FreeRTOS For Analog Devices Processors: User's Guide

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

Contents

1 Introduction

The Analog Devices FreeRTOS product is an add-on to the FreeRTOS Real-time operating system that provides additional support for the Analog Devices ADuCM3029/4050 EZ-Kit board,ADSP-SC589/ADSP-SC584/ADSP-SC573 EZ-Kit board, BF707 EZ-Kit board.

This User's Guide document provides instructions on getting started with FreeRTOS for these boards using the following development environments:

- CrossCore Embedded Studio
- IAR Embedded Workbench
- Keil MDK5

This product also relies on the FreeRTOS Operating System product. In order to avoid confusion the FreeRTOS components from Analog Devices are always referred to as the Analog Devices FreeRTOS product or the FreeRTOS product from Analog Devices.
The components available from FreeRTOS are simply referred to as the FreeRTOS product.

1.1 What is included in the Analog Devices FreeRTOS product

The Analog Devices FreeRTOS product contains FreeRTOS example applications for Analog Devices processors. It is intended to be installed on top of version 9.0.0 of the FreeRTOS product from FreeRTOS.

2 Requirements for using Analog Devices FreeRTOS

To run the included examples users should ensure that they have the following software and hardware:

- FreeRTOS version 9.0.0 from freertos.org
- Analog Devices FreeRTOS product from analog.com

For developers using CrossCore Embedded Studio:

- Analog Devices CrossCore Embedded Studio version 2.6.0 or later
- ADuCM3029/4050 EZ-Kit board
- DFP for CCES
    - ADuCM3029 EZ-Kit: ADuCM302x Software for Keil version 1.0.6 or later (http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/eval-aducm3029-ezkit.html#eb-relatedsoftware)
    - ADuCM4050 EZ-Kit: ADuCM4x50 Device Family Pack version 1.0.0 or later (http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/ADZS-U4050LF-EZKIT.html#eb-relatedsoftware)
- ICE 1000 or ICE 2000 emulator
- Note: CCES and KEIL use same DFP package. Unzip downloaded file and get pack file. For CCES, this pack file should be imported by CMSIS Pack Manager. CMSIS Pack Manager can be viewed on menu bar after this be enabled by "Open Perspective" menu which in menu bar.

For developers using IAR Embedded Workbench:

- IAR Embedded Workbench version 7.60 or later
- ADuCM3029/4050 EZ-Kit board
- BSP for IAR
    - ADuCM3029 EZ-Kit: ADuCM302x EZ-Kit Lite BSP for IAR version 1.0.6 (http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/eval-aducm3029-ezkit.html#eb-documentation)
    - ADuCM4050 EZ-Kit: ADuCM4x50 BSP for IAR version 1.1.0 (http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/ADZS-U4050LF-EZKIT.html#eb-relatedsoftware)
- J-Link connector

- Note: CCES and KEIL use same DFP package. Unzip downloaded file and get pack file. For KEIL, double click pack file and install it.

For developers using Keil MDK:

- Keil MDK for ARM processors version 5.21a or later
- ADuCM3029/4050 EZ-Kit board
- BSP for Keil
    - ADuCM4050 EZ-Kit: ADuCM4x50 EZ-KIT Board Support Pack version 1.0.0 (http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/ADZS-U4050LF-EZKIT.html#eb-relatedsoftware)
- DFP for Keil
    - ADuCM3029 EZ-Kit: ADuCM302x Software for Keil version 1.0.6 (http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/eval-aducm3029-ezkit.html#eb-relatedsoftware)
    - ADuCM4050 EZ-Kit: ADuCM4x50 Device Family Pack version 1.0.0 (http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/ADZS-U4050LF-EZKIT.html#eb-relatedsoftware)
- J-Link connector
- USB connector (for USB-to-UART connector)
- Note: Both DFP and BSP should be installed for ADuCM4050 EZ-Kit.

3 Obtaining the FreeRTOS Operating System

The FreeRTOS real-time operating system product can be downloaded from FreeRTOS website at the following URL: http://www.freertos.org/a00104.html.

This product has been developed to work with version 9.0.0 of the FreeRTOS product and it is strongly recommended that you use this version of the operating system.

The FreeRTOS Operating System product can be downloaded as either a zip file or as a zip executable. Either version will suffice.

4 Installing the FreeRTOS product for Analog Devices

To install the Analog Devices FreeRTOS product you will need to first unzip the FreeRTOS product and then install the Analog Devices FreeRTOS product on top of it:

1. Unzip the FreeRTOS product into a new directory.
   We refer to this throughout the User Guide as the FreeRTOS installation directory.
2. Unzip the Analog Devices FreeRTOS product on top of the FreeRTOS product, in the FreeRTOS installation directory.

5 Running the Examples on the ADuCM302x EZ-Kit

The FreeRTOS product for Analog Devices processors contains the following examples:

Processor	Toolchain	Example(s)
ADuCM3029	CrossCore Embedded Studio	Basic Demo
ADuCM3029	IAR Embedded Workbench	Basic Demo
ADuCM3029	Keil MDK	Basic Demo

The basic demo example is based on the **Standard Demo Tasks** that FreeRTOS recommend are provided for each port of the FreeRTOS Operating System.

For more information on the Standard Demo Tasks please refer to http://www.freertos.org/a00013.html.

The tasks performed in the Analog Devices Basic Demo include:

- LED flash
- Polled queue tasks
- Recursive Mutex tasks
- Blocking Queue tasks
- Statically allocated tasks
- Suicidal tasks

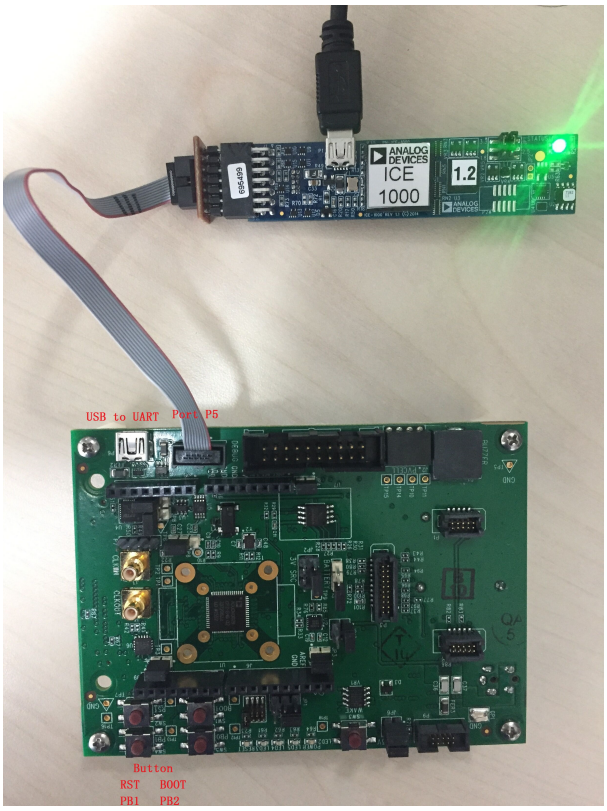
5.1 Running the Basic Example for ADuCM302x EZ-Kit with CrossCore Embedded Studio

