CC2650RC Getting Started with Development

From Texas Instruments Wiki

Contents

- 1 Getting started with development on the CC2650RC
  - 1.1 Sample CC2650RC application projects
    - 1.1.1 Getting started using CCS
    - 1.1.2 Getting started using IAR
- 2 Common components used by Bluetooth LE and ZigBee RF4CE
  - 2.1 CC2650RC Key Scanning
    - 2.1.1 Algorithm
    - 2.1.2 Implementation
      - 2.1.2.1 KeyInit
      - 2.1.2.2 KeyConfigure
  - 2.2 Voice Streaming on the CC2650RC
    - 2.2.1 Bluetooth Low Energy
      - 2.2.1.1 Voice over HID Report (VoHoGP)
    - 2.2.2 ZigBee RF4CE
    - 2.2.3 Limitation
  - 2.3 TI-RTOS PDM Driver And Voice Streaming Application API
- 3 Out of the box Software
  - 3.1 Key functions used by the Out of the box software and the supported sample applications
  - 3.2 Boot test sequence
  - 3.3 OAD upgrade using BLE Device Monitor
  - 3.4 Switching between installed images
  - 3.5 Downloads
- 4 References and Links

Getting started with development on the CC2650RC

The following sections are intended to guide developers in getting started with Bluetooth Low Energy and/or ZigBee RF4CE. After choosing Bluetooth Low Energy and/or ZigBee RF4CE, the developer should download and install the BLE SDK and/or RemoTI, respectively. Both SDKs, during the installation process, will install all dependent sub-product(s) (e.g. TI-RTOS for CC13xx/CC26xx) versions if they aren't already present on the workstation. Once the SDK is installed, the developer can get started with the CC2650RC sample applications included in the SDK.

Shown below is a brief table providing an overview of the supported SDKs:

	Bluetooth Low Energy	ZigBee RF4CE
Software Development Kit (SDK)	BLE SDK 2.02.00.31 (http://www.ti.com/tool/ble-stack)	RemoTI 2.00.00.13 (http://www.ti.com/tool/remoti)
Default (recommended) installation directory	Software Development Kit: `C:\ti\simplelink` TI-RTOS: `C:\ti`	
Supported toolchains	IAR Embedded Workbench for ARM 7.50.3 Code Composer Studio 6.1.x (http://processors.wiki.ti.com/index.php/Download_CCS#Download_the_latest_CCS)	
Products included with SDK (Will be installed if not already present)	TI-RTOS for CC13xx/CC26xx 2.18.00.04 SYS/BIOS 6.45.02.31 CC26xxware 2.23.03.17162 TI Drivers CC13xx/CC26xx 2.16.01.13 UIA 2.00.05.50	TI-RTOS for CC13xx/CC26xx 2.16.00.08 SYS/BIOS 6.45.01.30 CC26xxware 2.23.01.16780 TI Drivers CC13xx/CC26xx 2.16.00.08 UIA 2.00.05.50
Stand-alone Flash Programmer	For CC26XX devices use SmartRF Flash Programmer2 (http://www.ti.com/tool/flash-programmer) For CC25XX devices use SmartRF Flash Programmer (http://www.ti.com/tool/flash-programmer)*	
Hardware debugger (emulator)	For development on the CC2650RC a Debug DevPack (http://www.ti.com/tool/cc-devpack-debug) debugger is suggested. To program CC25XX devices, a CC-Debugger (http://www.ti.com/tool/cc-debugger) is required.	

*Used for evaluation purposes to flash pre-built binary .hex files on CC25XX devices.

Sample CC2650RC application projects

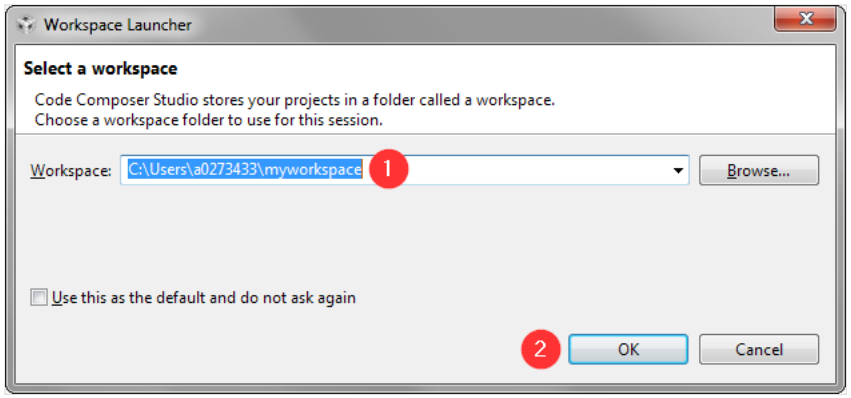
Both the BLE SDK and the RemoTI SDK have sample projects for the CC2650RC and are shown in the table below. This section goes through the basic steps in opening, building and flashing these sample applications onto the CC2650RC.

	BLE SDK 2.02.00.31	RemoTI 2.00.00.13
Directory location	`C:\ti\simplelink\ble_sdk_2_02_00_31\examples\cc2650rc\hid_adv_remote`	`C:\ti\simplelink\remoti_2_00_00_13\examples\cc2650rc\remote_control`
Application project	hid_adv_remote_cc2650rc_app	cc2650rc_app

Stack project	hid_adv_remote_cc2650rc_stack	cc2650rc_stack

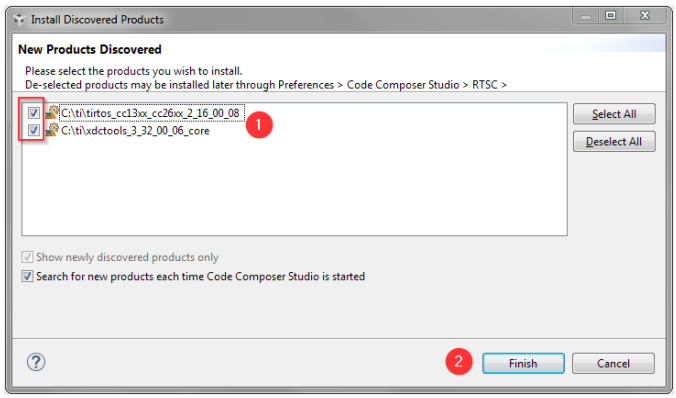
Getting started using CCS

1. Open CCS and select/create a workspace. A "workspace" in context of CCS is a root directory in which one can store one or more CCS projects. In this example, we'll call it *myworkspace*.



