TI-RTOS 1.21

Getting Started Guide

Literature Number: SPRUHD3G
January 2014

Contents

Read This First

About This Manual

This manual describes TI-RTOS. The version number as of the publication of this manual is v1.21.

Notational Conventions

This document uses the following conventions:

- Program listings, program examples, and interactive displays are shown in a special typeface. Examples use a bold version of the special typeface for emphasis.

  Here is a sample program listing:

```
#include <xdc/runtime/System.h>
Int main(Void)
{
    System_printf("Hello World!\n");
    return (0);
}
```

- Square brackets ([and]) identify an optional parameter. If you use an optional parameter, you specify the information within the brackets. Unless the square brackets are in a **bold** typeface, do not enter the brackets themselves.

Trademarks

Registered trademarks of Texas Instruments include Stellaris and StellarisWare. Trademarks of Texas Instruments include: the Texas Instruments logo, Texas Instruments, TI, TI.COM, C2000, C5000, C6000, Code Composer, Code Composer Studio, Concerto, controlSUITE, DSP/BIOS, SPOX, Tiva, Tivaware, TMS320, TMS320C5000, TMS320C6000 and TMS320C2000.

ARM is a registered trademark, and Cortex is a trademark of ARM Limited.

Windows is a registered trademark of Microsoft Corporation.

Linux is a registered trademark of Linus Torvalds.

IAR Systems and IAR Embedded Workbench are registered trademarks of IAR Systems AB:

All other brand or product names are trademarks or registered trademarks of their respective companies or organizations.

January 2, 2014

Chapter 1

SPRUHD3G—January 2014

About TI-RTOS

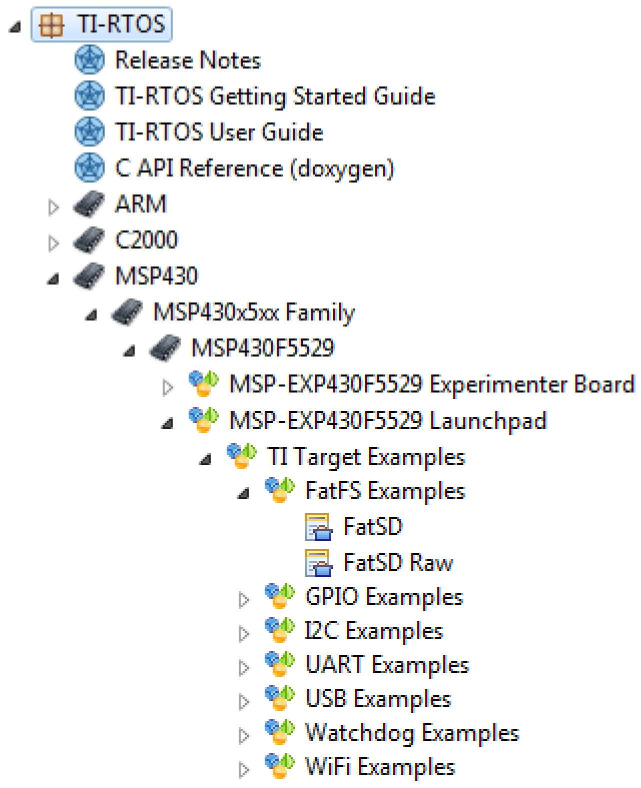
This chapter provides an overview of TI-RTOS.

Topic **Page**

1.1 What is TI-RTOS?

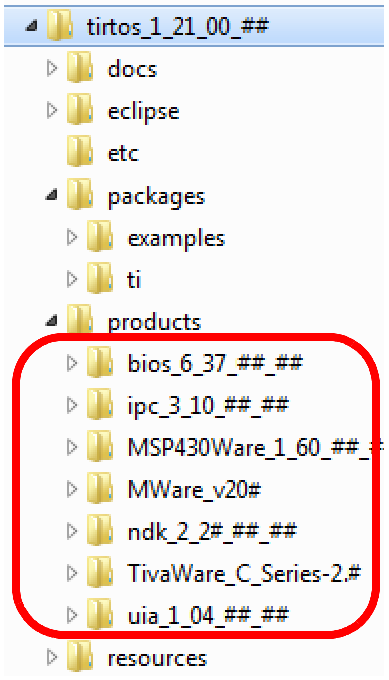
TI-RTOS is a scalable, one-stop embedded tools ecosystem for TI devices. TI-RTOS makes it easier to develop embedded applications that run on TI devices.

TI-RTOS contains several software components and examples that use these components together. It provides an OS kernel, communications support, drivers, and more. It is tightly integrated with TI's Code Composer Studio (CCS) development environment. In addition, examples demonstrate how to use each supported device and driver. These can be used as a starting point for your own projects.



1.2 What are the TI-RTOS Components?

TI-RTOS contains its own source files, pre-compiled libraries (both instrumented and non-instrumented), and examples. Additionally, TI-RTOS contains a number of components within its "`products`" subdirectory as shown here.



```
tirtos_1_21_00_##
  docs
  eclipse
  etc
  packages
    examples
    ti
  products
    bios_6_37_##_##
    ipc_3_10_##_##
    MSP430Ware_1_60_##_#
    MWare_v20#
    ndk_2_2#_##_##
    TivaWare_C_Series-2.#
    uia_1_04_##_##
  resources
```

The components in the "`products`" subdirectory are:

- **SYS/BIOS.** SYS/BIOS is a scalable real-time kernel. It is designed to be used by applications that require real-time scheduling and synchronization or real-time instrumentation. It provides preemptive multi-threading, hardware abstraction, real-time analysis, and configuration tools. SYS/BIOS is designed to minimize memory and CPU requirements on the target. The FatFs module used by several examples is part of SYS/BIOS.
- **IPC.** This is a component containing packages that are designed to allow communication between processors in a multi-processor environment and communication to peripherals. This communication includes message passing, streams, and linked lists. These work transparently in both uni-processor and multi-processor configurations.
- **MSP430Ware.** This software is an extensive suite of drivers, code examples, and design resources designed to simplify and speed development of MSP430 microcontroller applications. This component is customized and built as a library to support MSP430F5xx_6xx devices with TI-RTOS.[1]
- **MWare.** The M3 portion of controlSUITE. It includes low level drivers and examples.[2]
- **NDK.** The Network Developer's Kit (NDK) is a platform for development and demonstration of network enabled applications on TI embedded processors, currently limited to the TMS320C6000 family and ARM processors.
- **TivaWare.** This software is an extensive suite of software designed to simplify and speed development of Tiva-based microcontroller applications. (TivaWare was previously called StellarisWare.) This component is rebuilt to include only the portions required by TI-RTOS.[2]
- **UIA.** The Unified Instrumentation Architecture provides target content that aids in the creation and gathering of instrumentation data (for example, Log data).
- **XDCtools.** This component provides the underlying tooling for configuring and building SYS/BIOS, IPC, NDK, and UIA. TI-RTOS installs XDCtools only if the version needed by TI-RTOS has not already been installed as part of a CCS or SYS/BIOS installation. If TI-RTOS installs XDCtools, it places it in the top-level CCS directory (for example, c:\ti), not the TI-RTOS products directory.

To see the release notes for each component, you can select a component in the TI Resource Explorer under **TI-RTOS > Products**.

TI-RTOS installs versions of these components that have been reduced in size by removing files that apply only to device families not supported by TI-RTOS.

1. Customization details for MSP430Ware and instructions for building MSP430Ware's driverlib libraries for MSP430F5xx6xx families are provided in the TI-RTOS.README in the top-level directory of the MSP430Ware component.

2. The MWare and TivaWare libraries distributed with TI-RTOS have been rebuilt with the following compiler option: --define=USE_RTOS. See the TI-RTOS.README file in the top-level folders of the MWare and TivaWare components of TI-RTOS for details.

1.3 What Boards and Devices Have TI-RTOS Examples?

Currently, TI-RTOS provides examples for the following boards:

Family	Device on Board	Board
Concerto (ARM M3 + DSP 28x)	F28M35H52C1	TMDXDOCKH52C1 Experimenter Kit
Concerto (ARM M3 + DSP 28x)	F28M36P63C2	TMDXDOCK28M36 Experimenter Kit
ARM (Tiva)	TM4C123GH6PM	EK-TM4C123GXL LaunchPad
ARM (Stellaris)	LM4F120H5QR	EK-LM4F120XL LaunchPad (earlier version of EK-TM4C123GXL LaunchPad)
ARM (Tiva)	TM4C123GH6PGE	DK-TM4C123G Evaluation Kit
ARM (Stellaris)	LM4F232H5QD	EKS-LM4F232 Evaluation Kit (earlier version of DK-TM4C123G Evaluation Kit)
ARM (Tiva)	TM4C129XNCZAD	DK-TM4C129X Evaluation Kit
MSP430F5xx/6xx	MSP430F5529	MSP-EXP430F5529LP LaunchPad
MSP430F5xx/6xx	MSP430F5529	MSP-EXP430F5529 Experimenter Board

Both M3 and 28x sides of Concerto boards are supported.

If you want to use any of these components with other device families, you will need to download and install the complete component separately. For example, if you want to use SYS/BIOS with a C64x+ device, you will need to download and install the full SYS/BIOS product.

Examples are provided specifically for the supported boards, but libraries are provided for each of these device families, so that you can port the examples to similar boards. Porting information for TI-RTOS is provided on the Texas Instruments Wiki, including a topic on Creating TI-RTOS Projects for Other MSP430 Devices.

For MSP430 devices, TI-RTOS drivers are prebuilt only for MSP430F5529 devices. TI-RTOS drivers for other MSP430 devices in the MSP430F5xx and MSP430F6xx families must be added into the TI-RTOS build system. See "Rebuilding MSP430Ware's driverlib for TI-RTOS and Its Drivers" in the TI-RTOS User’s Guide (SPRUHD4) for details.

1.4 What Drivers Does TI-RTOS Include?

TI-RTOS includes drivers for the following peripherals. These drivers are in the `<install_dir>/packages/ti/drivers` directory. TI-RTOS examples show how to use these drivers. Note that all of these drivers are built on top of MWare, MSP430Ware, and TivaWare.

- **EMAC.** Ethernet driver used by the networking stack (NDK) and not intended to be called directly.
- **I^{2}C.** API set intended to be used directly by the application or middleware.
- **GPIO.** API set intended to be used directly by the application or middleware to manage the GPIO interrupts, pins, and ports.
- **SPI.** API set intended to be used directly by the application or middleware to communicate with the Serial Peripheral Interface (SPI) bus. SPI is sometimes called SSI (Synchronous Serial Interface).
- **SDSPI.** Driver for SD cards using a SPI (SSI) bus. This driver is used by the FatFS and not intended to be called directly by the application.
- **UART.** API set intended to be used directly by the application to communicate with the UART.

- **USBMSCHFatFs.** USB MSC Host under FatFS (for Flash drives). This driver is used by FatFS and is not intended to be called directly by the application.
- **Other USB functionality.** See the USB examples for reference modules that provide support for the Human Interface Device (HID) class (mouse and keyboard) and Communications Device Class (CDC). This code is provided as part of the examples, not as a separate driver.
- **Watchdog.** API set intended to be used directly by the application or middleware to manage the watchdog timer.
- **WiFi.** Driver used by a Wi-Fi device's host driver to exchange commands, data, and events between the host MCU and the wireless network processor. Not intended to be called directly.

In addition, TI-RTOS provides the following MessageQ transport:

- **SPIMessageQTransport.** MessageQ transport for the SPI driver for use in multicore applications that use the IPC component.

1.5 What Examples Are Available?

A number of examples are provided with TI-RTOS. These use sub-components provided with TI-RTOS. The examples are listed in Section 3.4.

There is a separate <*example_name*>_readme file for each of the examples. These files are added to your CCS project when you use the TI Resource Explorer to create a project. You can open the <*example_name*>_readme file within CCS.

The examples share the following features:

- Most examples use the SysMin System Support module. See the readme files in the individual example projects for details.
- The empty, demo, and most UART examples use the ti.uia.sysbios.LoggingSetup module with stop mode data collection. The UART Console example uses run-time data collection during Idle thread processing. For more details on data collection, see Chapter 2 of the TI-RTOS User’s Guide (SPRUHD4).
- All examples have the same <board>.c and <board>.h files. These files perform board-specific configuration of the drivers provided by TI-RTOS. For more details, see Chapter 4 of the TI-RTOS User’s Guide (SPRUHD4).

1.6 For More Information

To learn more about TI-RTOS and the software components used with it, refer to the following documentation. In addition, you can select a component in the TI Resource Explorer under **TI-RTOS > Products** to see the release notes for that component.

- **TI-RTOS**
  - TI-RTOS User’s Guide (SPRUHD4)
  - TI-RTOS on the Texas Instruments Wiki
  - TI-RTOS forum on TI’s E2E Community
  - TI-RTOS Porting Guide
  - Embedded Software Download Page

- **Code Composer Studio (CCS)**
  - CCS online help
  - CCSv5 on the Texas Instruments Wiki
  - Code Composer forum on TI's E2E Community
- **SYS/BIOS**
  - SYS/BIOS 6 Getting Started Guide at <*tirtos_install*>/products/bios_#_##_##_##/docs/Bios_Getting_Started_Guide.pdf
  - SYS/BIOS User's Guide (SPRUEX3)
  - SYS/BIOS online reference (also called "CDOC"). Open from CCS help or run <*tirtos_install*>/products/bios_#_##_##_##/docs/cdoc/index.html.
  - SYS/BIOS on the Texas Instruments Wiki
  - TI-RTOS forum on TI's E2E Community
  - SYS/BIOS 6.x Product Folder
- **XDCtools**
  - XDCtools online reference. Open from CCS help or run <*xdc_install*>/docs/xdctools.chm.
  - RTSC-Pedia Wiki
  - TI-RTOS forum on TI's E2E Community
- **IPC**
  - IPC User's Guide (SPRUGO6)
  - IPC online API reference. Run <*tirtos_install*>/products/ipc_#_##_##_##/docs/doxygen/index.html.
  - IPC online configuration reference. Open from CCS help or run <*tirtos_install*>/products/ipc_#_##_##_##/docs/cdoc/index.html.
- **NDK**
  - NDK User's Guide (SPRU523)
  - NDK Programmer's Reference Guide (SPRU524)
  - NDK online API reference. Run <*tirtos_install*>/products/ndk_#_##_##_##/docs/doxygen/html/index.html.
  - NDK online configuration reference. Open from CCS help or run <*tirtos_install*>/products/ndk_#_##_##_##/docs/cdoc/index.html.
  - NDK on the Texas Instruments Wiki
  - TI-RTOS forum on TI's E2E Community
- **UIA**
  - System Analyzer User's Guide (SPRUH43)
  - UIA online reference. Open from CCS help or run <*tirtos_install*>/products/uia_#_##_##_##/docs/cdoc/index.html.
  - System Analyzer on the Texas Instruments Wiki

- **MWare and controlSUITE**
  - Documents in <*tirtos_install*>/products/MWare_##_##_##_##/doc
  - controlSUITE on the Texas Instruments Wiki
  - controlSUITE Product Folder
- **MSP430Ware**
  - Documents in <*tirtos_install*>/products/MSP430Ware_##/doc
  - MSP430Ware Product Folder
- **TivaWare**
  - Documents in <*tirtos_install*>/products/TivaWare_C_Series-#.#/docs
  - TivaWare Product Folder
  - Online StellarisWare Workshop
- **FatFS API**
  - Open source documentation
  - FatFS for SYS/BIOS wiki page
  - SYS/BIOS online reference (also called "CDOC").
    Open from CCS help or run <*tirtos_install*>/products/bios_#_##_##_##/docs/cdoc/index.html.
    Navigate to the ti.sysbios.fatfs.FatFS module topic in the SYS/BIOS API reference documentation.
- **General microcontroller information**
  - Microcontrollers forum on TI's E2E Community
- **Concerto boards and devices**
  - Concerto F28M35x Technical Reference Manual
  - Concerto F28M36x Technical Reference Manual
  - C2000 on the Texas Instruments Wiki
  - Concerto on the Texas Instruments Wiki
  - Concerto Product Folder
  - H52C1 Concerto Experimenter Kit
  - F28M35H52C Concerto Microcontroller datasheets
  - H63C2 Concerto Experimenter Kit
  - F28M36P63C2 Concerto Microcontroller datasheets
- **MSP430 boards and devices**
  - MSP-EXP430F5529LP LaunchPad Evaluation Kit
  - MSP430 USB Developers Package (which includes the USB Descriptor Tool)
  - MSP430F5529 Microcontroller

- **Tiva boards and devices**
  - Tiva C Series TM4C123G LaunchPad Evaluation Kit
  - TM4C123GH6PM Tiva C Series Microcontroller
  - EKS-LM4F232 Evaluation Kit
  - TM4C123GH6PGE Tiva C Series Microcontroller
- **BoosterPacks**
  - EM Adapter BoosterPack
  - RF430CL330H NFC Dynamic Tag Target Board
  - SD Card BoosterPack
  - TMP006 BoosterPack
  - I2C TPL0401EVM BoosterPack
- **SD Cards**
  - Specification
- **I^{2}C**
  - Specification
- **WiFi**
  - SimpleLink Wi-Fi CC3000 Wiki
  - CC3000 Product Folder
  - SimpleLink Wi-Fi SmartConfig Apps

Installing TI-RTOS

This chapter covers the required steps needed to install and build TI-RTOS.

Topic **Page**

2.1 System Requirements

The Windows version of TI-RTOS can be installed on systems running Windows 7, Windows Vista, or Windows XP (SP2 or SP3).

A Linux version of TI-RTOS is also available.

In order to install TI-RTOS, you must have at least 2 GB of free disk space. (If you have not yet installed Code Composer Studio, you will also need at least 2 GB of disk space for that installation.)

2.2 Installing Code Composer Studio

TI-RTOS is used in conjunction with Code Composer Studio 5.5 or higher. (TI-RTOS can also be used with the IAR Embedded Workbench IDE. See page 15 for more information.)

We strongly recommend that you install CCS in the default installation directory of `c:\ti`. If you install in c:\Program Files (or c:\Program Files (x86) with Windows 7), you are likely to run into problems related to Windows security permissions.

Note: Do not install CCS in a location that contains any spaces in the full path. For example, CCS should not be installed in c:\Program Files. Makefiles may not function correctly with directory paths that include spaces.

To install CCS 5.5, go to the product page at http://www.ti.com/tool/ccstudio and follow a link to download the software for your license type.

Run the executable installer, and answer the prompts as appropriate. We strongly recommend that you install CCS in the default installation directory of `c:\ti`.

When you are prompted for the type of Setup, select "Complete Feature Set".

Note: TI-RTOS installs versions of SYS/BIOS and XDCtools that may be newer than the versions installed by CCS. For TI-RTOS to work properly, you should use the software versions delivered with TI-RTOS.

2.3 Installing TI-RTOS

TI-RTOS product comes delivered as an installer that needs to be installed in Code Composer Studio's installation directory. See the Embedded Software Download Page to download the latest installers.