These instructions assume that you are familiar with basic tasks in CrossCore Embedded Studio such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

To run the application within CrossCore Embedded Studio:

1. Connect the ICE1000 or ICE2000 emulator to DEBUG P5 port of EZ-Kit and the host PC as in the diagram below:



2. Start CrossCore Embedded Studio
3. Import the FreeRTOS example into CrossCore Embedded Studio:
    a. Select the **File** menu and then select the **Import** option from the menu
    b. When the **Import** project window appears:
        i. Click on the **General** folder, then click on the **Existing Projects into Workspace** entry, and click **Next**
        ii. Click the **Select root directory** radio button and then click the **Browse** button
        iii. Browse the root folder where you previously installed the FreeRTOS product and then browse down into the **FreeRTOSv9.0.0 \FreeRTOS\Demo\CORTEX_M3_ADuCM302x_CCES** folder
        iv. Click **Finish** to close the file browser dialog
        v. A single project should appear in the **projects** pane of the **Import** window
        vi. Check the entry in the **projects** pane and click **Import**

4. Build the project in CrossCore Embedded Studio:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES** project and select the **Build Project** option from the menu
5. Run the example:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES** project and select the **Debug As** option from the menu
    b. From the popup menu select **Debug Configurations** option to create a new debug configuration that matches your emulator and target board
    c. Click the **Debug** button to close the **Debug Configuration** window
    d. Click the **Run/Resume** button to start running your application

Output from the application should be visible within the **Console** window in the CrossCore Embedded Studio IDE. You should see three LEDs on the EZ-Kit begin to flash. **The test is ok for <num> round(s)** will be printed constantly and **Test Passed** which means the test is successful will be printed after 25 rounds in the console.

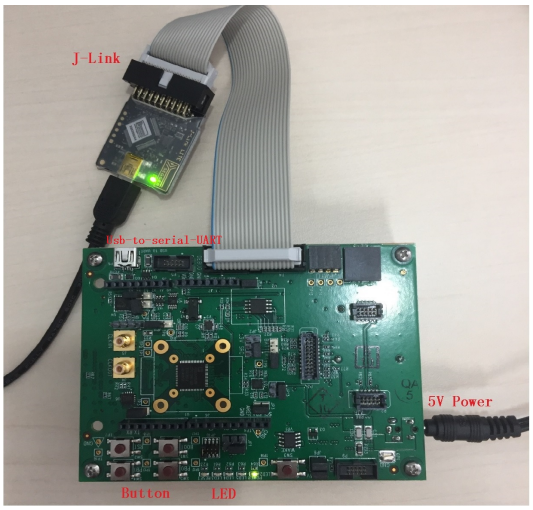
5.2 Running the Basic Example for ADuCM302x EZ-Kit with IAR Embedded Workbench

These instructions assume that you are familiar with basic tasks in IAR Embedded Workbench such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

To run the application within IAR Embedded Workbench:

1. Connect DEBUG P4 port of the EZ-KIT board to a PC running IAR Embedded WorkBench, using the J-Link.



2. Before starting the IAR Workbench you will need to configure an environment variable: **ADI_CM302x_BSP_PATH**: This should point to the ADuCM302x_EZ_Kit/ directory of your ADuCM302x BSP installation
3. Import the project into the IAR Workbench:
    a. In the IAR IDE select the **Add Existing Project** from the **Project Menu**
    b. Browse to the **FreeRTOSv9.0.0 \FreeRTOS\Demo\CORTEX_M3_ADuCM302x_IAR\iar** folder within the FreeRTOS product directory and import the project
4. Build the project within the IAR workbench:
    a. From the **Project** menu select the **Make** option
5. Run the application:
    a. From the **Project menu** select the **Download and Debug** option
    b. The application should load and halt at main. Continue the application to see it run.

Output from the application should be visible within the **Terminal I/O** window which can be found in View menu in the IAR Embedded Workbench IDE. You should see three LEDs on the EZ-Kit begin to flash. **The test is ok for <num> round(s)** will be printed constantly and **Test Passed** which means the test is successful will be printed after 25 rounds in the console.

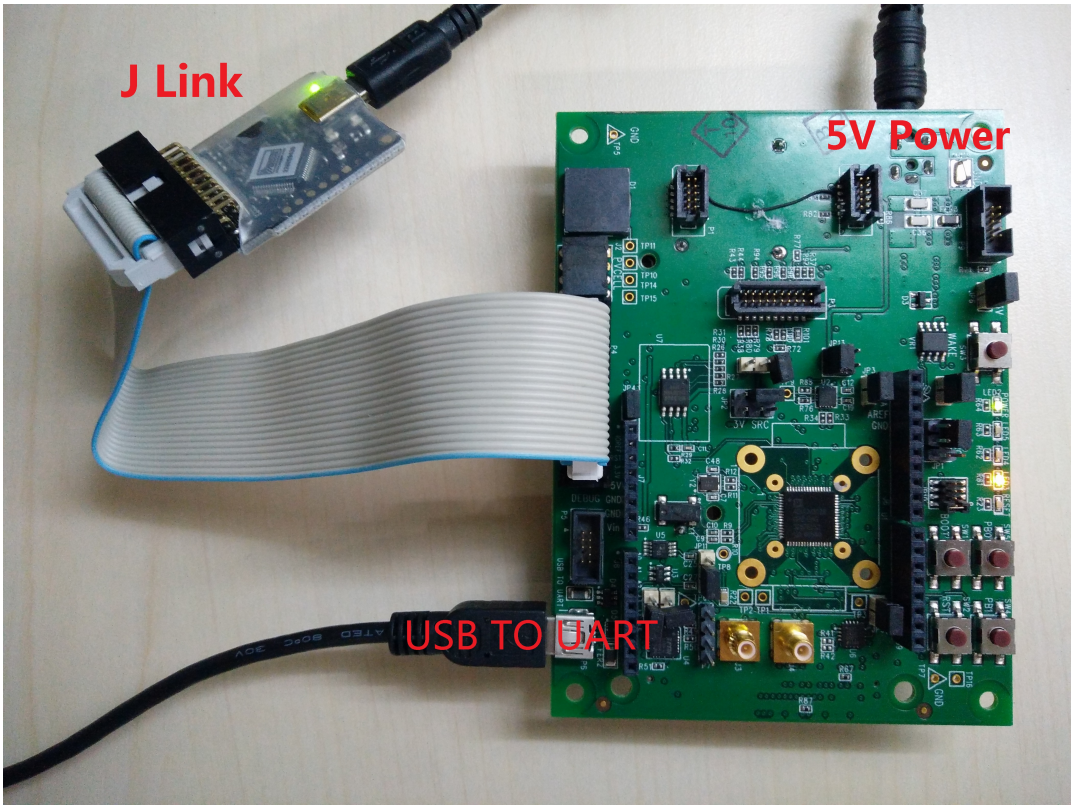
5.3 Running the Basic Example for ADuCM302x EZ-Kit with Keil MDK

