Stateflow and Stateflow Coder Release Notes

The "Stateflow 5.0 and Stateflow Coder 5.0 Release Notes" on page 1-1 describe the changes introduced in the latest version of Stateflow and Stateflow Coder.

The following topics, as applicable, are discussed for each version documented in these Release Notes:

- "New Features" on page 1-2
- "Major Bug Fixes" on page 1-8
- "Upgrading from an Earlier Release" on page 1-9
- "Known Software and Documentation Problems" on page 1-10

If you are upgrading from a release earlier than Release 12.1, you should also see:

- "Stateflow 4.1 and Stateflow Coder 4.1 Release Notes" on page 2-1
- "Stateflow 4.0 and Stateflow Coder 4.0 Release Notes" on page 3-1

Printing the Release Notes

If you would like to print the Release Notes, you can link to a PDF version.

Contents

Stateflow 5.0 and Stateflow Coder 5.0 Release Notes

1

Stateflow 4.1 and Stateflow Coder 4.1 Release Notes

2

Stateflow 4.0 and Stateflow Coder 4.0 Release Notes

3

Stateflow 5.0 and Stateflow Coder 5.0 Release Notes

New Features

Note Stateflow 5.0 and Stateflow Coder 5.0 incorporate changes introduced in Versions 4.1.1 and 4.1.2 of those products, which were initially released in Web-downloadable form after Release 12.1 was released, but before Release 13. These Release Notes describe those changes, as well as changes introduced after Version 4.1.2.

New features for Stateflow 5.0 and Stateflow Coder 5.0 are as follows:

- “Fixed-Point Data”
- “Undo Operation in the Stateflow Diagram Editor”
- “The Stateflow API”
- “Matrix Support for Data to and from Simulink”
- “New Model Report Available in Stateflow”
- “Any Chart or Library Chart Can Export a Function”
- “ml Data Type”
- “Array and Matrix Support for ml Namespace Operator and Function Call”
- “Stateflow Allows up to 254 Events”
- “User Comments Included in Generated Code”
- “For-Loops Emitted in Generated Code”
- “Enhanced Stateflow Integration with Real-Time Workshop”
- “Stateflow Explorer Remembers Its Position and Size”
- “Trailing “F” Specifier for Single Precision Floating-Point Numbers”
- “Graphical Function Inlining In Generated Code”
- “Stateflow Code Generation Improvements”
- “Unnecessary Data Initialization Removed”
- “Simple If Statements Optimized”

If you are upgrading from a release earlier than Release 12.1, then you should also see “New Features” on page 2-2.

Fixed-Point Data

Stateflow now supports fixed-point data with the following features:

- Support for fixed point data with both binary point scaling and slope and bias scaling
- Support for fixed point operations included comparison, addition, subtraction, multiplication, and division
- Full coupling with input from and output to Simulink
- Detection of overflow for fixed-point and integer types
- Convenient notation for expressing fixed-point literal constants in action language
- Automatic type promotion rules that select the default result type of an operation for maximum computational efficiency
- Full control for overflow prevention and precision retention using special `:=` assignment operator

This feature is introduced in Stateflow 5.0 and is documented in "Fixed-Point Data" in the Stateflow documentation.

Undo Operation in the Stateflow Diagram Editor

You can undo and redo operations you perform in the Stateflow diagram editor. When you undo an operation in the Stateflow diagram editor, you reverse the last edit operation you performed. After you undo operations in the diagram editor, you can also redo them one-at-a-time. When you place your mouse cursor over the Undo or Redo buttons, the tooltip that appears indicates the nature of the operation to undo or redo.

This feature is introduced in Version 5.0 and is documented in "Undoing and Redoing Editor Operations" in the *Stateflow User's Guide* and Stateflow online help.

The Stateflow API

The new Stateflow API provides a programmatic access to Stateflow. Through individual MATLAB commands or scripts of commands, you can manipulate Stateflow objects (machines, charts, states, boxes, functions, notes, transitions, junctions, data, events, and targets) to perform actions previously only available through the Stateflow graphical interfaces. This includes constructing new diagrams from scratch and modifying existing ones.