2.3.1 Installing TI-RTOS on Windows for Use in Code Composer Studio

To install TI-RTOS on a Windows host, follow these steps:

1. Exit from CCS if it is currently open.
2. Download the installer for TI-RTOS. For example, tirtos_setupwin32_1_##_##_##.exe.
3. Run the downloaded file to install the full TI-RTOS product in the directory where CCS 5.5 is installed. By default, this is `c:\ti`. This is the recommended location for installing TI-RTOS.

   Note: TI-RTOS installs XDCtools only if the version needed by TI-RTOS has not already been installed as part of a CCS or SYS/BIOS installation. If TI-RTOS installs XDCtools, it places it in the top-level CCS directory (for example, c:\ti), not the TI-RTOS products subdirectory. Note that this version has been reduced in size by removing support for target families not supported by TI-RTOS.
4. Start CCS 5.5 or higher.
5. Wait for CCS to scan for newly installed products and display the **Extension Sites** window to notify you of the products that have been discovered. You should see the TI-RTOS installation directory listed (for example, `c:\ti\tirtos_1_##_##_##` by default). Leave the TI-RTOS item checked, and click **Finish** to add it to CCS.
6. You will see a window that asks if you want to restart CCS now. Click **Yes**.

2.3.2 If TI-RTOS is Installed Outside CCS on Windows

If you have installed TI-RTOS in a directory outside the Code Composer Studio installation and do not want to reinstall TI-RTOS in the correct location, you can follow these steps to make CCS use TI-RTOS correctly.

1. In CCS, choose **Window > Preferences** from the menus.
2. Select the **Code Composer Studio > RTSC > Products** category in the left pane.
3. Click the **Add** button next to the Tool discovery path.
4. Select the folder where you installed TI-RTOS and click **OK**.
5. Restart CCS.
6. In CCS, choose **Window > Preferences** from the menus again.
7. Select the **Code Composer Studio > RTSC** category in the left pane.
8. In the **Products and Repositories** tab, expand the **TI-RTOS** item and check the box next to the version of TI-RTOS you installed. Click **OK**.
9. Choose **View > TI Resource Explorer** from the CCS menus.

2.3.3 *Installing TI-RTOS on Linux for Use in Code Composer Studio*

To install TI-RTOS on a Linux host, follow these steps:

1. Exit CCS if it is currently open.
2. Download the installer for TI-RTOS. For example, tirtos_setuplinux_1_##_##_##.bin.
3. You may want to log in as root before performing the installation. It is also possible to run the installation from your user account.
4. Run the downloaded file to install the full TI-RTOS product. Accept the defaults from the Linux installer. (The installer detects whether you are running it as user or root.)

   Note: TI-RTOS installs XDCtools only if the version needed by TI-RTOS has not already been installed as part of a CCS or SYS/BIOS installation. If TI-RTOS installs XDCtools, it places it in the top-level CCS directory, not the TI-RTOS products subdirectory. Note that this version has been reduced in size by removing support for target families not supported by TI-RTOS.
5. Start CCS 5.5 or higher.
6. Wait for CCS to scan for newly installed products and display the **Extension Sites** window to notify you of the products that have been discovered. You should see the TI-RTOS installation directory listed. Leave the TI-RTOS item checked, and click **Finish** to add it to CCS.

2.3.4 *Installing TI-RTOS for Use in IAR Embedded Workbench*

You can also install TI-RTOS for use with the IAR Embedded Workbench IDE. TI-RTOS provides support for both the ARM and MSP430 versions of IAR.

1. Install the IAR version for Texas Instruments ARM or MSP430 devices.
2. Download the Windows installer for TI-RTOS. For example, tirtos_setupwin32_1_##_##_##.exe.
3. Run the downloaded file to install the full TI-RTOS product. You can install TI-RTOS in a standalone directory or the directory where CCS 5.5 is installed (if you have both CCS and IAR). Installing in a directory path that contains spaces (such as `C:\Program Files (x86)` is not recommended.)

   Note: TI-RTOS also installs the XDCtools component if you have not already installed the version needed by TI-RTOS as part of a CCS or SYS/BIOS installation. TI-RTOS places XDCtools in a separate directory at the same level where you install TI-RTOS. For example, if the TI-RTOS installation directory is `C:\mysoftware\tirtos_1_21_##_##`, the XDCtools directory will be `C:\mysoftware\xdctools_3_25_##_##`.

Follow the instructions in Section 3.2 and Section 3.3 to complete the installation of the TI-RTOS examples, build the examples with the IAR compiler, and load and run the examples with IAR Embedded Workbench.

Chapter 3

SPRUHD3G—January 2014

Examples for TI-RTOS

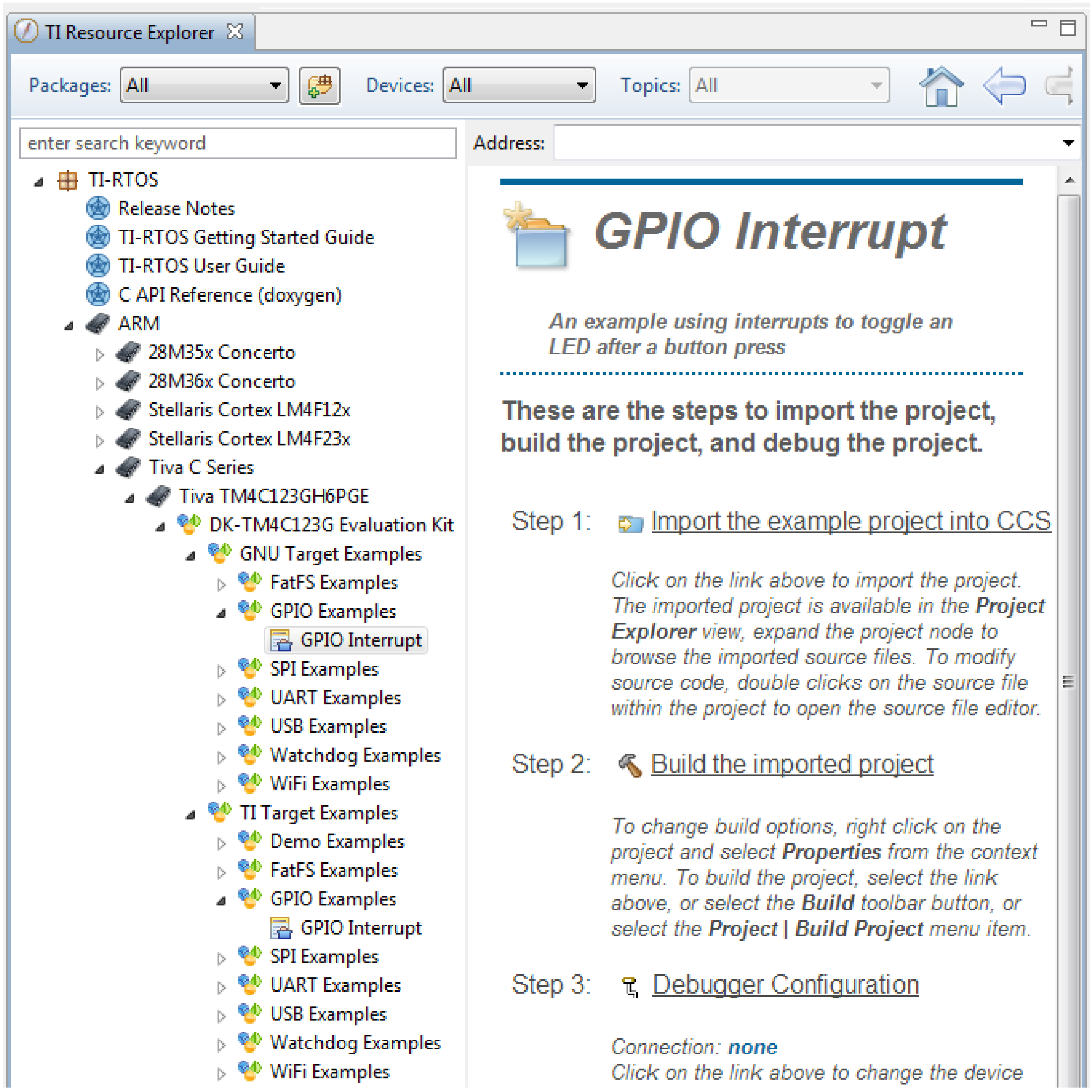
TI-RTOS comes with a number of examples that illustrate on how to use the individual components. This chapter explains how to create and use these examples.

Topic **Page**

3.1 Creating Example Projects Using the TI Resource Explorer

TI-RTOS uses TI Resource Explorer within CCS to let you quickly create example projects and open documentation. TI Resource Explorer provides TI-RTOS examples for both the TI and GNU tool chains for some boards (ARM M4F devices). Follow these steps to use TI Resource Explorer to create and use TI-RTOS examples:

1. If the TI Resource Explorer tab in CCS is not shown, choose **View > TI Resource Explorer** to open it.



An example using interrupts to toggle an LED after a button press

These are the steps to import the project, build the project, and debug the project.

Step 1: Import the example project into CCS

*Click on the link above to import the project. The imported project is available in the **Project Explorer** view, expand the project node to browse the imported source files. To modify source code, double clicks on the source file within the project to open the source file editor.*

Step 2: Build the imported project

*To change build options, right click on the project and select **Properties** from the context menu. To build the project, select the link above, or select the **Build** toolbar button, or select the **Project | Build Project** menu item.*

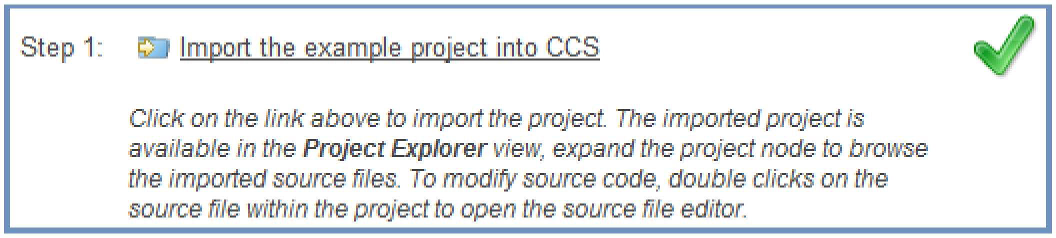
Step 3: Debugger Configuration

*Connection: **none***
Click on the link above to change the device

2. Expand the TI-RTOS item in the tree to show the projects available for your board. The examples for a particular board are grouped based on the tool chain (TI or GNU) used for compilation and then based on the peripheral used by the example, such as UART or SPI.

3. Select an example to create. See Section 3.4, *Summary of Example Peripheral Use and Target Support* and the readme files provided in the example projects for information about each example.

4. Click the **Step 1** link in the right pane of the TI Resource Explorer to **Import the example project into CCS**. This adds a new project to your Project Explorer view. A green checkmark is placed next to each step you have completed for the selected example.



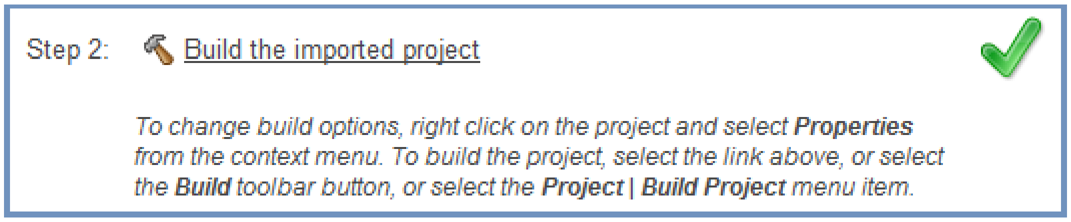
Step 1: Import the example project into CCS

*Click on the link above to import the project. The imported project is available in the **Project Explorer** view, expand the project node to browse the imported source files. To modify source code, double clicks on the source file within the project to open the source file editor.*

The project created will have a name with the format <*example*>_<*device*>.You can expand the project to view or change the source code and configuration file.

Note: CCS will not allow two projects with the same name in a workspace. If you want to import both TI and GNU projects for the same board and example in the same workspace, you would need to rename the first project you import before importing the second version.

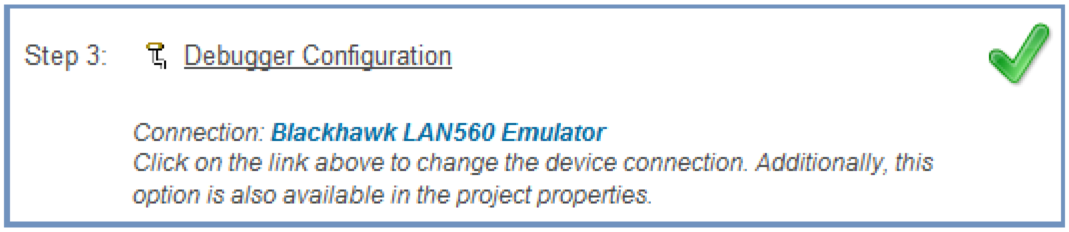
5. Click the **Step 2** link when you are ready to build the project. If you want to change any build options, right click on the project and select **Properties** from the context menu. For example, you can change compiler, linker, and RTSC (XDCtools) options.



Step 2: Build the imported project

*To change build options, right click on the project and select **Properties** from the context menu. To build the project, select the link above, or select the **Build** toolbar button, or select the **Project | Build Project** menu item.*

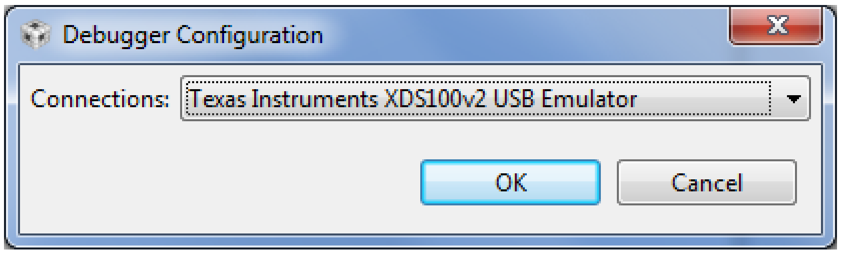
6. When you are ready to debug the example, click the **Step 3** link to create a target configuration to connect with the board.



Step 3: Debugger Configuration

*Connection: **Blackhawk LAN560 Emulator***
Click on the link above to change the device connection. Additionally, this option is also available in the project properties.

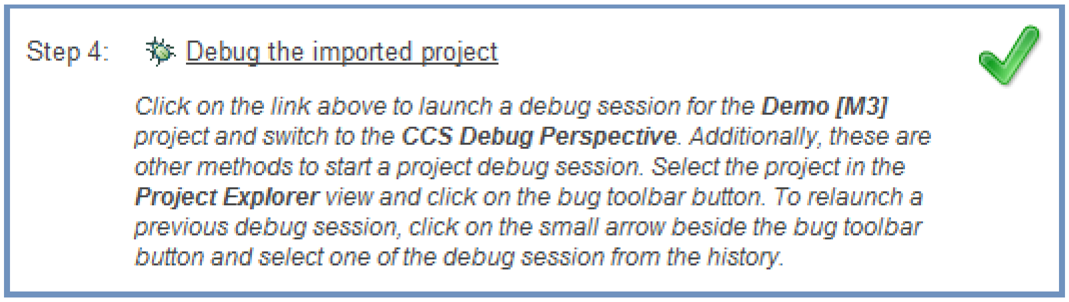
7. You will see the Debugger Configuration dialog. Choose an emulator from the list. For the F28M3x devices, choose the **Texas Instruments XDS 100v2 USB Emulator**. For Tiva devices, choose the **Stellaris In-Circuit Debug Interface**. For MSP430 devices, choose the **TI MSP430 USB1**.



Debugger Configuration

Connections: Texas Instruments XDS100v2 USB Emulator

8. Click the **Step 4** link to launch a debug session for the project and switch to the CCS Debug Perspective.



Step 4: Debug the imported project

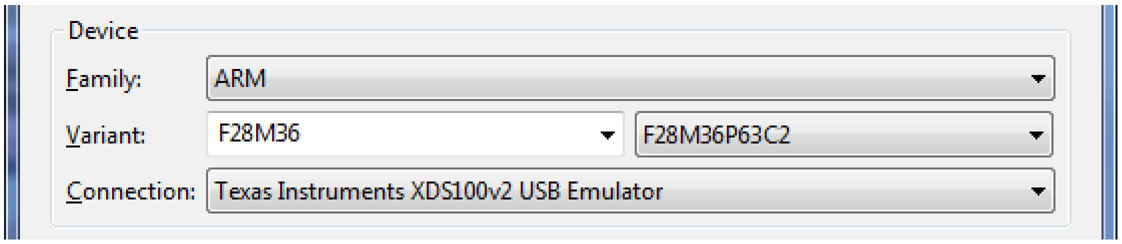
*Click on the link above to launch a debug session for the **Demo [M3]** project and switch to the **CCS Debug Perspective**. Additionally, these are other methods to start a project debug session. Select the project in the **Project Explorer** view and click on the bug toolbar button. To relaunch a previous debug session, click on the small arrow beside the bug toolbar button and select one of the debug session from the history.*

3.1.1 Creating an Empty TI-RTOS Project

These examples provide blank projects you can use as a starting point for creating a project that utilizes TI-RTOS. Both "Empty" and "Empty (Minimal)" versions are provided. The "Empty" version enables more kernel features and debug capabilities at the cost of large footprint. The "Empty (Minimal)" version disables various kernel features and debug capabilities to minimize the footprint. See the "Memory Usage with TI-RTOS" chapter in the TI-RTOS User's Guide (SPRUHD4) for details about techniques used to minimize the footprint.

Empty projects are not created with the TI Resource Explorer. Instead, create empty projects as follows:

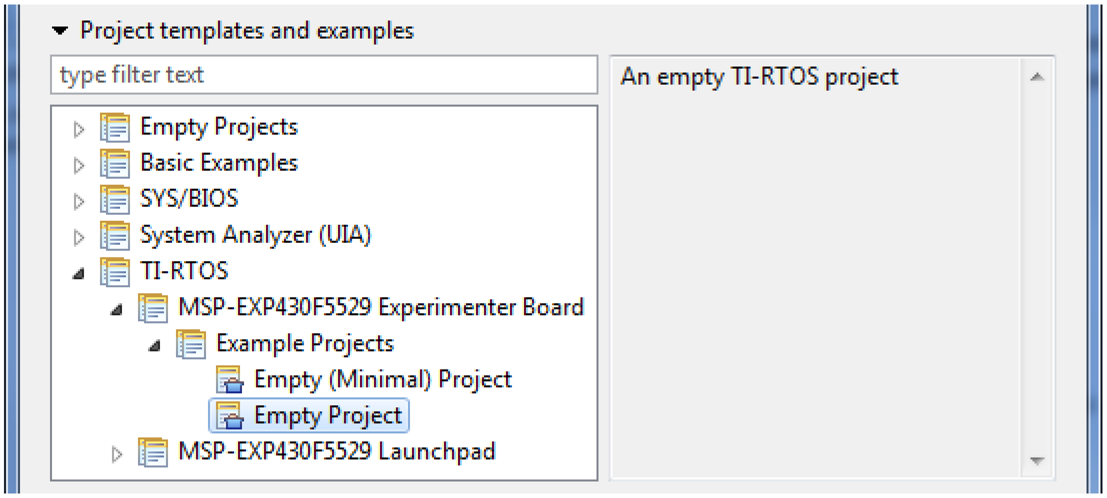
1. Choose the **Project > New CCS Project** menu command. (This has the same effect as using the **File > New > CCS Project** menu command.)
2. Name the project.
3. Select a device for which TI-RTOS provides examples.