---

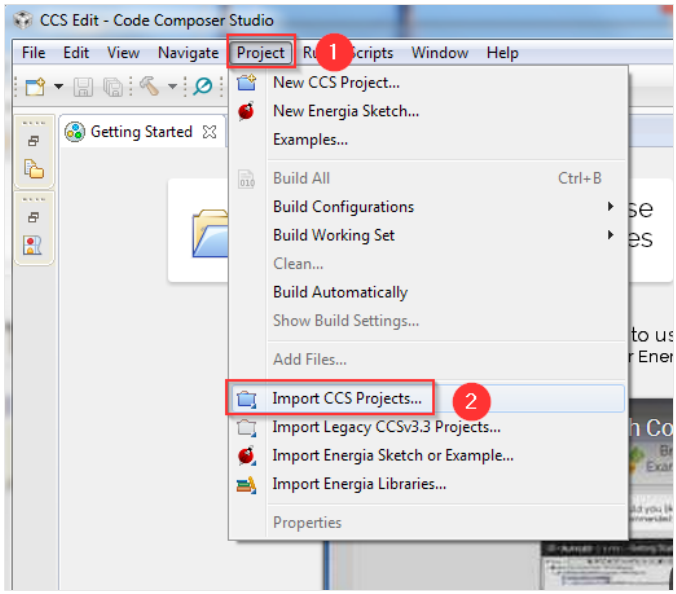
2. Every time CCS starts up it performs a product discovery in the default installation directory. For Windows users, the default installation directory is `C:/ti`. In this step, it is important that the newly installed TI-RTOS and XDCTools products are discovered by CCS. After the discovery process, CCS will prompt for a restart.

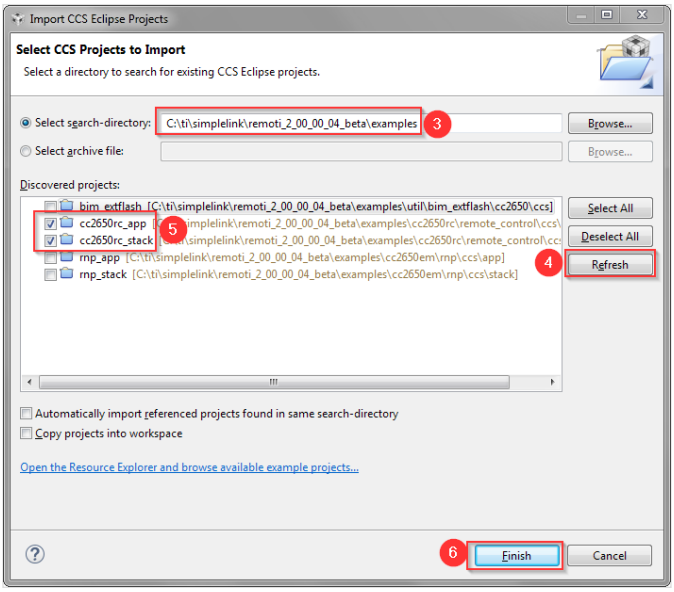


BLE SDK 2.02.00.31	RemoTI 2.00.00.13
TI-RTOS for CC13xx/CC26xx	**TI-RTOS for CC13xx/CC26xx**
`2.18.00.04`	`2.16.00.08`

---

3. After restarting CCS, the next step is to *Import* the example projects included with the SDK. An example consists of an "app" and "stack" projects. You need to select both projects in order for the application to work properly. Each project need to be flashed to the device separately.

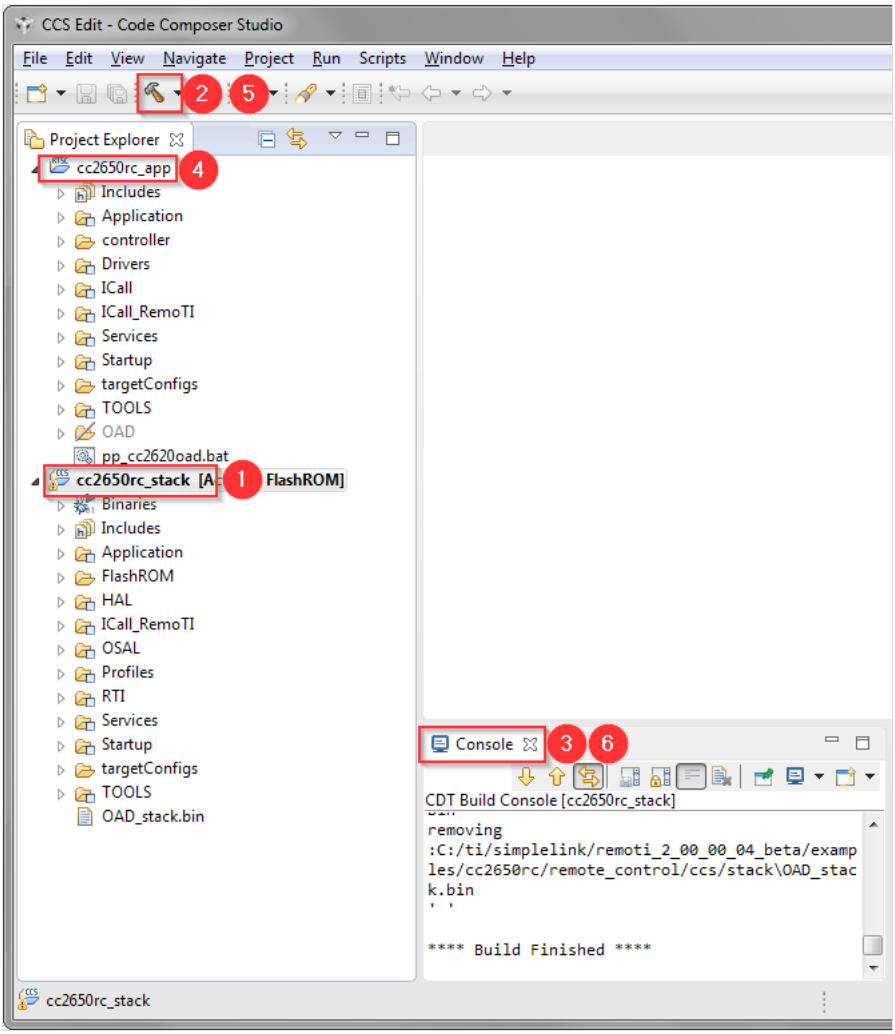




BLE SDK 2.02.00.31	RemoTI 2.00.00.13
Select search-directory	**Select search-directory**
`C:\ti\simplelink\ble_sdk_2_02_00_31\examples`	`C:\ti\simplelink\remoti_2_00_00_13\examples`
Application project	**Application project**
hid_adv_remote_cc2650rc_app	cc2650rc_app
Stack project	**Stack project**
hid_adv_remote_cc2650rc_stack	cc2650rc_stack