The Stateflow API provides control for the following Stateflow actions:

- Search for and find existing Stateflow objects, including charts, at any level of containment
- Set the triggering behavior of new or existing charts
- Create new Stateflow objects within charts with complete control over their positioning, scope, containment, and decomposition (states)
- Set the properties for all Stateflow objects
- Copy and paste Stateflow objects from one location to another
- Delete existing Stateflow objects
- Modify the graphical appearance of all Stateflow objects
- Parse Stateflow charts
- Change the debugging (simulation) behavior for simulation targets
- Build Stateflow simulation targets
- Change the deployment properties for Stateflow non-simulation targets
- Build, compile, or generate code for other targets including RTW targets

This feature was introduced in Version 4.2 and is documented in “API User’s Guide” in the Stateflow documentation.

Matrix Support for Data to and from Simulink

Stateflow now supports two-dimensional matrices of any type for data with the scope **Data input from Simulink** or **Data output to Simulink**.

This feature is introduced in Version 5.0 and is documented in “Defining Input Data” and “Defining Output Data” in the Stateflow documentation.

New Model Report Available in Stateflow

A new model report is available in Stateflow (also in Simulink) for making comprehensive reports of Stateflow objects. Make this report in the Stateflow diagram editor by selecting **Print Details...** from the **File** menu.

This feature is introduced in Stateflow 5.0 and is documented in “Generating a Model Report in Stateflow” in the Stateflow documentation.

Any Chart or Library Chart Can Export a Function

Any chart or library chart can now export a graphical function and any other chart or library chart can call it as long as they are both (the caller and the called) accessible through the same main model.

This feature is introduced in Stateflow 5.0 and is documented in “Exporting Graphical Functions” in the Stateflow documentation.

ml Data Type

Stateflow supports a new data type called `ml`. Data of this type is typed internally with the MATLAB type `mxArray`. This means that you can assign (store) any type of data available in Stateflow to a data of type `ml`. This includes any type of data defined in Stateflow or returned from MATLAB with the `ml` namespace operator or `ml` function.

This feature is introduced in Version 5.0 and is documented in “ml Data Type” in the Stateflow documentation.

Array and Matrix Support for ml Namespace Operator and Function Call

Stateflow now supports vector arrays and n-dimensional matrices as arguments and return values for the `matlab` (`ml` for short) namespace operator and `matlab` (`ml` for short) function call. Before, only scalar data was supported.

This feature is introduced in Version 5.0 and is documented in “MATLAB Functions and Variables” in the Stateflow documentation.

Stateflow Allows up to 254 Events

Stateflow now handles up to 254 events per chart. The previous maximum was 127. Stateflow now throws an error if your chart has more than 254 events.

This feature is introduced in Version 5.0 and is documented in “Adding Events to the Data Dictionary” in the Stateflow documentation.

User Comments Included in Generated Code

Stateflow has the option for including comments you enter in the action language of Stateflow diagrams in generated code for rtw and custom targets. This option is enabled by default.

This feature was introduced in Version 4.2 and is documented in “Specifying Code Generation Options” in the Stateflow documentation.

For-Loops Emitted in Generated Code

The Stateflow Code generator detects and emits `for`-loops when applicable. In previous versions, Stateflow always emitted `while`-loops.

This feature was introduced in Version 4.2.

Enhanced Stateflow Integration with Real-Time Workshop

The code generated for Stateflow blocks is seamlessly integrated with Simulink, leading to more efficient and readable code.

This feature was introduced in Version 4.2.

Stateflow Explorer Remembers Its Position and Size

Stateflow Explorer now remembers its position and size across sessions with Stateflow.

This feature was introduced in the Web-downloadable Version 4.1.1.

