TI-RTOS 2.14 for TivaC

Getting Started Guide

Contents

Read This First

About This Manual

This manual describes TI-RTOS for TivaC. The version number as of the publication of this manual is v2.14.

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, MSP430, MSP430Ware, MSP432, SimpleLink, Sitara, SPOX, TI-RTOS, 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.

July 9, 2015

About TI-RTOS

This chapter provides an overview of TI-RTOS for TivaC.

1.1 What is TI-RTOS?

TI-RTOS is a scalable, one-stop embedded tools ecosystem for TI devices. It scales from a real-time multitasking kernel (SYS/BIOS) to a complete RTOS solution including additional middleware components and device drivers. By providing essential system software components that are pre-tested and pre-integrated, TI-RTOS enables you to focus on differentiating your application.

TI-RTOS is *not* installed automatically as part of the Code Composer Studio v6.0 installation. You can install TI-RTOS from the CCS App Center (choose **View > CCS App Center** in CCS). Choose the version of TI-RTOS for your device family. If you use devices in multiple families, you can install multiple TI-RTOS versions. See Section 2.3 for details.

If you do not use CCS, you can download and install TI-RTOS as a standalone product (see Section 2.4). In addition to the Texas Instruments Code Generation Tools, TI-RTOS includes support for the IAR and GNU tool chains (see Section 2.5).

TI-RTOS is provided with full source code and requires no up-front or runtime license fees.

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. The components of TI-RTOS for TivaC are as follows.

Table 1–1. TI-RTOS Components

TI-RTOS Component	Name	PDF Documentation Location
TI-RTOS	TI-RTOS examples	Chapter 3 of this Getting Started Guide
TI-RTOS Kernel	SYS/BIOS	*SYS/BIOS (TI-RTOS Kernel) User's Guide* -- SPRUEX3
TI-RTOS Instrumentation	UIA	*System Analyzer User's Guide* -- SPRUH43
TI-RTOS Networking	NDK	*TI Network Developer's Kit (NDK) Guide* -- SPRU523 *TI Network Developer's Kit (NDK) API Reference* -- SPRU524
TI-RTOS File System	FatFS	*TI-RTOS User's Guide* -- SPRUHD4
TI-RTOS USB	USB stack	*TI-RTOS User's Guide* -- SPRUHD4
TI-RTOS Drivers and Board Initialization	Drivers and TivaWare	*TI-RTOS User's Guide* -- SPRUHD4
TI-RTOS Network Services	Network Services	*TI-RTOS User's Guide* -- SPRUHD4

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

- **TI-RTOS Kernel — 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.

- **TI-RTOS Instrumentation — UIA.** The Unified Instrumentation Architecture (UIA) provides target content that aids in the creation and gathering of instrumentation data (for example, Log data).

- **TI-RTOS Networking — NDK.** The Network Developer's Kit (NDK) is a platform for development and demonstration of network enabled applications on TI embedded 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.[1]

- **XDCtools.** This core component provides the underlying tooling for configuring and building TI-RTOS and its components. XDCtools is installed as part of CCS v6.x. If you install TI-RTOS outside CCS, a compatible version of XDCtools is installed automatically. XDCtools is installed in a directory at the same level as TI-RTOS, not in the "products" directory of the TI-RTOS installation.

- **wolfSSL.** This embedded SSL library (formerly CyaSSL) is a lightweight, C-language-based SSL/TLS library. The wolfSSL library supports industry standards up to the current TLS 1.2 and DTLS 1.2, and can be as small as 1/20th the size of OpenSSL. The wolfSSL library is not installed as part of TI-RTOS. You can download and install it separately.

---

1. The 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 TivaWare component of TI-RTOS for details.

1.3 How Can I Find Example Projects?

TI-RTOS and its components provide numerous examples that you can import using the **TI Resource Explorer** in Code Composer Studio (CCS). These examples use TI-RTOS and its components and have all the settings needed for your device. Expand the tree in the TI Resource Explorer to see the examples that are available for your device.TI Resource Explorer provides TI-RTOS examples for both the TI and GNU tool chains.

- **Driver Examples** are TI-RTOS driver examples.

- **Instrumentation Examples** are UIA examples.

- **Kernel Examples** are the SYS/BIOS examples.

- **Network Examples** show how to use capabilities such as WiFi, PPP, Ethernet, and HTTP/HTTPS.

Follow the steps in Section 3.1 to import, build, and run these examples.
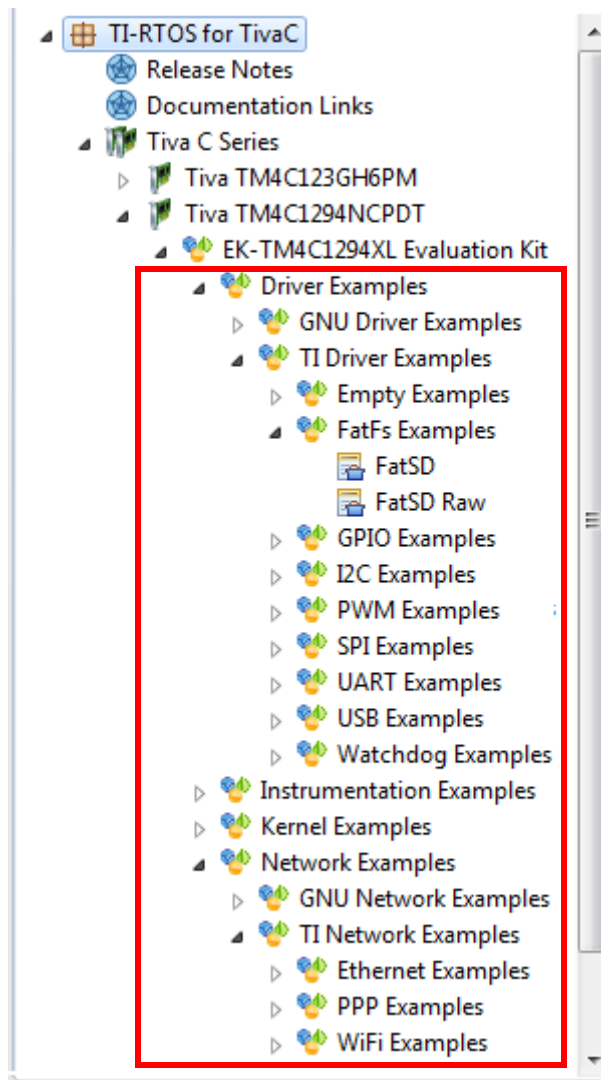
1.4   What Compilers and Targets are Supported?

The following code generation tool (compilers and linkers) versions are supported. The versions listed are recommended because they were used to build the TI-RTOS libraries and to perform testing. More recent versions are expected to be compatible.

- **Texas Instruments:** ARM CodeGen Tools v5.2.2

- **GCC:** gcc-arm-none-eabi-4_8-2014q3

- **IAR Workbench for ARM:** 7.40.1

The configuration uses a "target" specification during the build. This specification is sometimes called the "RTSC target." The targets supported are:

- ti.targets.arm.elf.M4F

- iar.targets.arm.M4F

- gnu.targets.arm.M4F

1.5   What Boards and Devices Have TI-RTOS Driver Examples?

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

Family	Device on Board	Board
ARM (Tiva)	TM4C123GH6PM	EK-TM4C123GXL LaunchPad
ARM (Stellaris)	LM4F120H5QR	EK-LM4F120XL LaunchPad (earlier version of EK-TM4C123GXL LaunchPad)
ARM (Tiva)	TM4C129XNCZAD	DK-TM4C129X Evaluation Kit
ARM (Tiva)	TM4C1294NCPDT	EK-TM4C1294XL Evaluation Kit
ARM (Tiva)	TM4C129ENCPDT	EK-TM4C129EXL Evaluation Kit

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.

1.6   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 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.

- **PWM.** API set intended to be used directly by the application or middleware to generate Pulse Width Modulated signals.

- **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.

1.7 For More Information

To see release notes for a component, go to the subdirectory for that component within the TI-RTOS products directory. For example, `C:\ti\tirtos_tivac_2_##_##_##\products\bios_6_40_##_##` contains release notes for SYS/BIOS.

To see user guide PDFs and other documentation for a component, go to the "docs" subdirectory within the directory that contains the release notes.