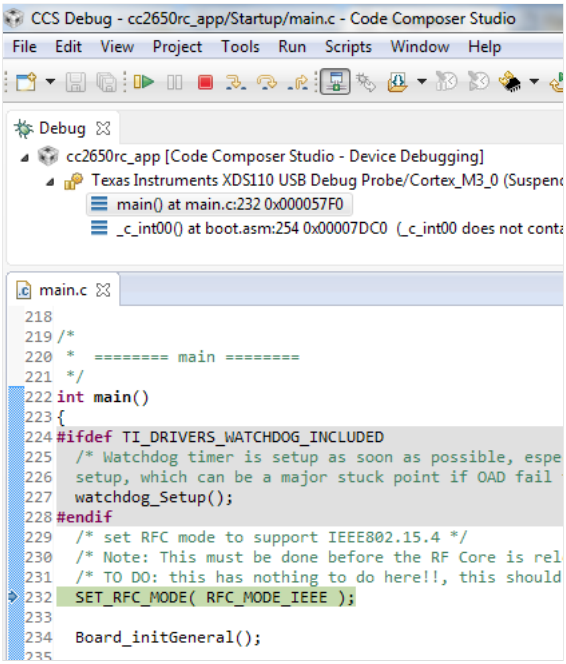
---

4. The next step is to build the imported projects. Select the *Stack Project* and click on  . Observe that the project builds without any errors in the *Console* window. Repeat these steps with the *Application Project*.



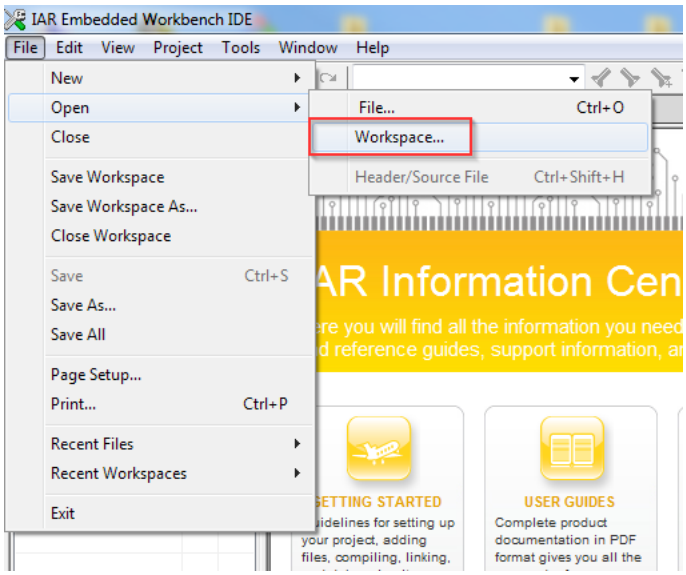
Note, the project names depicted in the image above depend on which software development kit you are using. See step 3.

5. With both projects built, the next step is to load each image (.out) file onto the target. Select the *Stack Project* project and click on . This project does not have a main(), so CCS (and IAR) will complain that it can't run to the main symbol. This is OK. After loading the *Stack Project*, terminate the debug session and repeat the debug steps for the *Application Project* project.



