Getting Started Guide
SWRU467A–February 2017–Revised June 2018

CC3120 SimpleLink™ Wi-Fi® Internet-on-a chip™ Solution SDK Getting Started Guide

The CC3120 device is part of the SimpleLink™ microcontroller (MCU) platform, which consists of Wi-Fi®, Bluetooth® low energy, Sub-1 GHz, and host MCUs. All share a common, easy-to-use development environment with a single core software development kit (SDK) and rich tool set. A one-time integration of the SimpleLink platform lets you add any combination of devices from the portfolio into your design. The ultimate goal of the SimpleLink platform is to achieve 100 percent code reuse when your design requirements change. For more information, visit www.ti.com/simplelink.

For more information, visit www.ti.com/simplelink.

This guide is intended to help users in the initial setup and demonstration of the different demos in the CC3120 SDK. The guide lists the software and hardware components required to get started, and explains how to install the supported integrated development environment (IDE), SimpleLink CC3120 SDK, and the various other tools required.

Contents

Trademarks

1 Introduction

The goal of this tutorial is to show users how to set up the development environment, and execute a SimpleLink Wi-Fi CC3120 project, on both the MSP-EXP432P401R LaunchPad™ and SimpleLink Studio, using several IDEs:

- MSP-EXP432P401R LaunchPad
  - Code Composer Studio™ (CCS)
  - IAR Embedded Workbench®
- SimpleLink Studio
  - Visual Studio 2015

2 Prerequisites

2.1 Hardware

The user is expected to have the following:

- SimpleLink CC3120 BoosterPack™ Evaluation Board (CC3120BOOST or BOOSTXL-CC3120MOD)
- SimpleLink CC31xx Emulation BoosterPack Debug Board (CC31XXEMUBOOST)
- MSP-EXP432P401R LaunchPad Development Kit (MSP-EXP432P401R)
- Micro-USB cable
- 802.11 b/g/n wireless router with Internet access
- PC running at least a Microsoft® Windows® 7 operating system with Internet access

2.2 Software Installation

The user is expected to have the following:

- SimpleLink MSP432™ SDK
- SimpleLink CC3120 SDK (also referred to as the MSP432 Wi-Fi plug-in)
- SimpleLink CC3120 ServicePack. It is part of the SDK under
  <SDK install dir>/ tools/cc32xx_tools/servicepack-cc3x20
- UniFlash v4.x for flashing the device
- CCS Version 7.0.0 or later
- IAR Embedded Workbench Version 7.80.1 or later, for ARM
- Visual Studio 2015, download the Visual Studio Community package.
- FreeRTOSv9.0.0 – the SDK does not include FreeRTOS sources. The user is expected to download the FreeRTOS sources.

This tutorial assumes the packages are installed at the default location. The paths for these installations may be different, depending on the version of the software you downloaded. Regardless, users can manually define these paths without affecting the overall experience.

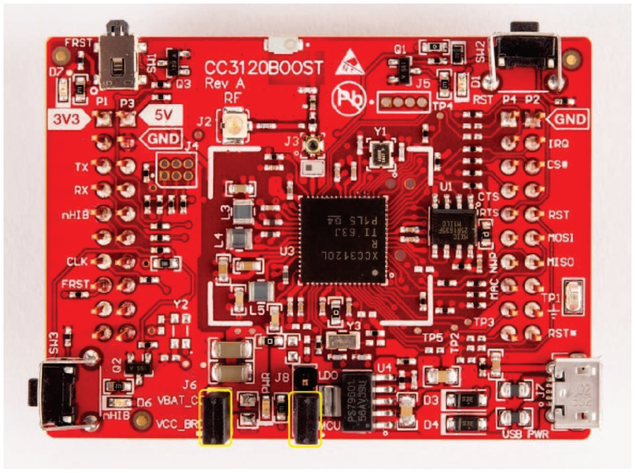
If FTDI drivers are not installed on your Windows PC, install the drivers when prompted by the SDK installer.

3 Board Configuration

3.1 Board Setup For Programming

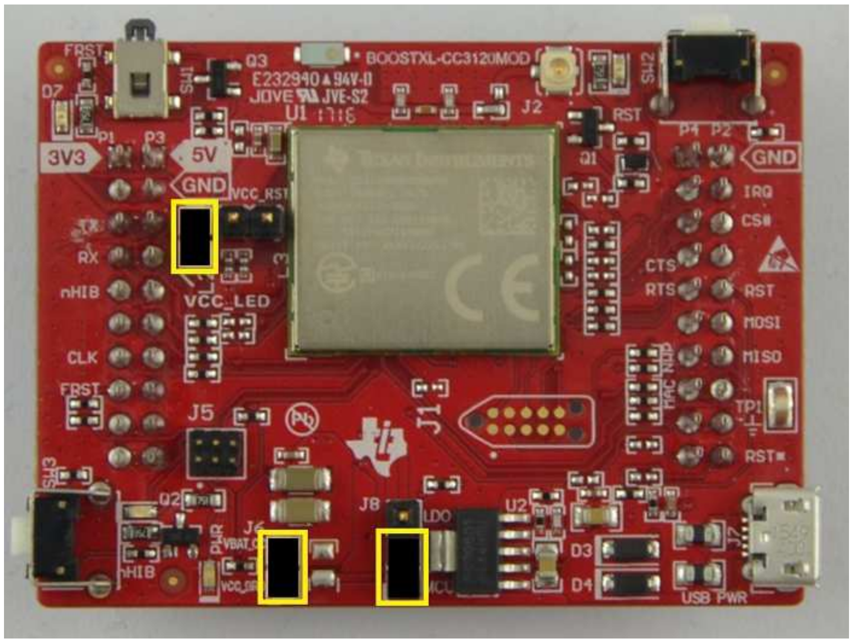
To program the CC3120BOOST or BOOSTXL-CC3120MOD board, the CC31XXEMUBOOST debug board is required. The CC31XXEMUBOOST board uses an FTDI chipset, which connects to the CC3120 through UART for programming purposes, and also connects to the reset line for automatic reset. The CC31XXEMUBOOST debug board is required for programming only.

Figure 1, Figure 2, and Figure 3 show how to properly configure the jumpers on the CC3120BOOST or BOOSTXL-CC3120MOD and CC31XXEMUBOOST boards.



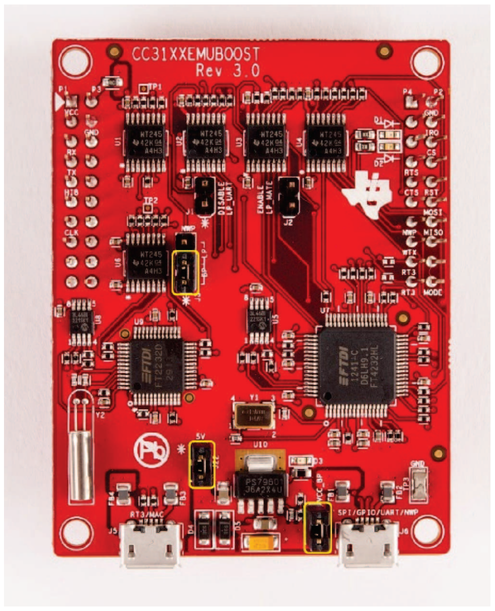
Copyright © 2017, Texas Instruments Incorporated

Figure 1. CC3120BOOST Board With Jumper Configuration



Copyright © 2017, Texas Instruments Incorporated

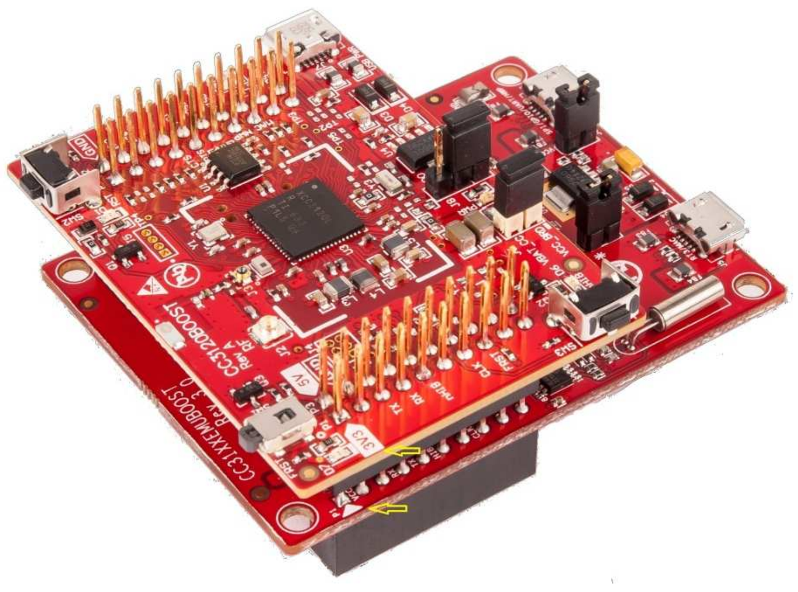
Figure 2. BOOSTXL-CC3120MOD Board With Jumper Configuration



Copyright © 2017, Texas Instruments Incorporated

Figure 3. CC31XXEMUBOOST Board With Jumper Configuration

Figure 4 shows how to mount the CC3120BOOST board on top of the CC31XXEMUBOOST board. Figure 5 shows how to mount the BOOSTXL-CC3120MOD on top of the CC31XXEMUBOOST board. Ensure P1.1 of both boards are aligned with each other, as indicated by the yellow arrows on both boards. The LDO and MCU jumper (J8) configuration is used for choosing the power source. This guide uses the MCU configuration. To supply power directly to either of the BoosterPack boards, change the jumper to LDO, and connect USB PWR (J7) with a USB cable.



Copyright © 2017, Texas Instruments Incorporated

Figure 4. CC3120BOOST Board Mounted on Top of CC31XXEMUBOOST Board

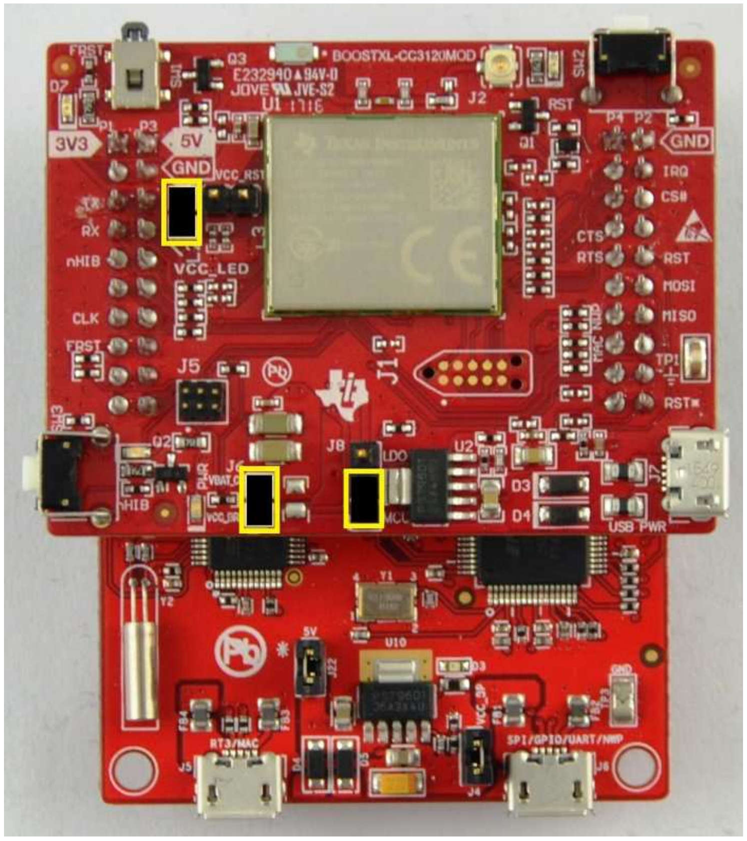

Copyright © 2017, Texas Instruments Incorporated

Figure 5. BOOSTXL-CC3120MOD Mounted on Top of CC31XXEMUBOOST

To power the platforms, connect the J6 port of the CC31XXEMUBOOST board to the Windows PC using the micro USB cable (see Figure 6).

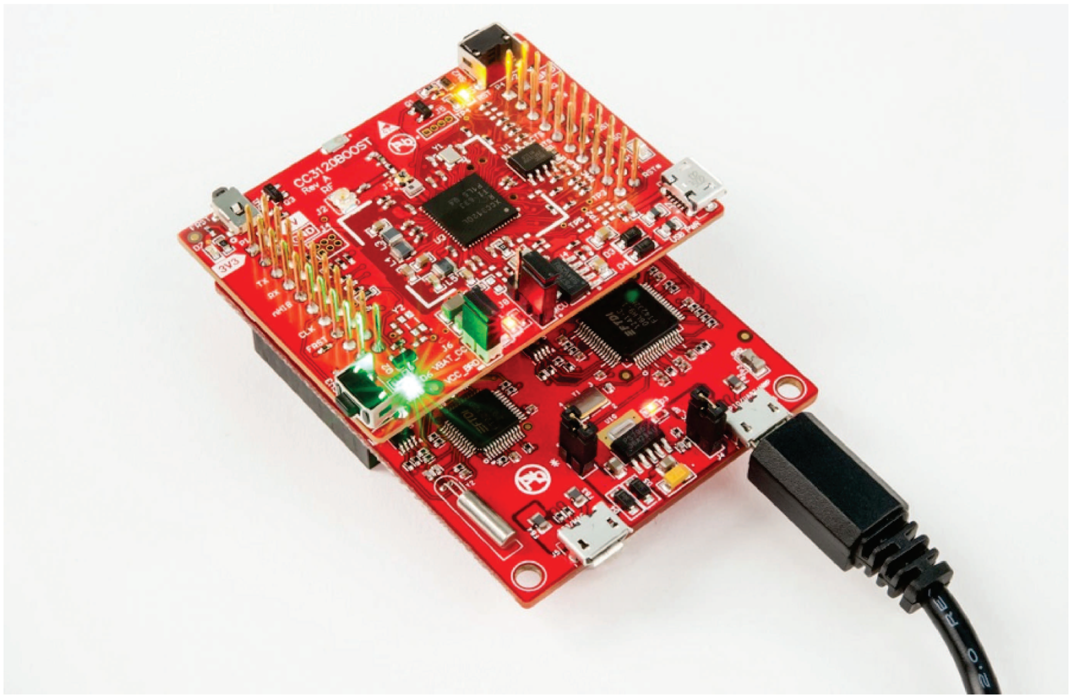


Figure 6. Powering the Platforms

If the drivers are properly installed as mentioned during the SDK installation, then the PC should recognize and list the connected device in the Device Manager (see Figure 7). The boards are now ready for programming.

