Design Minimum and Maximum Values . . . . . 13-2

Improved Support for Trigonometric Functions . . . . . 13-2

Improved Support for Large Lookup Tables . . . . . 13-2

Optimized Handling for Extending Existing Test Cases . . . 13-2

Support for Trigger and Enable Ports for Model Blocks . . 13-2

Changed Format for sldvruntest and sldvruncgvtest Output . . . . . 13-2

Conversion of Error and Warning Message Identifiers . . . . 13-3

## R2011a

Automatic Detection of Overflow and Divide-by-Zero Design Errors . . . . . 14-2

Improved Analysis Results Workflow . . . . . 14-2

Improved Support for Nonlinear Arithmetic and Math Operations . . . . . 14-2

New Capability to Highlight Analysis Results on the Model . . . . . 14-3

New Capability to Review Model Analysis Results in Model Explorer . . . . . 14-3

New Temporal Operator Blocks . . . . . 14-3

Support for Simulink Blocks . . . . . 14-4

## R2010bSP1

**Bug Fixes**

## R2010b

Support for 64-Bit Windows Operating Systems . . . . . 16-2

New Support for Specified Input Minimum and Maximum Values as Analysis Constraints . . . . . 16-2

New Built-In Support for Automating Test Execution in SIL/PIL Mode via the CGV API . . . . . 16-2

New Support for Extracting and Analyzing Stateflow Atomic Subcharts . . . . . 16-2

New Capability to Eliminate Unused Signals from the Generated Harness . . . . . 16-3

Support for Simulink Blocks . . . . . 16-3

sldvlogsignals Replaces sldvlogdata . . . . . 16-3

sldvmergeharness Replaces sldvharnessmerge . . . . . 16-4

## R2010a


**Generate Test Cases for Missing Coverage Data** . . . . . 17-2

**sldvlogdata Function for Logging Test Cases During Simulation** . . . . . 17-2

**Extend Existing Test Cases** . . . . . 17-2

**Demo Library and Models to Support Temporal Properties Specification** . . . . . 17-2

**Support for Stateflow Absolute-Time Temporal Logic Operators** . . . . . 17-3

**Support for Simulink Blocks** . . . . . 17-3


## R2009bSP1


**Bug Fixes**


## R2009b


**New Functions for Verification Objectives and Constraints** . . . . . 19-2

**Support for Enumerated Signals and Parameters** . . . . . 19-2

**New Option to Stop Simulation on Proof Violation** . . . . . 19-3

**New sldvmakeharness Function** . . . . . 19-3

**New sldvreport Function** . . . . . 19-3

New Support for Simulink Blocks . . . . . 19-3

Support for New Blocks . . . . . 19-4

## R2009a

Automatic Stubbing for Unsupported Operations . . . . . 20-2

Long Test Case Optimization . . . . . 20-2

New Support for Blocks . . . . . 20-2

Analyzing External Functions for Embedded MATLAB Function Blocks . . . . . 20-3

Enhanced Block Replacement Capability for Subsystems and Model Blocks . . . . . 20-3

New Implies Block . . . . . 20-3

New Property-Proving Examples and Demos . . . . . 20-3

sldvisactive Function . . . . . 20-4

## R2008b

Simulink Bus Signals and Bus Objects Support . . . . . 21-2

Fixed-Point Data Support . . . . . 21-2

Generating Test Harness Model with Model Reference . . . 21-2

Generating SystemTest TEST-File . . . . . 21-2

**Improved Search Algorithms** . . . . . 21-3

**New Data File Format** . . . . . 21-3

**New HTML Report** . . . . . 21-4

**Blocks with No Input Ports Limitation** . . . . . 21-4

## R2008a+

**Bug Fixes**

## R2008a

**Embedded MATLAB Subset Support** . . . . . 23-2

**Enhanced Support for Stateflow Truth Tables** . . . . . 23-2

**New Simulink Design Verifier Data File Options** . . . . . 23-2

**New Test Suite Optimization Setting** . . . . . 23-2

## R2007b+

**Bug Fixes**

## R2007b

**Fixed-Point Data Type Support** . . . . . **25-2**

## R2007a+

**Introducing the Simulink Design Verifier Software** . . . . . **26-2**

# R2016b

Version: 3.2

New Features

Bug Fixes

## Model Slicer for Stateflow: Highlight active states and transitions for specified simulation time window

You can use the Model Slicer to highlight active Stateflow® states and transitions for a specified simulation time window. See “Highlight Functional Dependencies” and “Using Model Slicer with Stateflow”.

## Model Slicer: Highlight unexpected behavior in test harnesses created by Simulink Test

You can use the Model Slicer to highlight functional dependencies in test harnesses created by Simulink® Test™. See “Highlight Functional Dependencies”.

## MCDC Test Generation: Generate tests from MCDC coverage for cascaded Simulink logic blocks

Use Simulink Design Verifier™ to generate tests to satisfy MCDC coverage objectives for models containing cascaded Simulink logic blocks if “Treat Simulink logic blocks as short-circuited” is enabled. See “Analyzing MCDC for Cascaded Logic Blocks”, “Logical Operator Cascade Patterns”, and “Modified Condition and Decision Coverage in Simulink Design Verifier”.

## Incremental Test Generation: Generate tests to increase coverage for model objects in a test harness

Use Simulink Design Verifier to incrementally increase coverage for model objects in a test harness. See “Increase Coverage for Referenced Models in a Test Harness”.

## Dead Logic Detection: Precisely detect dead logic without rational approximations

Improved precision of dead logic detection by removing approximation effects, such as relational boundary effects.

## Detect Run-Time Errors in Supported S-Functions

When you run a design error detection analysis, Simulink Design Verifier supports the following subset of run-time error detection for supported S-Functions:

- Division by zero
- Out of bound array, including invalid pointer access
- Dead logic detection

For more information, see “What Is Design Error Detection?” and “Support Limitations for S-Functions”.

