Link for Code Composer Studio™ Development Tools Release Notes

Contents

Summary by Version

This table provides quick access to what's new in each version. For clarification, see "About Release Notes" on page 1.

Version (Release)	New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Latest Version V3.0 (R2007a)	Yes Details	Yes Summary	Bug Reports	Printable Release Notes: PDF Current product documentation
Latest Version V2.1 (R2006b)	Yes Details	Yes Summary	Bug Reports	Printable Release Notes: PDF
Latest Version V2.0 (R2006a+)	Yes Details	Yes Summary	Bug Reports	Printable Release Notes: PDF
V1.5 (R2006a)	Yes Details	Yes Summary	Bug Reports	Printable Release Notes: PDF
V1.4.2 (R14SP3)	Yes Details	Yes Summary	Bug Reports	Printable Release Notes: PDF

About Release Notes

Use release notes when upgrading to a newer version to learn about new features and changes, and the potential impact on your existing files and practices. Release notes are also beneficial if you use or support multiple versions.

If you are not upgrading from the most recent previous version, review release notes for all interim versions, not just for the version you are installing. For example, when upgrading from V1.0 to V1.2, review the New Features and Changes, Version Compatibility Considerations, and Bug Reports for V1.1 and V1.2.

New Features and Changes

These include

- New functionality
- Changes to existing functionality
- Changes to system requirements (complete system requirements for the current version are at the MathWorks Web site)
- Any version compatibility considerations associated with each new feature or change

Version Compatibility Considerations

When a new feature or change introduces a known incompatibility between versions, its description includes a **Compatibility Considerations** subsection that details the impact. For a list of all new features and changes that have compatibility impact, see the "Compatibility Summary for Link for Code Composer Studio Development Tools" on page 20.

Compatibility issues that become known after the product has been released are added to Bug Reports at the MathWorks Web site. Because bug fixes can sometimes result in incompatibilities, also review fixed bugs in Bug Reports for any compatibility impact.

Fixed Bugs and Known Problems

MathWorks Bug Reports is a user-searchable database of known problems, workarounds, and fixes. The MathWorks updates the Bug Reports database as new problems and resolutions become known, so check it as needed for the latest information.

Access Bug Reports at the MathWorks Web site using your MathWorks Account. If you are not logged in to your MathWorks Account when you link

to Bug Reports, you are prompted to log in or create an account. You then can view bug fixes and known problems for R14SP2 and more recent releases.

The Bug Reports database was introduced for R14SP2 and does not include information for prior releases. You can access a list of bug fixes made in prior versions via the links in the summary table.

Related Documentation at Web Site

Printable Release Notes (PDF). You can print release notes from the PDF version, located at the MathWorks Web site. The PDF version does not support links to other documents or to the Web site, such as to Bug Reports. Use the browser-based version of release notes for access to all information.

Product Documentation. At the MathWorks Web site, you can access complete product documentation for the current version and some previous versions, as noted in the summary table.

Version 3.0 (R2007a) Link for Code Composer Studio Development Tools

This table summarizes what's new in V3.0 (R2007a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports	Printable Release Notes: PDF V3.0 product documentation

New features and changes introduced in this version are described here.

Project Generator Enables Project Generation for C2000, C5000, and C6000 Processors from Simulink Models

A new component, Project Generator, enables you to generate projects into Code Composer Studio from Simulink models. Link for CCS now comprises the following components:

- Automation Interface component that provides all the earlier product features, such as project debugging, data manipulation, and function call. For more information, refer toGetting Started with Automation Interface
- Project Generator component that provides the new project generation features. Refer to Introducing Project Generator for more information.

Note Project Generator component use requires Simulink and Real-Time Workshop.

Processor in the Loop Cosimulation with C2000, C5000, and C6000 Processors

Use processor in the loop (PIL) cosimulation techniques to verify the performance of your generated code running on an instruction set simulator or hardware target. For further information, refer to Using Processor-in-the-Loop.

Note Real-Time Workshop Embedded Coder Required to use the PIL feature

---

Blocks for Interrupt Handling and Accessing Memory in Projects Generated by Project Generator

A new library, `ccslinklib`, contains blocks that enable you to use interrupt handling and access the memory in the projects you generate with Project Generator. The blocks cover the C280x, C281x, C5000, and C6000 processor families. To open the new library, enter:

```
ccslinklib
```

at the MATLAB command prompt.