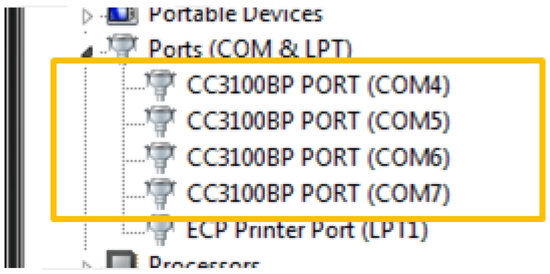

Portable Devices
Ports (COM & LPT)
CC3100BP PORT (COM4)
CC3100BP PORT (COM5)
CC3100BP PORT (COM6)
CC3100BP PORT (COM7)
ECP Printer Port (LPT1)
Processors

Figure 7. Device Manager Showing Ports

3.2 Board Setup With An External MCU

The CC3120 BoosterPack evaluation board requires a host MCU to execute the user application. This guide uses the MSP-EXP432P401R LaunchPad as a reference.

Figure 8 shows how the MSP-EXP432P401R LaunchPad jumpers must be configured.

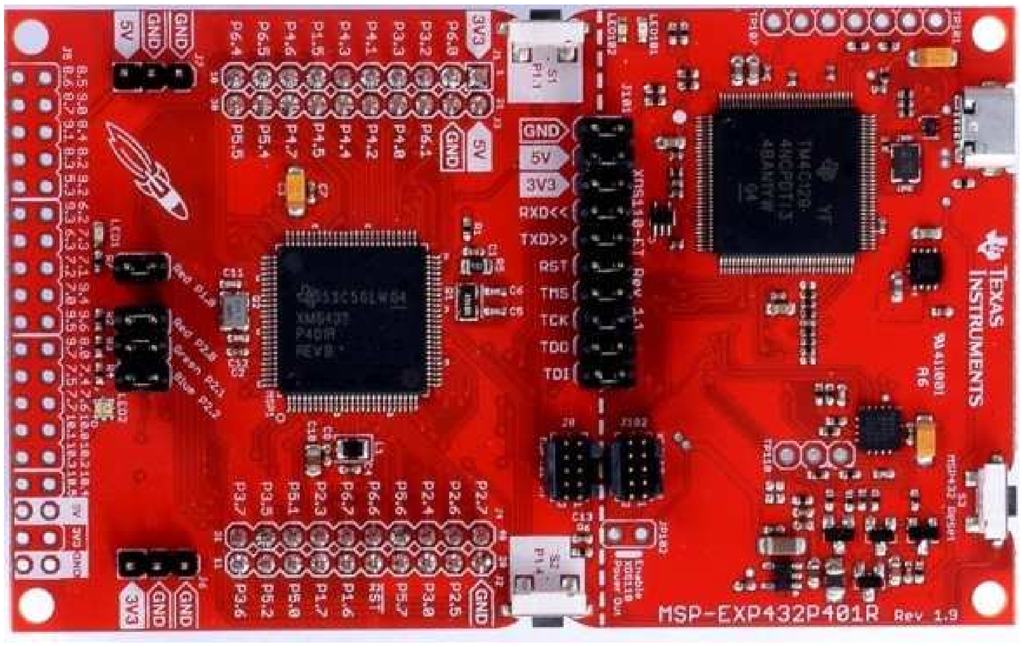


Figure 8. MSP-EXP432P401R With Jumper Configuration

Figure 9 shows how to mount the CC3120BOOST board on top of the MSP-EXP432P401R board. Ensure P1.1 of both boards are aligned with each other, as indicated by the white arrows on both boards. The micro USB connector on the MSP-EXP432P401R board is used as the power source and also as the programming interface for the user application.

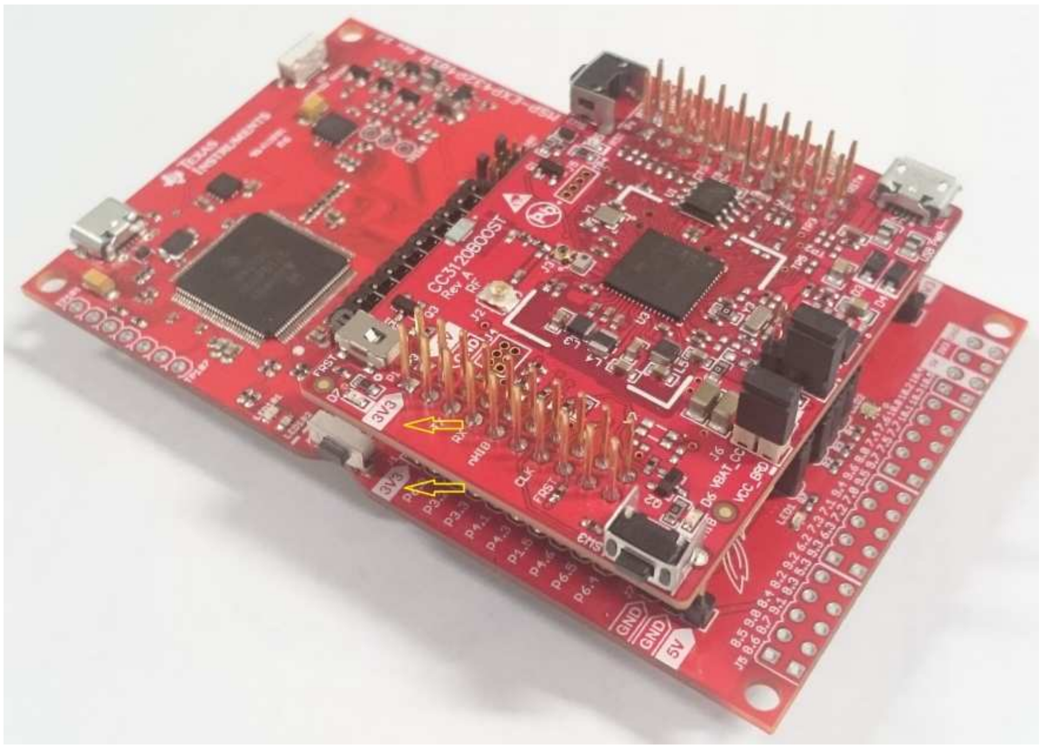


Figure 9. CC3120BOOST Board Mounted on Top of the MSP-EXP432P401R Board

4 Programming the CC3120 BoosterPack™

Regardless of the host MCU (MSP432 or Windows PC as SimpleLink Studio mode), the CC3120BOOST or BOOSTXL-CC3120MOD board must be programmed. As a minimum, the Service Pack is required. Other system and configuration files are also programmed by default.

To program a CC3120 BoosterPack, the CC31XXEMUBOOST board is required. Section 3.1 describes the setup.

To program a CC3120 BoosterPack, install the UniFlash utility and perform the following steps:

1. Connect the CC31XXEMUBOOST board to the PC using the micro USB cable.
2. Launch UniFlash, select *CC3120/CC3220* device, and then click the *Start Image Creator* button (see Figure 10).

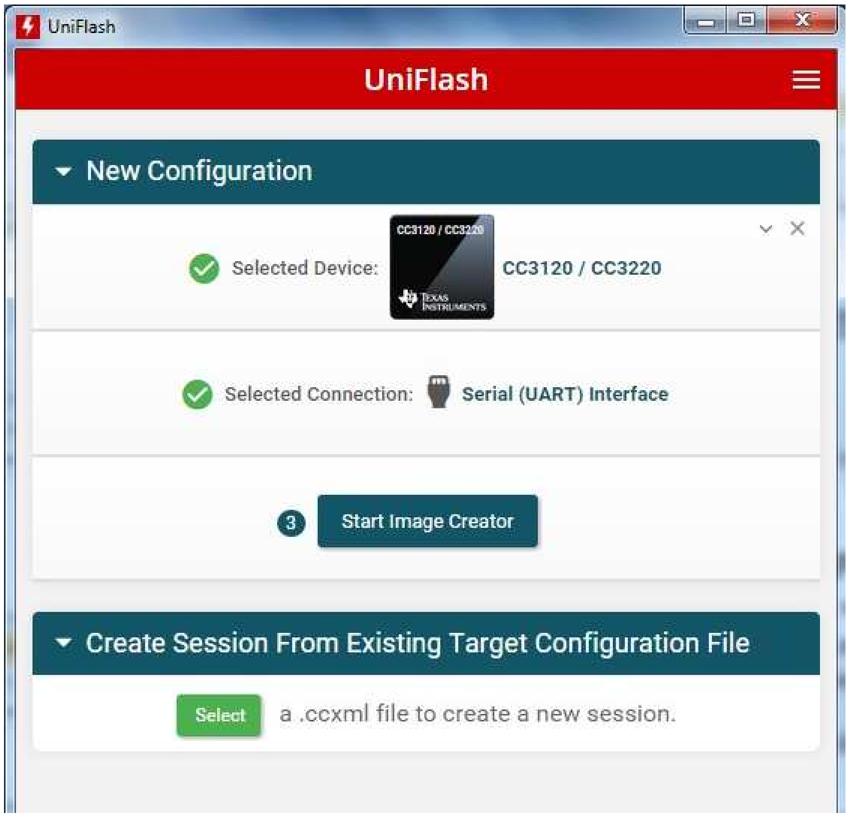
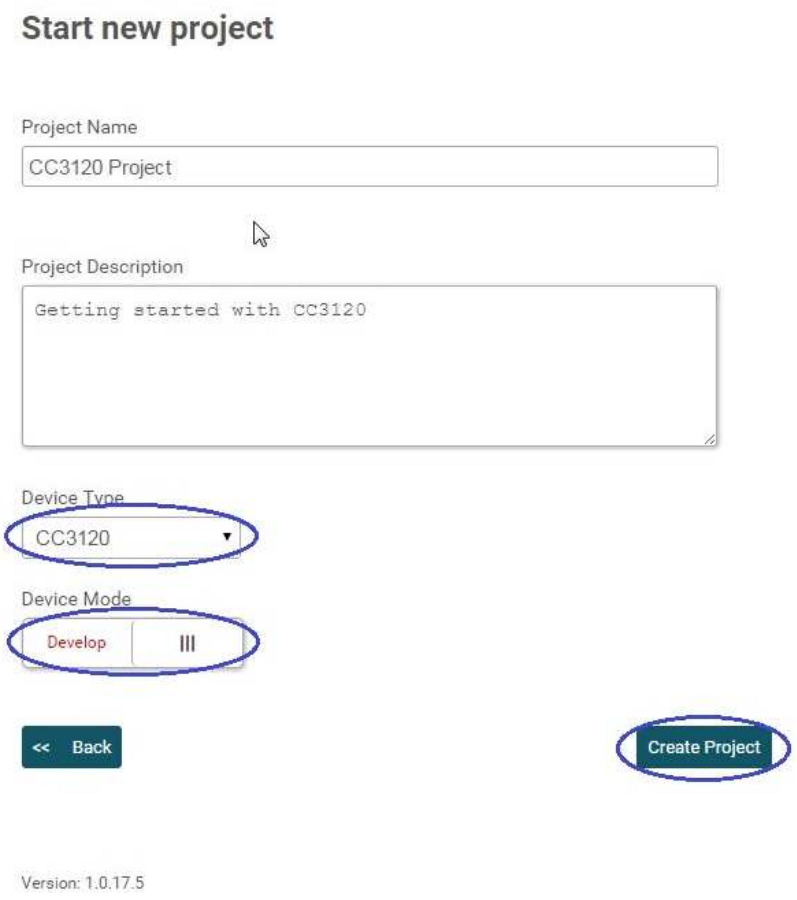


Figure 10. UniFlash Main Configuration Page

3. Select the *New Project* button to enter the project information (see Figure 11).
4. Enter a project name.
5. Select *CC3120* as the Device Type.
6. Select *Develop* as the Device Mode (choose *Production* mode in final production).
7. Click the *Create Project* button.



Start new project

Project Name: CC3120 Project

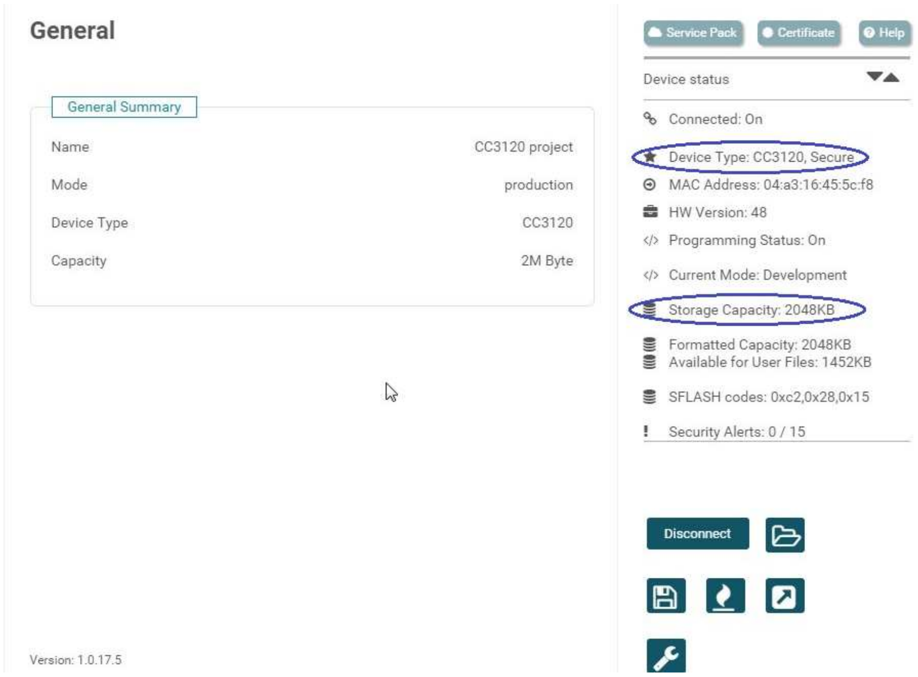
Project Description: Getting started with CC3120

Device Type: CC3120

Device Mode: Develop

Figure 11. UniFlash Create Project

8. Click the *Connect* button on the right side of the screen to connect to the board. If the connection is successful, the device information appears on the right side of the web page (see Figure 12).