Device
Family: ARM
Variant: F28M36 F28M36P63C2
Connection: Texas Instruments XDS100v2 USB Emulator

4. In the **Connection** field, choose the **Texas Instruments XDS 100v2 USB Emulator** for F28M3x devices. For Tiva devices, choose the **Stellaris In-Circuit Debug Interface**. For MSP430 devices, choose the **TI MSP430 USB1**.

5. In the **Project templates and examples** list, expand the TI-RTOS category and select the **Empty Project** item. If TI-RTOS is not in the list, no examples are provided for the device you selected.



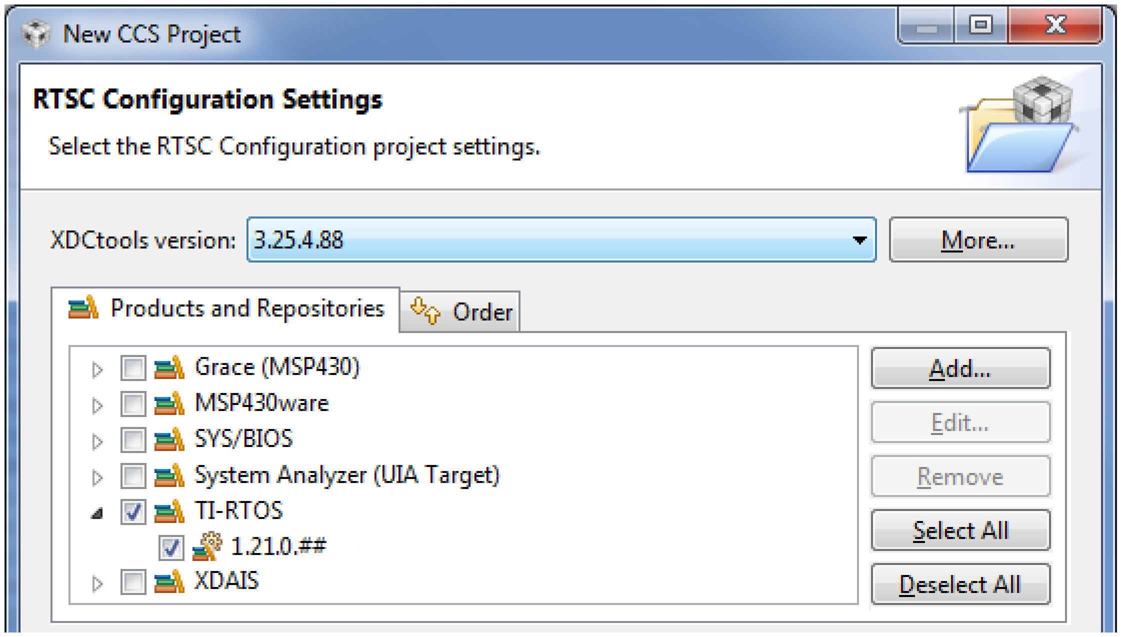
Project templates and examples

type filter text

- Empty Projects
- Basic Examples
- SYS/BIOS
- System Analyzer (UIA)
- TI-RTOS
  - MSP-EXP430F5529 Experimenter Board
    - Example Projects
      - Empty (Minimal) Project
      - Empty Project
  - MSP-EXP430F5529 Launchpad

An empty TI-RTOS project

6. Click **Next**. The next page of the new project wizard shows only the TI-RTOS product selected. The SYS/BIOS, IPC, and UIA components do not need to be selected, because they are components that are included with TI-RTOS.



New CCS Project

RTSC Configuration Settings

Select the RTSC Configuration project settings.

XDCtools version: 3.25.4.88

- [ ] Grace (MSP430)
- [ ] MSP430ware
- [ ] SYS/BIOS
- [ ] System Analyzer (UIA Target)
- [x] TI-RTOS
  - [x] 1.21.0.##
- [ ] XDAIS

7. Click **Finish**. The files in the empty project example include:

   — Key C files: empty.c, <*board*>.c/.h
   — Key configuration files: empty.cfg
   — Linker command file: <*board*>.cmd

8. Add to the example as needed to implement your application.

Note: Additional configuration might be needed as you add to the example. For example, if you add networking, you will likely need to increase the heap sizes.

3.2 Creating Example Projects to Build via a Command Line

TI-RTOS has a command-line utility called examplesgen that generates example projects along with the makefiles needed to build the examples for all the supported tool chains (TI, IAR, and GNU). The files are created in a location you specify on the command line. The `tirtos.mak` file in the top level directory of your TI-RTOS installation can be used to run examplesgen.

You only need to perform these steps once:

1. If you installed TI-RTOS in a location other than the default location of C:\ti, edit the `tirtos.mak` file in the TI-RTOS installation directory. Modify the following variables as needed to make them point to the correct locations, and save your changes.
   - TIRTOS_INSTALLATION_DIR: Full path to the TI-RTOS installation
   - XDCTOOLS_INSTALLATION_DIR: Full path to the XDCtools installation
   - CODEGEN_INSTALLATION_DIR: Full path to the TI code generation tools installation
   - IAR_COMPILER_INSTALLATION_DIR: Full path to the IAR code generation tools installation
   - GCC_INSTALLATION_DIR: Full path to the GCC code generation tools installation
2. Open a command line window, and use the following commands to run the examplesgen utility. (If you installed TI-RTOS in a protected directory, you should run the command window as the administrator.)

```
> cd <tirtos_install>
> ..\xdctools_3_25_04_88\gmake -f tirtos.mak examplesgen DEST="YOURPATH"
```

   For the destination path, use a UNIX-style path. That is, use forward slashes (/) instead of backslashes (\). For example, `DEST="C:/myfiles"`.

   The output from this command is a tirtos_1_21_##_##_examples directory tree containing folders for the supported boards. Each board folder contains folders for all the examples available for that board.

Examples for TI, IAR and GNU are generated for boards supported by TI-RTOS. Each board directory contains a `makedefs` file that can be modified to specify other installation paths or compiler/linker options and a `makefile` that can be used to build all the examples for that board. Each example directory has its own `makefile` that can be used to build that example specifically.

3.3 Creating Examples with IAR Embedded Workbench

After you have installed IAR Embedded Workbench and TI-RTOS as described in Section 2.3.4, you can follow the steps in this section to use the examples that come with TI-RTOS.

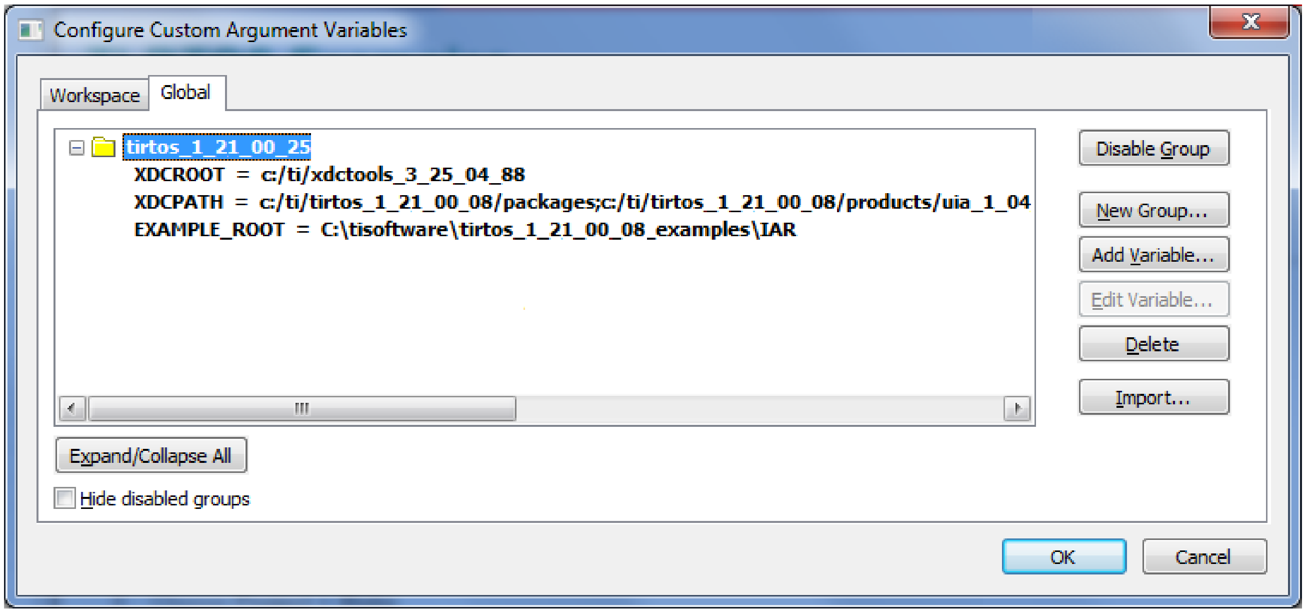
3.3.1 Preparing the TI-RTOS Examples for Use in IAR Embedded Workbench

Before using any of the TI-RTOS examples, you must generate the necessary example files and project files needed to build the examples with the IAR Embedded Workbench IDE. After you perform the TI-RTOS installation, follow the steps in Section 3.2, *Creating Example Projects to Build via a Command Line*. You only need to perform those setup steps once.

3.3.2 Integrating TI-RTOS Examples with IAR IDE

In order for IAR to discover your generated TI-RTOS examples, you must do the following once before you can create, build, and run TI-RTOS examples within IAR:

1. Open IAR Embedded Workbench.
2. Choose **Tools > Options** to open the IDE Options dialog. Select the **Project** category and check the box to **Enable project connections**. Click **OK**.
3. Choose **Tools > Configure Custom Argument Variables**. This opens a dialog that allows you to define paths to integrate TI-RTOS with IAR Embedded Workbench. The IAR folder in the examples you generated with the examplesgen utility, there is a `tirtos_#_##_##_##.custom_argvars` file that sets variables for these paths.
4. Select the **Global** tab, and click **Import**.
5. Browse to the IAR subdirectory of the examples tree you generated with the examplesgen utility. For example, if you specified `C:\myfiles` as the destination, browse to the `C:\myfiles\tirtos_#_##_##_##_examples/IAR` directory.
6. Select the `tirtos_#_##_##_##.custom_argvars` file, and click **Open**.
7. The result should look similar to this. Click **OK** to finish.

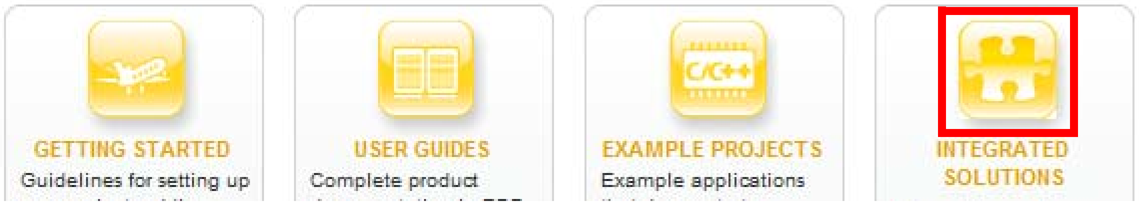


tirtos_1_21_00_25
XDCROOT = c:/ti/xdctools_3_25_04_88
XDCPATH = c:/ti/tirtos_1_21_00_08/packages;c:/ti/tirtos_1_21_00_08/products/uia_1_04
EXAMPLE_ROOT = C:\tisoftware\tirtos_1_21_00_08_examples\IAR

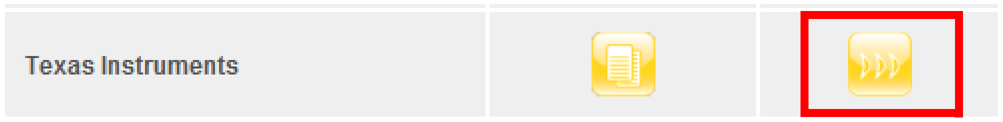
3.3.3 Creating TI-RTOS Examples with IAR Embedded Workbench

Follow these steps to import a TI-RTOS example into an IAR project:

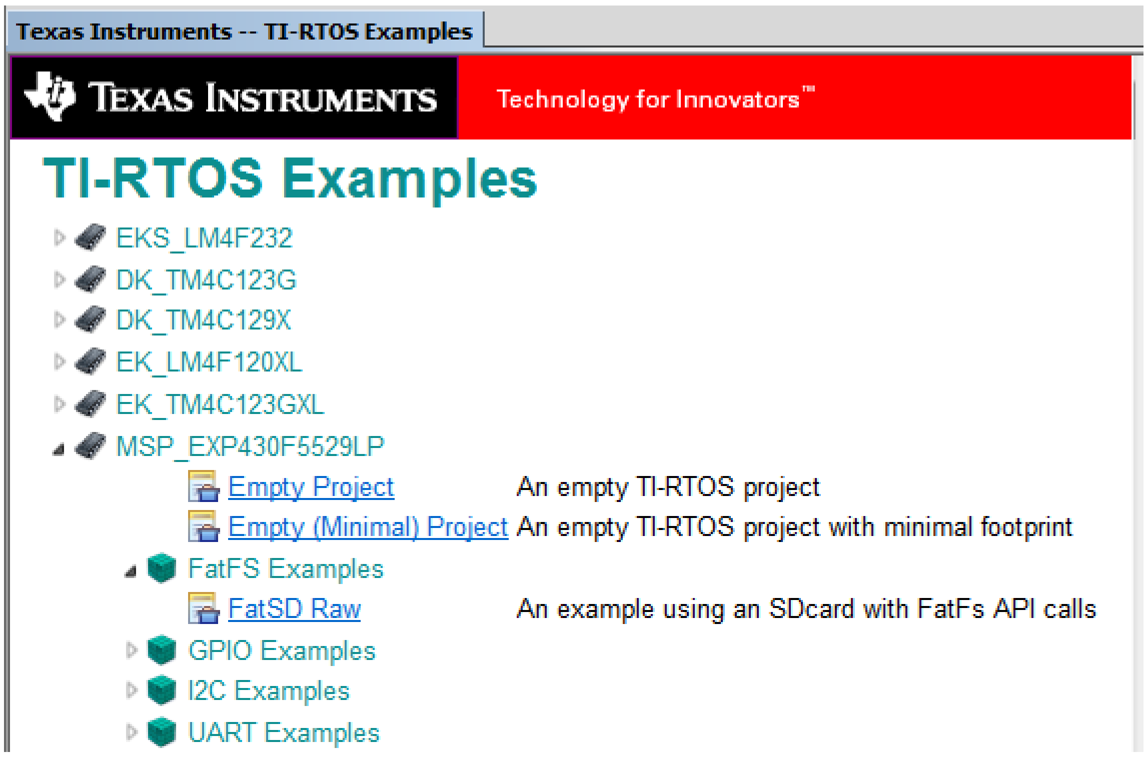
1. First, choose **Project > Create New Project** from the IAR menus.
2. Select the correct **Tool chain**, and choose the **Empty project** template. Click **OK**.
3. Browse to the location where you want to save this project, and type a name for the project file (*.ewp). It is best to use a directory that does not contain other projects or workspaces. Click **Save**.
4. If the IAR Information Center is not open, choose **Help > Information Center** from the IAR menus.
5. Click the **Integrated Solutions** link.



6. In the partners table, click the **Example Projects** button in the Texas Instruments row.



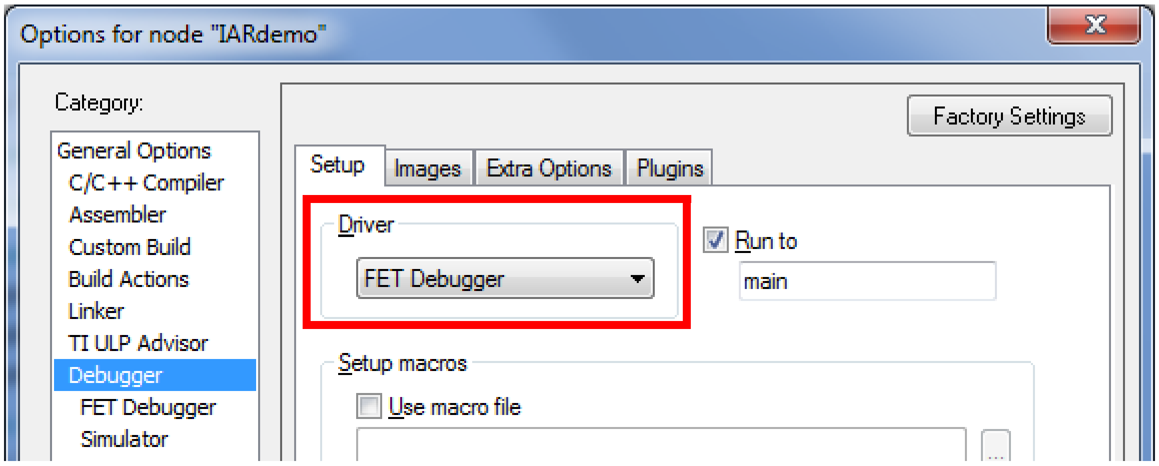
7. Click the "example applications" link to open the **Texas Instruments -- TI-RTOS Examples** page.
8. Click the link to browse example applications.
9. Expand the list of examples. Click on an example to add files for that example to your current project.


Texas Instruments -- TI-RTOS Examples
TEXAS INSTRUMENTS
Technology for Innovators™

TI-RTOS Examples

- EKS_LM4F232
- DK_TM4C123G
- DK_TM4C129X
- EK_LM4F120XL
- EK_TM4C123GXL
- MSP_EXP430F5529LP
  - Empty Project An empty TI-RTOS project
  - Empty (Minimal) Project An empty TI-RTOS project with minimal footprint
  - FatFS Examples
    - FatSD Raw An example using an SDcard with FatFs API calls
  - GPIO Examples
  - I2C Examples
  - UART Examples

10. Right-click on the project file, and choose **Options**. Select the **Debugger** category, and change the Driver to **FET Debugger** in the **Setup** tab.



11. Move to the **Plugins** tab and put a checkmark in the box next to **TI-RTOS**. This enables the RTOS Object View (ROV) plugin tools. Click **OK**. Note that you will need to change these Debugger options separately for each TI-RTOS project you create in IAR.
12. If you are using an ARM target, move to the **Download** tab and put a checkmark in the **Use flash loader(s)** box.
13. Choose **File > Save Workspace** from the IAR menus.
14. Browse to the location where you want to save this workspace, and type a name for the workspace file (*.eww). It is common to save the project and workspace in the same directory. Click **Save**.

