CC3200 SimpleLink™ Wi-Fi® and Internet-of-Things Solution, a single-chip wireless MCU

Software Development Kit (SDK) v1.1.0 Release Notes

TABLE OF CONTENTS

1 Introduction

This document describes the Software Development Kit (SDK) version 1.1.0 for use with the CC3200 SimpleLink Wi-Fi MCU device mounted on the CC3200 LaunchPad evaluation kit.

The same SDK is also applicable to Pre-production devices. Errata section of the document includes minor performance limitations of pre-production Devices.

2 Getting Started

Please follow the on-line CC3200 Quick Start Guide to start using the CC3200 LaunchPad development platform.

Please download the CC3200 Getting Started Guide to get started with your project development.

2.1 Procedure to Upgrade from SDKv1.0.0 to SDKv1.1.0

To upgrade from SDKv1.0.0 to SDKv1.1.0, servicepack “servicepack_1.0.0.10.0” needs to be flashed on CC3200. Service pack “servicepack_1.0.0.10.0” is hosted in *CC31xx_CC32xx_ServicePack-1.0.0.10.0-windows-installer.exe* downloadable from http://www.ti.com/tool/cc3200sdk. Please refer to UNIFLASH Quick start guide on details of flashing (http://processors.wiki.ti.com/index.php/CC31xx_%26_CC32xx_UniFlash) the service pack.

3 Main changes from SDK Package 1.0.0

3.1 Networking

3.1.1 Enable SHA256 Client mode

- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256

3.1.2 Disable Calibration every 24Hours

Calibration are not needed when exiting from Hibernate and 24hours have passed

3.1.3 Reduce the AP aging time to 1Min (instead of 5Min)

Detection of a STA disconnected from the AP (not gracefully) will be detected after 1Min

3.1.4 Add an option to set the DNS and Default gateway to 0.0.0.0

Setting the DNS and Default gateway to 0.0.0.0 in iOS 8 can allow the mobile device to keep data over 3G and still have Wi-Fi connection

3.1.5 Increase SSL Time out

Increase the SSL connection time out from 5 sec to 30 sec

3.1.6 Host Driver fixes and code optimizations

- Clean up and reduced the code size to less than 7KB
- Robustness fixes
- Allow easier integration with Network Application Libraries

3.2 SDK Content

Please refer to Section 11 (Revision History) for changes in the SDK components.

4 Release Content

Item	Version	Type
Device	CC3200R1 [ES1.33 Chip Id : 0x4000010] XCC3200HZ [ES1.32 Chip id : 0x3000013]	Production device Pre-production device
Development boards	CC3200-LAUNCHXL Rev3.2 onwards	Orderable from TI
SDK Installer	CC3200SDK-1.0.0-windows-installer.exe For Windows 7,8 and Windows XP	Provided with a click wrap license
Firmware	2.4.0.2.31.1.3.0.1.1.0.3.34 (Production Device) 2.4.7.2.31.1.3.4.1.1.5.3.34 (Pre-Production Device)	servicepack_1.0.0.10.0 is provided thru ServicePack *CC31xx_CC32xx_ServicePack-1.0.0.10.0-windows-installer.exe* downloadble from http://www.ti.com/tool/cc3200sdk
Network Processor host driver	Version 1.0.0.10	Source code
MCU Peripherals Drivers	Version 1.1.0	Source code
Power Management Framework Library	Version 1.1.1	Source code
Supported IDE	IAR version : 7.20 IAR version : 7.30 CCS version : 6.0.1	Accessible separately. Not a part of this package
Demo	Embedded HTML web-site	Pre-flashed on LaunchPad and source code provided
User guides	CC3200 Getting started guide CC3200LAUNCHXL User Guide SimpleLink Host Driver Programmer's Guide	PDF PDF Doxygen HTML
Tools	USB driver for CC3200LAUNCHXL for Windows7, 8 and Windows XP	Executable

5 Directory structure of SDK

Double-Click on the package installation file and follow the installation guidelines to copy the directories (and files) to the preferred location.
The first level directory structure is as shown in the table below.

Directory Name	Content
Docs	• Getting Started Guide for application development • Launchpad User Guide • SimpleLink Host Driver Programmer's Guide • Peripheral Driver Library User's Guide • Power Management Framework Guide • Application notes for sample applications • API documents for MQTT and HTTP Client libraries • Simplelink OTA Extlib API user's guide
Examples	Example application in source code
Driverlib	• Peripheral driver library source files • The driverlib.a is provided in CCS, GCC and IAR directories
Inc	• Register definition header files
middleware	• Power management framework providing an easy to use infrastructure for power aware solution
Oslib	• Interface file to configure Free-RTOS or TI-RTOS
SimpleLink	• The SimpleLink Network Processor host driver code. • Simplelink library binaries for different configurations are available under IAR, CCS and GCC directories
Simplelink_extlib	• Contains the OTA (Over the Air) library
netapps	• SMTP client library source files • XMPP client library source files • HTTP server library source files • HTPP client library source files • TFTP client library source files • JSON xml parser library source files • MQTT client library source files • MQTT client server library source files
third_party	• FatFS source files • FreeRTOS souce files
ti_rtos	• Abstraction layer files for TI-RTOS
tools	• **ccs_patch –** Files required for CCS-FTDI-LP connection • **gcc_scripts –** Script files required for GCC compiler. • **ftdi-** Contains CC3200 FTDI-USB driver for Windows • **Iar_patch –** Files required for IAR-FTDI-LP connection

6 MCU and Associated Peripherals

CC3200 Device provides a user programmable Cortex M4 core in conjunction with user programmable peripheral. Drivers and example applications are provided in the SDK package for the following peripheral interfaces -

ADC	Analog to digital converter.
AES	Advanced encryption standard
CAMERA	Parallel interface (8 bits) that can be used to connect to camera sensors.
CRC	Cyclic redundancy check
DES	Data encryption standard.
GPIO	General purpose input/output
I2C	Standard I2C interface to communicate with various sensors
Interrupt	Interrupt module
I2S	Audio signal processor
PinMux	Pinmux setting for Pads
PRCM	Power reset and clock module
SDHost	Secure digital host controller
SHAMD5	Secure hash algorithm, message digest algorithm
SPI	Serial peripheral interface
Systick	System tick functionality
Timer	General purpose timers
UART	Standard UART interface
UDMA	Direct memory access
WDT	Watchdog timer

