RemoTI Sample Applications User's Guide

Document Number: SWRU201B

Table of Contents

Table of Figures

Table of Tables

Acronyms and Definitions

ACK	Acknowledgement
CC	Chipcon
CERC	Consumer Electronics Remote Controller
EM	Evaluation module
EW	Embedded Workbench
IAR	IAR Systems
LED	Light Emitting Diode
MAC	Medium Access Control
OAD	Over-The-Air Download
OTA	Over-The-Air
PER	Packet Error Rate
RC	Remote Controller
RF	Radio Frequency
RF4CE	Radio Frequency for Consumer Electronics
RNP	RemoTI Network Processor
RTI	RemoTI Application Framework
SB	Serial Bootloader
STB	Set Top Box

1 References

[1] ZigBee RF4CE Specification (ZigBee Alliance document 094945r00ZB)
[2] ZigBee RF4CE CERC Profile Specification (ZigBee Alliance document 094946r00ZB)
[3] RemoTI Target Emulator User's Guide, SWRU202
[4] RemoTI Network Processor Interface Specification, SWRA271
[5] RemoTI API, SWRA268
[6] RemoTI Development Kit Hardware User's Guide, SWRU211
[7] CC253X System-on-Chip Solution for 2.4-GHz IEEE 802.15.4/ZigBee/RF4CE User's Guide, SWRU 191
[8] CC2531 USB Hardware User's Guide, SWRU221

2 Introduction

This user's guide provides the information necessary to run the sample applications included with the RemoTI software installer. This includes a description of each sample application, information for how to program the device and step-by-step procedure to run the sample applications.

Chapter 5 of this document provides information about the memory map for the different sample application configurations, and chapter 6 describes the compiler flags used with the sample applications.

3 Development Environment

3.1 Hardware

The RemoTI development kit includes all the hardware required to run the sample applications. The RemoTI kit contains the following components:

- RC reference design
- Target board with socket for EM module and mini-USB cable
- CC2531 USB dongle
- CC debugger with flat ribbon and mini-USB cable

3.2 Software

The RemoTI software installer includes the software (source and libraries) and IAR projects for each of the sample applications. The following sample applications are included with the RemoTI software installer:

- Basic Remote Controller
- HID dongle
- Network processor
- Over-the-air download
- Serial Bootloader

The RemoTI software installer also includes the following tools:

- Target Emulator
- Over-The-Air download (OAD) demo application
- Serial Bootloader (SB) demo application
- Win32 RemoTI dll

3.3 Development Tools

The sample application project configurations and the libraries shipped with the RemoTI software installer are using the IAR Systems Embedded Workbench toolset. The version required to build the software is described in the readme file of the RemoTI software installer.

IAR Systems provides a 30-day evaluation version of their full-featured toolset. This can be downloaded from www.iar.com. After the 30-day evaluation period expires, you can get a full version of the EW.

3.4 RemoTI Software Installation

The RemoTI software installer is provided for Microsoft Windows based PCs. After successfully installing the software on your computer, you can use the windows browser to examine the folders and files that were places in the *C:\Texas Instruments\ RemoTI-CC2530DK-1.2* folder. The sample applications are located in the *C:\Texas Instruments\ RemoTI-CC2530DK-1.2\Projects\RemoTI* folder as seen in Figure 1.