## Generate Tests for Relational Boundary Code Coverage for Supported S-Functions

When you generate tests for models with compatible S-Functions, Simulink Design Verifier can create test cases for relational boundary objectives for the S-Functions. For more information, see “Configuring S-Function for Test Case Generation” and “Relational Boundary”.

## Analyze Models that Contain Cell Arrays

Simulink Design Verifier supports analysis of models that contain cell arrays.

## Detect Dead Logic Only without Floating-Point to Rational Number Conversion Approximation or While Loop Approximation

When you detect dead logic only, Simulink Design Verifier analyzes your model without floating-point to rational number conversion approximation or while loop approximation. For more information about the types of design error detection for dead logic in Simulink Design Verifier, see “Dead Logic Detection”. For more information about approximations in Simulink Design Verifier, see “Approximations”.

# R2016a

Version: 3.1

New Features

Bug Fixes

## Test Generation: Automatically generate tests for C/C++ S-Functions

Simulink Design Verifier generates tests for Decision, Condition, and MCDC objectives for C/C++ code within S-Functions. For more information, see:

- Configuring S-Function for Test Case Generation
- Workflow for Test Case Generation

## Model Slicer: Time window adjustment without the need to rerun simulations

When you Refine Highlighted Model with Model Slicer, you can refine the slice highlight time window without the need to resimulate the model.

## Variant Reducer: Create sliced models based on active variant configurations

You use the **Variant Manager** or Model Slicer to generate a simplified standalone model including only selected variant configurations. For more information, see Simplification of Variant Systems.

## Overflow Detection: Automatically find overflow errors for fixed-point types with nonstandard word length

Design error detection in Simulink Design Verifier supports fixed-point types with nonstandard word length. For more information, see Detect Integer Overflow and Division-by-Zero Errors.

## Simulink Functions: Perform verification of models that contain Simulink Functions

Simulink Design Verifier supports Simulink Functions.

## Report Generation: Generate analysis reports in PDF format

Simulink Design Verifier can generate a PDF report that contains detailed information about the analysis results if you have Simulink Report Generator™. For more information, see Simulink Design Verifier Reports.

## Model Slicer: Slice by using nonzero start time for simulation time window

When you Refine Highlighted Model with Model Slicer, you can use a nonzero start time for the simulation time window.

## Model Slicer: Support for root-level inports as starting points for downstream highlighting

Model Slicer supports root-level inports as starting points for downstream highlighting. Model Slicer now treats these root-level inports as nonvirtual blocks. For more information, see Highlight Functional Dependencies.

# R2015aSP1

**Version: 2.8.1**

**Bug Fixes**

# R2015b

Version: 3.0

New Features

Bug Fixes

## Analysis of C S-functions

Simulink Design Verifier provides the option to incorporate C S-function behavior for all modes of analysis.

## Model Slicer API

Simulink Design Verifier adds an API for programmatically analyzing functional dependencies using Model Slicer.

## Analyze minimum and maximum ranges specified for bus elements

Simulink Design Verifier analysis considers minimum and maximum values specified for bus elements in Simulink models. Minimum and maximum values specified for bus elements in Stateflow charts are not considered. See “Specify Input Range for a Bus Element”.

## Model Advisor checks for design error detection

The Model Advisor includes additional Simulink Design Verifier design error detection checks:

- Detect integer overflow
- Detect division by zero
- Detect out of bound array access
- Detect violation of minimum and maximum values

## Test Generation Advisor improvements

Test Generation Advisor reduces batch component extraction time and adds the ability to save results, reload results, and continue using previous analysis results.

## Generate test inputs and export them to test cases in Simulink Test

You can generate test inputs using Simulink Design Verifier analysis and export them to test cases in Simulink Test. This option appears in the Simulink Design Verifier results dialog box. You can also generate test cases from Simulink Design Verifier analysis results files. See “Export Test Cases to Simulink Test”.

## Support for Discrete Filter Blocks and Discrete Transfer Fcn Blocks

Simulink Design Verifier now supports Discrete Filter blocks and Discrete Transfer Fcn blocks for all **Reset Mode** settings.

# R2015a

Version: 2.8

New Features

Bug Fixes

## Isolate important model content and reduce model complexity based on design interests with Model Slicer

Model Slicer facilitates determining model block diagram dependencies, and generating simplified models that maintain simulation behavior. You use Model Slicer to analyze your model beginning with one or more starting points and an analysis direction. You can exclude model items and signal paths, and consider simulation effects in your analysis.

Model Slicer highlights items that your starting points depend on, and/or the items that depend on your starting points. You can generate a standalone, simplified model from certain model highlights.

You can access the Model Slice Manager by selecting **Design Verifier > Model Slicer**. See Model Simplification with Dependency Analysis.

## Load results from previous Test Generation Advisor analysis

Test Generation Advisor loads the previous model analysis results after closing the advisor window or the model.

## "Parameter table" replaces "Parameter configuration table"

The table that you access in the Model Configuration Parameters dialog box, on the **Design Verifier > Parameters** pane, is now called "Parameter table." Use this table to specify parameter value ranges or constraints. The function of the table has not changed.

# R2014b

Version: 2.7

New Features

Bug Fixes

Compatibility Considerations

## Test generation for relational boundary values

Simulink Design Verifier generates tests that satisfy a relational boundary objective.

The relational boundary objective applies to blocks that have an explicit or implicit relational operation. The objectives also apply to Stateflow transitions and MATLAB® function blocks that contain a relational operation.

For these blocks, the tests check the relational operations with:

- Equal operand values. This part of relational boundary objective applies only if both operands are integers or fixed-point numbers.
- Operand values that differ by a certain tolerance. This part of relational boundary objective applies to all operands. For integer and fixed-point operands, the tolerance is fixed. For floating-point operands, you can either use a predefined tolerance or specify your own tolerance value.

For more information, see Design Verifier Pane: Test Generation.

## Fast dead logic detection and Model Advisor check

Dead logic detection is optimized for faster performance. It is integrated with the Model Advisor.