`ccslinklib` includes the following block libraries:

Library Name	Contents
C280x DSP Chip Support	Blocks for memory operations and asynchronous scheduling in generated code
C281x DSP Chip Support	Blocks for memory operations and asynchronous scheduling in generated code
C5000 DSP Chip Support	Blocks for memory operations and asynchronous scheduling in generated code
C6000 DSP Chip Support	Blocks for memory operations and asynchronous scheduling in generated code
Target Preferences	Custom target preferences block for configuring the processor and mapping the processor internal and external memory

Real-Time Execution Profiling for Projects Generated with Project Generator and Running on Target Processors

The profiler includes a set of utilities for recording, uploading, and analyzing execution profile data for synchronous and asynchronous tasks. It generates a display that shows when tasks are activated, preempted, resumed, and completed. In addition, the profiler generates an HTML report with statistical information on each synchronous task. Refer to the demo Real-Time Task Execution Profiling in the online Help for an introduction. For more information, refer to Real-Time Execution Profiling.

Demos That Introduce PIL Cosimulation and Real-Time Execution Profiling

Two new demo programs in the online Help system show you how to use the new PIL and execution profiling capabilities. Access these demos from the **Demos** tab in the online Help.

- Comparing Simulation and Target Implementation with Processor-in-the-Loop (PIL)
- Real-Time Task Execution Profiling

Adding a New Processor

In the custom Target Preferences block in the Target Preferences library, a new feature lets you add a new processor to support. On the **Board Info** pane of the block, **Add new** opens a dialog box where you configure your new processor for support and code generation. You set the processor name and class, the internal memory mapping, the clock speed, the cache, and more.

General Notes for the Release

The compatibility considerations from the Version 2.0 release (R2006a+) remain applicable.

Compatibility Considerations

The following issues reflect changes to the support provided by Link for CCS:

- The product no longer supports the RTDX-based demos on the C54x simulator and on C6701 processor-based hardware. These include:
  - Real Time Data Exchange Tutorial
  - LMS Adaptive Filtering
  - Transferring Data Between Simulink and a Target Application
- When you use the C54x simulator, the `disp` method does not display the correct processor type for your target. The property `revfamily` is set to an improper processor revision.

- When you use the C6713 Device Cycle simulator, the `disp` method does not display the correct processor subfamily identifier. The property `subfamily` is set to an improper processor subfamily value of `100` instead of `103`.
- If you use the C6713 processor with a USB connection, RTDX communications does not work when you launch Code Composer Studio from Link for CCS. To work around this problem, launch CCS manually from Microsoft Windows®.
- With the C2808 processor as your target, you cannot enable more than one RTDX read channel from the host-side application. In some cases, enabling one channel from the host disables a previously enabled channel. To work around this problem, enable the RTDX channels programmatically in the C2808 application code.
- The product no longer support the Debugging a Target Application demo (`ccsdebugdemo`) on TMS470R2x hardware.
- When you launch CCS by issuing the command `cc=ccsdsp` at the MATLAB prompt, and then you use any DSP/BIOS plug-in or the `profile` function, you encounter an error that says `vbd.dll` could not be loaded. Work around this by opening CCS manually before you use `ccsdsp`.
- When you use `open` to open a project in a CCS IDE window and the project is already open in another CCS IDE window, Link for CCS returns a time-out error in MATLAB. You can avoid this error by:
  - Opening the project in a new CCS IDE window manually.
  - Closing the existing project in the other IDE before you open the new instance of your project.

Version 2.1 (R2006b) Link for Code Composer Studio Development Tools

This table summarizes what's new in V2.1 (R2006b):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports	Printable Release Notes: PDF V2.1 product documentation

New features and changes introduced in this version are described here.

Extended Support to Two Code Composer Studio Versions

- Added CCS 3.2 for the C64x+ and C67x+ processor families
- Continuing support for CCS 3.1 (C6000, C5000, C2000, Rxx, OMAP processor families)

Additional or continuing compatibility considerations are listed in "Compatibility Considerations" on page 13.

