



**µC/OS-II® Real-Time Kernel for
CrossCore® Embedded Studio version
2.4.1 Release Notes**

Contents

1	μC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.1 Release Notes	3
2	μC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.0 Release Notes	6
3	μC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.1.0 Release Notes	9
4	μC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.0.0 Release Notes	12
5	μC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 1.1.0 Release Notes	15
6	μCOS-II™ Real-Time Kernel for CrossCore® Embedded Studio version 1.0.0 Release Notes	18
7	Getting Started with μC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio	19

1 μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.1 Release Notes

Introduction

This document contains the release notes for μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.1. It describes the release in detail and provides latest information that supplements the main documentation.

This release includes anomaly support as in the next section, below. Users of previous releases should check the "Version Compatibility" section, below, for pertinent instructions on modifying existing applications for this new release

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>

Update Highlights

The focus of μ C/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio version 2.4.1 is to provide workarounds for anomalies 20-00-0069 and 20-00-0081, which affect the SHARC+ core (s) of ADSP-215xx/SC5xx processors (only).

Micrium software versions

μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.1 is based on Micrium's μ C/OS-II® Real-Time Kernel version 2.92.11

There are several CrossCore Embedded Studio products based on Micrium's software which share common add-ins. To ensure that the same version of these add-ins is used by all the products that require them, the common add-ins are installed in a location which is distinct from the $\mu\text{C}/\text{OS-II}$ install folder. The common add-ins are

- $\mu\text{C}/\text{CPU}$ which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-CPU v2.0.0. This installation includes $\mu\text{C}/\text{CPU}$ version 1.30.01.
- $\mu\text{C}/\text{LIB}$ which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-LIB v2.0.0. This installation includes $\mu\text{C}/\text{LIB}$ version 1.38.00.

Version Compatibility

$\mu\text{C}/\text{OS-II}$ ® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.1 requires CrossCore® Embedded Studio version 2.4.0 or later.

Software requirements

In order to be able to add $\mu\text{C}/\text{OS-II}$ Real-Time Kernel for CrossCore® Embedded Studio version 2.4.1 to a project you need to use CrossCore Embedded Studio version 2.4.0 or later.

$\mu\text{C}/\text{OS-II}$ ® Real-Time Kernel for CrossCore® Embedded Studio for Cortex-A cores

Supervisor vs System mode

By default, CrossCore Embedded Studio applications on Cortex-A cores run on system mode, which means that the SVC stack is dedicated to interrupt/exception support. This is not the case in applications that use $\mu\text{C}/\text{OS-II}$. These applications run in supervisor (svc) mode and therefore the svc stack is also used in some kernel operations.

General-purpose timer vs Core timer

Cortex-A cores do not have a core timer. In order to provide timer facilities out of the box, $\mu\text{C}/\text{OS-II}$ ® Real-Time Kernel for CrossCore® Embedded Studio uses general purpose timer 0. Application developers can configure timing facilities in the $\mu\text{C}/\text{OS-II}$ configuration options under the "Application Timer/Tick" tab.

$\mu\text{C}/\text{OS-II}$ ® Real-Time Kernel for CrossCore® Embedded Studio for SHARC+ cores

Core software interrupt vs System software interrupt

Previously, on all SHARC platforms, $\mu\text{C}/\text{OS-II}$ raised a core software interrupt (SFT3I) to initiate a task switch from task level. This interrupt was raised by performing a bit-set operation on the interrupt latch (IRPTL) register of the Core Interrupt Controller (CEC). In $\mu\text{C}/\text{OS-II}$ version 2.4.1 this method is still used for all SHARC variants up to and including ADSP214xx, but a different method is now used in ADSP-215xx/SC5xx due to the presence of anomaly 20-00-0081. This new method raises a system software interrupt by writing to the RAISE register of the System Event Controller (SEC). Since the SEC interrupts are system-wide rather than core-specific it is necessary to use a different SEC software interrupt for each core - SOFT6 is used by SHARC1 and SOFT7 is used by SHARC2.

The latency of raising a system software interrupt is greater than that of raising a core software interrupt. This increased latency will be reflected in the performance figures for task switches that are initiated from task level.

Previous versions of SHARC-specific add-ins are displayed

Previous versions of μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio allowed the addition of the add-in to a project based on the processor family. Since ADSP-SC58x is a SHARC processor with heterogeneous cores, this means that the add-in for SHARC gets displayed in the add-in selection for the Cortex-A5 core as well even if it is not applicable. μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio 2.4.1 has moved to only display add-ins based on the core type and therefore only the applicable 2.2.0 add-ins are displayed.

μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio Software Anomalies

This section enumerates the most significant anomalies which relate to μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio. For a comprehensive list of all the public software anomalies visit <http://www.analog.com/SoftwareAnomalies>