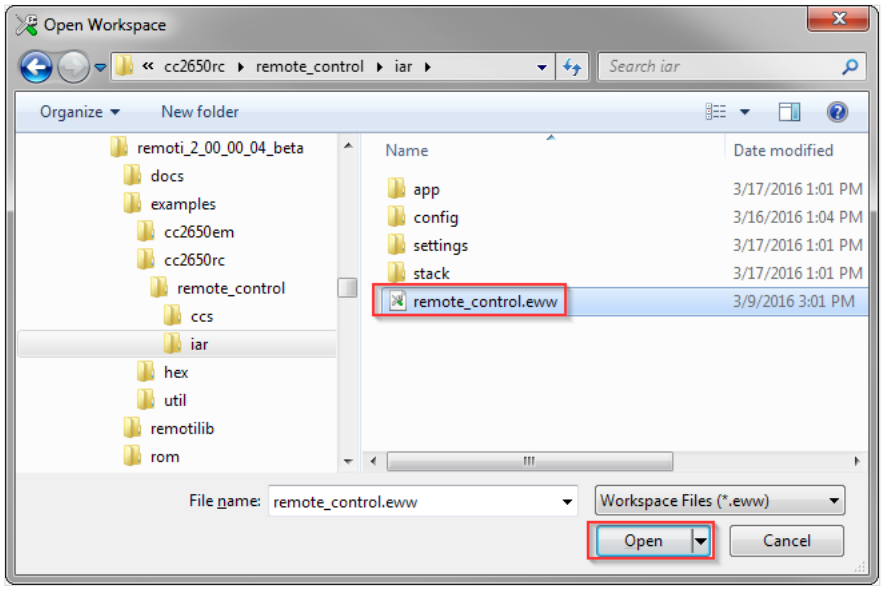
Getting started using IAR

1. Open IAR Embedded Workbench and go to File->Open->Workspace.



---

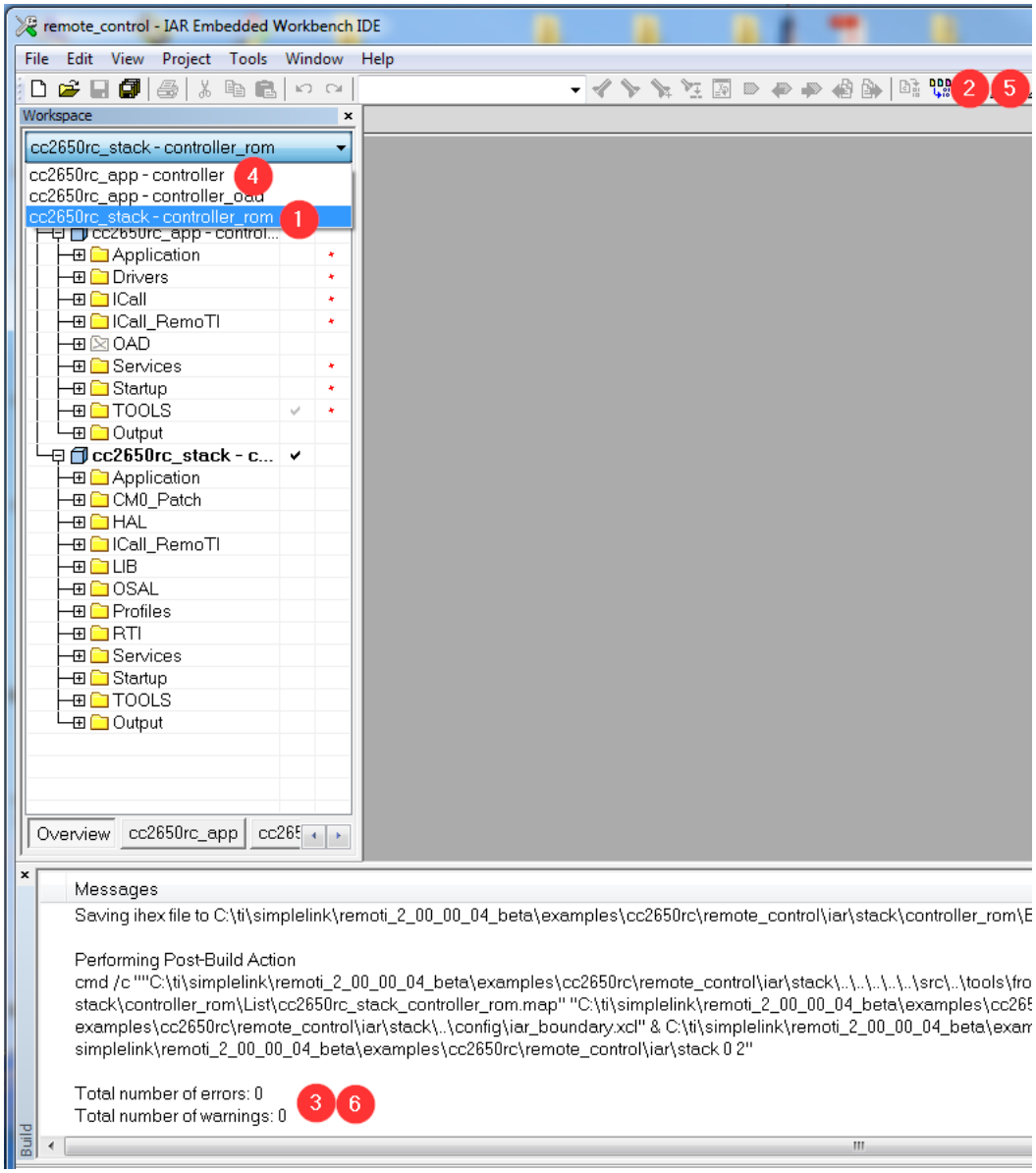
2. Navigate and open the IAR workspace for the software development kit you've chosen.



BLE SDK 2.02.00.31	RemoTI 2.00.00.13
Workspace directory	**Workspace directory**
`C:\ti\simplelink\ble_sdk_2_02_00_31\examples\cc2650rc\hid_adv_remote\iar`	`C:\ti\simplelink\remoti_2_00_00_13\examples\cc2650rc\remote_control\iar`
Workspace	**Workspace**
hid_adv_remote.eww	remote_control.eww

---

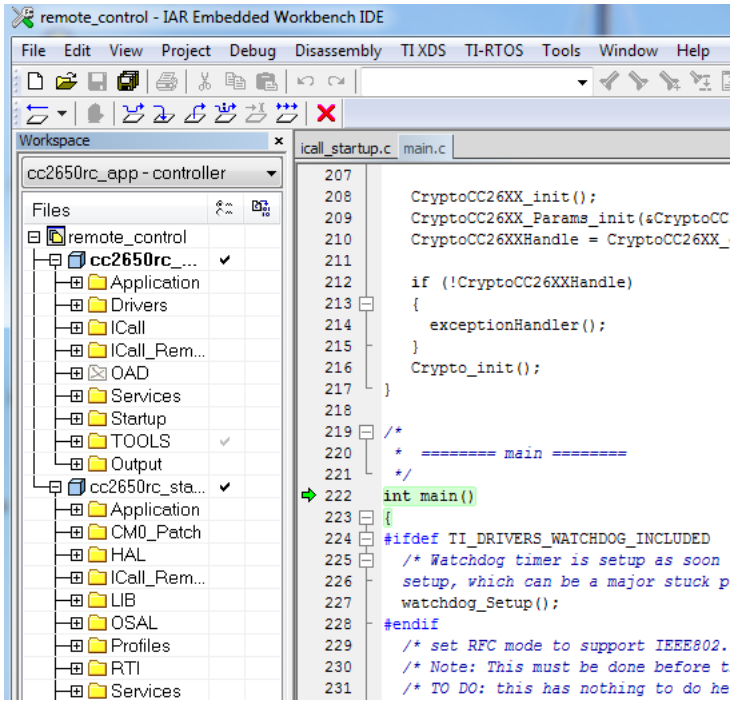
3. The IAR Embedded Workspace includes 2 projects; an *Application Project* and a *Stack Project*. Select the *Stack Project* and click on  . Observe that the project builds without any errors in the *Build output* window. Repeat these steps with the *Application Project*.



Note, the project configurations depicted in the image above depend on which software development kit you are using.

Embedded Workbench project configuration	BLE SDK 2.02.00.31	RemoTI 2.00.00.13
Application Project	cc2650rc_app - FlashROM	cc2650rc_app - controller
Stack Project	cc2650rc_stack - FlashROM	cc2650rc_stack - controller_rom

4. With both *Stack Project* and *Application Project* built, the next step is to load each image (.out) file onto the target. Select the *Stack Project* and click on . This project doesn't not have a main(), so IAR (and CCS) will complain that it can't run to the main symbol. That is OK. After loading the stack, terminate the debug session and repeat the debug steps for the *Application Project*.