Name	CC3120 project
Mode	production
Device Type	CC3120
Capacity	2M Byte

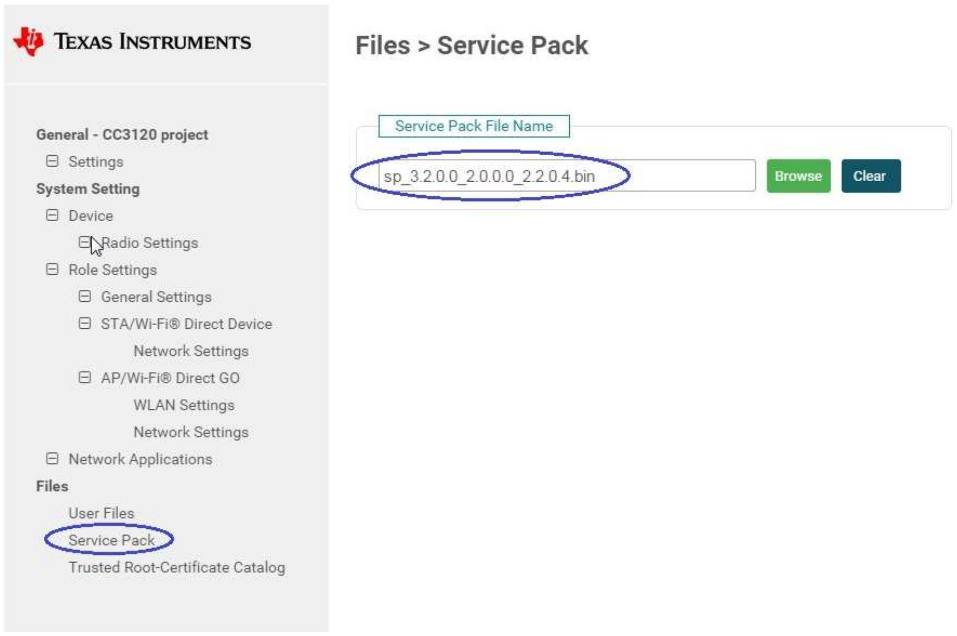
Connected: On
Device Type: CC3120, Secure
MAC Address: 04:a3:16:45:5c:f8
HW Version: 48
Programming Status: On
Current Mode: Development
Storage Capacity: 2048KB
Formatted Capacity: 2048KB
Available for User Files: 1452KB
SFLASH codes: 0xc2,0x28,0x15
Security Alerts: 0 / 15

Version: 1.0.17.5

Figure 12. UniFlash Connect to CC3120

9. On the left side of the page, click the *Service Pack* tab to bring up the Service Pack File Name selection dialog.
10. Click the *Browse* button and navigate to the Service Pack directory to select the binary (see Figure 13).

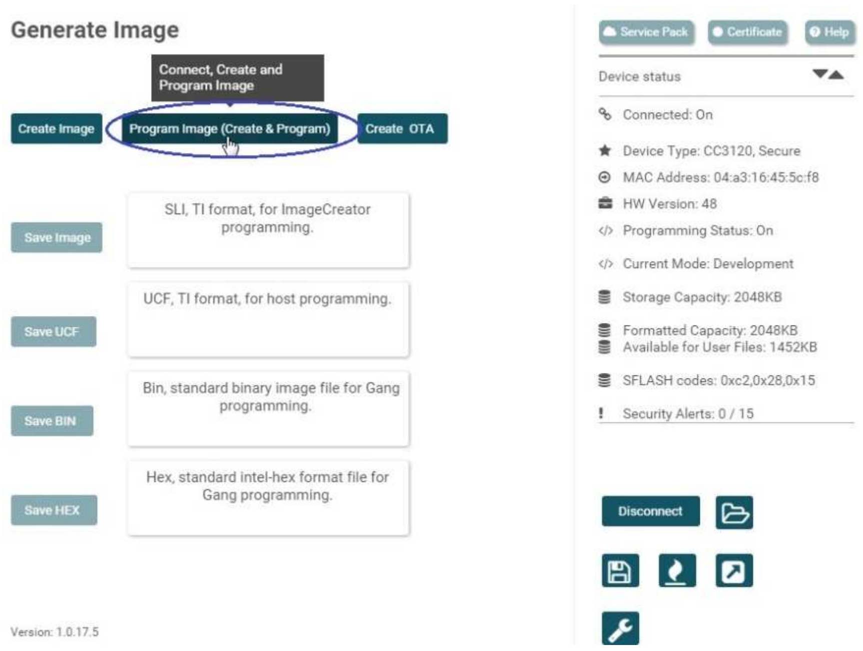
    The Service Pack is in the SDK under the <SDK install Dir>/tools/cc31xx_tools/servicepack-cc3x20 directory.



sp_3.2.0.0_2.0.0.0_2.2.0.4.bin

Figure 13. UniFlash Service Pack

11. Now click the *Generate Image* button on the right side of the web page to enter the programming page.
12. Click the center *Program Image (Create & Program)* button to create the image and start flashing it (see Figure 14). This operation may take a while to complete. Once the process is finished, a confirmation dialog appears.



SLI, TI format, for ImageCreator programming.

UCF, TI format, for host programming.

Bin, standard binary image file for Gang programming.

Hex, standard intel-hex format file for Gang programming.

Version: 1.0.17.5

Device status

Connected: On
Device Type: CC3120, Secure
MAC Address: 04:a3:16:45:5c:f8
HW Version: 48
Programming Status: On
Current Mode: Development
Storage Capacity: 2048KB
Formatted Capacity: 2048KB
Available for User Files: 1452KB
SFLASH codes: 0xc2,0x28,0x15
Security Alerts: 0 / 15

Figure 14. UniFlash Program Image

13. Once programming is complete, click the *Disconnect* button on the right side.

5 Getting Started With the MSP-EXP432P401R LaunchPad™

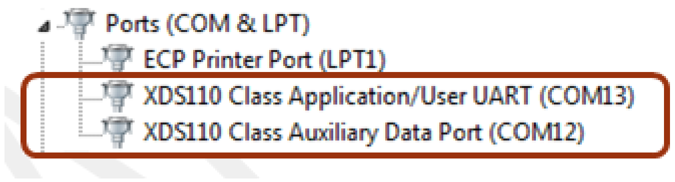
5.1 *Connecting the CC3120BOOST or BOOSTXL-CC3120MOD Board With the MSP-EXP432P401R Board*

Section 3.2 fully describes building this setup. Before connecting these two boards, the CC3120BOOST or BOOSTXL-CC3120MOD board must be programmed using UniFlash. Section 3.1 describes the programming setup.

5.2 *Terminal Setting*

The *network terminal* example demonstrates an interactive user interface in the form of a command line. To let the user type commands, get responses, and debug messages, a terminal utility is essential.

1. Connect the MSP-EXP432P401R LaunchPad to the PC with a micro USB cable. Windows should search for the driver automatically. To verify that the driver is properly installed, open the Device Manager on the PC, and then navigate to *Ports*. Two enumerated XDS110 ports, as shown in Figure 15, indicate if the driver was installed properly during CCS installation.



Ports (COM & LPT)
ECP Printer Port (LPT1)
XDS110 Class Application/User UART (COM13)
XDS110 Class Auxiliary Data Port (COM12)

Figure 15. Enumerated XDS Ports for MSP-EXP432P401R LaunchPad™

2. Launch a terminal emulation such as TeraTerm or PuTTY and configure it to the settings in Table 1.

Table 1. Serial Connection Settings

Parameter	Value
Port number	(The XDS110 application UART port number)
Baud rate	115,200
Data bits	8
Stop bits	1
Parity	None
Flow control	(Ignored)

The following sections describe setting up two different IDEs to compile and execute your first project: CCS and IAR.

5.3 *SDK Installation*

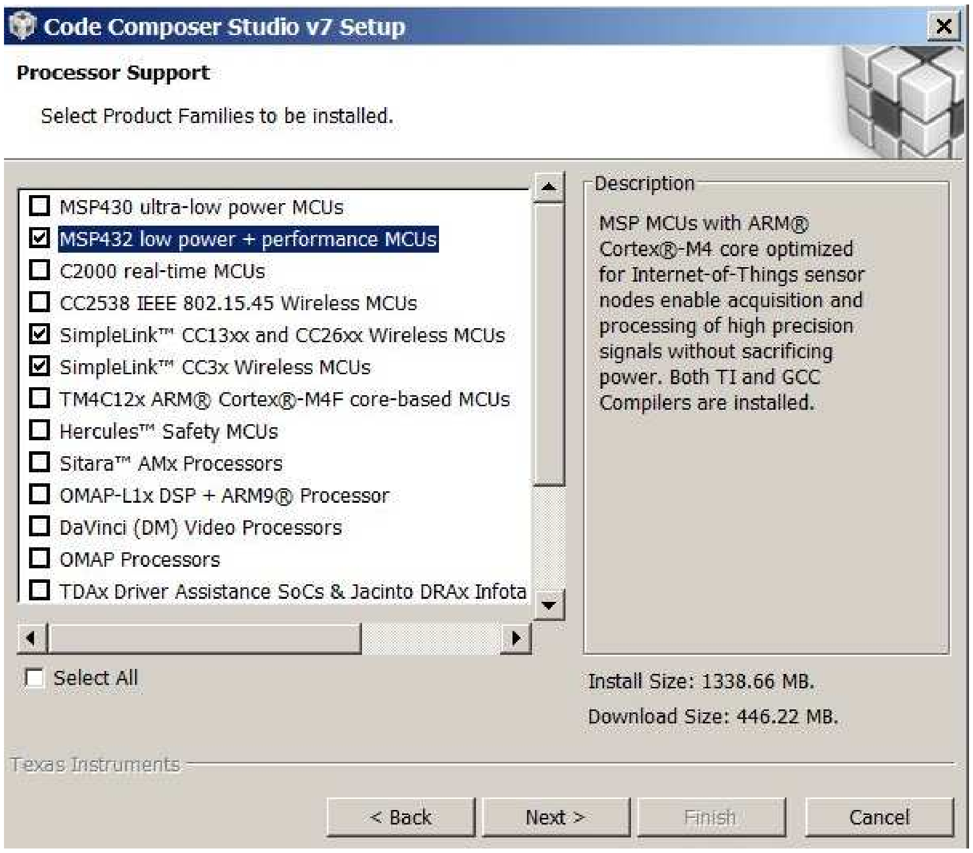
The SDK installation procedure involves two components: SimpleLink MSP432 SDK as the base SDK and SimpleLink CC3120 SDK as a Wi-Fi plug-in.

Install the base SDK first and then the plug-in in the default location pointed to by the installer.

5.4 Setting Up Code Composer Studio™

5.4.1 CCS™ Installation

Download CCS version 7.0.0 or later. During installation, TI recommends using the default directory of c:\ti\. When you are prompted to select the processor support, select the following options, as shown in Figure 16.



Processor Support

Select Product Families to be installed.

- [ ] MSP430 ultra-low power MCUs
- [x] MSP432 low power + performance MCUs
- [ ] C2000 real-time MCUs
- [ ] CC2538 IEEE 802.15.45 Wireless MCUs
- [x] SimpleLink™ CC13xx and CC26xx Wireless MCUs
- [x] SimpleLink™ CC3x Wireless MCUs
- [ ] TM4C12x ARM® Cortex®-M4F core-based MCUs
- [ ] Hercules™ Safety MCUs
- [ ] Sitara™ AMx Processors
- [ ] OMAP-L1x DSP + ARM9® Processor
- [ ] DaVinci (DM) Video Processors
- [ ] OMAP Processors
- [ ] TDAx Driver Assistance SoCs & Jacinto DRAx Infota

- [ ] Select All

Description

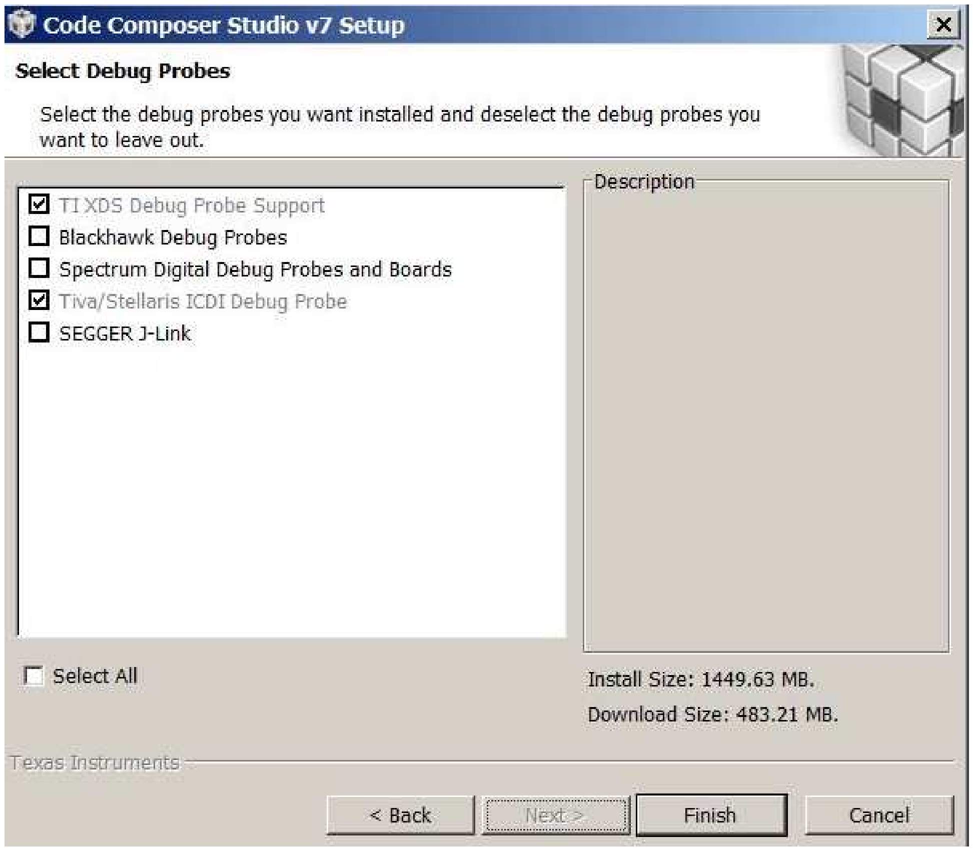
MSP MCUs with ARM® Cortex®-M4 core optimized for Internet-of-Things sensor nodes enable acquisition and processing of high precision signals without sacrificing power. Both TI and GCC Compilers are installed.