These instructions assume that you are familiar with basic tasks in Keil MDK such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

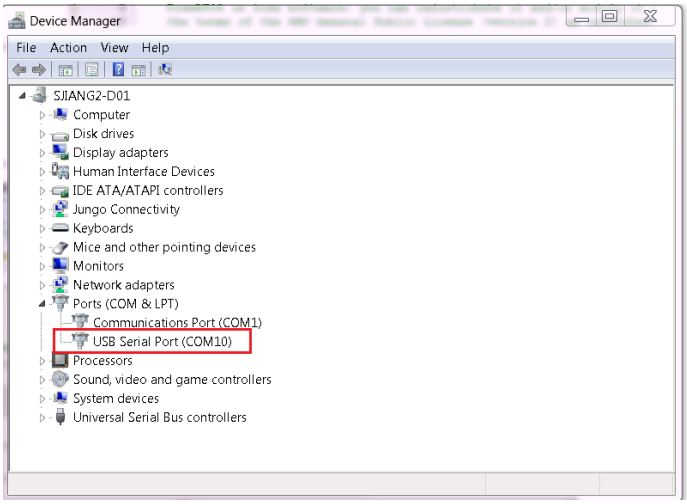
To run the application within Keil MDK:

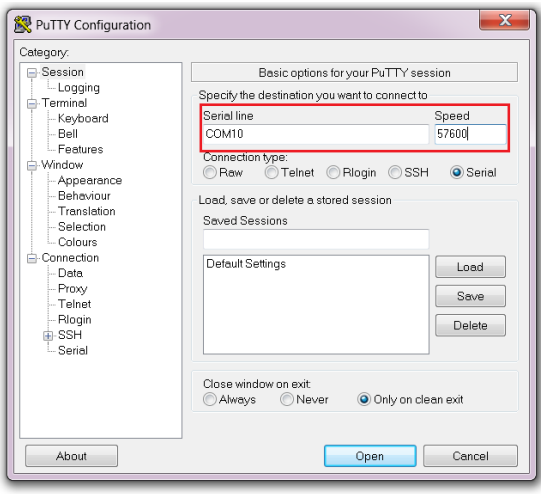
1. Connect DEBUG P4 port of the EZ-Kit to the host PC using the J-Link connector
2. Connect the **USB to UART** port on the EZ-Kit to the host PC using the USB cable provided



3. To import the project into Keil MDK:
    a. Select the **Open Project** option from the **Project** menu
    b. In the file tree window browse to the **FreeRTOSv9.0.0 \FreeRTOS\Demo\CORTEX_M3_ADuCM302x_KEIL** folder in the **FreeRTOS product installation** and select the **RTOSDemo.uvprojx** file.
    c. Click the **Open** button to import the project
4. To build the project:
    a. Select the **Build Target/Rebuild All Target Files** option from the **Project** menu

5. Configure a serial console application of your choice to view the output from the **UART to USB connection** on the HOST PC
   The easiest way to determine the correct USB device is to view the **Ports** entry in the Windows Device Manager.
   From here identify the COM port.
   Configure your serial console application to connec to the port with a baud rate of 57600





6. To run/debug the application:
   a. From the **Flash** menu select the **Download** sub-menu and then the **Start/Stop Debug Session** option
   b. Click the **Run** button to start running the application

The Keil MDK based application differs from the CrossCore Embedded Studio and IAR Workbench applications as it is unable to output and text to the console within the MDK IDE. The output of the application is transmitted via UART to the serial console.

When the application runs it will blink three LEDs on the EZ-Kit, **The test is ok for <num> round (s)** will be printed constantly and **Test Passed** which means the test is successful will be printed after 25 rounds on the console.

6 Running the Examples on the ADuCM4x50 EZ-Kit

The FreeRTOS product for Analog Devices processors contains the following examples:

Processor	Toolchain	Example(s)
ADuCM4050	CrossCore Embedded Studio	Basic Demo
ADuCM4050	IAR Embedded Workbench	Basic Demo
ADuCM4050	Keil MDK	Basic Demo

The basic demo example is based on the **Standard Demo Tasks** that FreeRTOS recommend are provided for each port of the FreeRTOS Operating System.

For more information on the Standard Demo Tasks please refer to http://www.freertos.org/a00013.html.

The tasks performed in the Analog Devices Basic Demo include:

- LED flash
- Polled queue tasks
- Recursive Mutex tasks
- Blocking Queue tasks
- Statically allocated tasks
- Suicidal tasks

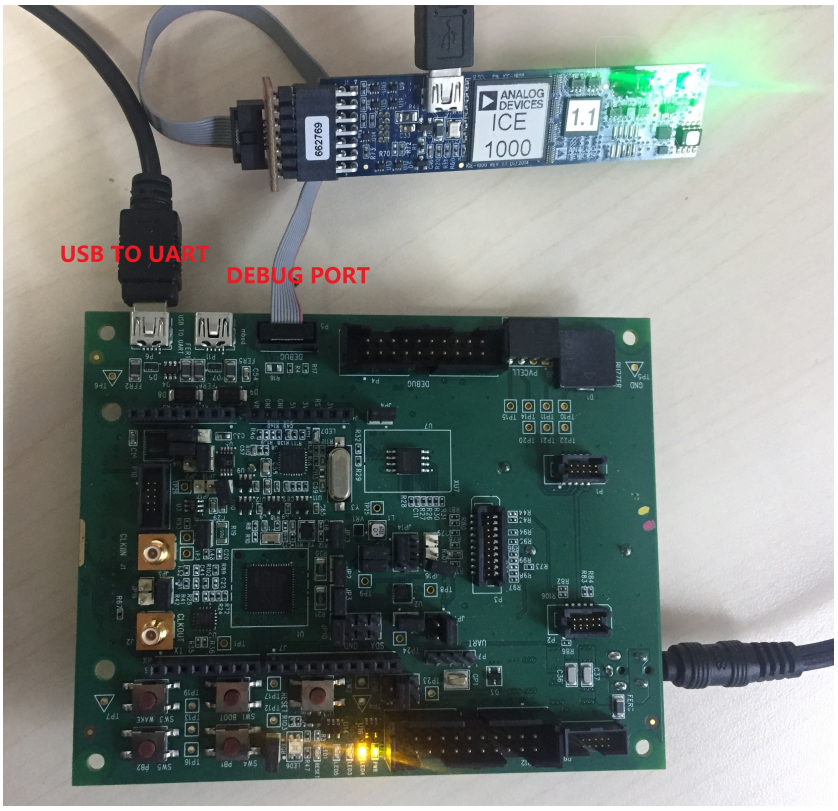
6.1 Running the Basic Example for ADuCM4x50 EZ-Kit with CrossCore Embedded Studio

