CC3200 SimpleLink™ Wi-Fi® and IoT Solution With MCU LaunchPad™ Getting Started Guide

User's Guide

Literature Number: SWRU376A
June 2014–Revised June 2014

Contents

1 **Introduction** **4**

1.1 Prerequisites 4

2 **Getting Started** **5**

2.1 Download and Install Software 5

2.2 Configure Board 5

2.3 Install USB Driver 5

3 **Compile, Download, and Debug** **9**

3.1 Option 1: Code Composer Studio (CCS) 9

3.2 Option 2: IAR Workbench 17

3.3 Option 3: GCC 22

4 **Summary** **28**

5 **Acronyms Used** **29**

Revision History **30**

List of Figures

1 Jumpers on the CC3200-LAUNCHXL 5
2 Windows Device Manager 6
3 Update Driver Software 6
4 Browse for Driver Software 7
5 USB Serial Port 7
6 Device Manager 8
7 Code Composer Studio v6 Setup 9
8 CCS App Center 10
9 Select CCS Projects to Import 11
10 Select CCS Projects to Import 12
11 Properties for *ti_rtos_config* 13
12 Select *simplelink* Project 14
13 Editing *main.c* 15
14 Target Configurations 15
15 Import Target Configuration 16
16 Set as Default 16
17 Debug *wlan_station* 17
18 IAR Embedded Workbench IDE 18
19 Open *simplelink.eww* 19
20 Rebuild the *simplelink* Project. 20
21 Editing *main.c* 21
22 Select TI Stellaris Driver 21
23 Debug Icon 22
24 Cygwin Setup 23
25 Cygwin Terminal 24
26 Cygwin Terminal 24
27 Running the Make Command 24
28 Running the Make Install Command 25
29 Editing *main.c* 25
30 Makefile Command 26
31 Output Screen 26
32 Debugging *wlan_station* 27

User's Guide

SWRU376A – June 2014 – Revised June 2014

CC3200 SimpleLink™ Wi-Fi® and IoT Solution With MCU LaunchPad™ Getting Started Guide

This guide is intended to assist users in the initial setup and demonstration of the *Getting Started with WLAN Station* application. The guide explains how to install an Integrated Development Environment (IDE), and then compile, download and debug *Getting Started with WLAN Station*.

1 Introduction

1.1 Prerequisites

The user should have the following items:

- One CC3200-LAUNCHXL
- An 802.11b/g/n (2.4 GHz) Wireless Access Point (AP).
- A computer running the Microsoft® Windows® 7 or XP operating systems.

SimpleLink, LaunchPad are trademarks of Texas Instruments.
Wi-Fi is a registered trademark of WiFi Alliance.

2 Getting Started

2.1 Download and Install Software

Download and install the following software:

- CC3200 SDK package.
  - This guide assumes the use of the default installation folder *C:\TI\CC3200SDK\*.

2.2 Configure Board

The jumpers on the CC3200-LAUNCHXL should be connected as shown in Figure 1. It may be necessary to move a jumper from P58-VCC to SOP2.

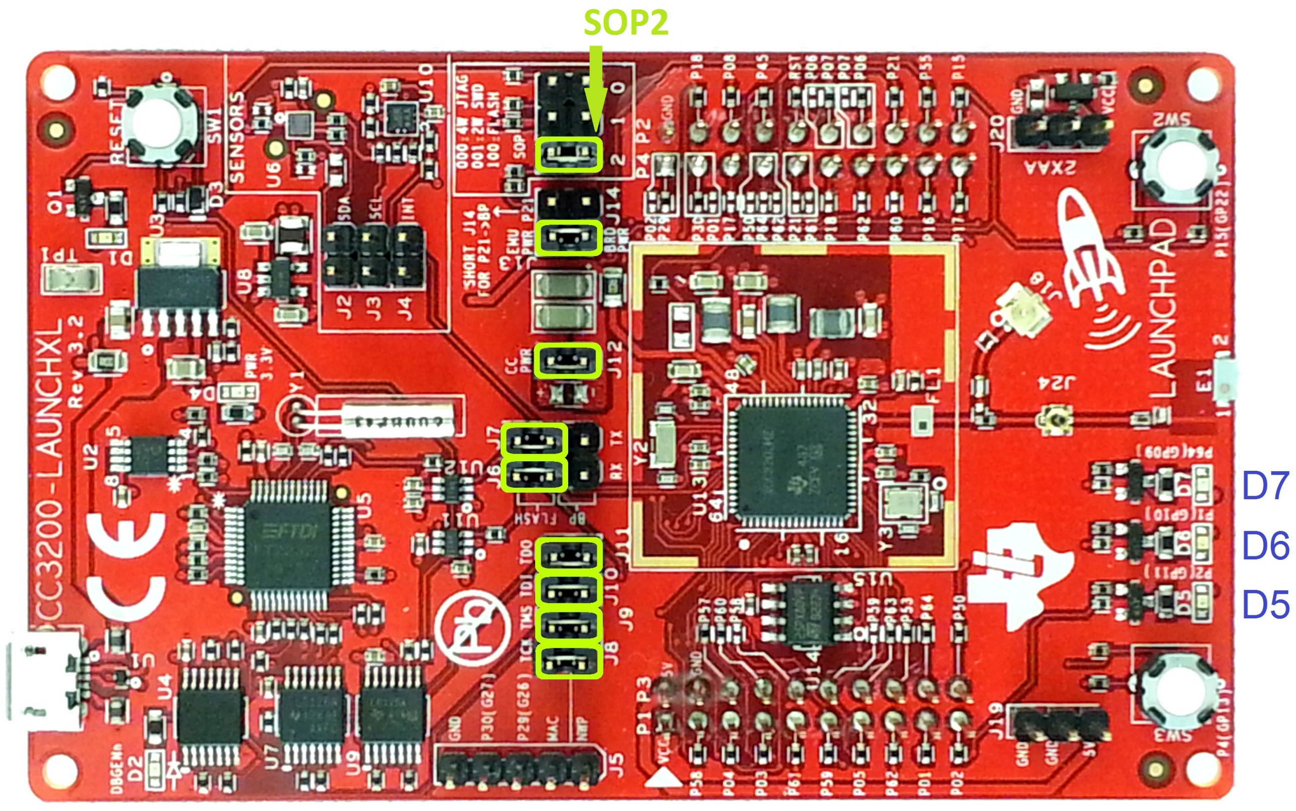


Figure 1. Jumpers on the CC3200-LAUNCHXL

2.3 Install USB Driver

1. Connect the CC3200-LAUNCHXL to the PC using the provided micro-USB cable.
2. Open the Windows Device Manager by selecting *Start Menu>Control Panel>Device Manager*. The CC3200-LAUNCHXL will appear as two instances of “USB <-> JTAG/SWD” under the category *Other Devices* as shown in Figure 2. For both of these instances, the driver software will need to be updated.

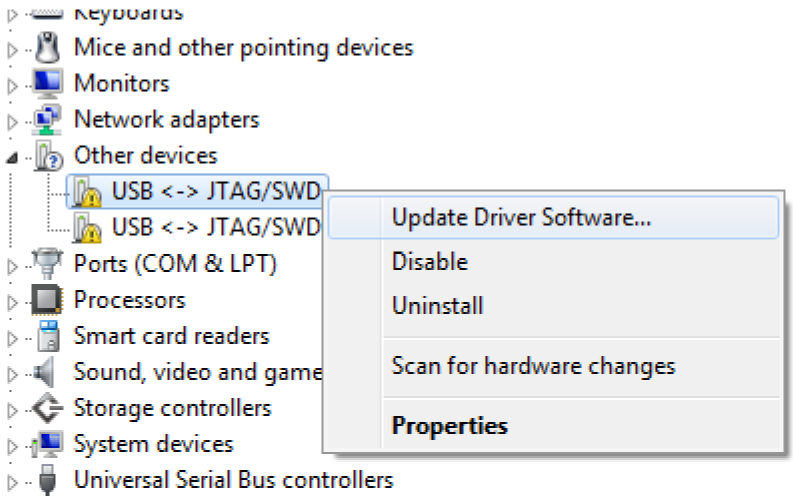
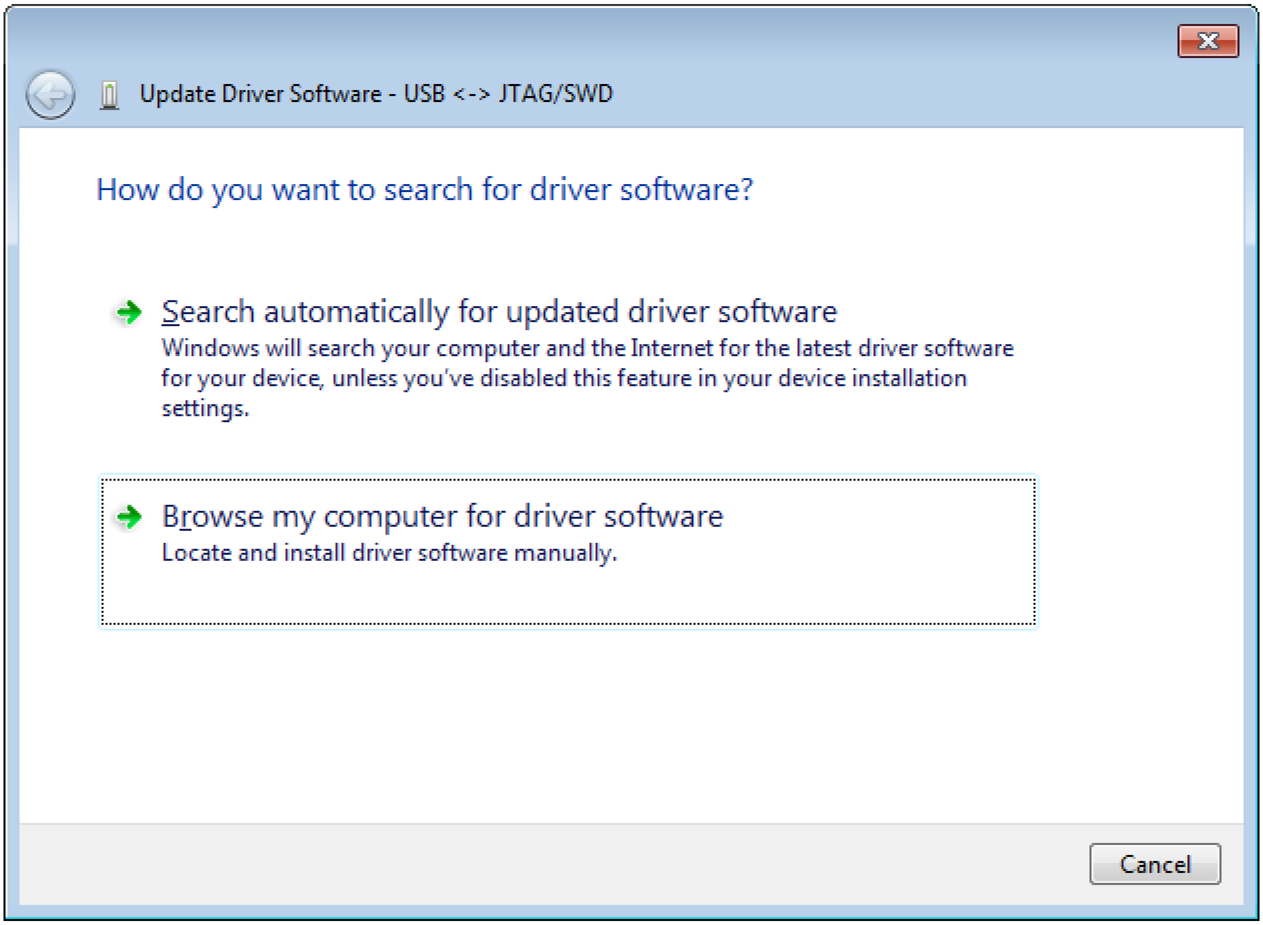


Figure 2. Windows Device Manager

3. Right click on the first instance of “USB <-> JTAG/SWD” and select “Update Driver Software…”
4. Select “Browse my computer for driver software.”



How do you want to search for driver software?