Install Size: 1338.66 MB.

Download Size: 446.22 MB.

Figure 16. Code Composer Studio™ Installation Packages Selection

To use a debug probe, select TI XDS Debug Probe Support and any other options you would like, as shown in Figure 17.


Code Composer Studio v7 Setup

Select Debug Probes

Select the debug probes you want installed and deselect the debug probes you want to leave out.

- [✓] TI XDS Debug Probe Support
- [ ] Blackhawk Debug Probes
- [ ] Spectrum Digital Debug Probes and Boards
- [✓] Tiva/Stellaris ICDI Debug Probe
- [ ] SEGGER J-Link

Description

[ ] Select All

Install Size: 1449.63 MB.

Download Size: 483.21 MB.

Texas Instruments

Figure 17. Code Composer Studio™ Installation Debug Probes

NOTE: The installation may take a while to complete, depending on the number of selections and the speed of the network, because CCS needs to download these files. Once the installation is finished, start CCS and users will be prompted to choose a workspace folder, which is the folder where the project files reside.

5.4.2 Post Code Composer Studio™ Installation Setup

Support for freeRTOSv9 must be added. To add the support, do the following:

1. Download the FreeRTOS official version 9 and install into file C:
2. Run CCS and choose Window → Preferences → Code Composer Studio → Build → Variables → Add (see Figure 18).

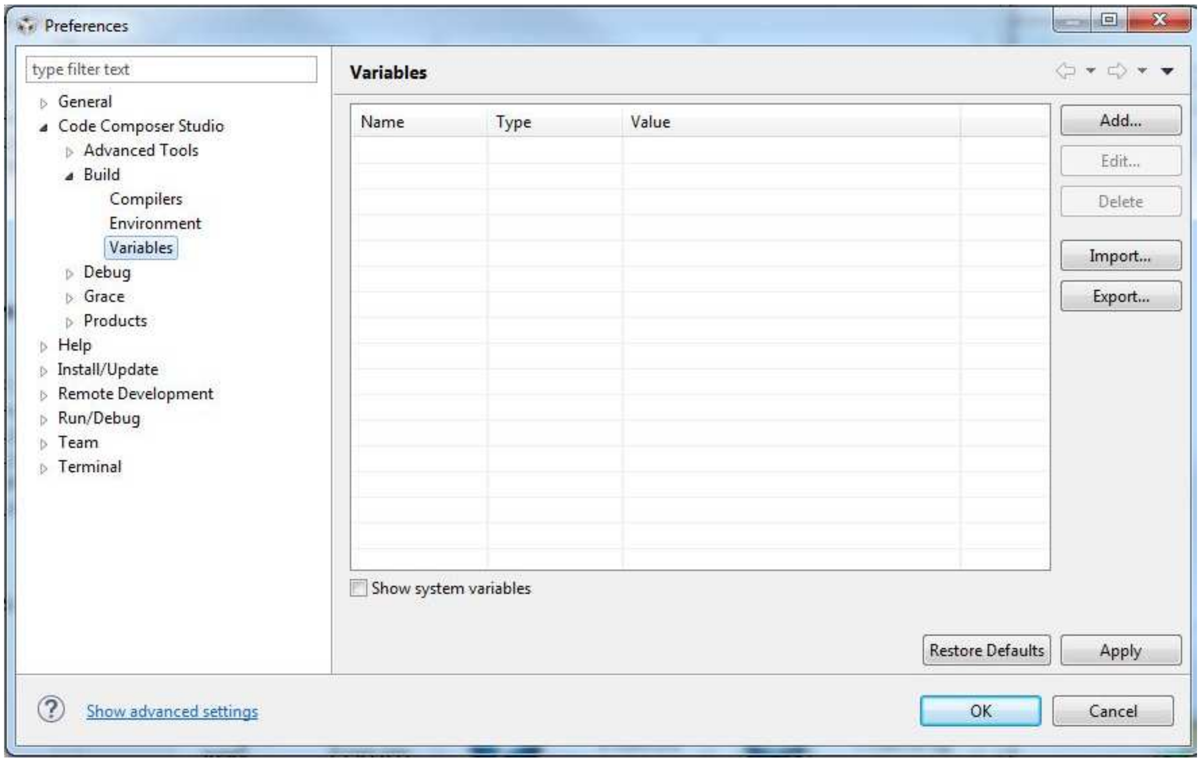
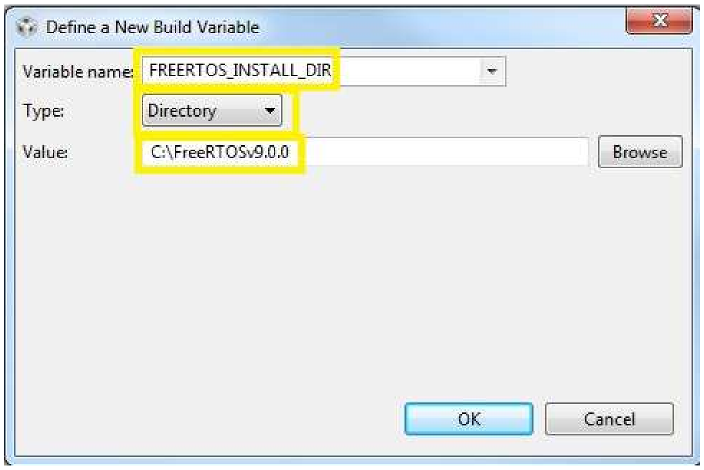


Figure 18. Code Composer Studio™ Installation – Add Variable

3. Fill the Variable name field with *FREERTOS_INSTALL_DIR* (see Figure 19).
4. Change the Type field to directory.
5. Click the *Browse* button to populate the Value field with the <FREERTOS_INSTALL_PATH> and then click *OK*.


Define a New Build Variable

Variable name: FREERTOS_INSTALL_DIR

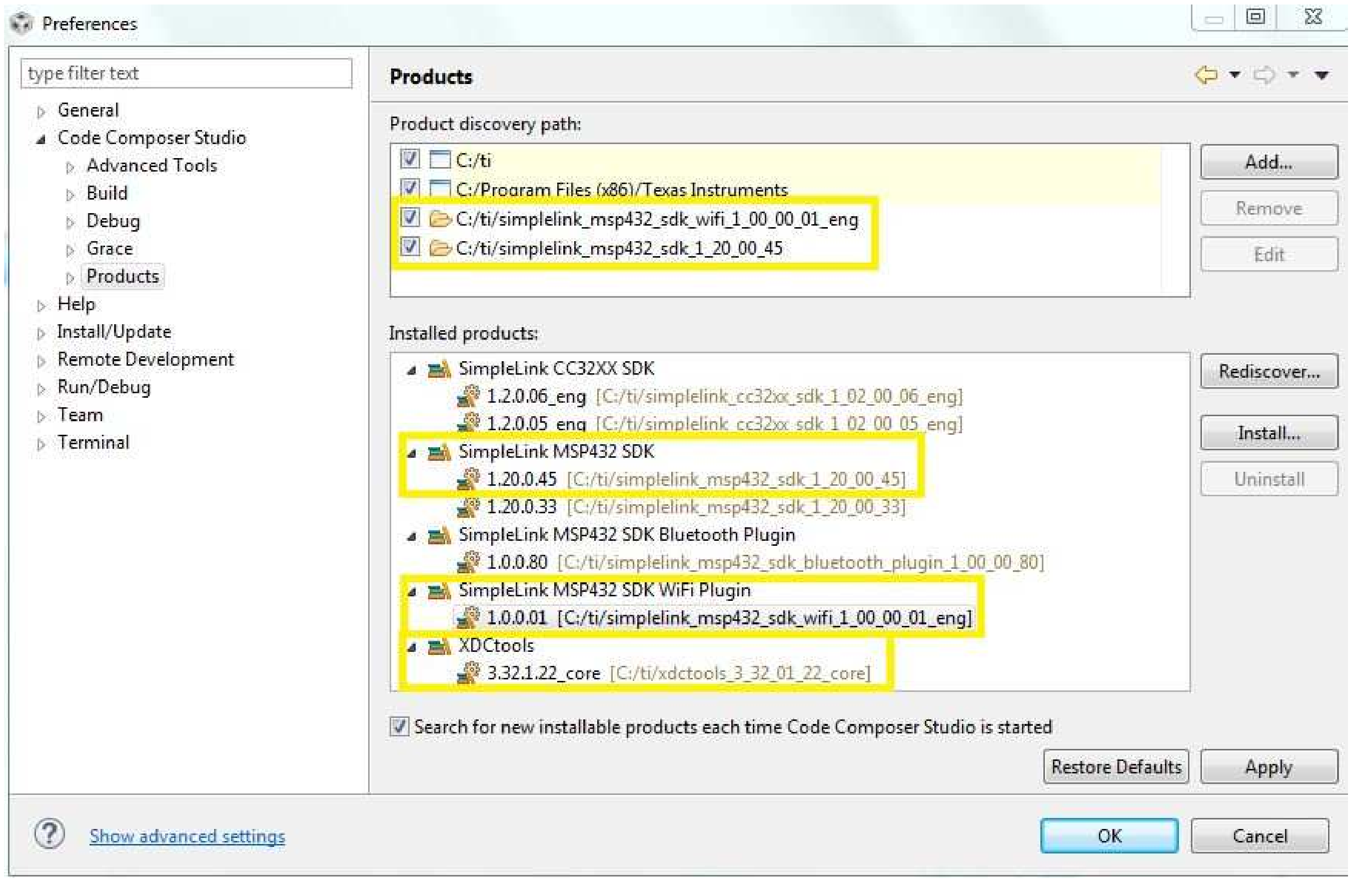
Type: Directory

Value: C:\FreeRTOSv9.0.0

Figure 19. Code Composer Studio™ Installation – FreeRTOS Support

Finally, installed products must be verified.

1. Choose Window → Preferences → Code Composer Studio → Products (see Figure 20).



Preferences

type filter text

- General
- Code Composer Studio
  - Advanced Tools
  - Build
  - Debug
  - Grace
  - Products
- Help
- Install/Update
- Remote Development
- Run/Debug
- Team
- Terminal

Products

Product discovery path:

- [✓] C:/ti
- [✓] C:/Program Files (x86)/Texas Instruments
- [✓] C:/ti/simplelink_msp432_sdk_wifi_1_00_00_01_eng
- [✓] C:/ti/simplelink_msp432_sdk_1_20_00_45

Installed products:

- SimpleLink CC32XX SDK
  - 1.2.0.06_eng [C:/ti/simplelink_cc32xx_sdk_1_02_00_06_eng]
  - 1.2.0.05_eng [C:/ti/simplelink_cc32xx_sdk_1_02_00_05_eng]
- SimpleLink MSP432 SDK
  - 1.20.0.45 [C:/ti/simplelink_msp432_sdk_1_20_00_45]
  - 1.20.0.33 [C:/ti/simplelink_msp432_sdk_1_20_00_33]
- SimpleLink MSP432 SDK Bluetooth Plugin
  - 1.0.0.80 [C:/ti/simplelink_msp432_sdk_bluetooth_plugin_1_00_00_80]
- SimpleLink MSP432 SDK WiFi Plugin
  - 1.0.0.01 [C:/ti/simplelink_msp432_sdk_wifi_1_00_00_01_eng]
- XDCtools
  - 3.32.1.22_core [C:/ti/xdctools_3_32_01_22_core]