New Function Call Block in New Demo -- Comparing Simulation and Target Implementation with Function Call

To complement the function objects and `createobj` in the product, this release includes a new demo that introduces a demonstration function call block — "Comparing Simulation And Target Implementation Through Function Call." You use the block in a Simulink model to identify and run a function on your target. The ability to communicate between Simulink and your target lets you run models that exercise functions and algorithms that reside on your target, and move data back and forth between MATLAB, your Simulink model, and your target-based process. Refer to the demos for Link for Code Composer Studio in MATLAB to find and run the new demo.

New Demo Featuring Debug Component -- Comparing Simulation and Target Implementation with Debug Methods

A Simulink-based demo featuring the Debug component that shows how you use the debug methods to work with targets.

Demo Support for New Processors

C64x+ and C67x+ processor families supported by all demos.

General Notes for the Release

The compatibility considerations from the Version 2.0 release (R2006a+) remain applicable as shown here.

Compatibility Considerations

The following issues reflect changes to the support provided by Link for CCS:

- The product no longer supports the RTDX-based demos on the C54x simulator and on C6701 processor-based hardware. These include
  - Real Time Data Exchange Tutorial
  - LMS Adaptive Filtering

  - Transferring Data Between Simulink and a Target Application

- When you use the C54x simulator, the `disp` method does not display the correct processor type for your target. The property `revfamily` is set to an improper processor revision.
- When you use the C6713 Device Cycle simulator, the `disp` method does not display the correct processor subfamily identifier. The property `subfamily` is set to an improper processor subfamily value of `100` instead of `103`.
- If you use the C6713 processor with a USB connection, RTDX communications does not work when you launch Code Composer Studio from Link for CCS. To work around this problem, launch CCS manually from Microsoft Windows®.
- With the C2808 processor as your target, you cannot enable more than one RTDX read channel from the host-side application. In some cases, enabling one channel from the host disables a previously enabled channel. To work around this problem, enable the RTDX channels programmatically in the C2808 application code.
- The product no longer support the Debugging a Target Application demo (`ccsdebugdemo`) on TMS470R2x hardware.
- When you launch CCS by issuing the command `cc=ccsdsp` at the MATLAB prompt, and then you use any DSP/BIOS plug-in or the `profile` function, you encounter an error that says `vbd.dll` could not be loaded. Work around this by opening CCS manually before you use `ccsdsp`.
- When you use `open` to open a project in a CCS IDE window and the project is already open in another CCS IDE window, Link for CCS returns a time-out error in MATLAB. You can avoid this error by
  - Opening the project in a new CCS IDE window manually.
  - Closing the existing project in the other IDE before you open the new instance of your project.

Version 2.0 (R2006a+) Link for Code Composer Studio Development Tools

This table summarizes what's new in V2.0 (R2006a+):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports	Printable Release Notes: PDF V2.0 product documentation

New features and changes introduced in this version are described here.

New architecture for Link for Code Composer Studio Development Tools

This release of the product uses a new internal structure to interact with Code Composer Studio™ (CCS). Link for CCS now installs components (also called plug-ins) in CCS IDE that enable the links and MATLAB® commands to run in the IDE process. The internal changes provide improved performance when you read or write data to your target. The interface you see in MATLAB has not changed, only the implementation of the software. All of your scripts and existing code work as before in earlier versions.

Installation Notes

Link for CCS registers a plug-in tool for Code Composer Studio (CCS). After you install Link for CCS, start CCS. CCS will detect the plug-in and prompt you that "New components were detected." Click **Yes** to enable tools for all compatible CCS releases.

The New Components dialog appears only once. If you do not accept the new components, you cannot use Link for CCS without reregistering the new components and then using the Component Manager in CCS to enable them.

Compatibility Considerations

The following changes result from the new architecture and the improved performance.

- Changing the architecture affects the way MATLAB connects to the CCS IDE. As a result, after you create `ccsdsp` objects that link to the CCS IDE, you should not close the IDE associated with the `ccsdsp` object without clearing the handle (your `ccsdsp` object) from your MATLAB workspace. Closing the IDE without clearing the handle leaves invalid handles. Use this command to clear the specified handles:

  ```
  clear(handlename)
  ```

  If you do not clear the handles, you get the following error when you try to use an invalid handle to the missing IDE instance:

  ```
  ??? CCS was closed independent of MATLAB. All existing handles are invalid.

  Please clear all CCS handles and recreate them.
  ```

  To proceed from here, you must clear the handles and then recreate your `ccsdsp` object.