To learn more about TI-RTOS and its components, refer to the following documentation:

- **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
  - CCSv6 on the Texas Instruments Wiki
  - Code Composer forum on TI's E2E Community

- **SYS/BIOS**
  - SYS/BIOS User's Guide (SPRUEX3)
  - SYS/BIOS API and configuration reference. In TI Resource Explorer (in CCS), choose the **Documentation Links** item for your version of TI-RTOS. In the Documentation Links page, choose the **TI-RTOS Kernel Runtime APIs and Configuration (cdoc)** item.
  - SYS/BIOS on the Texas Instruments Wiki
  - TI-RTOS forum on TI's E2E Community
  - SYS/BIOS 6.x Product Folder

- **XDCtools**
  — SYS/BIOS User's Guide (SPRUEX3)
  — 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

- **NDK**
  — NDK User's Guide (SPRU523)
  — NDK Programmer's Reference Guide (SPRU524)
  — NDK API reference. Run *<tirtos_install>*/products/ndk_#_##_##_##/docs/doxygen/html/index.html.
  — NDK configuration reference. In TI Resource Explorer (in CCS), choose the **Documentation Links** item for your version of TI-RTOS. In the Documentation Links page, choose the **TI-RTOS Networking Configuration (cdoc)** item.
  — NDK on the Texas Instruments Wiki
  — TI-RTOS forum on TI's E2E Community

- **UIA**
  — System Analyzer User's Guide (SPRUH43)
  — UIA API and configuration reference. In TI Resource Explorer (in CCS), choose the **Documentation Links** item for your version of TI-RTOS. In the Documentation Links page, choose the **TI-RTOS Instrumentation Runtime APIs and Configuration (cdoc)** item.
  — System Analyzer on the Texas Instruments Wiki

- **TivaWare**
  — Documents in *<tirtos_install>*/products/TivaWare_C_Series-#.#/docs
  — TivaWare Product Folder
  — Online StellarisWare Workshop

- **wolfSSL**
  — Download from TI Store (TI licensed product)
  — Download from wolfSSL (GPL licensed product)
  — Using wolfSSL with TI-RTOS wiki page

- **FatFS API**
  — Open source documentation
  — FatFS for SYS/BIOS wiki page
  — SYS/BIOS API and configuration reference. In TI Resource Explorer (in CCS), choose the **Documentation Links** item for your version of TI-RTOS. In the Documentation Links page, choose the **TI-RTOS Kernel Runtime APIs and Configuration (cdoc)** item and see help under the ti.sysbios.fatfs.FatFS module topic.

- **General microcontroller information**
  — Microcontrollers forum on TI's E2E Community

- **Tiva boards and devices**
  - — Tiva C Series TM4C123G LaunchPad Evaluation Kit
  - — TM4C123GH6PM Tiva C Series Microcontroller
  - — TM4C123GH6PGE Tiva C Series Microcontroller
  - — EK-TM4C1294XL LaunchPad Evaluation Board
  - — EK-TM4C129EXL LaunchPad Evaluation Board
- **BoosterPacks**
  - — EM Adapter BoosterPack
  - — RF430CL330H NFC Dynamic Tag Target Board
  - — MicroSD BoosterPack
  - — SD Card BoosterPack
  - — TMP006 BoosterPack
  - — I2C TPL0401EVM BoosterPack
- **SD Cards**
  - — Specification
- **I^{2}C**
  - — Specification
- **WiFi**
  - — SimpleLink Wi-Fi CC31xx/CC32xx Wiki Page
  - — CC3100 Product Folder
  - — CC3200 Product Folder
  - — SimpleLink Wi-Fi SmartConfig Apps

Installing TI-RTOS

This chapter covers the steps to install TI-RTOS within Code Composer Studio or as a standalone software product.

2.1 System Requirements

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

The Linux version of TI-RTOS can be installed on systems that are running Linux RedHat v4 and higher or Ubuntu v10.04 and higher.

Separate versions of TI-RTOS are available for various Texas Instruments device families.

In order to install TI-RTOS, you must have at least 1 GB of free disk space. (If you have not yet installed Code Composer Studio, you will also need at least 4 GB of disk space for that installation.)

2.2   Installing Code Composer Studio

TI-RTOS is used in conjunction with Code Composer Studio 6.0 or higher. (TI-RTOS can also be used with the IAR Embedded Workbench IDE. See page 13 for more information.) CCS is available for Microsoft Windows and Linux.

For Windows installations, we 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 6.x, go to the "Download CCS" page on the Texas Instruments wiki and follow a link to download the software for your license type. For multi-user licenses, see the CCS product page.

Run the installer, and answer the prompts as appropriate.

2.3   Installing TI-RTOS in Code Composer Studio

TI-RTOS is *not* installed automatically as part of the Code Composer Studio v6.0 installation. Instead, you install it through the CCS App Center. Follow these steps to install TI-RTOS:

1. Run CCS v6.1 or higher.

2. Choose **View > CCS App Center** in CCS.

3. Select the version of TI-RTOS for your device family. If you use devices from multiple families, you can select multiple TI-RTOS versions.

4. Click the **Install Software** button near the top of the App Center view.
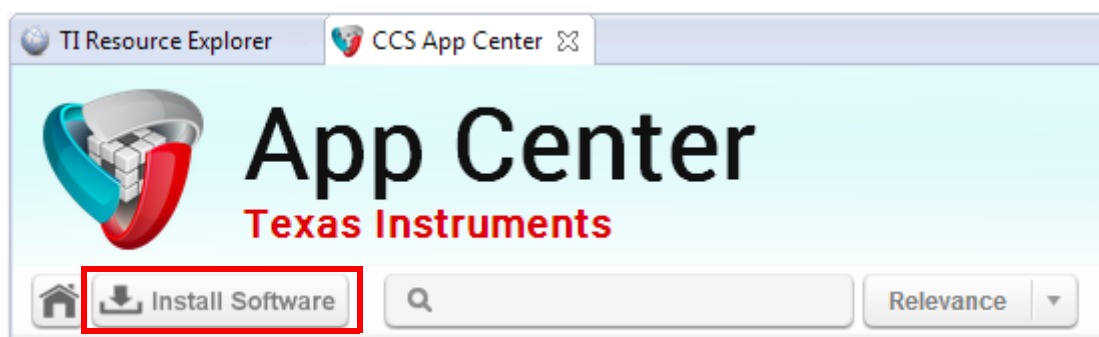
5. Answer the prompts as necessary to complete the TI-RTOS installation.

6. Restart CCS in order for TI-RTOS and its components to be available.

2.4   Installing TI-RTOS as a Standalone Product

If you do not use Code Composer Studio, you can install TI-RTOS as a standalone product. In addition to compiling and linking with the Texas Instruments Code Generation Tools, TI-RTOS includes support for the IAR and GNU tool chains.

1. Download the Windows or Linux installer for TI-RTOS for the device family you use. For example, `tirtos_setupwin32_tivac_2_##_##_##.exe` or `tirtos_setuplinux_tivac_2_##_##_##.bin`.

2. Run the downloaded file to install TI-RTOS. You can install TI-RTOS in a standalone directory. Installing in a directory path that contains spaces, such as `C:\Program Files (x86)`, is not recommended.

    Note: TI-RTOS installs the core functionality of the XDCtools component if you have not already installed the necessary version as part of a CCS 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 located in `C:\ti\tirtos_tivac_2_##_##_##`, the XDCtools directory will be in `C:\ti\xdctools_3_31_##_##_core`.

Follow the instructions in Section 3.3 to complete the installation of the TI-RTOS examples.

2.5   Installing TI-RTOS for Use in IAR Embedded Workbench

If you do not use Code Composer Studio, you can install TI-RTOS for use with IAR Embedded Workbench. In addition to compiling and linking with the Texas Instruments Code Generation Tools, TI-RTOS includes support for the IAR and GNU tool chains.

1. Install IAR Embedded Workbench for Texas Instruments ARM devices.

2. Download the Windows installer for TI-RTOS for TivaC. For example, `tirtos_setupwin32_tivac_2_##_##_##.exe`.

3. Run the downloaded file to install the full TI-RTOS product. You can install TI-RTOS in a standalone directory. Installing in a directory path that contains spaces, such as `C:\Program Files (x86)`, is not recommended.

    Note: TI-RTOS installs the core functionality of the XDCtools component if you have not already installed the necessary version as part of a CCS 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:\ti\tirtos_tivac_2_##_##_##`, the XDCtools directory will be `C:\ti\xdctools_3_31_##_##_core`.

Follow the instructions in Section 3.3 to complete the installation of the TI-RTOS examples. See Section 3.4 to build the examples with the IAR compiler and to load and run the examples with IAR Embedded Workbench.

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.

3.1   Creating Example Projects Using the TI Resource Explorer

You can use the **TI Resource Explorer** in Code Composer Studio (CCS) to create example projects that use TI-RTOS and its components and have all the settings needed for your device. Follow these steps:

1. Open CCS. If you do not see the TI Resource Explorer, make sure you are in the **CCS Edit** perspective and choose **View > Resource Explorer (Examples)** from the menus.

2. Type the name or part of the name of your device in the **enter search keyword** field to hide all the examples that don't apply to your device. Or, you can type "Driver Examples" to find TI-RTOS driver examples.