These instructions assume that you are familiar with basic tasks in CrossCore Embedded Studio such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

To run the application within CrossCore Embedded Studio:

1. Connect the ICE1000 or ICE2000 emulator to DEBUG P5 port of EZ-Kit and the host PC as in the diagram below:



2. Start CrossCore Embedded Studio
3. Import the FreeRTOS example into CrossCore Embedded Studio:
    a. Select the **File** menu and then select the **Import** option from the menu
    b. When the **Import** project window appears:
        i. Click on the **General** folder, then click on the **Existing Projects into Workspace** entry, and click **Next**
        ii. Click the **Select root directory** radio button and then click the **Browse** button
        iii. Browse the root folder where you previously installed the FreeRTOS product and then browse down into the **FreeRTOSv9.0.0 \FreeRTOS\Demo\CORTEX_M4_ADuCM4x50_CCES** folder
        iv. Click **OK** to close the file browser dialog
        v. A single project should appear in the **projects** pane of the **Import** window
        vi. Check the entry in the **projects** pane and click **Import**

4. Build the project in CrossCore Embedded Studio:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES** project and select the **Build Project** option from the menu
5. Run the example:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES** project and select the **Debug As** option from the menu
    b. From the popup menu select **Debug Configurations** option to create a new debug configuration that matches your emulator
    c. Click the **Debug** button to close the **Debug Configuration** window
    d. Click the **Run/Resume** button to start running your application

Output from the application should be visible within the **Console** window in the CrossCore Embedded Studio IDE. You should see three LEDs on the EZ-Kit begin to flash. **The test is ok for <num> round(s)** will be printed constantly and **Test Passed** which means the test is successful will be printed after 25 rounds in the console.

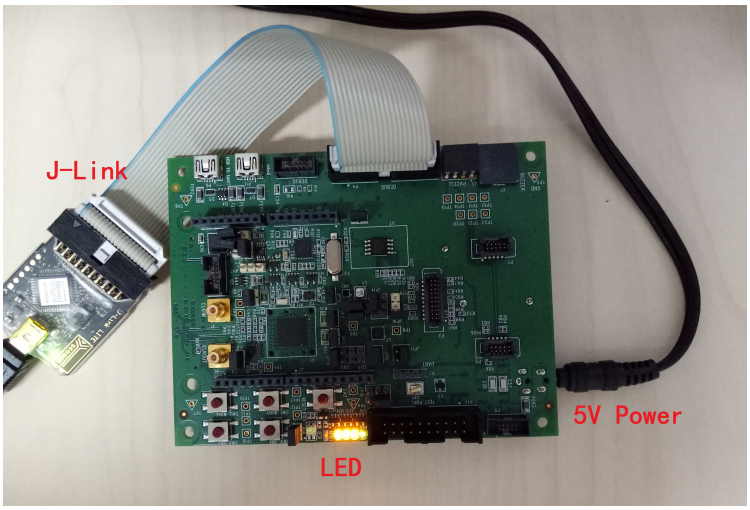
6.2 Running the Basic Example for ADuCM4x50 EZ-Kit with IAR Embedded Workbench

These instructions assume that you are familiar with basic tasks in IAR Embedded Workbench such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

To run the application within IAR Embedded Workbench:

1. Connect DEBUG P4 port of the EZ-KIT board to a PC running IAR Embedded WorkBench, using the J-Link.



2. Before starting the IAR Workbench you will need to configure an environment variable: **ADI_CM4X50_BSP_PATH**: This should point to the ADuCM4x50_EZ_Kit_Lite/ directory of your ADuCM4x50 BSP installation
3. Import the project into the IAR Workbench:
    a. In the IAR IDE select the **Add Existing Project** from the **Project Menu**
    b. Browse to the IAR demo source folder for example **FreeRTOSv9.0.0 \FreeRTOS\Demo\CORTEX_M4_ADuCM4x50_IAR** within the FreeRTOS product directory and import the project from iar folder.
4. Build the project within the IAR workbench:
    a. From the **Project** menu select the **Make** option
5. Run the application:
    a. From the **Project menu** select the **Download and Debug** option
    b. The application should load and halt at main. Continue the application to see it run.

Output from the application should be visible within the **Terminal I/O** window which can be found in View menu in the IAR Embedded Workbench IDE. You should see three LEDs on the EZ-Kit begin to flash. **The test is ok for <num> round(s)** will be printed constantly and **Test Passed** which means the test is successful will be printed after 25 rounds in the console.

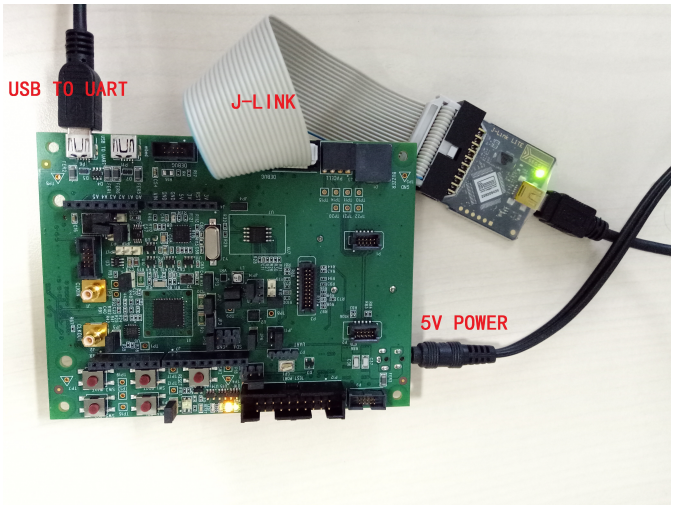
6.3 Running the Basic Example for ADuCM4x50 EZ-Kit with Keil MDK

