
# µCUSB Device™ Stack for CrossCore® Embedded Studio Rel.2.0.0 Release Notes


http://www.analog.com
processor.tools.support@analog.com

# Contents

# 1 µCUSB Device™ Stack for CrossCore® Embedded Studio Rel. 2.0.0 Release Notes

**Supported Processors**

Additional support is provided, in all of the µC/USB Device™products, for ADSP-SC589 and ADSP-BF707. ADSP-SC589 has two USB ports, USB0 (USB OTG) and USB1 (USB HS). Both USB0 and USB1 ports support the USB device mode.

**Tools**

µC/USB Device™ Stack for CrossCore® Embedded Studio 2.0.0 requires the installation of CrossCore® Embedded Studio 2.0.0. USB device driver available in CrossCore® Embedded Studio 2.0.0. is updated to support the ADSP-SC589 processor in CrossCore® Embedded Studio 2.0.0.

**New µC/USB Device™ Stack and updated USB device driver**

µC/USB Device™ Stack CrossCore® Embedded Studio 2.0.0 is updated to the latest version V4. 04.01.

µC\USB Device Vendor Class product now provides the corrected WinUSB driver installer (WinUSB_composite.inf) for the Windows host PC for the USB Composite Devices. This driver will automatically get installed during the installation of the µC\USB Device Vendor Class product.

µC/USB Device™ Stack currently doesn't supports the associated interfaces, so now when you use the Micrium Echo Demo Application with both the Echo Sync and the Echo Async options enabled, two different "ADI Vendor Specific USB Composite Devices" will get installed. Please refer to the readme provided with the µC\USB Device Vendor Class demo examples for more information.

USB Device driver is updated to add the support of ADSP-SC589. Couple of bugs related to detection of USB MSC device were also fixed.

**Knows Issues**

- The USB stack supports the connection to any of the USB0 (USB OTG) or USB1 (USB HS) ports provided on the ADSP-SC589 EZ-Board, but connection to both the ports at the same time is not supported. However, you can switch between USB0(USB OTG) and USB1(USB HS) ports while your application is running.

- Demo application provided in the USB Device CDC_ACM Class from µC/USB Device™ Stack version 4.04.01 is changed. When you upgrade from a lower release version to 2.0.0 release, the application files present in the existing demo examples will not be overwritten with the updated ones available with the 2.0.0 installation. Hence it is recommended to first remove the existing µC-USBD CDC-ACM add-in from your demo example and then add the µC-USBD CDC-ACM 2.0.0 add-in.

# 2 µCUSB Device™ Stack for CrossCore® Embedded Studio Rel. 1.1.0 (Beta) Release Notes

**Supported Processors**

Support is provided, in all of the µC/USB Device™products, for ADSP-BF52x, ADSP-BF54x, ADSP-BF60x. Support for ADSP-BF533 and ADSP-BF561 is only provided in µC/USB Device™ Class Vendor for CrossCore® Embedded Studio.

**Hardware Requirements**

The ADSP-BF533 and ADSP-BF561 implementation requires the addition of the Blackfin USB-LAN EZ-Extender rev 1.1 (part number: ADDS-USBLAN-EZEXT) or above.

Software Requirements

**Tools**

µC/USB Device™ Stack for CrossCore® Embedded Studio 1.1.0 requires the installation of CrossCore® Embedded Studio 1.1.0 or later. CCES 1.0.3 can also be used in conjunction with a USB driver patch.

**What's New**

**New classes**

New to µC/USB Device™ Stack for CrossCore® Embedded Studio 1.1.0:

**Improvements**

All internal buffers no matter how large or small are allocated from a memory pool, and aligned according to the USBD_CFG_BUF_ALIGN_OCTETS macro defined in the µC/Application horizontal tab of the µC/USB Device horizontal tab in the System Configuration Editor. The memory pool size is assigned in the µC/LIB tab.

**Software Products**

µC/USB Device™ Stack for CrossCore® Embedded Studio requires support for uC/OS-III. All of the examples ship with a pre-configured version of µC/OS-III™ Real Time Kernel for CrossCore® Embedded Studio 1.0.0, provided as a binary library that is linked into the examples. The exact configuration of the RTOS is provided in the example documentation.

