

# TI-RTOS 1.01

## Getting Started Guide



Literature Number: SPRUHD3D  
January 2013

---



---

<b>Preface</b> .....	<b>3</b>
<b>1 About TI-RTOS</b> .....	<b>4</b>
1.1 What is TI-RTOS? .....	4
1.2 What are the TI-RTOS Components? .....	5
1.3 For What Boards and Devices Does TI-RTOS Provide Examples? .....	6
1.4 What Drivers Does TI-RTOS Include? .....	6
1.5 Hardware Resources Used by TI-RTOS Components .....	7
1.5.1 TMDXDOCKH52C1 Resources Used .....	7
1.5.2 TMDXDOCK28M36 Resources Used .....	7
1.5.3 DK-LM3S9D96 Resources Used .....	8
1.5.4 EK-LM4F120XL Resources Used .....	8
1.5.5 EKS-LM4F232 Resources Used .....	9
1.6 For More Information .....	9
<b>2 Installing TI-RTOS</b> .....	<b>12</b>
2.1 System Requirements .....	13
2.2 Installing Code Composer Studio .....	13
2.3 Installing TI-RTOS .....	14
2.3.1 Installing on Windows .....	14
2.3.2 Installing on Linux .....	14
<b>3 Examples for TI-RTOS</b> .....	<b>15</b>
3.1 Creating an Example Project Using the TI Resource Explorer .....	16
3.2 Example Overview .....	18
3.3 Example Settings .....	20
3.3.1 TMDXDOCKH52C1 Development Board .....	20
3.3.2 TMDXDOCK28M36 Development Board .....	21
3.3.3 DK-LM3S9D96 Board .....	22
3.3.4 EK-LM4F120XL Board .....	22
3.3.5 EKS-LM4F232 Board .....	22
3.4 Installing USB Drivers for the USB Device Examples .....	24
3.5 Using Examples for Individual Components .....	26
<b>4 Configuring TI-RTOS</b> .....	<b>27</b>
4.1 Starting the Configuration Tool .....	28
4.2 Configuring TI-RTOS .....	29
4.3 Configuring Individual Sub-Components .....	30
<b>Index</b> .....	<b>31</b>

## Read This First

---

---

---

### About This Manual

This manual describes TI-RTOS. The version number as of the publication of this manual is v1.01.

### 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, 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.

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

January 25, 2013

# About TI-RTOS

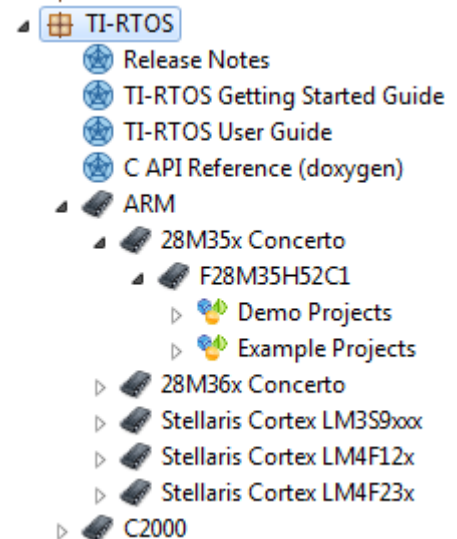
This chapter provides an overview of TI-RTOS.

Topic	Page
1.1 What is TI-RTOS? .....	4
1.2 What are the TI-RTOS Components? .....	5
1.3 For What Boards and Devices Does TI-RTOS Provide Examples? ..	6
1.4 What Drivers Does TI-RTOS Include? .....	6
1.5 Hardware Resources Used by TI-RTOS Components .....	7
1.6 For More Information .....	9

## 1.1 What is TI-RTOS?

TI-RTOS makes it easier to develop applications for TI microcontrollers. This product contains several software components and examples that use these components together.

TI-RTOS is a one-stop solution for developing applications for TI embedded processors. 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.

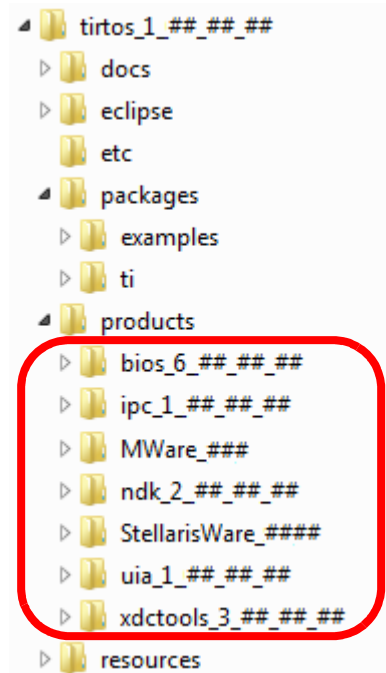


## 1.2 What are the TI-RTOS Components?

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.

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.
- **MWare.** The M3 portion of ControlSuite. It includes low level drivers and examples.<sup>1</sup>
- **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.
- **StellarisWare.** This software is an extensive suite of software designed to simplify and speed development of Stellaris-based microcontroller applications. This component is rebuilt to include only the portions required by TI-RTOS.<sup>1</sup>
- **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.



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. The MWare and StellarisWare 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 StellarisWare components of TI-RTOS for details.

### 1.3 For What Boards and Devices Does TI-RTOS Provide Examples?

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

*Table 1-1 Boards supported by TI-RTOS*

Board	Device on Board
TMDXDOCKH52C1	F28M35H52C1
TMDXDOCK28M36	F28M36P63C2
DK-LM3S9D96	LM3S9D96
EK-LM4F120XL	LM4F120H5QR
EKS-LM4F232	LM4F232H5QD

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 Embedded Processors Wiki](#).

### 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 and StellarisWare.

- **EMAC.** Ethernet driver used by the networking stack (NDK) and not intended to be called directly.
- **SDSPI.** Driver for SD cards using an SPI (SSI) bus. This driver is used by the FatFS and not intended to be interfaced directly.
- **I<sup>2</sup>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 (and therefore the LEDs).
- **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.
- **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.