These instructions assume that you are familiar with basic tasks in Keil MDK such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

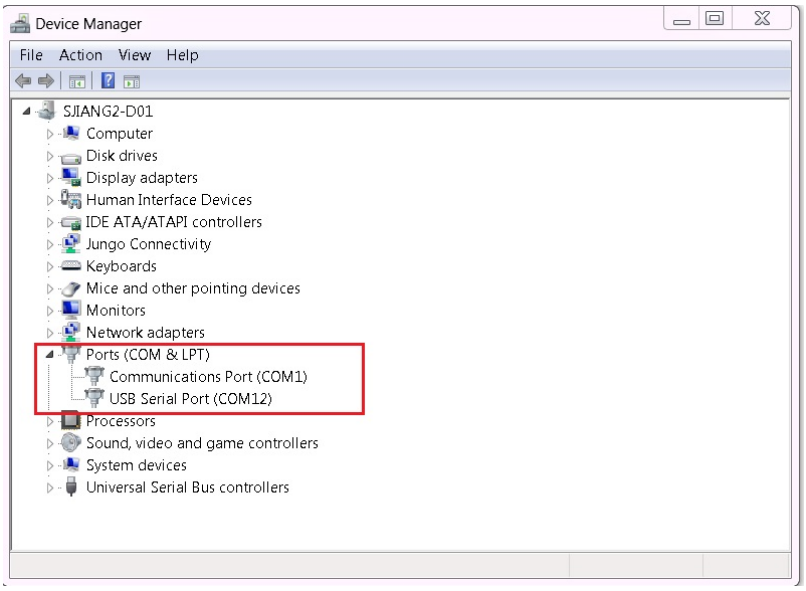
To run the application within Keil MDK:

1. Connect DEBUG P4 port of the EZ-Kit to the host PC using the J-Link connector
2. Connect the **USB to UART** port on the EZ-Kit to the host PC using the USB cable provided

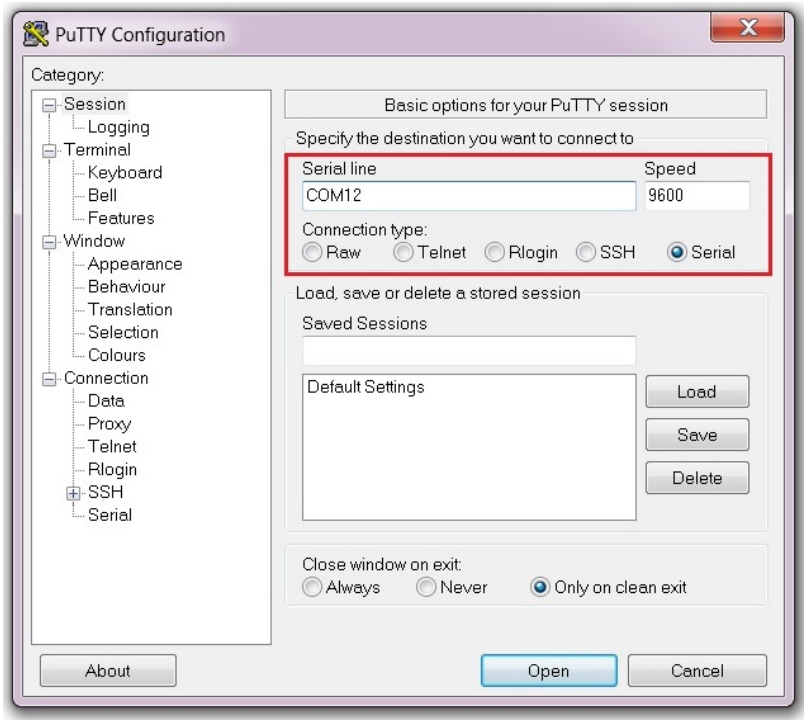


3. To import the project into Keil MDK:
    a. Select the **Open Project** option from the **Project** menu
    b. In the file tree window browse to the keil source folder for example **FreeRTOSv9.0.0 \FreeRTOS\Demo\CORTEX_M4_ADuCM4x50_KEIL** in the **FreeRTOS product installation** and select the **RTOSDemo.uvprojx** file.
    c. Click the **Open** button to import the project
4. To build the project:
    a. Select the **Build Target/Rebuild All Target Files** option from the **Project** menu

5. Configure a serial console application of your choice to view the output from the **UART to USB connection** on the HOST PC
   The easiest way to determine the correct USB device is to view the **Ports** entry in the Windows Device Manager.
   From here identify the COM port.



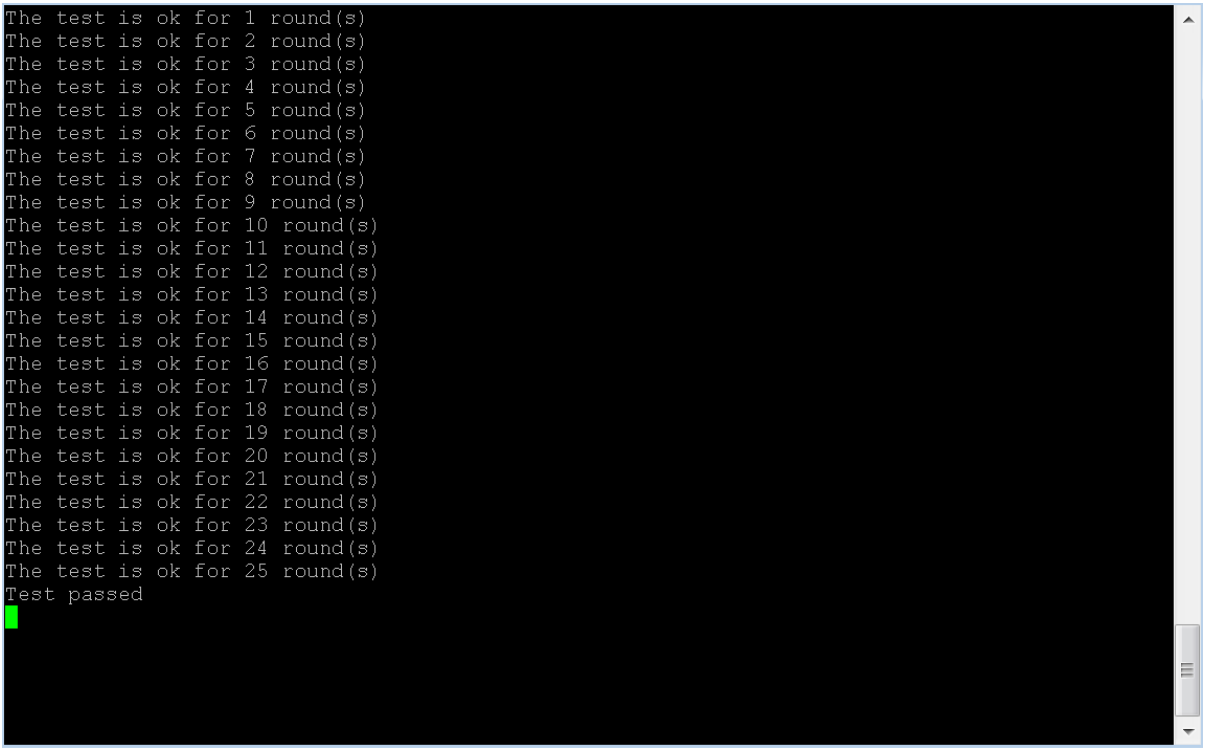
   Configure your serial console application to connect to the port with a baud rate of 9600.