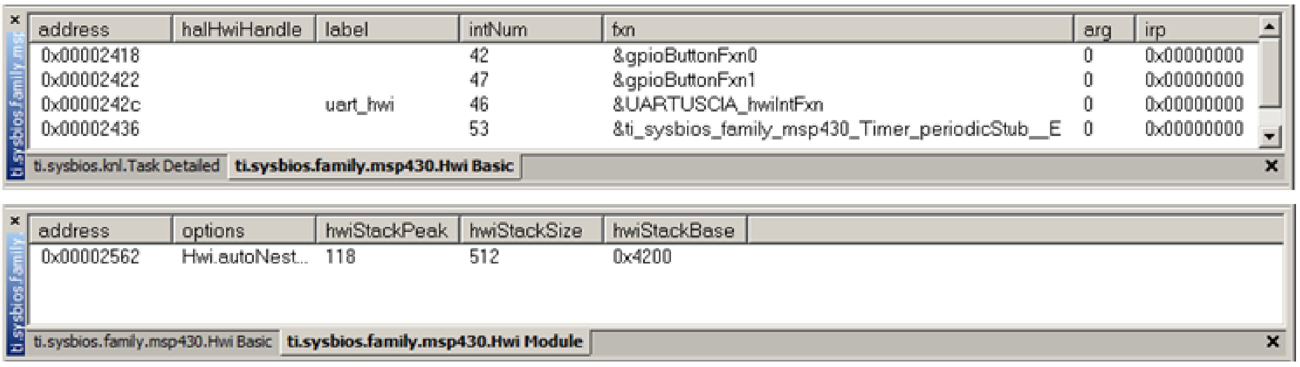
3.3.4 Using TI-RTOS Examples for IAR Embedded Workbench

You can use the text editor in IAR to modify *.c and *.cfg files in the example. The graphical configuration tool that is available for editing *.cfg files in CCS is not available within IAR.

To build examples, choose **Project > Make** or press F7. You may be asked to save the workspace.

To run examples, choose **Project > Download and Debug** or press Ctrl+D.

You can halt the target and choose any of the tools in the **TI-RTOS** menu to run the RTOS Object View (ROV) plugins. These plugins let you see information such as the peak stack size and addresses. The plugins work similarly to the ROV tool in CCS. However, you open a separate pane for each module that you want to view. See the http://rtsc.eclipse.org/docs-tip/RTSC_Object_Viewer web page for more about using the ROV tool.


ti.sysbios.family.msp
ti.sysbios.knl.Task Detailed
ti.sysbios.family.msp430.Hwi Basic
ti.sysbios.family
ti.sysbios.family.msp430.Hwi Basic
ti.sysbios.family.msp430.Hwi Module

address	halHwiHandle	label	intNum	fxn	arg	irp
0x00002418			42	&gpioButtonFxn0	0	0x00000000
0x00002422			47	&gpioButtonFxn1	0	0x00000000
0x0000242c		uart_hwi	46	&UARTUSCIA_hwiIntFxn	0	0x00000000
0x00002436			53	&ti_sysbios_family_msp430_Timer_periodicStub__E	0	0x00000000

address	options	hwiStackPeak	hwiStackSize	hwiStackBase
0x00002562	Hwi.autoNest...	118	512	0x4200

3.3.5 Using Command Line Builds with TI-RTOS Examples for IAR Embedded Workbench

You can import, build, and run TI-RTOS examples from within IAR Workbench as described in the previous section.

Alternately, if you want to build TI-RTOS examples from the command line, you can use the provided makefiles outside the IAR Embedded Workbench IDE. The executable files you build will have a *.out file extension. To build an example project from the command line, follow these steps:

1. Move to the board-level directory of the destination you specified for the `examplesgen` command. For example, if you built the examples tree in C:\myfiles and you are using the MSP-EXP430F5529 board, move to the `C:\myfiles\tirtos_1_21_##_##_examples\IAR\MSP_EXP430F5529` directory.
2. Use a text editor to edit the `makedefs` file in that directory.
3. Modify the CODEGEN_INSTALLATION_DIR variable in the `makedefs` file as needed to make it point to the location of the IAR compiler. For example:

```
CODEGEN_INSTALLATION_DIR = C:/Program Files (x86)/IAR Systems/Embedded Workbench 6.5/430
            or
CODEGEN_INSTALLATION_DIR = C:/Program Files (x86)/IAR Systems/Embedded Workbench 6.5/arm
```

4. To build all the examples for a particular board, navigate to that board's directory and run the following command, where <xdctools_dir> is the location of the XDCtools component that was installed with TI-RTOS. For example, `C:\myfiles\xdctools_3_25_##_##`.

```
> <xdctools_dir>\gmake all
```

5. To build only one example, go to that example's directory and run the same `gmake all` command.

After you build a TI-RTOS example from the command line, follow these steps to use IAR to load the externally-built executable onto the target and debug the application:

1. Choose **Project > Create New Project** from the IAR menus.
2. Select the **Tool chain**, and choose the **Externally built executable** template. Click **OK**.
3. Browse to the location where you want to save this project, and type a name for the project file (*.ewp). Click **Save**.
4. Delete any readme.txt file that is added to the project.
5. Choose **File > Save Workspace** from the IAR menus. Browse to the location where you want to save this workspace, and type a file name for the workspace file (*.eww). Click **Save**.
6. Choose **Project > Options**.
7. In the **General Options** category, go to the **Target** tab and select your specific **Device**.
8. In the **Debugger** category of the Options dialog, replace the default **Simulator** driver with the appropriate TI debugger. Choose the FET Debugger for MSP430 and the TI Stellaris debugger for a Tiva or Stellaris board. Click **OK**.
9. Choose **Project > Add Files**. Change the file types filter to show **All Files (*.*)**.
10. Browse to the location of the executable file you built. Select the <*examplename*>.out executable file and click **Open**.
11. Choose **Project > Download and Debug** to load the executable onto the target hardware.
12. Use the debugging features in IAR to debug the application.

3.3.6 Managing Program Stack Sizes for TI-RTOS Examples for IAR

For IAR, the program stack size for each device is set in its linker command file. The Program.stack line in an example's configuration file is not recognized by the IAR environment, so the value you specify is not the actual stack size for your example.

For the MSP430F5529 board, the following lines in the linker command file (the *.xcl file in the generated examples folder) set the stack size for your example:

```
//Uncomment for command line use (Modified for TI-RTOS examples)
-D_STACK_SIZE=768
-D_DATA16_HEAP_SIZE=0
-D_DATA20_HEAP_SIZE=0
```

For ARM boards, the *.icf linker command file has lines like the following to set the program stack size:

```
/*-Sizes-*/
define symbol __ICFEDIT_size_cstack__  =  0x1000;
define symbol __ICFEDIT_size_heap__      = 0x2000;
```

Note: The default stack sizes provided are sufficient for the TI-RTOS examples.

SYS/BIOS manages its own heap via the BIOS.heapSize configuration parameter. Therefore, the IAR heap can be set to 0.

3.4 Summary of Example Peripheral Use and Target Support

The components and hardware used by the TI-RTOS examples are shown in the following table. Details about these examples are provided in the readme files in the example projects.

There are three example categories. The Empty and Empty (Minimal) projects are configured to make TI-RTOS available but do not contain specific code that uses TI-RTOS. The Demo examples use several peripherals working together. The remaining examples show how to use a specific peripheral.

Table 3-1. Components Used by TI-RTOS Examples

Example	SYS/BIOS	SD Card / SDSPI	USB MSC Host	USB HID	USB CDC	I^{2}C	GPIO	SPI	Watchdog Timer	WiFi	UART	NDK / EMAC	UIA	IPC
Empty and Empty (Minimal) Projects	X						X						X	
Demo [M3] / Demo [C28]	X	X			X	X	X				X	X	X	X
IPC SPI Master / Slave Demo	X						X	X					X	X
Graphic Library Demo	X	X			X		X						X	
TCP Echo (IPv4 and IPv6)	X						X					X		
TCP Echo for CC3000	X						X	X		X				
CC3000 Patcher	X						X	X		X				
UDP Echo (IPv4 and IPv6)	X						X					X		
UDP Echo for CC3000	X						X	X		X				
SPI Loopback	X						X	X						
FatSD: File Copy	X	X					X							
FatSD Raw: File Copy & FatFs APIs	X	X					X							
FatSD USB Copy: SD Card & USB Drive	X	X	X				X							
GPIO Interrupt	X						X							
I^{2}C EEPROM	X					X	X							
I^{2}C RF430CL330 Load (with BoosterPack)	X					X	X							
I^{2}C TMP006(with BoosterPack)	X					X	X							
I^{2}C TPL0401EVM(with BoosterPack)	X					X	X							
UART Console	X				X		X				X		X	
UART Echo	X						X				X		X	
UART Logging	X						X				X		X	
USB Keyboard Device	X			X			X							
USB Keyboard Host	X			X			X							
USB Mouse Device	X			X			X							
USB Mouse Host	X			X			X							
USB Serial Device	X				X		X							
USB CDC Mouse Device	X			X	X		X							
Watchdog	X						X		X					

The boards for which TI-RTOS examples are provided are shown in the following table.

Table 3-2. Example Availability by Board

Example	TMDXDOCKH52C1	TMDXDOCK28M36	EK-TM4C123GXL / EK-LM4F120XL	DK-TM4C123G / EKS-LM4F232	DK-TM4C129X	MSP-EXP430F5529LP	MSP-EXP430F5529
Empty and Empty (Minimal) TI-RTOS Projects	X	X	X	X	X	X	X
Demo [M3] / Demo [C28]	X	X					
IPC SPI Master / Slave	X						
Graphic Library Demo				X			
TCP Echo	X	X			X		
TCP Echo for CC3000 [1]			X	X		X	X
CC3000 Patcher [1]			X	X		X	X
UDP Echo	X	X			X		
UDP Echo for CC3000 [1]			X	X		X	X
SPI Loopback	X	X	X	X	X		
FatSD: FatFs File Copy	X	X	X[2]	X	X	X[2]	X
FatSD Raw: FatFs File Copy using FatFs APIs	X	X	X[2]	X	X	X[2]	X
FatSD USB Copy: (SD Card and USB Drive)	X	X		X	X		
GPIO Interrupt	X	X	X	X	X	X	X
I^{2}C EEPROM	X						
I^{2}C RF430CL330 Load [3]			X		X	X	X
I^{2}C TMP006 [4]			X		X	X	
I^{2}C TPL0401EVM [5]			X		X	X	
UART Console	X	X	X	X	X	X	X
UART Echo	X	X	X	X	X	X	X
UART Logging	X	X	X	X	X	X	X
USB Keyboard Device	X	X	X	X	X	X	X
USB Keyboard Host	X	X		X	X		
USB Mouse Device	X	X	X	X	X	X	X
USB Mouse Host	X	X		X	X		
USB Serial Device	X	X	X	X	X	X	X
USB CDC Mouse Device	X	X	X	X	X	X	X
Watchdog	X	X	X	X	X	X	X

[1]This example requires either a CC3000 EM board or CC3000 BoosterPack. See the "Jumper Settings: section in Section 3.7 through Section 3.11 for details.

[2]This example requires an SD Card reader BoosterPack.

[3]This example requires a RF430CL330 NFC transponder module and the Wireless Connectivity Adaptor. See Section 3.12.4.

[4]This example requires a TMP006 BoosterPack. See Section 3.12.2.

[5]This example requires a TPL0401EVM BoosterPack. See Section 3.12.5.

There is a separate <*example_name*>_readme file for each of the examples. These files are added to your CCS project when you use the TI Resource Explorer to create a project.

The <*example_name*>_readme files contain the following types of information:

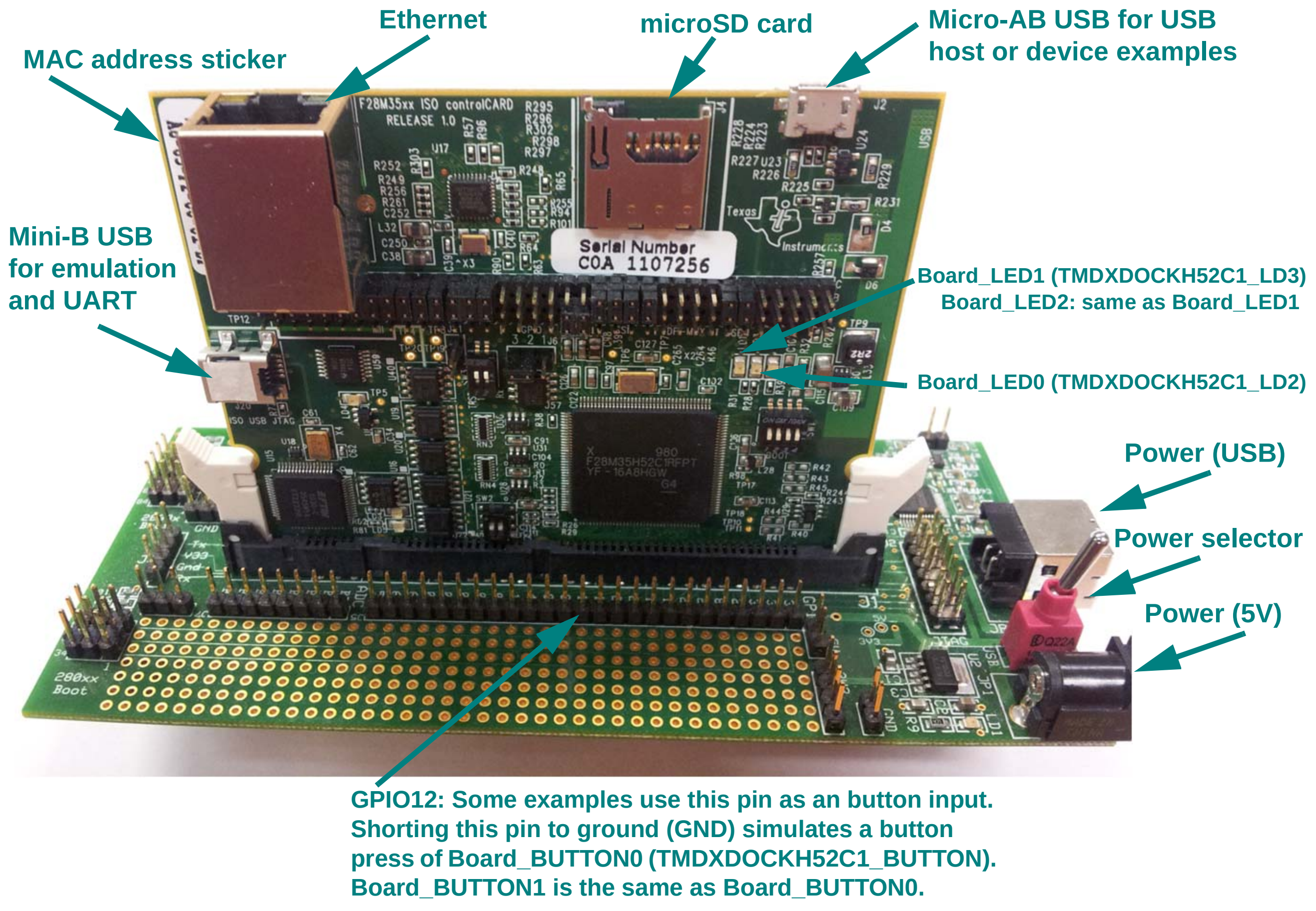
- Actions performed by functions in the example.
- Hardware-specific descriptions of buttons, LEDs, etc…
- Which external components are (or may be) needed to run with particular examples.

The sections that follow list settings required to run the TI-RTOS examples on the supported boards. They also list the hardware resources that TI-RTOS and its dependent components use by default. Some of these resources offer flexible options, whereas others are fixed in the current design or implementation.

3.5 Concerto TMDXDOCKH52C1 Settings and Resources

The TMDXDOCKH52C1 Experimenter Kit has a Concerto F28M35H52C1 device.

Concerto TMDXDOCKH52C1 connections:


Ethernet
microSD card
Micro-AB USB for USB host or device examples
MAC address sticker
Mini-B USB for emulation and UART
Board_LED1 (TMDXDOCKH52C1_LD3)
Board_LED2: same as Board_LED1
Board_LED0 (TMDXDOCKH52C1_LD2)
Power (USB)
Power selector
Power (5V)
GPIO12: Some examples use this pin as an button input. Shorting this pin to ground (GND) simulates a button press of Board_BUTTON0 (TMDXDOCKH52C1_BUTTON). Board_BUTTON1 is the same as Board_BUTTON0.

The USB connector for devices can accept a Micro-B cable for USB Device examples and a Micro-A dongle for USB Host examples.

Jumper settings:

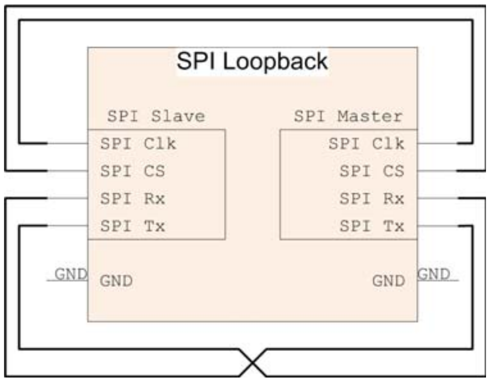
- J01-J15: B-C position (Ethernet)
- J20-J21: B-C position (I2C EEPROM)
  - J6: 2-3 position (I2C EEPROM Write protection off)
- J22-J25: B-C position (SPI/SSI SD Card slot)
- J30-J31: B-C position (USB Host and Device)
  - A board modification is required for the USB host examples (see Section 3.5.2)

Switch settings:

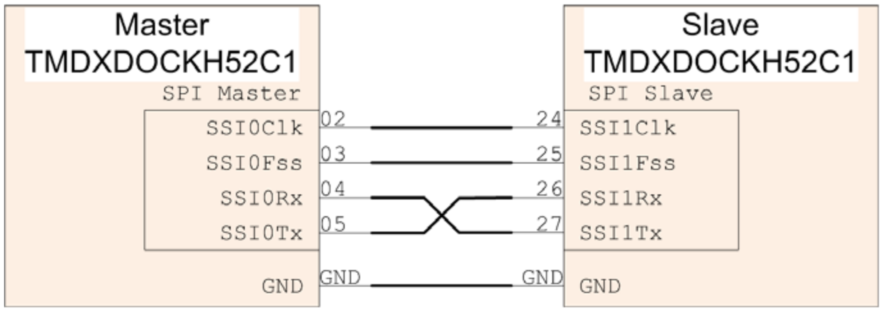
- SW1: Open all 4 switches by bringing them into the "down" position. This allows the M3 (master subsystem) to boot out of Flash memory.
- GPIO12: Some examples use this pin as an input. When shorting this pin to ground (GND), it will simulate a button press.

SPI Loopback example pin connections: When wiring for SPI loopback, pins on a single board are wired to other pins on the same board.

Master SPI Pin (Function)		Slave SPI Pin (Function)
02 (SPI0CLK)	→	24 (SPI1CLK)
03 (SPI0FSS)	→	25 (SPI1FSS)
04 (SPI0RX)	→	27 (SPI1TX)
05 (SPI0TX)	→	26 (SPI1RX)



SPI IPC Master/Slave example pin connections: When wiring for the IPC SPI Master/Slave example, two boards are used:



Resources Used:

The following list shows which TMDXDOCKH52C1 resources are used by TI-RTOS examples that make use of a particular component or peripheral.