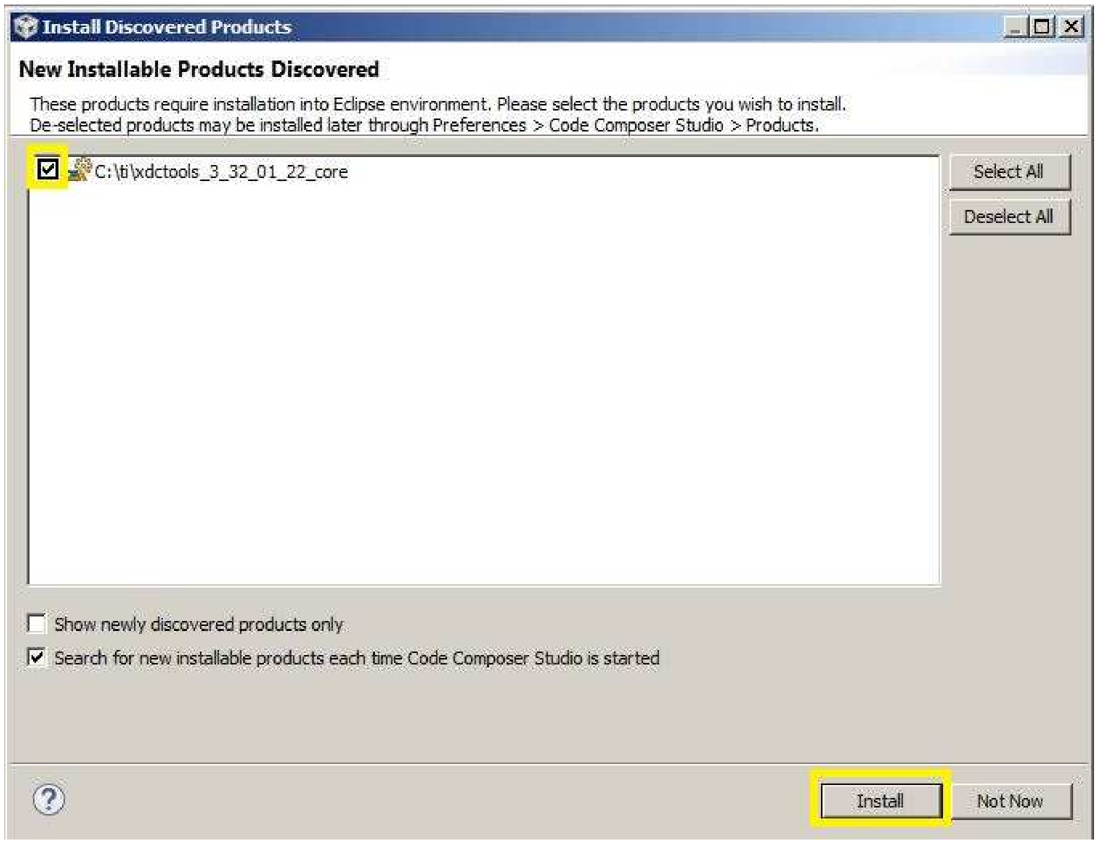
[✓] Search for new installable products each time Code Composer Studio is started

Figure 20. Code Composer Studio™ Installation Products

2. Check that the paths in the Product Discovery Path include the following:
   - SimpleLink MSP432 SDK WiFi plug-in
   - SimpleLink MSP432 SDK
   - XDCtools

3. If the paths are not in the Product Discovery Path, add them and then click the Rediscover button.

   An Install Discovered Products window may pop up (see Figure 21). If the window appears, it means that new products have been discovered. If the products in Step 2 are in the list, choose them and select install.



Install Discovered Products

New Installable Products Discovered

These products require installation into Eclipse environment. Please select the products you wish to install.
De-selected products may be installed later through Preferences > Code Composer Studio > Products.

[✓] C:\ti\xdctools_3_32_01_22_core

[ ] Show newly discovered products only
[✓] Search for new installable products each time Code Composer Studio is started

Figure 21. Code Composer Studio™ Installation – New Products Discovered

5.4.3 Importing network_terminal Example in Code Composer Studio™

1. Open CCS.
2. Click on the *Project* button, go to import CCS projects, and browse to the location of the plug-in installation (<Plugin_INSTALL_DIR>/examples). CCS lists all the CCS projects inside the specified directory and its subdirectories.

   The SDK provides TI-RTOS and FreeRTOS-based examples. TI-RTOS and FreeRTOS examples have the word *tirtos* or *freertos* in the project name. Also, CCS lists both CCS- and GCC-based examples.

   Importing TI-RTOS and FreeRTOS examples also imports kernel projects in the same workspace. The kernel project is a dependent project and is automatically built when the example is built.

   For more details and information on how to change configurations, see the <Plugin_INSTALL_DIR>/docs/simplelink_mcu_sdk/Users_Guide.html document.

3. Select the network_terminal-based example and click *Finish* (see Figure 22).

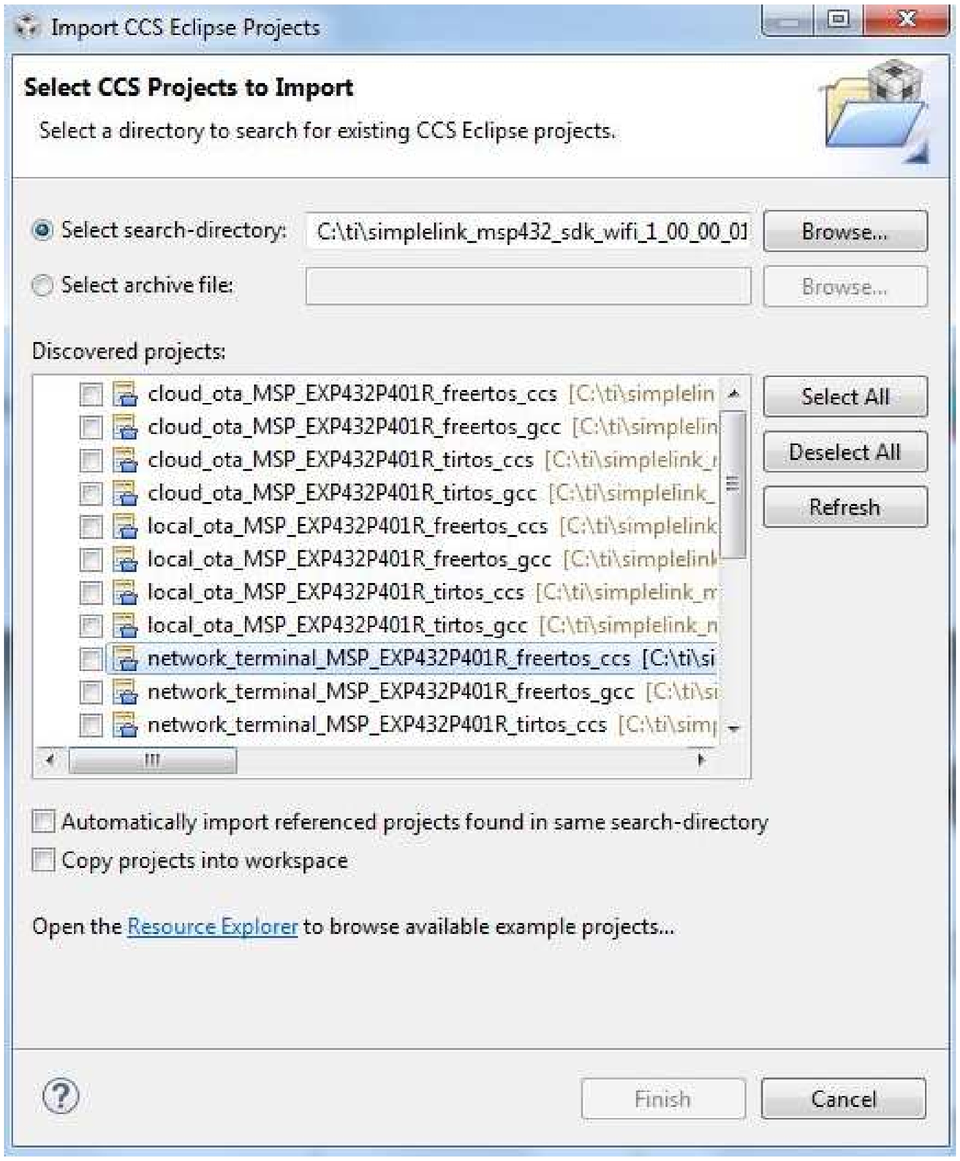

Import CCS Eclipse Projects
Select CCS Projects to Import
Select a directory to search for existing CCS Eclipse projects.
Select search-directory: C:\ti\simplelink_msp432_sdk_wifi_1_00_00_01
Browse...
Select archive file:
Browse...
Discovered projects:
cloud_ota_MSP_EXP432P401R_freertos_ccs [C:\ti\simplelin
cloud_ota_MSP_EXP432P401R_freertos_gcc [C:\ti\simplelin
cloud_ota_MSP_EXP432P401R_tirtos_ccs [C:\ti\simplelink_
cloud_ota_MSP_EXP432P401R_tirtos_gcc [C:\ti\simplelink_
local_ota_MSP_EXP432P401R_freertos_ccs [C:\ti\simplelink
local_ota_MSP_EXP432P401R_freertos_gcc [C:\ti\simplelink
local_ota_MSP_EXP432P401R_tirtos_ccs [C:\ti\simplelink_m
local_ota_MSP_EXP432P401R_tirtos_gcc [C:\ti\simplelink_n
network_terminal_MSP_EXP432P401R_freertos_ccs [C:\ti\si
network_terminal_MSP_EXP432P401R_freertos_gcc [C:\ti\si
network_terminal_MSP_EXP432P401R_tirtos_ccs [C:\ti\sim
Select All
Deselect All
Refresh
Automatically import referenced projects found in same search-directory
Copy projects into workspace
Open the Resource Explorer to browse available example projects...
Finish
Cancel

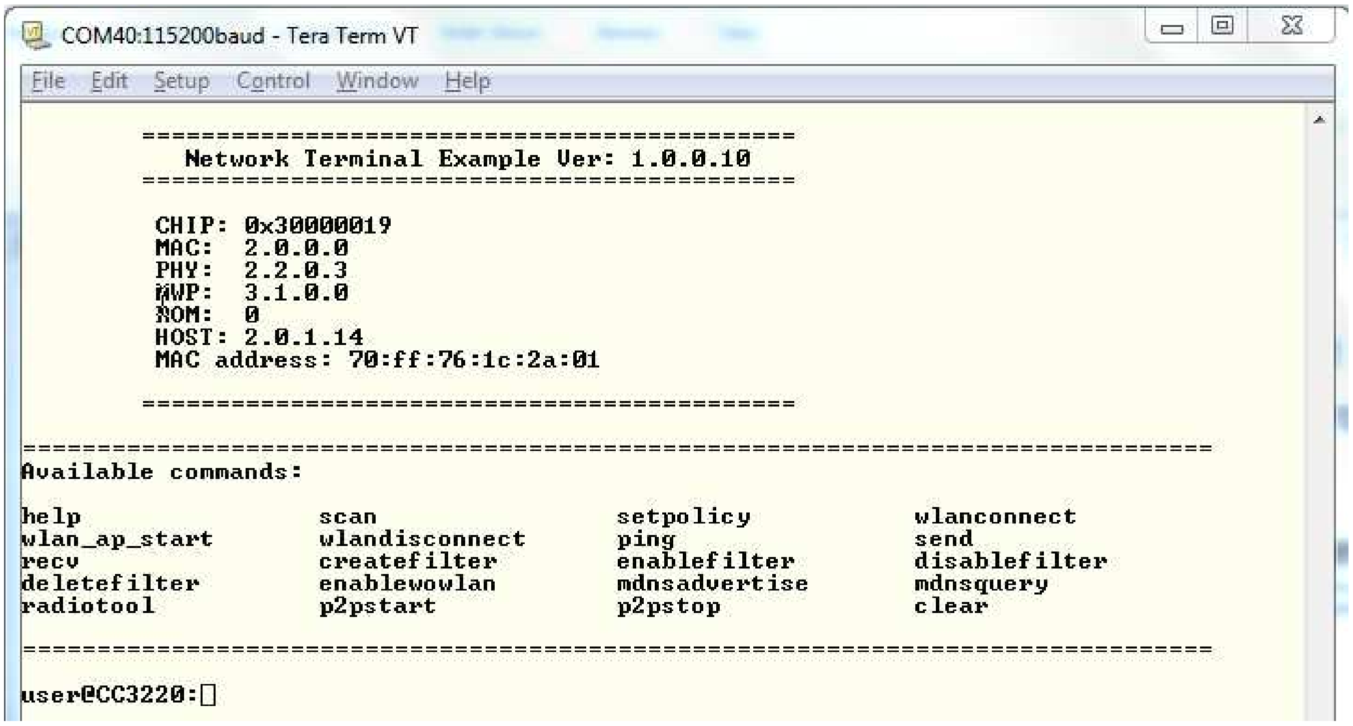
Figure 22. Code Composer Studio™ Import network_terminal Example

5.4.4 Compiling and Debugging the Project in Code Composer Studio™

Right-click on the network_terminal project from the Project Explorer window, and select *rebuild project*. CCS compiles the chosen kernel first and then the netwok_terminal example. To debug the example, follow these steps:

1. Open a terminal window and set it as described in Section 5.2.
2. Right-click on the project from the Project Explorer window, and select Debug As → Code Composer Debug Session, to start the debug process. This process downloads the code to the device and begins debugging. If this is your first time debugging the code, the process may take longer to complete.
3. Click the green arrow on the top of the screen to start executing your code.

4. Go back to the terminal window. You can see the message being printed as the application runs on the MSP432 device (see Figure 23).