6. To run/debug the application:
    a. From the **Flash** menu select the **Download** sub-menu and then the **Start/Stop Debug Session** option in the **Debug** menu
    b. Click the **Debug** button to start running the application

The Keil MDK based application differs from the CrossCore Embedded Studio and IAR Workbench applications as it is unable to output and text to the console within the MDK IDE. The output of the application is transmitted via UART to the serial console.

When the application runs it will blink three LEDs on the EZ-Kit, **The test is ok for <num> round (s)** will be printed constantly and **Test Passed** which means the test is successful will be printed after 25 rounds on the console.



7 Running the Examples on the ADSP-SC589 EZ-Kit

The FreeRTOS product for Analog Devices processors contains the following examples:

Processor	Toolchain	Example(s)
ADSP-SC589	CrossCore Embedded Studio	Basic Demo

The basic demo example is based on the **Standard Demo Tasks** that FreeRTOS recommend are provided for each port of the FreeRTOS Operating System. For more information on the Standard Demo Tasks please refer to http://www.freertos.org/a00013.html.

The tasks performed in the Analog Devices Basic Demo include:

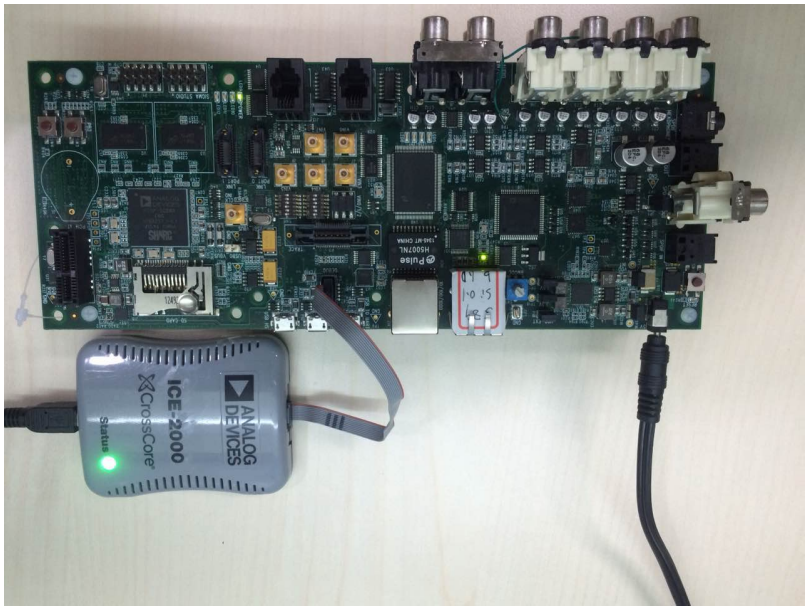
- LED flash
- Polled queue tasks
- Recursive Mutex tasks
- Blocking Queue tasks
- Statically allocated tasks
- Suicidal tasks

7.1 Running the Basic Example for ADSP-SC589 EZ-Kit with CrossCore Embedded Studio

These instructions assume that you are familiar with basic tasks in CrossCore Embedded Studio such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

To run the application within CrossCore Embedded Studio:



1. Connect the ICE1000 or ICE2000 emulator to DEBUG P3 port of EZ-Kit and the host PC as in the diagram above
2. Start CrossCore Embedded Studio
3. Import the FreeRTOS example into CrossCore Embedded Studio:
    a. Select the **File** menu and then select the **Import** option from the menu
    b. When the **Import** project window appears:
        i. Click on the **General** folder, then click on the **Existing Projects into Workspace** entry, and click **Next**
        ii. Click the **Select root directory** radio button and then click the **Browse** button
        iii. Browse the root folder where you previously installed the FreeRTOS product and then browse down into the **FreeRTOSv9.0.0 \FreeRTOS\Demo\CORTEX_A5_ADSP_SC589_CCES** folder
        iv. Click **Finish** to close the file browser dialog
        v. A single project should appear in the **projects** pane of the **Import** window
        vi. Check the entry in the **projects** pane and click **Import**
4. Build the project in CrossCore Embedded Studio:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES_Core0** project and select the **Build Project** option from the menu

5. Run the example:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES_Core0** project and select the **Debug As** option from the menu
    b. From the popup menu select **Debug Configurations** option to create a new debug configuration that matches your emulator and target board
    c. Click the **Debug** button to close the **Debug Configuration** window
    d. Click the **Run/Resume** button to start running your application

Output from the application should be visible within the **Console** window in the CrossCore Embedded Studio IDE. You should see three LEDs on the EZ-Kit begin to flash. **Test Passed** will be printed if the all tests passed.

8 Running the Examples on the ADSP-SC584 EZ-Kit

The FreeRTOS product for Analog Devices processors contains the following examples:

Processor	Toolchain	Example(s)
ADSP-SC584	CrossCore Embedded Studio	Basic Demo

The basic demo example is based on the **Standard Demo Tasks** that FreeRTOS recommend are provided for each port of the FreeRTOS Operating System. For more information on the Standard Demo Tasks please refer to http://www.freertos.org/a00013.html.