7 Networking

7.1 Package Quality

7.1.1 Interoperability - IOP

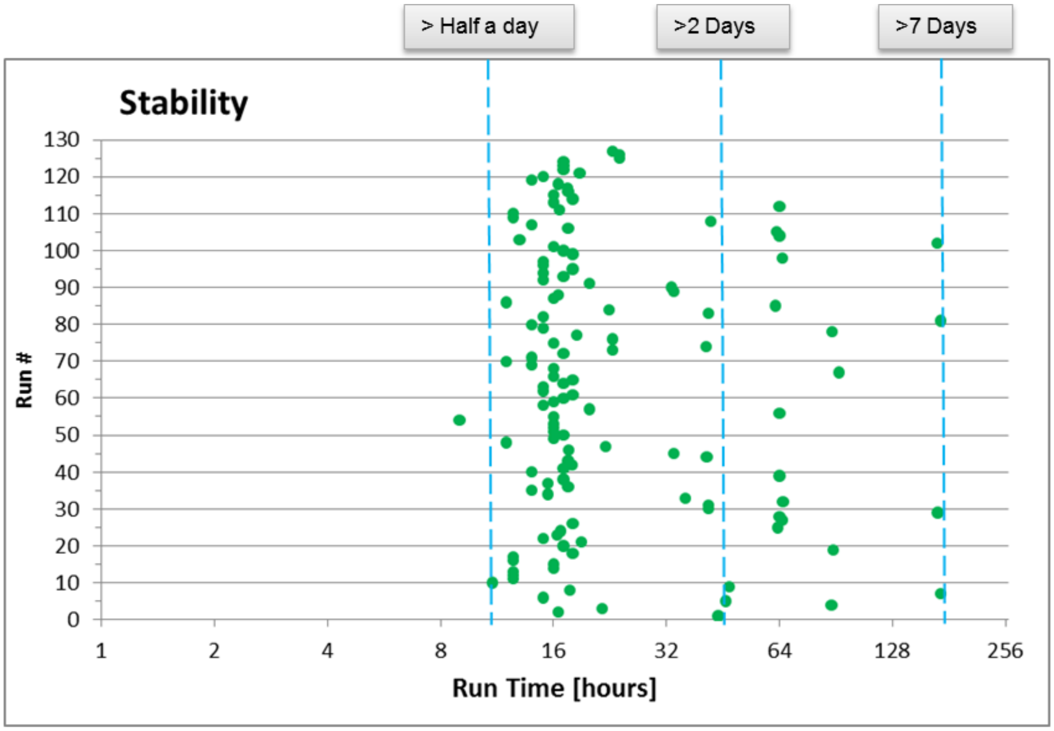
- STA mode was tested for connection, traffic and power consumption with more than 200 AP
- AP mode was tested for connection and traffic with more than 50 STA

7.1.2 Robustness

- Use cases were tests for 1000 of cycles – for example:
  - Connect/Disconnect
  - On/Off
  - Connect, Send Packet, Disconnect

7.1.3 Stability

- Stability in all traffic scenarios was tested for at least 12 hours (major use cases were tested for at least 24hours) and upto 7 days – User may rarely experience:
  - Traffic Stops
  - System freeze
- Main traffic scenarios including TCP/UDP (RX and TX) have been tested with the following set of APs
  - Cisco – 1250.
  - Cisco - 1240AG
  - Ralink - RT3800PDAP3.
  - Netgear - WNDR3700.
  - Broadcom AP – BCM94718NR
  - Dlink - DAP-2553.
  - Apple TimeCapsule
  - Apple Extreme.
  - Apple - A1355
  - Buffalo - WHR-G300N.
  - Linksys - WRT610N.
  - Linksys - E1550.
  - TP Link - WR740N.



- ○ Green Dot - Test was stopped after a preconfigured time

7.2 Features

7.2.1 Wi-Fi

Standards	802.11b/g/n (20MHz SISO) **Station** and Wi-Fi Direct **Client**
Supported Channels	1-13 The default regulatory domain is US (1-11)
Personal Security	WEP, WPA and WPA2
Enterprise Security	WPA-2 Enterprise EAP Fast, EAP PEAPv0 MSCHAPv2, EAP PEAPv0 TLS, EAP PEAPv1 TLS, EAP TLS, EAP TTLS TLS, EAP TTLS MSCHAPv2
Provisioning	SmartConfig^{TM} technology Wi-Fi Protected Setup (WPS2) Access Point mode with internal HTTP Web Server
Standards	802.11b/g **Access Point** and Wi-Fi Direct **Group Owner**
Clients	1
Personal Security	WEP, WPA and WPA2

7.2.2 Networking protocols

IP	IPv4
Transport	UDP TCP RAW ICMP
Cross-Layer	DHCP ARP DNS
Application	mDNS DNS-SD HTTP 1.0 web server
Transport Layer Security	SSLV3 SSL_RSA_WITH_RC4_128_SHA SSLV3 SSL_RSA_WITH_RC4_128_MD5 TLSV1 TLS_RSA_WITH_RC4_128_SHA TLSV1 TLS_RSA_WITH_RC4_128_MD5 TLSV1 TLS_RSA_WITH_AES_256_CBC_SHA TLSV1 TLS_DHE_RSA_WITH_AES_256_CBC_SHA TLSV1 TLS_ECDHE_RSA_WITH_RC4_128_SHA TLSV1 TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA TLSV1_1 TLS_RSA_WITH_RC4_128_SHA TLSV1_1 TLS_RSA_WITH_RC4_128_MD5 TLSV1_1 TLS_RSA_WITH_AES_256_CBC_SHA TLSV1_1 TLS_DHE_RSA_WITH_AES_256_CBC_SHA TLSV1_1 TLS_ECDHE_RSA_WITH_RC4_128_SHA TLSV1_1 TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA TLSV1_2 TLS_RSA_WITH_RC4_128_SHA TLSV1_2 TLS_RSA_WITH_RC4_128_MD5 TLSV1_2 TLS_RSA_WITH_AES_256_CBC_SHA TLSV1_2 TLS_DHE_RSA_WITH_AES_256_CBC_SHA TLSV1_2 TLS_ECDHE_RSA_WITH_RC4_128_SHA TLSV1_2 TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA TLSV1_2 TLS_RSA_WITH_AES_128_CBC_SHA256* TLSV1_2 TLS _RSA_WITH_AES_256_CBC_SHA256* TLSV1_2 TLS _ECDHE_RSA_WITH_AES_128_CBC_SHA256* TLSV1_2 TLS _ECDHE_ECDSA_WITH_AES_128_CBC_SHA256* * Client mode
User application sockets	Up to 8 open sockets Up to 2 secured application sockets: - One server (listen socket and accept socket) + client (data socket) - Up to two clients (data socket)

7.2.3 Advanced Features

802.11 Transceiver	Transmit and Receive raw Wi-Fi packets with full control over payload. Wi-Fi disconnect mode. Can be used for general-purpose applications (e.g. tags, sniffer, RF tests)
Traffic Filters	Embedded filters to reduce power consumption and Wake-on-LAN trigger packets (IP and MAC layer)

7.2.4 Power modes

MCU Power Management framework library allows user applications to exploit the following device power management modes:

Sleep	RAM retention – Yes CPU context retention - Yes Peripheral context retention – Enable Sleep clock for the peripherals to retain their context Wake source – Any interrupt
Deep Sleep	RAM retention – RAM is retained by default, but the application user can decide whether and which blocks to retain when the MCU enter Deep Sleep. CPU context retention - Yes Peripheral context retention – Enable DeepSleep clock for the peripherals to retain their context Wake source – Any interrupt
Low Power Deep Sleep(LPDS)	RAM retention – RAM is retained by default, but the application user can decide whether and which blocks to retain when the MCU enter LPDS. CPU context retention – No (if required, context need to be saved and restored in SW) Peripheral context retention – No (Peripherals need to be reconfigured after exiting from LPDS) Wake source – GPIO (2,4,11,13,17,24 – only one), LPDS Wake timer and Network wakeup
HIbernate	RAM retention - No CPU context retention - No Peripheral context retention – No Wake source – GPIO (2,4,11,13,17,24), Slow clock counter

For further details, please refer to CC32xx Power Management Framework

The Wi-Fi network processor core supports the following low power policies

Low Power mode	Uses 802.11 Power Save and Device Deep Sleep Power with three user configurable policies
Configurable Power Policies	• Normal (Default) - Best tradeoff between traffic delivery time and power performance

	• Low power –Used only for Transceiver mode application (Disconnect mode) • Long Sleep Interval – wakes up for the next DTIM after a configurable sleep interval, up to 2 seconds. This policy is only applicable for client socket mode