Trailing “F” Specifier for Single Precision Floating-Point Numbers

Stateflow action language now recognizes a trailing “F” for specifying single precision floating-point numbers as in the action statement `x = 4.56F;`. In Stateflow action language, if a trailing “F” does not appear with a number, it is assumed to be double precision. Specifying single precision numbers allows you to save memory in generated code.

This feature was introduced in the Web-downloadable Version 4.1.1 and is documented in “Special Symbols” in the Stateflow documentation.

Graphical Function Inlining In Generated Code

Graphical functions with I/O can now be inlined in the generated code. You can specify inlining behavior (auto, force inline, force no inline) for every graphical function via its property dialog box. “auto” refers to a default strategy in which

Stateflow Coder itself decides when it is advantageous to inline a graphical function.

This feature was introduced in Version 4.1.1 and is documented in “Specifying Graphical Function Properties” in the Stateflow documentation.

Stateflow Code Generation Improvements

When possible, Simulink input and output data to a Stateflow chart are made local, reducing RAM size. Whenever possible, these local inputs are conditionally evaluated (via “expression folding”) resulting in execution speed improvements.

This feature was introduced in Version 4.1.1.

Unnecessary Data Initialization Removed

Unnecessary data initialization statements are now removed from the code generated for graphical functions.

This feature was introduced in Version 4.1.1.

Simple If Statements Optimized

A simple Boolean expression evaluation scheme is used to optimize `if` statements such as `if(1)`, `if(0)`, `if(ON==OFF)`, etc.

This feature was introduced in Version 4.1.1.

Major Bug Fixes

Stateflow 5.0 and Stateflow Coder 5.0 include several bug fixes made since Version 4.1. This section describes the particularly important Version 5.0 bug fixes.

If you are viewing these Release Notes in PDF form, please refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site and use the link provided.

If you are upgrading from a release earlier than Release 12.1, then you should also see “Major Bug Fixes” on page 2-4.

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving from previous versions of Stateflow to Version 5.0.

Warning on Loading Machines from Previous Stateflow Versions

Stateflow automatically upgrades all features introduced in previous versions of Stateflow to work with the current release of the MATLAB product family.

If you open a machine made with a previous version of Stateflow, you will see a warning message similar to the following:

Warning: An old Stateflow machine 'sf_car' is loaded.

> This machine was saved with an older Stateflow 3.0311061000001.
>
> Please save this machine again!

By saving the machine in the most recent version of Stateflow, it is automatically upgraded.

Error on Transition Action into Junction with Following Condition Action

Transition actions for transitions into junctions with condition actions that follow as part of a state-to-state path are flagged by an error indicating that the actual execution of these actions is in reverse order to the apparent segment order in the diagram.

This error was introduced in version 4.1.1. The error and its workaround is documented in “Code Generation Error Messages” in the Stateflow documentation.

Known Software and Documentation Problems

This section includes a link to a description of known software and documentation problems in Version 5.0. If you are viewing these Release Notes in PDF form, please refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site and use the link provided.

For a list of bugs reported in the previous releases that remain open, see “Known Software and Documentation Problems” on page 3-7.

If you are viewing these Release Notes in PDF form, please refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site and use the link provided.

Stateflow 4.1 and Stateflow Coder 4.1 Release Notes

New Features

This section introduces new features and enhancements added in Stateflow 4.1 and Stateflow Coder 4.1 (Release 12.1) since the Web-downloadable release of Stateflow 4.0.2 (Release 12).

For information about the features of Stateflow 4.1 added since the release of Stateflow 4.0 (Release 12) and prior versions, see “New Features” on page 3-2 or the “What's New” link from the Stateflow product page at `www.mathworks.com`.

Stateflow New Features

Smart Transitions

Stateflow charts now include the graphical innovation of smart transitions whose ends slide around the surfaces of states and junctions. When the source and/or destination objects are moved and resized in the Stateflow chart, these transitions use sliding and other behaviors to enable users to produce an aesthetically pleasing diagram.