3. Expand the tree until you see the examples for your device. Any **Driver Examples** listed are TI-RTOS driver examples. Any **Instrumentation Examples** listed are UIA examples. The **Kernel Examples** are the TI-RTOS Kernel (SYS/BIOS) examples. **Network Examples** use capabilities such as WiFi, PPP, Ethernet, and HTTP/HTTPS. TI Resource Explorer provides TI-RTOS examples for both the TI and GNU tool chains.
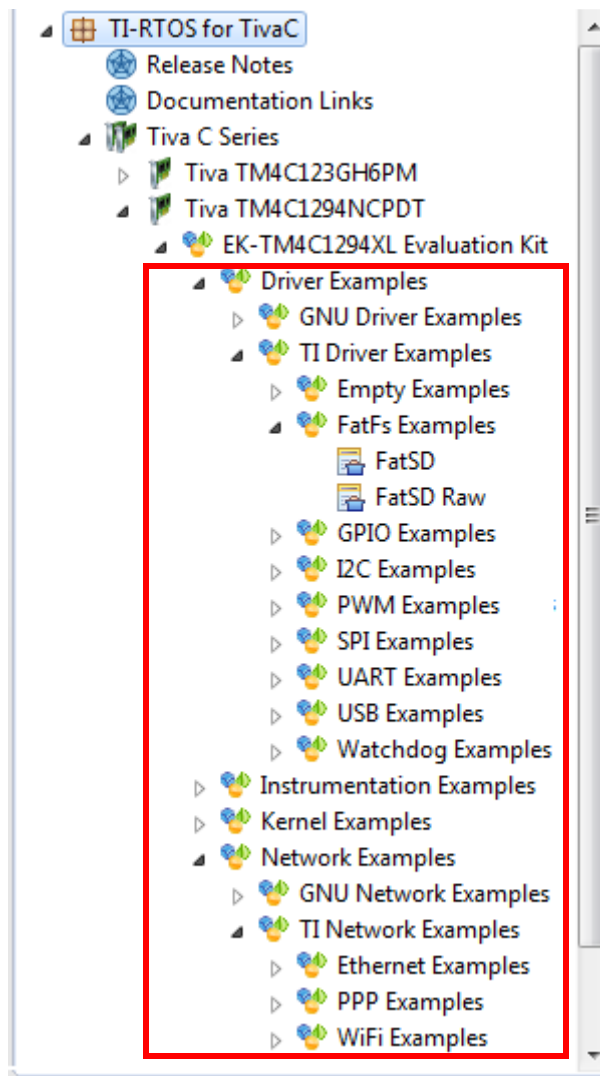
4. Select the example you want to create. A description of the selected example is shown to the right of the example list.
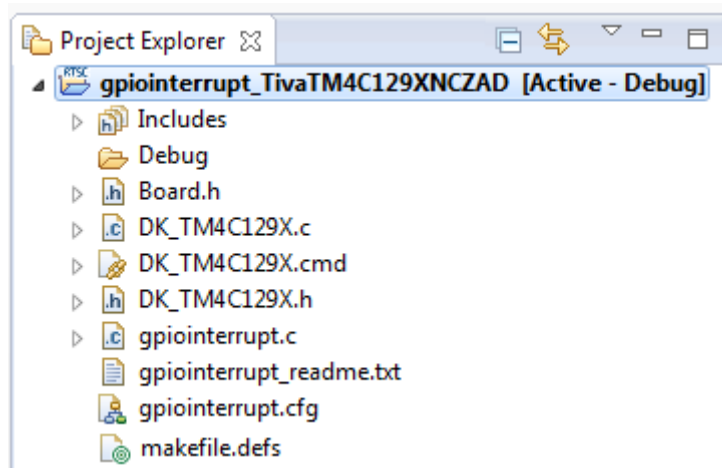
5. Click the **Step 1** link in the right pane of the TI Resource Explorer to **Import the example project into CCS**. This adds a new project to your Project Explorer view. Once you have completed a step for a particular example and device, a green checkmark will be shown next to that step.

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.*

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.

6. The project created will have a name with the format *<example_name>_<board>*. You can expand the project to see the source code, configuration, and other files in the project.

Project Explorer ⊠
▲ 🗁 **gpiointerrupt_TivaTM4C129XNCZAD [Active - Debug]**
    ▷ 🔺 Includes
        🗀 Debug
    ▷ 🗎 Board.h
    ▷ 🗎 DK_TM4C129X.c
    ▷ 🗎 DK_TM4C129X.cmd
    ▷ 🗎 DK_TM4C129X.h
    ▷ 🗎 gpiointerrupt.c
        🗎 gpiointerrupt_readme.txt
        🗎 gpiointerrupt.cfg
        🗎 makefile.defs

7. The page shown when you select an example in the TI Resource Explorer provides additional links to perform common actions with that example.

8. Use 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.

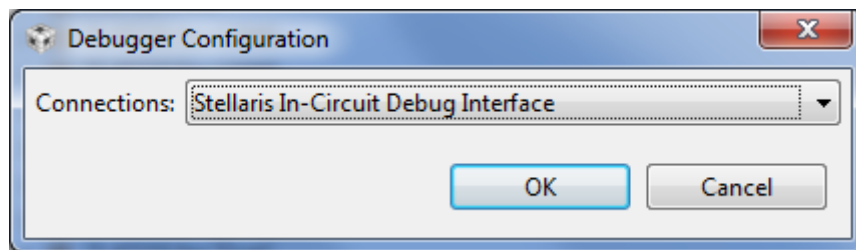Step 2: 🔧 Build the imported project

*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.*

9. Use the **Step 3** link to change the connection used to communicate with the board. The current setting is shown in the TI Resource Explorer page for the selected example. (If you want to use a simulator instead of a hardware connection, double-click the *.ccxml file in the targetConfigs folder of the project to open the Target Configuration File editor. Change the **Connection** as needed, and click **Save**.)

10. You will see the Debugger Configuration dialog. Choose an emulator from the list. For Tiva devices, choose the **Stellaris In-Circuit Debug Interface**.

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

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 Driver Examples share the following features:

- Most TI-RTOS driver 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).

- Driver Examples for a particular target all 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).

3.2 Creating an Empty TI-RTOS Project

TI-RTOS provides blank projects you can use as a starting point for creating your own projects that utilize 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 TI-RTOS driver projects can be created with the TI Resource Explorer (see Section 3.1).

After you create the example, 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

Add to the example as needed to implement your application.

3.3 Creating Examples 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 supported tool chains—TI and 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.

Note:	If you installed TI-RTOS using the standalone installer (Section 2.4 or Section 2.5), this step is not necessary because pre-generated examples are included as part of the installation for all supported boards and tool chains. The provided examples are located in the `TIRTOS_INSTALL_DIR\tirtos_tivac_2_##_##_##_examples` directory. Pre-generated examples are not provided if you installed TI-RTOS through the CCS App Center.

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.

   — DEFAULT_INSTALLATION_DIR: Full path to the location where TI tools are installed.

   — 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.

   — TIRTOS_INSTALLATION_DIR: Full path to the TI-RTOS installation.

   — XDCTOOLS_INSTALLATION_DIR: Full path to the XDCtools installation.

2. If you plan to use TI-RTOS with IAR, set the IAR_BUILD variable in the `tirtos.mak` file to true.

```
IAR_BUILD ?= true
```

3. If you plan to use TI-RTOS with GCC, set the GCC_BUILD variable in the `tirtos.mak` file to true.

```
GCC_BUILD ?= true
```

4. 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_31_##_##_core\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_tivac_2_##_##_##_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 and 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. The `makefile.defs` file in each example is used when cleaning up a previous *.cfg file build within CCS.

3.4   Creating Examples with IAR Embedded Workbench

For information about using TI-RTOS examples with IAR Embedded Workbench, see the wiki page on Creating TI-RTOS Applications in IAR Embedded Workbench on the Texas Instruments wiki.

3.5 Summary of Example Peripheral Use and Target Support

The boards, 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 several 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. TI-RTOS for TivaC Examples: Boards, Components, and Drivers

Example	EK-TM4C123GXL / EK-LM4F120XL	DK-TM4C129X	EK-TM4C1294XL, EK-TM4C129EXL	SYS/BIOS	SD Card / SDSPI	USB MSC Host	USB HID	USB CDC	I²C	GPIO	PWM	SPI	Watchdog Timer	WiFi	UART	NDK / EMAC	Network Services	NDK / PPP	UIA
Empty and Empty (Minimal) ProjectsTI-RTOS	X	X	X	X						X									X
TCP Echo / TCP Echo IPv6		X	X	X						X						X			
TCP Echo for CC3100 [1]	X			X						X		X		X					
TCP Echo with TLS [6]			X	X						X						X			
TCP Echo with PPP [7]		X	X	X				X		X								X	
HTTP Client GET			X	X						X						X	X		
HTTPS Client GET [6]			X	X						X						X	X		
HTTP Client GET for CC3100 [1]			X	X						X		X		X			X		
HTTPS Client GET for CC3100 [1]			X	X						X		X		X			X		
UDP Echo / UDP Echo IPv6		X	X	X						X						X			
UDP Echo for CC3100 [1]	X			X						X		X		X					
SPI Loopback	X	X	X	X						X		X							
FatSD: FatFs File Copy	X^2	X	X^2	X	X					X									
FatSD Raw: FatFs File Copy using FatFs APIs	X^2	X	X^2	X	X					X									
FatSD USB Copy: (SD Card and USB Drive)		X	X^2	X	X	X				X									
GPIO Interrupt	X	X	X	X						X									
I²C RF430CL330 Load [3]	X	X	X	X					X	X									
I²C TMP006 [4]	X	X	X	X					X	X									
I²C TPL0401EVM [5]	X	X	X	X					X	X									
PWM Driver	X	X	X	X						X	X								
PWM LED	X	X	X	X						X	X								
UART Console	X	X	X	X				X		X					X				X