In releases prior to R2014b and after R2012b, design error detection included a single analysis option to detect dead logic. This previous dead logic detection also reported active logic.

Design error detection for dead logic now comprises two analysis options:

- Detection of dead logic only. Does not report active logic or undecided objectives. Available in the Model Advisor and in the Configuration Parameters dialog box under **Design Verifier > Options > Design Error Detection**.
- Detection of active logic. Runs concurrently with dead logic detection. In rare cases, can also find additional dead logic. Available in the Configuration Parameters dialog box under **Design Verifier > Options > Design Error Detection**.

By default, detection of active logic is turned off. However, if you have run dead logic detection on a model using a previous release, when you load your model in the new

release, both detection of dead logic and active logic are turned on. Therefore, you have the same results of dead logic detection as before.

For more information, see:

- Detect Dead Logic Only
- Detect Dead and Active Logic

## Analysis for arrays of buses, For Each block, and For Each Subsystem block

Analysis supports arrays of buses.

Analysis supports For Each and For Each Subsystem blocks, with the following limitation:

- When For Each Subsystem contains one or more Simulink Design Verifier Test Condition, Test Objective, Proof Assumption, or Proof Objective blocks, this block is not supported.

## Test Generation Advisor to guide component analysis

Test Generation Advisor runs preliminary analysis on your model. For each component, it shows recommended Simulink Design Verifier options for test generation analysis. Test Generation Advisor also reports the number and status of test objectives for each component.

For each component, Test Generation Advisor reports one of the following statuses:

- Analyzable, meaning that you can use Simulink Design Verifier to analyze this component.
- Complex, meaning that you can use Simulink Design Verifier to analyze this component, but the analysis can be time-consuming. It is possible that the component does not receive full coverage from generated tests. For more information, see Sources of Model Complexity.
- Incompatible, meaning that this component is incompatible with Simulink Design Verifier. For more information, see Check Model Compatibility.

To open Test Generation Advisor, from the Simulink Editor, select **Analysis > Design Verifier > Generate Tests > Advisor**.

## Improved test generation performance for lookup tables and timers

Test generation analysis includes optimizations to improve performance for 1-D Lookup Table and 2-D Lookup Table blocks. These optimizations are in effect for floating-point lookup tables using linear interpolation.

# R2014a

Version: 2.6

New Features

Bug Fixes

## Parameter Configuration Table for constraint specification and management

You can use the Parameter Configuration Table to specify parameters to treat as variables during analysis. Previously, you specified parameter configurations only with a MATLAB code file.

Use the Parameter Configuration Table to:

- Autogenerate value ranges for parameters in your model.
- Enter your own value ranges for parameters in your model.
- Highlight objects in your model that have parameters configured to act as variables during analysis.
- Import and export parameter configurations from MATLAB code files.

For more information, see Define Constraint Values for Parameters and Store Parameter Constraints in MATLAB Code Files.

## Compatibility check integrated with Model Advisor

You can check the compatibility of your model for Simulink Design Verifier analysis using the Model Advisor. See Check compatibility with Simulink Design Verifier.

For more information on using the Model Advisor, see Consult the Model Advisor in the Simulink documentation.

## Condition coverage test generation for Relational Operator blocks

Analysis supports test case generation for condition coverage for Relational Operator, Compare to Constant, and Compare to Zero blocks. For more information, see Model Objects That Receive Coverage in the Simulink Verification and Validation™ documentation.

## For Each Subsystem block analysis

Analysis supports For Each and For Each Subsystem blocks, with the following limitations:

- When For Each Subsystem contains another For Each Subsystem, not supported.

- When For Each Subsystem contains one or more Simulink Design Verifier Test Condition, Test Objective, Proof Assumption, or Proof Objective blocks, not supported.

## Parameter handling for Simulink data dictionary

Analysis supports models with parameters stored in data dictionaries. For more information, see What Is a Data Dictionary? in the Simulink documentation.

## Japanese language localization support

Simulink Design Verifier is available in Japanese for Japanese localized systems. For more information, see Internationalization.

# R2013b

Version: 2.5

New Features

Bug Fixes

Compatibility Considerations

## Highlighting of partial results in model during analysis for visualizing progress

Before Simulink Design Verifier analysis on your model is complete, you can view its progress. Seeing the analysis progress can help you identify components of your model that take a longer time to analyze than others.

During analysis, in the Simulink Design Verifier log window, click **Highlight** to view the analysis results. The Simulink Design Verifier Results window shows the elapsed time of the analysis. Highlighting appears on model objects for which the analysis has begun. For more information, see Highlighted Results on the Model.

## Summarizing and highlighting of prior analysis results

You can load previously generated analysis results for a model. In the Simulink Editor, select **Analysis > Design Verifier > Results > Load** to choose the data file with the results of a previous analysis. The Simulink Design Verifier Results window opens with options that you can use to explore the results. For more information, see Load Previous Results.

You can also use the sldvloadresults and sldvhighlight functions to explore previous analysis results for a model.

## Performance improvements for test generation with input constraints

For test case generation analysis, especially when using **Nonlinear Extended** test suite optimizations, there are improved heuristics for handling input constraints defined using the Test Condition block.

## Analysis time information for objectives in results window, report, and data file

The Simulink Design Verifier Results window, generated analysis reports, and data files include information about time spent analyzing objectives.

## Mac i64 support

Simulink Design Verifier supports the 64-bit Intel® Macintosh platform. For more information, see System Requirements.

## Improved `while` loop bound detection

If your model contains a `while` loop, Simulink Design Verifier tries to detect a conservative constant bound that allows the `while` loop to exit. If the software cannot find a constant bound, it performs a `while` loop approximation. With this approximation, the analysis does not prove objectives to be valid or unsatisfiable and it does not prove dead logic. The generated analysis report notes this approximation.