Search automatically for updated driver software

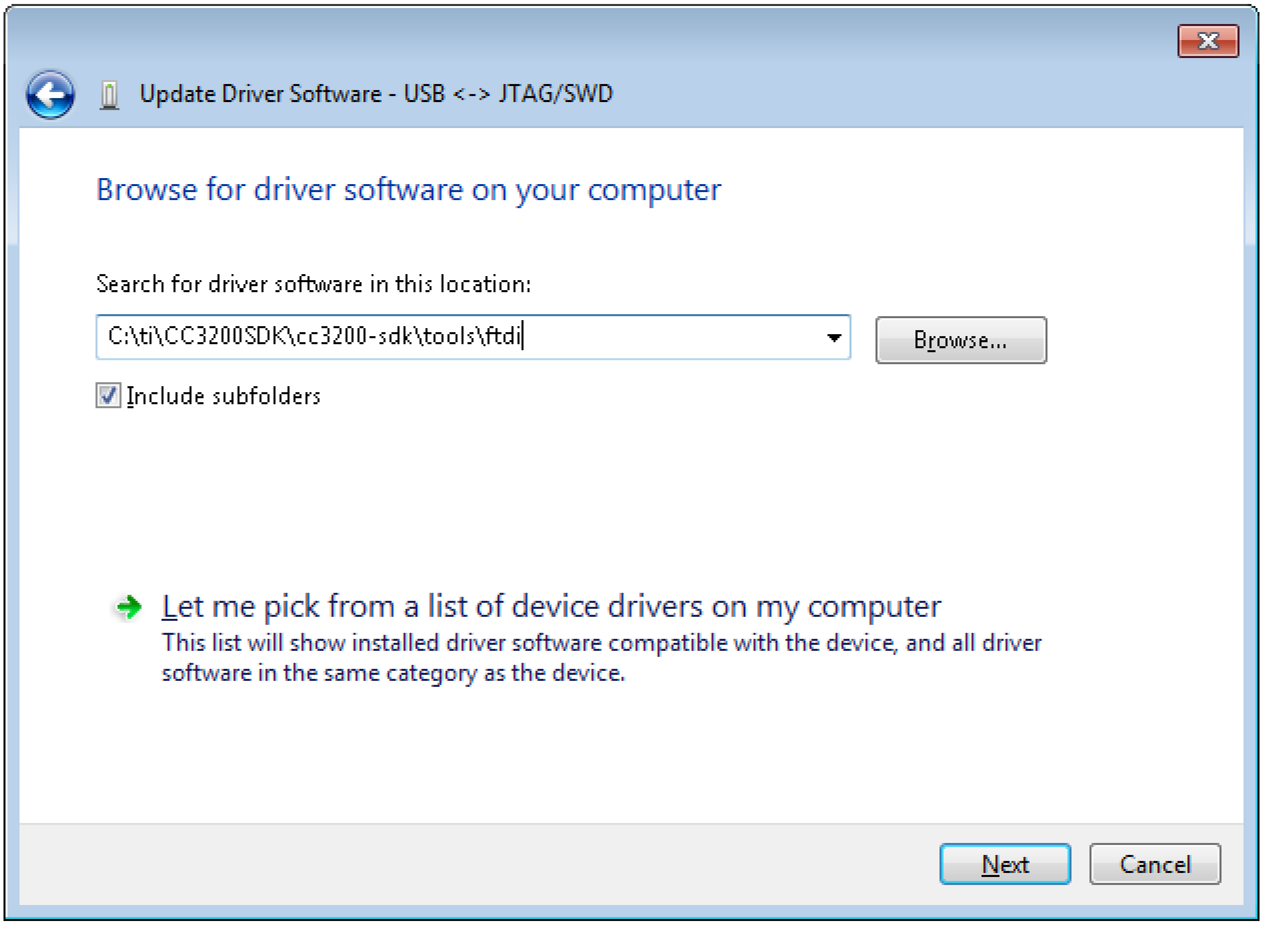
Windows will search your computer and the Internet for the latest driver software for your device, unless you've disabled this feature in your device installation settings.

Browse my computer for driver software

Locate and install driver software manually.

Figure 3. Update Driver Software

5. Fill the search path as *C:\TI\CC3200SDK\cc3200-sdk\tools\ftdi*, and press next. There is no need to restart the PC.



Browse for driver software on your computer

Search for driver software in this location:

C:\ti\CC3200SDK\cc3200-sdk\tools\ftdi

☑ Include subfolders

Let me pick from a list of device drivers on my computer

This list will show installed driver software compatible with the device, and all driver software in the same category as the device.

Figure 4. Browse for Driver Software

6. Repeat the above three steps for the other instance of "USB <-> JTAG/SWD."
7. Repeat the same steps for the instance of “USB Serial Port” that should have appeared as shown in Figure 5.

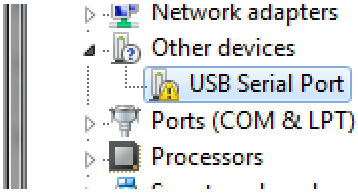
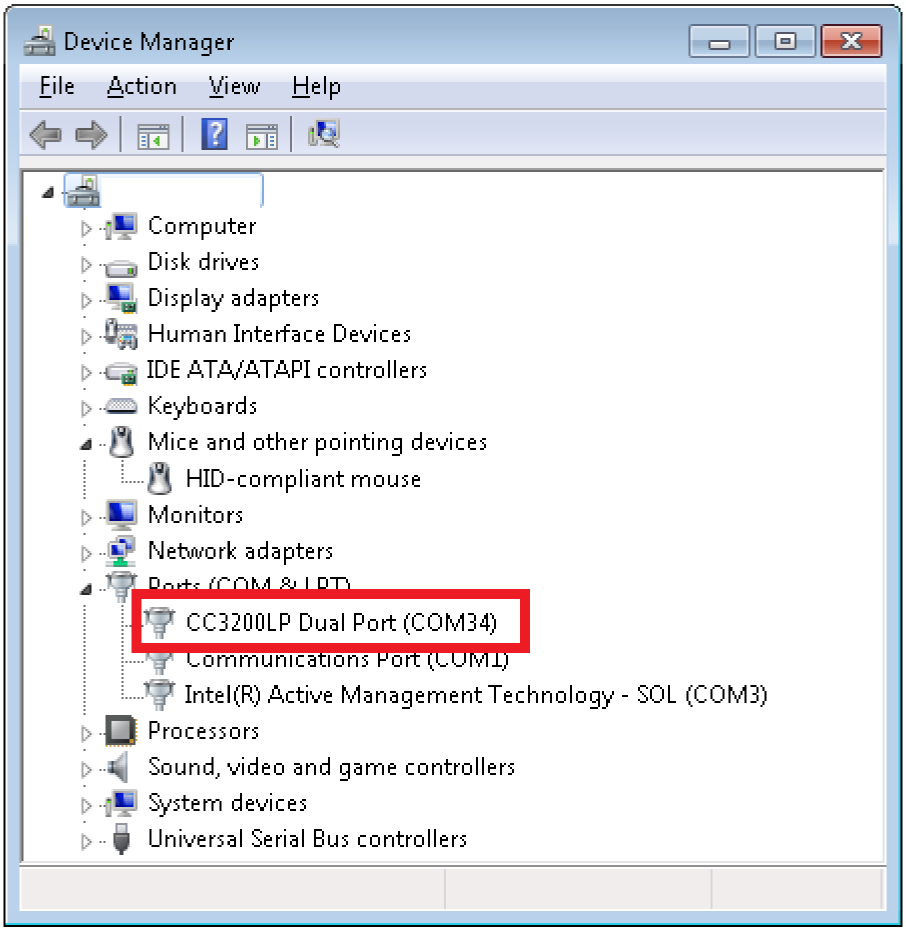


Figure 5. USB Serial Port

8. The CC3200-LAUNCHXL will now be visible in the Device Manager as shown in Figure 6. Note the COM port number that appears.



Computer
Disk drives
Display adapters
Human Interface Devices
IDE ATA/ATAPI controllers
Keyboards
Mice and other pointing devices
HID-compliant mouse
Monitors
Network adapters
Ports (COM & LPT)
CC3200LP Dual Port (COM34)
Communications Port (COM1)
Intel(R) Active Management Technology - SOL (COM3)
Processors
Sound, video and game controllers
System devices
Universal Serial Bus controllers

Figure 6. Device Manager

3 Compile, Download, and Debug

The CC3200 SDK supports CCS 6.0, IAR 7.10.3 and GCC IDE/compiler. The example shown here is *Getting Started with WLAN Station*, and performs the following functions:

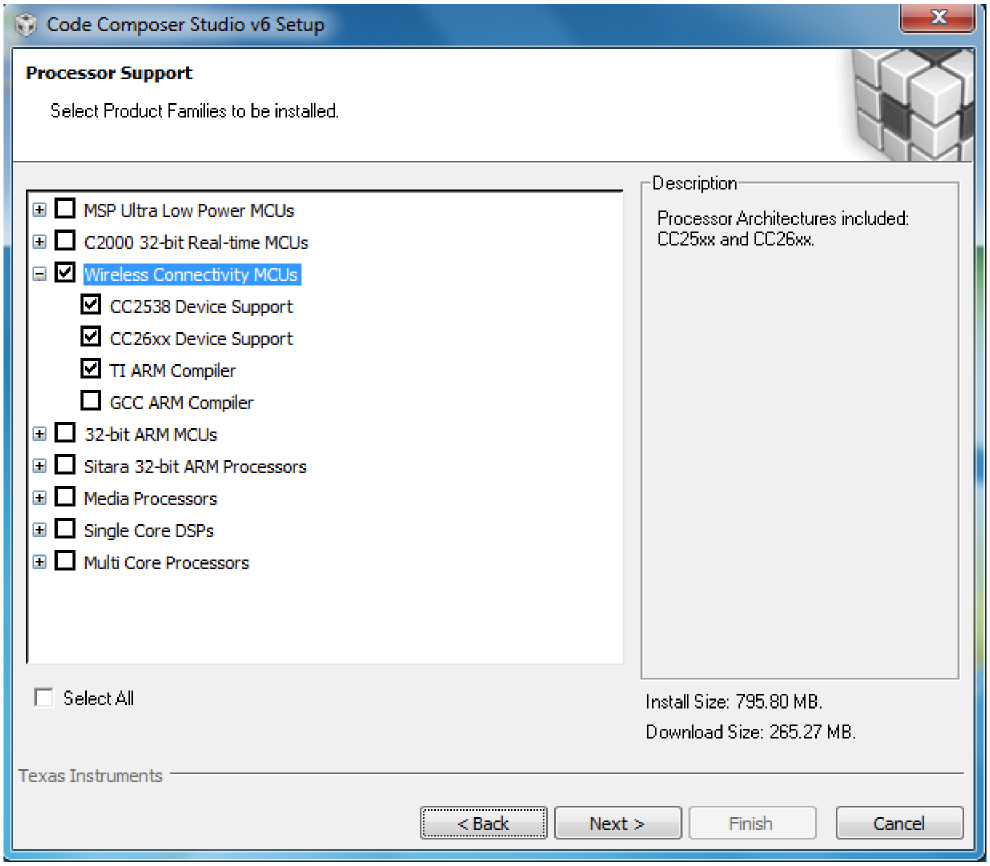
1. Switches to Station mode if the device is in AP mode.
2. Connects to the user's Access Point (default SSID is ‘cc3200demo’). If the connection to the AP is successful, the red LED (D7) will switch on.
3. Pings the user's Access Point. If the ping test is successful, the green LED (D5) will switch on.
4. Pings to www.ti.com to check Internet connectivity. If the ping test is successful, the orange LED (D6) switches on.

This example uses a Real Time Operating System (RTOS).

3.1 Option 1: Code Composer Studio (CCS)

3.1.1 Download and Install

Download and run the Code Composer Studio 6.0 (CCS) installation wizard (*ccs_setup_win32.exe*) from http://processors.wiki.ti.com/index.php/Category:Code_Composer_Studio_v6 The program must be **Version 6.0.0.00190** or later. Select the Wireless Connectivity MCUs option for processor support. The remaining options for the installer should be left as the default. Installation time is typically 20 minutes, but can vary based on internet connection speed.



Code Composer Studio v6 Setup

Processor Support

Select Product Families to be installed.

- ⊞ □ MSP Ultra Low Power MCUs
- ⊞ □ C2000 32-bit Real-time MCUs
- ⊟ ☑ Wireless Connectivity MCUs
  - ☑ CC2538 Device Support
  - ☑ CC26xx Device Support
  - ☑ TI ARM Compiler
  - □ GCC ARM Compiler
- ⊞ □ 32-bit ARM MCUs
- ⊞ □ Sitara 32-bit ARM Processors
- ⊞ □ Media Processors
- ⊞ □ Single Core DSPs
- ⊞ □ Multi Core Processors

□ Select All

Description

Processor Architectures included: CC25xx and CC26xx.

Install Size: 795.80 MB.

Download Size: 265.27 MB.