Common components used by Bluetooth LE and ZigBee RF4CE

The following sections discuss software and hardware components of the CC2650RC that are used with both BLE SDK and RemotTI SDK.

CC2650RC Key Scanning

This section describes the key scanner on CC2650RC.

There are 32 buttons on the CC2650RC that are scanned with a 3x11 matrix.

- 3 columns
- 11 rows

To save IOs for the developer we use to shift registers to extend the number of IOs. With two shift registers we get 11 IOs controlled with 3 IOs. With the shift register we use only 6 IOs. Thus, we save 6 IOs. The spare IOs are connected to the DevPack connector.

Note that these 32 buttons could have been scanned with 12 IOs in a 6x6 matrix.

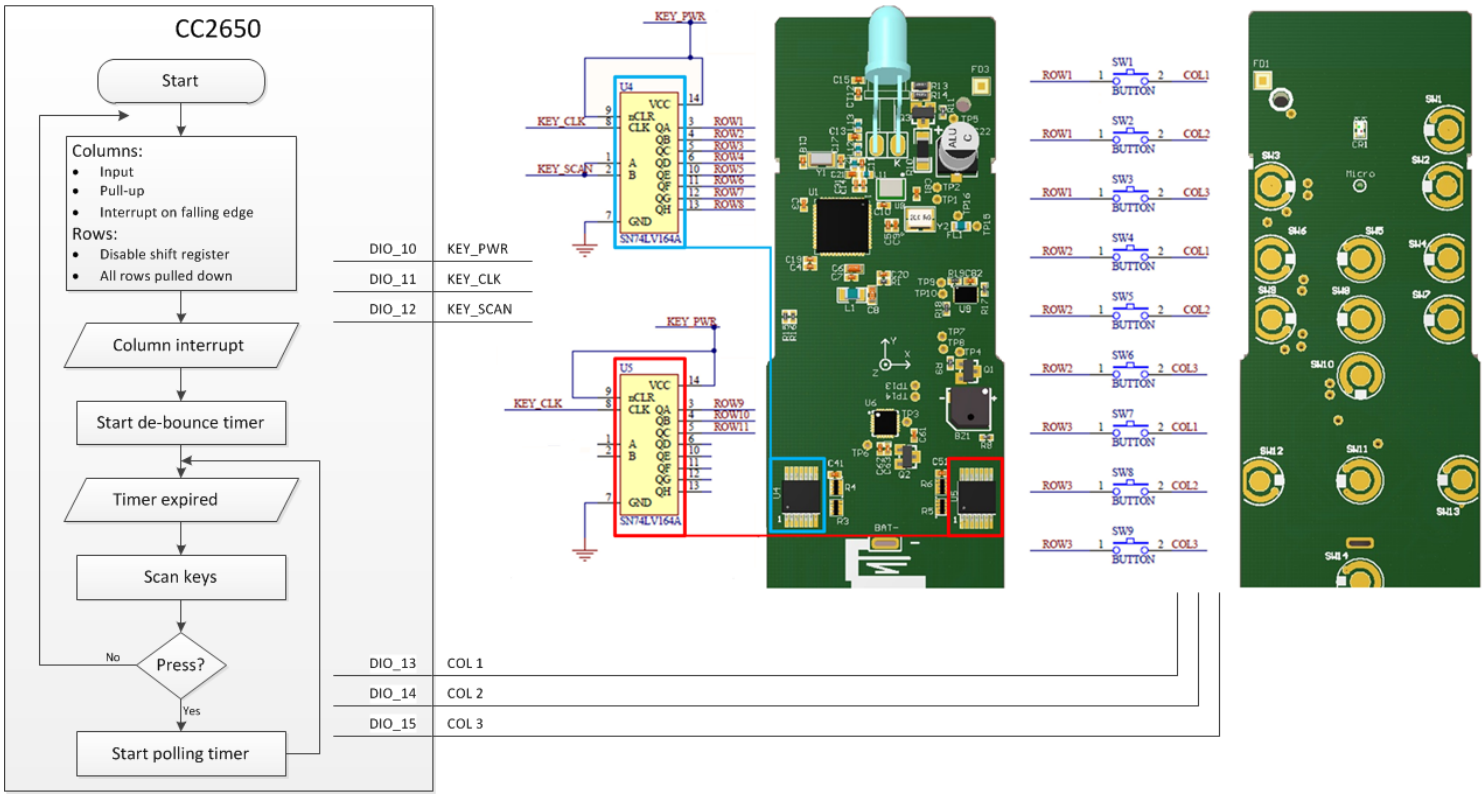
- 6 columns
- 6 rows

Algorithm

As shown in the diagram below there are 3 IOs for columns and 11 IOs for rows. When the remote is in standby the IOs are configured such that a button press generates an interrupt. The interrupt triggers on the column pins.

1. Interrupt detected
2. start de-bounce timer
3. key scan is repeated periodically as long as a press is detected.

De-bounce
: refers to unintentional changes in voltage, or that the voltage "bounce". This typically happens as the button is pressed. As the button shorts the connection between row and column the voltage can "bounce". We don't want to scan the keys as this happens. Thus, the keys are only scanned after some time has passed. This time is greater than the time it takes for the voltage to settle.



Implementation

The key scan algorithm discussed is implemented in the module `key_scan.c/.h`. This module has two public functions:

1. `KeyInit(void)`
2. `KeyConfig(KeyEvtCBack_t key_cback, uint16_t initialKeyRepeatInterval, uint16_t debounceTime, uint16_t pollRate)`

KeyInit

This function must be called before `KeyConfig()` is called. It initializes and configures all IOs. It also constructs the timer resources the key scanner requires.

KeyConfigure

This function is called to register callbacks and configure timer values. The parameters are as follows:

1. `debounceTime`, de-bounce period
2. `pollRate`, scanning period.
3. `initialKeyRepeatInterval`. It determines how frequent the callback `key_cback` is called. If a key is kept pressed for a longer duration, then the callback is called every `initialKeyRepeatInterval`/`pollRate` ms.