Example	EK-TM4C123GXL / EK-LM4F120XL	DK-TM4C129X	EK-TM4C1294XL, EK-TM4C129EXL	SYS/BIOS	SD Card / SDSPI	USB MSC Host	USB HID	USB CDC	I²C	GPIO	PWM	SPI	Watchdog Timer	WiFi	UART	NDK / EMAC	Network Services	NDK / PPP	UIA
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									
USB Keyboard Host		X	X	X			X			X									
USB Mouse Device	X	X	X	X			X			X									
USB Mouse Host		X	X	X			X			X									
USB Serial Device	X	X	X	X				X		X									
USB CDC Mouse Device	X	X	X	X			X	X		X									
Watchdog	X	X	X	X						X			X						

[1]This example requires either a CC3100 EM board or CC3100 BoosterPack. See the "Jumper Settings: section in Section 3.6 through Section 3.8 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 3.10.4.

[4]This example requires a TMP006 BoosterPack. See Section 3.10.2.

[5]This example requires a TPL0401EVM BoosterPack. See Section 3.10.5.

[6]In order to use TLS with the TI-RTOS NDK stack, you must download, install, and set up the wolfSSL Embedded SSL Library as described in the Using wolfSSL with TI-RTOS wiki page.

[7]This example demonstrates how to make a PPP connection to a Linux PC; Windows PPP connections are not demonstrated by this example.

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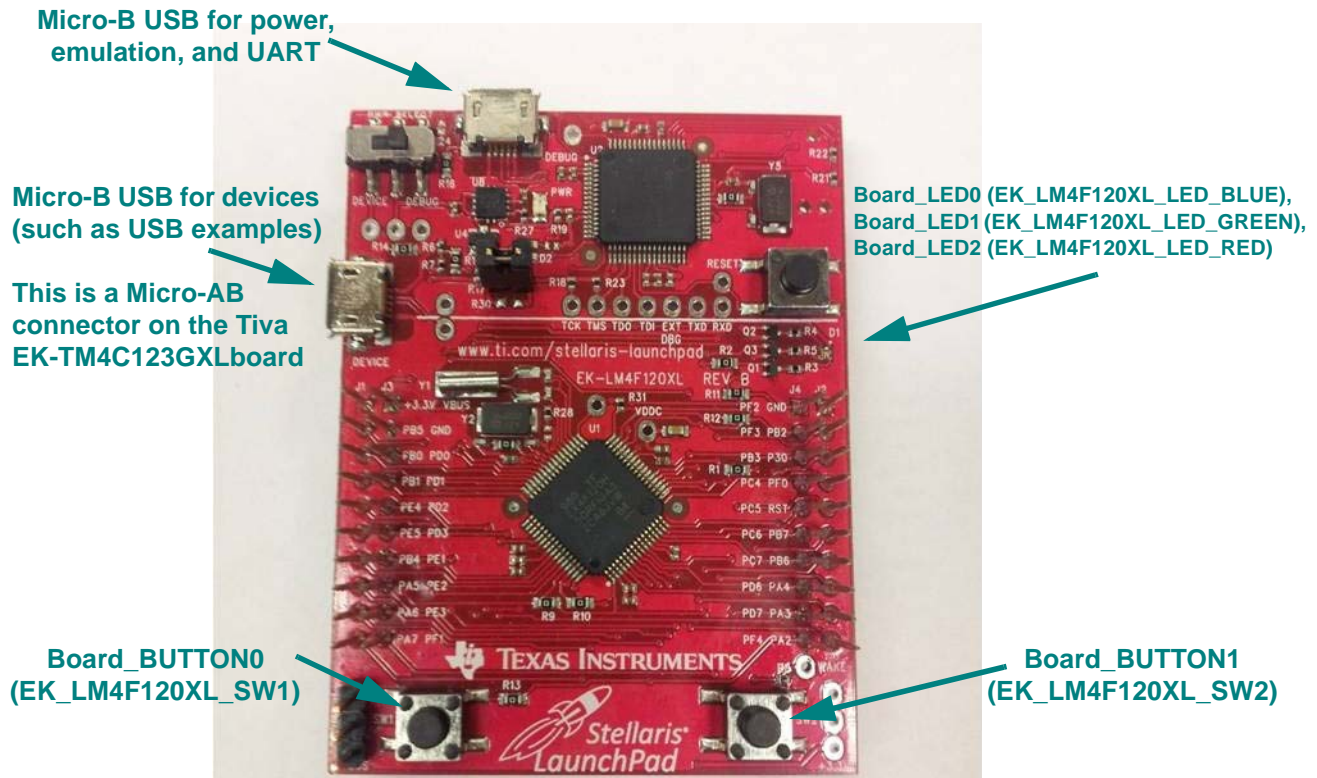
- Actions performed by functions in the example.
- Hardware-specific descriptions of buttons, LEDs, etc…
- Which external components are (or may be) needed to run with particular examples.

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.

3.6 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:

If you use a BoosterPack with this board (Section 3.10), the 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)
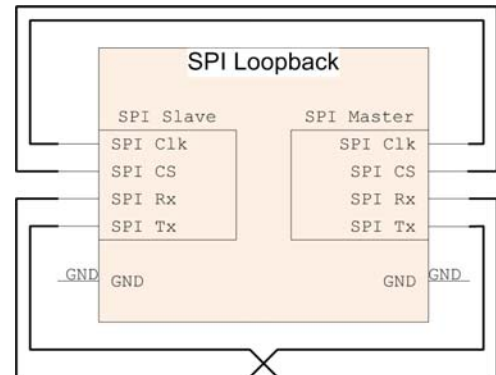
Resources Used:

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

- **TI-RTOS Kernel (SYS/BIOS).** Uses the first general-purpose timer available and that timer's
  associated interrupts. Generally, this will be Timer 0. The TI-RTOS Kernel 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 various BoosterPacks.
  - **PWM.** The PWM driver uses the onboard RGB LED (B:PF2 and G:PF3). While these Pins
    coincide with GPIO driver Pins, they are configured for the PWM driver instead for PWM
    examples. GPIO driver APIs should not be used.
  - **SD Card.** Uses FatFs and the SDSPI driver on SSI2 without interrupts to read and write to files.
  - **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 CC3100 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.