Texas Instruments

Figure 7. Code Composer Studio v6 Setup

3.1.2 Install TI-RTOS for SimpleLink and CC3200 Support Package

Install TI-RTOS for SimpleLink from the CCS App Center:

1. Start CCS, and choose a Workspace folder (the folder where the projects reside).

2. Open the App Center from the *Help->Getting Started* screen.
3. Search 'CC3200' in the App Center to find 'TI-RTOS for SimpleLink' and 'CC3200 Add-On'
4. Select TI-RTOS
5. Select the CC3200 Add-On
6. Press 'Install Software'

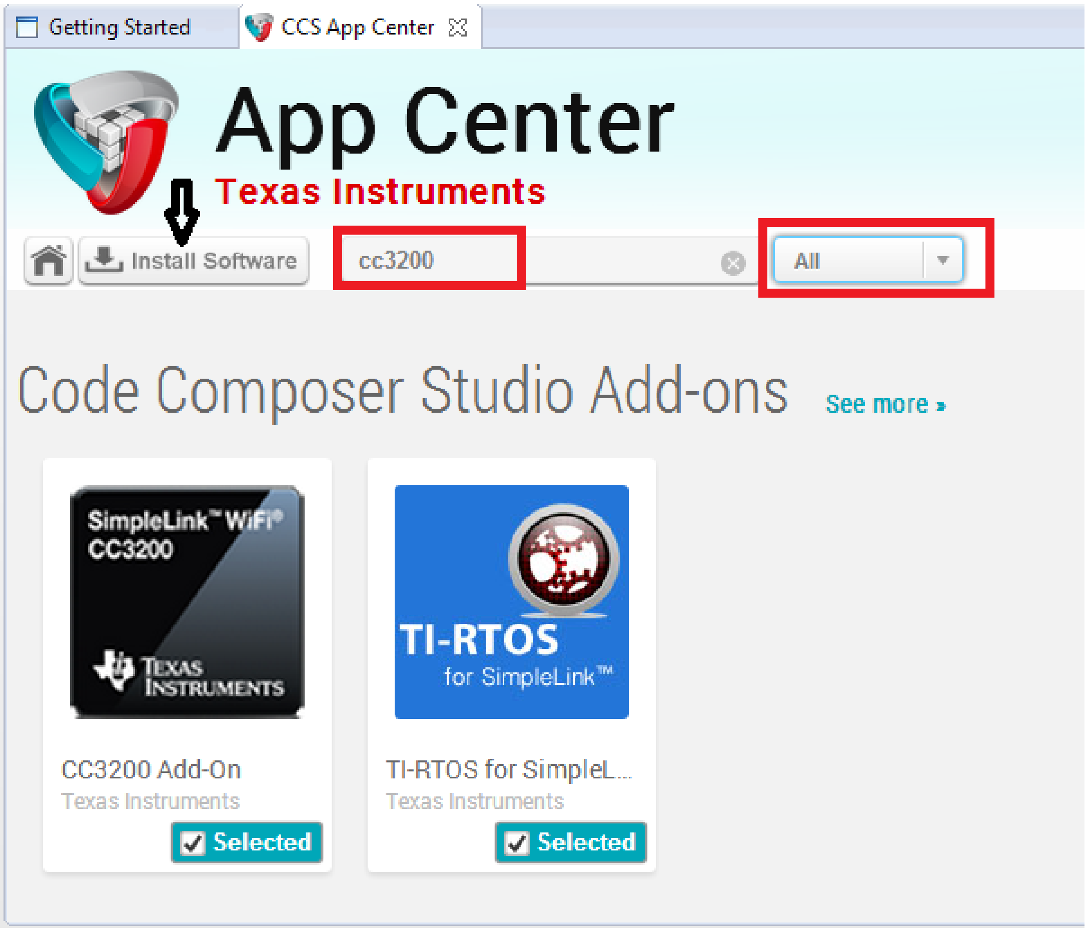
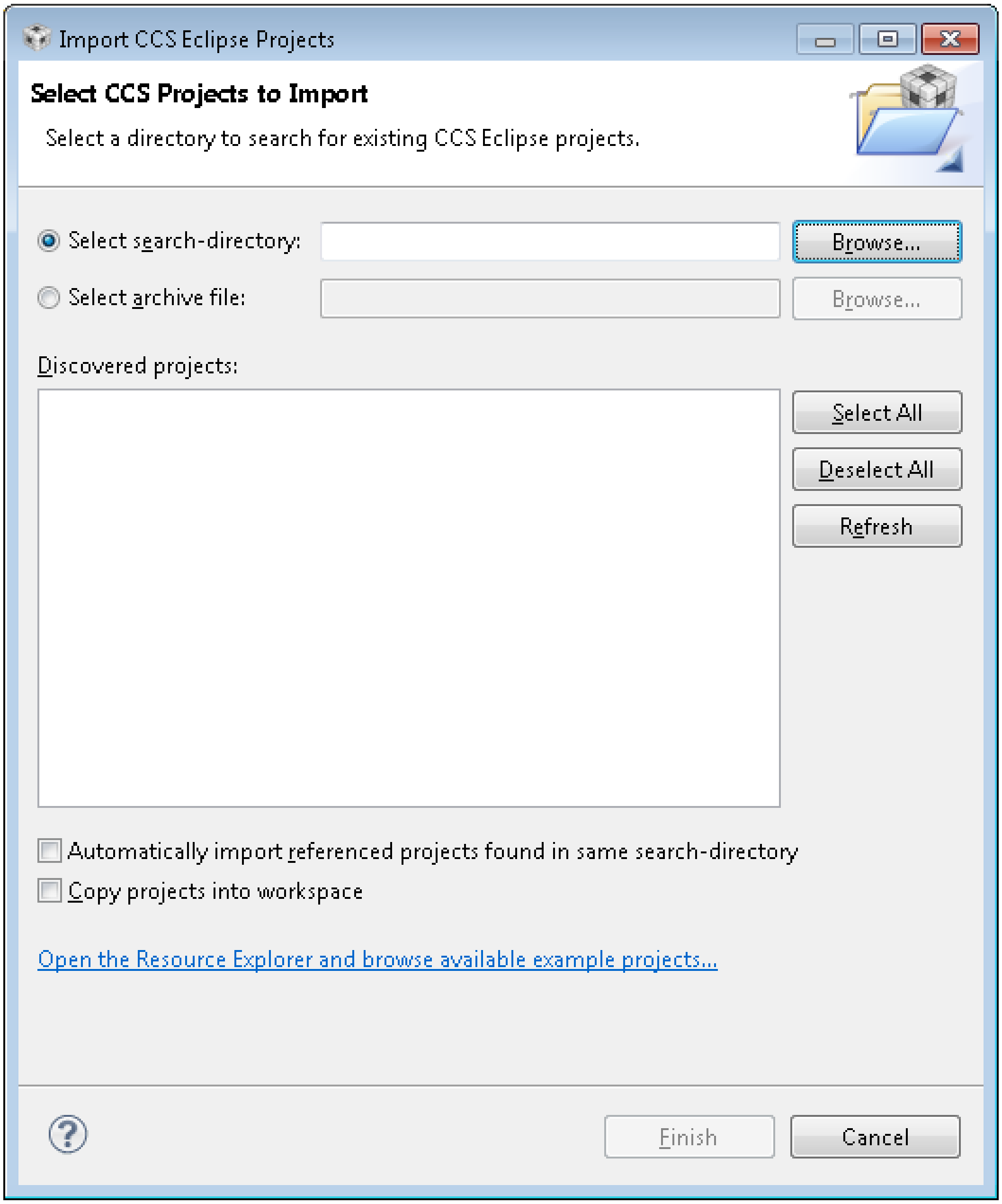


Figure 8. CCS App Center

3.1.3 Import and Configure Project

1. Choose *Projects>Import CCS Projects* from the menu.
2. Select the Browse button in the Import CCS Eclipse Projects dialog, and Select the directory *C:\TI\CC3200SDK\cc3200-sdk*.



Import CCS Eclipse Projects

Select CCS Projects to Import

Select a directory to search for existing CCS Eclipse projects.

● Select search-directory: [ ] Browse...

○ Select archive file: [ ] Browse...

Discovered projects:

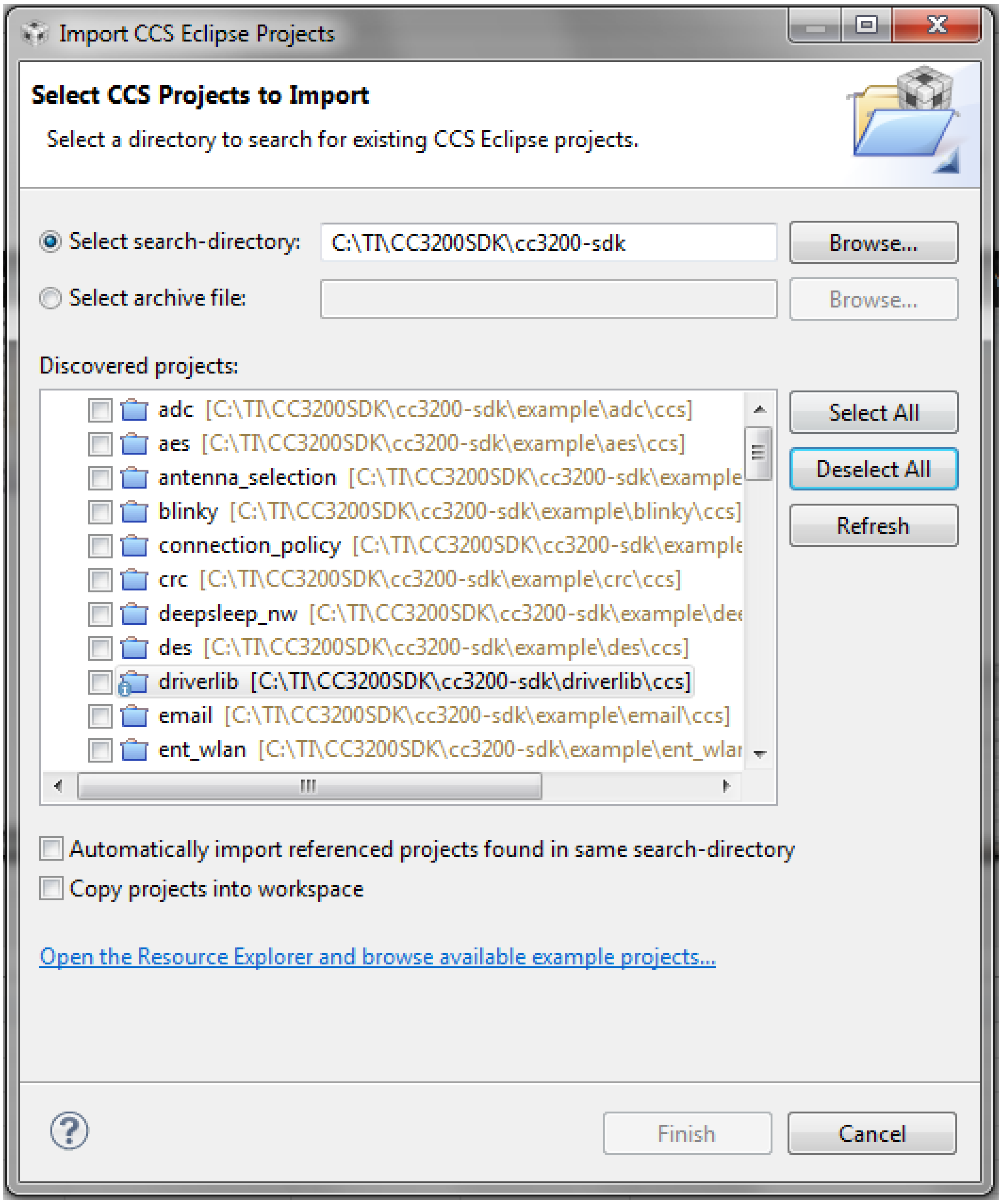
□ Automatically import referenced projects found in same search-directory

□ Copy projects into workspace

Open the Resource Explorer and browse available example projects...

Figure 9. Select CCS Projects to Import

3. Select the *wlan_station*, *driverlib*, *simplelink*, *oslib* and *ti_rtos_config* projects. Click Finish. For this tutorial, do not check the 'Copy projects into workspace' option. This would cause the project's links to it's dependencies to be broken since relative paths are used.



Import CCS Eclipse Projects

Select CCS Projects to Import

Select a directory to search for existing CCS Eclipse projects.

● Select search-directory: C:\TI\CC3200SDK\cc3200-sdk

○ Select archive file:

Discovered projects:

- □ adc [C:\TI\CC3200SDK\cc3200-sdk\example\adc\ccs]
- □ aes [C:\TI\CC3200SDK\cc3200-sdk\example\aes\ccs]
- □ antenna_selection [C:\TI\CC3200SDK\cc3200-sdk\example
- □ blinky [C:\TI\CC3200SDK\cc3200-sdk\example\blinky\ccs]
- □ connection_policy [C:\TI\CC3200SDK\cc3200-sdk\example
- □ crc [C:\TI\CC3200SDK\cc3200-sdk\example\crc\ccs]
- □ deepsleep_nw [C:\TI\CC3200SDK\cc3200-sdk\example\dee
- □ des [C:\TI\CC3200SDK\cc3200-sdk\example\des\ccs]
- □ driverlib [C:\TI\CC3200SDK\cc3200-sdk\driverlib\ccs]
- □ email [C:\TI\CC3200SDK\cc3200-sdk\example\email\ccs]
- □ ent_wlan [C:\TI\CC3200SDK\cc3200-sdk\example\ent_wlar