- If you use the `load` and `build` functions from Link for CCS in a MATLAB script, you might find that your script no longer builds and loads projects properly. Now MATLAB returns a message about an unspecified error. The error can occur because the load and build operations issued by your script are conflicting with the **Load Program After Build** option in CCS. If you select **Load Program After Build** in CCS, CCS automatically loads programs after building them. When your script issues the build and load requests, they conflict with the CCS build and load and the process fails.

To prevent your scripts from failing in this case, go to **Option > Customize > Program/Project Load** in CCS IDE and clear the **Load Program After Build** option before you run your script.

Timeout Argument Added to insert and delete Methods

Both `insert` and `delete` now offer an input argument, `timeout`, that lets you specify explicitly the time the insert or delete operations wait while CCS processes the commands. You specify the time-out period in seconds. Please note the next new feature below that relates to this one. Note also that the `timeout` input option applies to all debug point options—break points, probe points, and profile points (called *debug* points).

insert and delete Now Blocking Operations

When you use `insert` or `delete` to add or remove a debug point (a break point, probe point, or profile point) in CCS, the process now operates in *blocking* mode. Blocking mode means that after you issue a `insert` or `delete` command in MATLAB, the MATLAB method retains control until your requested process reaches completion (it adds or deletes the debug point) or the process exceeds the time-out period as specified by either the optional `timeout` input argument or the default time-out value in `cc.timeout`. This represents a change to the earlier `insert` and `delete` operations.

Compatibility Considerations

The following changes result from the new `insert` and `delete` operation:

- Making `insert` and `delete` blocking means unsuccessful insert or delete operations return errors when the operation fails. In earlier releases, `insert` and `delete` did not return errors related to the success or failure of the operation. When you use the `line` input argument to insert a debug point on a specified line, `line` must represent a valid line. If `line` does not specify a valid line, `insert` returns an error and does not insert the debug point.
- `delete(...,'all',...)` removes all valid break points in the project code. It does not remove probe points.

RTDX Simulation Blocks in New Demo RTDX Simulation

A new demo, Transferring Data Between Simulink and a Target Application (`rtdxsumdiffdemo`), introduces two new blocks for working with RTDX in Simulink models. The new blocks, RTDX Read and RTDX Write, enable you to simulate RTDX communications with an application running on a hardware platform.

These new blocks are for demonstration purposes only. And they only apply in Simulink® models on the host. They do not generate code for target applications.

build Supports Output Arguments

`build` now supports a pair of output arguments that return information about the build process. This syntax

```
[result,numwarns]=build(...)
```

returns two output values that report the results of the build operation. For a successful build, the output arguments are the following:

- `result` equals `1` for the build
- `numwarns` reports the number of build warnings that occurred during the build.

When the build is not successful, `build` displays an error and a message that contains the build error string in the MATLAB Command Window.

createobj Improved to Work With All Functions

`createobj` now returns complete `function` objects for library functions as well as C functions. In earlier releases, `createobj` could not return `function` objects from library functions that contained all necessary information to run the function.

Compatibility Considerations

The following changes result from the upgraded `createobj`.

- When you create a `function` object for a library function, you do not need to perform the `declare` operation to specify the function arguments. If you have scripts that use `createobj` followed by `declare`, remove the `declare` command from the scripts.
- If you are using any C28xx target, you cannot use the `declare` function after you create your `function` object with `createobj`. Using the `declare` causes an error. If you have scripts that use `createobj` followed by `declare`, remove the `declare` command from the scripts.
- Before you use a project built with any release of CCS before V3.0, rebuild the project in CCS 3.1.
- Always use the `Full Symbolic Debug` setting for the **Generate Debug Info** build option for your project in CCS.

CCS 3.1 Compatibility Updates

This release of Link for CCS continues to support Code Composer Studio 3.1 with additional compatibility considerations as listed here.