Known issues with μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio 2.4.1

- **UCOS2-102: ThreadLocalStorage destruction limitations in Cortex-A cores** On Cortex-A cores there are limitations to the functions and APIs that can be called from the Thread-Local Storage "destructor function" which was registered with `OS_TLS_SetDestruct()`. Users must not call any functions that directly or indirectly can cause a context switch. This includes heap management functions like "free". Calling these functions can result in a badly destroyed task and in the "destructor function" not reaching completion. As a workaround, the function can be surrounded by the following code sequence:

```
CPU_CRITICAL_ENTER(); OSIntNestingCtr++; ... function call ... OSIntNestingCtr--; CPU_CRITICAL_EXIT();
```
- **CCES-12802: Adding a ucos/ucusb/ucfs addin version 1.1.0 to CCES 1.x when a version 2.3.0 add-in is installed results in warnings** CCES 1.x gives a warning when adding a Micrium-based add-in to a project if the machine contains any Micrium-based products version 2.0.0 or later. Although the 2.3.0 add-ins are not supported in CCES 1.x, when adding a Micrium based add-in version 1.1.0 the UI tries to add some common components which are version 2.3.0 resulting in a warning which says "Some Add-in dependencies are not resolved, please check the selected Add-ins.". In order to add your add-ins correctly you can do one of the following things: 1.- Close CCES, uninstall all Micrium-based products version 2.3.0 or later, restart CCES and add your add-in normally. 2.- In the Add-in selection window, click on the advanced tab and select uc-cpu and uc-lib version 1.1.0 instead of the 2.3.0 which are preselected.

2 μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.0 Release Notes

Introduction

This document contains the release notes for μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.1. It describes the release in detail and provides latest information that supplements the main documentation.

This release includes support for the processors listed in the next section, below. Users of previous releases should check the "Version Compatibility" section, below, for pertinent instructions on modifying existing applications for this new release

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>

Update Highlights

The focus of μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.0 is to support the release of the ADSP-SC57x processor family which includes support for both the ARM Cortex-A5 core and the SHARC+ core(s).

The version of the Micrium's kernel included with μ C/OS-II® Real-Time Kernel for CCES version 2.4.0 remains at v 2.92.11.

Micrium's documentation has migrated to their website at address <http://doc.micrium.com> instead of being included with μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio. Any documentation specific to Analog Devices processors can still be found within this product.

Micrium software versions

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.0 is based on Micrium's µC/OS-II® Real-Time Kernel version 2.92.11

There are several CrossCore Embedded Studio products based on Micrium's software which share common add-ins. To ensure that the same version of these add-ins is used by all the products that require them, the common add-ins are installed in a location which is distinct from the µC/OS-II install folder. The common add-ins are

- µC/CPU which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-CPU v2.0.0. This installation includes µC/CPU version 1.30.01.
- µC/LIB which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-LIB v2.0.0. This installation includes µC/LIB version 1.38.00.

Version Compatibility

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.4.0 requires CrossCore® Embedded Studio version 2.4.0 or later.

Software requirements

In order to be able to add µC/OS-II Real-Time Kernel for CrossCore® Embedded Studio version 2.4.0 to a project you need to use CrossCore Embedded Studio version 2.4.0 or later.

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio for Cortex-A cores

Supervisor vs System mode

By default, CrossCore Embedded Studio applications on Cortex-A cores run on system mode, which means that the SVC stack is dedicated to interrupt/exception support. This is not the case in applications that use µC/OS-II. These applications run in supervisor (svc) mode and therefore the svc stack is also used in some kernel operations.

General-purpose timer vs Core timer

Cortex-A cores do not have a core timer. In order to provide timer facilities out of the box, µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio uses general purpose timer 0. Application developers can configure timing facilities in the µC/OS-II configuration options under the "Application Timer/Tick" tab.

Previous versions of SHARC-specific add-ins are displayed

Previous versions of µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio allowed the addition of the add-in to a project based on the processor family. Since ADSP-SC58x is a SHARC processor with heterogeneous cores, this means that the add-in for SHARC gets displayed in the add-in selection for the Cortex-A5 core as well even if it is not applicable. µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio 2.4.0 has moved to only display add-ins based on the core type and therefore only the applicable 2.2.0 add-ins are displayed.

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio Software Anomalies

This section enumerates the most significant anomalies which relate to μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio. For a comprehensive list of all the public software anomalies visit <http://www.analog.com/SoftwareAnomalies>

Known issues with μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio 2.4.0

- **UCOS2-102: ThreadLocalStorage destruction limitations in Cortex-A cores** On Cortex-A cores there are limitations to the functions and APIs that can be called from the Thread-Local Storage "destructor function" which was registered with `OS_TLS_SetDestruct()`. Users must not call any functions that directly or indirectly can cause a context switch. This includes heap management functions like "free". Calling these functions can result on a badly destroyed task and on the "destructor function" not reaching completion. As a workaround, the function can be surrounded by the following code sequence:

```
CPU_CRITICAL_ENTER(); OSIntNestingCtr++; ... function call ... OSIntNestingCtr--; CPU_CRITICAL_EXIT();
```
- **CCES-12802: Adding a ucos/ucusb/ucfs addin version 1.1.0 to CCES 1.x when a version 2.3.0 add-in is installed results in warnings** CCES 1.x gives a warning when adding a Micrium-based add-in to a project if the machine contains any Micrium-based products version 2.0.0 or later. Although the 2.3.0 add-ins are not supported in CCES 1.x, when adding a Micrium based add-in version 1.1.0 the UI tries to add some common components which are version 2.3.0 resulting in a warning which says "Some Add-in dependencies are not resolved, please check the selected Add-ins.". In order to add your add-ins correctly you can do one of the following things: 1.- Close CCES, uninstall all Micrium-based products version 2.3.0 or later, restart CCES and add your add-in normally. 2.- In the Add-in selection window, click on the advanced tab and select uc-cpu and uc-lib version 1.1.0 instead of the 2.3.0 which are preselected.

3 μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.1.0 Release Notes

Introduction

This document contains the release notes for μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.1.0. It describes the release in detail and provides latest information that supplements the main documentation.