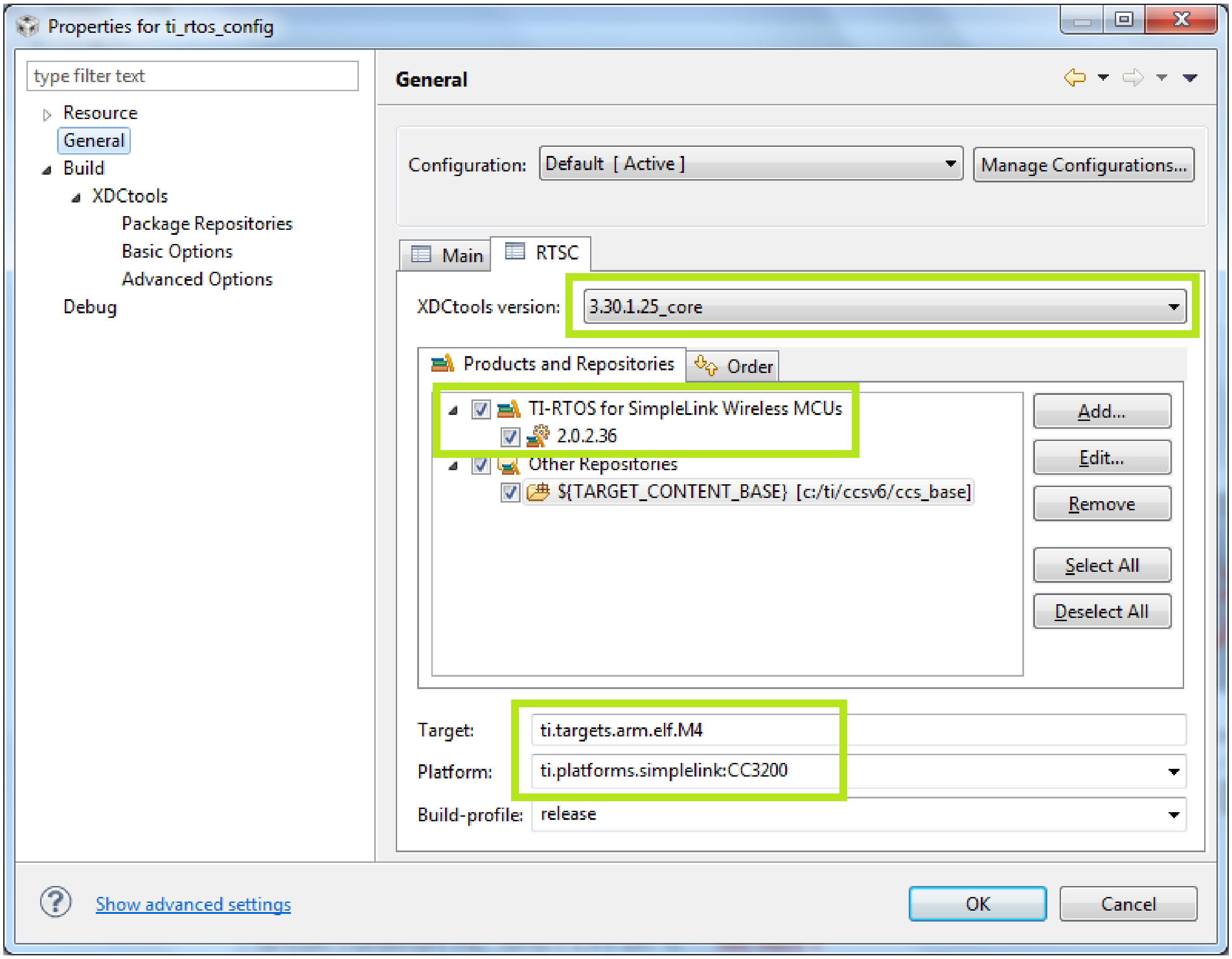
□ Automatically import referenced projects found in same search-directory

□ Copy projects into workspace

Open the Resource Explorer and browse available example projects...

Figure 10. Select CCS Projects to Import

4. Setup the *ti_rtos_config* project configuration as shown in Figure 11. Select the latest versions of XDCtools and TI-RTOS for SimpleLink. Also verify the platform is selected as ti.platforms.simplelink:CC3200.



Configuration: Default [Active]

XDCtools version: 3.30.1.25_core

☑ TI-RTOS for SimpleLink Wireless MCUs
☑ 2.0.2.36
☑ Other Repositories
☑ ${TARGET_CONTENT_BASE} [c:/ti/ccsv6/ccs_base]

Target: ti.targets.arm.elf.M4

Platform: ti.platforms.simplelink:CC3200

Build-profile: release

Figure 11. Properties for *ti_rtos_config*

5. Select the *simplelink* project and build it as shown in Figure 12.

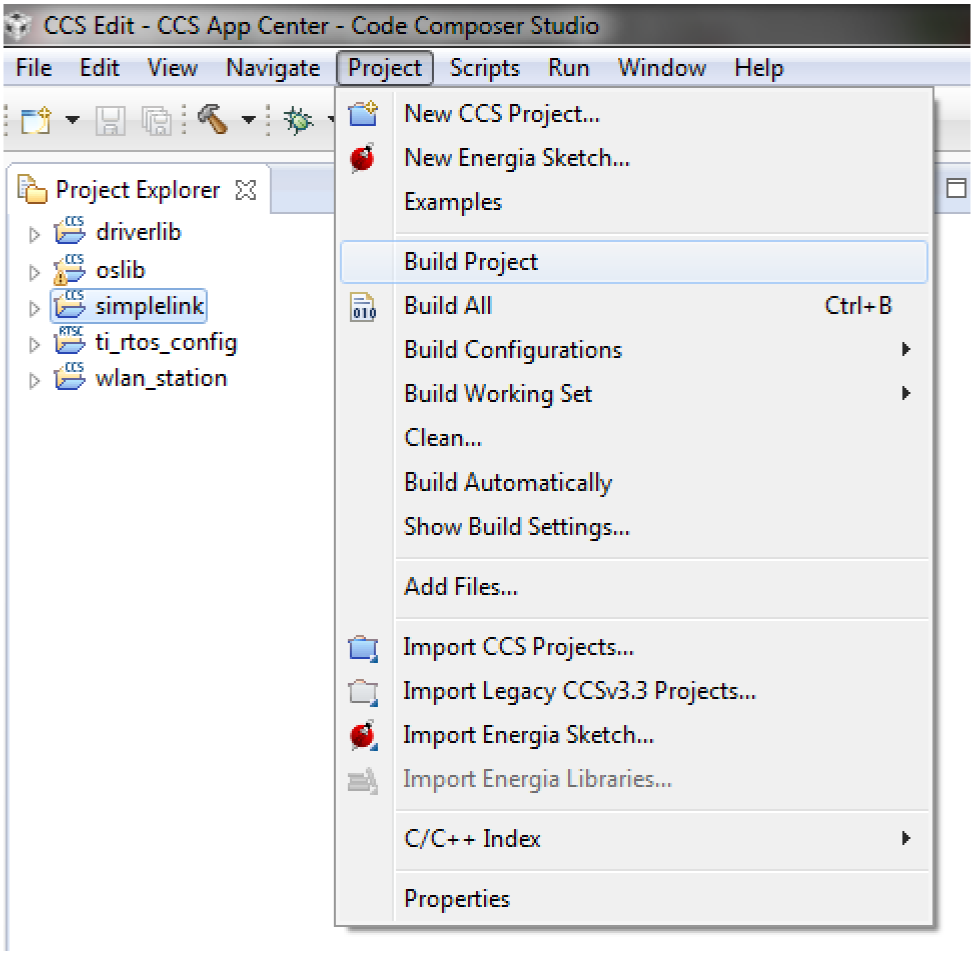
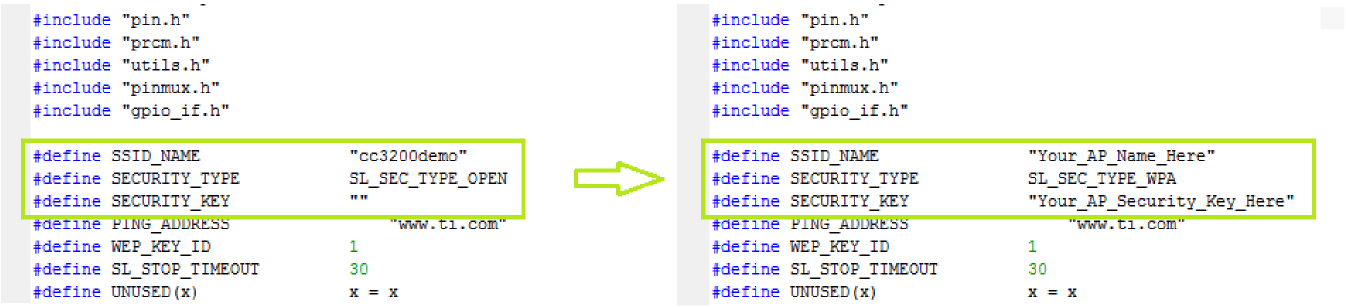


Figure 12. Select *simplelink* Project

6. Select the *ti_rtos_config* project and build it.
7. Select the *driverlib* project and build it.
8. Select the *oslib* project and build it.
9. Open the *main.c* file of the *wlan_station* project for editing at *C:\TI\CC3200SDK\cc3200-sdk\example\getting_started_with_wlan_station\main.c*.
10. Edit *main.c* to use the SSID, security type and security key of the Access Point being used. Edit the macros SSID_NAME, SECURITY_TYPE and SECURITY_KEY to contain the Access Point's information as shown in Figure 13. The security types supported for this demo are WPA/WPA2 and Open. For Open security, define SECURITY_TYPE as SL_SEC_TYPE_OPEN. For WPA and WPA2 security, define it as SL_SEC_TYPE_WPA. Alternatively, the SSID and security of the Access Point being used can be changed to match the default (SSID: cc3200demo, Security: Open).



```
#include "pin.h"
#include "prcm.h"
#include "utils.h"
#include "pinmux.h"
#include "gpio_if.h"

#define SSID_NAME            "cc3200demo"
#define SECURITY_TYPE        SL_SEC_TYPE_OPEN
#define SECURITY_KEY         ""
#define PING_ADDRESS             "www.ti.com"
#define WEP_KEY_ID           1
#define SL_STOP_TIMEOUT      30
#define UNUSED(x)            x = x
```

```
#include "pin.h"
#include "prcm.h"
#include "utils.h"
#include "pinmux.h"
#include "gpio_if.h"

#define SSID_NAME            "Your_AP_Name_Here"
#define SECURITY_TYPE        SL_SEC_TYPE_WPA
#define SECURITY_KEY         "Your_AP_Security_Key_Here"
#define PING_ADDRESS             "www.ti.com"
#define WEP_KEY_ID           1
#define SL_STOP_TIMEOUT      30
#define UNUSED(x)            x = x
```

Figure 13. Editing *main.c*

11. Save *main.c*.
12. Select the *wlan_station* project and build it.
13. The target configuration needs to be set before debugging from CCS. Navigate to *View>Target Configurations*.

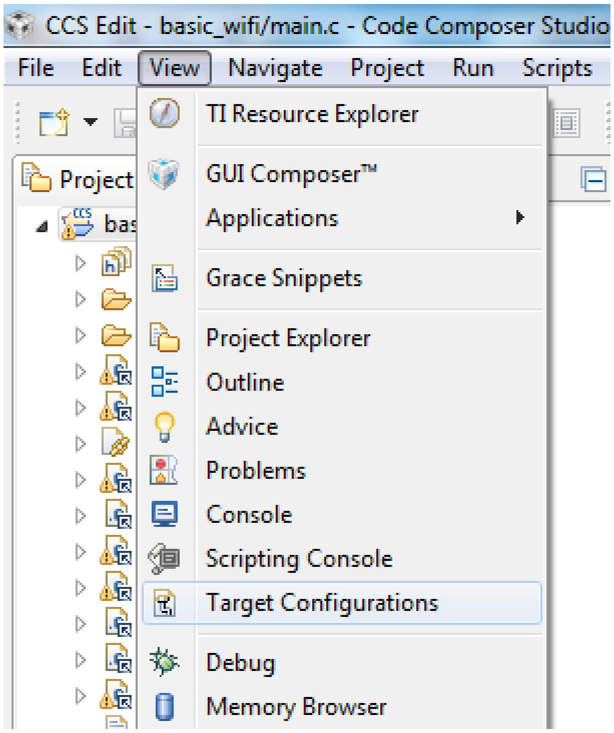


Figure 14. Target Configurations

14. Right Click on “User Defined,” select “Import Target Configuration” and select the file CC3200.ccxml from *C:\TI\CC3200SDK\cc3200-sdk\tools\ccs_patch\*. Select the Copy files option when prompted.

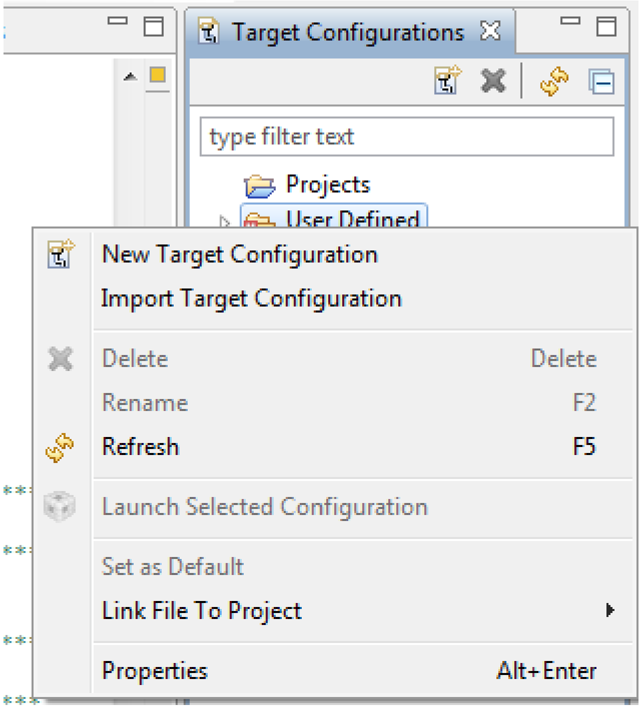


Figure 15. Import Target Configuration

15. Set this new configuration as the default by right clicking on the file name as shown in Figure 16.

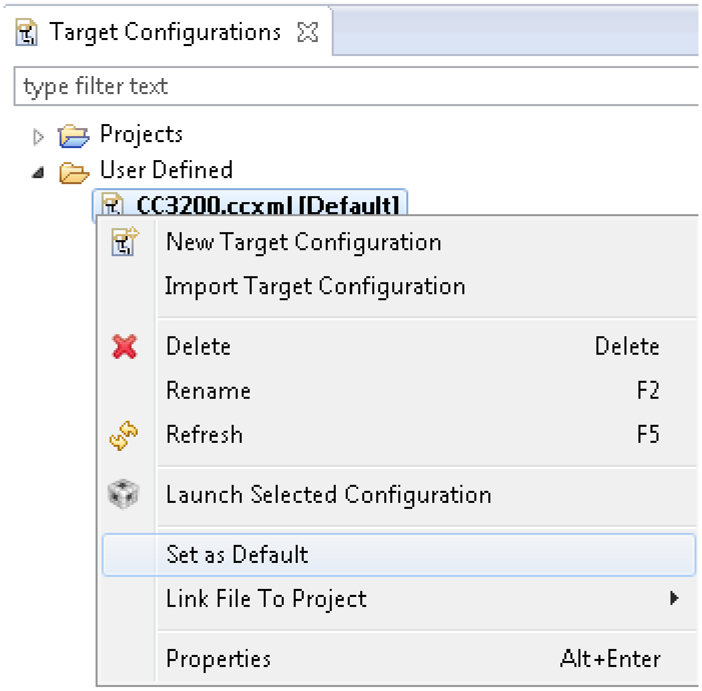


Figure 16. Set as Default

16. Launch application. Select the *wlan_station* project in Project Explorer, then click the debug icon as shown in Figure 17 to download code to the device and begin debugging. Press F8 to begin execution.