The tasks performed in the Analog Devices Basic Demo include:

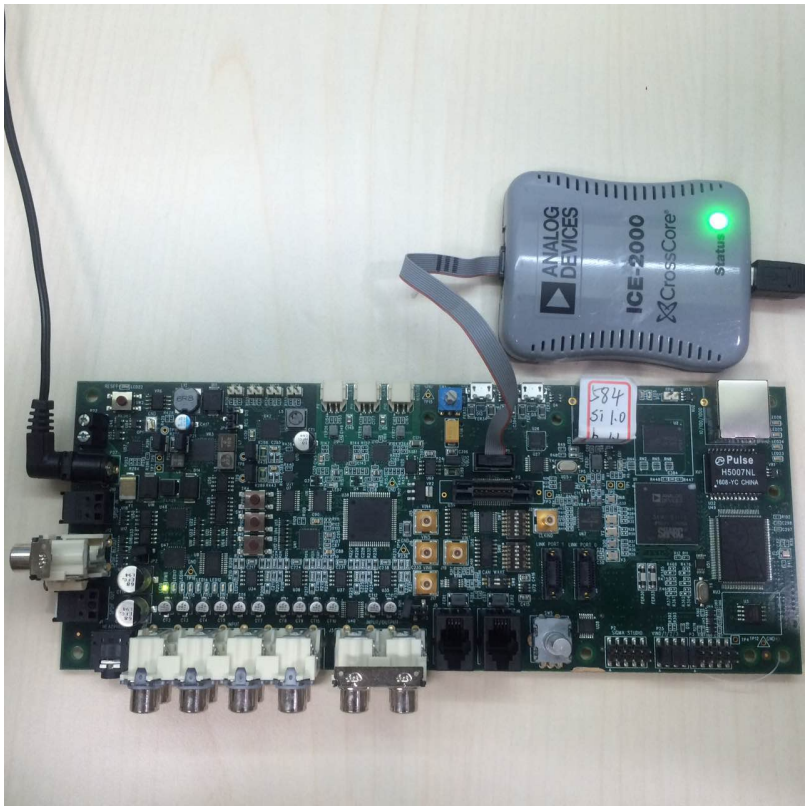
- LED flash
- Polled queue tasks
- Recursive Mutex tasks
- Blocking Queue tasks
- Statically allocated tasks
- Suicidal tasks

8.1 Running the Basic Example for ADSP-SC584 EZ-Kit with CrossCore Embedded Studio

These instructions assume that you are familiar with basic tasks in CrossCore Embedded Studio such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

To run the application within CrossCore Embedded Studio:



1. Connect the ICE1000 or ICE2000 emulator to DEBUG P1 port of EZ-Kit and the host PC as in the diagram above
2. Start CrossCore Embedded Studio
3. Import the FreeRTOS example into CrossCore Embedded Studio:
    a. Select the **File** menu and then select the **Import** option from the menu
    b. When the **Import** project window appears:
        i. Click on the **General** folder, then click on the **Existing Projects into Workspace** entry, and click **Next**
        ii. Click the **Select root directory** radio button and then click the **Browse** button
        iii. Browse the root folder where you previously installed the FreeRTOS product and then browse down into the **FreeRTOSv9.0.0 \FreeRTOS\Demo\CORTEX_A5_ADSP_SC584_CCES** folder
        iv. Click **Finish** to close the file browser dialog
        v. A single project should appear in the **projects** pane of the **Import** window
        vi. Check the entry in the **projects** pane and click **Import**

4. Build the project in CrossCore Embedded Studio:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES_Core0** project and select the **Build Project** option from the menu
5. Run the example:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES_Core0** project and select the **Debug As** option from the menu
    b. From the popup menu select **Debug Configurations** option to create a new debug configuration that matches your emulator and target board
    c. Click the **Debug** button to close the **Debug Configuration** window
    d. Click the **Run/Resume** button to start running your application

Output from the application should be visible within the **Console** window in the CrossCore Embedded Studio IDE. You should see three LEDs on the EZ-Kit begin to flash. **Test Passed** will be printed if the all tests passed.

9 Running the Examples on the ADSP-SC573 EZ-Kit

The FreeRTOS product for Analog Devices processors contains the following examples:

Processor	Toolchain	Example(s)
ADSP-SC573	CrossCore Embedded Studio	Basic Demo

The basic demo example is based on the **Standard Demo Tasks** that FreeRTOS recommend are provided for each port of the FreeRTOS Operating System.

For more information on the Standard Demo Tasks please refer to http://www.freertos.org/a00013.html.

The tasks performed in the Analog Devices Basic Demo include:

- LED flash
- Polled queue tasks
- Recursive Mutex tasks
- Blocking Queue tasks
- Statically allocated tasks
- Suicidal tasks

9.1 Running the Basic Example for ADSP-SC573 EZ-Kit with CrossCore Embedded Studio

These instructions assume that you are familiar with basic tasks in CrossCore Embedded Studio such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

To run the application within CrossCore Embedded Studio:

1. Connect the ICE1000 or ICE2000 emulator to DEBUG P1 port of EZ-Kit and the host PC as in the diagram above
2. Start CrossCore Embedded Studio
3. Import the FreeRTOS example into CrossCore Embedded Studio:
    a. Select the **File** menu and then select the **Import** option from the menu
    b. When the **Import** project window appears:
        i. Click on the **General** folder, then click on the **Existing Projects into Workspace** entry, and click **Next**
        ii. Click the **Select root directory** radio button and then click the **Browse** button
        iii. Browse the root folder where you previously installed the FreeRTOS product and then browse down into the **FreeRTOSv9.0.0 \FreeRTOS\Demo\CORTEX_A5_ADSP_SC573_CCES** folder
        iv. Click **Finish** to close the file browser dialog
        v. A single project should appear in the **projects** pane of the **Import** window
        vi. Check the entry in the **projects** pane and click **Import**

4. Build the project in CrossCore Embedded Studio:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES_Core0** project and select the **Build Project** option from the menu
5. Run the example:
    a. In the **Project Explorer** right click on the **RTOSDemo_CCES_Core0** project and select the **Debug As** option from the menu
    b. From the popup menu select **Debug Configurations** option to create a new debug configuration that matches your emulator and target board
    c. Click the **Debug** button to close the **Debug Configuration** window
    d. Click the **Run/Resume** button to start running your application

Output from the application should be visible within the **Console** window in the CrossCore Embedded Studio IDE. You should see three LEDs on the EZ-Kit begin to flash. **Test Passed** will be printed if the all tests passed.

10 Running the Examples on the ADSP-BF7XX EZ-Kit

The FreeRTOS product for Analog Devices processors contains the following examples:

Processor	Toolchain	Example(s)
ADSP-BF707	CrossCore Embedded Studio	Basic Demo

The basic demo example is based on the **Standard Demo Tasks** that FreeRTOS recommend are provided for each port of the FreeRTOS Operating System.

For more information on the Standard Demo Tasks please refer to http://www.freertos.org/a00013.html.