## 1.5 Hardware Resources Used by TI-RTOS Components

This section briefly describes which hardware resources 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.

### 1.5.1 TMDXDOCKH52C1 Resources Used

The following list shows which TMDXDOCKH52C1 peripheral 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.
- **IPC.** Uses the IPC registers including their associated interrupts.
- **TI-RTOS.**
  - **Ethernet.** Uses the EMAC driver and its associated interrupts with the 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<sup>2</sup>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.
  - **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.

### 1.5.2 TMDXDOCK28M36 Resources Used

The following list shows which TMDXDOCK28M36 peripheral 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.
- **IPC.** Uses the IPC registers including their associated interrupts.
- **TI-RTOS.**
  - **Ethernet.** Uses the EMAC driver and its associated interrupts with the NDK to support networking.
  - **SD Card.** Uses FatFs and the SDSPI driver on SSI3 without interrupts to read and write to files on an SD Card.
  - **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 to facilitate serial communications.
  - **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.

### 1.5.3 DK-LM3S9D96 Resources Used

The following list shows which DK-LM3S9D96 peripheral 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.**
  - **Ethernet.** Uses the EMAC driver and its associated interrupts with the 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<sup>2</sup>C driver on I2C0 with its associated interrupts to read and write to the onboard EEPROM.
  - **GPIOs.** The GPIO driver is used on the onboard LED1 (PF3) and SW1 (PJ7).
  - **Serial.** The UART driver uses UART0 which is attached to the FTDI USB chip to facilitate serial communications.
  - **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.

### 1.5.4 EK-LM4F120XL Resources Used

The following list shows which EK-LM4F120XL peripheral 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.**
  - **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).
  - **Serial.** The UART driver uses UART0 which is attached to the FTDI USB chip to facilitate serial communications.
  - **USB.** The USB reference examples use the USB library and the USB controller with its associated interrupts. This device only supports USB in device mode.
  - **Watchdog.** The Watchdog driver example uses Watchdog Timer 0 and its associated interrupt.



### 1.5.5 EKS-LM4F232 Resources Used

The following list shows which EKS-LM4F232 peripheral 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 SSI0 without interrupts to read and write to files on an SD Card.
  - **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.
  - **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.

## 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 \(SPRUHD4\)](#)
  - [SYS/BIOS on TI Embedded Processors Wiki](#)
  - [BIOS forum on TI's E2E Community](#)
  - [TI-RTOS Porting Guide](#)
- **Code Composer Studio (CCS)**
  - [CCS online help](#)
  - [CCSv5 on TI Embedded Processors Wiki](#)
  - [Code Composer forum on TI's E2E Community](#)
- **SYS/BIOS**
  - [SYS/BIOS 6 Getting Started Guide. <sysbios\\_install>/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 <sysbios\_install>/docs/cdoc/index.html.
  - [SYS/BIOS on TI Embedded Processors Wiki](#)
  - [BIOS forum on TI's E2E Community](#)
  - [SYS/BIOS 6.x Product Folder](#)
  - [Embedded Software Download Page](#)

- **XDCtools**
  - XDCtools online reference. Open from CCS help or run <xdc\_install>/docs/xdctools.chm.
  - [RTSC-Pedia Wiki](#)
  - [BIOS forum on TI's E2E Community](#)
  - [Embedded Software Download Page](#)
- **IPC**
  - [IPC User's Guide \(SPRUG06\)](#)
  - IPC online API reference. Run <ipc\_install>/docs/doxygen/index.html.
  - IPC online configuration reference. Open from CCS help or run <ipc\_install>/docs/cdoc/index.html.
  - [Embedded Software Download Page](#)
- **NDK**
  - [NDK User's Guide \(SPRU523\)](#)
  - [NDK Programmer's Reference Guide \(SPRU524\)](#)
  - [NDK on TI Embedded Processors Wiki](#)
  - [BIOS forum on TI's E2E Community](#)
  - [Embedded Software Download Page](#)
- **UIA**
  - [System Analyzer User's Guide \(SPRUH43\)](#)
  - UIA online reference. Open from CCS help or run <uia\_install>/docs/cdoc/index.html.
  - [System Analyzer on TI Embedded Processors Wiki](#)
  - [Embedded Software Download Page](#)
- **MWare and ControlSuite**
  - Documents in <tirtos\_install>/products/MWare\_###/docs
  - [ControlSuite on TI Embedded Processors Wiki](#)
  - [ControlSuite Product Folder](#)
- **StellarisWare**
  - Documents in <tirtos\_install>/products/StellarisWare\_####/docs
  - [StellarisWare 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 <sysbios\_install>/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 TI Embedded Processors Wiki](#)
  - [Concerto on TI Embedded Processors Wiki](#)
  - [Concerto Product Folder](#)
  - [H52C1 Concerto Experimenter Kit](#)
  - [F28M35H52C Concerto Microcontroller datasheets](#)
  - [H63C2 Concerto Experimenter Kit](#)
  - [F28M36P63C2 Concerto Microcontroller datasheets](#)
- **Stellaris boards and devices**
  - [DK-LM3S9D96 Development Kit](#)
  - [LM3S9D96 Stellaris Microcontroller datasheets](#)
  - [Stellaris LM4F120 LaunchPad Evaluation Kit](#)
  - [LM4F120H5QR Stellaris Microcontroller datasheets](#)
  - [EKS-LM4F232 Evaluation Kit](#)
  - [LM4F232H5QD Stellaris Microcontroller datasheets](#)
- **SD Cards**
  - [Specification](#)
- **I<sup>2</sup>C**
  - [Specification](#)

# Installing TI-RTOS

---

---

---

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

Topic	Page
2.1 System Requirements . . . . .	13
2.2 Installing Code Composer Studio . . . . .	13
2.3 Installing TI-RTOS . . . . .	14

## 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 5.3 or higher.

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 5.3, 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 a version of SYS/BIOS that may be newer than the version installed by CCS. For TI-RTOS to work properly, you should use the SYS/BIOS version 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.