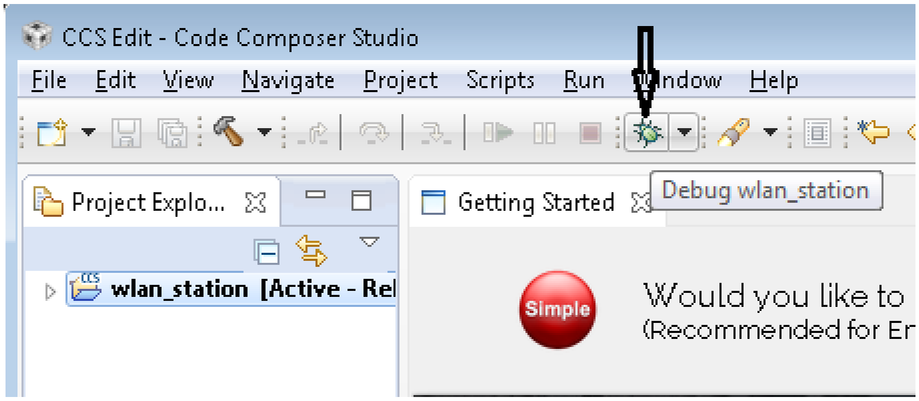

CCS Edit - Code Composer Studio
File Edit View Navigate Project Scripts Run Window Help
Debug wlan_station
Project Explo...
Getting Started
wlan_station [Active - Rel
Simple
Would you like to
(Recommended for Er

Figure 17. Debug *wlan_station*

Caution: Only one FTDI board should be connected to the PC while CCS downloads code to device.

3.2 *Option 2: IAR Workbench*

3.2.1 Download IAR

The CC3200 SDK has been built and tested with IAR 7.10.3, and older versions of IAR projects might not work properly on IAR 7.10.x. Most examples will only run with the fully licensed IAR Workbench.

1. Download IAR for ARM processors from the IAR System website, and install it using the installation wizard.
2. Copy the file c:\TI\CC3200SDK\cc3200-sdk\tools\iar_patch\armLMIFTDI.dll into the folder C:\Program Files (x86)\IAR Systems\Embedded Workbench 7.0\arm\bin (will need to replace existing file).

3.2.2 Rebuild the SimpleLink Driver

1. Start IAR and select *File>Open>Workspace* from the menu.

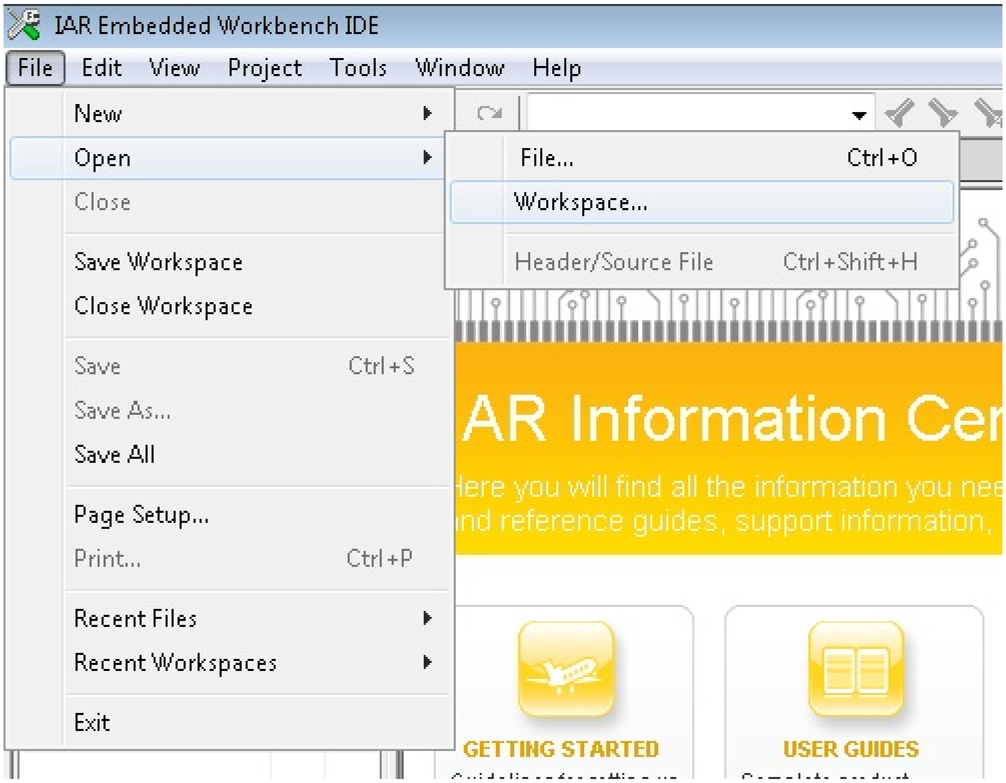
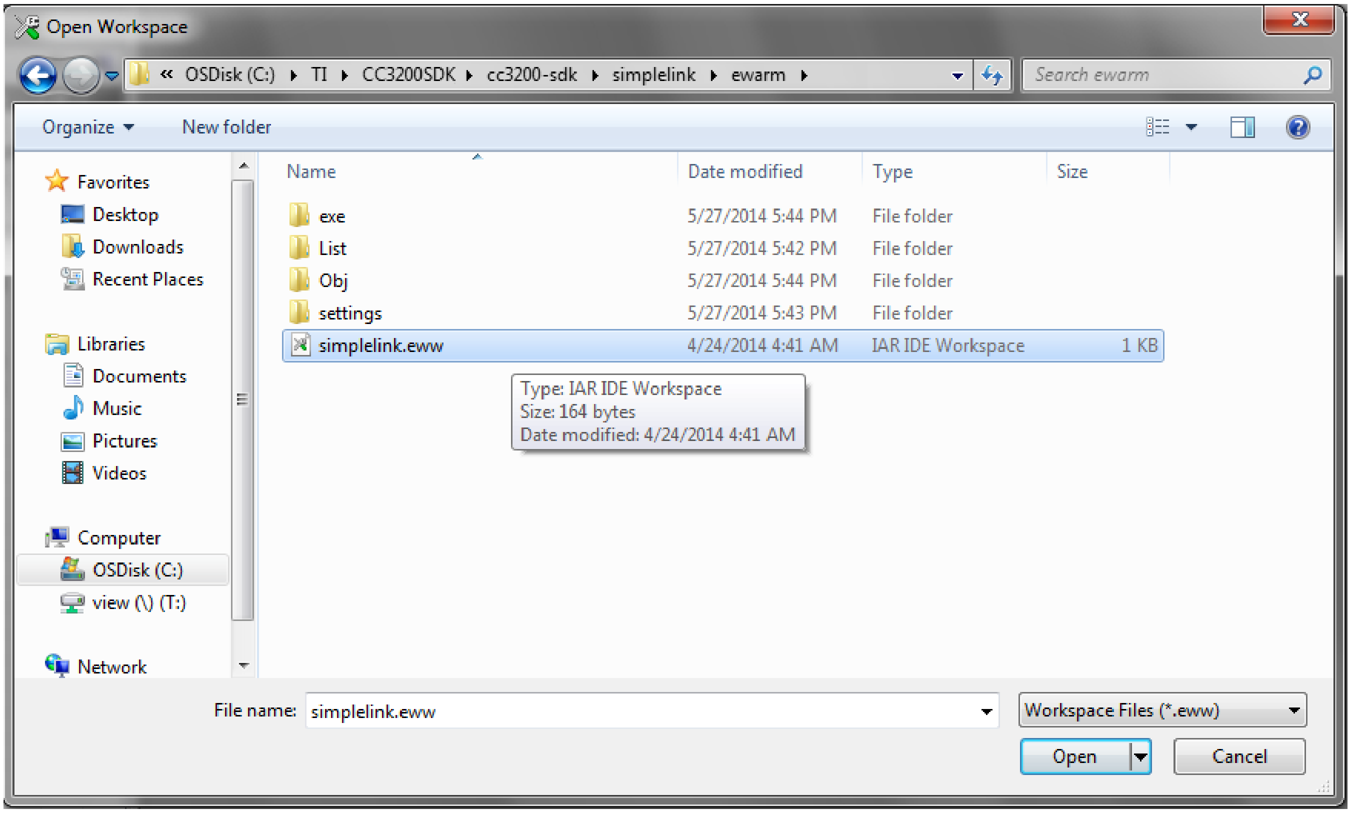


Figure 18. IAR Embedded Workbench IDE

2. Open the *simplelink* project by navigating to *C:\TI\CC3200SDK\cc3200-sdk\simplelink\ewarm* and opening *simplelink.eww*.



« OSDisk (C:) ▸ TI ▸ CC3200SDK ▸ cc3200-sdk ▸ simplelink ▸ ewarm ▸

Name	Date modified	Type	Size
exe	5/27/2014 5:44 PM	File folder	
List	5/27/2014 5:42 PM	File folder	
Obj	5/27/2014 5:44 PM	File folder	
settings	5/27/2014 5:43 PM	File folder	
simplelink.eww	4/24/2014 4:41 AM	IAR IDE Workspace	1 KB

Type: IAR IDE Workspace
Size: 164 bytes
Date modified: 4/24/2014 4:41 AM

File name: simplelink.eww

Workspace Files (*.eww)

Figure 19. Open *simplelink.eww*

3. Rebuild the *simplelink* project by selecting *Project>Rebuild All* from the menu as shown in Figure 20.

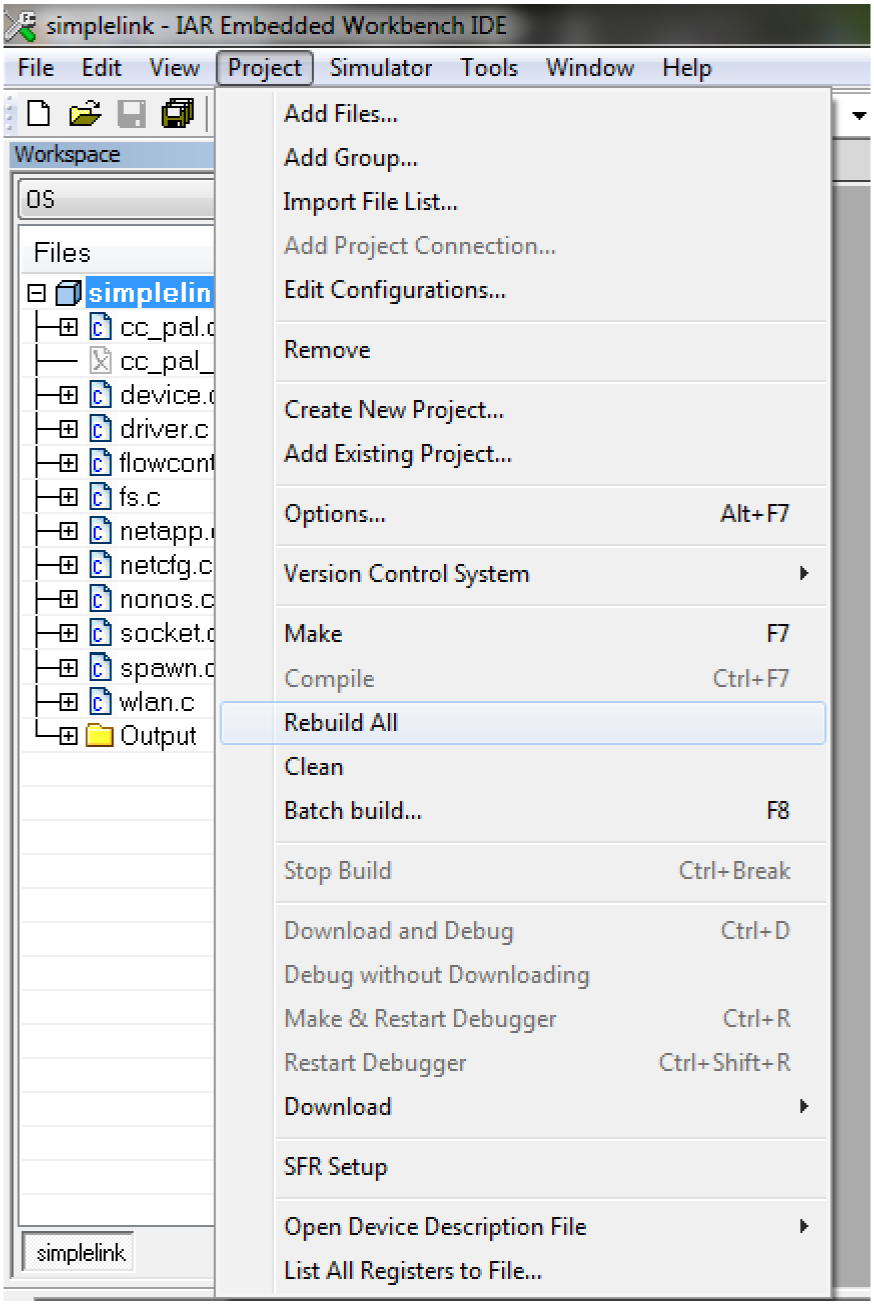
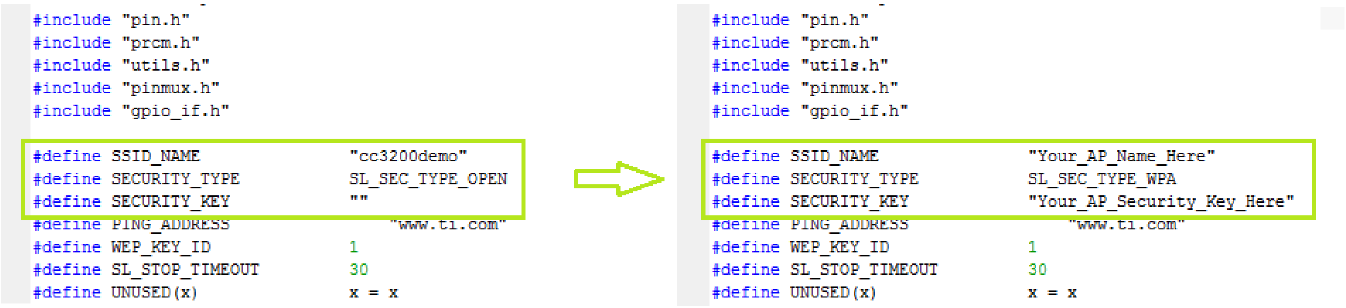