The behavior of the `while` loop approximation is consistent in all modes of analysis, as described in the following table.

| Analysis Mode | While Loop Approximation |
|---|---|
| Design Error Detection | Sets number of `while` loop iterations to 3. Does not report dead logic or valid objectives. |
| Test Case Generation | Sets number of `while` loop iterations to 3. Does not report unsatisfiable objectives. |
| Property Proving | Sets number of `while` loop iterations to 3. Does not report valid objectives. |

## Compatibility Considerations

In previous versions of Simulink Design Verifier, if the software did not detect a bound for a `while` loop, it terminated the analysis or performed an approximation, as described in the following table.

| Analysis Mode | While Loop Approximation (pre-R2013b) |
|---|---|
| Design Error Detection | Set number of `while` loop iterations to 3. |
| Test Case Generation | Set number of `while` loop iterations to 3. |
| Property Proving | Terminated analysis. |

This generated analysis report noted this approximation.

## Internationalization support for Simulink Design Verifier Options pane

Japanese internationalization is supported for the Simulink Design Verifier **Options** pane. For more information, see Internationalization.

## Additional status information for undecided objectives

Simulink Design Verifier reports objectives that the analysis was unable to decide because of nonlinearities or division by zero. The software also reports objectives that the analysis was unable to decide because of stubbing. Objectives that have status Undecided Due to Stubbing can include objectives that, in releases prior to R2013b, were marked as Satisfied – No Test Case or Falsified – No Counterexample. For more information, see Objectives Undecided.

## Example of property proving using Truth Table

The Property Proving Using MATLAB Truth Table Block example shows how to verify safety properties using Simulink Design Verifier property proving analysis. The example model uses sldv.prove and sldv.assume functions in a Truth Table block that uses MATLAB as the action language.

To see this example, at the MATLAB command prompt, type:

```
sldvexSBRVerificationTruthTableExample
```

## Continuous state-space block family not stubbable

The Continuous Simulink blocks State-Space, Transfer Fcn, and Zero-Pole are not supported and not stubbable for Simulink Design Verifier analysis. For information on blocks that are supported for Simulink Design Verifier analysis, see Simulink Block Support.

## Compatibility Considerations

If you have a model that contains one or more continuous State-Space, Transfer Fcn, or Zero-Pole blocks, your model is incompatible with Simulink Design Verifier. Consider analyzing smaller portions of your model to work around this incompatibility, as described in Extract Subsystems for Analysis.

# R2013a

Version: 2.4

New Features

Bug Fixes

## Detection of out-of-bound array access design errors

Before you simulate a model, you can use design error detection analysis to find out of bound array access errors. To detect out of bound array access errors in your model, from the Simulink Editor, select **Analysis > Design Verifier > Options**. In the Configuration Parameters dialog box, on the **Design Verifier > Design Error Detection** pane, select **Out of bound array access**. On the **Design Verifier** pane, click **Detect Errors**.

For more information, see Detect Out of Bound Array Access Errors.

# R2012b

Version: 2.3

New Features

Bug Fixes

Compatibility Considerations

## Support for Discrete Filter Blocks and Discrete Transfer Fcn Blocks

Simulink Design Verifier now supports Discrete Filter blocks and Discrete Transfer Fcn blocks if their **Reset Mode** is set to `None`. Discrete Filter blocks and Discrete Transfer Fcn blocks with their **Reset Mode** set to any other option are not supported.

# R2012a

Version: 2.2

New Features

Bug Fixes

## Design Error Detection For Dead Logic

You can now use design error detection to find dead logic in your model. Simulink Design Verifier uses multiple analysis engines, including Polyspace® and Prover®, to identify the dead logic. Previously, to uncover dead logic in your model, you had to analyze the results of test case generation. To conduct design error detection analysis, in the Configuration Parameters dialog box, on the **Design Verifier > Design Error Detection** pane, select **Dead Logic**. See Detecting Dead Logic.

## Filtering Model Objects From Model Coverage

Simulink Design Verifier now allows you to filter certain model objects from model coverage during test case generation. The objects that you specify to exclude are stored in an external file. In the Configuration Parameters dialog box, on the **Design Verifier > Test Generation** pane, select **Ignore objectives based on filter**. In the **Coverage filter file** field, enter the file name.

For more information about coverage filtering, see Excluding Model Objects From Coverage in the Simulink Verification and Validation documentation.

## Improved Property Proving For Look-Up Tables

Simulink Design Verifier now automatically optimizes property proving for Look-Up Table blocks, improving performance.

# R2011b+

Version: 2.1.1

New Features

Bug Fixes

## sldvtimer Function Available For Generating Test Cases

In R2011b+, the sldvtimer function is available to identify, change, and display timer optimizations. When generating test cases, you can use the function to:

- Identify where you can apply timer optimization in your model.
- Determine if Simulink Design Verifier applied timer optimizations during test generation.
- Configure timer optimizations.

# R2011b

Version: 2.1

New Features

Bug Fixes

Compatibility Considerations

## Checking Specified Design Minimum and Maximum Values

If you have specified minimum and maximum values on your Simulink model, the Simulink Design Verifier software can analyze your model to make sure that the intermediate and output signals do not violate the specified values. If the analysis finds a violation, the software creates a test case that demonstrates the violation.

This check is a new option for the design error detection analysis mode. For detailed information, see Checking for Specified Intermediate Minimum and Maximum Signal Values.

## Improved Support for Trigonometric Functions

For a test case generation analysis, the `CombinedObjectives (Nonlinear)` and `LargeModel (Nonlinear Extended)` test suite optimizations have improved support for trigonometric and nonlinear math functions.

## Improved Support for Large Lookup Tables

All the Simulink Design Verifier analysis modes have improved support for large lookup tables.

## Optimized Handling for Extending Existing Test Cases

Existing test cases are now represented in a more memory-efficient way internally in the test generation engine. This improvement makes it possible to extend significantly longer existing test cases with more time steps.

## Support for Trigger and Enable Ports for Model Blocks

The Simulink Design Verifier software fully supports root-level Trigger and Enable ports for referenced models.