### 2.3.1 *Installing on Windows*

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 5.3 is installed. By default, this is `c:\ti`. This is the recommended location for installing TI-RTOS.
4. Start CCS 5.3 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 *Installing on Linux*

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_setulinux_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.)
5. Start CCS 5.3 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.

## Examples for TI-RTOS

---

---

---

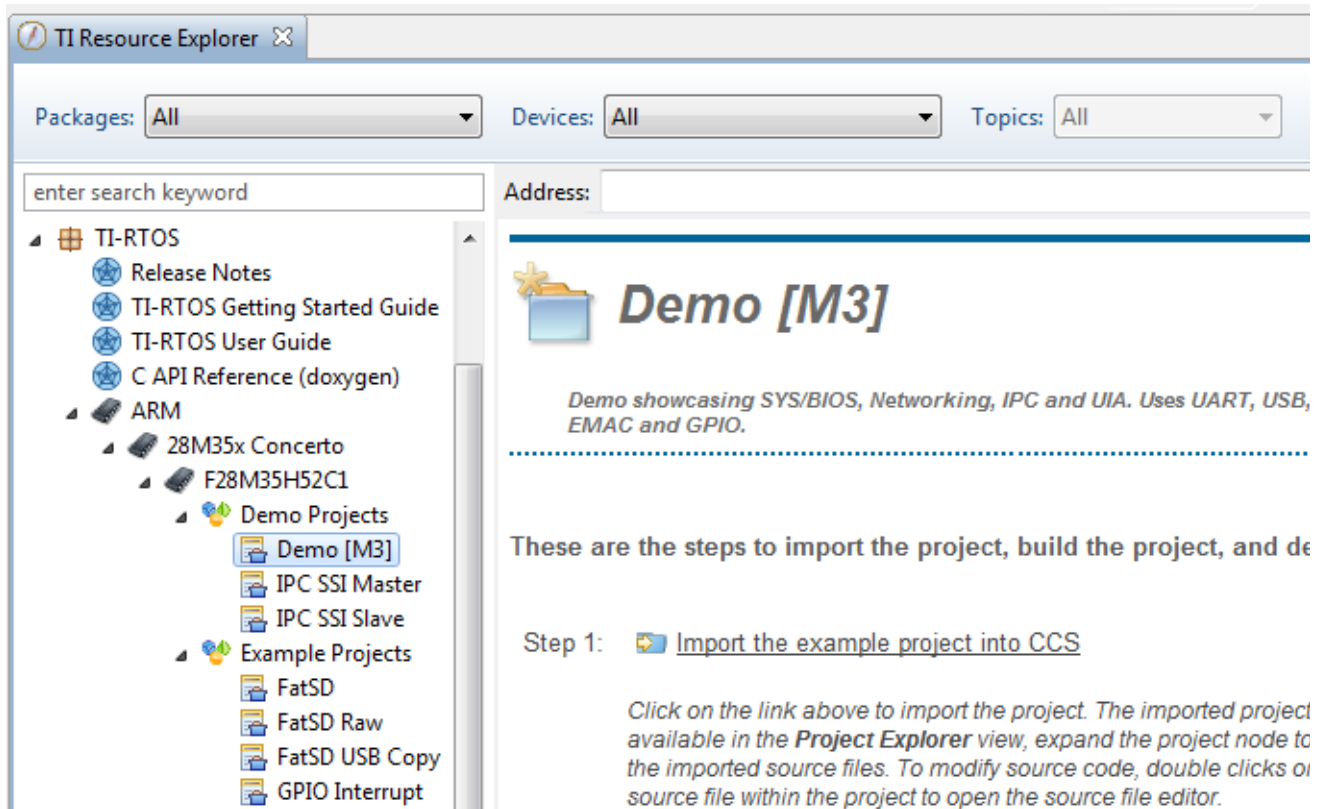
TI-RTOS comes with a number of examples that illustrate on how to use the individual components. This chapter explains how to create and use these examples.

Topic	Page
3.1 Creating an Example Project Using the TI Resource Explorer . . . . .	16
3.2 Example Overview . . . . .	18
3.3 Example Settings . . . . .	20
3.4 Installing USB Drivers for the USB Device Examples . . . . .	24
3.5 Using Examples for Individual Components . . . . .	26



### 3.1 Creating an Example Project Using the TI Resource Explorer

TI-RTOS uses TI Resource Explorer within CCS to let you quickly create example projects and open documentation. 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 closed, choose **View > TI Resource Explorer** to open it.



2. Expand the TI-RTOS item in the tree to show the **Demo Projects** and **Example Projects** for your platform.
3. Select an example to create. See Section 3.2, *Example Overview* and the *TI-RTOS User's Guide* (SPRUHD4) 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.





- 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](#) 

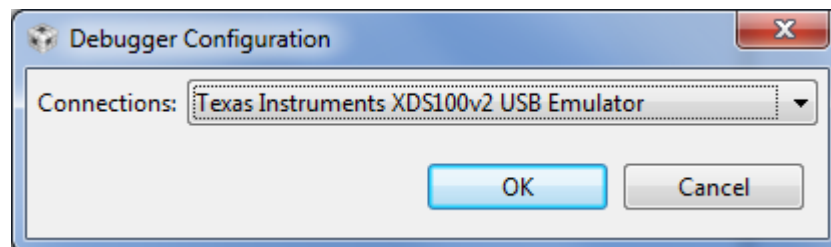
*To change build options, right click on the project and select **Properties** from the context menu. To build the project, select the link above, or select the **Build** toolbar button, or select the **Project | Build Project** menu item.*

- 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.*

- 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 Stellaris devices, choose the **Stellaris In-Circuit Debug Interface**.