COM40:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
        ============================================
           Network Terminal Example Ver: 1.0.0.10
        ============================================

         CHIP: 0x30000019
         MAC:  2.0.0.0
         PHY:  2.2.0.3
         NWP:  3.1.0.0
         ROM:  0
         HOST: 2.0.1.14
         MAC address: 70:ff:76:1c:2a:01

        ============================================

================================================================================
Available commands:

help                scan                setpolicy           wlanconnect
wlan_ap_start       wlandisconnect      ping                send
recv                createfilter        enablefilter        disablefilter
deletefilter        enablewowlan        mdnsadvertise       mdnsquery
radiotool           p2pstart            p2pstop             clear

================================================================================

user@CC3220:
```

Figure 23. network_terminal Shell Command

5.5 Setting Up IAR Embedded Workbench® for ARM

5.5.1 IAR Installation

Download and install IAR Embedded Workbench for ARM Version 7.80.1 or later.

Ensure to select TI XDS as one of the debug probe drivers. Users may also select other drivers. The driver is installed toward the end of the installation of IAR.

During the installation process, users are prompted to install multiple additional software and drivers. Follow the instructions on the screen to finish the installation.

Installation may take a while to complete, depending on the number of selections and network speed, because IAR must download these files. When installation is complete, start IAR and enter your license information.

5.5.2 Post IAR Installation Setup

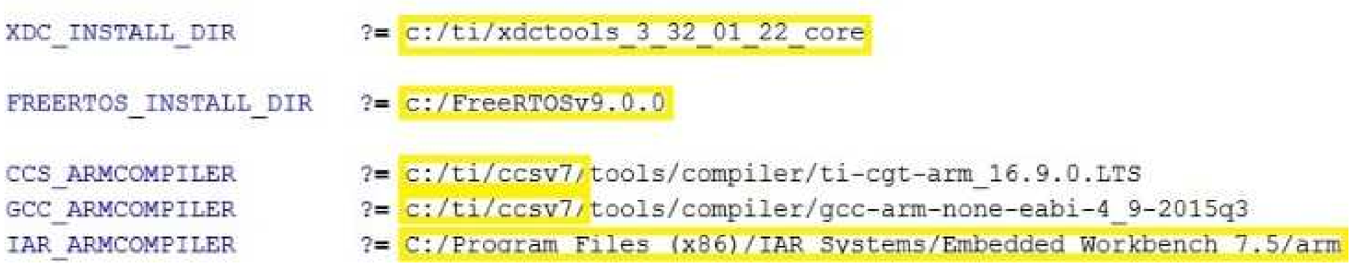
Once the IAR is installed and before starting to work with any of the projects in the SDK, you need to load a set of environment variables that will be set for your current workspace.

1. Browse to Tools → Configure Custom Argument Variables.
2. Go to the Global tab and click *Import*.
3. Choose the argvars which are stored in the .custom_argvars file in the <SDK install dir>\tools\iar directory of the SDK.
4. Edit the argvars to match your settings.
5. Restart IAR.

5.5.3 Importing the network_terminal Example in IAR

Before importing any example to IAR, building the kernel is required.

First, installation paths must be updated. Edit imports.mak on the top-level directory of the MSP432 SDK and adjust the locations of the compiler tools as well as the FreeRTOS and XDCtools installation paths (see Figure 24). XDCTools installation is included in the SimpleLink MSP432 Wi-Fi plug-in.