8 Advanced Information

8.1 System/Software Capabilities

- Host SPI interface max speed: 20MHz (production device) and 14MHz (Pre-Production)
- Robustness tests
  - Start/Stop with WiFi Connect/Disconnect and data Tx burst was tested for 5000 cycles and found to be stable
  - WiFi Connect/Disconnect without data was tested for 5000 cycles and found stable
- TCP/IP
  - TCP Window size is 32KB for production device
  - The memory resources are divided among all user sockets and the TCP windows size might change accordingly
  - IP Fragmentation is not supported for Tx UDP and RAW sockets
  - In connection mode Tx and Rx traffic should be done after IP is acquired
  - Max Tx payload for Raw packet with IP header is 1460 bytes
  - Max Tx payload for Raw Transceiver (disconnected mode) is 1476 bytes (including data and header)
  - Min Tx payload for Raw Transceiver (disconnect mode) is 14 bytes (including Data and Header)
  - Closing socket should be done in a proper way (for example not to close a socket while there is blocking receive command on it) - a timeout can be used in this scenarios
  - TCP socket keep alive timeout is set to 5Min (non configurable)
- SSL/TLS
  - Elliptic-curve based ciphers (e.g. ECDH) implies a longer connection time
  - Supported modes
    - Up to one Server (Listen Socket and Accept Socket) + Client (Data socket)
    - Up to Two clients (Data socket)
  - Certificate Authority (CA) certificates needs to be installed if server authentication is required
  - Client mode
    - Signature authentication check – must be less or equal to 4096
    - Key exchange and challenge – must be less or equal to 4096
    - Client authentication – must be less or equal to 2048
  - Server mode
    - Signature authentication check – must be less or equal to 2048
    - Key exchange and challenge – must be less or equal to 2048
    - Client authentication – must be less or equal to 2048
  - Packets will be truncated above 1386Bytes (two TCP packets will be transmitted)
- Tx Power
  - Tx power in AP mode takes effect only after reset
- Wi-Fi Direct
  - In Group Owner mode FAST connection policy should be set to TRUE

- Rx Filters
  - BSSID can't be filtered while STA is connected (if filtered will cause disconnection)
- Power Management
  - The device will remain in active after initialization until the host reads all events
- File System
  - Up to 100 user files
  - File size is limited to 1MByte (No error will be returned while trying to create a larger size)
- Setting device Mode
  - Changing the device role (STA<->AP<->P2P) requires to reset the device
  - Setting network configurations after setting the device role (without reset) can lead to system halt
  - Setting the device mode is persistent and SFLASH endurance must be considered on use cases that requires switching between roles
  - Network configuration is applicable to the current role of the device
- Default State - With no other configuration the default state of the device is as follows:
  - STA mode
  - Regulatory domain is US (channel 1-11)
  - Connection policy – AutoStart and AutoSmartConfig
  - DHCP - Enable
- HTTP Server
  - Support HTTP 1.0
  - Built-in ROM WEB Pages
  - Additional WEB pages could be stored on the File System
  - Dynamic content through proprietary Token mechanism (limited to 64 Characters)
- HTTP internal WEB Pages – main limitations
  - Values entered are not validated – for example:
  - Adding longer/short key in password fields (will be accepted)
  - Typing letters in DHCP lease time (instead of numbers)
  - WPA password is requested to be entered in Hex format when it should be ASCII
  - The length of the AP SSID field is limited to 15 characters (instead of 32)
  - The length of the AP Password field is limited to 24 characters (instead 63)
  - The length of the Device name is limited to 15 characters (instead of 32)
  - Adding/configuring Hidden SSID is not supported
- WEP
  - Supporting only WEP open using ASCII pre shared key however a small code can be used to support Hex format (more details and code example included in the programmer's guide)
- WPS
  - Up to 4 seconds delay between association and EAPOL-Start
- SmartConfig
  - Not supported with 5GHz AP (802.11a/n/ac)
  - Not supported for MIMO-capable configuration devices

  - Not supported with non-standard proprietary modulation schemes
  - Only Group 0 is supported in auto start mode
  - In Auto Start Mode the key is transferred not encrypted
- MCU Deep Sleep Mode
  - User Application should handle the behavior that MCU Deep Sleep mode invocation leads to change in Peripheral Clocks (Frequency gets reduced by a factor of ‘2’)
- Serial Flash

  CC3200 supports JEDEC specification compliant Serial Flash devices with 4Kbyte sector size erase. The following parts were validated:

Micron	N25Q128-A13BSE40	128Mbit
Spansion	S25FL208K	8Mbit
Winbond	W25Q16V	16Mbit
Adesto	AT25DF081A	8Mbit
Macronix	MX25L12835F-M2	128Mbit

For MCU specific peripherals capabilities, please refer to Technical Reference Manual .

9 Networking sample applications

The release package includes several sample applications developed for the CC3200 Cortex M4 processor. The applications come with

- Detailed Application Note (readme documents)
- Project files for IAR, CCS or Both as well as GCC makefiles for a few applications

9.1 Antenna Selection

This is a reference implementation for antenna-selection scheme running on the CC3200 MCU, to enable improved radio performance inside buildings

9.2 Connection Policies

This application demonstrates the usage of the CC3200 profiles and connection-policies.

9.3 Send Email

This application sends an email using SMTP to a user-configurable email address at the push of a button.

9.4 Enterprise Network Connection

This application demonstrates the procedure for connecting the CC3200 to an enterprise network.

9.5 File System

This application demonstrates the use of the file system API to read and write files from the serial Flash.

9.6 Get Time

This application connects to an SNTP cloud server and receives the accurate time.

9.7 Get Weather

This application connects to ‘Open Weather Map’ cloud service and receives weather data.

9.8 Getting Started in AP Mode

This application configures the CC3200 in AP mode. It verifies the connection by pinging the connected client.

9.9 Getting Started in STA Mode

This application configures the CC3200 in STA mode. It verifies the connection by pinging the connected Access Point.

9.10 HTTP Server

This application demonstrates using the on-chip HTTP Server APIs to enable static and dynamic web page content.

9.11 MDNS

This application registers the service for broadcasting and attempts to get the service by the name broadcasted by another device.

9.12 Mode Configuration

This application demonstrates switching between STA and AP modes.

9.13 NWP Filters

This application demonstrates the configuration of Rx-filtering to reduce the amount of traffic transferred to the host, and to achieve lower power consumption.

9.14 P2P (Wi-Fi Direct)

This application configures the device in P2P (Wi-Fi Direct) mode and demonstrates how to communicate with a remote peer device.

9.15 Provisioning AP

This application demonstrates the use of the on Chip HTTP server for Wi-Fi provisioning in AP Mode, building upon example application 9.8 above.

9.16 Provisioning with SmartConfig

This application demonstrates the usage of TI's SmartConfig™ Wi-Fi provisioning technology. The *Wi-Fi Starter Application* for iOS and Android is required to use this application. It can be downloaded from following link: http://www.ti.com/tool/wifistarter or from the Apple App store and Google Play.

9.17 Provisioning with WPS

This application demonstrates the usage of WPS Wi-Fi provisioning with CC3200.

9.18 Scan Policy

The application demonstrates the scan-policy settings in CC3200.

9.19 SSL/TLS

The application demonstrates the usage of certificates with SSL/TLS for application traffic privacy and device or user authentication

9.20 TCP Socket

The application demonstrates simple connection with TCP traffic.

9.21 Transceiver Mode

The application demonstrates the CC3200 transceiver mode of operation.

9.22 UDP Socket

The application demonstrates simple connection with UDP traffic.

9.23 XMPP Client

The application demonstrates instant messaging using a cloud based XMPP server.