3.7 Tiva DK-TM4C129X Settings and Resources

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

**Tiva DK-TM4C129X Evaluation Kit connections:**
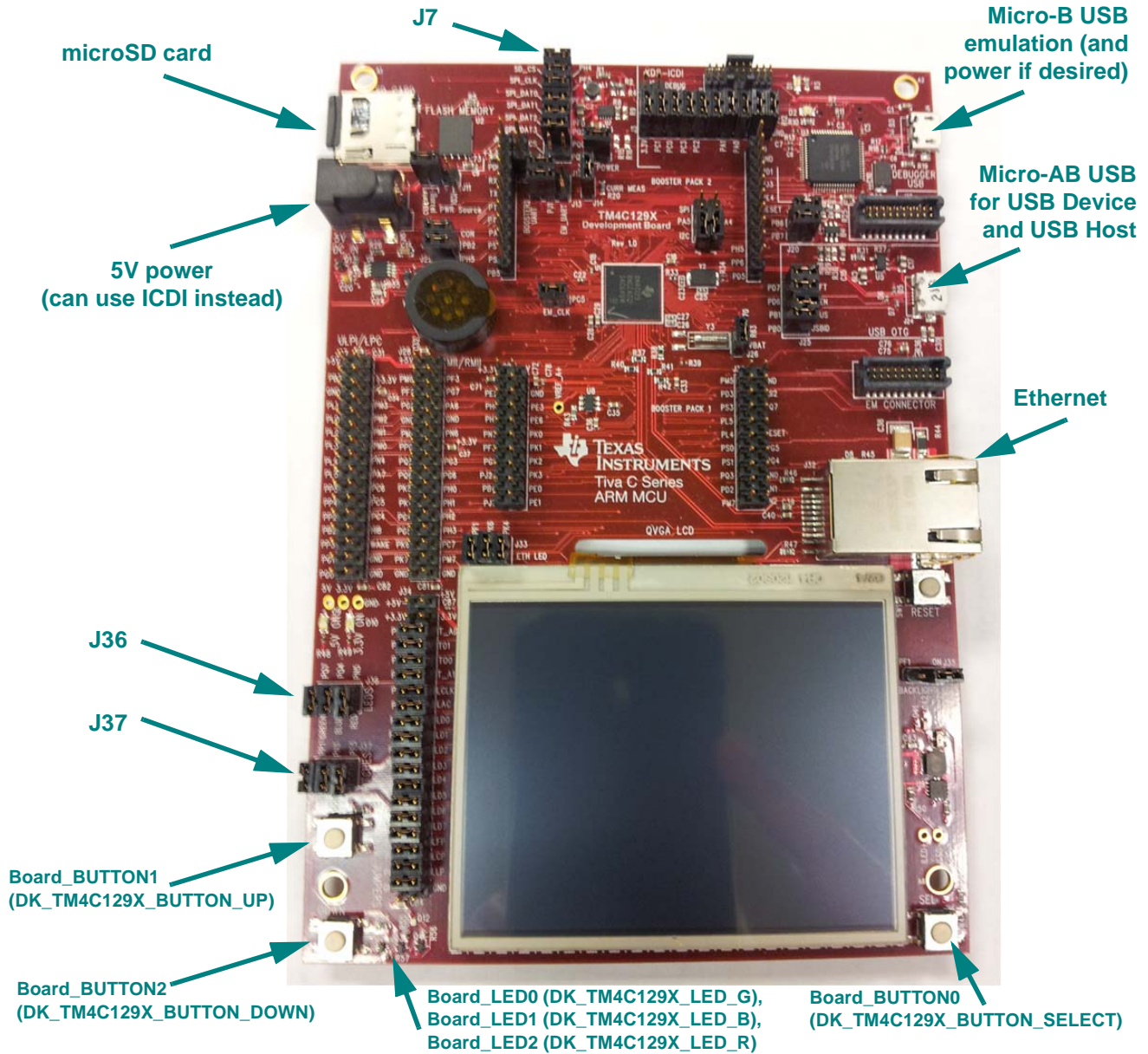
Jumper settings:

- **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))
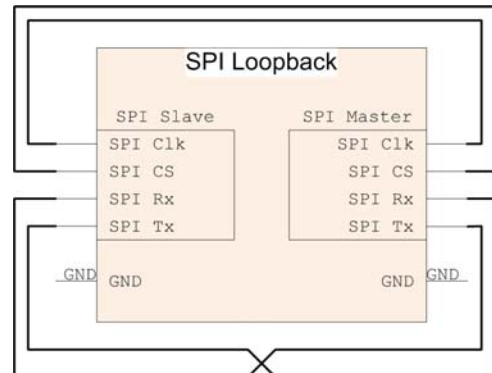
Resources Used:

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

- **TI-RTOS Kernel (SYS/BIOS).** Uses the first general-purpose timer available and that timer's
  associated interrupts. Generally, this will be Timer 0. The TI-RTOS Kernel manages the Interrupt
  Vector Table.

- **TI-RTOS.**
  — **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 various BoosterPacks.
  — **PWM.** The PWM driver uses onboard LEDs D16 (PF1) and D8 (PK4). These LEDs coincide with
    EMAC driver Link and Speed LEDs respectively. However for the PWM examples they are
    configured for the PWM driver instead.
  — **SD Card.** Uses FatFs and the SDSPI driver on QSSI3 without interrupts to read and write to files.
  — **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.

3.8   Tiva EK-TM4C1294XL Settings and Resources

The EK-TM4C1294XL Evaluation Kit contains a Tiva TM4C1294NCPDT device.

Tiva EK-TM4C1294XL Evaluation Kit connections:
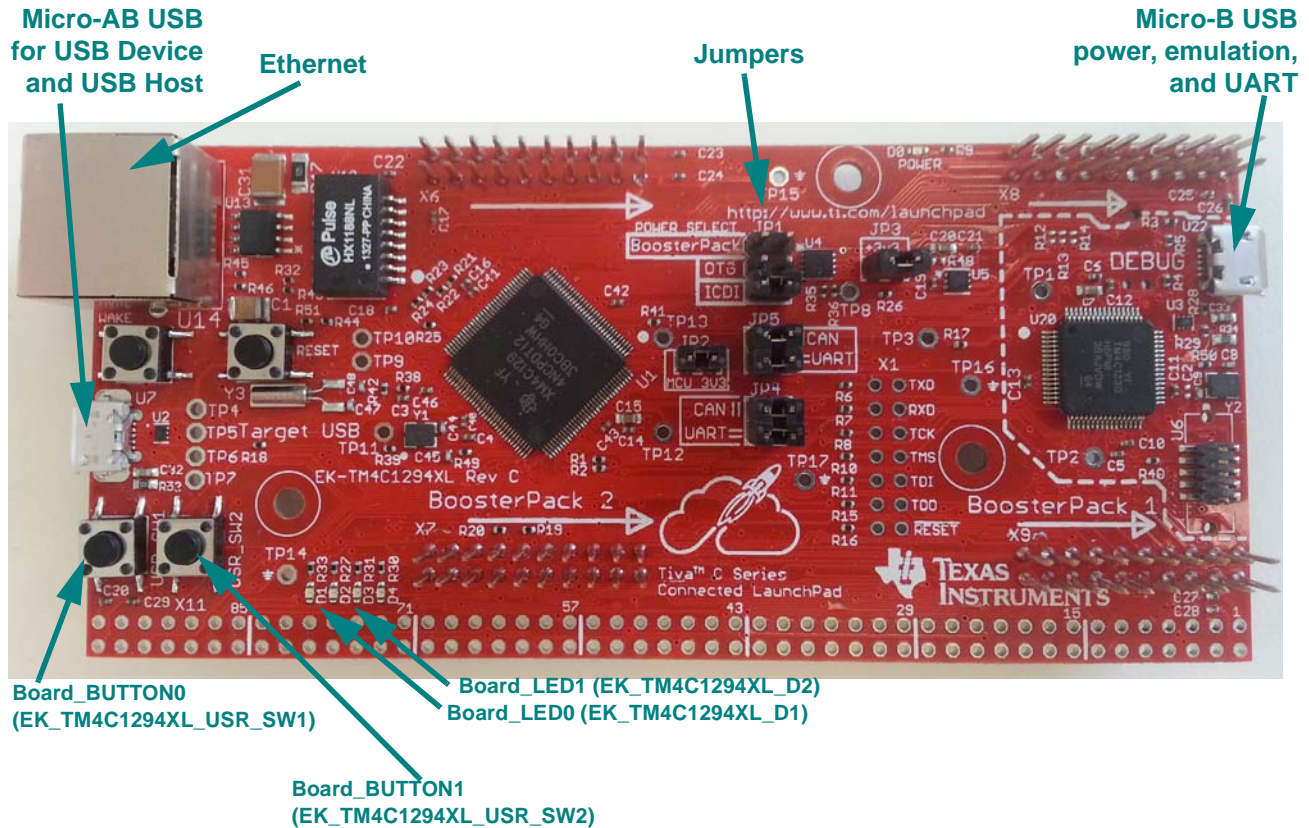
BoosterPack examples:

BoosterPack examples are configured to use the pins for the BoosterPack1 headers.

Jumper settings:

- **JP1:** power selection
- **JP2:** 3.3V power to MCU
- **JP3:** Power from USB Debug interface.
- **JP4:** UART position (Connect UART RX to ICDI)
- **JP5:** UART position (Connect UART TX to ICDI)

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)
PD3 (SPI0CLK) on BP1* ——▶	PQ0 (SPI1CLK) on BP2
PD2 (SPI0FSS) on BP2* ——▶	PQ1 (SPI1FSS) on BP2
PD0 (SPI0RX) on BP1 ——▶	PQ2 (SPI1TX) on BP2
PD1 (SPI0TX) on BP1 ——▶	PQ3 (SPI1RX) on BP2

* BP1 refers to BoosterPack1 headers. BP2 refers to BoosterPack2 headers.
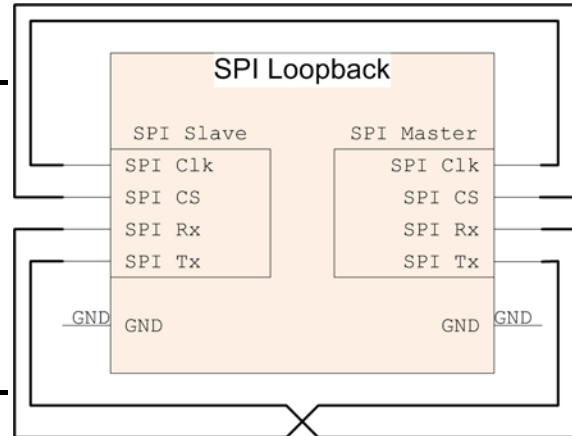
Resources Used:

The following list shows which EK-TM4C1294XLperipheral resources are used by TI-RTOS applications on that platform. TI-RTOS examples control which peripherals (and which ports) are used

- **TI-RTOS Kernel (SYS/BIOS).** Uses the first general-purpose timer available and that timer's associated interrupts. Generally, this will be Timer 0. The TI-RTOS Kernel manages the Interrupt Vector Table.

- **TI-RTOS.** All examples use peripherals on BoosterPack1 headers by default.