Compatibility Considerations

The following issues reflect changes to the support provided by Link for CCS:

- The product no longer supports the RTDX-based demos on the C54x simulator and on C6701 processor-based hardware. These include
  - Real Time Data Exchange Tutorial
  - LMS Adaptive Filtering
  - Transferring Data Between Simulink and a Target Application
- When you use the C54x simulator, the `disp` method does not display the correct processor type for your target. The property `revfamily` is set to an improper processor revision.
- When you use the C6713 Device Cycle simulator, the `disp` method does not display the correct processor subfamily identifier. The property `subfamily` is set to an improper processor subfamily value of `100` instead of `103`.
- If you use the C6713 processor with a USB connection, RTDX communications does not work when you launch Code Composer Studio

from Link for CCS. To work around this problem, launch CCS manually from Microsoft Windows®.

- With the C2808 processor as your target, you cannot enable more than one RTDX read channel from the host-side application. In some cases, enabling one channel from the host disables a previously enabled channel. To work around this problem, enable the RTDX channels programmatically in the C2808 application code.
- The product no longer support the Debugging a Target Application demo (`ccsdebugdemo`) on TMS470R2x hardware.
- When you launch CCS by issuing the command `cc=ccsdsp` at the MATLAB prompt, and then you use any DSP/BIOS plug-in or the `profile` function, you encounter an error that says `vbd.dll` could not be loaded. Work around this by opening CCS manually before you use `ccsdsp`.
- When you use `open` to open a project in a CCS IDE window and the project is already open in another CCS IDE window, Link for CCS returns a time-out error in MATLAB. You can avoid this error by
  - Opening the project in a new CCS IDE window manually.
  - Closing the existing project in the other IDE before you open the new instance of your project.

Version 1.5 (R2006a) Link for Code Composer Studio Development Tools

This table summarizes what's new in V1.5 (R2006a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports	Printable Release Notes: PDF V1.5 product documentation

New features and changes introduced in this version are described here:

CCS 3.1 Compatibility

Link for CCS supports CCS 3.1 exclusively. To use V1.5 of Link for CCS, you must upgrade your CCS installation to Version 3.1.

Compatibility Considerations

Link for CCS V1.5 does not support operation with any version of Code Composer Studio before 3.1.

Version 1.4.2 (R14SP3) Link for Code Composer Studio Development Tools

This table summarizes what's new in V1.4.2 (R14SP3):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes—Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports	Printable Release Notes: PDF

New features and changes introduced in this version are described here:

Improved Performance When Creating Function Objects

For this release, we changed the `createobj` method to reduce the time it takes to create new objects. When you create an object, `createobj` no longer saves any registers while creating the object. Earlier versions stored default registers from a list named `saveregs`. That list is now empty by default. As a result, the process for constructing new objects is faster. You can still add new registers to the saved registers list with `addregister`, and you can delete saved registers from the list with `deleteregister`. You must add the registers to the saved registers list immediately after you create the object.

Compatibility Considerations

Since we no longer save any default registers, the `cleanup` method does not perform any function unless you added registers to `saveregs`.

Compatibility Summary for Link for Code Composer Studio Development Tools

This table summarizes new features and changes that might cause incompatibilities when you upgrade from an earlier version, or when you use files on multiple versions. Details are provided in the description of the new feature or change.

Version (Release)	New Features and Changes with Version Compatibility Impact
Latest Version V3.0 (R2007a)	See the **Compatibility Considerations** for this change "General Notes for the Release" on page 7
Latest Version V2.1 (R2006b)	See the **Compatibility Considerations** for this change "General Notes for the Release" on page 10
Latest Version V2.0 (R2006a+)	See the **Compatibility Considerations** subheading for this new feature or change: • "New architecture for Link for Code Composer Studio Development Tools" on page 12 • "insert and delete Now Blocking Operations" on page 14 • "createobj Improved to Work With All Functions" on page 15 • "CCS 3.1 Compatibility Updates" on page 16

Version (Release)	New Features and Changes with Version Compatibility Impact
Latest Version *V1.5 (R2006a)*	See the **Compatibility Considerations** subheading for this new feature or change: • "CCS 3.1 Compatibility" on page 18
V1.4.2 (R14SP3)	See the **Compatibility Considerations** subheading for this new feature or change: • "Improved Performance When Creating Function Objects" on page 19