9.24 File Download

This application demonstrates file download from a cloud server to the on board serial Flash.

9.25 Deep-sleep

This application demonstrates the deep-sleep low power mode a networking application.

9.26 Hibernate

This application demonstrates the hibernate ultra-low-power mode in a networking application using an UDP client.

9.27 Serial Wi-Fi

This application implements a wireless terminal over a Wi-Fi network and an "AT commands" like control interface

9.28 Out of Box Application

This application demonstrates the out of box experience where user can view different demo and SDK web links on their web-browser.

9.29 Wi-Fi Audio

This application demonstrates Bi-directional wireless audio on a CC3200 LaunchPad. It uses two LaunchPad boards in STA mode and streams the audio from one LaunchPad to the other.

9.30 Camera Application

This example demonstrates image capture using the CC3200 fast parallel interface

9.31 Sensor Profile

This application shows how to use hibernate mode through the Power Management Framework

9.32 Idle Profile

This Application exercises low power modes (LPDS) using Power Management Framework (middleware).

9.33 Watchdog System Demo

This application illustrates full system recovery, using watchdog, including network subsystem.

9.34 Websocket Camera

This application illustrates Websocket HTTP Server functionality with camera JPEG streaming to HTML 5 based web client

9.35 TFTP Client

This application illustrates Trivial File Transfer Protocol client by reading/writing file on TFTP server

9.36 HTTP Client Demo

This application illustrates the usage of HTTP Client library to enable the device as an HTTP Client

9.37 Idle Profile (Non OS)

This application exercises the low power modes (LPDS) using Power Management Framework in a Non OS environment.

9.38 MQTT Client

This application showcases the device acting as a MQTT client in a fully functional MQTT network

9.39 MQTT Server

This application showcases the device acting as an MQTT Server capable of managing multiple local clients as well as allowing the local clients to communicate with remote MQTT clients.

9.40 OTA Update

This application illustrates Over-The-Air (OTA) update of Service Pack, User application and User files.

10 MCU sample applications

10.1 LED Blink

This application uses the GPIO DriverLib APIs to blink an LED on the CC3200 Launchpad.

10.2 Timer Demo

This application demonstrates the CC3200 timer DriverLib APIs. It uses 16 bit timers to generate interrupts which in turn toggle the state a GPIO driving LEDs.

10.3 Watchdog Demo

This application demonstrates the use of the Watch Dog timer (WDT) DriverLib APIs. It shows how the watchdog timer resets the device on system failure.

10.4 UART Demo

This application uses the UART DriverLib APIs to demonstrate typing echo on a UART terminal.

10.5 Interrupt Application

This application uses the Interrupt DriverLib APIs to demonstrate the CC3200 MCU interrupt preemption and tail-chaining capabilities.

10.6 I2C Demo

This application uses the I2C DriverLib APIs to read and write the I2C peripherals on the CC3200 LaunchPad.

10.7 MCU Sleep-DS

This application demonstrates the Sleep and Deep Sleep modes of the CC3200 MCU.

10.8 uDMA Application

This application uses the UDMA DriverLib APIs to show various DMA mode functionalities.

10.9 FreeRTOS Demo

This application demonstrates using FreeRTOS for multiple task creation and inter-task communication using queues.

10.10 AES Demo

This application uses the AES Driverlib APIs to exercise various AES encryption modes.

10.11 DES Demo

This application uses the DES Driverlib APIs to exercise various AES encryption modes.

10.12 CRC Demo

This application uses the CRC Driverlib APIs to exercise various CRC calculation modes.

10.13 SHA-MD5 Application

This application uses the SHA-MD5 Driverlib APIs to exercise various SHA-MD5 modes.

10.14 ADC Demo Application

This application demonstrates the C3200 ADC module using the Driverlib APIs.

10.15 PWM Demo

This application demonstrates the PWM mode of the CC3200 General Purpose Timers (GPT).

10.16 SD Host Demo

This application demonstrates the functionality of the Secure Digital Host (SD Host) controller of CC3200, which interfaces with standard SD memory cards in 1-bit transfer mode.

10.17 SDHost FatFS Demo

This application uses the FatFS library for block level read/write access to SD card, using the SD Host controller on CC3200.

10.18 SPI Demo

This application shows the initialization sequence that enables the CC3200 SPI module in full duplex 4-wire master and slave modes.

10.19 UART DMA

This application demonstrates using the CC3200 UART interface with uDMA and interrupts.

10.20 Timer Count Capture

This application demonstrates measuring the frequency of an external signal using the CC3200 Timer count capture feature.

10.21 Application Bootloader

This application showcases the secondary bootloader operations to manage updates to application image.

10.22 Dynamic Library Loader

This application exercises an approach to enable dynamic loading of an application-binary from non-volatile memory while the program is being executed.

11 Revision History

SDK Version	Updates from previous version
1.1.0	**New Additions:** • Examples - idle_profile_nonos, mqtt_client, mqtt_server, ota_update, dynamic_lib_loader, application_bootloader, http_client_demo • Libraries - netapps/json, netapps/mqtt, netapps/http/client, simplelink_extlib **Version Updates:** • Updated Network Processor Host Driver to version – 1.0.0.10 • Updated Driverlib source to version 1.1.0 **Simplelink:** • Host Driver Changes are listed in section 12 • Added bug fix to ensure correct operation of SPI transfer by using a combination of DMA and CPU transfers depending upon the byte alignment of the input buffer. • Added macro DISABLE_DEBUGGER_RECONNECT • To be defined for deployment scenarios and also if the system can be reset between debugger sessions. • Added new library build configuration – Simplelink_Non_OS_PM **Driverlib:** • Updated files are: aes.c/.h, camera.c/.h, i2s.c/.c, i2c.c, pin.c, prcm.c/.h, sdhost.c, uart.c, des.c, timer.c/h • Please refer to the section 13 for the detailed changes. **Netapps:** • Fixed memory leak issues in http server. **OSlib:** • Added task_yield in osi_MsgQWrite in osi_tirtos.c. **Middleware:** • Power management framework ○ Automatic entry into “sleep” low power mode added to power management framework when idling

  - Conditional saving of PSP or MSP performed based on the control register value
- Timer framework
  - In cc_timer.c, bug fix to handle the software timer list handling appropriately.
- Device drivers
  - In rtc_hal.c,
    - RTC timer operations optimized as the register accesses from the 32KHz domain was slower.
    - Slow clock counter read from the 40MHz domain and the workaround to ensure correct value read also incorporated.
  - UART driver updated,
    - Added echo feature
    - Added read termination based on configured termination character.
    - DMA operation bug fix

Reference examples:

- In all relevant applications, updates done to socket event handler due to changes in Host driver
- Removed peripheral reset in crypto examples (AES, CRC, DES, SHA) as the engines are a shared resource
- In camera application, PIN_01 muxed to non-I2C mode (GPIO).
- In interface files, macros updated to ensure the interrupt registration is done using osi* APIs in case of OS application.
- In gpio_if, the naming convention changed to use GPIO naming with GPIO APIs.
- In startup_ccs, macro added to avoid user configured vector table in case of TIRTOS.
- In timer_if, updated API to specify parameter in millisec.
- In udma_if, the channel select condition check corrected.
- In freertos_demo example, modifications done to run the same code with TIRTOS as well.
- In get-weather example, HTTP client APIs used to demonstrate the API usage.
- In idle profile example,
  - Bug fix to correct length of key passed
  - Added sleep clock enable to WDT peripheral
  - Interrupt mode operations made default for UART
- In oob example, updated html pages to indicate correct device version and AP profile parameters.
- In sdhost example, Added internal Pull-ups on Data and CMD.
- In sdhost_fatfs example,

	o Aligned with new FatFS o Fixed corruption issue with large file writes/reads o Added internal Pull-ups on Data and CMD • In sensor profile example, o Bug fix to correct length of key passed o Added sleep clock enable to WDT peripheral o Interrupt mode operations made default for UART • In websock_camera example, o Added pinmux to remove dependency on J2 and J3 jumpers o Enhanced camera interface - added option to set resolution • In wifi_audio_app example, o Improved the I2S interface - added option to configure sampling rate and bits per sample o Enhanced the Audio Codec Interface
1.0.0	• Removing filters while configuring the device to default state • Updated the “file_download” example to remove the use of temporary file • Modified uniflash session files to use the relative paths • Enabled automatic FTDI driver installation capability • Increased SPI Speed • Added netapps folder containing xmpp, smtp, http, tftp protocol library • Added watchdog_system_demo, websocket_camera and tftp_client examples • Modified CCS project files to have include path from SDK ROOT • Changed XMPP server IP Address in xmpp_client application • Changed CCS target configuration cc3200.ccxml file to use CC3200 Debug Interface • Added error handling in all the applications • Reference linker command files added to support 256KB RAM available in production device • Update Network Processor Host Driver to version 1.0.0.1 • Updated Driverlib source to version 1.0.2 • Moved xmpp and email library to netapps folder • Moved AP configuration macro and networking status bit enum to common header file “common.h” • Updated to FreeRTOS version 8.0.1 • TI-RTOS support for IAR and GCC IDEs
0.5.2	Added a function to configure the firmware to default state across all applications.
0.5.1	First Release

12 Host Driver Changes (With respect to SDK 1.0.0 release)

All the APIs are documented with the HTML programmers guide.

12.1 Defines and Structures

1.0.0.10 Release	1.0.0.1 Release	Action
SlGetRxStatResponse_t: ReceivedAddressMismatchPacketsNumber	SlGetRxStatResponse_t: ReceivedPlcpErrorPacketsNumber	Renamed
SlSockEvent_t	SlSockEvent_t	Changed(replaced by union and internal variable definition)

12.2 Additional changes

- Fix volatile qualifiers to remove warning in some environments (for example Arc)
- Add const qualifiers to allow using constant values

13 Driverlib Changes (With respect to SDK 1.0.0 release)

All the APIs are documented with the Driver library user's guide.

Files	Issue	Fix
aes.c/.h	Use case Issue	Added IV get API – void AESIVGet() Fixed AESDataProcess()
camera.c	-	Removed CAM_BT_CORRECT_EN from comment.
camera.h	Wrong macro value	Fixed macros CAM_HS_POL_LO and CAM_PCLK_FALL_EDGE
i2c.c	Compiler Warning	Removed warning when DEBUG is defined
i2s.c	Enhancement	Added support for 8-bit slot sizes - I2S_SLOT_SIZE_8
	Rotated bits in 24-bit slot	Fixed issue with 24-bit slot size
	Enhancement	Added support for I2S slave
	Enhancement	Added Support to set active slots within frame - I2SRxActiveSlotSet() & I2STxActiveSlotSet
i2s.h	Enhancement	Added new macros
	Wrong macro value	Fixed I2S_PORT_CPU macro
	Enhancement	Added new function declarations I2S_MODE_MASTER I2S_MODE_SLAVE
pin.c	ADC requires any internal pulls disabled	Modified PinConfigSet() to disable internal pulls, if any, when configuring analog pins
prcm.c/.h	Enhancement	Fixed PRCMLPDSEnter() for F Parts Added new API to configure the Camera Clock
	-	Added API to read RTC from fast, 40MHz domain, interface PRCMSlowClkCtrFastGet()
	Typo in function call	Fixed compilation issue in PRCMCC3200MCUInit() for CC3200_ES_1_2_1
sdhost.c	Wrong block count set Wrong Interrupt Status	Fixed SDHostBlockCountSet() Fixed SDHostIntStatus()
uart.c	-	Fixed compilation error when DEBUG is defined
	-	Corrected the Doxygen comment for UARTModemControlSet()
timer.c/.h	Enhancement	Added new API to set the current value of timer
Des.c		Fixed DESDataProcess()

14 Fixed items in this release (With respect to SDK 1.0.0)

14.1 Issues fixed in Sample Applications

ID	MCU00009442
Title	Security key length in idle profile application is wrong
Description	Security key length was referring to the size of the pointer(to key), rather than length of the key

ID	MCU00010334
Title	Uart driver for Power Management Framework in DMA mode is not working
Description	Uart driver is still using the interrupt mode irrespective of the mode(interrupt or DMA) specified by the user

ID	MCU00009444
Title	High Resolution timer in Power Management Framework is not working properly
Description	Using High resolution periodic timer is causing the system to loop unexpectedly.

ID	MCU00010534
Title	OOB setup page input field character limitation bugs
Description	Limits imposed by OOB page - 24 character “Password” limit for AP mode configuration - 15 character limit for “Device Name” field - Invalid description for “Security Key” field

14.2 Wi-Fi issues fixed in this release

ID	MCS00131961
Title	Scan: sl_WlanGetNetworkList does not return APs that their SSID contains only a single character
Description	AP with one character SSID are not presented in the Scan list. No problem to connect using explicitly command or profiles

ID	MCS00131623
Title	AP: SPI Big Endian Mode - missing Endianness conversion of the IP field in sl_NetCfgSet()
Description	In SPI Big Endian only sl_NetCfgSet() didn’t worked correctly due to incorrect conversion in the code

ID	MCS00131560

Title	Host (SPI) driver synchronization lost might occur if read operation is interrupted for a long period of time
Description	In case of the Read operation from the Host is been interrupted by a higher interrupt for a long duration the Host and device can get out of Sync – Interrupting the communication between the Host and Device in the middle of transaction should be avoided

ID	MCS00131573
Title	Host: Unsupported error codes appeared in wlan.h
Description	wlan.h contains list of error codes, which some of them are not supported.

ID	MCS00131575
Title	Host: sl_NetCfgGet might returns -2001 when getting the MACaddress of the device
Description	While getting the MAC address the NWP sends 8 bytes but the request was for 6 bytes (size of MAC). The result is trimmed to 6 bytes and error is returned although the value in the buffer contains the correct MAC address. Fix was provided in the Host Driver side.

ID	MCS00131717
Title	Host: Missing const qualifiers for some API parameters
Description	it's a common use case to have a const string in the code and pass it to functions

ID	MCS00131648
Title	Host: Driver may get stuck on ARC compiler by default due to inconsistency of volatile definition
Description	The ARC compiler has a non-standard mode on by default where any volatile variable is accessed as "uncached". This may cause issues if the same memory is accessed both as volatile pointer and without the volatile qualification as the compiler will access directly (bypassing the cache) on some accesses and through the cache on others (causing a cache coherency issue). The following data structures/pointers introduce this problem. g_pCB g_StatMem

ID	MCS00131508
Title	Host: Wrong command complete detection observed during HTTP_GETTOKEN_VALUE event handling
Description	wrong detection of the command complete can happen If the Host gets an HTTP_GETTOKEN_VALUE event after sending any command which requires a command complete response (and before its command complete received)

ID	MCS00131372
Title	Host: PlcpErrorPackets count definition has wrong meaning
Description	In SlGetRxStatResponse_t structure (wlan.h) the PlcpErrorPackets variable name was incorrectly used and renamed to AddressMismatchPacket in order to reflect the correct meaning
ID	MCS00131613
Title	File System: Updating a file while doing High traffic transfer in Rx transceiver mode can lead to a corruption of the file
Description	A file corruption can occur if traffic is been receiving in transceiver-mode socket while updating a file. The issue occur only in transceiver mode (disconnect)

ID	MCS00131703
Title	HTTP: TCP Timeout while constant access to the HTTP server
Description	A TCP socket timeout might happen due to traffic starvation caused by constantly accessing the internal HTTP

14.3 Documentation issues fixed in the release

ID	MCU00009234
Title	SWRU369: Typo in the CC3200 Programmer guide
Description	Description for JTAG mode includes putting on the SOP2 jumper, which conflicts with the marking on the Launchpad.