**Note:** If you want to use a simulator instead of a hardware connection, select any emulator in the Debugger Configuration dialog and click **OK**. Then choose **View > Target Configurations**. Expand the **Projects** list and double-click on the \*.ccxml file for your example project to open the target configuration editor. Select **Texas Instruments Simulator** in the Connection field, and the simulator for your device in the Device list. Then click **Save**.

- Click the **Step 4** link to launch a debug session for the project and switch to the CCS Debug Perspective. (If you are running a multi-core example, see the "Examples for TI-RTOS" chapter in the *TI-RTOS User's Guide* (SPRUHD4) for instructions for running the example.)

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.*

## 3.2 Example Overview

The components and hardware used by the TI-RTOS examples are shown in the following table. Details about these examples are provided in the *TI-RTOS User's Guide* (SPRUHD4).

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

Example	SYS/BIOS	FatFS		USB Classes		I <sup>2</sup> C	UART	NDK / EMAC	UIA	IPC	GPIO (& LED)	Watchdog Timer
		SD Card / SDSPI	USB MSC Host	HID	CDC							
Empty TI-RTOS Project	X								X		X	
Demo [M3] / Demo [C28]	X	X			X	X	X	X	X	X	X	
Graphic Library Demo	X	X			X				X		X	
TCP Echo	X							X	X		X	
FatSD: FatFs File Copy	X	X							X		X	
FatSD Raw: FatFs File Copy using FatFs APIs	X	X							X		X	
FatSD USB Copy: (SD Card and USB Drive)	X	X	X						X		X	
GPIO Interrupt	X								X		X	
I <sup>2</sup> C EEPROM	X					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		X	
USB Keyboard Host	X			X					X		X	
USB Mouse Device	X			X					X		X	
USB Mouse Host	X			X					X		X	
USB Serial Device	X				X				X		X	
USB CDC Mouse Device	X			X	X				X		X	
Watchdog	X								X		X	X

\* UART is used by SysFlex for sending System\_printf() and printf() output to a console. Other examples use SysMin.

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

**Table 3-2. Example Availability by Board**

Example	TMDXDOCKH52C1	TMDXDOCK28M36	DK-LM3S9D96	EK-LM4F120XL	EKS-LM4F232
Empty TI-RTOS Project	X	X	X	X	X
Demo [M3] / Demo [C28]	X	X			
Graphic Library Demo					X
TCP Echo	X	X	X		
FatSD: FatFs File Copy	X	X	X		X
FatSD Raw: FatFs File Copy using FatFs APIs	X	X	X		X
FatSD USB Copy: (SD Card and USB Drive)	X	X	X		X
GPIO Interrupt	X	X	X	X	X
I <sup>2</sup> C EEPROM *	X		X		
UART Console	X	X	X	X	X
UART Echo	X	X	X	X	X
UART Logging	X	X	X	X	X
USB Keyboard Device	X	X	X	X	X
USB Keyboard Host	X	X	X		X
USB Mouse Device	X	X	X	X	X
USB Mouse Host	X	X	X		X
USB Serial Device	X	X	X	X	X
USB CDC Mouse Device	X	X	X		X
Watchdog	X	X	X	X	X

\* I2C communications with the onboard EEPROM for DK-LM3S9D96 requires an additional daughterboard.

### 3.3 Example Settings

There is a separate `<example_name>_readme.txt` 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.txt` 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 subsections that follow list settings required to run the TI-RTOS examples on the supported boards.

#### 3.3.1 TMDXDOCKH52C1 Development Board

##### 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 3.3.1.2)

##### 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.

##### 3.3.1.1 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:

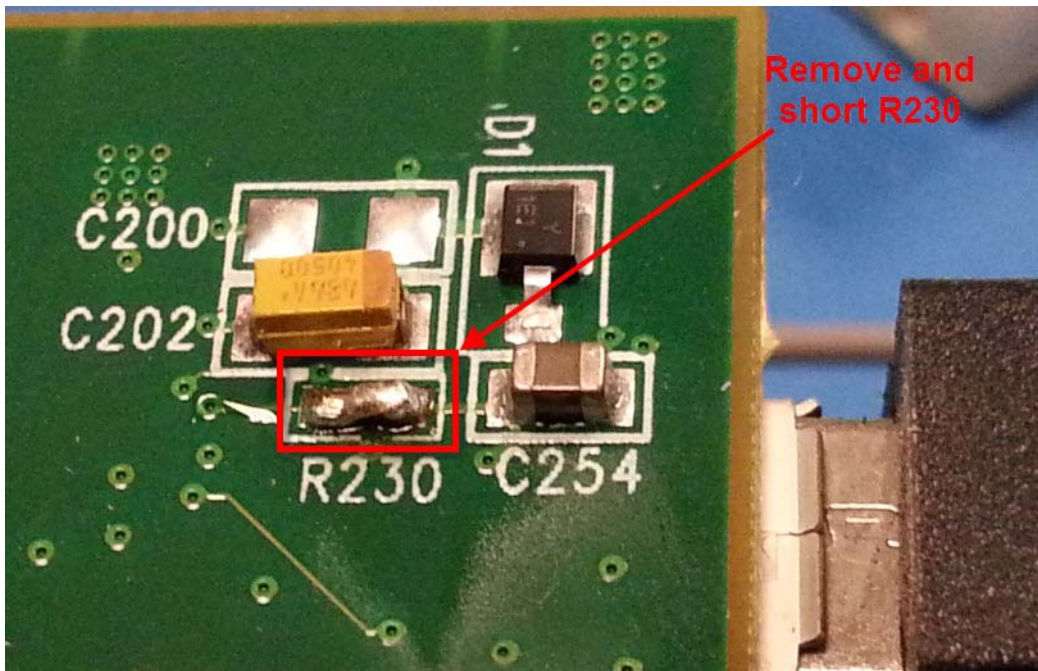
```

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

```

### 3.3.1.2 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.

### 3.3.2 TMDXDOCK28M36 Development Board

#### Jumper settings:

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

#### Switch settings:

- A:SW1: 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.