Figure 20. Rebuild the *simplelink* Project.

3.2.3 Rebuild, Download and Debug the WLAN Station Example

1. Open the *wlan_station* project by selecting *File>Open>Workspace* from the menu, navigating to *C:\TI\CC3200SDK\cc3200-sdk\example\ getting_started_with_wlan_station\ewarm*, and opening *wlan_station.eww*.
2. Open the *main.c* file of the *wlan_station* project for editing at *C:\TI\CC3200SDK\cc3200-sdk\example\getting_started_with_wlan_station\main.c*.
3. Edit *main.c* to use the SSID, security type and security key of the Access Point being used. Edit the macros SSID_NAME, SECURITY_TYPE and SECURITY_KEY to contain the Access Point's information as shown in Figure 21. The security types supported for this demo are WPA/WPA2 and Open. For Open security, define SECURITY_TYPE as SL_SEC_TYPE_OPEN. For WPA and WPA2

security, define it as SL_SEC_TYPE_WPA.

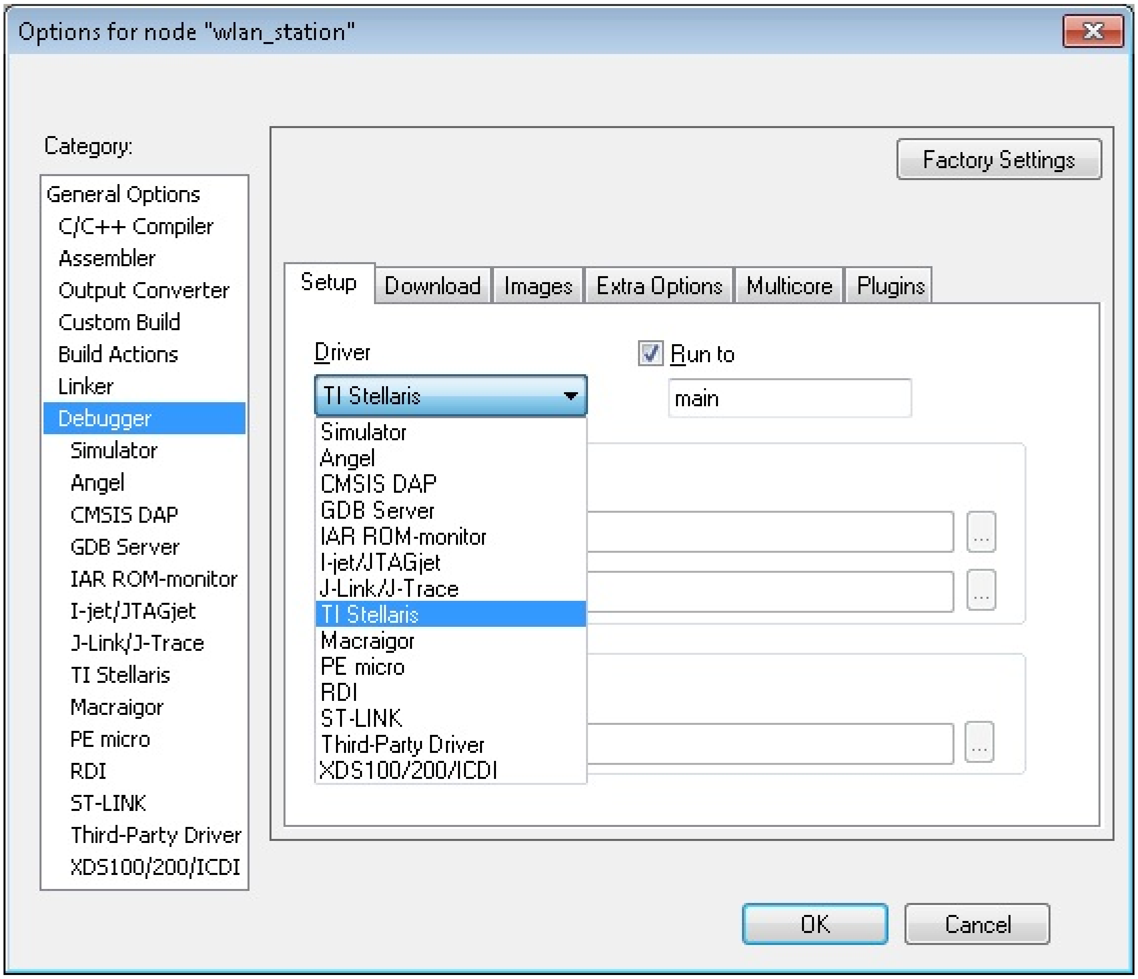


```
#include "pin.h"
#include "prcm.h"
#include "utils.h"
#include "pinmux.h"
#include "gpio_if.h"

#define SSID_NAME                "cc3200demo"
#define SECURITY_TYPE            SL_SEC_TYPE_OPEN
#define SECURITY_KEY             ""
#define PING_ADDRESS                 "www.ti.com"
#define WEP_KEY_ID               1
#define SL_STOP_TIMEOUT          30
#define UNUSED(x)                x = x
```

```
#include "pin.h"
#include "prcm.h"
#include "utils.h"
#include "pinmux.h"
#include "gpio_if.h"

#define SSID_NAME                "Your_AP_Name_Here"
#define SECURITY_TYPE            SL_SEC_TYPE_WPA
#define SECURITY_KEY             "Your_AP_Security_Key_Here"
#define PING_ADDRESS                 "www.ti.com"
#define WEP_KEY_ID               1
#define SL_STOP_TIMEOUT          30
#define UNUSED(x)                x = x
```

Figure 21. Editing *main.c*

4. Save *main.c*.
5. Rebuild the *wlan_station* project by selecting *Project>Rebuild All* from the menu.
6. The debugger must be configured to download code to the device. Select *Project>Options* from the menu, and select the Debugger category. In the Setup tab, choose TI Stellaris as the driver, as shown in Figure 22, and press OK.



Options for node "wlan_station"

Category: Debugger

Setup

Driver: TI Stellaris

☑ Run to: main

Figure 22. Select TI Stellaris Driver

7. Click the debug icon as shown in Figure 23 to download code to the device and start debugging. Select *Debug>Go* from the menu or press F5 to begin execution.

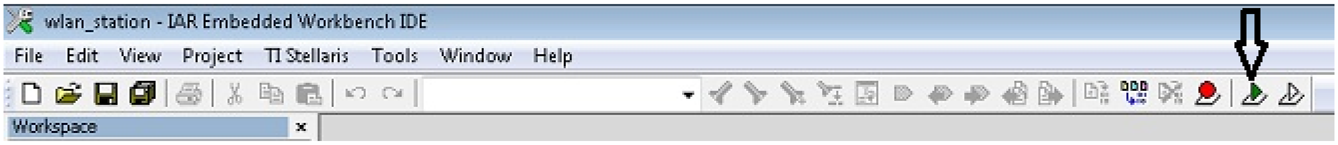

wlan_station - IAR Embedded Workbench IDE
File Edit View Project TI Stellaris Tools Window Help
Workspace

Figure 23. Debug Icon

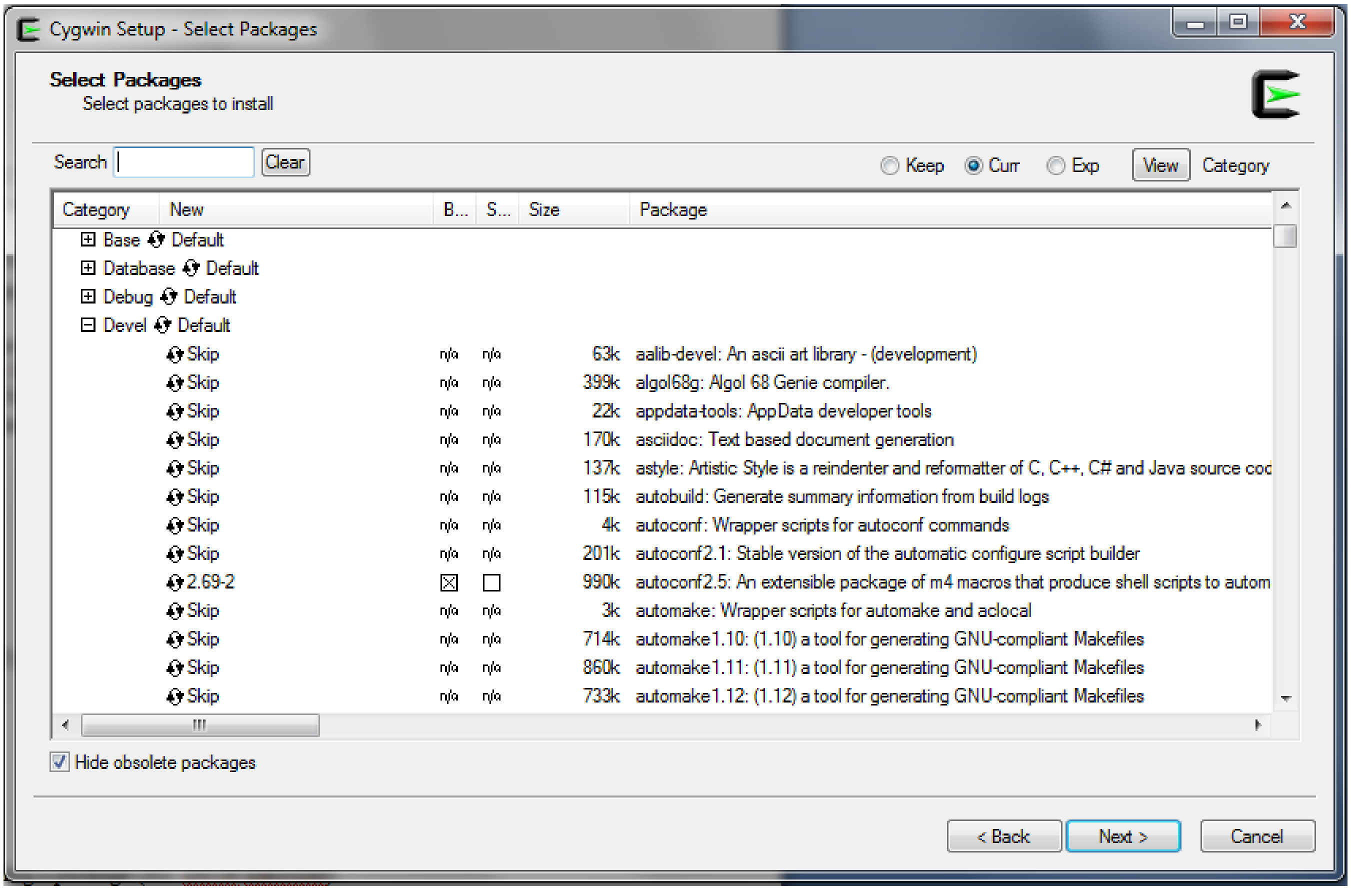
3.3 Option 3: GCC

This section demonstrates the GCC setup for the Windows 7 environment. GCC installation requires other dependencies to be installed to work with ARM-based devices.