ID	MCU00010068
Title	Need to add info about mcuimg.bin in the document
Description	Information about mcuimg.bin needs to be added in Getting started Guide

15 Errata

The following section covers known issues and performance limitations with CC3200 pre-production and production devices.

15.1 Hardware

15.1.1 Pre-regulated 3.3v to Pin 47

For preproduction devices connect an external pre-regulated 3.3v +/- 5% supply to pin 47 (VDD_ANA2). This adds 12mA average current and up to 100mA peak current over 20uSec to the total system current at 3.3V.

The CC300 LaunchPad version 3.2 already includes the correct supply configuration for the pre-production device and also adds a 10uF capacitor to filter the peak currents. No further action is required.

The external 3.3V supply is not required in the CC3200 production device in which case pin 47 can be left not connected. If pin 45 is used in the production device as a GPIO, pin 47 has to be connected to any on-board power supply.

15.1.2 Power consumption increase

Power consumption of the CC3200 pre-production device in all active modes is 1-2 mA higher compared to the CC3200 production devices

15.1.3 RAM size

The production devices have 256 Kbyte RAM available while pre-production devices have 192 Kbyte RAM available.

15.1.4 MCU Low-Power Deep Sleep (LPDS)

Pre-production devices do not support LPDS mode.

15.2 Peripherals driver library

On CC3200 pre-production devices the peripheral driver library needs to be compiled and executed from RAM. On CC3200 production device, the peripheral driver library is burned in the device ROM leaving more RAM space for user applications.

15.3 Network Processor Performance

Item	Pre-Production device	Production device
MCU-NWP SPI link	13 MHz	20 MHz
Init time from hibernate until device ready*	250 mSec	75 mSec
Init time from hibernate until WPA2 connection*	300 mSec	120 mSec
Maximum UDP throughput, open socket	13 Mbps	16 Mbps
Maximum TCP throughput, open socket	11 Mbps	13 Mbps
Maximum TLS/SSL throughput with RC4_128 cipher	5 Mbps	9 Mbps
Maximum TLS/SSL throughput with AES_256 cipher	7 Mbps	12 Mbps
Minimum TLS/SLL connection time with ECC cipher	2.5 Sec	1.3 Sec
Minimum TLS/SSL connection time with RSA cipher	200 mSec	130 mSec

* Excluding user application boot time

15.4 Wi-Fi known issues

ID	MCS00130040
Title	WiFi Direct Reliability: 65% Success rate when Peer device is initiator of connection
Description	Negotiation with peer device is not always successful at first attempt
Impact	The first connection doesn't success
Workaround	Try to connect again

ID	MCS00123349
Title	WiFi Security: CC3100/CC3200 Supports only WEP with Key Index 0 (==> AP Key index 1)
Description	When using WEP security – only WEP index 0 is supported
Impact	Can't use more than one key in WEP security
Workaround	None

ID	MCS00106970
Title	WiFi Security: Traffic Stop while WPA EAP-TLS Enterprise and Reauthentication enabled
Description	In WPA EAP-TLS security the traffic stopped when Reauthentication packet is received

Impact	Traffic stopped
Workaround	Disabled Reauthentication or set it to a very long time

ID	MCS00131174
Title	Scan: Results list contain duplicate networks
Description	The SimpleLink might returns duplicate networks when the network list is not totally filled and the get scan results ask for fewer entries than what was actually found.
Impact	duplicate networks in Scan results list
Workaround	Read the maximum entries at once (20 entries) or to read one by one starting from the end to the beginning and check for duplicates. Once a duplicate was found the list is completed

15.5 Networking known issues

ID	MCS00127876
Title	sl_NetAppDnsGetHostByName returns with no answer
Description	In high Rx traffic conditions some DNS packets can be dropped, causing GetHostByName to fail
Impact	No answer on request
Workaround	Run the API again

ID	MCS00128959
Title	DHCP: SL continues using its previous IP address if an invalid IP in the DHCPACK (before lease time expired)
Description	DHCPACK arrives to SL with invalid address in the DHCPACK params address field but also the IP destination is the same invalid address (MAC address is the valid SL address). SL does not listen to other IPs address as destination but his own therefore this DHCPACK is not processed and SL continue to use his old address until the lease time expires
Impact	The device will continue to use the previous IP address
Workaround	N/A

ID	MCS00128353
Title	UDP/RAW socket data payload is limited to MTU size
Description	Tx IP Fragmentation is not supported for UDP and RAW Tx
Impact	Packet bigger than MTU size will lead that portion of the packet will be discard
Workaround	Use packet size <= MTU size

ID	MCS00131564

Title	NS: SL_ENOBUFS when trying to open 4 TCP server sockets while the internal HTTP server is running
Description	While the HTTP server is running one of the TCP server is been used and limit the number of user TCP Servers
Impact	Only 3 TCP servers can be used while the HTTP is running
Workaround	Disable the internal HTTP Server if 4 TCP Server need to be used

ID	MCS00131966
Title	NS: blocking accept on secure socket doesn’t return
Description	procedure: open secured socket bind listen select on socket => select not return when other side connected
Impact	Select doesn’t return
Workaround	Don’t use select method for accept on secure socket

ID	MCS00131612
Title	Transceiver mode: Changing Channel for RAW Socket is not possible if opened for receive
Description	Changing the channel while a RAW socket is open to receive by using SetSockOpt command can halt the Host. The command response on SetSockOpt doesn’t return. As a result, the host is might get stuck if it configured to blocking mode
Impact	Host get stuck
Workaround	Close the socket and open it again with the correct channel

ID	MCS00129407
Title	NS: SL device should discard datagram with problem in IP Header
Description	If the gateway or host processing a datagram finds a problem with the header parameters such that it cannot complete processing the datagram it discards the datagram
Impact	The SL device sends ICMP reply message
Workaround	N/A

15.6 Host driver known issues

ID	MCS00127283
Title	Free RTOS OS is not stable when running UDP traffic and Ping
Description	Known issue with free RTOS that can cause deadlock
Impact	Deadlock in OS

Workaround	Use TI RTOS

ID	MCS00130291
Title	WPS PIN Connect failure if pin code is not null-terminated
Description	If the PIN code from the HOST is not null terminated connection can fail in some cases
Impact	Connection doesn't succeed
Workaround	Add null termination to the PIN code string

ID	MCS00131563
Title	Host: Set/Get time is limited up to year 2038
Description	Due to a bug in Standard time library that is limited to signed 32-bit integer, and this number is interpreted as the number of seconds since 00:00:00 UTC on 1 January 1970
Impact	Certifications that are bounded by date will expire after year 2038
Workaround	N/A

15.7 Power Management known issues

ID	MCS00128947
Title	In Enterprise network the device will Frequently Wakeup due to IPV4 BRDCST Rx frames
Description	On enterprise network there are lot of BRDCST packets
Impact	Increase in power consumption
Workaround	Add a filter to block the broadcast packets (will be different for each enterprise network)
Remarks	Fix is not expected – the filter is specific to the network