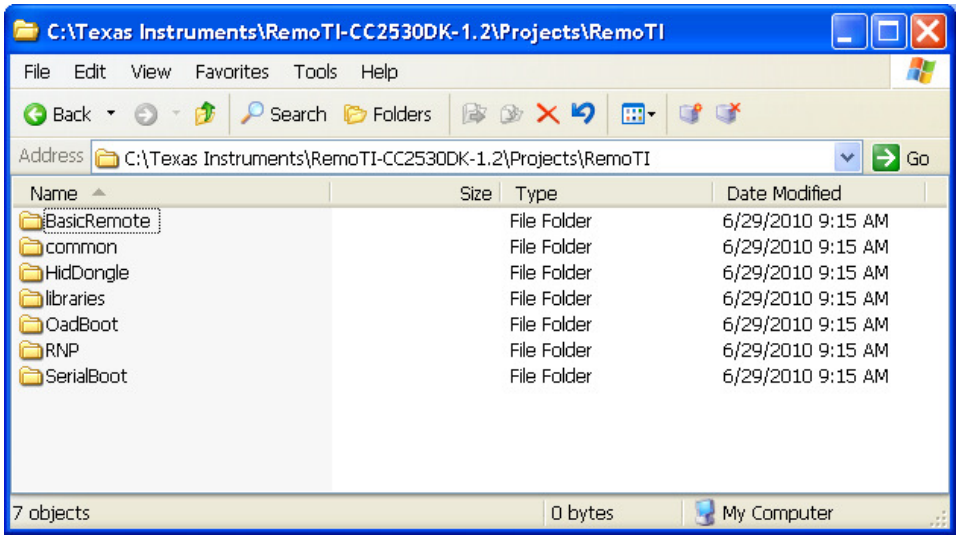


Figure 1. Windows Folder View of RemoTI Sample Applications

4 Sample Applications

This chapter describes the sample applications, how to program the hardware and execute the applications.

4.1 Basic Remote Controller (RC)

The basic RC sample application runs on the RC reference hardware. Its main purpose is to process key inputs and send the corresponding CERC command over-the-air to a target node. The RC is battery powered and it is very important to consume as little current as possible. As such, the RC will be in deep sleep and consume the least possible current, unless it is processing a key press. After the key press is processed and the message is sent, and possibly the ACK is received, it goes back to deep sleep mode.

The sample application is built on top of the RemoTI Application Framework (RTI) which implements the CERC profile [2]. Information about the RTI APIs can be found in the RemoTI API document [5].

The sample application also includes a test mode that can be used to measure Packet Error Rate (PER) and packet latency. This is convenient in the development and test phase of a product to qualify the performance of the product. TI vendor specific command extensions to the CERC profile are used for communication when in this mode.

The RC node can control multiple target nodes, granted a pairing is established with each of them. The sample application is constructed such that it can be paired with different target node device types (TV, STB etc) or with multiple target nodes of the same device type. Media selection keys are used to select the particular target node. These key presses won't send over-the-air messages but configure the sample application to send subsequent messages to the selected target node.

4.1.1 Key Mapping

The basic remote sample application maps some of the RC keys to special purpose operations. These keys are chosen arbitrarily for demonstration purposes only, and should be mapped differently in a production RC. The special purpose keys are listed below. See Figure 2 for a picture of the mapping.

- 'ZOOM' - This key is used to pair the RC with a target node. The key is also used to abort on-going pairing if the key is pressed again during pairing.
- 'FRZ' – This key is used to toggle the RC in/out of test mode. See the chapter 4.5 for details of test mode operations.
- '□' (Stop) – This key clears all the RC pairing table entries if held down for ~5 seconds. A normal key press will send the specific CERC command as usual.
- 'TV' – This key selects the target node initialized as 'Television' (TV) as the recipient of the subsequent key presses. If the RC is paired with multiple TV target nodes, it will toggle between them. The key press has no effect if the RC is not paired with a TV target node.
- 'DVD – This key selects the target node initialized as 'Video player/recorder' (DVD) as the recipient of the subsequent key presses. If the RC is paired with multiple DVD target nodes, it will toggle between them. The key press has no effect if the RC is not paired with a DVD target.

- STB - This key selects the target node initialized as 'Set Top Box' (STB) as the recipient of the subsequent key presses. If the RC is paired with multiple STB target nodes, it will toggle between them. The key press has no effect if the RC is not paired with a STB target node.
- 'AUX' – This key selects the target node initialized as a 'Media Center PC' device type as the recipient of the subsequent key presses. If the RC is paired with multiple PC target nodes, it will toggle between them. The key press has no effect if the RC is not paired with a PC target node.

Note that the RC will send commands to the last selected target node if a target select button (TV, DVD, STB or AUX) doesn't have a valid pairing entry. After a re-boot of the RC e.g. when batteries are replaced, the RC will initialize with the first pairing as the destination of button presses.