3.3.1 Install Cygwin (Windows)

1. Download *setup-x86.exe* from http://cygwin.com/install.html and run it. Select the Install from Internet option.
2. Specify a proxy if necessary, depending on the network.
3. Choose a download site (for example, http://mirrors.kernel.org).
4. Include the latest versions of the following packages in the Cygwin installation (in addition to those included in the base installation):
    - Archive/unzip
    - Archive/zip
    - Devel/autoconf
    - Devel/automake
    - Devel/libtool
    - Devel/make
    - Devel/subversion (**Note**: if using TortoiseSVN/Windows7, skip this file)
    - Devel/gcc-core
    - Devel/gcc-g++
    - Devel/mingw-gcc-core
    - Devel/mingw-gcc-g++
    - Devel/mingw-runtime

   See Figure 24 for an example of selecting a package (as example: Devel/autoconf).



Select Packages
Select packages to install

Search [ ] Clear ○ Keep ● Curr ○ Exp View Category

Category	New	B...	S...	Size	Package
⊞ Base ⟳ Default					
⊞ Database ⟳ Default					
⊞ Debug ⟳ Default					
⊟ Devel ⟳ Default					
	⟳ Skip	n/a	n/a	63k	aalib-devel: An ascii art library - (development)
	⟳ Skip	n/a	n/a	399k	algol68g: Algol 68 Genie compiler.
	⟳ Skip	n/a	n/a	22k	appdata-tools: AppData developer tools
	⟳ Skip	n/a	n/a	170k	asciidoc: Text based document generation
	⟳ Skip	n/a	n/a	137k	astyle: Artistic Style is a reindenter and reformatter of C, C++, C# and Java source coc
	⟳ Skip	n/a	n/a	115k	autobuild: Generate summary information from build logs
	⟳ Skip	n/a	n/a	4k	autoconf: Wrapper scripts for autoconf commands
	⟳ Skip	n/a	n/a	201k	autoconf2.1: Stable version of the automatic configure script builder
	⟳ 2.69-2	☒	☐	990k	autoconf2.5: An extensible package of m4 macros that produce shell scripts to autom
	⟳ Skip	n/a	n/a	3k	automake: Wrapper scripts for automake and aclocal
	⟳ Skip	n/a	n/a	714k	automake1.10: (1.10) a tool for generating GNU-compliant Makefiles
	⟳ Skip	n/a	n/a	860k	automake1.11: (1.11) a tool for generating GNU-compliant Makefiles
	⟳ Skip	n/a	n/a	733k	automake1.12: (1.12) a tool for generating GNU-compliant Makefiles

☑ Hide obsolete packages

Figure 24. Cygwin Setup

5. The system will find dependencies. Press Next.
6. After a successful Cygwin installation, add its path (*c:\cygwin\bin\*) to the Windows environment variable PATH by going into *Control Panel>System>Advanced System Settings>Environment Variables*. Under *System Variables*, select PATH and press Edit. Append “;C:\cygwin\bin\” to the end of the line and press Ok.

3.3.2 Get GNU Tools for ARM Embedded Processors

Download and run the latest version of *gcc-arm-none-eabi-<version>-win32.exe* from https://launchpad.net/gcc-arm-embedded. The link to the file should be on the right side of the page and will appear as a green button with the text: ”gcc-arm-non...4-win32.exe.” Install under the Cygwin root directory (default: *c:\cygwin*).

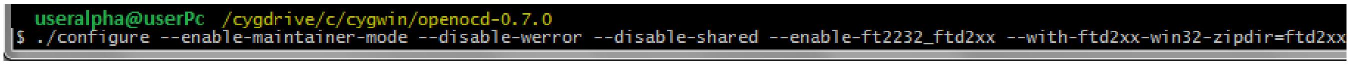
3.3.3 Build OpenOCD for FTDI Interface

1. Download the Open On-Chip Debugger (OpenOCD) source from http://sourceforge.net/projects/openocd/files/openocd/0.7.0/ Look for the zip file *openocd-0.7.0.zip*.
2. Extract the OpenOCD source into the Cygwin directory (*c:\cygwin*). This will create a directory called *openocd-<version>* (for example, *c:\cygwin\openocd-0.7.0*) under the Cygwin directory containing all the OpenOCD source contents.
3. Download the FTDI driver library (x86 [32-bit] zip version) at http://www.ftdichip.com/Drivers/CDM/CDM%20v2.10.00%20WHQL%20Certified.zip.
4. Extract the FTDI source into the path *c:\cygwin\openocd-<version> ftd2xx* (for example, *c:\cygwin\openocd-0.7.0\ ftd2xx*).
5. Run the Cygwin terminal and change the directory to *openocd-<version>* (for example, by using a command such as: *cd c:cygwin/openocd-0.7.0*).

6. Run the following command:

```
./configure --enable-maintainer-mode --disable-werror --disable-shared --enable-ft2232_ftd2xx -
-with-ftd2xx-win32-zipdir=ftd2xx
```

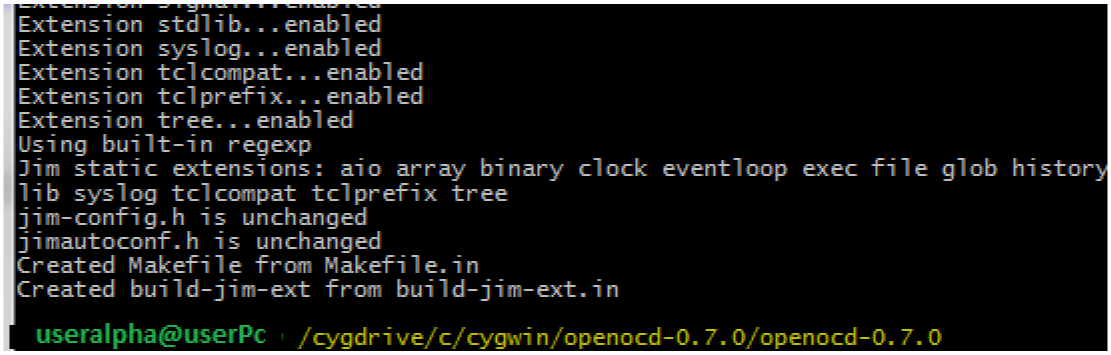
   The command should look similar to Figure 25.



```
useralpha@userPc /cygdrive/c/cygwin/openocd-0.7.0
$ ./configure --enable-maintainer-mode --disable-werror --disable-shared --enable-ft2232_ftd2xx --with-ftd2xx-win32-zipdir=ftd2xx
```

Figure 25. Cygwin Terminal

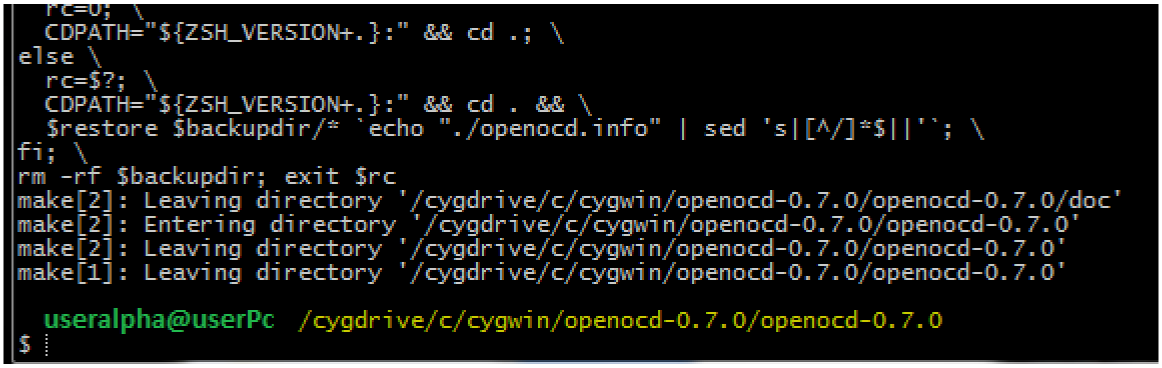
The last lines of the result should appear as in Figure 26.



```
Extension signal...enabled
Extension stdlib...enabled
Extension syslog...enabled
Extension tclcompat...enabled
Extension tclprefix...enabled
Extension tree...enabled
Using built-in regexp
Jim static extensions: aio array binary clock eventloop exec file glob history
lib syslog tclcompat tclprefix tree
jim-config.h is unchanged
jimautoconf.h is unchanged
Created Makefile from Makefile.in
Created build-jim-ext from build-jim-ext.in

 useralpha@userPc /cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0
```

Figure 26. Cygwin Terminal

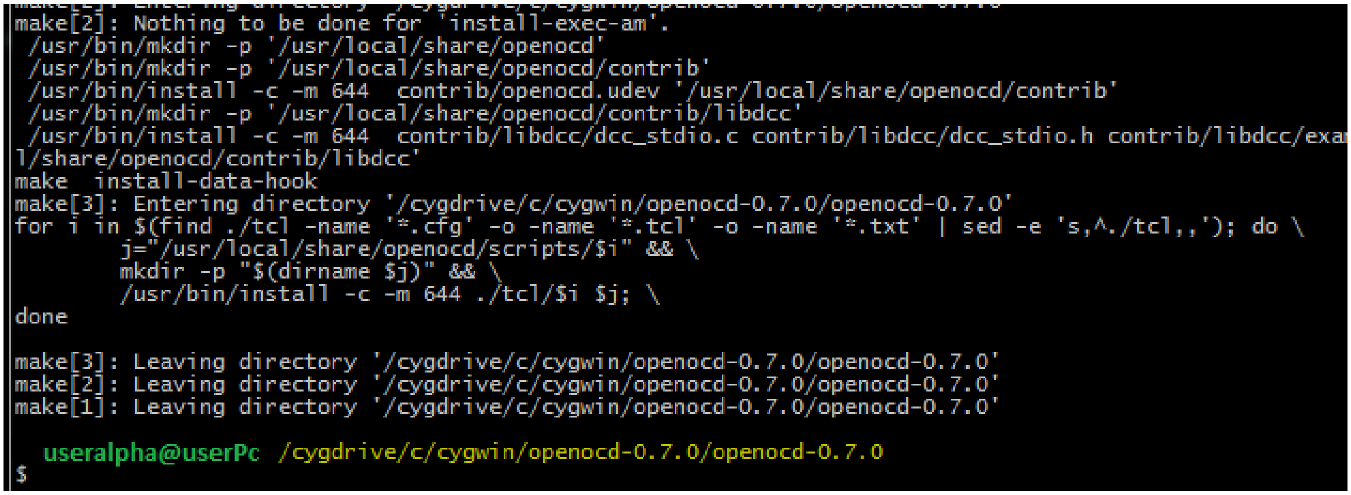
7. Run the command ‘**autoreconf --force --install**.’
8. Run the command ‘**make**.’ This may take several minutes. The last lines of the result should appear as in Figure 27.



```
  rc=0; \
  CDPATH="${ZSH_VERSION+.}:" && cd .; \
else \
  rc=$?; \
  CDPATH="${ZSH_VERSION+.}:" && cd . && \
  $restore $backupdir/* `echo "./openocd.info" | sed 's|[^/]*$||'`; \
fi; \
rm -rf $backupdir; exit $rc
make[2]: Leaving directory '/cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0/doc'
make[2]: Entering directory '/cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0'
make[2]: Leaving directory '/cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0'
make[1]: Leaving directory '/cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0'

  useralpha@userPc /cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0
$
```

Figure 27. Running the Make Command

9. Run the command ‘**make install**.’ The last lines of the result should appear as in Figure 28.



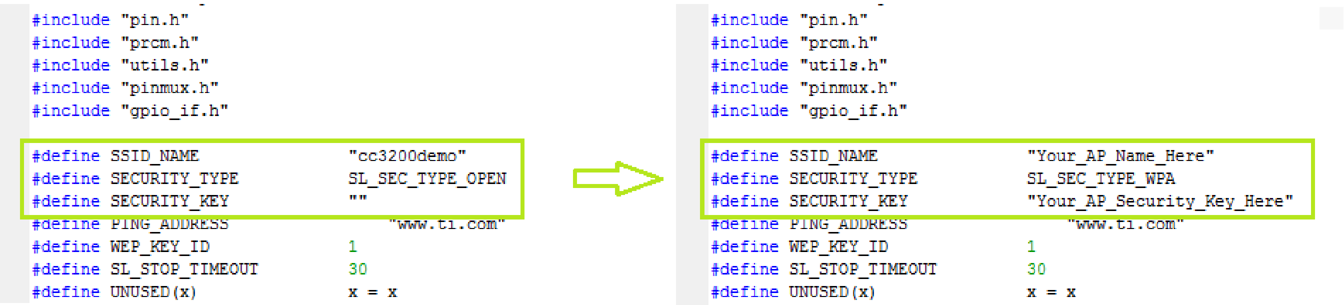
```
make[2]: Nothing to be done for 'install-exec-am'.
 /usr/bin/mkdir -p '/usr/local/share/openocd'
 /usr/bin/mkdir -p '/usr/local/share/openocd/contrib'
 /usr/bin/install -c -m 644  contrib/openocd.udev '/usr/local/share/openocd/contrib'
 /usr/bin/mkdir -p '/usr/local/share/openocd/contrib/libdcc'
 /usr/bin/install -c -m 644  contrib/libdcc/dcc_stdio.c contrib/libdcc/dcc_stdio.h contrib/libdcc/exa
l/share/openocd/contrib/libdcc'
make  install-data-hook
make[3]: Entering directory '/cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0'
for i in $(find ./tcl -name '*.cfg' -o -name '*.tcl' -o -name '*.txt' | sed -e 's,^./tcl,,'); do \
        j="/usr/local/share/openocd/scripts/$i" && \
        mkdir -p "$(dirname $j)" && \
        /usr/bin/install -c -m 644 ./tcl/$i $j; \
done

make[3]: Leaving directory '/cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0'
make[2]: Leaving directory '/cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0'
make[1]: Leaving directory '/cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0'

  useralpha@userPc /cygdrive/c/cygwin/openocd-0.7.0/openocd-0.7.0
$
```

Figure 28. Running the Make Install Command

10. After the command has run successfully, check that the file *openocd.exe* is generated at path *C:\cygwin\usr\local\bin*. Add this path to the Windows PATH environment variable.