This feature is fully documented in “Setting Smart Behavior in Transitions” in the Stateflow documentation and Stateflow online help.

Note Transitions are automatically created with smart behavior on the assumption that this behavior is almost always desirable. However, *self-loop* transitions, must be created without smart transition behavior. See the section “Creating Self-Loop Transitions” in online Help for instructions.

Search & Replace Tool Enhancements

The Stateflow Search & Replace tool has been modified with the following enhancements:

- Regular expression tokenized replacements

  Allows you to dynamically choose replacement text based on matched text.

- Case insensitivity/case preservation

  Replaces text with different sensitivities to the use of upper or lower case characters in the found text.

- Addition of a **Search in:** field

  Now you can select any Stateflow chart in your Simulink model or select the entire Stateflow machine.

These enhancements are fully documented in "Searching and Replacing in Charts" in the Stateflow documentation and Stateflow online help.

Stateflow Chart Notes

You can now enter annotations to your Stateflow charts that are similar to annotations in Simulink.

This feature is introduced in Stateflow 4.1 and is fully documented in "Entering Chart Notes" in the Stateflow documentation and Stateflow online help.

Model Coverage Tool

The Simulink Model Coverage tool has been modified to perform model coverage calculations for decisions and conditions of decision in the Stateflow machine and its charts, states, and transitions. This includes Condition and MCDC coverage.

This enhancement is introduced in Stateflow 4.1 and is fully documented in "Stateflow Chart Model Coverage"in the Stateflow documentation and Stateflow online help.

Stateflow Coder New Features

Single-Precision Constants in Code Generation

Code generation now emits single-precision constants with a trailing `F` to the C-compiler. This results in lower ROM size. For example, the action language statement `x = y + single(1.375);` now generates the code `x = y + 1.375F;`.

Major Bug Fixes

The following is a list of known bugs from Stateflow 4.0.2 (Release 12.0), which are fixed in Stateflow 4.1. For information about bugs fixed Stateflow 4.0 and Stateflow Coder 4.0, see “Major Bug Fixes” on page 2-4 or the “What’s New in Stateflow” on the MathWorks Web site.

Stateflow Bug Fixes

Editing Crossing Transitions out of Grouped Subcharts

Editing crossing transitions out of grouped subcharts caused model corruptions in Stateflow Versions 3.0 through 4.0.2.

Disabled and Restored Library Chart Links

Stateflow library chart links that are disabled and then restored caused model corruptions in Stateflow Versions 3.0 through 4.0.2.

Too Many Action Statements During Simulation

Stateflow Debugger caused an error during simulation when a state has more than 255 action statements in Stateflow Versions 1.0 through 4.0.2.

False State Inconsistency Run-time Error

Charts within a Simulink enabled subsystem produced a false state inconsistency run-time error when the subsystem was disabled in Stateflow Versions 2.0 through 4.0.3.

MATLAB Variables Improperly Overwritten

Selecting **Save final value to base workspace** in the properties dialog for a data item caused unrelated MATLAB variables in the workspace to be overwritten in Stateflow Versions 4.0 through 4.0.3.

Target Options Fields Overwritten

The **Custom Initialization** and **Custom Termination** fields in the **Target Options** dialog box were overwritten by empty strings in the data dictionary in Stateflow 4.0.3.

Transitions Assertions

Transitions containing a temporal trigger combined with other event triggers, such as `after(3E)|E2|E3`, caused assertion errors during parsing in Stateflow 4.0.3.

Build Failures with Custom Code

When including custom code, build failures occurred due to DOS shell command line length limitations because all user-defined directories on the MATLAB path were added to the include directory list in Stateflow 4.0.3.

Stateflow Coder Bug Fixes

Code Generation for Default Transitions in Parallel States

States with Parallel decomposition with default transition paths generated incorrect code in Stateflow Versions 3.0 through 4.0.2.