  — **GPIOs.** The GPIO driver uses 2 pins as outputs for the onboard LEDs: PN0 (LED0) and PN1 (LED1). Additionally, 2 pins are used as inputs for onboard switches: PJ0 (SW1) and PJ1 (SW2).

  — **I2C.** The I^{2}C driver is configured on I2C7 to support various BoosterPacks.

  — **PWM.** The PWM driver uses onboard LED D4 (PF0). This LED coincides with EMAC driver Link LED. However for PWM examples it is configured for the PWM driver instead.

  — **SD Card.** Uses FatFs and the SDSPI driver on SSI2 without interrupts to read and write to files.

  — **Serial.** The UART driver uses UART0, which is attached to the onboard ICDI emulator chip to facilitate serial communications.

  — **SPI.** The SPI driver is configured to use SSI2 for Board SPI0 and SSI3 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.

3.9   Tiva EK-TM4C129EXL Settings and Resources

The EK-TM4C129EXL Evaluation Kit contains a Tiva TM4C129ENCPDT device, which supports crypto hardware accelerators.

The board layout for EK-TM4C129EXL is same as for the EK-TM4C1294XL. The same settings and resources apply to both boards. See Section 3.8 for details.
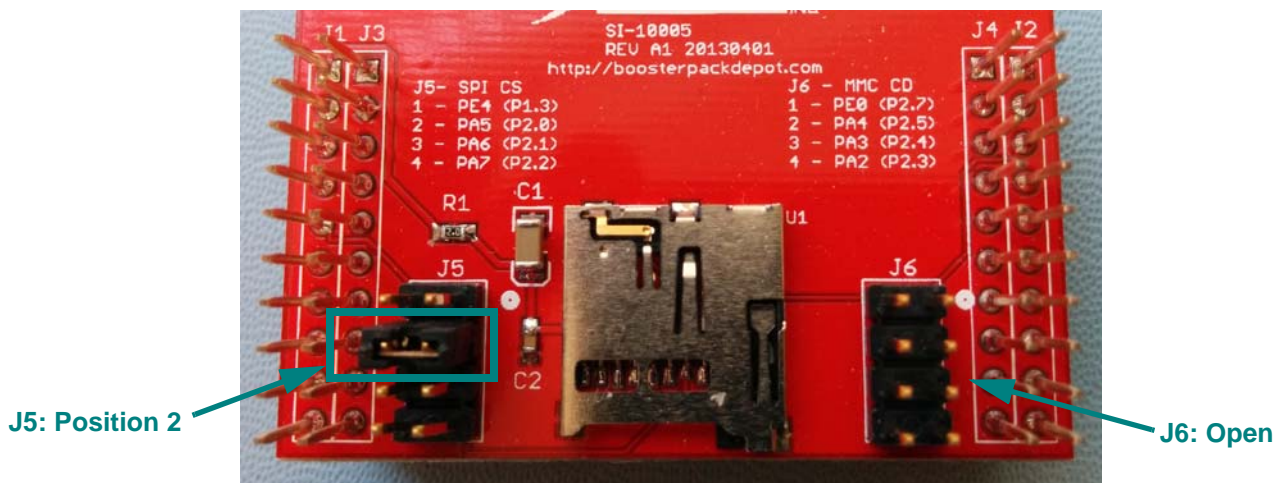
3.10   BoosterPacks

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

3.10.1   SD Card BoosterPack

A microSD BoosterPack or 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, EK-TM4C1294XL, and EK-TM4C129EXL.

The figure below shows the jumper needed on the microSD BoosterPack to make it pin-compatible with the SD Card BoosterPack.

3.10.2   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, and DK-TM4C129X.

3.10.3   *CC3100 BoosterPacks*

The TI CC3100 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.

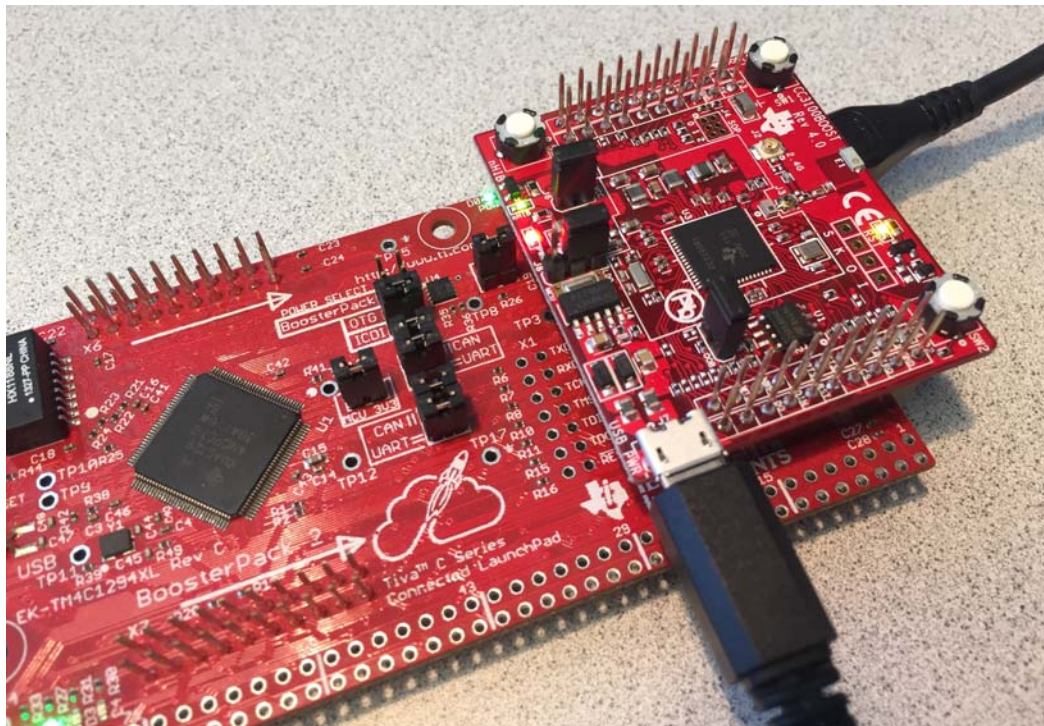The CC3100 BoosterPack is available to be used with the following boards (pictures are provided to show the correct way to mount the booster pack onto each board):

- EK-TM4C123GXL
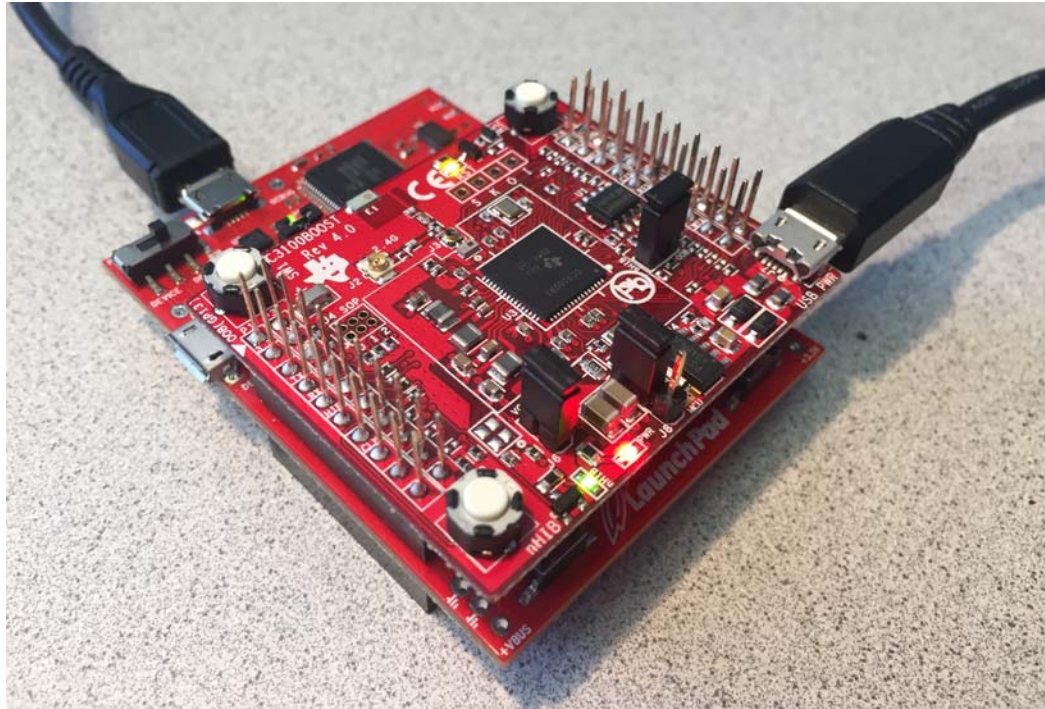
- DK-TM4C129X
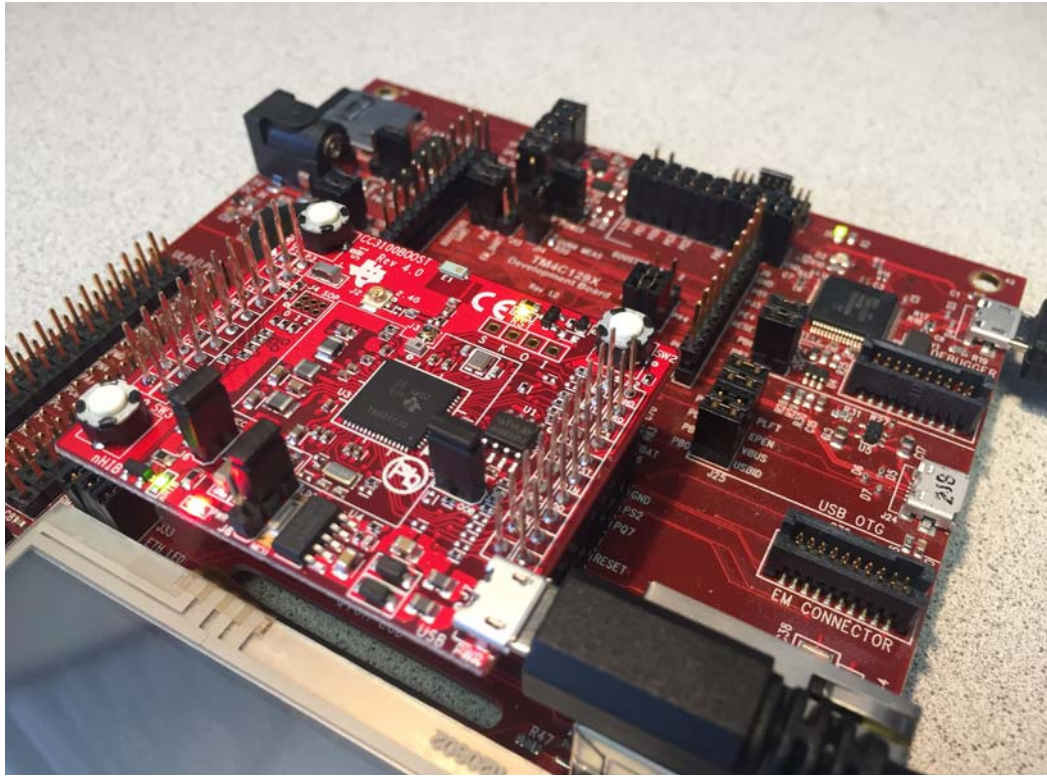
- EK-TM4C1294XL and EK-TM4C129EXL