- **SYS/BIOS.** Uses the first general-purpose timer available and that timer’s associated interrupts. Generally, this will be Timer 0. SYS/BIOS manages the Interrupt Vector Table.
- **IPC.** Uses the IPC registers including their associated interrupts.
- **TI-RTOS.**
  - **Ethernet.** Uses the EMAC driver and its associated interrupts with NDK to support networking.
  - **SD Card.** Uses FatFs and the SDSPI driver on SSI0 without interrupts to read and write to files on an SD Card.
  - **EEPROM.** Uses the I^{2}C driver on I2C0 with its associated interrupts to read and write to the onboard EEPROM.
  - **GPIOs.** The GPIO driver is used on 2 onboard LEDs: LD2 (PC6_GPIO70) and LD3 (PC7_GPIO71) as output pins and one pin as an input pin PB4_GPIO12.
  - **Serial.** The UART driver uses UART0 which is attached to the FTDI USB chip to facilitate serial communications.
  - **SPI.** The SPI driver uses SPI0 for Board SPI0 and SPI1 for Board SPI1.
  - **SPIMessageQTransport.** The master SSI peripheral uses SPI0, and the slave SSI peripheral uses SPI1.
  - **USB.** The USB reference examples use the USB library and the USB controller with its associated interrupts.
  - **Watchdog.** The Watchdog driver example uses Watchdog Timer 0 and its associated interrupt.

3.5.1 Setting the MAC Address

If you are using the NDK and the EMAC peripheral, you will need to edit the MAC address in the board-specific C file (for example, TMDXDOCKH52C1.c) in the examples. Modify the following definition in the file to match the MAC address printed on your board.

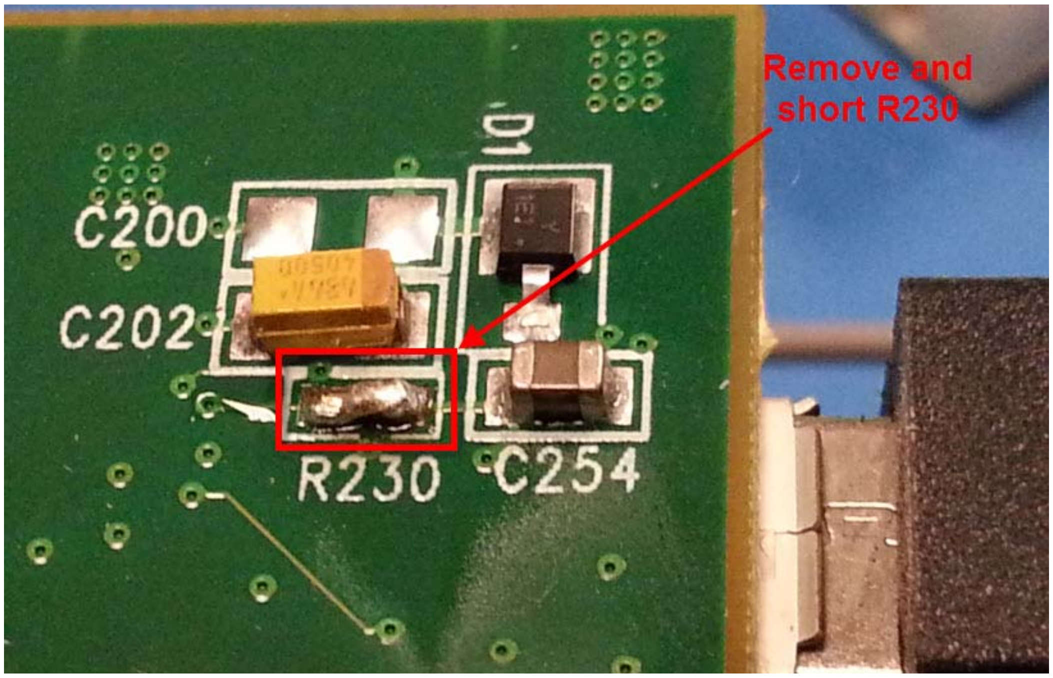
```
/*
 *  EMAC configuration structure
 *  Set user/company specific MAC octates. The following sets the address
 *  to ff-ff-ff-ff-ff-ff. Users need to change this to make the label on
 *  their boards.
 */
UInt8 macAddress[6] = {0xff, 0xff, 0xff, 0xff, 0xff, 0xff};
```

For example, the following would set the MAC address to A8-63-F2-00-05-1A:

```
UInt8 macAddress[6] = {0xA8, 0x63, 0xF2, 0x00, 0x05, 0x1A};
```

3.5.2 USB Host Mode Board Modification

Using the USB controller in host mode on the TMDXDOCKH52C1 requires a hardware modification to the control card. This modification is not required, but can be performed without causing problems, when using the USB controller in device mode.



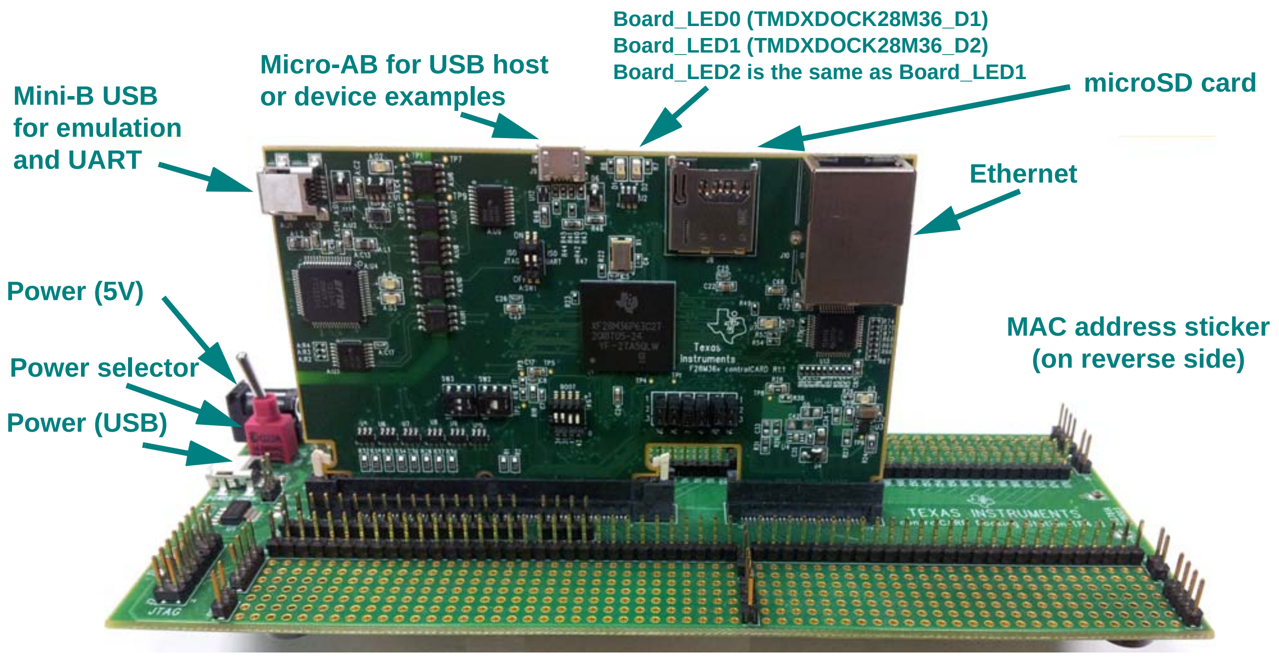
Remove and short resistor R230 on the control card.

This modification allows the USB_VBUS pin to correctly detect the VBUS voltage level; preventing a false VBUS_ERR from being generated by the USB controller.

3.6 Concerto TMDXDOCK28M36 Settings and Resources

The TMDXDOCK28M36 Experimenter Kit has a Concerto F28M36P63C2 device.

Concerto TMDXDOCK28M36 connections:


Board_LED0 (TMDXDOCK28M36_D1)
Board_LED1 (TMDXDOCK28M36_D2)
Board_LED2 is the same as Board_LED1
Micro-AB for USB host or device examples
Mini-B USB for emulation and UART
microSD card
Ethernet
Power (5V)
Power selector
Power (USB)
MAC address sticker (on reverse side)

The USB connector for devices can accept a Micro-B cable for USB Device examples and a Micro-A dongle for USB Host examples.

Jumper settings:

- J2-J7: 1-2 position (USB Host and Device)

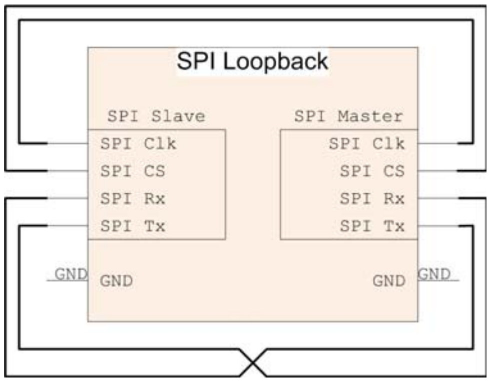
Switch settings:

- A:SWI: Both switches should be in the ON position
- SW1: Place all switches in the 1 (up) position to allow the M3 to boot out of Flash memory.
- GPIO 58: Some examples use pin 58 as it is labeled on the docking station as an input. Shorting this pin to ground (GND) will simulate a button press.

SPI Loopback example pin connections:

When wiring for SPI loopback, pins on a single board are wired to other pins on the same board.

Master SPI Pin (Function)		Slave SPI Pin (Function)
71 (SPI0CLK)	→	75 (SPI1CLK)
73 (SPI0FSS)	→	77 (SPI1FSS)
69 (SPI0RX)	→	81 (SPI1TX)
67 (SPI0TX)	→	79 (SPI1RX)



Resources Used:

The following list shows which TMDXDOCK28M36 resources are used by TI-RTOS examples that make use of a particular component or peripheral.

- **SYS/BIOS.** Uses the first general-purpose timer available and that timer's associated interrupts. Generally, this will be Timer 0. SYS/BIOS manages the Interrupt Vector Table.
- **IPC.** Uses the IPC registers including their associated interrupts.
- **TI-RTOS.**
  - **Ethernet.** Uses the EMAC driver and its associated interrupts with the NDK for networking.
  - **SD Card.** Uses FatFs and the SDSPI driver on SSI3 without interrupts to read and write to files.
  - **GPIOs.** The GPIO driver is used on 2 onboard LEDs: D1 (PE7_GPIO31) and D2 (PF2_GPIO34) as output pins and one pin as an input pin PB4_GPIO12, pin 58 on the docking station.
  - **Serial.** The UART driver uses UART0, which is attached to the FTDI USB chip.
  - **SPI.** The SPI driver uses SPI0 for Board SPI0 and SPI1 for Board SPI1.
  - **USB.** The USB reference examples use the USB library and the USB controller with its associated interrupts.
  - **Watchdog.** The Watchdog driver example uses Watchdog Timer 0 and its associated interrupt.

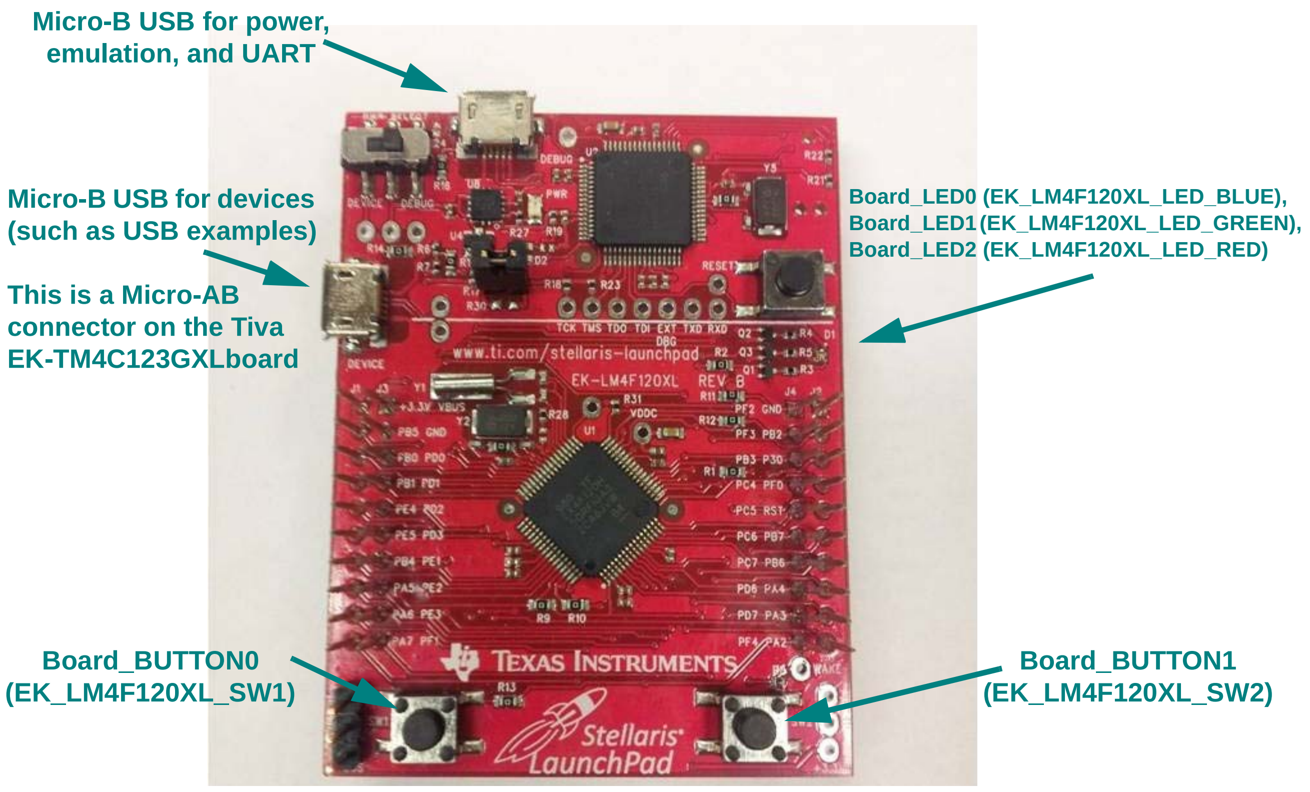
3.6.1 Setting the MAC Address

If you are using the NDK and the EMAC peripheral, you will need to edit the MAC address in the board-specific C file (for example, TMDXDOCK28M36.c) in the examples. See Section 3.5.1 for details.

3.7 Tiva EK-TM4C123GXL Settings and Resources

The EK-TM4C123GXL LaunchPad contains a Tiva TM4C123GH6PM device. This board is the newer version of the EK-LM4F120XL LaunchPad, which contains a Stellaris LM4F120H5QR device. TI-RTOS provides separate examples for both versions of the board for convenience.

Stellaris EK-LM4F120XL LaunchPad connections (EK-TM4C123GXL is similar):


Micro-B USB for power, emulation, and UART
Micro-B USB for devices (such as USB examples)
This is a Micro-AB connector on the Tiva EK-TM4C123GXLboard
Board_LED0 (EK_LM4F120XL_LED_BLUE),
Board_LED1 (EK_LM4F120XL_LED_GREEN),
Board_LED2 (EK_LM4F120XL_LED_RED)
Board_BUTTON0 (EK_LM4F120XL_SW1)
Board_BUTTON1 (EK_LM4F120XL_SW2)

The Micro-B connector on the Stellaris EK-LM4F120XL LaunchPad can be used for USB Device connections only. The Micro-AB connector on the Tiva EK-TM4C123GXL LaunchPad can be used for both USB Device and USB Host connections.

Jumper settings:

Examples that use the WiFi driver—for example, the TCP Echo and UDP Echo examples for CC3000—require a CC3000 BoosterPack. This BoosterPack fits over J1 and J2.

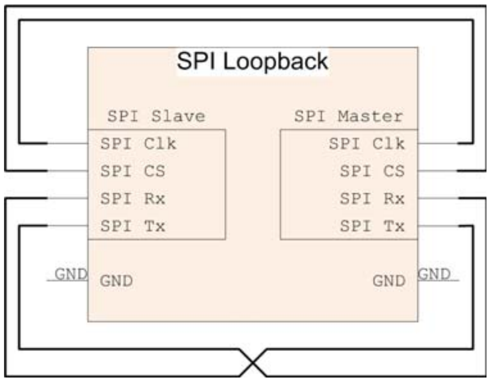
Switch settings:

- SW1: Some examples use PF4 as an input.
- SW2: Some examples use PF0 as an input

SPI Loopback example pin connections:

When wiring for SPI loopback, pins on a single board are wired to other pins on the same board.

Master SPI Pin (Function)		Slave SPI Pin (Function)
PA2 (SPI0CLK)	→	PD0 (SPI3CLK)
PA3 (SPI0FSS)	→	PD1 (SPI3FSS)
PA4 (SPI0RX)	→	PD3 (SPI3TX)
PA5 (SPI0TX)	→	PD2 (SPI3RX)



Resources Used:

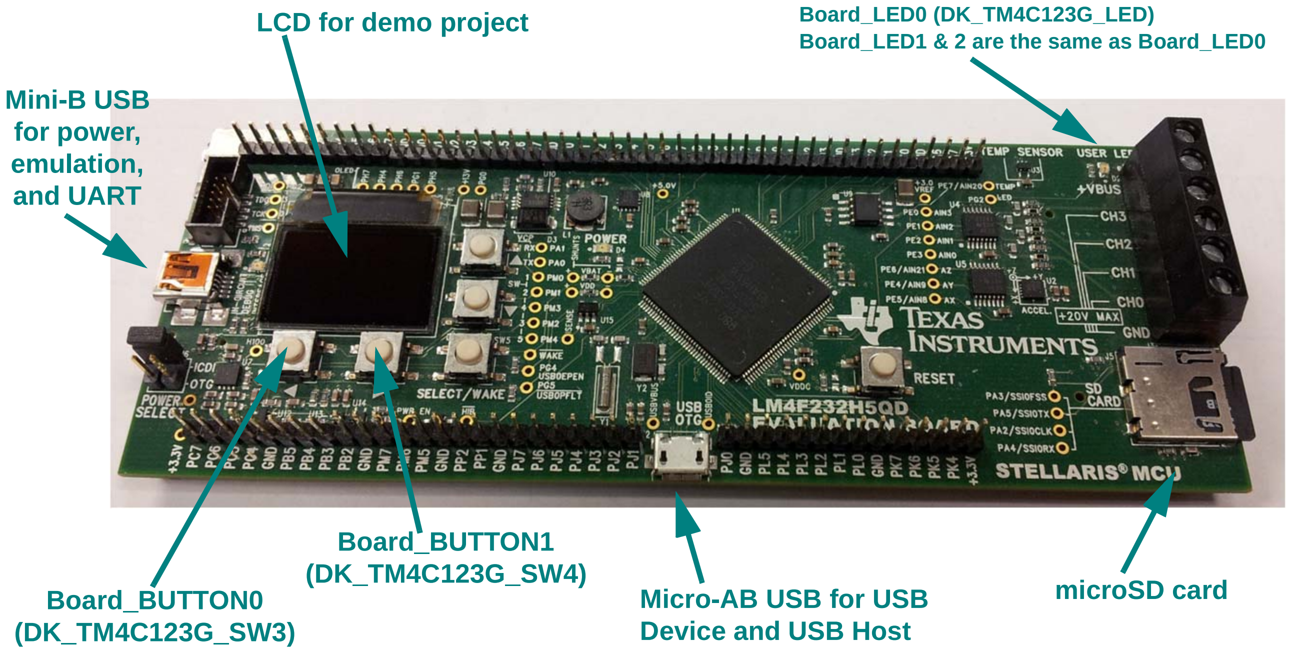
The following list shows which EK-TM4C123GXL resources are used by TI-RTOS examples that make use of a particular component or peripheral.