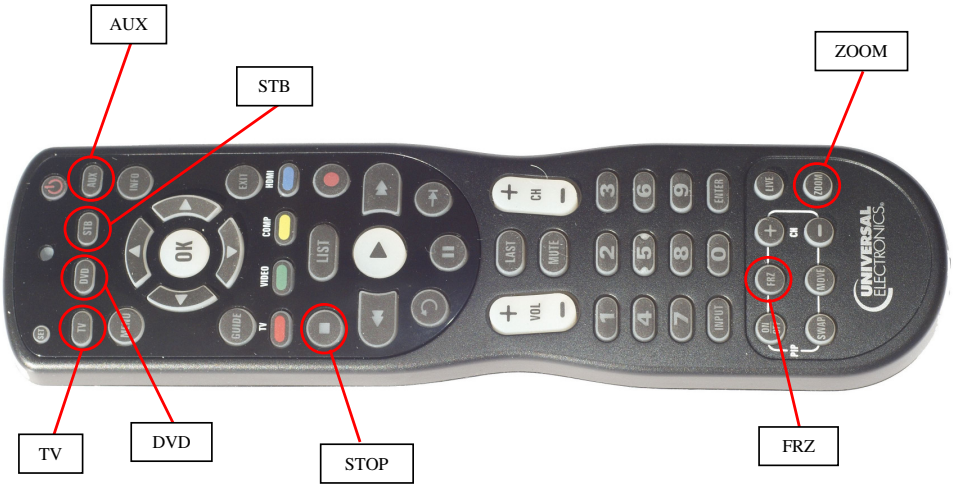


Figure 2. Remote Controller Key Mapping

4.1.2 Building the Sample Application

Start the IAR workbench application, select '*File -> Open -> Workspace…*', browse to the *\Projects\BasicRemote\CC2530RC* directory and select the *rsa_cc2530.eww* file. You should see the workspace as shown in Figure 3.

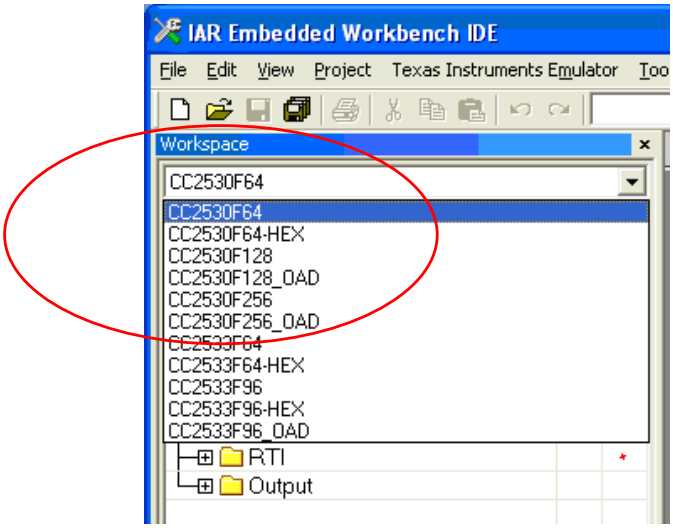


Figure 3. IAR Workbench for BasicRemote Sample Application

The workspace includes multiple project configurations as seen in the drop-down box in the workspace window of Figure 3. Choosing the appropriate project options will include the correct source and linker files, and set the project compile options accordingly. The available project options for a RC without OAD bootloader capabilities are listed below. See chapter 4.7 for instructions for how to build an RC image with OAD bootloader capabilities (configuration options with *_OAD*).

- CC2530F64 – This configuration targets the CC2530 64KB flash device and is using non-banked code model.
- CC2530F128 – This configuration targets the CC2530 128KB flash device and is using banked code model.
- CC2530F256 – This configuration targets the CC2530 256KB flash device and is using banked code model.
- CC2530F64-HEX – This configuration is used to generate a hex file for use by SmartRF programmer for a CC2530 target.
- CC2533F64 – This configuration targets the CC2533 64KB device and is using non-banked code model
- CC2533F96 – This configuration targets the CC2533 96KB device and is using banked code model
- CC2533F64-Hex – This configuration is used to generate a hex file for use by SmartRF programmer for a CC2533F64 target.
- CC2533F96-Hex – This configuration is used to generate a hex file for use by SmartRF programmer for a CC2533F96 target.