#### 3.3.2.1 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 3.3.1.1 for details.

### 3.3.3 **DK-LM3S9D96 Board**

**Jumper settings:**

- JP01-JP02: Closed (Ethernet jack LEDs)
- JP05-JP06: Closed (User LED and User Switch)
- JP09-JP12: Closed (SPI/SSI SD Card slot)
- JP03, JP37-JP38: Closed (VBUS, USB0EPE, and USB0PFLT)

**Switch settings:**

- SW1: Some examples use PJ7 as an input.

### 3.3.4 **EK-LM4F120XL Board**

**Jumper settings:** N/A

**Switch settings:**

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

### 3.3.5 **EKS-LM4F232 Board**

**Jumper settings:** N/A

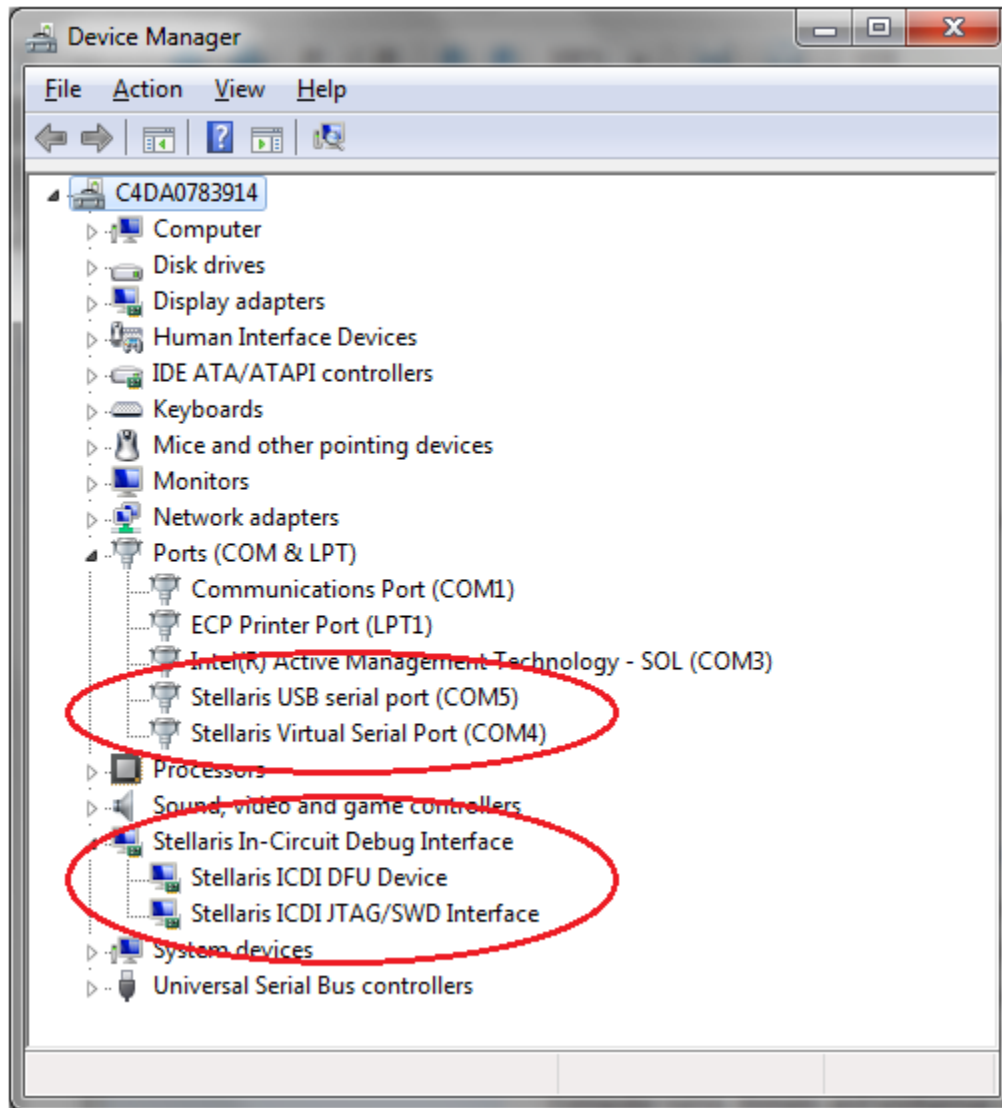
**Switch settings:**

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

**USB connection and ports:**

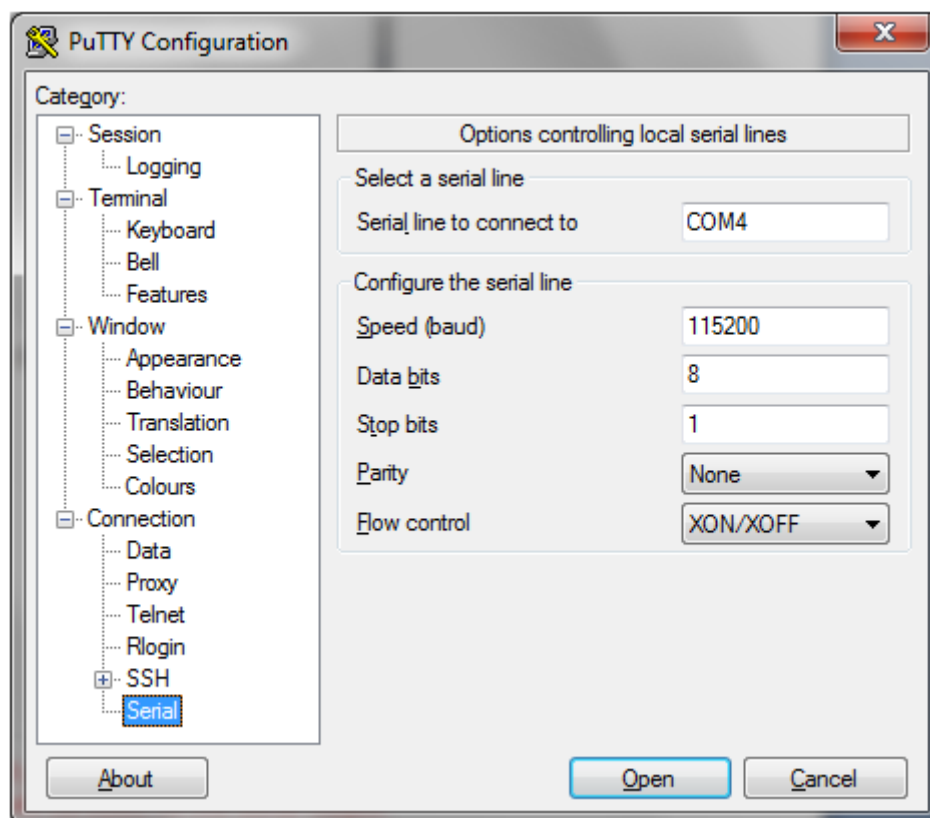
Connect the power/CDI USB connector on the board to a USB port on your host PC. This connection is used for both emulation and the serial output. If you would also like to see instrumentation (UIA), also connect the Host/Device/OTG USB connector on the board to your host PC. See the *Stellaris LM4F232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