## Changed Format for sldvruntest and sldvruncgvtest Output

The output format for sldvruntest and sldvruncgvtest has changed in R2011b. In R2011b, the output argument contains the following data for each test case executed in an array of `Simulink.SimulationOutput` objects.

| Field | Description |
| --- | --- |
| `tout_sldvruncgvtest` | Simulation time |
| `xout_sldvruncgvtest` | State data |
| `yout_sldvruncgvtest` | Output signal data |
| `logsout_sldvruncgvtest` | Signal logging data for:<br>• Signals that are connected to outports<br>• Signals that are configured for logging on the model |

## Compatibility Considerations

If you have scripts that depend on the output from `sldvruntest` and `sldvruncgvtest`, you can temporarily specify the output format. Use the nonvisible field `outputFormat` in the `runOpts` structure that sldvruntestopts creates as follows:

```
runOpts = sldvruntestopts;
runOpts.outputFormat = 'TimeSeries';
sldvruntest(model_name, sldvData, runOpts);
```

or

```
runOpts = sldvruntestopts;
runOpts.outputFormat = 'StructureWithTime';
sldvruntest(model_name, sldvData, runOpts);
```

## Conversion of Error and Warning Message Identifiers

For R2011b, error and warning message identifiers have changed in Simulink Design Verifier.

## Compatibility Considerations

If you have scripts or functions that use message identifiers that changed, you must update the code to use the new identifiers. Typically, message identifiers are used to turn off specific warning messages, or in code that uses a `try/catch` statement and performs an action based on a specific error identifier.

For example, the `SLDV:InvalidConfSet` identifier has changed to `SLDV:configcomp_get:InvalidConfSetRef`. If your code checks for `SLDV:InvalidConfSet`, you must update it to check for `SLDV:configcomp_get:InvalidConfSetRef` instead.

To determine the identifier for a warning, run the following command just after you see the warning in the MATLAB command window:

```
[MSG,MSGID] = lastwarn;
```

This command saves the message identifier to the variable `MSGID`.

To determine the identifier for an error that appears at the MATLAB prompt, run the following commands just after you see the error:

```
exception = MException.last;
MSGID = exception.identifier;
```

---

**Note:** Warning messages indicate a potential issue with your code. While you can turn off a warning, a suggested alternative is to change your code so it runs warning-free.

---

# R2011a

Version: 2.0

New Features

Bug Fixes

## Automatic Detection of Overflow and Divide-by-Zero Design Errors

If your Simulink model performs arithmetic operations, the Simulink Design Verifier software can analyze the model to identify design errors that occur at run time. The analysis detects two types of errors:

- Integer or fixed-point data overflow
- Division by zero

After the analysis is complete, you can create a harness model that includes test cases for each error.

For more information, see Detecting Integer Overflow and Division-by-Zero Errors.

## Improved Analysis Results Workflow

After you analyze a Simulink model using the Simulink Design Verifier software, you can choose how you want to review the results. You have the following options:

- Highlight the analysis results on the model.
- Generate the detailed analysis report.
- Create the harness model with the generated test cases in the Signal Builder.
- Simulate the test cases and produce a model coverage report (test case generation only).

## Improved Support for Nonlinear Arithmetic and Math Operations

The Simulink Design Verifier test-generation software includes improved support for nonlinear arithmetic and math operations that:

- Improves test generation for models containing nonlinear operations.
- Improves support of math operations such as trigonometric functions.
- Improves scalability to very large models.

To use these new strategies, in the Configuration Parameters dialog box, on the **Design Verifier > Test Generation** pane, set the **Test suite optimization** parameter to one of the following:

- `CombinedObjectives (Nonlinear Extended)`

- `LargeModel (Nonlinear Extended)`

## New Capability to Highlight Analysis Results on the Model

When a Simulink Design Verifier analysis is complete, you can specify that the software use color highlighting on the model to indicate the analysis results for individual objects. When you click an object, you see the detailed results specific to that object.

You can generate a detailed analysis report or create a harness model that contains test case signals at any time.

After you run a design error detection analysis, the model is highlighted by default. After you run a test case generation or property-proving analysis, you can highlight the model.

For more information, see Highlighted Results on the Model.

## New Capability to Review Model Analysis Results in Model Explorer

If you close the Simulink Design Verifier analysis results so you can fix the causes of problems in your model, you might need to review the analysis results again. As long as your model remains open, you can view the results of your most recent analysis results in the Model Explorer by selecting **Tools > Design Verifier > Latest Results**.

From the Model Explorer, you can:

- Highlight the analysis results on the model.
- Generate the detailed analysis report.
- Create the harness model with the generated test cases in the Signal Builder.
- Simulate the test cases and produce a model coverage report (test case generation only).

For more information, see Reviewing Analysis Results in the Model Explorer.

## New Temporal Operator Blocks

The Simulink Design Verifier block library includes three new blocks that allow you to define temporal properties on Boolean signals in your model:

- Detector — Detect true duration on input and construct output true duration based on output type

- Extender — Extend true duration of input
- Within Implies — Capture within implication if observed input is `true` within each true duration of first input

## Support for Simulink Blocks

The Simulink Design Verifier software now supports the following Simulink blocks:

- Discrete Derivative
- Function-Call Feedback Latch (new block)
- Probe
- Rate Limiter Dynamic
- Trigonometric Function — Supported when **Function** is `sin`, `cos`, or `sincos` and **Approximation method** is `CORDIC`.
- Variant Subsystem
- Weighted Sample Time
- Weighted Sample Time Math

# R2010bSP1

**Version: 1.7.1**

**Bug Fixes**

# R2010b

Version: 1.7

New Features

Bug Fixes

Compatibility Considerations

## Support for 64-Bit Windows Operating Systems

The Simulink Design Verifier software can now execute in 64-bit mode on 64-bit Windows® systems.