- **SYS/BIOS.** Uses the first general-purpose timer available and that timer's associated interrupts. Generally, this will be Timer 0. SYS/BIOS manages the Interrupt Vector Table.
- **TI-RTOS.**
  - **GPIOs.** The GPIO driver uses 3 output pins for the onboard RGB LED (R:PF1 G:FP3 B:PF2) and 2 input pins for switches SW1 (PF4) and SW2 (PF0).
  - **I2C.** The I^{2}C driver is configured on I2C3 to support with various BoosterPacks.
  - **Serial.** The UART driver uses UART0 which is attached to the FTDI USB chip to facilitate serial communications.
  - **SPI.** The SPI driver uses SPI0 for Board SPI0 and SPI3 for Board SPI1.
  - **USB.** The USB reference examples use the USB library and the USB controller with its associated interrupts. These boards only support USB in device mode. (The EK-TM4C123GXL LaunchPad can be made to support USB Host, but only with hardware modifications.) USB Host examples are not provided for either of these boards.
  - **Watchdog.** The Watchdog driver example uses Watchdog Timer 0 and its associated interrupt.
  - **WiFi.** The WiFi examples for use with the CC3000 use GPIO pins PB2, PB5, and PE0 as well as the GPIO interrupt for port B. They also use the SPI driver on SSI2 with interrupts and the associated uDMA channels.

3.8 Tiva DK-TM4C123G Settings and Resources

The DK-TM4C123G Evaluation Kit contains a Tiva TM4C123GH6PGE device. This board is the newer version of the EKS-LM4F232 Evaluation Kit, which contains a Stellaris LM4F232H5QD device. The TI-RTOS examples function equivalently on both versions of this board. Separate examples are provided for both versions of the board to prevent confusion for legacy users.

Stellaris EKS-LM4F232 Evaluation Kit connections (DK-TM4C123G is similar):


LCD for demo project
Board_LED0 (DK_TM4C123G_LED)
Board_LED1 & 2 are the same as Board_LED0
Mini-B USB for power, emulation, and UART
Board_BUTTON1 (DK_TM4C123G_SW4)
Board_BUTTON0 (DK_TM4C123G_SW3)
Micro-AB USB for USB Device and USB Host
microSD card

The USB connector for devices can accept a Micro-B cable for USB Device examples and a Micro-A dongle for USB Host examples.

Jumper settings: N/A

Examples that use the WiFi driver—for example, the TCP Echo and UDP Echo examples for CC3000 and the CC3000 Patcher example—require a CC3000 evaluation module with EM headers. These EM headers are typically installed as part of the DK-TM4C123G Evaluation Kit, but they are not included with the EKS-LM4F232 Evaluation Kit. The EM Headers can be ordered from various third-party suppliers, including Samtex (part TFM-110-02-S-D). EM headers for the EKS-LM4F232 go on the bottom of the board at J9 and J10.

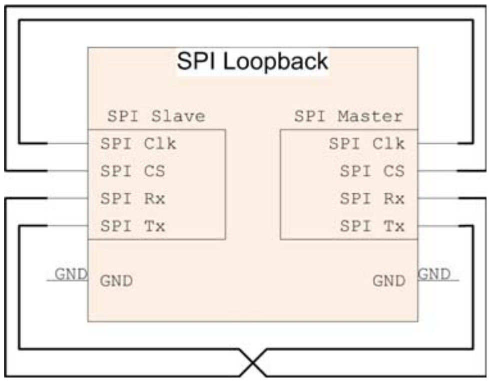
Switch settings:

- SW3: Some examples use PM2 as an input
- SW4: Some examples use PM3 as an input

SPI Loopback example pin connections:

When wiring for SPI loopback, pins on a single board are wired to other pins on the same board.

Master SPI Pin (Function)		Slave SPI Pin (Function)
PF2 (SPI1CLK)	→	PK0 (SPI3CLK)
PF3 (SPI1FSS)	→	PK1 (SPI3FSS)
PF0 (SPI1RX)	→	PK3 (SPI3TX)
PF1 (SPI1TX)	→	PK2 (SPI3RX)



Resources Used:

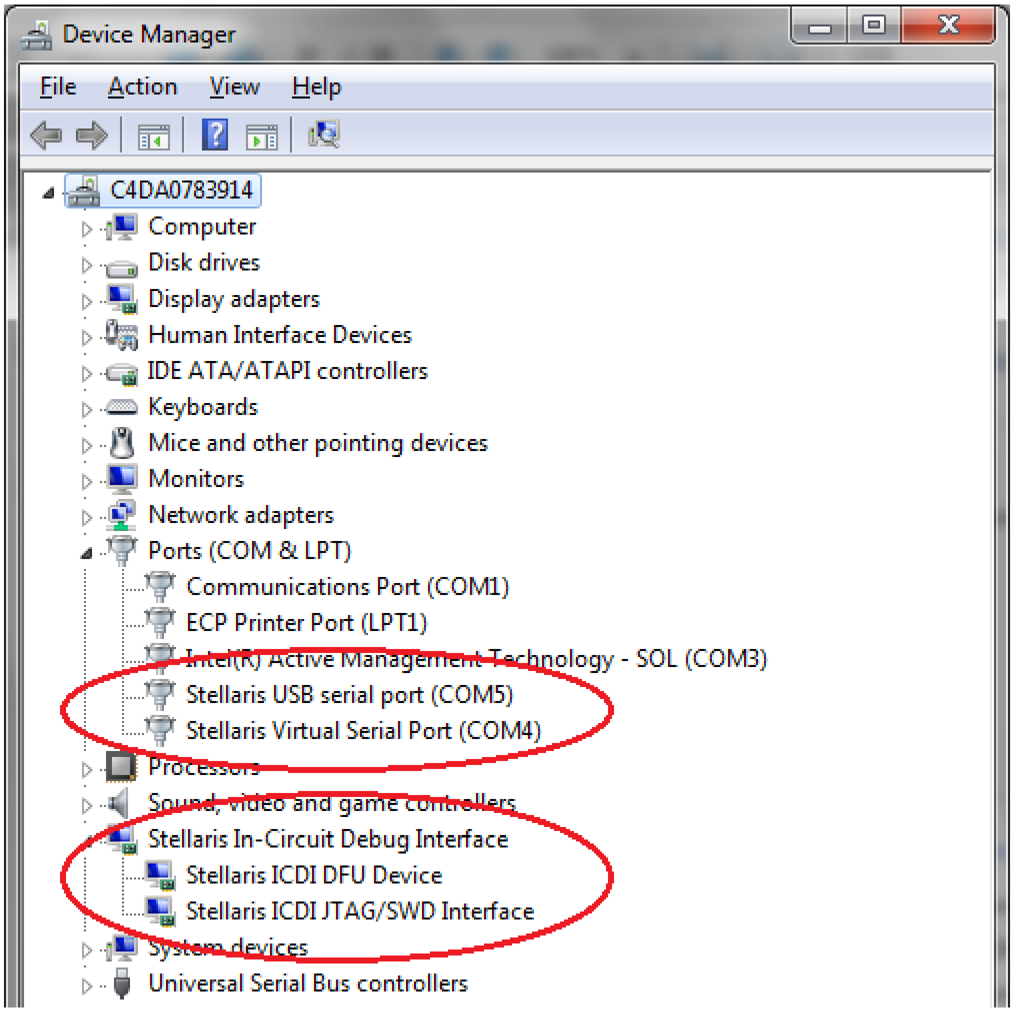
The following list shows which DK-TM4C123G resources are used by TI-RTOS examples that make use of a particular component or peripheral.

- **SYS/BIOS.** Uses the first general-purpose timer available and that timer's associated interrupts. Generally, this will be Timer 0. SYS/BIOS manages the Interrupt Vector Table.
- **TI-RTOS.**
  - **SD Card.** Uses FatFs and the SDSPI driver on SSI0 without interrupts to read and write to files.
  - **GPIOs.** The GPIO driver is used on the onboard User LED (PG2) as an output pin and on 5 input pins SW1-SW5 (PM0-4).
  - **Serial.** The UART driver uses UART0 which is attached to the FTDI USB chip to facilitate serial communications.
  - **SPI.** The SPI driver uses SPI1 for Board SPI0 and SPI3 for Board SPI1.
  - **USB.** The USB reference examples use the USB library and the USB controller with its associated interrupts.
  - **Watchdog.** The Watchdog driver example uses Watchdog Timer 0 and its associated interrupt.
  - **WiFi.** The WiFi examples for use with the CC3000 use GPIO pins PC7, PH1, and PC6 as well as the GPIO interrupt for port C. It also uses the SPI driver on SSI3 with interrupts and the associated uDMA channels.

USB connection and ports:

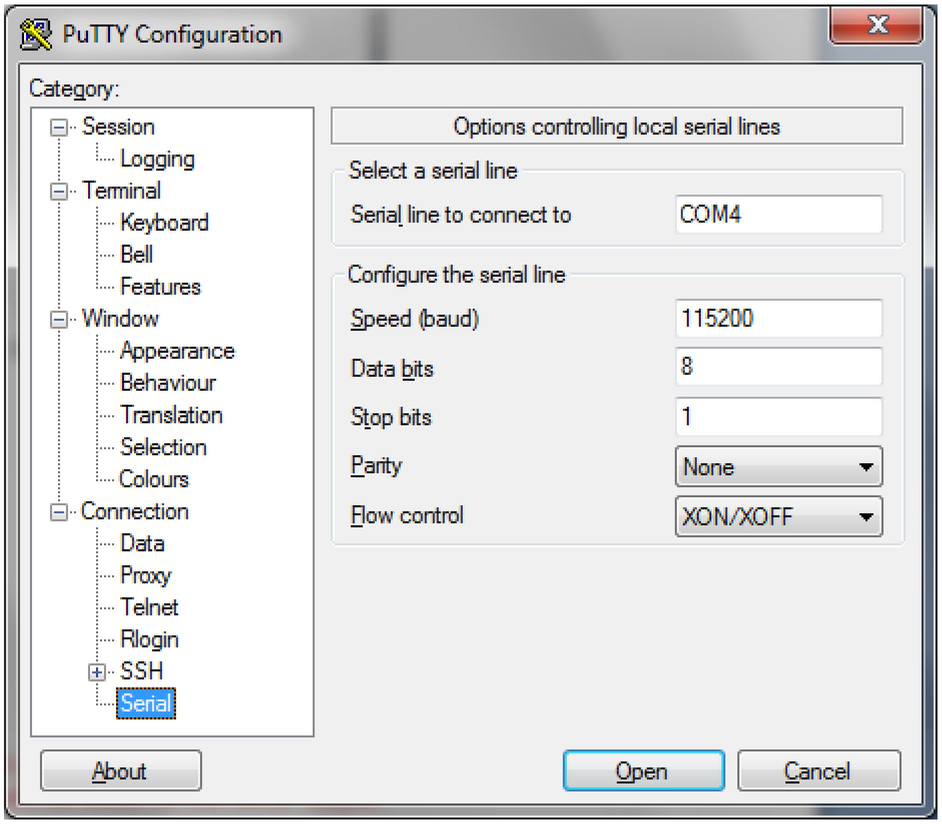
Connect the power/CDI USB connector on the board to a USB port on your host PC. This connection is used for both emulation and the serial output. If you would also like to see instrumentation (UIA), also connect the Host/Device/OTG USB connector on the board to your host PC. See the *Stellaris LM4F232 Evaluation Board User's Manual* (SPMU272) for details about USB connectors.

After you connect both USB ports, you can make sure the Stellaris drivers are installed correctly by opening the Windows Device Manager. To do this, right-click on **Computer** in Windows Explorer and select **Properties**. Choose the **Device Manager**. You should see a device list similar to the following:



```
C4DA0783914
  Computer
  Disk drives
  Display adapters
  Human Interface Devices
  IDE ATA/ATAPI controllers
  Keyboards
  Mice and other pointing devices
  Monitors
  Network adapters
  Ports (COM & LPT)
    Communications Port (COM1)
    ECP Printer Port (LPT1)
    Intel(R) Active Management Technology - SOL (COM3)
    Stellaris USB serial port (COM5)
    Stellaris Virtual Serial Port (COM4)
  Processors
  Sound, video and game controllers
  Stellaris In-Circuit Debug Interface
    Stellaris ICDI DFU Device
    Stellaris ICDI JTAG/SWD Interface
  System devices
  Universal Serial Bus controllers
```

The virtual serial port (in the previous figure, COM4) will be used for serial output. You will need to set up a terminal emulator to see the console output from the TI-RTOS UART Console example. Use a serial connection to the Virtual Serial Port of your device, with 115200 bps, 8 data bits, 1 stop bit, and no parity. For example, if you had the Virtual Serial Port, COM4, using the PuTTY terminal emulator, you would set it up as follows:



Options controlling local serial lines

Select a serial line

Serial line to connect to: COM4

Configure the serial line

Speed (baud): 115200

Data bits: 8

Stop bits: 1

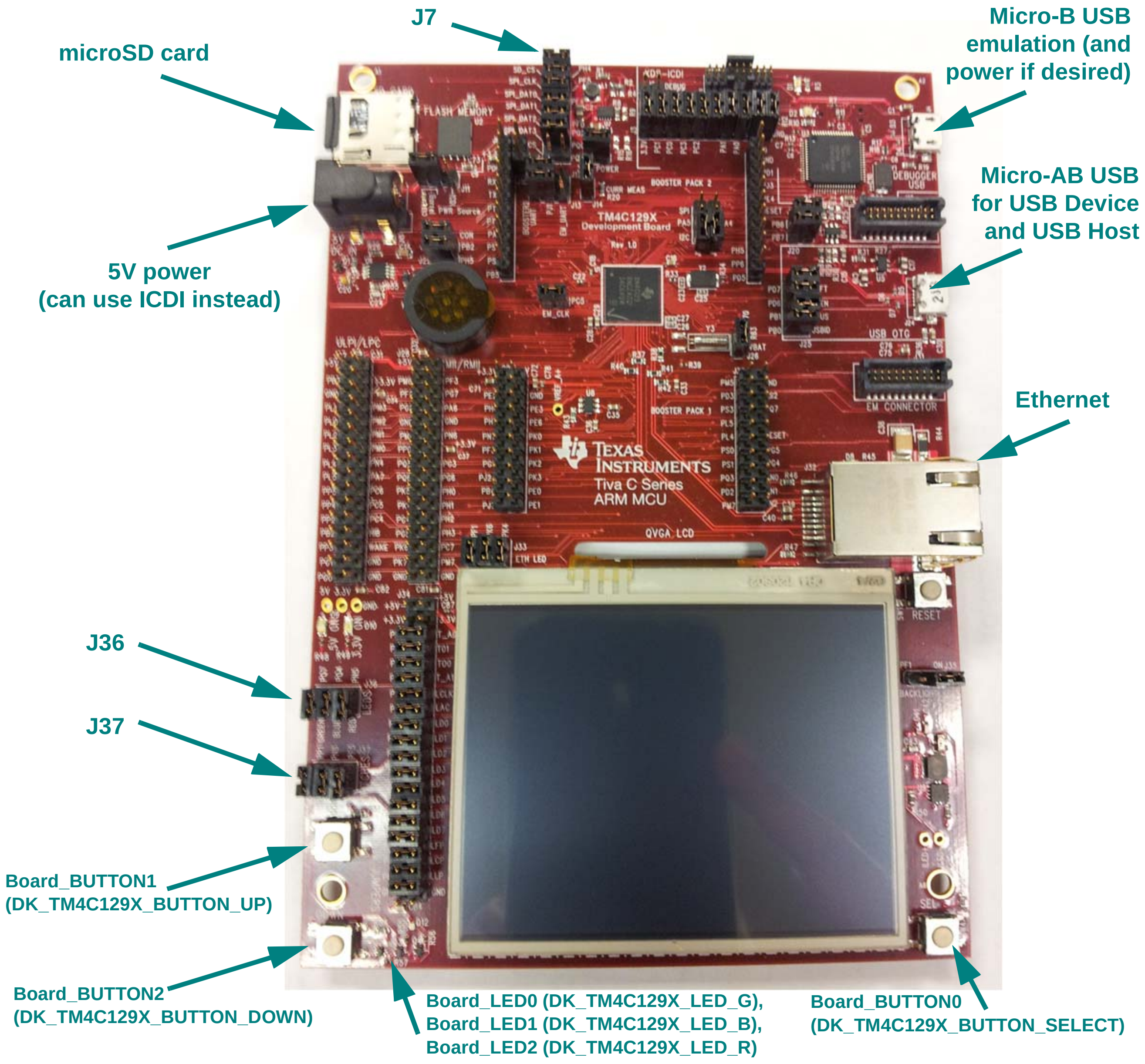
Parity: None

Flow control: XON/XOFF

3.9 Tiva DK-TM4C129X Settings and Resources

The DK-TM4C129X Evaluation Kit contains a Tiva TM4C129XNCZAD device.

Tiva DK-TM4C129X Evaluation Kit connections:


J7
microSD card
Micro-B USB emulation (and power if desired)
Micro-AB USB for USB Device and USB Host
5V power (can use ICDI instead)
Ethernet
J36
J37
Board_BUTTON1 (DK_TM4C129X_BUTTON_UP)
Board_BUTTON2 (DK_TM4C129X_BUTTON_DOWN)
Board_LED0 (DK_TM4C129X_LED_G),
Board_LED1 (DK_TM4C129X_LED_B),
Board_LED2 (DK_TM4C129X_LED_R)
Board_BUTTON0 (DK_TM4C129X_BUTTON_SELECT)

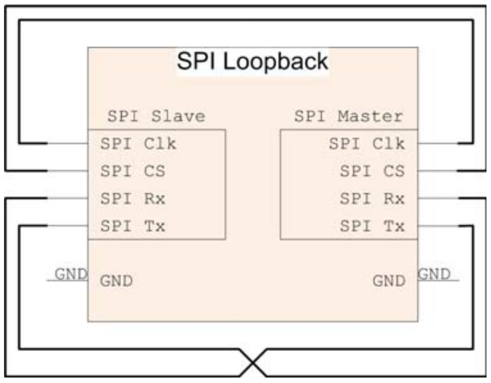
Jumper settings:

- **J7:**
  - — **To use SPI/SSI SD Card slot:** Close SD_CS, SPI_CLK, SPI_DAT0, SPI_DAT1
  - — **To run SPI loopback example:** Open SD_CS, SPI_CLK, SPI_DAT0, SPI_DAT1
- **J37:** Close SEL, DOWN, UP (enables Switches)
- **J36:** Close GREEN, BLUE, RED (enables LEDs)

SPI Loopback example pin connections:

When wiring for SPI loopback, pins on a single board are wired to other pins on the same board.