Voice Streaming on the CC2650RC

Both applications uses the same mechanism for the voice sampling: The input audio data from the digital PDM microphone on the remote control is using a special mode of the Audio Interface hardware module. A PDM driver handles the decimation of the PDM input to a PCM data stream, with a 16kHz sample rate and 16 bit resolution. The PDM driver also encodes the PCM data using a software codec based on IMA ADPCM with a 4:1 compression rate. This is used when streaming the audio over any of the RF protocols. The voice quality has been qualified by **Nuance** and is sufficient for voice recognition solutions. The required application layer throughput is **66.67kbps**, which includes protocol overhead that is independent of RF protocol.

Bluetooth Low Energy

A detailed User Guide for BLE, including a description of the Voice over GATT (VoGATT) audio service can be found on the BLESDK-2.2.x Developer's Guide and User's Guide for Bluetooth 4.2 (http://software-dl.ti.com/lprf/sdg-latest/html/voice/ble_voice.html#ti-voice-profile-vogp%7CBLE-Stack).

To make the voice streaming even more robust in noisy environments, the output power is switched to +5dBm in our Bluetooth Low Energy example application before the streaming starts:

```
HCI_EXT_SetTxPowerCmd(HCI_EXT_TX_POWER_5_DBM);
```

When the audio stream is stopped, the TX output power will be set back to 0 dBm in the Bluetooth Low Energy example:

```
HCI_EXT_SetTxPowerCmd(HCI_EXT_TX_POWER_0_DBM);
```

Voice over HID Report (VoHoGP)

For users that have the CC2640R2 device on the CC2650RC development kit, you can use Simplelink CC2640R2 SDK 1.40 and the Simplelink CC2640R2 BLE Example packs to send Voice over HID Reports. By using Voice over HID-over-GATT-Profile, you can eliminate the need for the CC254x BLE Dongle and use HID reports natively on operating systems that support the HoGP profile.

The Voice of HID-over-GATT-Profile is found the CC2640R2 BLE Example pack (http://www.ti.com/tool/download/SIMPLELINK-CC2640R2-SDK-BLE-EXAMPLE-PACK%7CSimplelink). A *voice_hogp_remote* example that uses voice over HID reports can also be found in the example pack. To demonstrate on how a basic script can extract voice HID reports on Windows 10 see the ble_examples.git GitHub project (https://github.com/ti-simplelink/ble_examples/tree/master). For documentation on the VoHoGP profile, see the VoHoGP chapter under the Voice over BLE section (http://software-dl.ti.com/lprf/blestack-latest).

ZigBee RF4CE

In the RemoTI example, the original output power is set to +5dBm already, therefore there is no need to adjust any TX output power before and after audio streaming.

Limitation

There current limitations for the voice streaming feature in the hid_adv_remote application are described below:

1. The voice streaming is implemented as a one way stream from Peripheral device to Central device
2. 16 bit 16 kHz mono PCM data is the only supported audio format
3. The PDM driver only support digital PDM microphones
4. The PDU size of the audio notifications are fixed at 20 bytes

For more detail about voice streaming, http://www.ti.com/lit/an/swra506/swra506.pdf

TI-RTOS PDM Driver And Voice Streaming Application API

The audio sampling, decimation and encoding is handled inside the PDM driver, this gives a simple application API for this feature. Once the PDM driver has been started from the application, the PDM driver will notify the application, through a callback function, whenever there is an encoded audio frame ready to be sent.

The Voice Streaming API functions used in the hid_adv_remote(Bluetooth Low Energy)/remote_control(ZigBee RF4CE) application are described below:

hid_adv_remote (BLE SDK 2.2) functions	remote_control (RemoTI 2.0) functions	Description
HIDAdvRemote_startStreamingVoice	RSA_voiceAction(<pressed>)	Start PDM driver and enable audio streaming to connected device
HIDAdvRemote_stopStreamingVoice	RSA_voiceAction(<released>)	Stop PDM driver and disable audio streaming to connected device
HIDAdvRemote_pdmCB	RSA_PDMCC26XX_callbackFxn	Callback function for the PDM driver
HIDAdvRemote_processPdmData	RSA_processPDMData	Processes audio frame from PDM driver
HIDAdvRemote_transmitAudioFrame	RSA_buildAndSendAudioData	Transmit audio frame over the Bluetooth Low Energy/ZigBee RF4CE connection using the Audio Profile
HIDAdvRemote_transmitAudioStreamCmd	RSA_sendPDMStartReq RSA_sendPDMStopReq	Transmit GATT Notification to indicate start or stop of audio stream/Transmit data

Before opening the PDM driver, the application is responsible for setting up the parameters (`PDMCC26XX_Params`) for the driver. These parameters are passed as an input argument to the PDMCC26XX_open() function:

Parameter	Description
`.callbackFxn`	Callback function for the PDM driver
`.useDefaultFilter`	Set to TRUE to apply default filter
`.decimationFilter`	Filter that is applied if useDefaultFilter is set to FALSE
`.micGain`	Microphone gain (0 dB, 6 dB, 12 dB, 18 dB or 24 dB)
`.micPowerActiveHigh`	Set to TRUE if setting the GPIO high powers the microphone
`.applyCompression`	Set to TRUE to apply compression. Setting it to FALSE allows user to apply own compression scheme
`.startupDelayWithClockInSamples`	Digital Microphone startup delay before input starts
`.retBufSizeInBytes`	Size of returned buffers
`.mallocFxn`	Malloc function pointer
`.freeFxn`	Free function pointer
`.custom`	Not used

Before opening the PDM driver, the application is responsible for setting up the parameters(PDMCC26XX_Params) for the driver. These are sample parameters that are passed as an input argument to the PDMCC26XX_open function in the sample applications.