15.8 Network Applications known issues

ID	MCS00128652
Title	HTTP Server: When entering the internal web page with Huawei phone, GUI is zoomed in
Impact	Web page displayed incorrectly
Workaround	N/A

ID	MCS00128658
Title	HTTP Server: GUI is only displayed correctly after refresh in Nexus one phone
Impact	Web page displayed incorrectly

Workaround	N/A

ID	MCS00128130
Title	HTTP Server: Within Dolphin web application cursor is sometimes seen on two rows simultaneously
Impact	Double cursor
Workaround	N/A

ID	MCS00128425
Title	HTTP Server: Default Galaxy Tablet browser shows wrong authentication GUI
Impact	Wrong GUI is displayed
Workaround	Use different browser or disable authentication option

ID	MCS00129384
Title	HTTP Server: GUI - In IE7 browser, GUI border is truncated
Impact	Web page displayed incorrectly
Workaround	Use different browser
Remarks	This is a browser issue, hence fix is not expected

ID	MCS00129385
Title	HTTP Server: On some mobile devices, "WiFi Connectivity" & "Profile Settings" are seen in two lines
Impact	Web page displayed incorrectly
Workaround	Use different browser
Remarks	This is a browser issue, hence fix is not expected

ID	MCS00129390
Title	HTTP Server: On some mobile devices "some parameters were changed, System may require reset" is seen in two lines
Impact	Web page displayed incorrectly
Workaround	Use different browser
Remarks	This is a browser issue, hence fix is not expected

ID	MCS00129392
Title	HTTP Server: On some mobile devices all tabs are merged together in browser
Impact	Web page displayed incorrectly
Workaround	Use different browser
Remarks	This is a browser issue, hence fix is not expected

ID	MCS00129393, MCS00129394, MCS00129397, MCS00129399, MCS00129401
Title	HTTP Server: On some mobile devices lines and tabs are displayed incorrectly
Impact	Web page displayed incorrectly

Workaround	Use different browser
Remarks	This is a browser issue, hence fix is not expected

ID	MCS00130155
Title	HTTP Server: Can't configure the Default Gateway from the HTTP Server pages (with default tokens)
Impact	When working with default HTTP server pages, only default gateway can be used (192.168.1.xxx)
Workaround	Add proprietary token to modify the default Gateway for user pages

ID	MCS00130240
Title	DNS Server: In AP mode the internal DNS Server can't be disabled
Impact	Can't disabled the internal DNS server – can't use external DNS server in AP mode
Workaround	DNS server in AP mode can't be disabled – It can be bypassed using IP UDP Raw socket and disable the DHCP server

ID	MCS00130241
Title	HTTP Server: 'AnyP2P' and 'Auto smart config' policies can be changed only in station or P2P mode
Impact	Can't change these specific configurations from the HTTP server in AP mode
Workaround	Change the configurations in STA mode

ID	MCS00131120
Title	HTTP Server: The System Up Time will get reset after 49Days
Impact	The displayed system up time won't be accurate after 49days
Workaround	Get Time from sl_DevGet SL_DEVICE_GENERAL_CONFIGURATION_DATE_TIME

ID	MCS00132268
Title	NetApp: the Ping response is sent to the Host only on timeout
Description	The Ping response is sent to the Host only on timeout and not when the response was actually received
Impact	The Ping reply received very fast but the Host will have to wait few seconds until it will know that it received correctly
Workaround	Set pingCommand.Flags = 1 - this will return response for every ping

ID	MCS00131570
Title	HTTP Server: Version number displayed in hexadecimal instead of decimal
Description	The HTTP Pages display the SW version number in hexadecimal instead of decimal
Impact	Low impact – SW version is not displayed correctly
Workaround	Convert the version numbers to Dec in the HTTP page (user files)

ID	MCS00132159
Title	DHCP Server: Same address is provided if pool is full
Description	When all of the addresses in the DHCP server pool are assigned, it will continue to offer and assign the last address in the pool to new connected
Impact	The DHCP lease time is not kept for the last disconnected STA. Since only one client can connect at a time to the AP the STA will still get an IP and connect
Workaround	NA

ID	MCS00132200
Title	HTTP Server: SSID is limited to 16 characters
Description	From the HTTP web pages only the SSID string is limited to 16 characters instead of 32 characters
Impact	Can't add a SSID string longer than 16 characters from the HTTP using the device tokens
Workaround	Only the device tokens are limited – implementing user tokens for this field can overcome the issue

ID	MCS00132203
Title	HTTP Server: Password key is limited to 32 characters
Description	From the HTTP web pages only the password key is limited to 32 characters instead of 63 alphanumeric characters
Impact	Can't add a password key longer than 32 characters from the HTTP using the device tokens
Workaround	Only the device tokens are limited – implementing user tokens for this field can overcome the issue

ID	MCS00132206
Title	HTTP Server: Sending a page with no checkbox return "HTTP- No Content Length" message appears
Description	The internal web pages of the device returns "HTTP- No Content Length" if no checkbox is set
Impact	HTTP pages design
Workaround	Insuring that the form will never be empty by adding to the HTML form (that is sent via an HTTP POST) an additional input (can be set with type=hidden)

16 Notes for CC3200 MCU Application Developers

16.1 Simplelink build - Deployment Scenarios Vs. Development mode

Development mode:

To be able to support the usage of the reloading of application image using the debugger without having to reset the device (LaunchPad),

- the implementation in the cc_pal* (simplelink) file required a NwpPowerOnPreamble routine to stop networking services and
- Delay in NwpPowerOn() function to be introduced for proper operation.

This was required as a core reset from the debugger will only reset the APPs processor and the networking engine would still be active. Hence, on the next debug session the networking engine has to be gracefully stopped and started again.

This feature is required only in development mode and is also the default in the build configurations simplelink (OS) and Simplelink_Non_OS.

- Undef DISABLE_DEBUGGER_RECONNECT

The macro will be undefined in development mode, when resetting the device while reconnecting the debugger is not possible.

<u>Deployment scenarios:</u>

In case of deployment scenarios, these statements needn't be executed. Hence, this can be disabled by defining the macro "DISABLE_DEBUGGER_RECONNECT" in the simplelink library build settings.

The macro is defined by default in the simplelink library build configurations Simplelink_OS_PM and Simplelink_Non_OS_PM (primarily because the debugger will disconnect on entry into low power modes).

- Define DISABLE_DEBUGGER_RECONNECT

The macro will be defined under the following scenarios:

1. Deployment scenarios

If resetting the LP before every debug session can be afforded (this will keep the development mode closer to deployment scenario)

16.2 Recovering from I2C lock-up scenarios

On the pre-production (XCC3200GZ) silicon parts, there is a known issue of repeated I2C transactions causing an I2C bus lock-up scenario. Further, this is observed only when the power policy of the networking layer allows it to enter low power modes. Recovery from this scenario requires a power cycle of the device. While the cause of the issue is still being debugged, a software workaround to recover from the stuck scenario is proposed here.

Note: **No** such problem has been detected on the production parts (CC3200R1).