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:



### 3.4 Installing USB Drivers for the USB Device Examples

The USB examples build upon the examples provide with StellarisWare and MWare. Because the examples mimic the same functionality, you can use the same drivers delivered with standalone installations of StellarisWare and Control Suite (MWare).

In TI-RTOS, the Windows USB drivers are located in the `<tirtos_install>/products/StellarisWare_####/windows_drivers` directory.

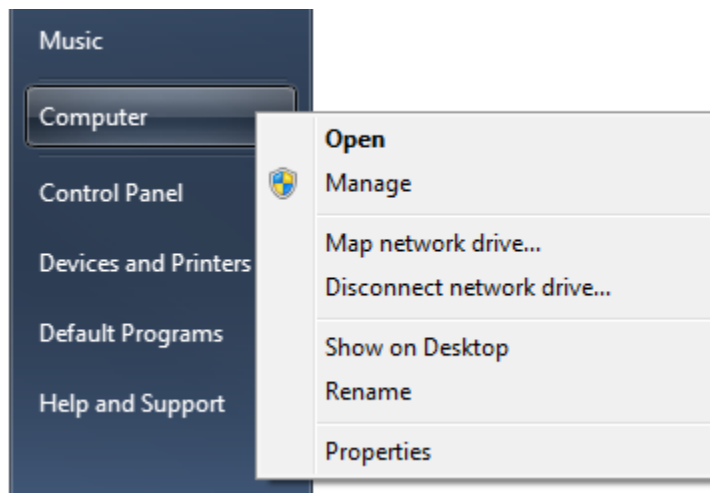
The Windows menus and dialogs you see may be slightly different from those shown here, depending on your version of Windows.

To install the USB driver, follow these steps:

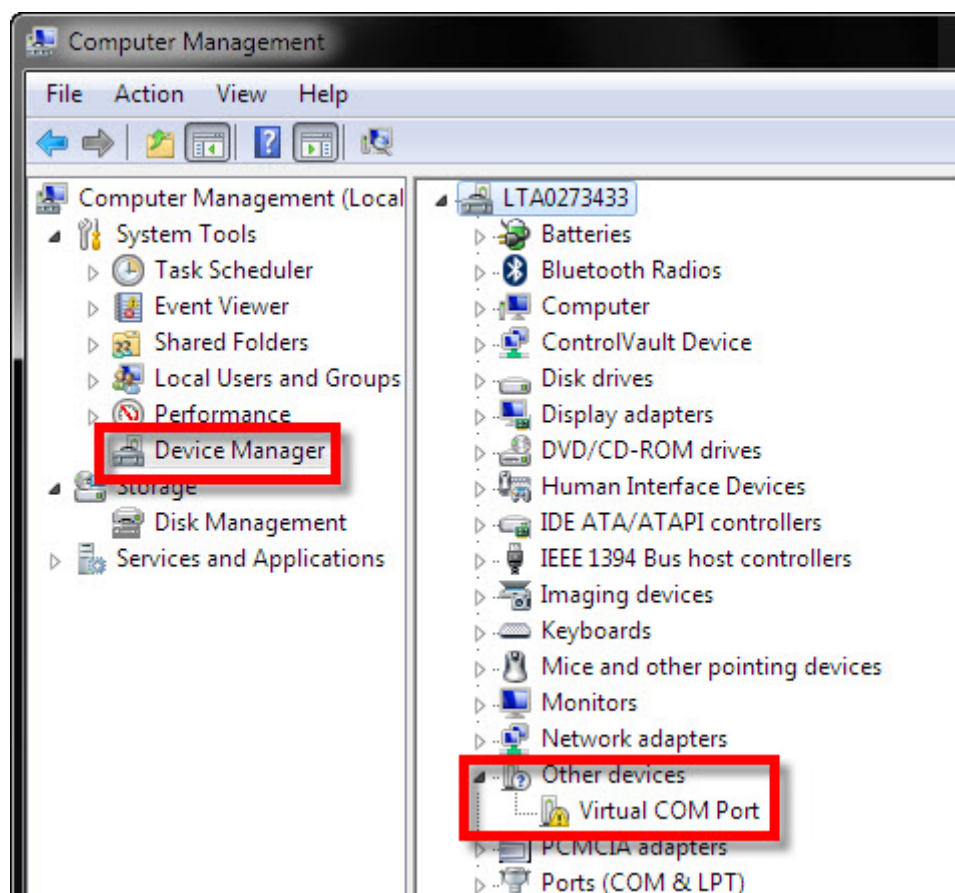
1. Load and run a USB device reference example—USB Keyboard Device, USB Mouse Device, or USB CDC Mouse Device.
2. While the example is running, connect the device to the Windows PC via a USB cable. At this point, Windows will detect the device and attempt to enumerate it.