When multiple µC/USB Device Products are used in the same project, only µC/USB Device products with the same revision number can be used. Please note all µC/USB Device Class products require the µC/USB Device Core product.

The RTOS library can be replaced with either release 1.0.0 or 1.0.1 of µC/OS-III™ Real Time Kernel for CrossCore® Embedded Studio, as detailed in the example documentation.

**Board Support Packages (BSP)**

The ADSP-BF533 and ADSP-BF561 implementation requires the installation of the Blackfin USB-LAN EZ-Extender BSP revision 1.0.1 or higher.

Examples

Examples are available for ADSP-BF526, ADSP-BF527, ADSP-BF548, ADSP-BF533 and ADSP-BF561. In the case of ADSP-BF533 and ADSP-BF561, examples are provided with the µC/USB Device™ Class Vendor product. Examples can be built for both Debug and Release configurations.

All examples now copy to the workspace, when opened from the Examples Browser. Please note that, due to the way in which the CrossCore® Embedded Studio IDDE generates the path, it is common to exceed the Windows 255 character path limit. You are strongly advised to use a workspace other than the default for CCES, and with as short a path as possible, (e.g. C:\WORKSPACE-1\).

The µC/USB Device™ Class Vendor product examples have been extended to add support for the Micrium supplied example (StressTest). The example can either be built as the original (release 1.0.0) bulk loopback example or as the Micrium stress test example. An additional host application is supplied to use with the stress test example. Please refer to the example documentation for more information.

# 3 µCUSB Device™ Stack for CrossCore® Embedded Studio Rel. 1.0.0 Release Notes

What is µC/USB Device™ Stack for CrossCore® Embedded Studio

C/USB Device™ Stack for CrossCore® Embedded Studio is the result of a partnership between Analog Devices and Micriµm to provide a user-friendly programming environment for embedded applications that require USB device functionality. It is designed for embedded systems with USB device controllers such as provided with the Analog Devices' ADSP-BF60x processor family. Use of the stack requires the C /USB Device™ Core for CrossCore® Embedded Studio product along with one or more supported device class products described below.

The C /USB Device™ Stack is provided with a hardware abstraction layer and it is modified to easily work with any USB device controllers supported by Analog Devices' processor family. Drivers for several common device classes (Mass Storage, CDC-ACM, HID, PHDC) are offered. A framework for developing vendor-specific class drivers is also provided.

The C/USB-Device™ Stack uses a modular architecture with three software layers between the application and the hardware:

- The Device Controller layer interfaces with the USB Controller driver to process interrupts, notify the device core of bus events, and to receive/transmit packets.
- The Device Core layer controls packet reception and transmission, and responds to standard host requests during enumeration (the process by which a host determines the features of a device).
- The Class layer provides functionality to the host using one or more class drivers. Each class driver responds to class-specific requests and may provide an API for controlling some features and receiving/transmitting information.

**Class Support**

**µC/USB Device™ Class CDC-ACM for CrossCore® Embedded Studio** - The Communication Device Class (CDC) encompasses several communication models. Typical applications include modems, telephone systems and fax machines. The Abstract Control Model (ACM) converts the USB device into a serial communication device, and the target is recognized by the host as a serial interface (USB2COM, Virtual COM port).

**µC/USB Device™ Class HID for CrossCore® Embedded Studio** - The Human Interface Device (HID) Class allows you to implement any kind of user-input device. It can also be used to communicate with the host (without a special host driver) using a vendor-specific communication protocol. Typical applications include mouse, keyboard, game pad, etc....

**µC/USB Device™ Class MSC for CrossCore® Embedded Studio** - The Mass Storage Class (MSC) allows you to use the embedded target device as a USB mass storage device. Typical applications include USB memory stick, digital camera, MP3 player, DVD player, etc....

**µC/USB Device™ Class PHDC for CrossCore® Embedded Studio** - The Personal Healthcare Device Class (PHDC) allows you to set up the embedded target as a personal healthcare device, which can use a vendor-defined or IEEE-11073 based protocol. Typical applications include glucose meter, blood pressure monitor, weighing scale, etc....