This release includes support for the processors listed in the next section, below. Users of previous releases should check the "Version Compatibility" section, below, for pertinent instructions on modifying existing applications for this new release

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>

Update Highlights

The only update of μ C/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio version 2.1.0 is a bug fixing about the failure to restore R4 in the context switch on the ADSP-SC5xx SHARC cores and also fix the SIMD context switching bug on SHARC.

The version of the Micrium's kernel included with μ C/OS-II® Real-Time Kernel for CCES version 2.1.0 remains at v 2.92.11.

Micrium's documentation has migrated to their website at address <http://doc.micrium.com> instead of being included with μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio. Any documentation specific to Analog Devices processors can still be found within this product.

Micrium software versions

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.1.0 is based on Micrium's µC/OS-II® Real-Time Kernel version 2.92.11

There are several CrossCore Embedded Studio products based on Micrium's software which share common add-ins. To ensure that the same version of these add-ins is used by all the products that require them, the common add-ins are installed in a location which is distinct from the µC/OS-II install folder. The common add-ins are

- µC/CPU which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-CPU v2.0.0. This installation includes µC/CPU version 1.30.01.
- µC/LIB which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-LIB v2.0.0. This installation includes µC/LIB version 1.38.00.

Version Compatibility

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.1.0 requires CrossCore® Embedded Studio version 2.1.0 or later.

Software requirements

In order to be able to add µC/OS-II Real-Time Kernel for CrossCore® Embedded Studio version 2.1.0 to a project you need to use CrossCore Embedded Studio version 2.1.0 or later.

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio for Cortex-A cores

Supervisor vs System mode

By default, CrossCore Embedded Studio applications on Cortex-A cores run on system mode, which means that the SVC stack is dedicated to interrupt/exception support. This is not the case in applications that use µC/OS-II. These applications run in supervisor (svc) mode and therefore the svc stack is also used in some kernel operations.

General-purpose timer vs Core timer

Cortex-A cores do not have a core timer. In order to provide timer facilities out of the box, µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio uses general purpose timer 0. Application developers can configure timing facilities in the µC/OS-II configuration options under the "Application Timer/Tick" tab.

Previous versions of SHARC-specific add-ins are displayed

Previous versions of µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio allowed the addition of the add-in to a project based on the processor family. Since ADSP-SC58x is a SHARC processor with heterogeneous cores, this means that the add-in for SHARC gets displayed in the add-in selection for the Cortex-A5 core as well even if it is not applicable. µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio 2.1.0 has moved to only display add-ins based on the core type and therefore only the applicable 2.1.0 add-ins are displayed.

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio Software Anomalies

This section enumerates the most significant anomalies which relate to μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio. For a comprehensive list of all the public software anomalies visit <http://www.analog.com/SoftwareAnomalies>

Known issues with μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio 2.1.0

- **UCOS2-102: ThreadLocalStorage destruction limitations in Cortex-A cores** On Cortex-A cores there are limitations to the functions and APIs that can be called from the Thread-Local Storage "destructor function" which was registered with `OS_TLS_SetDestruct()`. Users must not call any functions that directly or indirectly can cause a context switch. This includes heap management functions like "free". Calling these functions can result on a badly destroyed task and on the "destructor function" not reaching completion. As a workaround, the function can be surrounded by the following code sequence:
`CPU_CRITICAL_ENTER(); OSIntNestingCtr++; ... function call ... OSIntNestingCtr--; CPU_CRITICAL_EXIT();`
- **CCES-12802:** Adding a `ucos/ucusb/ucfs` addin version 1.1.0 to CCES 1.x when a version 2.2.0 add-in is installed results in warnings CCES 1.x gives a warning when adding a Micrium-based add-in to a project if the machine contains any Micrium-based products version 2.0.0 or later. Although the 2.2.0 add-ins are not supported in CCES 1.x, when adding a Micrium based add-in version 1.1.0 the UI tries to add some common components which are version 2.2.0 resulting in a warning which says "Some Add-in dependencies are not resolved, please check the selected Add-ins.". In order to add your add-ins correctly you can do one of the following things: 1.- Close CCES, uninstall all Micrium-based products version 2.2.0 or later, restart CCES and add your add-in normally. 2.- In the Add-in selection window, click on the advanced tab and select `uc-cpu` and `uc-lib` version 1.1.0 instead of the 2.2.0 which are preselected.

4 μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.0.0 Release Notes

Introduction

This document contains the release notes for μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.0.0. It describes the release in detail and provides latest information that supplements the main documentation.

This release includes support for the processors listed in the next section, below. Users of previous releases should check the "Version Compatibility" section, below, for pertinent instructions on modifying existing applications for this new release.

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>

Update Highlights

The focus of μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.0.0 is to support the release of the ADSP-SC58x processor family which includes support for both the ARM Cortex-A5 core and the SHARC+ core(s).

The version of the Micrium's kernel included with μ C/OS-II® Real-Time Kernel for CCES version 2.0.0 remains at v 2.92.11.

Micrium's documentation has migrated to their website at address <http://doc.micrium.com> instead of being included with μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio. Any documentation specific to Analog Devices processors can still be found within this product.