The tasks performed in the Analog Devices Basic Demo include:

- LED flash
- Polled queue tasks
- Recursive Mutex tasks
- Blocking Queue tasks
- Statically allocated tasks
- Suicidal tasks

10.1 Running the Basic Example for ADSP-BF707 EZ-Kit with CrossCore Embedded Studio

These instructions assume that you are familiar with basic tasks in CrossCore Embedded Studio such as building projects and loading application to the target hardware.

It is also assumed that you have installed the FreeRTOS product and the Analog Devices FreeRTOS product, following the previous instructions in this document.

To run the application within CrossCore Embedded Studio:

1. Connect the ICE1000 or ICE2000 emulator to DEBUG P3 port of EZ-Kit and the host PC as in the diagram above
2. Start CrossCore Embedded Studio
3. Import the FreeRTOS example into CrossCore Embedded Studio:
    a. Select the **File** menu and then select the **Import** option from the menu
    b. When the **Import** project window appears:
        i. Click on the **General** folder, then click on the **Existing Projects into Workspace** entry, and click **Next**
        ii. Click the **Select root directory** radio button and then click the **Browse** button
        iii. Browse the root folder where you previously installed the FreeRTOS product and then browse down into the **\FreeRTOSv9.0.0 \FreeRTOS\Demo\Blackfin_ADSP_BF707** folder
        iv. Click **Finish** to close the file browser dialog
        v. A single project should appear in the **projects** pane of the **Import** window
        vi. Check the entry in the **projects** pane and click **Import**
4. Build the project in CrossCore Embedded Studio:
    a. In the **Project Explorer** right click on the **RTOSDemo_BF707** project and select the **Build Project** option from the menu

5. Run the example:
    a. In the **Project Explorer** right click on the **RTOSDemo_BF707** project and select the **Debug As** option from the menu
    b. From the popup menu select **Debug Configurations** option to create a new debug configuration that matches your emulator and target board
    c. Click the **Debug** button to close the **Debug Configuration** window
    d. Click the **Run/Resume** button to start running your application

Output from the application should be visible within the **Console** window in the CrossCore Embedded Studio IDE. You should see three LEDs on the EZ-Kit begin to flash. **Test Passed** will be printed if the all tests passed.

11 Appendix I FreeRTOS Performance

11.1 ADZS-BF707 Benchmark

ADZS-BF707 Performance Metrics

		cycles
xEventGroupWaitBits	xEventGroupWaitBits (flag available)	442
	xEventGroupWaitBits (flag unavailable, context switch to new task)	1882
	xEventGroupSetBits (no task pending, no context switch)	386
	xEventGroupSetBits (task waiting, context switch to pending task)	1844
	xEventGroupSetBits (from an ISR, switching to a pending task)	3428
Interrupt Service Time	Interrupt service time (FreeRTOS)	141
	Time to return from an ISR (FreeRTOS, no task switch)	298
Message Queues	xQueueReceive(message available)	548
	xQueueReceive(message unavailable, context switch to new task)	3020
	xQueueSend(no task pending, no context switch)	638
	xQueueSend(task waiting, context switch to pending task)	2439

	xQueueSend(from an ISR, switching to a pending task)	1640
Mutexes	xSemaphoreTake(mutex available)	474
	xSemaphoreTake(mutex unavailable, context switch to new task	3592
	xSemaphoreGive(no task pending, no context switch)	678
	xSemaphoreGive(task waiting, context switch to pending task)	2724
Semaphores	xSemaphoreTake(semaphore available)	406
	xSemaphoreTake(semaphore unavailable, context switch to new task)	3046
	xSemaphoreGive(no task pending, no context switch)	528
	xSemaphoreGive(task waiting, context switch to pending task)	2220
	xSemaphoreGive (from an ISR, switching to a pending task)	1537

ADZS-BF707 Sizing Metrics

		Data	Code	Total
NONE	Basic project	6543	14954	21497
Message Queues	Basic project using 1 static object	6631	15026	21657
Message Queues	Basic project using 2 static objects	6715	15026	21741
Flags	Basic project using 1 static object	6575	15666	22241

Flags	Basic project using 2 static objects	6607	15666	22273
Mutexes	Basic project using 1 static object	6627	15090	21717
Mutexes	Basic project using 2 static objects	6711	15090	21801
Semaphores	Basic project using 1 static object	6627	15026	21653
Semaphores	Basic project using 2 static objects	6711	15026	21737
ALL	Basic project using 1 static object	6663	15922	22585
ALL	Basic project using 2 static objects	6779	15922	22701

11.2 ADSP-SC589 (Cortex-A Core) Benchmark

ADSP-SC589 Cortex-A Core Performance Metrics

		cycles
xEventGroupWaitBits	xEventGroupWaitBits (flag available)	326
	xEventGroupWaitBits (flag unavailable, context switch to new task)	1023
	xEventGroupSetBits (no task pending, no context switch)	323
	xEventGroupSetBits (task waiting, context switch to pending task)	1102
	xEventGroupSetBits (from an ISR, switching to a pending task)	3159
Interrupt Service Time	Interrupt service time (FreeRTOS)	87
	Time to return from an ISR (FreeRTOS, no task switch)	15

Message Queues	xQueueReceive(message available)	229
	xQueueReceive(message unavailable, context switch to new task)	2204
	xQueueSend(no task pending, no context switch)	222
	xQueueSend(task waiting, context switch to pending task)	1237
	xQueueSend(from an ISR, switching to a pending task)	586
Mutexes	xSemaphoreTake(mutex available)	259
	xSemaphoreTake(mutex unavailable, context switch to new task	2645
	xSemaphoreGive(no task pending, no context switch)	261
	xSemaphoreGive(task waiting, context switch to pending task)	1363
Semaphores	xSemaphoreTake(semaphore available)	151
	xSemaphoreTake(semaphore unavailable, context switch to new task)	2204
	xSemaphoreGive(no task pending, no context switch)	249
	xSemaphoreGive(task waiting, context switch to pending task)	1135
	xSemaphoreGive (from an ISR, switching to a pending task)	526

ADSP-SC589 Cortex-A Core Sizing Metrics

		Data	Code	Total
NONE	Basic project	97836	18480	116316
Message Queues	Basic project using 1 static object	97836	18536	116372
Message Queues	Basic project using 2 static objects	97836	18536	116372
Flags	Basic project using 1 static object	97836	18952	116788
Flags	Basic project using 2 static objects	97836	18952	116788
Mutexes	Basic project using 1 static object	97836	18544	116380
Mutexes	Basic project using 2 static objects	97836	18544	116380
Semaphores	Basic project using 1 static object	97836	18528	116364
Semaphores	Basic project using 2 static objects	97836	18528	116364
ALL	Basic project using 1 static object	97836	19112	116948
ALL	Basic project using 2 static objects	97836	19112	116948