**µC/USB Device™ Class Vendor for CrossCore® Embedded Studio** - The Vendor class allows you to develop a custom vendor-specific device (at the application level). This class interacts with WinUSB on Microsoft Windows using a combination of Control, Interrupt and Bulk transfers. A Vendor Specific host application is required to communicate with the device.

Getting Started with µC/USB Device™ Stack for CrossCore® Embedded Studio

Installation

CrossCore® Embedded Studio v.1.0.0 or newer must be installed prior to installing any of the µC /USB Device™ Stack for CrossCore® Embedded Studio products. In addition, µC/USB Device™ Stack operation requires the support of an RTOS. µC/OS-III™ Real-Time Kernel for CrossCore® Embedded Studio is a separate product that may be purchased and installed in support of µC/USB Device Stack. All of the examples that are provided in the various µC/USB Device Stack products require µC/OS-III support. The examples, however, make use of a pre-configured µC/OS-III library, for a quick out-of-the-box experience.

As previously outlined, there are three software layers involved and each is delivered in the products as outlined below:

- The USB Controller driver is delivered in source and binary form as part of the CrossCore Embedded Studio product. All of the remaining products leverage the binary form of the driver. The sources, though not normally needed, are available.
- The Device Core layer product is licensed as a stand alone product and must be installed before any of the Class layer products can be used. It is recommended, though not required, that the Device Core product be installed before installing any of the Class layer products.
- The Class layer products are licensed and installed as individual products.

| Software Layer | Product | Notes |
|---|---|---|
| Layer 1 | USB Contoller Driver | Binary Form as part of libdrv.dlb in:<br>• $CCES_INSTALL\Blackfin\lib\bf609_rev_any<br>• $CCES_INSTALL\Blackfin\lib\bf609_rev_none |

| | | |
|---|---|---|
| | | Source Form: $CCES_INSTALL\Blackfin\lib\src\drivers\source\usb |
| Layer 2 | USB Device Core | |
| Layer 3 | USB Device Class Drivers | |
| | **µC/USB Device™ Class CDC-ACM** | |
| | **µC/USB Device™ Class Vendor** | |
| | **µC/USB Device™ Class PHDC** | |
| | **µC/USB Device™ Class MSC** | |
| | **µC/USB Device™ Class HID** | |

Please make sure to close CrossCore Embedded Studio before proceeding with the installation. If CrossCore Embedded Studio is left open during the installation, it will have to be restarted after installing the µC/USB Device Stack products in order for the changes to take effect, and for µC /USB Device to be available.

Both the µC/USB Device Stack and uC/OS-III install the following common products:

- µC/LIB . This software is always installed into Common Program Files directory. This location is determined by the %CommonProgramFiles(x86)% environment variable in 64-bit operating systems or by %CommonProgramFiles% in 32-bit operating systems.
- µC/CPU. This software is always installed into Common Program Files directory. This location is determined by the %CommonProgramFiles(x86)% environment variable in 64-bit operating systems or by %CommonProgramFiles% in 32-bit operating systems.

The default location for installation is under C:\Analog Devices, e.g. C:\Analog Devices\uCUSB_Device_Core-Rel1.0.0. Should you wish to use a different location, Analog Devices strongly strongly recommends installing the µC/USB Device Stack products outside of the Program Files directory to prevent possible permission issues related to UAC (User Access Control). If you have already installed the product under Program Files then we recommend that you uninstall it and re-install it in a different location.

Note: Multiple versions of the µC/USB Device Stack can be installed on the same system. Only a single instance of a specific version of the product can be installed on a system.

**License Checking**

The installation process checks for a separate license for each of the µC/USB Device Stack products. If a valid license is not detected, the installer will start the Manage Licenses utility for entering and activating a license. The installer will fail in a non-interactive mode when valid license is not present.

**Installation Logging**

The installer does not create a log file by default. If you encounter installation issues, you can generate an installation log file by running the installer from the command prompt.