## New Support for Specified Input Minimum and Maximum Values as Analysis Constraints

The Simulink Design Verifier software can consider the specified minimum and maximum values for input signals during a test case generation or property proving analysis. This feature allows you to:

- Constrain the test cases based on the specified minimum and maximum values.
- Assume the specified minimum and maximum values during property-proving analysis.

## New Built-In Support for Automating Test Execution in SIL/PIL Mode via the CGV API

The new sldvruncgvtest function allows you to execute test cases in Software-in-the-Loop (SIL) or Processor-in-the-Loop (PIL) mode using the Code Generation Verification (CGV) API methods.

## New Support for Extracting and Analyzing Stateflow Atomic Subcharts

The Simulink Design Verifier software now provides support for Stateflow atomic subcharts. Atomic subcharts make it easier to isolate parts of a Stateflow chart for development and analysis.

The Simulink Design Verifier support for atomic subcharts allows you to:

- Generate test cases and prove properties for an atomic subchart in a Stateflow chart within a Simulink model.
- Use sldvextract to extract the contents of an atomic subchart and create a model for the Simulink Design Verifier software to analyze.

## New Capability to Eliminate Unused Signals from the Generated Harness

To improve the performance of harness model creation during a Simulink Design Verifier analysis, from the Signal Builder block, the software can omit signals that have no effect on the output of the model.

sldvmakeharness supports this capability if you set the `usedSignalsOnly` harness model option to `true`.

## Support for Simulink Blocks

The Simulink Design Verifier software now supports the following Simulink blocks:

- Dead Zone and Dead Zone Dynamic
- Lookup Table Dynamic
- Probe (Partial support)
- Width

## sldvlogsignals Replaces sldvlogdata

The sldvlogsignals function replaces the `sldvlogdata` function. Use `sldvlogsignals` to:

- Simulate a Simulink model and in that model, log all the inputs to a specified Model block.
- Simulate all or some of the test cases in a harness model created by:
  - Simulink Design Verifier analysis
  - sldvmakeharness
  - slvnvmakeharness

## Compatibility Considerations

The sldvlogsignals function replaces the `sldvlogdata` function. Currently, if you use the `sldvlogdata` function, it automatically redirects to `sldvlogsignals`. Update your scripts to use `sldvlogsignals`.

## sldvmergeharness Replaces sldvharnessmerge

The sldvmergeharness function replaces the `sldvharnessmerge` function. `sldvmergeharness` combines the test cases and initializations from any specified harness models into a single harness model.

## Compatibility Considerations

Currently, if you use the `sldvharnessmerge` function, it automatically redirects to `sldvmergeharness`. Update your scripts to use `sldvmergeharness`.

# R2010a

Version: 1.6

New Features

Bug Fixes

## Generate Test Cases for Missing Coverage Data

The Simulink Design Verifier software now offers the option to isolate test generation to objectives that are not satisfied in simulation coverage results. If you simulate your model, but do not achieve 100% coverage, you can analyze the model using the Simulink Design Verifier test-generation capability to find test cases that achieve the missing coverage.

If you select the **Ignore objectives satisfied in existing coverage data** parameter in the Configuration Parameters dialog box, you can import the coverage data file; the analysis eliminates all objectives satisfied in the coverage results.

## sldvlogdata Function for Logging Test Cases During Simulation

With the new `sldvlogdata`function, you can:

- Simulate a Simulink model and in that model, log all the inputs to a Model block.
- Simulate all or some of the test cases in a harness model created by the Simulink Design Verifier software and log all the inputs to the test unit.

You can save logged data to a MAT-file and use that file as input to the Simulink Design Verifier software for extending tests. This allows you to generate more realistic test cases and extends the analysis to complete the test suite.

## Extend Existing Test Cases

The Simulink Design Verifier software now offers the option to extend existing test cases with additional time steps to generate complete test suites. This allows the software to generate test cases for parts of your model that are hard to analyze.

If you enable the **Extend existing test cases** parameter in the Configuration Parameters dialog box, the software imports the logged test cases from a MAT-file. If you also enable the **Ignore objectives satisfied by existing test cases** parameter, the analysis generates results, ignoring the coverage objectives satisfied by the logged test cases. Otherwise, the analysis efficiently creates a complete test suite.

## Demo Library and Models to Support Temporal Properties Specification

The Simulink Design Verifier software includes a new **Temporal Property Specification** demo category that includes:

- A Temporal Operator Blocks demo library that contains the following blocks and examples:
  - Detector — Detects a user-specified length of true duration on the input signal and constructs an output true duration of length based on the output type.
  - Extender — Extends the true duration of the input signal by a fixed number of time steps or indefinitely.
  - Within Implies — Captures the within implication by observing whether the second input is true for at least one time step within each true duration of the first input.
  - Temporal Property Specification examples — A library model that includes examples that use the Detector, Extender, and Within Implies blocks
- Two demo models that contain these blocks:
  - Debounce Temporal Properties
  - Power Window Controller Temporal Properties

## Support for Stateflow Absolute-Time Temporal Logic Operators

The Simulink Design Verifier software now supports the Stateflow absolute-time temporal logic operators. For more information, see Operators for Absolute-Time Temporal Logic in the Stateflow documentation.

## Support for Simulink Blocks

The Simulink Design Verifier software now fully supports the following blocks:

- Backlash
- Cosine
- Discrete Derivative
- Sine

The Simulink Design Verifier software now provides improved support for the following blocks:

- Interpolation Using Prelookup
- Lookup Table (n-D)

For more information, see Simulink Block Support.