hid_adv_remote (BLE SDK 2.2)

```
PDMCC26XX_Params pdmParams = {
  .callbackFxn = HIDAdvRemote_pdmCB,
  .useDefaultFilter = true,
  .decimationFilter = NULL,
  .micGain = PDMCC26XX_GAIN_18,
  .micPowerActiveHigh = true,
```

remote_control (RemoTI 2.0)

```
PDMCC26XX_Params pdmParams = {
  .callbackFxn = RSA_PDMCC26XX_callbackFxn,
  .useDefaultFilter = true,
  .decimationFilter = NULL,
  .micGain = PDMCC26XX_GAIN_12,
  .micPowerActiveHigh = true,
```

```
    .applyCompression = true,
    .startupDelayWithClockInSamples = 512,
    .retBufSizeInBytes = BLEAUDIO_HDRSIZE + BLEAUDIO_BUFSIZE,
    .mallocFxn = (PDMCC26XX_MallocFxn) HIDAdvRemote_audioMalloc,
    .freeFxn = (PDMCC26XX_FreeFxn) HIDAdvRemote_audioFree,
    .custom = NULL
};
```

```
    .applyCompression = true,
    .startupDelayWithClockInSamples = 256,
    .retBufSizeInBytes = AUDIO_BUF_COMPRESSED_SIZE+PCM_METADATA_SIZE,
    .mallocFxn = (PDMCC26XX_MallocFxn) RSA_audioMalloc,
    .freeFxn = (PDMCC26XX_FreeFxn) RSA_audioFree,
    .custom = NULL
};
```

Full API documentation for the PDM driver can be found in TI-RTOS' doxygen pages.

- PDM Driver doxygen in TI-RTOS 2.18.00.04 (http://software-dl.ti.com/dsps/dsps_public_sw/sdo_sb/targetcontent/mcusdk/2_18_00_04/exports/tirtos_full_2_18_00_04/products/tidrivers_cc13xx_cc26xx_2_16_01_13/docs/doxyg (Used with BLE SDK 2.2.0)
- PDM Driver doxygen in TI-RTOS 2.16.00.08 (http://software-dl.ti.com/dsps/dsps_public_sw/sdo_sb/targetcontent/mcusdk/2_16_00_08/exports/tirtos_full_2_16_00_08/products/tidrivers_cc13xx_cc26xx_2_16_00_08/docs/doxyg (Used with RemoTI 2.0)

Out of the box Software

After the start-up self test sequence, the remote control operates as a Bluetooth LE device and it does not provide any remote control functionality other than IR commands. The software includes an over-the-air download (OAD) service which is used to update the running software. The buttons on the remote has functionality described in the below table. There are no RF-based remote control commands in the out of box software, although there are IR functionality on some buttons (for Samsung TV). The preferred method of updating the remote control is via Android or iOS. Optionally, it is possible to perform an OAD upgrade using the BLE Device Monitor.

Key functions used by the Out of the box software and the supported sample applications

Key	Out of the Box Software	Bluetooth LE sample OAD application	ZigBee RF4CE sample OAD application
NUM_0	(Red LED)	HID_KEYBOARD_0	
NUM_1	(Red LED)	HID_KEYBOARD_1	
NUM_2	(Red LED)	HID_KEYBOARD_2	
NUM_3	(Red LED)	HID_KEYBOARD_3	
NUM_4	(Red LED)	HID_KEYBOARD_4	
NUM_5	(Red LED)	HID_KEYBOARD_5	
NUM_6	(Red LED)	HID_KEYBOARD_6	
NUM_7	(Red LED)	HID_KEYBOARD_7	
NUM_8	(Red LED)	HID_KEYBOARD_8	
NUM_9	(Red LED)	HID_KEYBOARD_9	
UP	(Red LED)	HID_KEYBOARD_UP_ARROW	
DOWN	(Red LED)	HID_KEYBOARD_DOWN_ARROW	
LEFT	(Green LED) IR_LEFT	HID_KEYBOARD_LEFT_ARROW	
RIGHT	(Green LED) IR_RIGHT	HID_KEYBOARD_RIGHT_ARROW	
Power	BLE Discoverable Mode (Toggle)	None	HDMI_CEC_POWER_TOGGLE_FUNCTION
PLAY/PAUSE	(Red LED)	HID_CC_RPT_PLAY_PAUSE	HDMI_CEC_PLAY
MUTE	(Green LED) IR_MUTE	HID_CC_RPT_MUTE	HDMI_CEC_MUTE
V+	(Green LED) IR_VOLUME_UP	HID_CC_RPT_VOLUME_UP	HDMI_CEC_VOLUME_UP
V-	(Green LED) IR_VOLUME_DOWN	HID_CC_RPT_VOLUME_DOWN	HDMI_CEC_VOLUME_DOWN
REC	(Red LED)	None	HDMI_CEC_RECORD
FAST FORWARD	(Red LED)	HID_CC_RPT_SCAN_NEXT_TRK	HDMI_CEC_FAST_FORWARD
FAST REWIND	(Red LED)	HID_CC_RPT_SCAN_PREV_TRK	HDMI_CEC_BACKWARD
MIC	(Red LED)	Start streaming Voice	
MENU	(Green LED) IR_INPUT_SELECT	None	HDMI_CEC_ROOT_MENU
PAIR	(Red LED)	None	Initiate Pairing/Binding
INFINITE	(Red LED)	None	
STB	(Red LED)	None	
BACK	(Red LED)	None	
OK	(Green LED) IR_SELECT	None	HID_KEYBOARD_RETURN
HOME	Hold 6s for Image Select*		
	(Red LED)	None	
AV	(Red LED)	None	
TV	(Green LED) IR_POWER	None	

*Image Select allows user to load image stored in the external flash. The selection is made by pressing 1,2 or 3.