Change to the directory containing downloaded installer executable and run the following from the command prompt to install the Device Core layer product:

ADI_uCUSB_Device_Core-Rel1.0.0.exe /v"/l*v c:\temp\installer.log"

Similarly, the Class layer products may also be installed from the command line as follows

| Class Layer Product | Command Line Executable Name |
|---|---|
| **µC/USB Device™ Class CDC-ACM** | ADI_uCUSBD_Class_CDC_ACM-Rel1.0.0.exe /v"/l*v c:\temp\installer.log |
| **µC/USB Device™ Class Vendor** | ADI_uCUSBD_Class_Vendor-Rel1.0.0.exe /v"/l*v c:\temp\installer.log |
| **µC/USB Device™ Class PHDC** | ADI_uCUSBD_Class_PHDC-Rel1.0.0.exe /v"/l*v c:\temp\installer.log |
| **µC/USB Device™ Class MSC** | ADI_uCUSBD_Class_MSC-Rel1.0.0.exe /v"/l*v c:\temp\installer.log |
| **µC/USB Device™ Class HID** | ADI_uCUSBD_Class_HID-Rel1.0.0.exe /v"/l*v c:\temp\installer.log |

License

The installation process checks for a valid license for each of the µC/USB Device™ Stack products as listed below. Refer to the Licensing Guide in your CCES installation which can also be found in http://www.analog.com/CrossCoreLicensingGuide.

| |
|---|
| **µC/USB Device™ Stack for CrossCore®Embedded Studio Products** |
| **µC/USB Device™ Core for CrossCore®Embedded Studio Products** |
| **µC/USB Device™ Class CDC-ACM for CrossCore®Embedded Studio Products** |
| **µC/USB Device™ Class Vendor for CrossCore®Embedded Studio Products** |
| **µC/USB Device™ Class PHDC for CrossCore®Embedded Studio Products** |
| **µC/USB Device™ Class MSC for CrossCore®Embedded Studio Products** |
| **µC/USB Device™ Class HID for CrossCore®Embedded Studio Products** |

Support and Assistance

There are several options for contacting support:

- Submit your questions online at:

http://www.analog.com/support

- E-mail your Processor and DSP software and development tools questions from within CrossCore Embedded Studio.

Go to “Help->E-mail Support…”. This will create a new e-mail addressed to processor.tools.support@analog.com, and will automatically attach your CrossCore Embedded Studio version information (ProductInfo.html).

- E-mail your Processors and DSP applications and processor questions to:
- processor.support@analog.com OR
- processor.china@analog.com (Greater China support)
- Post your questions in the Processors and DSP online technical support community in Engineer Zone at

http://ez.analog.com/community/dsp

Supported processors

At the time of its release the only processor family supported in the released version of CrossCore Embedded Studio is ADSP-BF60x.

Software requirements

µC/USB Device™ Stack for CrossCore® Embedded Studio requires the support of an RTOS.

All of the examples ship with a pre-configured version of µC/OS-III™ Real Time Kernel for CrossCore® Embedded Studio. The RTOS is shipped in the form of a binary library that is linked into the examples. The exact configuration of the RTOS is provided in the example documentation.

Getting started with a project that uses uC/USB Device Stack

**Adding uC/USB Device Stack to a project**

Every CrossCore Embedded Studio project contains a System Configuration file called system.svc which is located in the root of the project. The file is the IDE's interface for managing the various pre-written software components used in the "system" implemented by a project. Double-clicking any system.svc file in a navigation view opens that file in the System Configuration Utility which allows you to see the add-ins that you currently have in your project. Clicking on Add and selecting one of the following add-in from the Middleware section under the USB Device category adds the selected product to your project.

Please note that **µC-USB Device Core for Blackfin (1.0.0)** is required for all other products (as previously discussed in this release note). Therefore, when you add in any one of the Class layer products, the µC/USB Device Core product will be automatically added in also.

If an RTOS has not been added in, when you select "Next" in the Add-In dialog, you will be presented with a warning screen indicating that an RTOS product does not yet exist in your application. You will not be able to proceed unless you also select **uCOS-III for Blackfin (1.0.0)** as an additional Add-In.