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



- If you are prompted by a security warning in Windows 7, click **Yes**.
- Select the **Device Manager** category in the left pane.
- 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/StellarisWare_####/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.

### 3.5 Using Examples for Individual Components

An "empty" TI-RTOS project is provided in the CCS new project wizard. It contains some common code excerpts that enable different TI-RTOS components. This example is described in the *TI-RTOS User's Guide* (SPRUHD4).

## Configuring TI-RTOS

---

---

---

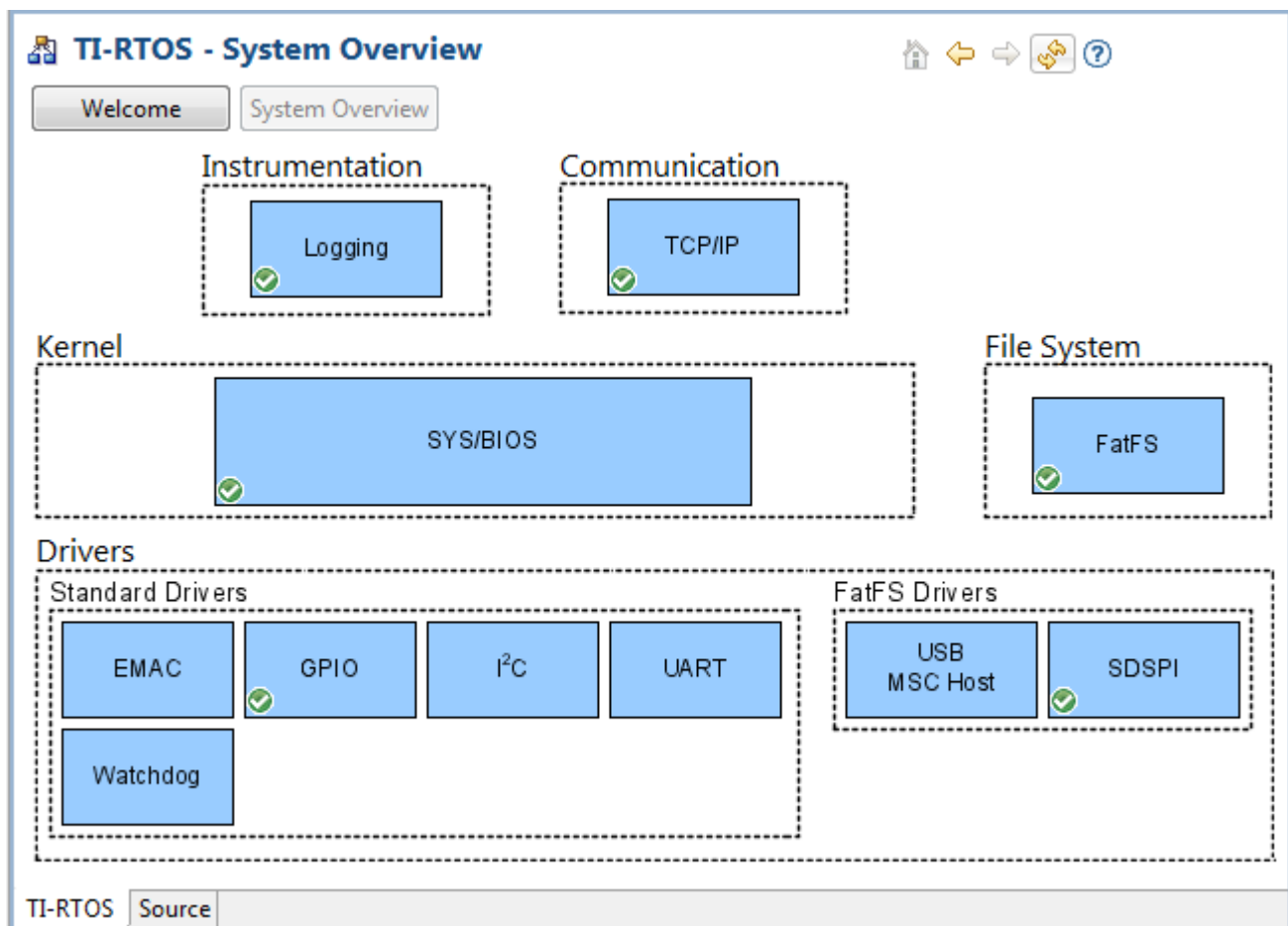
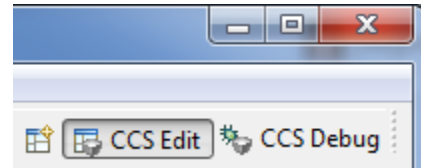
This chapter describes how to configure how TI-RTOS and its components will be used by your application.

Topic	Page
4.1 Starting the Configuration Tool . . . . .	28
4.2 Configuring TI-RTOS. . . . .	29
4.3 Configuring Individual Sub-Components . . . . .	30

## 4.1 Starting the Configuration Tool

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.

**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.

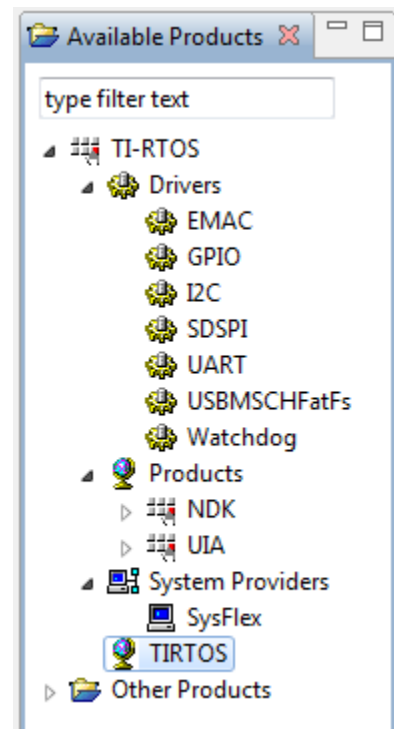
For details about how to use XGCONF, see Chapter 2 of the [SYS/BIOS User's Guide \(SPRUEX3\)](#).