# R2009bSP1

**Version: 1.5.1**

**Bug Fixes**

# R2009b

Version: 1.5

New Features

Bug Fixes

Compatibility Considerations

## New Functions for Verification Objectives and Constraints

Use these four new functions to specify objectives and constraints within an Embedded MATLAB® script. You can use these functions instead of the corresponding Simulink Design Verifier blocks.

| Function | Purpose | Corresponding Block |
|---|---|---|
| sldv.assume | Proof assumption | Proof Assumption |
| sldv.condition | Test condition | Test Condition |
| sldv.prove | Proof objective | Proof Objective |
| sldv.test | Test objective | Test Objective |

These functions:

- Identify mathematical relationships for objectives and constraints in a form that can be more natural than using block parameters
- Support specifying multiple constraints without complicating the model
- Provide access to the power of the MATLAB software
- Support separation of verification and model design

## Compatibility Considerations

The following functions will be removed in a future release:

- `dv.assume`
- `dv.condition`
- `dv.prove`
- `dv.test`

So that your models with those functions will work in future releases, replace these functions with the corresponding new function added in this release. For example, replace `dv.assume` with `sldv.assume`.

## Support for Enumerated Signals and Parameters

The Simulink Design Verifier software now supports Simulink models with enumerations. All the Simulink Design Verifier library blocks support enumerated parameters, constants, and inputs.

## New Option to Stop Simulation on Proof Violation

The Simulink Design Verifier software allows you to stop a model simulation if it encounters a property violation. You enable this capability by inserting a Proof Objective block into a model and setting the **Stop simulation when the property is violated** parameter. If the simulation detects a violation of the property specified in the Proof Objective block, it terminates with an error.

Therefore, you can now verify a counterexample that was detected during a Simulink Design Verifier analysis.

## New sldvmakeharness Function

With the new sldvmakeharness function, you can:

- Create a test harness model from existing Simulink Design Verifier analysis data.
- Create an empty test harness model directly from a Simulink model.

## New sldvreport Function

You can now generate and customize a report from existing Simulink Design Verifier analysis data with the new sldvreport function.

## New Support for Simulink Blocks

The Simulink Design Verifier software now supports the following blocks and parameters:

- Direct Lookup Table (n-D)
- Discrete Transfer Fcn
- Lookup Table and Lookup Table (2-D) — Except when the **Lookup method** block parameter specifies `Interpolation-Extrapolation` and the block's input and output signals do not have the same floating-point data type.
- Math Function — All signal types now support the `hermitian` and `transpose` function parameter settings
- Rate Limiter — For signals of all data types
- Shift Arithmetic — For all parameters and signals of all data types

- Tapped Delay
- Transfer Fcn Direct Form II
- Transfer Fcn Direct Form II Time Varying

## Support for New Blocks

The Simulink Design Verifier software supports the following new Simulink blocks:

- Discrete PID Controller
- Discrete PID Controller (2 DOF)
- Enumerated Constant

# R2009a

Version: 1.4

New Features

Bug Fixes

## Automatic Stubbing for Unsupported Operations

Automatic stubbing allows you to complete a test-generation or property-proving analysis even if the model contains blocks or functions that the Simulink Design Verifier software does not support, like S-functions and C math operations.

By default, this feature is unavailable. To enable automatic stubbing before running an analysis, on the Configuration Parameters **Design Verifier** main pane, select **Automatic stubbing of unsupported blocks and functions**. In addition, if the compatibility check finds unsupported blocks that automatic stubbing can handle, you can enable automatic stubbing at that time.

## Long Test Case Optimization

`Long test cases` is a new option for the **Test suite optimization** parameter. The `Long test cases` option instructs the Simulink Design Verifier software to create fewer but longer test cases that each satisfy multiple test objectives. With this option, you can customize the analysis results, run a more efficient analysis, and create easier-to-review results, in both Signal Builder and in the HTML report that the software generates.

## New Support for Blocks

The Simulink Design Verifier software now supports models containing the following blocks:

- Combinatorial Logic
- Decrement Time To Zero
- Discrete Filter
- Fixed-Point State-Space
- Integer Delay
- Model blocks that reference other models
- Prelookup
- Relay

## Analyzing External Functions for Embedded MATLAB Function Blocks

If your model contains an Embedded MATLAB Function block that calls any external functions, the Simulink Design Verifier software can now accumulate coverage results for those functions.

## Enhanced Block Replacement Capability for Subsystems and Model Blocks

You can write your own replacement rules to replace subsystem or Model blocks that reference another model with the Simulink Design Verifier block replacement capability. The software replaces a subsystem or Model block with a different subsystem or with a built-in block as defined in the block replacement rules.

## New Implies Block

The new Implies block simplifies property specification. You can now specify conditions that produce a given response. For example, you can quickly create expressions indicating that pressing the brake pedal implies the cruise control must be inactive.

You can use the Implies block in any model, not just when running the Simulink Design Verifier software.

## New Property-Proving Examples and Demos

The Simulink Design Verifier block library includes four new example models that demonstrate how to define complex properties for property-proving analysis.

In addition, the following demo models are shipping with R2009a:

- `sldvdemo_sbr_design.mdl` — Finding property violations
- `sldvdemo_sbr_verification.mdl` — Proving that properties are valid
- `sldvdemo_thrustrvs_verification.mdl` — Analyzing model and properties to prove correctness or to identify counterexamples
- `sldvdemo_cruise_control_fxp_verification.mdl` — Proving properties for fixed-point arithmetic with block replacements
- `sldvdemo_cruise_control_verification.mdl` — Supporting model reference and verification subsystems

## sldvisactive Function

The `sldvisactive` function checks whether the Simulink Design Verifier software is actively translating the model. This function is called from the masked initialization of masked subsystems and other model or block callbacks to configure the model for Simulink Design Verifier analysis.

For example, the mask initialization of the Environment Controller block invokes the `sldvisactive` function to output the signal at its Sim port when you start analyzing a model that contains the block.

# R2008b

Version: 1.3

New Features

Bug Fixes

Compatibility Considerations

## Simulink Bus Signals and Bus Objects Support

Simulink Design Verifier now supports Simulink buses and bus objects:

- The root Inport and Outport blocks accept bus signals.
- Nonvirtual buses are propagated through the model elements.
- The test harness model reconstructs the bus signals from the underlying bus elements.