Solution: The stuck condition is caused by an extra clock (noise) generated, leading to clock mismatch between the I2C master and I2C slave, with both sides relinquishing the control of the I2C_SDA line. This situation leaves the I2C Slave in hung (wait) state. This workaround make the I2C_SCL line as GPIO and bit bangs the rest of the clock cycles from master to bring the slave out of the hung (wait) state. This is followed by reconfiguration of the pin as I2C_SCL again. The calling function will be made aware of the situation with a specific return value, which can be handled accordingly.
Note: The following code assumes that i2c lines are pin muxed on default pins:
I2C_SCL = PIN_1
I2C_SDA = PIN_2

Add the following functions in your i2c_if. C file:

```
void GeneratePulse(int num, int freq)
{
        int i = 0;
        unsigned char ucPin;
        unsigned int uiPort;
        GPIO_IF_GetPortNPin(10, &uiPort,&ucPin);
        for(i = 0; i < num; i++)
        {
                GPIO_IF_Set(10,uiPort,ucPin,1);
                UtilsDelay(80000000/(freq*24));
                GPIO_IF_Set(10,uiPort,ucPin,0);
                UtilsDelay(80000000/(freq*24));
        }
}

void GenerateStop()
{
        //
        // Configure PIN_02 for GPIO Output
        //
        MAP_PinTypeGPIO(PIN_02, PIN_MODE_0, false);
        MAP_GPIODirModeSet(GPIOA1_BASE, 0x8, GPIO_DIR_MODE_OUT);
        unsigned char ucPin;
        unsigned int uiPort;
        GPIO_IF_GetPortNPin(11, &uiPort,&ucPin);
        GPIO_IF_Set(11,uiPort,ucPin,0);
        UtilsDelay(30);
        GPIO_IF_GetPortNPin(10, &uiPort,&ucPin);
        GPIO_IF_Set(10,uiPort,ucPin,1);
        UtilsDelay(30);
        GPIO_IF_GetPortNPin(11, &uiPort,&ucPin);
        GPIO_IF_Set(11,uiPort,ucPin,1);

}

void ReconfigureI2C()
{
  //
  // Configure PIN_01 for I2C0 I2C_SCL
  //
  MAP_PinTypeI2C(PIN_01, PIN_MODE_1);
  MAP_PinConfigSet(PIN_01, PIN_STRENGTH_6MA, PIN_TYPE_OD_PU);
  //
  // Configure PIN_02 for I2C0 I2C_SDA
  //
  MAP_PinTypeI2C(PIN_02, PIN_MODE_1);
  MAP_PinConfigSet(PIN_02, PIN_STRENGTH_6MA, PIN_TYPE_OD_PU);

  //
        // I2C Init
        //
        I2C_IF_Open(I2C_MASTER_MODE_FST);
}

int I2cEmulation()
{
        I2C_IF_Close();
        //
        // Configure PIN_01 for GPIO Output
        //
        MAP_PinTypeGPIO(PIN_01, PIN_MODE_0, false);
        MAP_GPIODirModeSet(GPIOA1_BASE, 0x4, GPIO_DIR_MODE_OUT);

        GeneratePulse(9, 400000);
        GenerateStop();
        ReconfigureI2C();
        return (WORK_AROUND);
}
```

I2cEmulation() function need to be called inside I2CTransact after timing out of the while loop.

16.3 Patch Required for IAR

For IAR 7.2 or lower, IAR patch provided under tools folder in SDK is required. This dependency is removed for IAR 7.3 or higher. Make sure the IAR patch should not be applied, if working with IAR version 7.3 or higher, as it is not compatible.

If multiple FTDI devices are connected to a PC, which is being used for downloading the code from IAR debugger, a prompt listing all the FTDI connected to the PC will pop up. There is no definite way of identifying the intended target. So, it is preferred that only one device should be connected to the PC while executing the code from the debugger.

16.4 Connecting back debugger on waking up from LPDS

The Test Power Domain is shutdown whenever the system enters LPDS (by default). In order to avoid this and allow for connecting back the debugger after waking up from LPDS, the macro KEEP_TESTPD_ALIVE has to be defined while building the "driverlib" library.

In addition to this step, the networking sub-system has to be placed in active mode by executing the below statement

```
//iRetVal = sl_WlanPolicySet(SL_POLICY_PM, SL_NORMAL_POLICY, NULL, 0);
iRetVal = sl_WlanPolicySet(SL_POLICY_PM, SL_ALWAYS_ON_POLICY, NULL, 0);
```

This is recommended for development purposes only, as it adds to the current consumption of the system.

Note: The debugger can connect back reliably only when the application processor is active.

16.5 Pinmux pre-requisites

16.5.1 Pinmuxing I2C

The pins on which the I2C functionality is made available by default (MODE1) are

- SCL - PIN_1 (GPIO_10) and
- SDA - PIN_2 (GPIO_11)

If the SCL and SDA pins are chosen to be muxed onto any other pins (other than PIN_1 and PIN_2), it is mandatory to mux the PIN_1 and PIN_2 to some other functionality for correct operation (for ex., GPIO (MODE0)).

16.5.2 Pinmuxing UART

When operating in UART_LOAD mode (SOP[2:0] =100), the PIN_55 and PIN_57 is muxed to the UART functionality by default.

- UART0_TX - PIN_55 (GPIO_01, MODE3) and
- UART0_RX - PIN_57 (GPIO_02, MODE3)

If the UART Tx and UART Rx pins are chosen to be muxed onto any other pins (other than PIN_55 and PIN_57), it is mandatory to mux the PIN_55 and PIN_57 to some other functionality for correct operation (for ex., GPIO (MODE0)) – only while operating in UART_LOAD mode.

Note: Muxing the PIN_55 and PIN_57 to some other functionality is not required while working in FUNCTIONAL mode (SOP[2:0]=000) or 2-wire JTAG mode (SOP[2:0]=001). Hence, it is recommended to choose the settings in development mode similar to deployment scenarios. In such cases, the FUNCTIONAL mode is recommended for using the debugger.

16.6 Updating Simplelink spawn task stack size

By default the simplelink spawn task stack size is set to 2048 bytes. While this size is sufficient for the applications available in the SDK, this value may need to be changed depending on the callback implementation/call flows.
In order to change the stack size, the "oslib" library has to be rebuilt with the macro "SPAWN_TASK_STACK" set to the optimum stack size value
(for ex. -DSPAWN_TASK_STACK=3072).

Implementation in the file "osi.h" in oslib is as below:

```
#ifndef SPAWN_TASK_STACK
#define STACK_LEN          (2048) /*Stack Size*/
#else
#define STACK_LEN          (SPAWN_TASK_STACK)
#endif
```

16.7 Configuring Default Settings for Network Processor

In order to execute the applications in the SDK one after another, the networking engine is configured to a default mode by invoking the simplelink APIs in a predefined sequence. This is done to ensure that the implementation of the current application is not impacted by the sequence of APIs invoked by applications prior to this one. In particular some APIs can cause information to be saved in non-volatile memory for use in subsequent power ON cycles.

The function does the following:
- Set the mode to STATION
- Configures connection policy to Auto and AutoSmartConfig
- Deletes all the stored profiles
- Enables DHCP
- Disables Scan policy

- Sets Tx power to maximum
- Sets power policy to normal
- Unregister mDNS services
- Remove all filters

For deployment scenarios, invoking the function "ConfigureSimpleLinkToDefaultState" is not mandatory.

16.8 Implementing Library Callbacks

The user applications implement the callbacks that are registered with the libraries. While using the Simplelink and MQTT libraries, invoking the core library APIs from a callback should be avoided and can lead to lockup scenarios.
It is recommended to signal another task from the callback routines invoked from the library and invoke the core library API calls from that task.

16.9 Serial Flash Access

Serial Flash should only be accessed through the file system APIs. Refer to the "file_operations" example for more information.

16.10 OpenOCD build with Cygwin

With the latest Cygwin version (as of 1st march, 2015), the OpenOCD compilation for the ftd2xx drivers fails with errors. It is recommended to use old version of Cygwin to work around this issue.