Micrium software versions

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.0.0 is based on Micrium's µC/OS-II® Real-Time Kernel version 2.92.11

There are several CrossCore Embedded Studio products based on Micrium's software which share common add-ins. To ensure that the same version of these add-ins is used by all the products that require them, the common add-ins are installed in a location which is distinct from the µC/OS-II install folder. The common add-ins are

- µC/CPU which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-CPU v2.0.0. This installation includes µC/CPU version 1.30.01.
- µC/LIB which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-LIB v2.0.0. This installation includes µC/LIB version 1.38.00.

Version Compatibility

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 2.0.0 requires CrossCore® Embedded Studio version 2.0.0 or later.

Software requirements

In order to be able to add µC/OS-II Real-Time Kernel for CrossCore® Embedded Studio version 2.0.0 to a project you need to use CrossCore Embedded Studio version 2.0.0 or later.

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio for Cortex-A cores

Supervisor vs System mode

By default, CrossCore Embedded Studio applications on Cortex-A cores run on system mode, which means that the SVC stack is dedicated to interrupt/exception support. This is not the case in applications that use µC/OS-II. These applications run in supervisor (svc) mode and therefore the svc stack is also used in some kernel operations.

General-purpose timer vs Core timer

Cortex-A cores do not have a core timer. In order to provide timer facilities out of the box, µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio uses general purpose timer 0. Application developers can configure timing facilities in the µC/OS-II configuration options under the "Application Timer/Tick" tab.

Previous versions of SHARC-specific add-ins are displayed

Previous versions of µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio allowed the addition of the add-in to a project based on the processor family. Since ADSP-SC58x is a SHARC processor with heterogeneous cores, this means that the add-in for SHARC gets displayed in the add-in selection for the Cortex-A5 core as well even if it is not applicable. µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio 2.0.0 has moved to only display add-ins based on the core type and therefore only the applicable 2.0.0 add-ins are displayed.

µC/OS-II® Real-Time Kernel for CrossCore® Embedded Studio Software Anomalies

This section enumerates the most significant anomalies which relate to μ C/OS-II[®] Real-Time Kernel for CrossCore[®] Embedded Studio. For a comprehensive list of all the public software anomalies visit <http://www.analog.com/SoftwareAnomalies>

Known issues with μ C/OS-II[®] Real-Time Kernel for CrossCore[®] Embedded Studio 2.0.0

- **UCOS2-102: ThreadLocalStorage destruction limitations in Cortex-A cores**

On Cortex-A cores there are limitations to the functions and APIs that can be called from the Thread-Local Storage "destructor function" which was registered with OS_TLS_SetDestruct(). Users must not call any functions that directly or indirectly can cause a context switch. This includes heap management functions like "free". Calling these functions can result on a badly destroyed task and on the "destructor function" not reaching completion. As a workaround, the function can be surrounded by the following code sequence: CPU_CRITICAL_ENTER(); OSIntNestingCtr++; ... function call ... OSIntNestingCtr--; CPU_CRITICAL_EXIT();

- **CCES-12802: Adding a ucos/ucusb/ucfs addin version 1.1.0 to CCES 1.x when a version 2.0.0 add-in is installed results in warnings**

CCES 1.x gives a warning when adding a Micrium-based add-in to a project if the machine contains any Micrium-based products version 2.0.0 or later. Although the 2.0.0 add-ins are not supported in CCES 1.x, when adding a Micrium based add-in version 1.1.0 the UI tries to add some common components which are version 2.0.0 resulting in a warning which says "Some Add-in dependencies are not resolved, please check the selected Add-ins.". In order to add your add-ins correctly you can do one of the following things: 1.- Close CCES, uninstall all Micrium-based products version 2.0.0 or later, restart CCES and add your add-in normally. 2.- In the Add-in selection window, click on the advanced tab and select uc-cpu and uc-lib version 1.1.0 instead of the 2.0.0 which are preselected.

5 μ C/OS-II® Real-Time Kernel for CrossCore® Embedded Studio version 1.1.0 Release Notes

Introduction

This document contains the release notes for μ C/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio version 1.1.0. It describes the release in detail and provides latest information that supplements the main documentation.

This release includes support for the processors listed in the next section, below. Users of previous releases should check the "Version Compatibility" section, below, for pertinent instructions on modifying existing applications for this new release.

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>

Update Highlights

The focus of μ C/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio version 1.1.0 is to support the release of ADSP-BF70x processor family.

The version of the Micrium's kernel included with μ C/OS-II™ Real-Time Kernel for CCES version 1.1.0 has been upgraded to v 2.92.11.

Micrium's documentation has migrated to their website at address <http://doc.micrium.com> instead of being included with μ C/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio. Any documentation specific to Analog Devices processors can still be found within this product.

ADSP-BF70x processor support

The μ C/OS-II support has been extended to support the following processors:

- ADSP-BF700, ADSP-BF701, ADSP-BF702, ADSP-BF703, ADSP-BF704, ADSP-BF705, ADSP-BF706, ADSP-BF707

To find more information about these processors go to <http://www.analog.com/BF70x>

Dedicated ISR stack support in Blackfin processors

µC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio version 1.1.0 includes the check box "Enable dedicated ISR stack" in the µC/OS-II Application General Options which allows you to select whether interrupt service routines (ISR) should use a separate stack instead of the running task stack. Enabling this setting uses a stack of size "ISR stack size". Although the ISR stack size setting was available in previous versions of the product, this was ignored by the kernel. This is still the case in SHARC applications which do not currently support dedicated ISR stacks.

Setting dedicated ISR stacks might reduce the footprint of an application since task stacks no longer need to be large enough to accommodate the stack usage of all nested ISRs.