## Fixed-Point Data Support

Simulink Design Verifier blocks now support fixed-point parameters and inputs. These blocks include:

- Test Condition
- Test Objective
- Assumption
- Proof Objective

The `Slvd.Point` and `Sldv.Interval` constructors now accept fixed-point data.

## Generating Test Harness Model with Model Reference

To use this option, select **Reference input model in generated harness** in the **Design Verifier** > **Results** pane of the Configuration Parameters dialog box. Simulink Design Verifier software then uses model reference to run the original model from the test harness.

## Generating SystemTest TEST-File

To use this option, select **Save test harness as SystemTest TEST-File** in the **Design Verifier** > **Results** pane of the Configuration Parameters dialog box. The software creates a TEST-file instead of a test harness model. Using a TEST-file allows you to run the test cases in the SystemTest™ environment and configure the model coverage settings using the SystemTest software.

## Improved Search Algorithms

This release includes search algorithms for the following two modes that improve the performance and the quality of the results:

- Test case generation — The combined objectives options minimizes the number of test cases by generating cases that address more than one test objective.
- Property proving — Proving that model properties are valid.

## New Data File Format

When the Simulink Design Verifier software completes an analysis, it creates a data file. Now the data file supports bus input ports and includes more information about the analyzed model. For more information, see Simulink Design Verifier Data Files in the Simulink Design Verifier documentation.

## Compatibility Considerations

To convert an `sldvData` structure from the old format to the new format, use the `Sldv.DataUtils.convertToCurrentFormat` utility with the following syntax:

```
new_sldvData = Sldv.DataUtils.convertToCurrentFormat(model, old_sldvData)
```

The arguments used for this conversion comprise:

- `model` — The name of the model that was analyzed
- `old_sldvData` — The name of an `sldvData` structure created using the old (pre-R2008b) format

To convert an `sldvData` structure in the new format to the old format, use the `Sldv.DataUtils.convertToOldFormat` utility with the following syntax:

```
old_sldvData = Sldv.DataUtils.convertToOldFormat(model, new_sldvData)
```

The arguments used for this conversion comprise:

- `model` — The name of the model that was analyzed
- `new_sldvData` — The name of an `sldvData` structure created using the format that is new with R2008b

## New HTML Report

The HTML report that Simulink Design Verifier software generates has been enhanced. Now, when you select **Generate report of the results** in the **Design Verifier > Report** pane of the Configuration Parameters dialog box, the report generated has several improvements:

- The report generates faster and is easier to understand.
- The report can display expected outputs.
- The software generates a report that reflect the analysis settings (for example, test case generation vs. property proving).

## Blocks with No Input Ports Limitation

If a Simulink model has any blocks with no input ports, Simulink Design Verifier software cannot generate the test harness.

# R2008a+

**Version: 1.2.1**

**Bug Fixes**

# R2008a

Version: 1.2

New Features

Bug Fixes

## Embedded MATLAB Subset Support

This release provides support for the Embedded MATLAB Function block in the Simulink software and Embedded MATLAB functions in the Stateflow software. For more information, see Support Limitations for MATLAB for Code Generation in the Simulink Design Verifier User's Guide.

## Enhanced Support for Stateflow Truth Tables

Previous releases support only the Stateflow Classic truth tables. However, this release introduces support for Embedded MATLAB truth tables in the Stateflow software, which includes support for the Truth Table block. See Truth Table Functions for Decision-Making Logic in the Stateflow documentation for more information.

## New Simulink Design Verifier Data File Options

This release introduces new options on the **Design Verifier > Results** pane of the Configuration Parameters dialog box:

- **Include expected output values** — Simulates the model using the test case signals and includes the output values in the Simulink Design Verifier data file.
- **Randomize data that does not affect outcome** — Assigns random values instead of zeros to input signals that have no impact on test or proof objectives.

## New Test Suite Optimization Setting

In this release, the **Test suite optimization** parameter that appears on the **Design Verifier > Test Generation** pane of the Configuration Parameters dialog box includes a new setting: `Large model`. This test generation strategy is tailored to large, complex models that contain nonlinearities and many test objectives.

# R2007b+

**Version: 1.1.1**

**Bug Fixes**

# R2007b

Version: 1.1

New Features

Bug Fixes

## Fixed-Point Data Type Support

This release provides support for fixed-point data types. For more information, see Fixed-Point Support Limitations in the Simulink Design Verifier User's Guide.

# R2007a+

Version: 1.0

New Features

## Introducing the Simulink Design Verifier Software

The Simulink Design Verifier software extends the Simulink and Stateflow products with formal methods that help you confirm your model and chart behavior. The Simulink Design Verifier software performs a mathematically rigorous analysis of your model to identify all of its possible execution pathways. Subsequently, the software can

- **Generate Tests**

  The Simulink Design Verifier software can generate tests that satisfy your model's coverage objectives, including decision coverage, condition coverage, and modified condition/decision coverage (MC/DC). You can even customize the tests that it generates by using Simulink Design Verifier blocks that allow you to specify your own objectives and to constrain signal values. After the software completes its analysis, it produces a test harness model with a Signal Builder block that contains test signals. Simply simulate the test harness model to confirm that the test signals achieve your model's objectives.

- **Prove Properties**

  The Simulink Design Verifier software can prove that signals in your model attain particular values or ranges. Use Simulink Design Verifier blocks to specify values and ranges that you desire signals to attain, or to constrain the values of other signals. If the software disproves any of the values or ranges given the constraints you specify, it produces a test harness model with a Signal Builder block that contains signals comprising counterexamples. Simply simulate the test harness model to confirm that the counterexamples falsify your model's properties.

The Simulink Design Verifier software documents its analysis results in an HTML report. Also, it produces a data file containing the analysis results, which you can postprocess for your own analyses and reports.

In short, the Simulink Design Verifier software gives you confidence in the behavior of your Simulink models and Stateflow charts.

Version 1.0 of the Simulink Design Verifier software was released in a Web-downloadable form after R2007a.