Code Generation for Double-Precision Whole Numbers

Double-precision whole numbers were incorrectly emitted as integers (e.g., 5 instead of 5.0) in code generation in Stateflow Versions 3.0 through 4.0.2.

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving from Stateflow 4.0.2 (Release 12.0) or earlier to Stateflow 4.1 (Release 12.1).

Transition Actions into Junctions Disallowed

Transition actions are now permitted only on transitions that terminate on states. For the following reasons, transition actions are no longer permitted on transitions that terminate on junctions; condition actions should be used instead:

- The semantics of transition actions for transitions into junctions are complex and easily misunderstood and misused.
- The complex semantics of these transition actions also result in generated code that is inefficient. Eliminating these transition actions not only simplifies Stateflow diagrams but also results in generated code that is much more efficient.

If your current model has these transition actions, they are flagged with an error. In most cases, replacing these transition actions with condition actions gives identical chart behavior.

Warning on Loading Models from Previous Stateflow Versions

Stateflow 4.1 automatically upgrades all features introduced in previous versions of Stateflow to work with Release 12 of the MATLAB product family.

If you open a machine (model) made with a previous version of Stateflow, you will see a warning message similar to the following:

```
Warning: An old Stateflow machine 'sf_car' is loaded.

    This machine was saved with an older Stateflow 3.0311061000001.

    Please save this machine again!
```

By saving the machine in the most recent version of Stateflow, it is automatically upgraded.

Stateflow 4.0 and Stateflow Coder 4.0 Release Notes

New Features

This section introduces the new features and enhancements added in Stateflow 4.0 and Stateflow Coder 4.0 since Stateflow 2.0 and Stateflow Coder 2.0.

Stateflow 4.0 New Features

Stateflow Works with MATLAB Release 12

Stateflow 4.0 upgrades all features introduced in Stateflow 3.0 to work with Release 12 of the MATLAB product family.

Note The following features were introduced in Stateflow 3.0.

Temporal Logic

You can now use temporal conditions (before, after, at, every time) to determine the activation of transitions and duration of state activation. Temporal logic provides a simple paradigm for event scheduling and allows your Stateflow model to express clearly and simply the time-dependent behavior of a system.

This feature is fully documented in “Temporal Logic Operators” in the Stateflow documentation and Stateflow online help.

Subcharts

Stateflow now allows you to create charts within charts. A chart that is embedded in another chart is called a subchart. A subchart can contain anything a top-level chart can, including other subcharts. In fact, you can nest subcharts to any level. A subchart appears as a labeled block in the chart that contains it. You can create transitions among objects residing in different subcharts existing at the same level or at different levels. A transition that crosses subchart boundaries in this fashion is called a supertransition.

Subcharts enable you to reduce a complex chart to a set of simpler, hierarchically organized diagrams. This makes the chart easier to understand and maintain, without changing the semantics of the chart in any way.

Stateflow ignores subchart boundaries when simulating and generating code from Stateflow models.

This feature is fully documented in “Working with Subcharts” in the Stateflow documentation and Stateflow online help.

Graphical Functions

A graphical function is a function defined by a flow graph. Graphical functions are similar to textual functions, such as MATLAB and C functions. Like textual functions, graphical functions can accept arguments and return results. You invoke graphical functions in transition and state actions in the same way you invoke MATLAB and C functions. Unlike C and MATLAB functions, however, graphical functions are full-fledged Stateflow objects. You use the Stateflow editor to create them and they reside in your Stateflow model along with the diagrams that invoke them. This makes graphical functions easier to create, access, and manage than textual functions, whose creation requires external tools and whose definitions reside separately from the model.

This feature is fully documented in “Creating a Graphical Function” in the Stateflow documentation and Stateflow online help.

Symbol Autocreation Wizard

The Symbol Autocreation Wizard helps you to add missing data and events to your Stateflow charts. When you parse the diagram or run the simulation, this wizard detects whenever data and events have not been previously defined in the Stateflow Explorer. The wizard then opens automatically and heuristically recommends attributes for the unresolved data or events to help you to quickly define these symbols.