```
XDC_INSTALL_DIR          ?= c:/ti/xdctools_3_32_01_22_core

FREERTOS_INSTALL_DIR     ?= c:/FreeRTOSv9.0.0

CCS_ARMCOMPILER          ?= c:/ti/ccsv7/tools/compiler/ti-cgt-arm_16.9.0.LTS
GCC_ARMCOMPILER          ?= c:/ti/ccsv7/tools/compiler/gcc-arm-none-eabi-4_9-2015q3
IAR_ARMCOMPILER          ?= C:/Program Files (x86)/IAR Systems/Embedded Workbench 7.5/arm
```

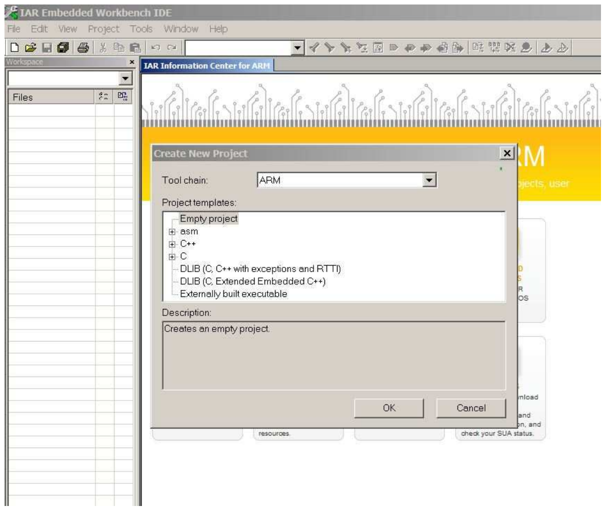
Figure 24. IAR Update Variables

To build the kernel, follow this procedure:

1. To work with FreeRTOS, copy and replace FreeRTOSConfig.h from the Wi-Fi plug-in <SDK install dir\kernel\freertos\builds\MSP_EXP432P401R\release\FreeRTOSConfig.h> into the <MSP432 SDK install dir\kernel\freertos\builds\MSP_EXP432P401R\release\FreeRTOSConfig.h> (TI recommends saving the original FreeRTOSConfig.h file).
2. To build all the TI-RTOS and FreeRTOS config projects, open the command shell at <MSP432 SDK install dir\kernel and execute c:/ti/xdctools_3_32_01_22_core/gmake.exe.

To import the network_terminal example into IAR, follow this procedure:

1. Create a new empty project by browsing to Project → Create New Project, choose the ARM toolchain and an empty project, then click *OK* (see Figure 25).



IAR Embedded Workbench IDE
File Edit View Project Tools Window Help
Workspace
Files
IAR Information Center for ARM
Create New Project
OK
Cancel

Tool chain: ARM

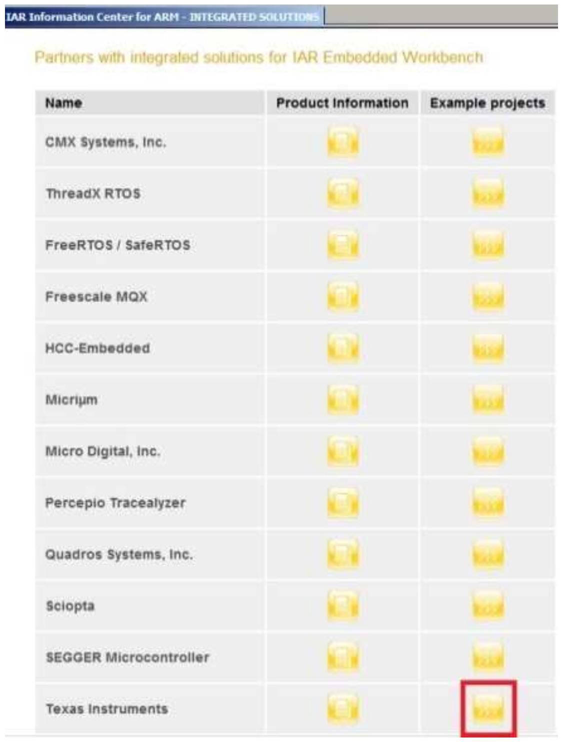
Project templates:
- Empty project
- asm
- C++
- C
- DLIB (C, C++ with exceptions and RTTI)
- DLIB (C, Extended Embedded C++)
- Externally built executable

Description:
Creates an empty project.

Figure 25. IAR Create New Project

2. Browse to the location where you want to save this project, and type a filename for the project file (*.ewp). Click *Save*.

3. Choose Help → IAR Information Center for ARM, choose INTEGRATED SOLUTIONS, and then scroll down and choose Texas Instruments – Example projects (see Figure 26).

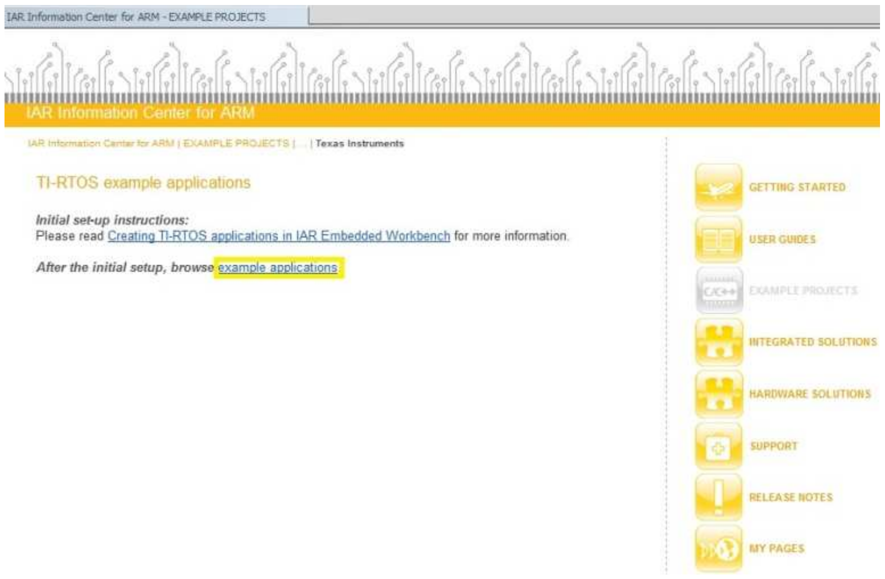

IAR Information Center for ARM - INTEGRATED SOLUTIONS

Partners with integrated solutions for IAR Embedded Workbench

Name	Product Information	Example projects
CMX Systems, Inc.		
ThreadX RTOS		
FreeRTOS / SafeRTOS		
Freescale MQX		
HCC-Embedded		
Micriμm		
Micro Digital, Inc.		
Percepio Tracealyzer		
Quadros Systems, Inc.		
Sciopta		
SEGGER Microcontroller		
Texas Instruments		

Figure 26. IAR TI Example Project

4. Click on the *example applications* link (see Figure 27). This link refers to the EXAMPLE_ROOT link in argvars.


IAR Information Center for ARM - EXAMPLE PROJECTS

IAR Information Center for ARM

IAR Information Center for ARM | EXAMPLE PROJECTS | ... | Texas Instruments

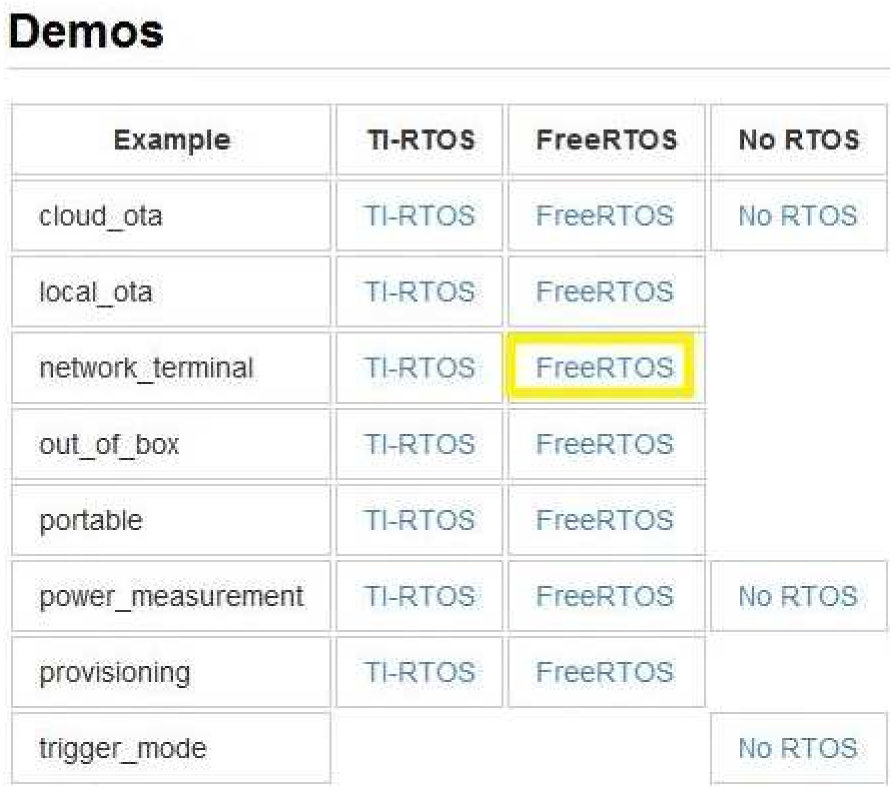
TI-RTOS example applications

Initial set-up instructions:
Please read Creating TI-RTOS applications in IAR Embedded Workbench for more information.

After the initial setup, browse example applications

Figure 27. IAR Example Application Link

5. Click on an example according to the type of device and rtos/nortos flavor (see Figure 28). All project files should be imported to the workspace.



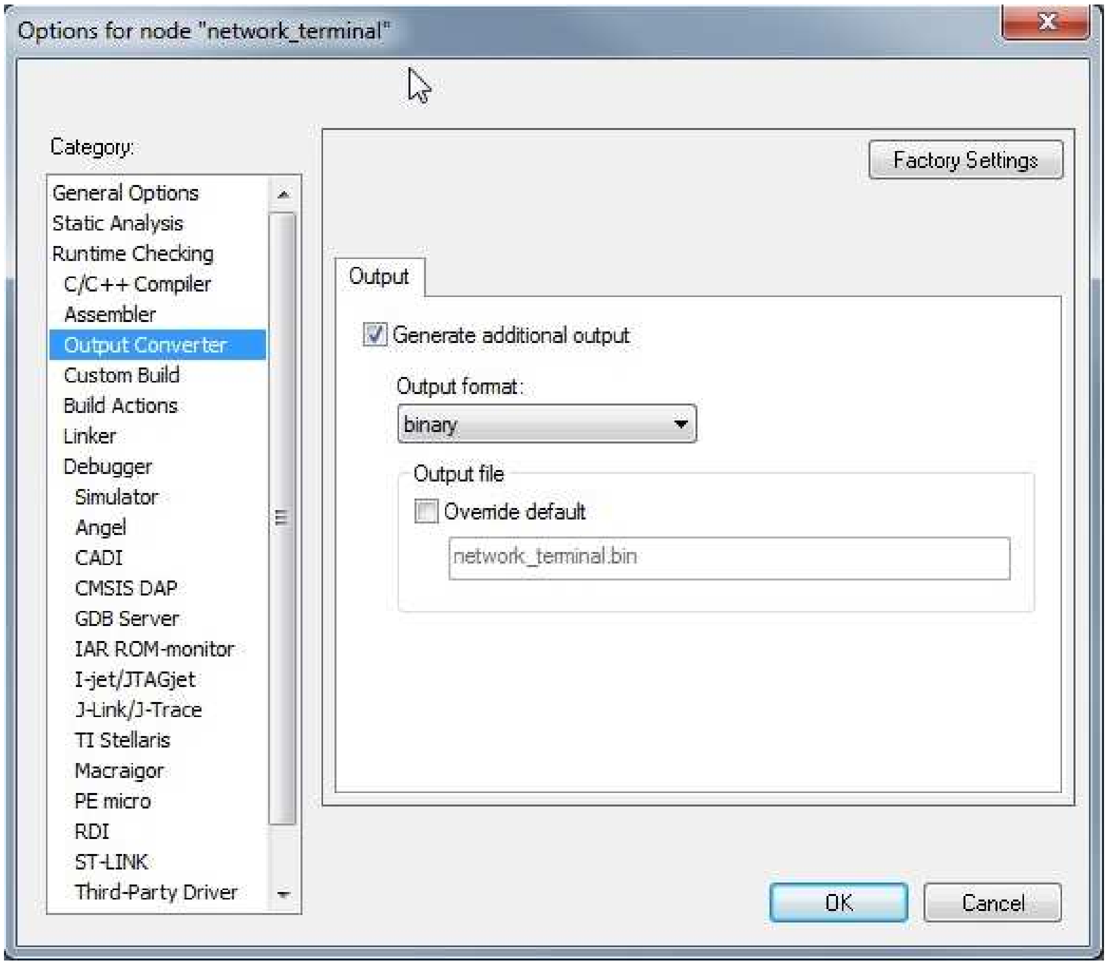
Demos

Example	TI-RTOS	FreeRTOS	No RTOS
cloud_ota	TI-RTOS	FreeRTOS	No RTOS
local_ota	TI-RTOS	FreeRTOS	
network_terminal	TI-RTOS	FreeRTOS	
out_of_box	TI-RTOS	FreeRTOS	
portable	TI-RTOS	FreeRTOS	
power_measurement	TI-RTOS	FreeRTOS	No RTOS
provisioning	TI-RTOS	FreeRTOS	
trigger_mode			No RTOS

Figure 28. IAR Choose network_terminal Example

6. Save the workspace.
7. Browse to the location where you want to save this workspace, and type a filename for the workspace file (*.eww). Click *Save*.
8. Make your changes and rebuild the project.
9. To create a binary image in addition to the *.out file (in case the MCU image is required as part of UniFlash image), a post build step is also required. Right-click the project and choose Options.

10. On the Output Converter tab, select the Generate additional output checkbox, and change the output format to binary (see Figure 29).



[✓] Generate additional output
Output format: binary
Output file
[ ] Override default
network_terminal.bin

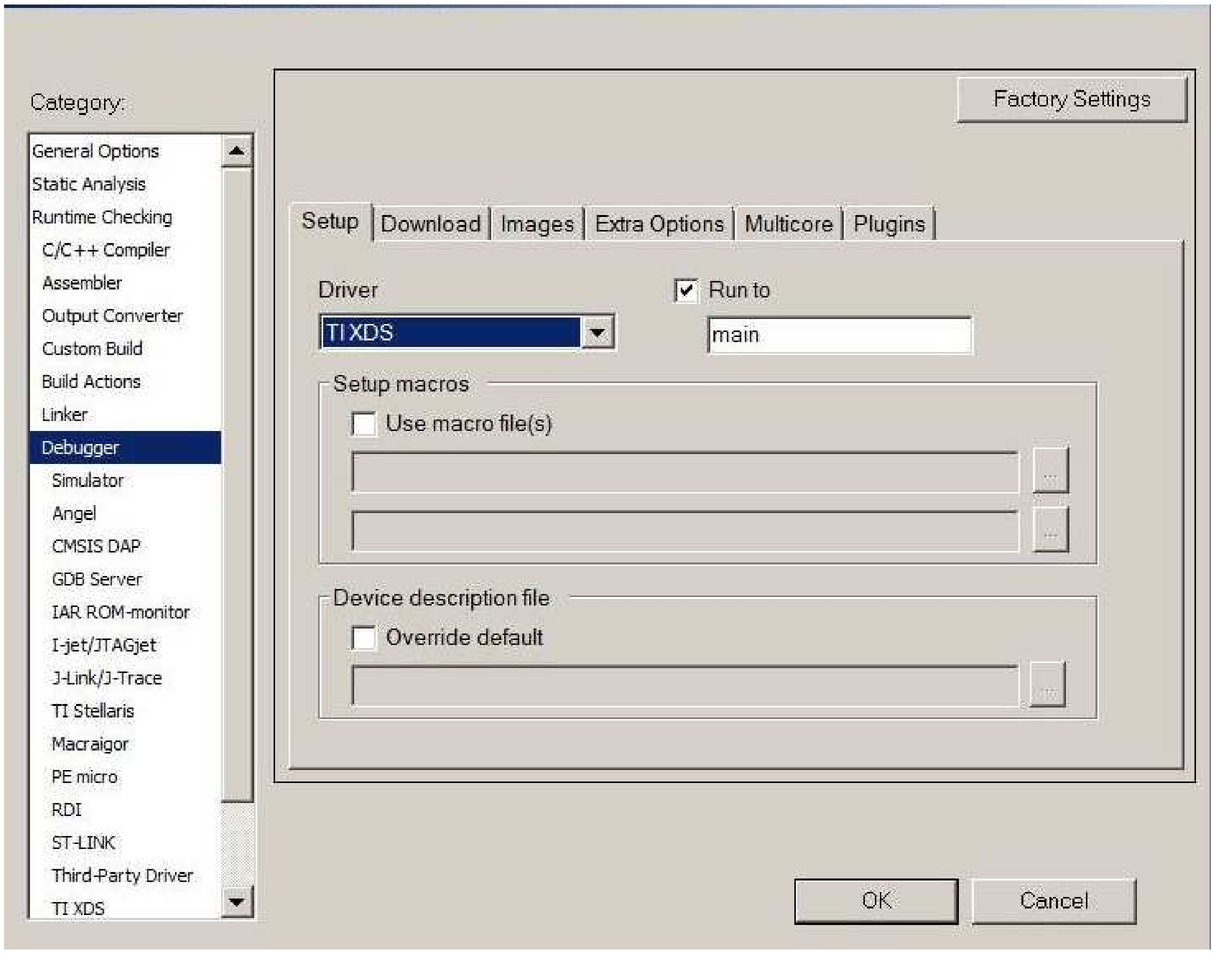
Figure 29. IR Generate Binary Output

11. Recompile and the binary should reside in Debug/Exe.

5.5.4 Compiling and Debugging the Project in IAR

For the debug session, complete the following steps:

1. Select Project → options from the menu, and select the Debugger category. In the Setup tab, choose TI XDS as the driver and click *OK* (see Figure 30).



Driver: TI XDS

[✓] Run to: main

Setup macros

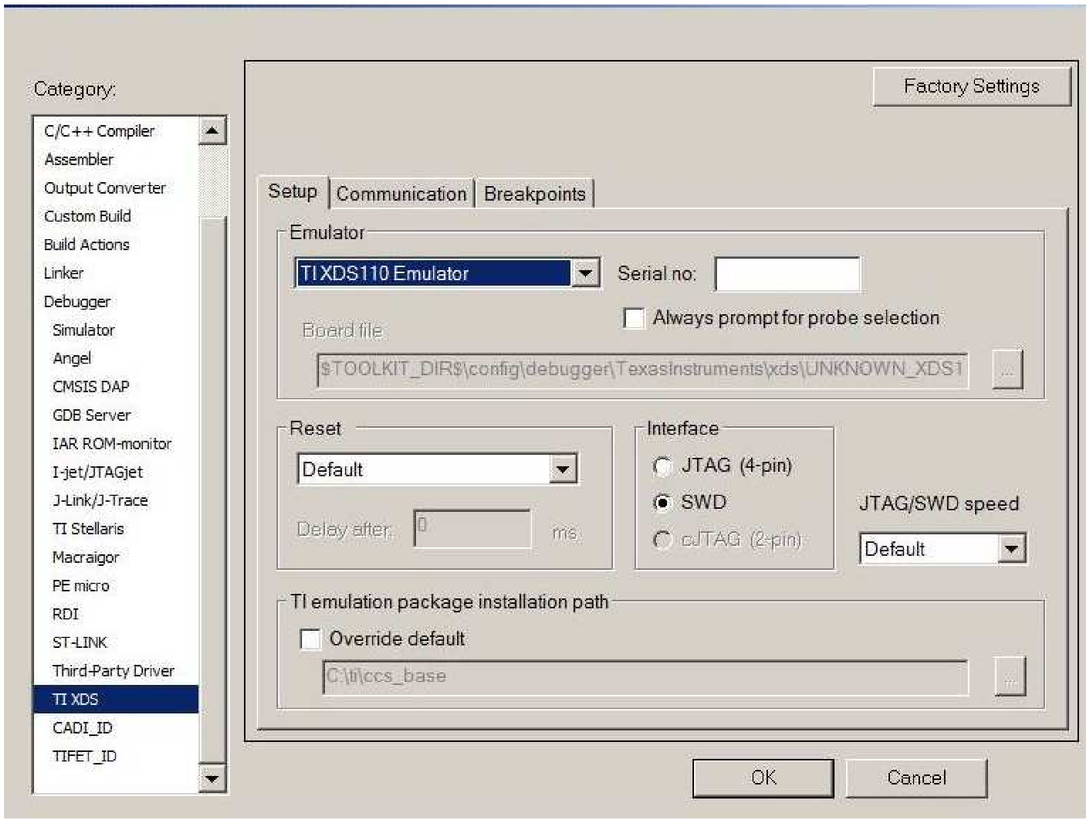
[ ] Use macro file(s)

Device description file

[ ] Override default

Figure 30. IAR TI XDS Debugger

2. Go to the TI XDS category, choose TI XDS110 Emulator as the emulator and JTAG (4-pin) as the interface (see Figure 31).



Emulator: TI XDS110 Emulator

Board file: $TOOLKIT_DIR$\config\debugger\TexasInstruments\xds\UNKNOWN_XDS1

Reset: Default

Delay after: 0 ms

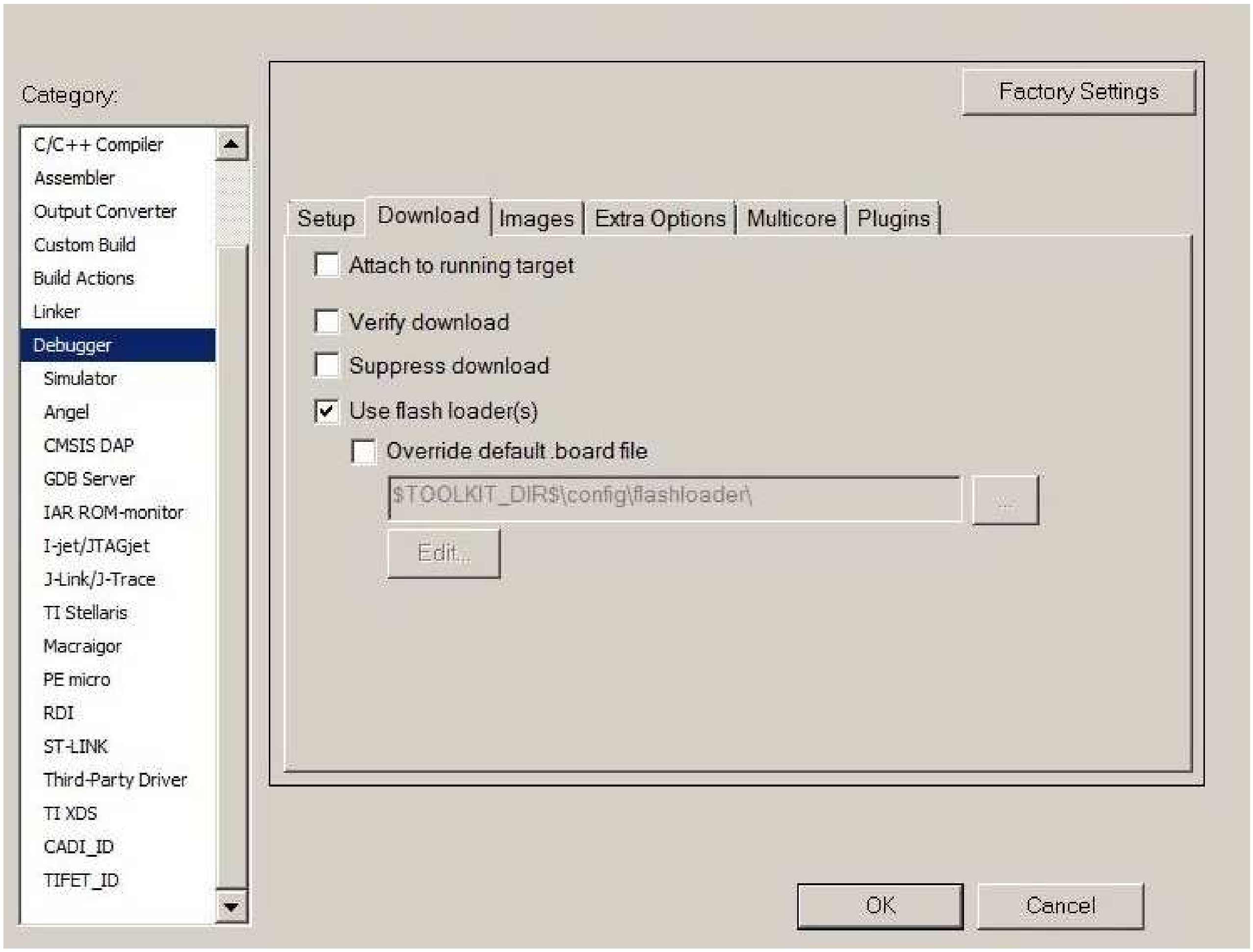
Interface: JTAG (4-pin) [✓ SWD] cJTAG (2-pin)

JTAG/SWD speed: Default

TI emulation package installation path: C:\ti\ccs_base

Figure 31. IAR XDS110 Debugger

3. Check the Use flash loader(s) checkbox in the Debugger → Download tab (see Figure 32).



- [ ] Attach to running target
- [ ] Verify download
- [ ] Suppress download
- [✓] Use flash loader(s)
  - [ ] Override default .board file

$TOOLKIT_DIR$\config\flashloader\

Figure 32. IAR Debugger Settings

4. Start debugging by clicking the green arrow on the top of the window to start the debugger.

6 Getting Started With SimpleLink™ Studio

SimpleLink Studio is a Microsoft Windows-based software used to aid in the development of applications designed to work with the SimpleLink Wi-Fi CC31xx family of wireless chips.

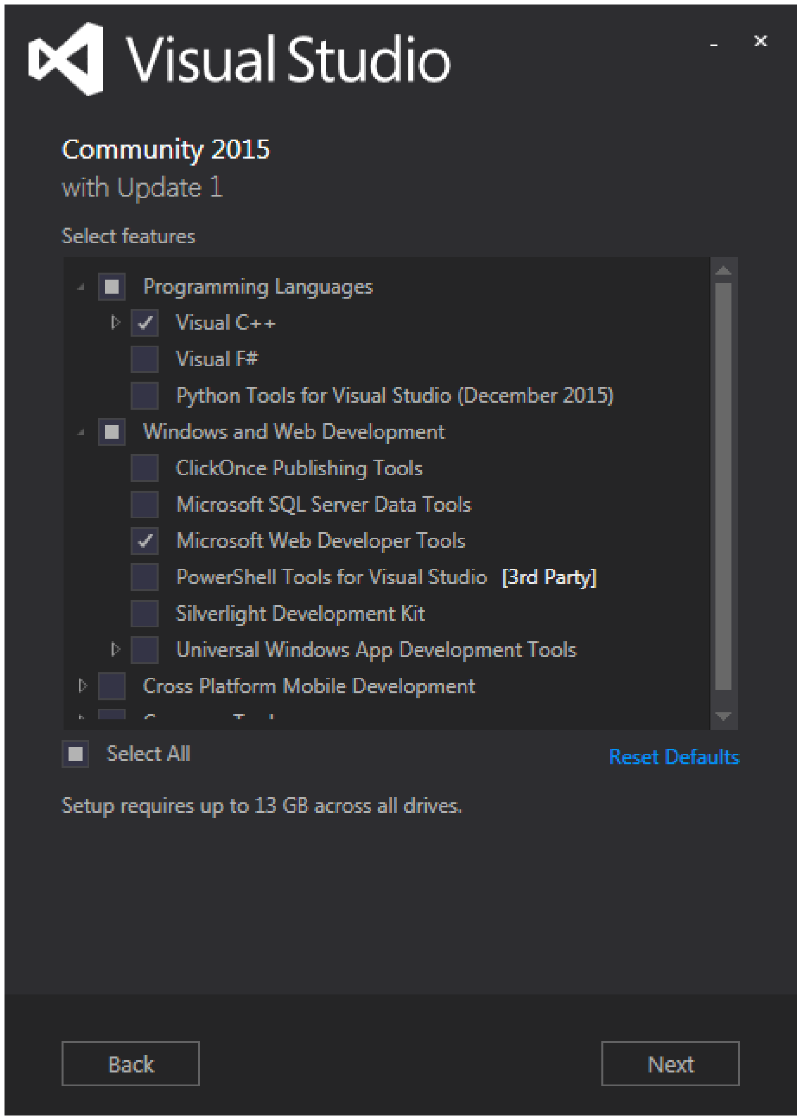
Using SimpleLink Studio, application code can be written and executed in a desktop IDE, such as Visual Studio and Eclipse, and communicate directly with the CC31xx device through the USB cable without a MCU. This process allows for easy testing of code while it is under development, and then later ported to a MCU without needing modification.

TI provides support for SimpleLink with Visual Studio 2015. This software requires the CC31XXEMUBOOST debug board to operate.

6.1 Visual Studio Installation

Download and install Visual Studio. In this tutorial, we are using the Visual Studio 2015 Community Edition.

1. Run the installer to install Visual Studio. On the Customization page, select *Custom*. On the Select Features page, ensure the Visual C++ option under Programming Languages is checked, as shown in Figure 33.


Visual Studio

Community 2015
with Update 1

Select features

- ■ Programming Languages
  - ✓ Visual C++
  - ☐ Visual F#
  - ☐ Python Tools for Visual Studio (December 2015)
- ■ Windows and Web Development
  - ☐ ClickOnce Publishing Tools
  - ☐ Microsoft SQL Server Data Tools
  - ✓ Microsoft Web Developer Tools
  - ☐ PowerShell Tools for Visual Studio [3rd Party]
  - ☐ Silverlight Development Kit
  - ☐ Universal Windows App Development Tools
- ☐ Cross Platform Mobile Development

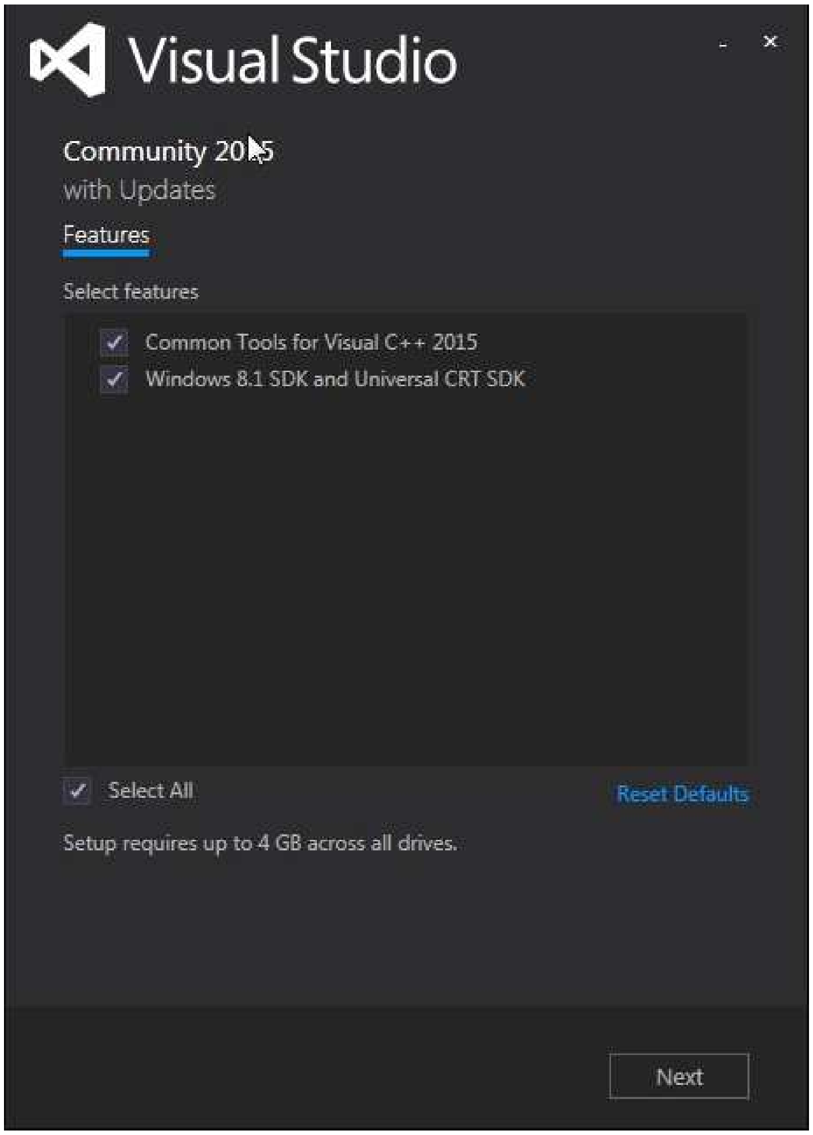
■ Select All

Setup requires up to 13 GB across all drives.

Figure 33. Visual Studio Installation

NOTE: Installation may take a while to complete, depending on the number of selections and network speed, because Visual Studio must download these files.

2. Once installation is complete, restart Visual Studio by clicking the *Restart* button.
3. After restarting, a dialog box requiring an update to additional packages may appear, as shown in Figure 34. Click *Next* and then *Update*.



Select features

- ✓ Common Tools for Visual C++ 2015
- ✓ Windows 8.1 SDK and Universal CRT SDK

✓ Select All Reset Defaults

Setup requires up to 4 GB across all drives.

Figure 34. Visual Studio Updates

6.2 Importing network_terminal Example in Visual Studio

The example can be loaded in two ways:

- Double-click the solution *.sln Launch Visual Studio, located under <SLS install dir>\examples\network_terminal\sl_studio>.
- Open Visual Studio, select File → Open → Project/Solution. Browse to <SLS install dir>\examples\network_terminal\sl_studio> and choose the *.sln.

The SDK provides the Network Terminal application, a Kernel project based on Windows and the Host Driver project.

6.3 Compiling and Debugging the Project in Visual Studio

To compile the project, select Rebuild solution, as shown in Figure 35.

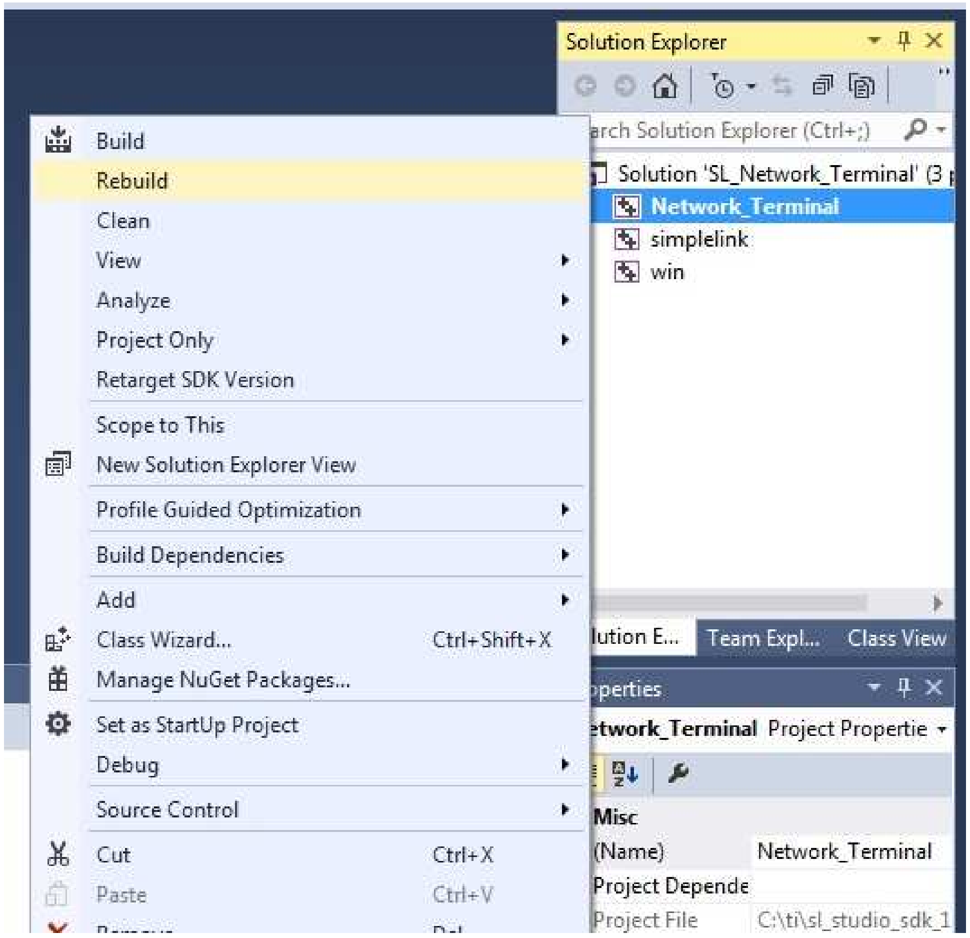
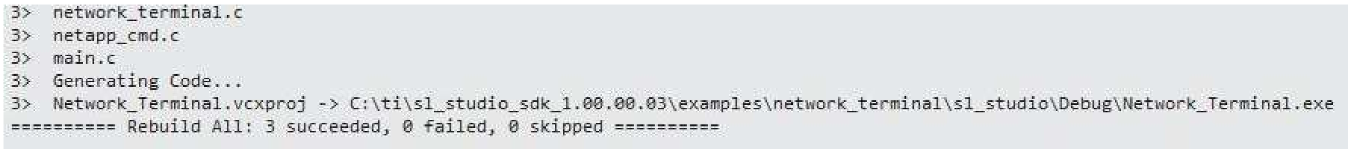


Figure 35. SimpleLink™ Studio Solution

When the build is finished, there will be three succeeded projects (see Figure 36).