Improvements on out-of-the-box performance due to user interface fix

Due to a user interface issue, un-checking the the µC/CPU check box Enable 'interrupts disabled' time measurement did not result on the appropriate macro being correctly generated. This issue has been fixed in µC/CPU version 1.1.0. This produces a significant performance improvement in many API calls so it should be noticeable in a great number of applications. The performance improvement that can be seen in the figures provided with this product is mainly due to this change.

Micrium software versions

µC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio version 1.1.0 is based on Micrium's µC/OS-II™ Real-Time Kernel version 2.92.11

There are several CrossCore Embedded Studio products based on Micrium's software which share common add-ins. To ensure that the same version of these add-ins is used by all the products that require them, the common add-ins are installed in a location which is distinct from the µC/OS-II install folder. The common add-ins are

- µC/CPU which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-CPU v1.1.0. This installation includes µC/CPU version 1.30.01.
- µC/LIB which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-LIB v1.1.0. This installation includes µC/LIB version 1.38.00.

Version Compatibility

µC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio version 1.1.0 requires CrossCore® Embedded Studio version 1.1.0 or later.

Software requirements

In order to be able to add µC/OS-II Real-Time Kernel for CrossCore® Embedded Studio version 1.1.0 to a project you need to use CrossCore Embedded Studio version 1.1.0 or later.

μC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio Software Anomalies

This section enumerates the most significant anomalies which relate to μC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio. For a comprehensive list of all the public software anomalies visit <http://www.analog.com/SoftwareAnomalies>

- **Known issues with μC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio 1.1.0**

Upgrading μCOS-II add-in to 1.1.0 results in warnings

CCES produces some unexpected warnings when upgrading the μCOS-II component to 1.1.0. The following warnings can be ignored safely and the resulting project should build without issues. Property "uccpu-ucos2-compatibility" required by Add-in "..." doesn't exist Property "uclib-ucos2-compatibility" required by Add-in "..." doesn't exist The root cause for this issue is a CrossCore Embedded Studio problem which has been recorded as CCES-7883

6 μ COS-II™ Real-Time Kernel for CrossCore® Embedded Studio version 1.0.0 Release Notes

What is μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio

μ C/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio is the result of a partnership between Analog Devices and Micrium to provide a user-friendly programming environment for μ C/OS-II applications running on Analog Devices' Blackfin and SHARC processors. μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio provides an integrated environment with CrossCore Embedded Studio which offers the advantage of an industry-standard IDE combined with Analog Devices' advanced optimizing compiler technology.

μ C/OS-II is Micrium's long standing RTOS, designed to save time on embedded system projects.

μ C/OS-II applications determine at build time the number and type of resources that they require, ensuring that there is no waste of memory due to resources that the application does not require.

μ C/OS-II requires that every task and mutual exclusion semaphore (mutex) are allocated to different priority levels. Applications can be configured to allow for tasks to change their priorities at runtime.

μ C/OS-II has a number of internal data structures and variables that it needs to access atomically. It protects these atomic regions by disabling interrupts for a small number of cycles. Interrupt response with μ C/OS-II is deterministic.

μ C/OS-II APIs return a comprehensive set of error codes which eases application development by being able to detect a varied set of user errors.

μ C/OS-II's footprint can be scaled to only contain the features required for a specific application. The execution time for most services provided by μ C/OS-II is both constant and deterministic; execution times do not depend on the number of tasks running in the application.

μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio version 1.0.0 is released with Micrium's μ C/OS-II version 2.92.08.

7 Getting Started with μ C/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio

Installation

CrossCore® Embedded Studio must be installed prior to installing μ C/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio.

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 μ C/OS-II Real-Time Kernel.

μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio installs the following:

- μ C/OS-II Real-Time Kernel. Its default installation directory is C:\Analog Devices\uCOS-II-Rel2.3.0. This release is based on Micrium's μ C/OS-II™ Real-Time Kernel version 2.92.08
- μ C/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. This release includes μ C/LIB 2.0.0 which is based on Micrium's μ C/LIB version 1.38.00
- μ C/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. This release includes μ C/CPU 2.0.0 which is based on Micrium's μ C/CPU version 1.30.01.

The documentation for all of these add-ins can be found in CrossCore® Embedded Studio Help under μ C/OS-II™ 2.3.0

Analog Devices strongly recommends installing μ C/OS-II Real-Time Kernel 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 it is recommend that you uninstall it and re-install it in a different location.

Note: Multiple versions of the μ C/OS-II Real-Time Kernel can be installed on the same system. Only a single instance of a specific version of the product can be installed on a system.

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:

```
ADI_uCOS-II-Rel2.3.0.exe /v"/l*v c:\temp\installer.log"
```

License

The installation process checks for a valid license for μ C/OS-II™ Real-Time Kernel. 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.

Refer to the Licensing Guide in your CCES installation which can also be found in <http://www.analog.com/CrossCoreLicensingGuide>.

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

μ C/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio (CCES) includes support for existing Blackfin and SHARC processors supported by CCES version 2.3.0.