The µC/USB product add-ins generate code for initializing the µC/USB Device Stack. To ensure timely initialization, when system components are configured the IDE adds any required code to a global C function named adi_initComponents(). A call to this function will be added to the main() function when the USB and RTOS components are added.

**Notes:**

- Please refer to the µC/OS-III Release Notes for RTOS related information.

**Configuration**

µC/USB Device Stack application developers traditionally configure applications by creating header files which contain a long list of macro definitions. µC/USB Device Stack for CrossCore Embedded Studio provides a more intuitive configuration mechanism by providing a tab in the

System Configuration utility, which can be accessed by double-clicking the system.svc file and selecting the µC/USB Device tab. Filling in all the required fields in the configuration tab generates the appropriate files, app_ucusbd_cfg.h and usbd_cfg.h located within the project under system/uc-USB/GeneratedSources.

Notes:

1. Each Class Layer product will introduce a section within the µC/USB Device tab for Class Layer specific configuration. For example, the Mass Storage Class will introduce mass storage specific configuration options.
2. The defaults may not necessarily be appropriate for your application and you should set them to suit your needs. For example, the default setting for the maximum number of logical units supported, 2, may not be correct for your application. The configuration window tool tips should provide an indication of how to set the appropriate values, or look in the documentation for more details.

µC/USB Device Stack project structure

When adding µC/USB Device Stack to a CrossCore Embedded Studio project all the µC/USB Device Stack specific files get placed in the system folder. Please do not change this organization. In the system folder the following structure gets created

- A uC-USB folder. This folder contains sub-folders as follows
- A uC-CPU folder. This folder contains any sources and header files which are required by Micriµm µC/CPU software. µC/CPU provides a processor-independent interface to the supported processors and toolchains that is used in all Micriµm products.
- A uC-LIB folder. This folder contains any sources and header files which are required by Micriµm µC/LIB software. µC/LIB provides a clean and organized implementation of some of the most common standard library functions, macros and constants. µC/LIB is required by many Micriµm products including C/USB.
- A uC-Common folder. This folder contains sources and headers which are common to several Micriµm products but that are not part of any Micriµm product themselves. These include app_cfg.h which is needed by all Micriµm applications.

**Examples**

µC/USB Device Stack for CrossCore Embedded Studio provides examples which show how to use the various Device Classes. Each example is shipped for the ADSP-BF609 platform and can be used in both Release and Debug configurations.

Note:.

- Double-clicking on an example from the example browser or the system overview page opens the project in the installation folder without copying it to your workspace. If you want to modify the example in any way, it is recommended that it gets copied to your workspace. If you would like to copy the project to your workspace note that you may have to copy the sources separately. See Known Issues for more information.

**Location**

In order to locate µC/USB Device Stack examples and sketches, you can use the following:

- Open CrossCore Embedded Studio's (CCES) Example Browser which can be found in CrossCore Embedded Studio under Help. You may then perform one of the following steps
- In the Product Pull-Down select the USB Product that you have licensed and installed
- In the Keyword textbox insert the keyword "USB"
- The result of either of these filters will be a list of USB examples in the Search results panel. The results of browsing by USB keyword with all of the USB products installed is shown below

.

After locating an example of interest, double-clicking on the project in the search results pane will result in the example being imported into the CCES Project Explorer.

System view

CrossCore Embedded Studio provides the System Configuration Utility which is used by µC/USB Device Stack for CrossCore Embedded Studio. Use the System Configuration Overview tab to add the µC/USB Device Stack product add-ins, as appropriate, to a CrossCore Embedded Studio project.

To access the System Configuration Overview tab, do one of the following:

- In a navigation view, double-click the system.svc file of a project. The System Configuration utility appears with the overview tab selected.
- If the utility is already open, select the Overview tab.

As well as being able to add, remove and upgrade add-ins from this window, it also provides a list of examples and sketches associated with the selected add-in.

For more information about the System Configuration utility, see the CrossCore Embedded Studio help.