3.3.4 Compile the GCC SDK project

1. Open the *main.c* file of the *wlan_station* project for editing at *C:\TI\CC3200SDK\cc3200-sdk\example\getting_started_with_wlan_station\main.c*.
2. Edit *main.c* to use the SSID, security type and security key of the Access Point being used. Edit the macros SSID_NAME, SECURITY_TYPE and SECURITY_KEY to contain the Access Point's information as shown in Figure 29. The security types supported for this demo are WPA/WPA2 and Open. For Open security, define SECURITY_TYPE as SL_SEC_TYPE_OPEN. For WPA and WPA2 security, define it as SL_SEC_TYPE_WPA.



```
#include "pin.h"
#include "prcm.h"
#include "utils.h"
#include "pinmux.h"
#include "gpio_if.h"

#define SSID_NAME                 "cc3200demo"
#define SECURITY_TYPE             SL_SEC_TYPE_OPEN
#define SECURITY_KEY              ""
#define PING_ADDRESS                  "www.ti.com"
#define WEP_KEY_ID                1
#define SL_STOP_TIMEOUT           30
#define UNUSED(x)                 x = x
```

```
#include "pin.h"
#include "prcm.h"
#include "utils.h"
#include "pinmux.h"
#include "gpio_if.h"

#define SSID_NAME                 "Your_AP_Name_Here"
#define SECURITY_TYPE             SL_SEC_TYPE_WPA
#define SECURITY_KEY              "Your_AP_Security_Key_Here"
#define PING_ADDRESS                  "www.ti.com"
#define WEP_KEY_ID                1
#define SL_STOP_TIMEOUT           30
#define UNUSED(x)                 x = x
```

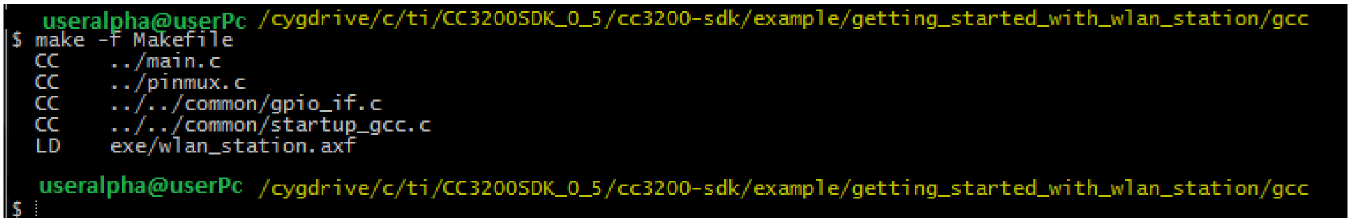
Figure 29. Editing *main.c*

3. Save *main.c*.

In the Cygwin terminal, change the directory to *C:\TI\CC3200SDK\cc3200-sdk\example\getting_started_with_wlan_station\gcc\* and run following command:

```
make –f  Makefile
```

This command should appear as in Figure 30. Note that Cygwin uses forward slashes to separate directories.



```
  useralpha@userPc /cygdrive/c/ti/CC3200SDK_0_5/cc3200-sdk/example/getting_started_with_wlan_station/gcc
$ make -f Makefile
  CC    ../main.c
  CC    ../pinmux.c
  CC    ../../common/gpio_if.c
  CC    ../../common/startup_gcc.c
  LD    exe/wlan_station.axf

  useralpha@userPc /cygdrive/c/ti/CC3200SDK_0_5/cc3200-sdk/example/getting_started_with_wlan_station/gcc
$ 
```

Figure 30. Makefile Command

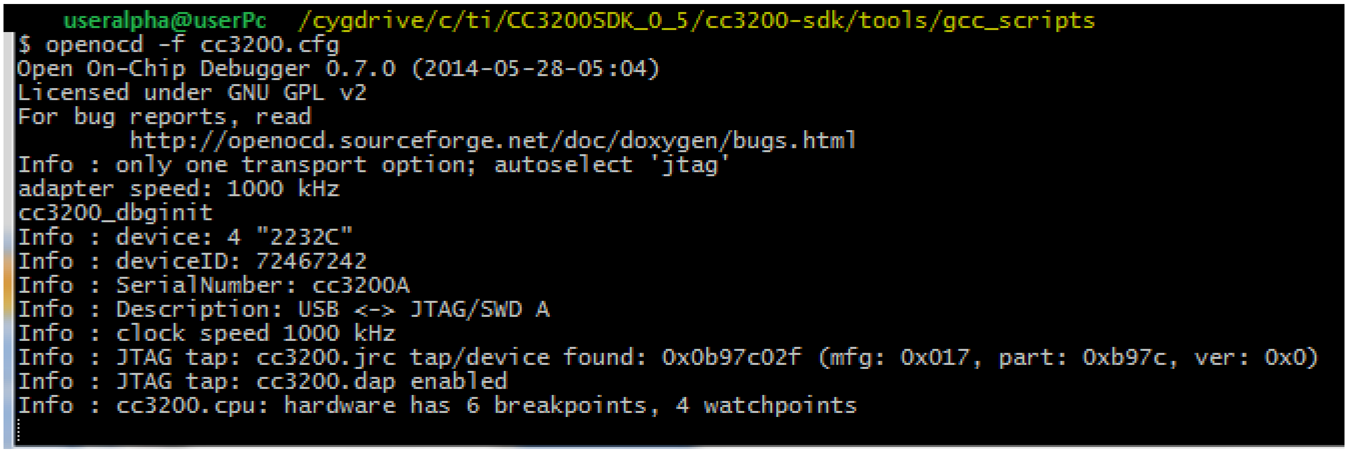
This generates the *wlan_station.axf* file under the *gcc\exe* folder.

3.3.5 Target Connection and Debug (GDB)

1. The OpenOCD configuration file for FTDI is present under the *C:\CC3200SDK\cc3200-sdk\tools\gcc_scripts\* folder. To test the connection to the CC3200 FTDI Launchpad, navigate to the *<cc3200-sdk>\tools\gcc_scripts* folder in the Cygwin terminal, run the following command and check the output to see if the connection happened properly.

   ```
   openocd -f cc3200.cfg
   ```

   See Figure 31 for the output screen while the CC3200 device is connected through GDB.



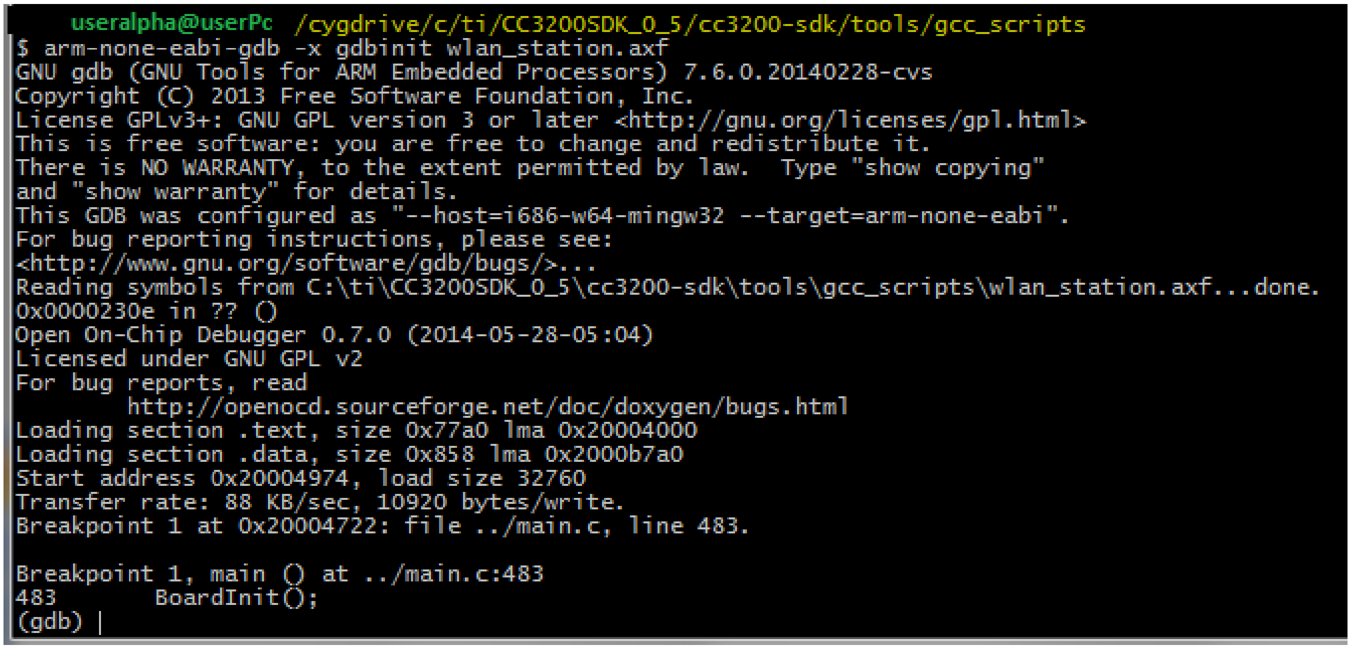
```
   useralpha@userPc  /cygdrive/c/ti/CC3200SDK_0_5/cc3200-sdk/tools/gcc_scripts
$ openocd -f cc3200.cfg
Open On-Chip Debugger 0.7.0 (2014-05-28-05:04)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.sourceforge.net/doc/doxygen/bugs.html
Info : only one transport option; autoselect 'jtag'
adapter speed: 1000 kHz
cc3200_dbginit
Info : device: 4 "2232C"
Info : deviceID: 72467242
Info : SerialNumber: cc3200A
Info : Description: USB <-> JTAG/SWD A
Info : clock speed 1000 kHz
Info : JTAG tap: cc3200.jrc tap/device found: 0x0b97c02f (mfg: 0x017, part: 0xb97c, ver: 0x0)
Info : JTAG tap: cc3200.dap enabled
Info : cc3200.cpu: hardware has 6 breakpoints, 4 watchpoints
```

Figure 31. Output Screen

2. Press <ctrl>+c to return to prompt.
3. Copy the *wlan_station.axf* file found in *C:\TI\CC3200SDK\cc3200-sdk\example\getting_started_with_wlan_station\gcc\exe\* to the directory *C:\TI\CC3200SDK\c3200-sdk\tools\gcc_scripts\*.
4. To start debugging using GDB on CC3200, go to *C:\TI\CC3200SDK\cc3200-sdk\tools\gcc_scripts\* and run the following command at the Cygwin prompt:

   ```
   arm-none-eabi-gdb -x gdbinit wlan_station.axf
   ```

   See Figure 32 for the result of debugging the *wlan_station* application from GCC.



```
    useralpha@userPc /cygdrive/c/ti/CC3200SDK_0_5/cc3200-sdk/tools/gcc_scripts
$ arm-none-eabi-gdb -x gdbinit wlan_station.axf
GNU gdb (GNU Tools for ARM Embedded Processors) 7.6.0.20140228-cvs
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "--host=i686-w64-mingw32 --target=arm-none-eabi".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from C:\ti\CC3200SDK_0_5\cc3200-sdk\tools\gcc_scripts\wlan_station.axf...done.
0x0000230e in ?? ()
Open On-Chip Debugger 0.7.0 (2014-05-28-05:04)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.sourceforge.net/doc/doxygen/bugs.html
Loading section .text, size 0x77a0 lma 0x20004000
Loading section .data, size 0x858 lma 0x2000b7a0
Start address 0x20004974, load size 32760
Transfer rate: 88 KB/sec, 10920 bytes/write.
Breakpoint 1 at 0x20004722: file ../main.c, line 483.

Breakpoint 1, main () at ../main.c:483
483       BoardInit();
(gdb) |
```

Figure 32. Debugging *wlan_station*

This results in a GDB prompt. To continue, type ‘continue’ and press enter. For other commands, consult the GDB Quick Guide.

4 Summary

After the development environment has been set up, see the following resources for further assistance in development:

- CC3200 Programmer's Guide – This guide contains information on how to use the SimpleLink API for writing WLAN-enabled applications.
- PinMux Tool – This utility helps determine how to best assign peripherals to the appropriate CC3200 package pins.
- Uniflash – The Uniflash tool manually stores files on the external serial flash. This includes the application binary and SimpleLink firmware patch files. Also, any configuration files, security certificates, web pages, and so forth can be stored using this tool.
- CC3200 Wiki – All information and tools for the CC3200, including the above, can be found on the CC3200 Wiki page.

5 Acronyms Used

STA – Wi-Fi Station

AP – Wi-Fi Access Point

WLAN – Wireless LAN

CCS – Code Composer Studio

GCC – GNU Compiler Collection

Revision History

Changes from Original (June 2014) to A Revision **Page**

- Updated image with SOP-2 Jumper marked 5
- Added function #4 9
- Changed to Select the Wireless Connectivity MCUs option for processor support. 9
- Updated image 9
- Changed to Select the CC3200 Add On. 10
- Updated image 13

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

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 and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265