Master Pin (Function)		Slave Pin (Function)
PQ0 (SPI3CLK)	⟶	PD3 (SPI2CLK)
PQ1 (SPI3FSS)	⟶	PD2 (SPI2FSS)
PF0 (SPI3DAT1 (RX))	⟶	PG5 (SPI2DAT0 (TX))
PQ2 (SPI3DAT0 (TX))	⟶	PG4 (SPI2DAT1 (RX))



Resources Used:

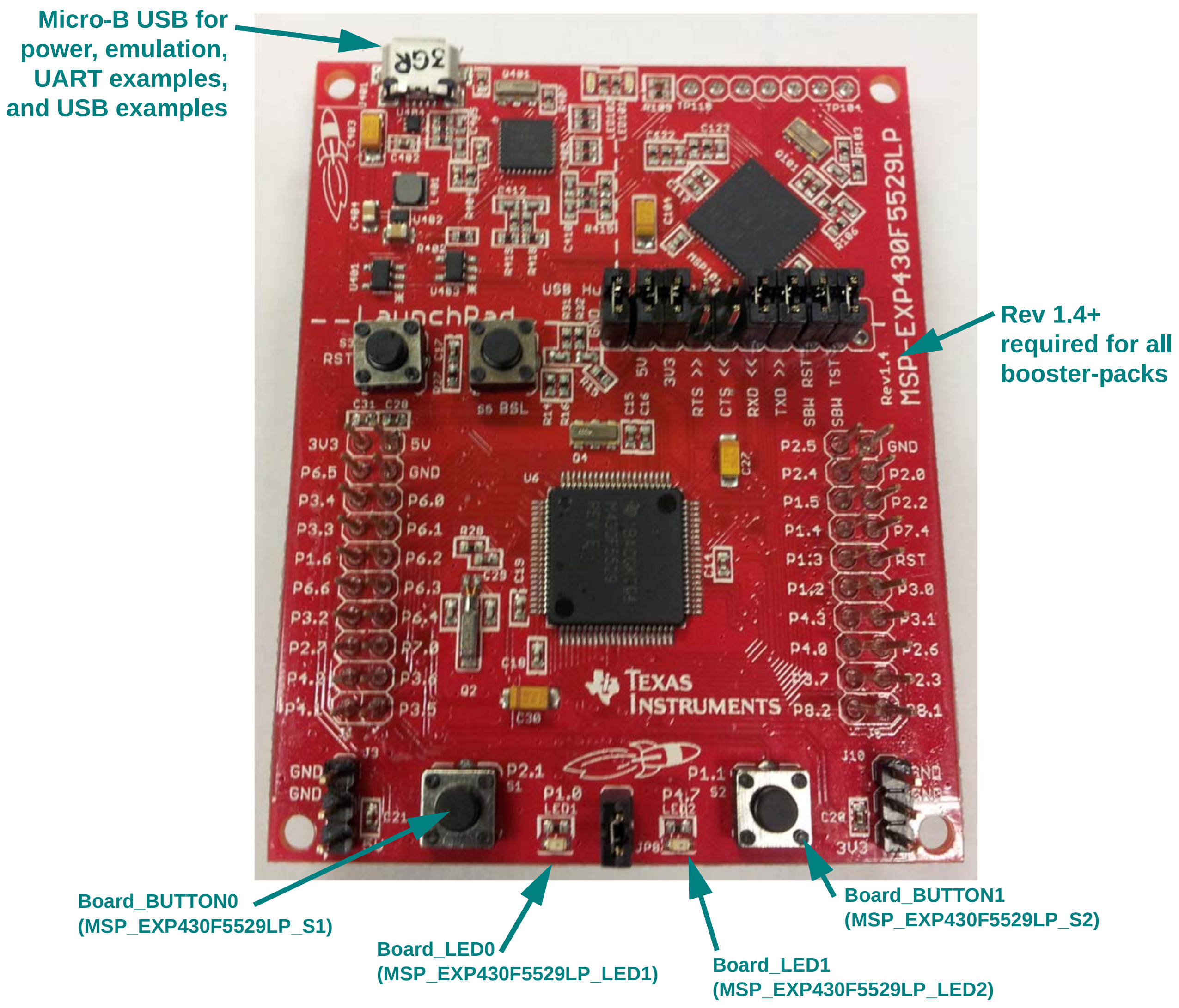
The following list shows which DK-TM4C129Xperipheral resources are used by TI-RTOS applications on that platform. TI-RTOS examples control which peripherals (and which ports) are used

- **SYS/BIOS.** Uses the first general-purpose timer available and that timer's associated interrupts. Generally, this will be Timer 0. SYS/BIOS manages the Interrupt Vector Table.
- **TI-RTOS.**
  - **SD Card.** Uses FatFs and the SDSPI driver on QSSI3 without interrupts to read and write to files.
  - **GPIOs.** The GPIO driver is used on the onboard user LEDs LED0-2 (PK4, PK6, and PF1) as output pins and on three input switches UP, DOWN, and SEL (PN3, PE5, and PP1).
  - **I2C.** The I^{2}C driver is configured on I2C3 to support with various BoosterPacks.
  - **Serial.** The UART driver uses UART0 which is attached to the FTDI USB chip to facilitate serial communications.
  - **SPI.** The SPI driver uses QSSI3 for Board SPI0 and QSSI2 for Board SPI1.
  - **USB.** The USB reference examples use the USB library and the USB controller with its associated interrupts.
  - **Watchdog.** The Watchdog driver example uses Watchdog Timer 0 and its associated interrupt.

3.10 MSP430 MSP-EXP430F5529LP Settings and Resources

The MSP-EXP430F5529LP board contains an MSP430 MSP430F5529 device.

MSP-EXP430F5529LP connections:


Micro-B USB for power, emulation, UART examples, and USB examples
Rev 1.4+ required for all booster-packs
Board_BUTTON0 (MSP_EXP430F5529LP_S1)
Board_LED0 (MSP_EXP430F5529LP_LED1)
Board_LED1 (MSP_EXP430F5529LP_LED2)
Board_BUTTON1 (MSP_EXP430F5529LP_S2)

Jumper settings:

- Set **RXD <<** and **TXD >>** to provide UART communications via the onboard USB debugger.

Examples that use the WiFi driver—for example, the TCP Echo and UDP Echo examples for CC3000—require a CC3000 BoosterPack. This BoosterPack fits over J1 and J5.

SPI Loopback example pin connections:

The SPI loopback example is not supported on MSP430F5529.

Resources Used:

The following list shows which MSP_EXP430F5529LP resources are used by TI-RTOS applications on that platform. TI-RTOS examples control which peripherals (and which ports) are used.

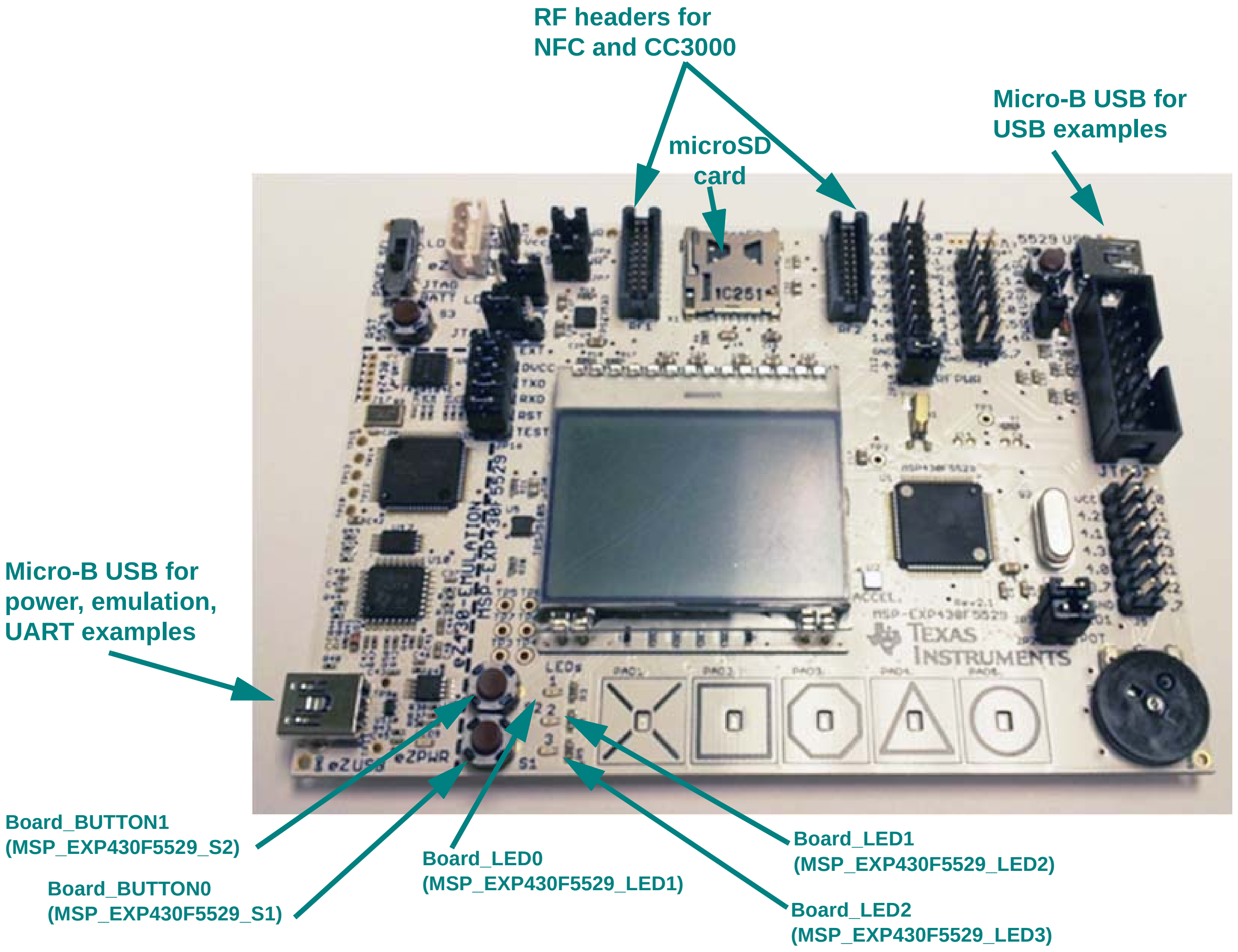
- **SYS/BIOS.** Uses the first general-purpose timer available and that timer's associated interrupts. Generally, this will be Timer0_A0. SYS/BIOS manages the interrupt controller statically without an interrupt dispatcher.
- **TI-RTOS.**
  - **SD Card.** Uses FatFs and the SDSPI driver on USCI_B0 without interrupts to read and write to files.
  - **I^{2}C.** The I^{2}C driver is configured on USCI_B0 to support various BoosterPacks.
  - **GPIOs.** The GPIO driver is used on 2 onboard LEDs: LED1 (P1.0) and LED2 (P4.7) as output pins and 2 switches SW1 P2.1 and SW2 P1.1 as inputs.
  - **Serial.** The UART driver uses USCI_A1, which is attached to the onboard EZ-FET MSP430 USB chip to facilitate serial communications.
  - **SPI.** The SPI driver is configured to use USCI_B1 and USCI_B0 for SPI communications. Each instance requires 2 DMA channels; therefore, only one SPI instance can be used at a time.
  - **USB.** The USB reference examples use the USB library and the USB controller with its associated interrupts.
  - **Watchdog.** The Watchdog driver example uses Watchdog Timer A.
  - **WiFi.** The WiFi examples for use with the CC3000 use GPIO pins P2.0, P2.2, and P6.5 as well as the GPIO interrupt for port 2. In addition, it uses P3.0 and P3.2 for SPI communications.

For MSP430, USB drivers are stored in the USB_config folder in TI-RTOS USB examples in CCS. The USB reference modules for MSP430 use a set of descriptor files that were generated with the MSP430 USB Descriptor Tool.

3.11 MSP430 MSP-EXP430F5529 Settings and Resources

The MSP-EXP430F5529 board contains an MSP430 MSP430F5529 device.

MSP-EXP430F5529 connections:


RF headers for NFC and CC3000
microSD card
Micro-B USB for USB examples
Micro-B USB for power, emulation, UART examples
Board_BUTTON1 (MSP_EXP430F5529_S2)
Board_BUTTON0 (MSP_EXP430F5529_S1)
Board_LED0 (MSP_EXP430F5529_LED1)
Board_LED1 (MSP_EXP430F5529_LED2)
Board_LED2 (MSP_EXP430F5529_LED3)

Jumper settings:

- Set **RXD** and **TXD** to provide UART communications via the onboard USB debugger.

Examples that use the WiFi driver—for example, the TCP Echo and UDP Echo examples for CC3000 and the CC3000 Patcher example—require a CC3000 evaluation module, which fits on the RF1 and RF2 headers.

SPI Loopback example pin connections:

The SPI loopback example is not supported on MSP430F5529.

Resources Used:

The following list shows which MSP_EXP430F5529 resources are used by TI-RTOS applications on that platform. TI-RTOS examples control which peripherals (and which ports) are used.

- **SYS/BIOS.** Uses the first general-purpose timer available and that timer's associated interrupts. Generally, this will be Timer0_A0. SYS/BIOS manages the interrupt controller statically without an interrupt dispatcher.
- **TI-RTOS.**
  - **SD Card.** Uses FatFs and the SDSPI driver on USCI_B1 without interrupts to read and write to files.
  - **I^{2}C.** The I^{2}C driver is configured on USCI_B0 and USCI_B1.
  - **GPIOs.** The GPIO driver is used on 8 onboard LEDs: LED1 (P1.0), LED2 (P8.1), LED3 (P8.2), PAD1 (P1.1), PAD2 (P1.2), PAD3 (P1.3), PAD4 (P1.4), and PAD5 (P1.5) as output pins and 2 switches S1 (P2.7) and S2 (P2.2) as inputs.
  - **Serial.** The UART driver uses USCI_A1, which is attached to the onboard EZ-FET MSP430 USB chip to facilitate serial communications.
  - **SPI.** The SPI driver is configured to use USCI_B1 and USCI_B0 for SPI communications. Each instance requires 2 DMA channels; therefore, only one SPI instance can be used at a time.
  - **USB.** The USB reference examples use the USB library and the USB controller with its associated interrupts.
  - **Watchdog.** The Watchdog driver example uses Watchdog Timer A.
  - **WiFi.** The WiFi examples for use with the CC3000 use GPIO pins P2.4, P2.6, and P2.3 as well as the GPIO interrupt for port 2. In addition, it uses P3.0 and P3.2 for SPI communications.

For MSP430, USB drivers are stored in the USB_config folder in TI-RTOS USB examples in CCS. The USB reference modules for MSP430 use a set of descriptor files that were generated with the MSP430 USB Descriptor Tool.

3.12 BoosterPacks

Several BoosterPack boards are used with the TI-RTOS examples. This section described those boards and provides any special notes about installing the board.

3.12.1 SD Card BoosterPack

An SD Card BoosterPack should be used with examples that require an SD Card reader on target boards that do not include an SD Card reader. This board may be used with the EK-TM4C123GXL, EK-LM4F120XL, and MSP-EXP430F5529LP.

3.12.2 TMP006 BoosterPack

The I^{2}C TMP006 example uses the TMP006 BoosterPack to connect the TMP006EVM circuit board to the LaunchPad. The TMP006 BoosterPack kit, which includes a TMP006EVM circuit board, is available online. This board may be used with the EK-TM4C123GXL, EK-LM4F120XL, DK-TM4C129X, and MSP-EXP430F5529LP.

3.12.3 CC3000 BoosterPacks

The TI CC3000 is a self-contained wireless network processor that simplifies the implementation of Internet connectivity. TI's SimpleLink Wi-Fi solution minimizes the software requirements of the host microcontroller (MCU) and is thus the ideal solution for embedded applications using any low-cost and low-power MCU.

Two types of CC3000 BoosterPacks are currently available:

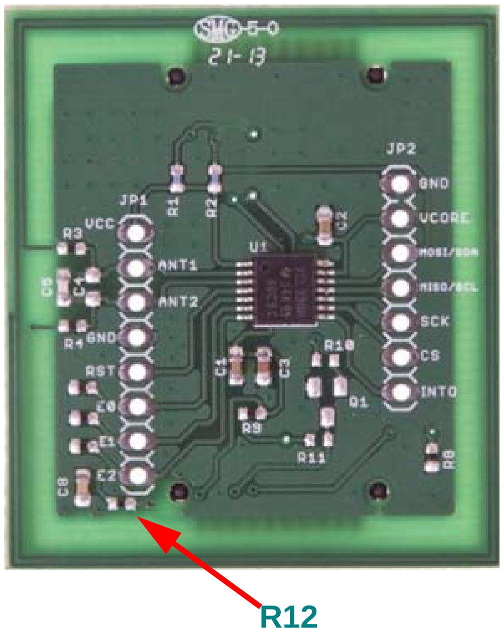
- CC3000 BoosterPack, which can be used with the following boards:
  - EK-LM4F120XL
  - EK-TM4C123GXL
  - MSP-EXP430F5529LP
- CC3000 Evaluation Module (EM), which can be used with the following boards:
  - DK-TM4C123G
  - EKS-LM4F232
  - MSP-EXP430F5529

3.12.4 RF430CL330 NFC Transponder Module

The I^{2}C RF430CL330 Load example uses the I^{2}C driver to communicate with a RF430CL330 NFC transponder module. This board may be used with the EK-TM4C123GXL, EK-LM4F120XL, DK-TM4C129X, and MSP-EXP430F5529LP.

A Wireless Connectivity Adaptor board is also required in order to use the NFC Transponder Module with a LaunchPad board.

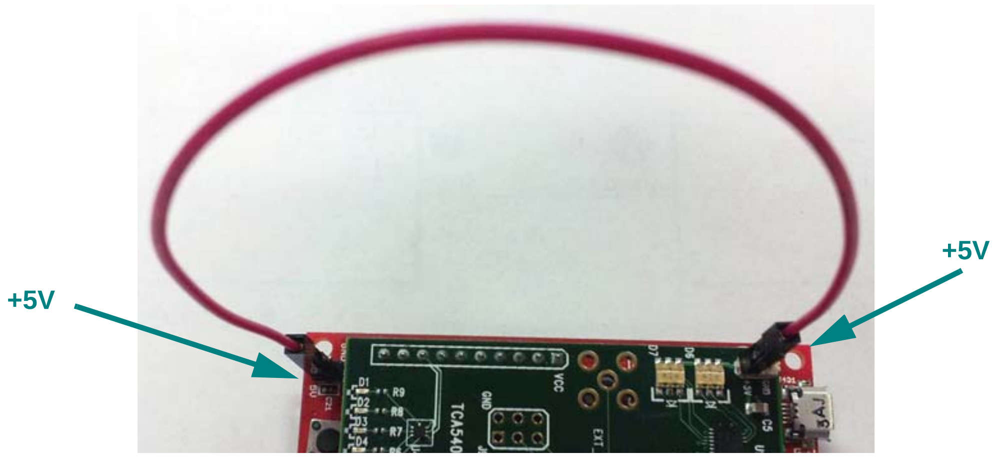
Hardware Modification Note: In order to use the RF430CL330 module with a Wireless Connectivity Adapter board, you need to remove R12 from the RF430CL330H board.