## 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 can be configured:

- **EMAC.** Driver for Ethernet with the NDK. See Section 3.3.1.1 for how to set the MAC address.
- **GPIO.** Driver for the GPIO pins (and therefore the LEDs).
- **I2C.** Driver for the I<sup>2</sup>C peripheral.
- **SDSPI.** Driver for the SD card using an SPI (SSI) bus and the FatFS.
- **UART.** Driver for the UART peripheral.
- **USBMSCHFatFs.** Driver for the USB MSC Host controller.
- **Watchdog.** Driver for the watchdog timer.



The SysFlex 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 SysFlex 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.)

# Index

---

---

---

## A

Available Products list 29

## B

building examples 17

## C

C28x  
  support 6  
CCS  
  installation 13  
  other documentation 9  
ccxml file 17  
CDC device 6  
  example 18  
components 5  
Concerto 6  
  other documentation 11  
  resources used 7  
configuration 27  
  graphical editor 28  
ControlSuite 5  
  other documentation 10

## D

debugging examples 17  
disk space 13  
DK-LM3S9D96 6, 19  
  resources used 8  
documentation 9

## E

EK-LM4F120XL 6, 19  
  resources used 8  
EKS-LM4F232 6, 19  
  resources used 9  
EMAC driver 6  
  configuration 29  
  resources used 7, 8  
Ethernet driver 6  
examples 15  
  building 17

creating projects 16  
debugging 17  
overview 18

## F

F28M35H52C1 6  
F28M36P63C2 6  
FatFs API  
  other documentation 10  
Flash drives 6  
forum 9

## G

GPIO driver 6  
  configuration 29  
  resources used 7, 8, 9

## H

HID device 6  
  example 18

## I

I2C driver 6  
  configuration 29  
  example 18  
  resources used 7, 8  
installation  
  CCS 13  
  directory 13  
  TI-RTOS 14  
instrumented libraries 29  
IPC 5  
  example 18  
  other documentation 10  
  resources used 7

## L

LEDs  
  managed by GPIO driver 6  
LM3S9D96 6  
LM4F120H5QR 6

---

LM4F232H5QD 6

## M

MAC address 20, 21  
MSC device 6  
MWare 5  
    other documentation 10

## N

NDK 5  
    example 18  
    MAC address 20, 21  
    other documentation 10  
non-instrumented libraries 29

## P

products directory 5

## R

readme.txt file 20

## S

SD driver  
    example 18  
SDSPI driver 6  
    configuration 29  
    resources used 7, 8, 9  
simulator, debugging with 17  
StellarisWare 5  
SYS/BIOS 5  
    examples 18  
    other documentation 9  
    resources used 7  
SysFlex module  
    configuration 29  
System Overview configuration 28

system requirements 13

## T

target configuration file 17  
TI-RTOS  
    other documentation 9  
TMDXDOCK28M36 6, 19  
    resources used 7  
TMDXDOCKH52C1 6, 19  
    resources used 7

## U

UART driver 6  
    configuration 29  
    resources used 7, 8, 9  
UIA 5  
    example 18  
    other documentation 10  
USB driver  
    example 18  
    resources used 7, 8, 9  
USBMSCHFatFs driver 6  
    configuration 29

## W

Watchdog driver 6  
    configuration 29  
    resources used 7, 8, 9  
wiki 9

## X

XDCtools 5  
    other documentation 10  
XGCONF  
    configuring other components 30  
    configuring TI-RTOS modules 29  
    starting 28



## 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

Audio	<a href="http://www.ti.com/audio">www.ti.com/audio</a>
Amplifiers	<a href="http://amplifier.ti.com">amplifier.ti.com</a>
Data Converters	<a href="http://dataconverter.ti.com">dataconverter.ti.com</a>
DLP® Products	<a href="http://www.dlp.com">www.dlp.com</a>
DSP	<a href="http://dsp.ti.com">dsp.ti.com</a>
Clocks and Timers	<a href="http://www.ti.com/clocks">www.ti.com/clocks</a>
Interface	<a href="http://interface.ti.com">interface.ti.com</a>
Logic	<a href="http://logic.ti.com">logic.ti.com</a>
Power Mgmt	<a href="http://power.ti.com">power.ti.com</a>
Microcontrollers	<a href="http://microcontroller.ti.com">microcontroller.ti.com</a>
RFID	<a href="http://www.ti-rfid.com">www.ti-rfid.com</a>
OMAP Mobile Processors	<a href="http://www.ti.com/omap">www.ti.com/omap</a>
Wireless Connectivity	<a href="http://www.ti.com/wirelessconnectivity">www.ti.com/wirelessconnectivity</a>

### Applications

Automotive and Transportation	<a href="http://www.ti.com/automotive">www.ti.com/automotive</a>
Communications and Telecom	<a href="http://www.ti.com/communications">www.ti.com/communications</a>
Computers and Peripherals	<a href="http://www.ti.com/computers">www.ti.com/computers</a>
Consumer Electronics	<a href="http://www.ti.com/consumer-apps">www.ti.com/consumer-apps</a>
Energy and Lighting	<a href="http://www.ti.com/energy">www.ti.com/energy</a>
Industrial	<a href="http://www.ti.com/industrial">www.ti.com/industrial</a>
Medical	<a href="http://www.ti.com/medical">www.ti.com/medical</a>
Security	<a href="http://www.ti.com/security">www.ti.com/security</a>
Space, Avionics and Defense	<a href="http://www.ti.com/space-avionics-defense">www.ti.com/space-avionics-defense</a>
Video & Imaging	<a href="http://www.ti.com/video">www.ti.com/video</a>
<b>TI E2E Community</b>	<a href="http://e2e.ti.com">e2e.ti.com</a>