These supported processors are:

Blackfin

- ADSP-BF504, ADSP-BF504F, ADSP-BF506F
- ADSP-BF512, ADSP-BF514, ADSP-BF516, ADSP-BF518
- ADSP-BF522, ADSP-BF524, ADSP-BF526, ADSP-BF523, ADSP-BF525, ADSP-BF527
- ADSP-BF531, ADSP-BF532, ADSP-BF533, ADSP-BF534, ADSP-BF536, ADSP-BF537, ADSP-BF538, ADSP-BF539
- ADSP-BF542, ADSP-BF542M, ADSPBF547, ADSP-BF547M, ADSP-BF548, ADSP-BF548M, ADSP-BF549, ADSP-BF549M

- ADSP-BF544, ADSP-BF544M
- ADSP-BF561
- ADSP-BF592-A
- ADSP-BF606, ADSP-BF607, ADSP-BF608, ADSP-BF609

SHARC

- ADSP-21160, ADSP-21161
- ADSP-21261, ADSP-21262, ADSP-21266
- ADSP-21362, ADSP-21363, ADSP-21364, ADSP-21365, ADSP-21366, ADSP-21367, ADSP-21368, ADSP-21369
- ADSP-21371, ADSP-21375
- ADSP-21467, ADSP-21469
- ADSP-21477, ADSP-21478, ADSP-21479
- ADSP-21483, ADSP-21486, ADSP-21487, ADSP-21488, ADSP-21489
- ADSP-SC589

Software requirements

µC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio requires CrossCore Embedded Studio version 2.3.0 or later to be installed to build projects.

Getting started with a project that uses the RTOS

Adding the RTOS to a project

Creating a project which includes the RTOS

In order to create a project you should follow the instructions provided in the CrossCore Embedded Studio help. As part of the project creation, the page Add-in selection contains a list of all the available add-ins for the project that you are creating based on the installed products and the project's chosen processor and type. You can see the µC/OS-II add-in in the Middleware section under the RTOS category.

The µC/OS-II add-in generates the OSInit() call for initializing the RTOS, and this code should be called prior to initializing other services/devices. To ensure timely initialization, the IDE adds any required code to a global C function named adi_initComponents(). If you choose to add a default C source to your project then the resulting project contains a call to adi_initComponents(). If the template C code is not selected at project creation then it is the application developer's responsibility to add initialization code for the components required including the RTOS.

There are no calls added to create start-up tasks or to OSStart () to start the OS. These API calls remain the application's responsibility.

Adding the RTOS to an existing 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 μ C/OS-II add-in from the Middleware section under the RTOS category adds μ COS-II to your project.

The μ C/OS-II add-in generates code for initializing the RTOS, and this code should be called prior to initializing other services/devices. To ensure timely initialization, when system components are configured the IDE adds any required code to a global C function named `adi_initComponents()`. You should call this function early in the `main()` routine.

Notes:

- If the IDE detects that `adi_initComponents()` is not yet present in `main()`, it prompts you to add it and offers to insert it for you.
- There are no calls added to create start-up tasks or to `OSStart()` to start the OS. These API calls remain the application responsibility.
- Adding the RTOS to a project means that thread-safe versions of libraries are used, including the Runtime library and the System Services/Device Drivers library. It is the application's responsibility to ensure that any functions that should only be called from task level are not called from main but from a task function.

Actions performed by CrossCore Embedded Studio when adding μ C/OS-II to a project

When a μ C/OS-II add-in is added to a CrossCore Embedded Studio project, the CCES add-in framework executes the following actions

- Adds links to the kernel files in the installation folder
- Adds generated files based on the configuration window settings
- Sets up the required toolchain options.
- Copies `os_app_hooks.c` to your project. This file is a copy of a Micrium template which contains hook functions which are used depending on configuration options. This file can be edited by the application developer and will not be removed from the project when the file system is removed or overwritten if it is already present in the destination folder.
- Copies `app_ucos2_cfg.h` to your project. This file is empty by default but can contain application-specific macros and definitions. This file can be edited by the application developer and will not be removed from the project when the file system is removed or overwritten if it is already present in the destination folder.

Configuration

μ C/OS-II application developers traditionally configure applications by creating header files which contain a long list of macro definitions. In order to have a more intuitive configuration mechanism, μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio contains a configuration tab which can be accessed by double-clicking the `system.svc` file and selecting the μ C/OS-II tab. Selecting all the desired fields in the configuration tab generates the appropriate files which are part of the project.

Note that the defaults may not necessarily be appropriate for your application and you should set them to suit your needs. The configuration window tool tips should give you an indication of how you should set your values or look in the documentation for more details.

μ C/OS-II project structure

When adding μ C/OS-II to a CrossCore Embedded Studio project all the μ C/OS-II specific files get placed in the `system` folder. Please do not change this organization. In the `system` folder the following structure gets created

- A `uCOS-II` folder. This folder contains any sources and header files which are part of the μ C/OS-II RTOS. It contains the folders:
 - Source, TLS and Ports. These folders contain the actual RTOS code which should not be modified.
 - AppSources and Cfg. These folders contain a copy of the templates of headers and sources that are provided by Micrium and that applications can customize to suit their needs.
 - GeneratedSources. This folder contains headers and sources generated by the product based on the GUI configurations. These files should not be modified.
- A `uC-CPU` folder. This folder contains any sources and header files which are required by Micrium μ C/CPU software. μ C/CPU provides a processor-independent interface to the supported processors and toolchains that is used in all Micrium products.
- A `uC-LIB` folder. This folder contains any sources and header files which are required by Micrium μ C/LIB software. μ C/LIB provides a clean and organized implementation of some of the most common standard library functions, macros and constants. μ C/LIB is required by many Micrium products including C/OS-II.
- A `uC-Common` folder. This folder contains sources and headers which are common to several Micrium products but that are not part of any Micrium product themselves. These include `app_cfg.h` which is needed by all Micrium applications.