3.12.5 TPL0401EVM Board

The I^{2}C TPL0401EVM example requires an attached TPL0401EVM board. This board may be used with the EK-TM4C123GXL, EK-LM4F120XL, DK-TM4C129X, and MSP-EXP430F5529LP.

A wire like the one shown in the following figure should be added to provide power to the TPL0401EVM board. This figure shows the wire on an MSP-EXP430F5529LP board. Similar wiring is required to connect the TPL0401EVM board to other boards supported by this example. A 5V power supply is recommended to provide sufficient LED brightness.



3.13 Installing USB Drivers for the USB Device Examples

The USB examples build upon the examples provided with TivaWare, MWare, and MSP430Ware. Because the examples mimic the same functionality, you can use the same drivers delivered with standalone installations of TivaWare, controlSUITE (MWare), and MSP430Ware.

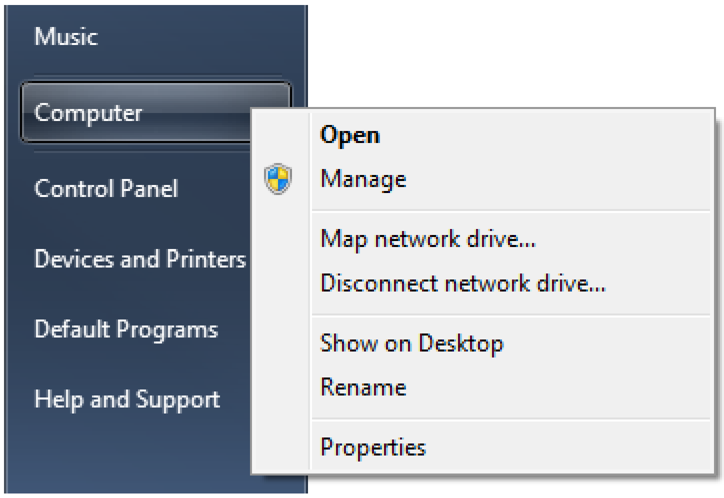
Windows USB drivers for Tiva and MWare are located in the `<tirtos_install>/products/TivaWare_C_Series-1.#/windows_drivers` and `<tirtos_install>/products/MWare_v20##/MWare/windows_drivers` directories, respectively. USB drivers for MSP430 are located in each example's `USB_config` folder.

The Windows menus and dialogs you see may be slightly different from those shown here, depending on your version of Windows.

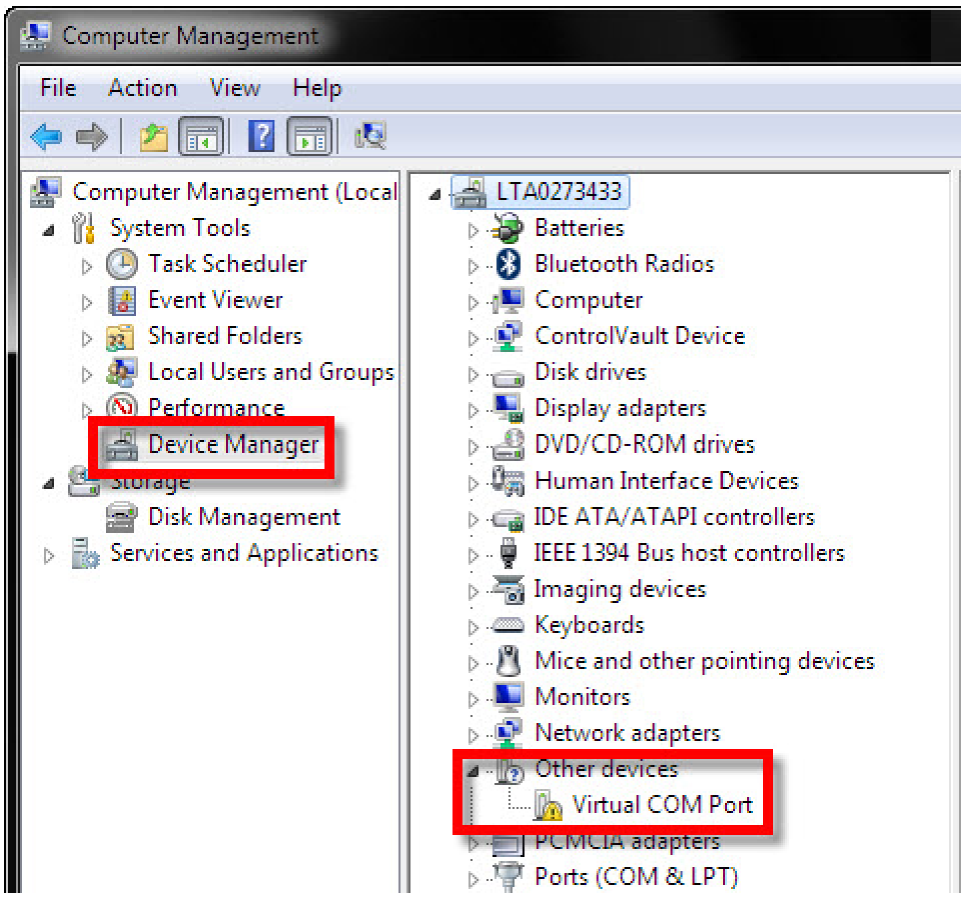
To install the USB driver, follow these steps:

1. Load and run a USB device reference example—USB Keyboard Device, USB Mouse Device, or USB CDC Mouse Device.
2. While the example is running, connect the device to the Windows PC via a USB cable. At this point, Windows will detect the device and attempt to enumerate it.

3. Open the Windows Device Manager by right-clicking on the **My Computer** desktop icon or **Computer** in the Start Menu and selecting **Manage**.



4. If you are prompted by a security warning in Windows 7, click **Yes**.
5. Select the **Device Manager** category in the left pane.
6. In the center pane, select the unknown driver that you are trying to install. For example, the device shown here is for the USB CDC driver.



7. Right-click on the device, and select **Update Driver Software**.
8. Select **Browse my computer for driver software** and browse to the location of the Windows USB drivers, `<tirtos_install>/products/TivaWare_C_Series-1.#/windows_drivers`. Make sure the box to **Include subfolders** is checked.
9. Click **Next** to run the installation wizard. If you see a Window Security prompt, click **Install**.
10. After the driver is installed, you can determine the COM port number for the CDC (Virtual COM Port) device.

Chapter 4

SPRUHD3G—January 2014

Configuring TI-RTOS

This chapter describes how to configure how TI-RTOS and its components will be used by your application.

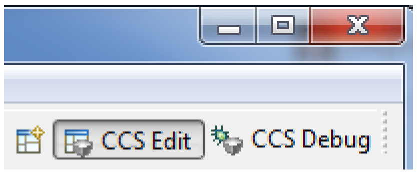
Topic **Page**

4.1 Starting the Configuration Tool

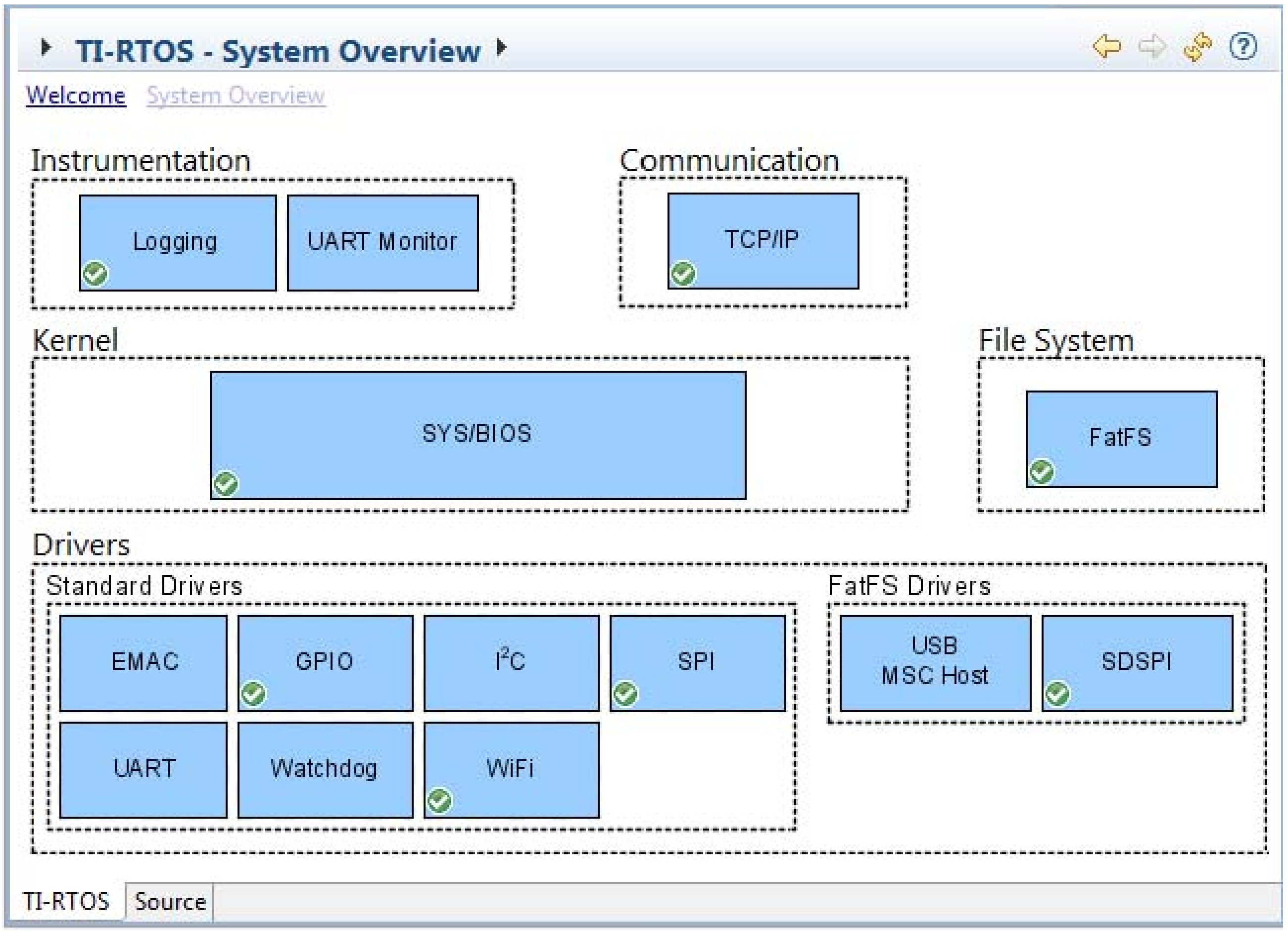
Note: The graphical configuration tool is not available within IAR Embedded Workbench. If you are using IAR, edit the project's *.cfg file within IAR as a text-based source file. See the Texas Instruments Wiki for more about using IAR with TI-RTOS.

To use CCS to open the graphical tool for editing configuration files (XGCONF), follow these steps:

1. Make sure you are in the **C/C++** perspective of CCS. If you are not in that perspective, click the C/C++ icon to switch back.



2. Double-click on the *.cfg configuration file for a TI-RTOS example project in the **Project Explorer** tree. (See Section 3.1 if you need to create an example project.) While XGCONF is opening, the CCS status bar shows that the configuration is being processed and validated.
3. When XGCONF opens, you see the **Welcome** sheet for TI-RTOS. This sheet provides links to TI-RTOS documentation resources.
4. Click the **System Overview** button to see a handy overview of the components available through the TI-RTOS. The green check marks indicate which modules are being used by the application.



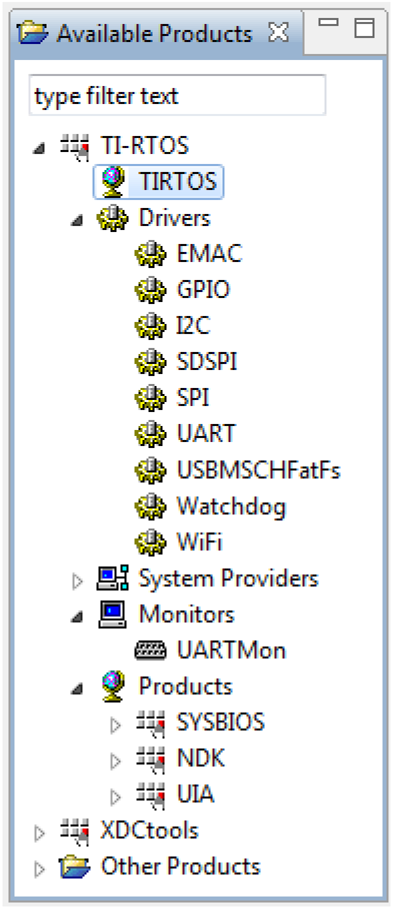
5. Click a blue box in the System Overview to go to the Welcome sheet for that component or the configuration page for a driver. If a feature is not supported for your target, you cannot click it. For example, you cannot click TCP/IP for MSP430.

Note: If the configuration is shown in a text editor instead of XGCONF, close the text editor window. Then, right-click on the .cfg file and choose **Open With > XGCONF**. If you are comfortable editing configuration scripts with a text editor, you can do that. However, you should not have the file open in both types of editor at the same time.

For details about how to use XGCONF, see Chapter 2 of the SYS/BIOS User's Guide (SPRUEX3).

4.2 Configuring TI-RTOS

When you open a configuration file in XGCONF, you see a list of Available Products. The components listed here are the same components that were checked in the RTSC Configuration Settings page when you created a new CCS project using one of the TI-RTOS project templates.



Under the TI-RTOS item, you can select one of the modules listed to configure it. Most of these modules are peripheral drivers. For each driver, you can select to use either the instrumented or non-instrumented libraries when linking. The instrumented libraries process Log events while the non-instrumented libraries do not. See the section on "Using Instrumented or Non-Instrumented Libraries" in the *TI-RTOS User's Guide* (SPRUHD4) for more information. The following drivers and transports can be configured:

- **EMAC.** Driver for Ethernet with the NDK. See Section 3.5.1 for how to set the MAC address.
- **GPIO.** Driver for the GPIO pins (and therefore the LEDs).
- **I2C.** Driver for the I^{2}C peripheral.
- **SDSPI.** Driver for the SD card using a SPI (SSI) bus and the FatFS.
- **SPI.** Driver for the Serial Peripheral Interface (SPI) bus.
- **SPIMessageQTransport.** MessageQ transport for the SPI driver for use in multicore applications.
- **UART.** Driver for the UART peripheral.
- **USBMSCHFatFs.** Driver for the USB MSC Host controller.
- **Watchdog.** Driver for the watchdog timer.
- **WiFi.** Driver to communicate with TI Wi-Fi device such as a SimpleLink Wi-Fi CC3000.

The SysCallback module lets you configure the functions that handle System output—for example, System_printf() and System_abort(). This module handles transmissions to System output only; it does not handle responses received. See the chapter on "TI-RTOS Utilities" in the *TI-RTOS User's Guide* (SPRUHD4) for more about the SysCallback module.

Other SystemSupport implementations are provided with XDCtools.

- **SysMin** stores System_printf() strings in an internal buffer in RAM. SysMin requires RAM, so it not ideal for devices with minimal RAM.
- **SysStd** writes System_printf() strings to STDOUT (the CCS Console window). By default, SysStd allows System_printf() to be called from Tasks only (not Swis or hardware interrupts); it can be modified to allow calls from Swis and Hwis, but this impacts real-time performance.

4.3 Configuring Individual Sub-Components

For information about configuring individual sub-components of TI-RTOS, see the documentation for that component. Chapter 2 of the SYS/BIOS User's Guide (SPRUEX3) provides information about using XGCONF.

Within XGCONF, you can see the full file path to the version of the component being used by hovering your mouse cursor over a component in the "Other Products" list in the **Available Products** area.

The CCS online help lets you access the CDOC reference help for various components. For information about properties you can configure and instances you can create, scroll down in CDOC to see the sections with red outlines. (You can use the XDCscript link at the top of each page to skip to the configuration information.)

Revision History

Table A–1 lists the changes made since the previous version of this document.

Table A–1. Revision History

Revision	Chapter	Location	Additions/Modifications/Deletions
SPRUHD3G	Preface		Current software version number is v1.21.
	About	Section 1.3	Added MSP-EXP430F5529 Experimenter Board.
	Examples	Section 3.1	TI-RTOS examples for compilation with TI or GNU code generation tools are now available for certain boards in TI Resource Explorer.
		Section 3.1.1	Minimal memory version of the Empty Project is now available.
		Section 3.2	Added section on generating the examples directory.
		Section 3.3 to Section 3.3.4	Revised section on using examples with IAR Embedded Workbench substantially.
		Section 3.4	Added MSP-EXP430F5529 Experimenter Board.
		Section 3.7 to Section 3.11	Distinguished between boards that require a CC3000 EM board or a CC3000 BoosterPack. See "Jumper Settings" in each section. Also identified when EM headers may need to be ordered separately.
		Section 3.11	Added MSP-EXP430F5529 Experimenter Board.
		Section 3.12.3	Added description of CC3000 BoosterPacks.
SPRUHD3F	Preface		Current software version number is v1.20.
	About	Section 1.2	MSP430Ware component added.
		Section 1.3	Several new boards added, including an MSP430 board. Also added information about building drivers for MSP430 boards other than MSP430F5529.
		Section 1.5	New section added to give an overview of the examples. Information about hardware resources previously in this section has been moved to Chapter 3.
		Section 1.6	Links to information about MSP430Ware, MSP430 boards, and Boost-erPacks added.

Table A–1. Revision History

Revision	Chapter	Location	Additions/Modifications/Deletions
	Installing	Section 2.1	Code Composer Studio 5.5 or higher is now required. Using TI-RTOS with IAR Embedded Workbench is also now supported.
		Section 2.3.2	New section added on how to fix installation if TI-RTOS is installed outside CCS.
		Section 2.3.4	New section added on how to install TI-RTOS for use with IAR Embedded Workbench.
	Examples	Section 3.1.1	Moved this section to this document from the *TI-RTOS User's Guide*.
		Section 3.3 to Section 3.3.6	New sections added on using examples with IAR Embedded Workbench.
		Table 3-1	New examples added for I^2C BoosterPacks. Fewer examples use UIA.
		Table 3-2	Several new boards added. Also added footnotes about BoosterPacks.
		Section 3.5 to Section 3.10	Made sections about boards higher-level headings. Added photos of boards with callouts for connections. Moved information from Chapter 1 about hardware resource requirements into these sections.
		Section 3.12 to Section 3.12.5	New sections added about the BoosterPacks that can be used for various examples.

Index

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video & Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Mobile Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		