```
3>  network_terminal.c
3>  netapp_cmd.c
3>  main.c
3>  Generating Code...
3>  Network_Terminal.vcxproj -> C:\ti\sl_studio_sdk_1.00.00.03\examples\network_terminal\sl_studio\Debug\Network_Terminal.exe
========== Rebuild All: 3 succeeded, 0 failed, 0 skipped ==========
```

Figure 36. SimpleLink™ Studio Compiled Solution

To start debugging, run the debugger (see Figure 37).

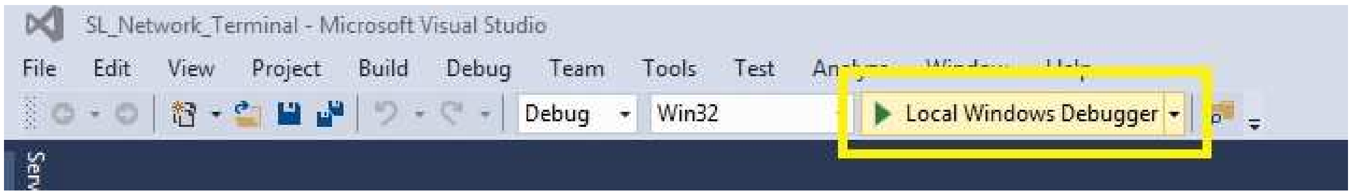


Figure 37. SimpleLink™ Studio Debugger

A command line window opens and you can see the message printing as the application executes (see Figure 23).

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from February 8, 2017 to June 30, 2018 (from * Revision (February 2017) to A Revision) **Page**

- Added BOOSTXL-CC3120MOD Board information 2
- Updated SimpleLink SDK links in Software Installation section. 2
- Added BOOSTXL-CC3120MOD Board With Jumper Configuration image to *Board Configuration* 3
- Added BOOSTXL-CC3120MOD Mounted on Top of CC31XXEMUBOOST image to *Board Configuration* 4
- Updated Getting Started With SimpleLink™ Studio section. 28