Boot test sequence

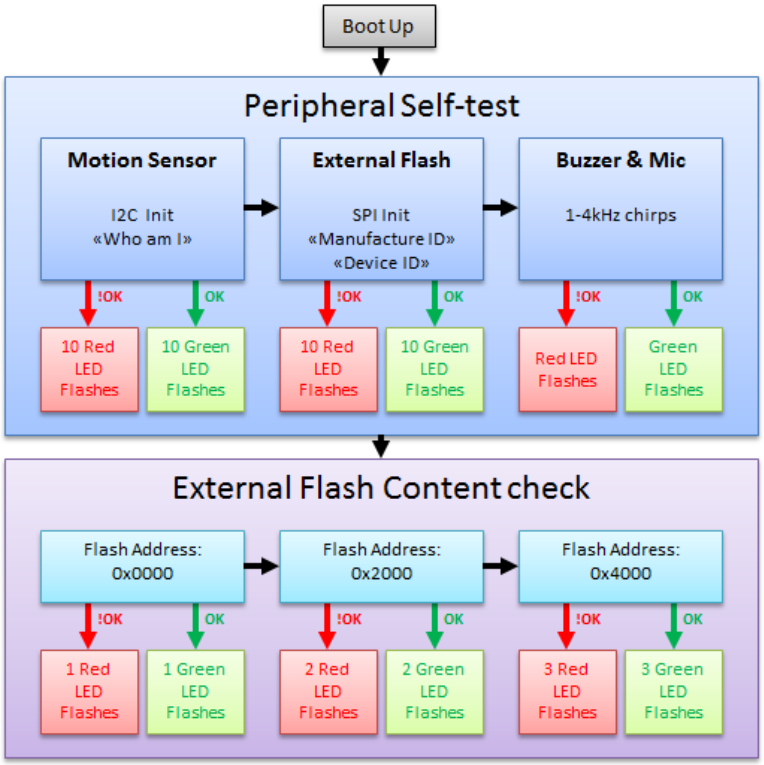
The default software on the remote control has a start-up test sequence. When the batteries are inserted, the remote will enter this self-test to check all peripherals on the PCB. The sequence takes around 9 seconds and the bi-color LED will indicate the various states;

1. MPU9250 self-test: I2C interface is initialized and the “who am I” register is read. MPU9250 is put into sleep mode.
    1. Test OK: 10 green LED flashes
    2. Test not OK: 10 red LED flashes
2. External flash self-test: SPI interface is initialized and Manufacture ID and device ID is read from external flash. External flash is then put into sleep mode.
    1. Test OK: 10 green LED flashes
    2. Test not OK: 10 red LED flashes
3. Buzzer and microphone test. The buzzer makes two chirps, 1kHz-4kHz, with partially normalized volume. (Might fail if PCB is not in plastic casing)
    1. Test OK: green LED flashes on chirps
    2. Test not OK: No LED (Future version: red LED flashes on silence followed by 10 red LED flashes)
4. External flash present images check
    1. External flash address 0x00000
        1. Image present: 1 green LED flash
        2. Image not present: 1 red LED flash
    2. External flash address 0x20000
        1. Image present: 2 green LED flashes
        2. Image not present: 2 red LED flashes
    3. External flash address 0x40000
        1. Image present: 3 green LED flashes
        2. Image not present: 3 red LED flashes

OAD upgrade using BLE Device Monitor

The BLE Device Monitor (http://www.ti.com/lit/zip/swrc258) is a windows application that displays services, characteristics and attributes of any Bluetooth low energy device. For CC2650RC it can be used to download firmware over the air (OAD). The BLE Device Monitor requires a CC2540USB dongle with a HostTestApplication to work. A new CC2540USB dongle is pre-programmed with sniffer firmware so you may have to re-program it with CC Debugger.
Follow these steps to download a new image to the remote:

1. The CC2650RC has to be programmed with the default image (Factory image)
2. On CC2650RC press the power ON/OFF key to toggle advertising on. The green LED will start to toggle.
3. Open Device Monitor and connect to the CC2650RC
4. Go to File -> Program (OAD)
5. Select between Image Type 1 or Image Type 3. Image type translates to where in the external flash the downloaded code will be stored. The factory image is by default Image Type 2, and cannot be overwritten.
6. Click browse and find the appropriate hex image. (It's important to select Image Type before you browse for the hex file)
7. Press Start button in the program field to start OAD.
8. When download is finished, the remote will disconnect and boot with new image.



Also note;

- To download a second image, switch back to the CC2650RC Factory image on the remote (press and hold the Home button) and re-do the process above. Remember to select new Image Type so the first image does not get overwritten.

Switching between installed images

There is only one image stored to the external flash by default (Location 2). It's the factory image which is identical to the out of the box software on the CC2650RC. OAD procedures allow users to add additional images to Location 1 or Location 3 from external flash. Once added, the following steps allow you to switch between the stored images from either Location 1 or Location 3:

- Press and hold the HOME key (1) for 6 s until the red led (2) starts to toggle.
- Release button when toggle stops
- Red led will light up when the remote enters the boot switch menu.
- Choose image you want to load (3).numeric key 1, 2 or 3. (Location 2 is default Factory Image)
- Red led will toggle while new image is copied over to internal flash
- Green led will toggle when the new image boots up

Note that the running software on the remote have to support the mechanism to switch to another image. This is built into the factory image as well as the remote control applications referred on this wiki.

Downloads

- File:CC2650RC Factory Image 2016-04-06.zip
- File:CC2650RC External Flash Erase 2016-04-06.zip
- BLE-HID-VOGATT-DONGLE (CC2540USB Dongle image for BLE demo with voice) (http://www.ti.com/tool/CC2650RC)

References and Links

- CC2650RC Quick Start Guide
- CC2650 BLE SDK 2.2 Bluetooth Low Energy Developer's Guide
- CC2650 RemoTI 2.0 ZigBee RF4CE Developer's Guide

For technical support please post your questions at http://e2e.ti.com. Please post only comments about the article ***CC2650RC Getting Started with Development*** *here.*

Links

Amplifiers & Linear (http://www.ti.com/lsds/ti/analog/amplifier_and_linear.page)
Audio (http://www.ti.com/lsds/ti/analog/audio/audio_overview.page)
Broadband RF/IF & Digital Radio (http://www.ti.com/lsds/ti/analog/rfif.page)
Clocks & Timers (http://www.ti.com/lsds/ti/analog/clocksandtimers/clocks_and_timers.page)
Data Converters (http://www.ti.com/lsds/ti/analog/dataconverters/data_converter.page)
DLP & MEMS (http://www.ti.com/lsds/ti/analog/mems/mems.page)
High-Reliability (http://www.ti.com/lsds/ti/analog/high_reliability.page)
Interface (http://www.ti.com/lsds/ti/analog/interface/interface.page)
Logic (http://www.ti.com/lsds/ti/logic/home_overview.page)
Power Management (http://www.ti.com/lsds/ti/analog/powermanagement/power_portal.page)
Processors (http://www.ti.com

- ARM Proces (http://www.
- Digital Signa (http://www.
- Microcontrol (http://www.
- OMAP Appl (http://www. processors/th

Retrieved from "http://processors.wiki.ti.com/index.php?title=CC2650RC_Getting_Started_with_Development&oldid=230169"