This feature is fully documented in “Symbol Autocreation Wizard” in the Stateflow documentation and Stateflow online help.

Command Toolbar

The Stateflow editor now contains a toolbar containing buttons for the most commonly used editing and simulation commands in Stateflow. The toolbar saves searching through menus for these commands.

Navigation Toolbar

This toolbar contains buttons for navigating a chart hierarchy.

Straight Line Transitions

You can now create straight lines between junctions. Transitions that are almost straight are automatically snapped straight during edit time. The snap-to-grid functionality helps align connected junctions vertically and horizontally.

Workspace-Based Data

Data items can now be initialized from identically named variables in the MATLAB workspace and/or copied back to the workspace at the end of a simulation. Workspace-initialized constants consume no memory in generated code.

This feature is fully documented in “Setting Data Properties” in the Stateflow documentation and Stateflow online help.

Explorer Copy Properties

Stateflow Explorer now allows you to pick up properties from one data/event/target item and apply them to another data/event/target item or a group of items. This speeds up the process of creating diagrams that have objects with similar sets of properties.

This feature is fully documented in “Transferring Object Properties” in the Stateflow documentation and Stateflow online help.

Library Link Icons

An arrow distinguishes icons of library links from those of actual charts in the Stateflow Explorer. Clicking a library link icon opens the library chart in the Stateflow Editor. Stateflow 4.0 requires Release 12, including Simulink 4.0.

Stateflow Coder 4.1 New Features

Improved Code Generation

The Stateflow Coder 4.0 code generation has been significantly improved:

- The code looks hand written.
- ROM and RAM size rivals hand-written code.
- Code generation is faster.

Temporal Logic

Note The temporal logic feature was introduced in Stateflow Coder 3.0.

You can now use temporal conditions (before, after, at, every time) to determine the activation of transitions and duration of state activation. Temporal logic provides a simple paradigm for event scheduling.

Temporal conditions allow your Stateflow model to express clearly and simply the time-dependent behavior of a system.

Stateflow Coder 4.0 supports the new temporal logic feature in Stateflow, which allows it to be extremely efficient with memory usage in the generated code. When you use temporal operators, Stateflow manages the various counters needed for event scheduling. Then, with behind-the-scenes optimizations, Stateflow Coder 4.0 can allocate memory more economically.

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving from previous versions of Stateflow to Version 4.0.

Warning on Loading Models from Previous Stateflow Versions

Stateflow 4.0 automatically upgrades all features introduced in previous versions of Stateflow to work with Release 12 of the MATLAB product family.

If you open a machine (model) made with a previous version of Stateflow, you will see a warning message similar to the following:

```
Warning: An old Stateflow machine 'sf_car' is loaded.
          This machine was saved with an older Stateflow 3.0311061000001.
          Please save this machine again!
```

By saving the machine in the most recent version of Stateflow, it is automatically upgraded.

Known Software and Documentation Problems

This section updates the Stateflow 4.0 documentation set, reflecting known Stateflow 4.0 software and documentation problems.

Many Open Windows Can Cause a Crash or Hang (Windows 98/Me)

On Microsoft Windows 98/Me platforms, if you keep approximately 12 Stateflow windows open, Stateflow may crash or hang. Note that the actual number of open windows that may cause this problem depends on the resources currently in use by other components and applications.

Calling MATLAB Functions from Stateflow Charts

Calling MATLAB functions from Stateflow charts using either the MATLAB namespace operator `ml.` or by using the function form `ml()` with an argument whose datatype is not `double` causes erroneous results. The workaround is to cast the argument as a `double`. For example

```
ml.myfun(x,y,1)
```

where `x` and `y` are Stateflow data items whose data types are `uint8` and `int32` and the third argument is an integer has to be rewritten as

```
ml.myfun(double(x),double(y),double(1))
```

inserting casts around the nondouble arguments.