3.10.4 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, and DK-TM4C129X.

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.
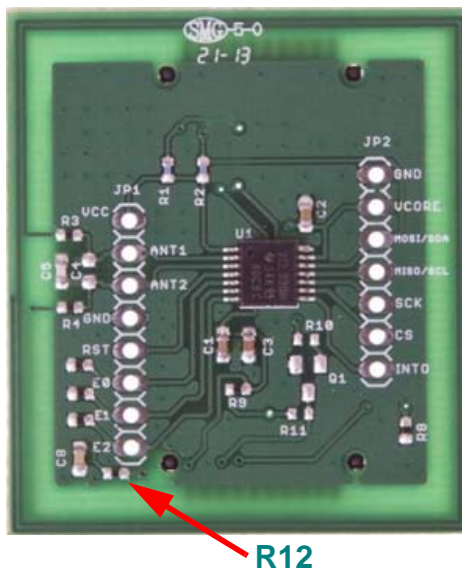
3.10.5 TPL0401EVM Board

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

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.
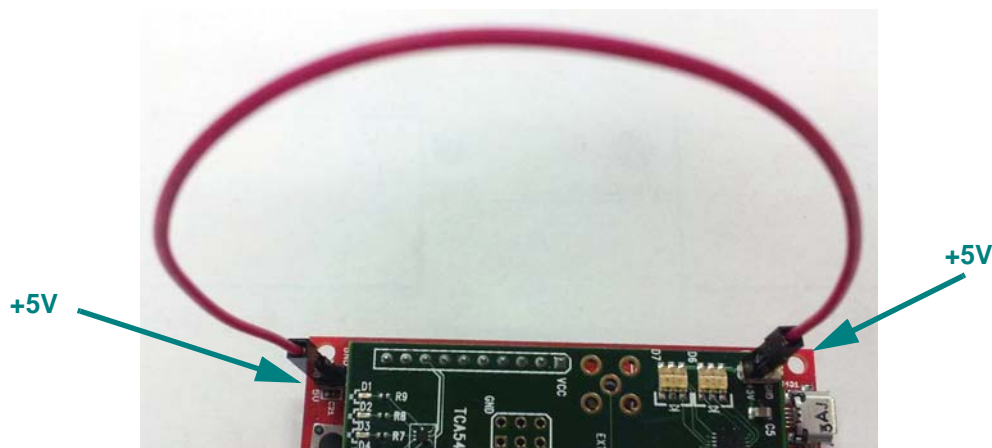
## 3.11 Installing USB Drivers for the USB Device Examples

The USB examples build upon the examples provided with TivaWare. Because the examples mimic the same functionality, you can use the same drivers delivered with standalone installations of TivaWare.

Windows USB drivers for Tiva are located in the `<tirtos_install>/products/TivaWare_C_Series-2.#/windows_drivers` directory.

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**.
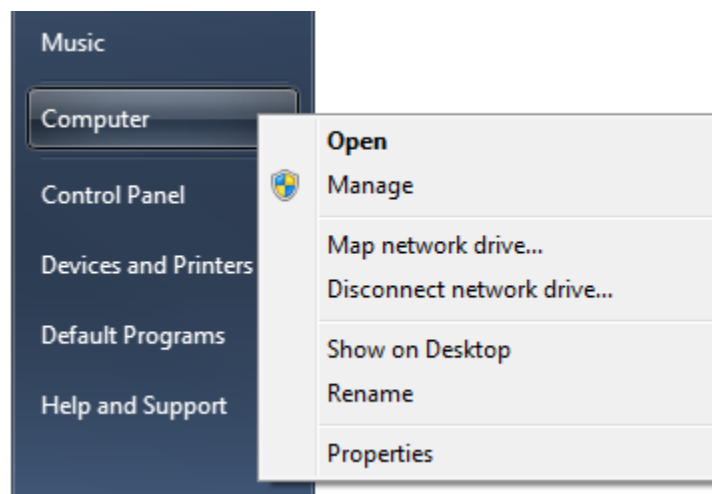
4. If you are prompted by a security warning in Windows 7, click **Yes**.

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.
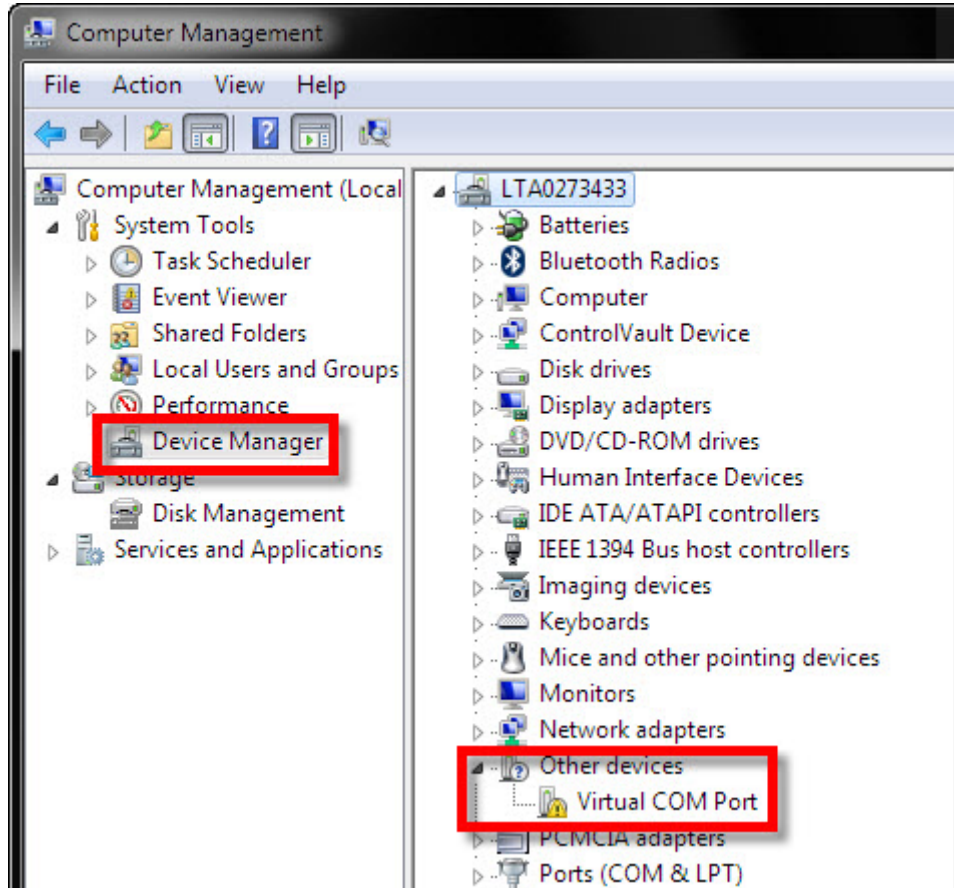
7. Right-click on the device, and select **Update Driver Software**.