Interrupts

CrossCore Embedded Studio for Analog Devices processors provides a coherent interrupt management mechanism which allows for the same interface to be used in RTOS and non-RTOS applications. This means that interrupt service routines in $\mu\text{C}/\text{OS-II}$ applications must be written in C and use the `adi_int` interface, regardless of whether they use $\mu\text{C}/\text{OS-II}$ APIs. Any thread-safety requirements or interactions with tasks are handled by the `adi_int` interface. For more information on the `adi_int` API, in CrossCore Embedded Studio go to Help > Search and enter `adi_int`.

Examples of the usage of this interrupt management mechanism are the System Services and Device Drivers provided with Crosscore Embedded Studio. By using the `adi_int` interface, the same services and drivers can be used in all applications regardless of whether an operating system is used.

BSP module

In some of the examples described in the $\mu\text{C}/\text{OS-II}$ manual provided by Micrium for all platforms, there is a mention to some BSP-specific functions. $\mu\text{C}/\text{OS-II}$ Real-Time Kernel for CrossCore Embedded Studio does not provide any BSP-specific functionality and the RTOS is ready to be used from main without any BSP calls. In order to access hardware-specific functionality the System Services and Device Drivers functionality provided with CrossCore Embedded Studio should be used.

Examples and sketches

Code sketches

CrossCore Embedded Studio provides a mechanism by which small code fragments, called code sketches, can be generated with parametrized input provided by the user. The resulting code can then be copied and pasted to a project. $\mu\text{C}/\text{OS-II}$ Real-Time Kernel for CrossCore Embedded Studio provides a few sketches which demonstrate the API calls that are required for

- Creating a task
- Creating a signal (semaphores, mutexes, etc)
- Pending on signals
- Posting signals

The CCES code sketch mechanism is designed for small code fragments. For this reason some of the $\mu\text{C}/\text{OS-II}$ code sketches hard-code certain elements like task name or signals to pend in `OSPendMulti`. The intention of these sketches is to quickly show the intended code sequence and they should be modified to suit the application's needs.

Examples

In addition to the code sketches, μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio provides a few simple examples which show how to use the RTOS. Each example is shipped for three platforms: ADSP-BF609, ADSP-BF533 and ADSP-21469 and can be used in both Release and Debug configurations. The examples can be run on an EZ-KIT for all platforms or on a simulator for the platforms that have both (ADSP-BF533 and ADSP-21469).

Notes

- The examples use features that can be disabled in the configuration windows. Disabling any of the used features may result in expected link errors.
- Double-clicking on an example from the example browser or the system overview page makes a copy of the example in your workspace. Any modifications to the example itself happen to your local copy but, changing any of the kernel sources changes the sources in the installation folder which affects all μ C/OS-II applications. For this reason it is not recommended that you change any sources which are under the system/uCOS-II/Source, system/uCOS-II/Ports and system/uCOS-II/TLS folders.

Location

In order to locate μ C/OS-II examples and sketches, you can use the following:

- Open CrossCore Embedded Studio's Example Browser which can be found in CrossCore Embedded Studio under Help. Select in the Product section "Micrium uCOS-II v 2.92.08 [1.0.0]" for a full list of examples and sketches.
- Open CrossCore Embedded Studio's System Overview page which can be found by double-clicking on the system.svc file. Select the C/OS-II add-in for a list of the available examples and sketches.

Documentation

μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio contains the processor-agnostic section of the μ C/OS-II Real-Time Kernel manual as provided by Micrium as well as the manuals for Micrium's μ C/CPU and μ C/LIB. To find these manuals go to CrossCore Embedded Studio Help > Help Contents where you can see a whole section on the Micrium products released with this add-in.

Integration with CrossCore® Embedded Studio

System view

CrossCore Embedded Studio provides the System view which is used by μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio. Use the System Configuration Overview tab to add the μ C/OS-II system add-in 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 μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio gets added to a project, several configuration tabs get added to the System view. These include tabs for Micrium components which are required by several products like μ C-LIB and a configuration tab for μ C/OS-II. These configuration tabs provide an easy mechanism to generate any macro definitions required by the Micrium products.

If the μ C/OS-II add-in is used by a CrossCore Embedded Studio project, its configuration tab includes the following pages:

- General
- Signals Options
- Signals- Messages Options
- Task Options
- Application - Timer and Tick Options

For more information about each of the configuration options see the section μ C/OS-II tab in the μ C/OS-II node in CrossCore Embedded Studio's help.

Application tasks view

μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio includes a window in which you can see the priorities and stack sizes of the tasks for all the Micrium-based products that are set up in the configuration windows. This page can be seen if you double click on the system.svc file and then select μ C/APP tab and then select Application Tasks.

RTOS Status window

To help with the debugging of μ C/OS-II applications, μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio integrates a window in CrossCore Embedded Studio's debug perspective which displays the current state of the application. When you halt execution of a μ C/OS-II program, CrossCore Embedded Studio reads data for tasks, semaphores, mutexes, message queues, event flags groups and mailboxes, and displays the state and status data in this window.

When one of the above μ C/OS-II objects is created, it is added to the display. An entity is removed from the display when it is deleted.