Configuration tabs

When the µC/USB Device Stack for CrossCore Embedded Studio gets added to a project, several configuration tabs become available in the System view. These include tabs for common Micriµm components such as µC-LIB and µC-CPU. In addition, configuration tabs for µC/OS-III as outlined in the µC/OS-III release notes. These configuration tabs provide an easy mechanism to generate any macro definitions required by the Micriµm products.

A µC/USB Device tab will also be added. Within the µC/USB Device tab a list of sub-tabs for each and every device class will be added.

For more information about each of the configuration options see the section µC/USB Device Tab in CrossCore Embedded Studio's help.

MISRA-C Support

MISRA C is a software development standard for the C programming language developed by the Motor Industry Software Reliability Association (MISRA). Its aims are to facilitate code safety, portability, and reliability in the context of embedded systems, specifically those systems programmed in ANSI C. The compiler detects violations of the MISRA rules at compile-time, link-time, and run-time.

As of this release a list of rules that uC/USB breaks is not available. The USB controller driver, provided by Analog Devices, suppresses the following MISRA rules

Rules 2.1, 8.5, 10.1.a, 10.1.b, 10.5, 11.5, 12.7, 13.7, 14.3, 14.7, 14.10, 16.2, 16.9, 16.10, 17.4, 18.4, 19.4, 19.10

µC/USB Device™ Stack for CrossCore® Embedded Studio RTOS Requirements

µC/USB Device Stack for CrossCore Embedded Studio requires the presence of an RTOS, although not necessarily µC/OS-III Real-Time Kernel for CrossCore Embedded Studio. When running in a µC/OS-III application, µC/USB Device Stack requires multiple µC/OS-III objects like semaphores and Task-specific registers slots.

Removing any of the µC/OS-III functionality that is required by a µC/USB Device application could cause link errors.

Note that adding µC/USB Device to a project which already has µC/OS-III may change some RTOS settings. The full list of specific changes is displayed by the µC/USB Device addition process.

Common Micriµm Components

There are several CrossCore® Embedded Studio add-ins based on Micriµm's products which share common components. To ensure that the same version of these components is used by all the add-ins that require them, these components are installed in a common location which is distinct from the add-in install folders. These common components are

- µC/CPU which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-CPU v1.0.0. This installation includes µC/CPU version 1.29.01.

- µC/LIB which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-LIB v1. 0.0. This installation includes µC/LIB version 1.36.02.

The documentation for these components can be found in CrossCore® Embedded Studio Help under µC/OS-III™ 1.0.0 > Components Shared by Add-ins.

Known issues with µC/USB Device™ Stack for CrossCore® Embedded Studio

These are the currently known problems which affect µC/USB Device™ Stack for CrossCore® Embedded Studio.

**TAR-48585: RTOS examples are not portable to other workspaces**

The µCOS-III examples released with the µCOS-III product reference their source folder (src) and their readme in terms of a relative path from the project location. The reason for this was so the same sources and readme could be used for all processors that the example can be run on.

This method of linking files means that if a user chooses to import the project with the "Copy projects into workspace" box selected then the src and readme files are not copied and the example does not work.

If you want to import one of the examples to your workspace then you should follow these instructions:

Import the example copying it to your workspace. This step creates a new project in the workspace which contains two invalid links, to the readme and to the src folder.
Remove the existing link to the src folder and to the readme by selecting them on the new project and either press the "Delete" key or right-click and choose Delete.
Close the project.
Copy the readme and src folder to the location of the project in the new workspace. This will automatically add those files to the project. The location of the µCOS-III examples in the file system is: %RTOS_INSTALL_FOLDER%\uCOS-III\Examples.
Reopen the project
If the project was previously built in the installation folder then run "Clean" before rebuilding.
The example should build as work as expected.

NOTE: This issue also applies to the µC/USB Device Stack for CrossCore Embedded Studio products.

**TAR-48536: µC/USB Device Stack for CrossCore Embedded studio can only be used with uCOS-III**

µC-USB Device Stack for CrossCore Embedded studio is currently configured only to use µCOS-III for the RTOS. Micriµm supply the OS layer files for µC/OS-II, which will be added in a future release of the product.