8. Select **Browse my computer for driver software** and browse to the location of the Windows USB drivers `<tirtos_install>/products/TivaWare_C_Series-2.#/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.

Configuring TI-RTOS

This chapter describes how to configure TI-RTOS and its components for use by your application.

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.

This section shows how to open XGCONF and view the System Overview. For details about how to use XGCONF, see Chapter 2 of the SYS/BIOS User's Guide (SPRUEX3).

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 if you are using a Driver example. (If this is the configuration file for a Kernel example or an Instrumentation example, the Welcome sheet for SYS/BIOS opens first, instead.) The Welcome sheet provides links to documentation resources.

4. Click the **System Overview** link to see a diagram of the components available through TI-RTOS. (SYS/BIOS modules are shown if you are using a Kernel example or an Instrumentation example.) A green check mark indicates which modules are being used by the application.
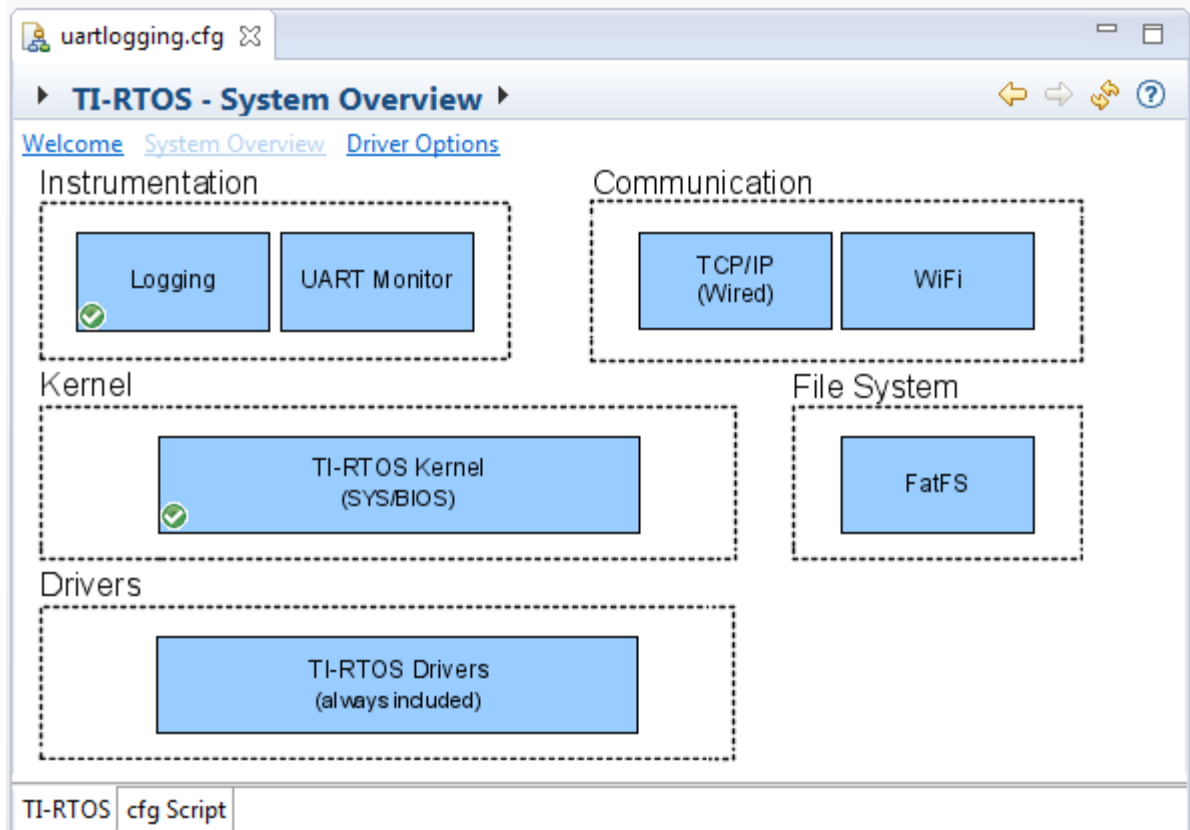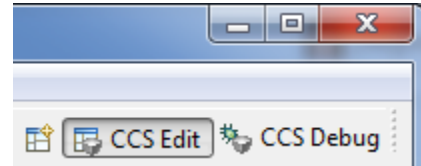
5.   You also see a list of Available Products in a pane on the left of the CCS window. This list allows you to select the TIRTOS module and any configurable modules in the products TI-RTOS provides.

6.   Click a blue box in the System Overview to go to the configuration page for a module.

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.
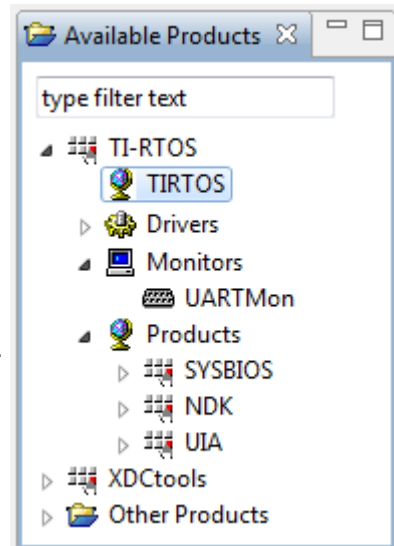
4.2   Configuring TI-RTOS Drivers

In the System Overview display for the TIRTOS module, select the **Driver Options** link.
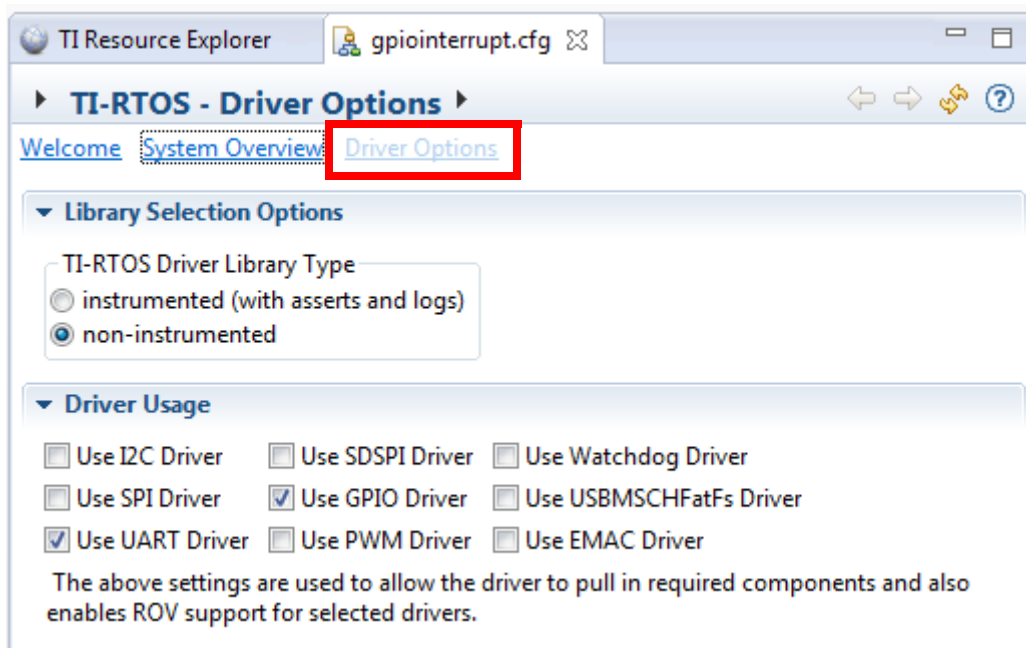
You can choose to use either the instrumented or non-instrumented driver libraries when linking with TI-RTOS. The instrumented libraries process Log events while the non-instrumented libraries do not. See the section on "Using Instrumented or Non-Instrumented Libraries" in the *TI-RTOS User's Guide* (SPRUHD4) for more information. This setting affects all the TI-RTOS drivers listed in Section 1.6 together.

Check the boxes for any drivers your application will use. For most drivers, this is the only configuration necessary. Enabling the drivers here also enables the instrumentation provided to the RTOS Object View (ROV) plugins.

The WiFi driver is individually configurable. See the section on Static Configuration of the WiFi Driver in the *TI-RTOS User's Guide* (SPRUHD4) for details.

4.2.1 Configuring System Support

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 Components of TI-RTOS

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 details about 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 TI Resource Explorer (in CCS) lets you access documentation for all the components of TI-RTOS. Choose the **Documentation Links** item for your version of TI-RTOS. Use the configuration reference documentation, and click the "Configuration settings" link at the top of each page to skip to the configuration information.

Index

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as "components") are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI's terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers' products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers' products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI's goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or "enhanced plastic" are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have not been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

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		