The following requirements must be met in order to display the contents of the RTOS Status window:

- μ C/OS-II add-in must have been added to the executable
- The "Include debug variables" tickbox must be selected in the μ C/OS-II configuration under μ C/OS-II General (enabled by default)
- The application must have been built with debug information (enabled by default in Debug configuration)

Initially, information displays in a collapsed state, showing only the name of the item. Clicking the triangle next to the item expands the view to display the existing instances and their properties including for example for tasks their state, priority, stack range etc.

You can find this window in the Debug perspective if you select Window > Show View > Other > Debug > RTOS Status

Options not supported

μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio version 1.0.0 does not allow the setting of some of the CrossCore Embedded Studio options and will raise an error if these options are set. The options supported may change in future releases. This version does not support:

- Stack overflow detection (-rtcheck-stack). CrossCore Embedded Studio can track the usage of the system stack and will report errors if the stack has overflowed. In order to provide a similar support in the RTOS case, the μ C/OS-II Real-Time Kernel needs to provide information about each task stack to CrossCore Embedded Studio so it could monitor the use of each of the tasks stacks. This information is not provided in this version of μ C/OS-II Real-Time Kernel.
- Instrumented profiling (-p). CrossCore Embedded Studio can provide a summary of cycle counts for functions within an application. In order to provide the correct information in the RTOS case, the μ C/OS-II Real-Time Kernel needs to provide information about each task. This information is not provided in this version of μ C/OS-II Real-Time Kernel so instrumented profiling cannot be set.

MISRA-C Support

The Motor Industry Software Reliability Association (MISRA) published a set of guidelines for the C programming language to promote best practice in developing safety related electronic systems in road vehicles and other embedded systems. The CrossCore® Embedded Studio compiler fully supports the MISRA-C 2004 Guidelines, and can detect violations of the rules at compile-time, link-time, and run-time.

The μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio complies with MISRA by documenting known violations of the MISRA Compliance standard. In order to be able to build applications with CrossCore Embedded Studio MISRA options, the compiler checks have been

suppressed from all the sources that are part of the kernel itself, including all files in the `system/uCOS-II/Source`, `system/uCOS-II/Ports` and `system/uCOS-II/TLS` folders. Any user sources that include Micrium header files will hit MISRA warnings/errors that will need to be disabled by the application.

Interaction with other CrossCore® Embedded Studio Software

Other add-ins for CrossCore Embedded Studio may make use of features of μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio which means that disabling those features in the μ C/OS-II configuration window might stop projects which include those add-ins from linking.

Thread-safe runtime libraries

Adding μ C/OS-II Real-Time Kernel for CrossCore Embedded Studio to a project switches automatically to use the thread-safe runtime libraries. These libraries use semaphores to protect certain I/O structures and a TLS slot for storage of information like `errno` or variables related to `rand()`, `strtok()` etc.

Removing μ C/OS-II semaphores or TLS support from an application could cause link errors if these are used by the thread-safe runtime library.

System Services and Device Driver libraries

Although the System Services and Device Drivers functionality does not require an RTOS, if μ C/OS-II is in the application then it uses some objects to ensure its thread-safety. Certain services or device drivers use semaphores or mutexes.

Removing μ C/OS-II mutex or semaphore support from an application could cause link errors if these are used by the System Services/Device Drivers library.

MCAPI

Although the Multi-core API (MCAPI) functionality does not require an RTOS, if μ C/OS-II is in the application then it uses some objects to co-ordinate between tasks. Analog Devices' implementation of MCAPI uses Event flags.

Removing μ C/OS-II flags support from an application could cause link errors if these are used by MCAPI.

μ C/FS™ File System for CrossCore® Embedded Studio

Although μ C/FS File System for CrossCore Embedded Studio does not require an RTOS, if μ C/OS-II is in the application then it uses some objects to ensure its behavior in a multi-threaded environment. In its first release μ C/FS requires semaphores, and 2 task-specific registers if working-directory functionality is enabled. Under certain conditions the API `OSTimeDlyHMSM()` can also be required.

Removing μ C/OS-II semaphores, task-specific registers or `OSTimeDlyHMSM` support from an application could cause link errors if these are used by μ C/FS.

μ C/USB Device™ Stack for CrossCore® Embedded Studio

μC/USB Device Stack for CrossCore Embedded Studio requires the presence of an RTOS. Currently only μC/OS-III is supported by μC/USB Device Stack for CrossCore Embedded Studio. μC/OS-II will be supported by μC/USB Device Stack for CCES in a future release.

lwIP Lightweight TCP/IP Stack for CrossCore® Embedded Studio

lwIP Lightweight TCP/IP Stack for CrossCore Embedded Studio requires the presence of an RTOS. The current release of lwIP creates multiple tasks at the same priority and therefore cannot be used with μC/OS-II without modifications to the lwIP product.

Common Micrium Components

There are several CrossCore® Embedded Studio add-ins based on Micrium'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 v1.0.2. This installation includes μC/CPU version 1.29.01.
- μC/LIB which is installed in %COMMONPROGRAMFILES%\Analog Devices\uC-LIB v1.0.2. This installation includes μC/LIB version 1.37.00.

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

Known issues with μC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio

This section enumerates the most significant anomalies which relate to μC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio. For a comprehensive list of all the public software anomalies visit <http://www.analog.com/SoftwareAnomalies>

These are the currently known problems which affect μC/OS-II™ Real-Time Kernel for CrossCore® Embedded Studio.

- TAR-50724 ucos2: Status window does not display memory partitions

The μC/OS-II Status window does not display any memory partitions. Memory partition APIs are available but they cannot be debugged with the Status window.