See chapter 5.1 for memory map details of these project configurations.

The RC included with the RemoTI development kit is capable of supporting all project configurations listed above. You can as such choose either of the configurations for use with the rest of the instructions in this chapter.

Choose '*Project -> Clean*' and then '*Project -> Rebuild All*' to build the basic remote sample application.

4.1.3 Programming the RC with IAR Workbench

Connect the CC debugger to the RC and the PC. Instructions for how to connect the cables correctly are described in [6]. Verify the LED on the CC debugger is 'green' to indicate a successful connection. If not, press the 'Reset;' button.

You can now program the RC by choosing '*Project -> Debug*' from the IAR workbench. If you don't want the debugger connected while executing, terminate the debug session:

1. Clicking '*Debug->Stop Debugging'* from IAR
2. Remove the flat ribbon cable from the RC

4.1.4 Programming the RC with SmartRF Flash Programmer

The SmartRF Flash Programmer can also be used for programming. The latest version can be downloaded from: http://focus.ti.com/docs/prod/folders/print/cc2530.html.
The hex files can be generated from IAR using a corresponding configuration intended for hex file generation such as CC2530F64-HEX.
In case of a part which does not have a selectable configuration intended for hex file generation, for instance, CC2530F128, use the following instruction to build a hex file.
The linker options of the project configuration must be changed in order to produce a hex file format the flash programmer can load onto the device. Right-click on the project name in bold font in the workspace window and choose 'options…'. Then select the 'Linker' category in the left column and set the output format to 'other' (intel-extended) as shown in Figure 4.

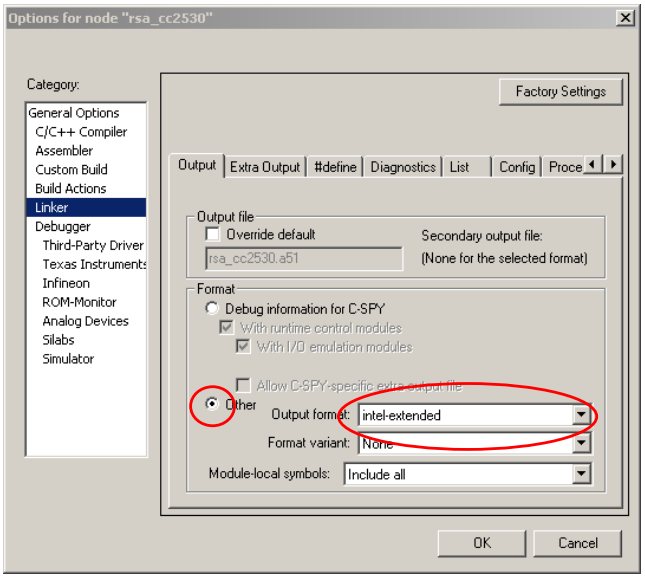


Figure 4. Linker Options for HEX File Generation

The following lines also need to be un-commented in the corresponding linker command file located in the */projects/ RemoTI /common/cc2530* directory:

```
// Include these three lines when generating a .hex file for banked code model:
-M(CODE)[(_CODEBANK_START+_FIRST_BANK_ADDR)-
(_CODEBANK_END+_FIRST_BANK_ADDR)]*\
_NR_OF_BANKS+_FIRST_BANK_ADDR=0x8000
-ww69=i
```

Note that the RemoTI software release also includes a hex image called *rsa_cc2530.hex* in the *C:\Texas Instruments\RemoT-CC2530DK-1.2\bin* folder for the CC2530F64 project configuration.

To program the hex image, connect the CC debugger and start the SmartRF Flash Programmer application. Make sure the CC debugger LED is 'green', if not press the 'Reset key'. Follow the steps below. Each step is highlighted in Figure 5 of the flash programmer below.

1. Verify that the CC253x device is listed in the '*System-on-Chip*' window. If no target is listed, verify the CC debugger is connected and the driver is installed.
2. Select the hex file to download by clicking the '…' button and browse to the file location.
3. Make sure "*erase, program and verify*" is checked.
4. Click the "*perform actions*" to start the programming.

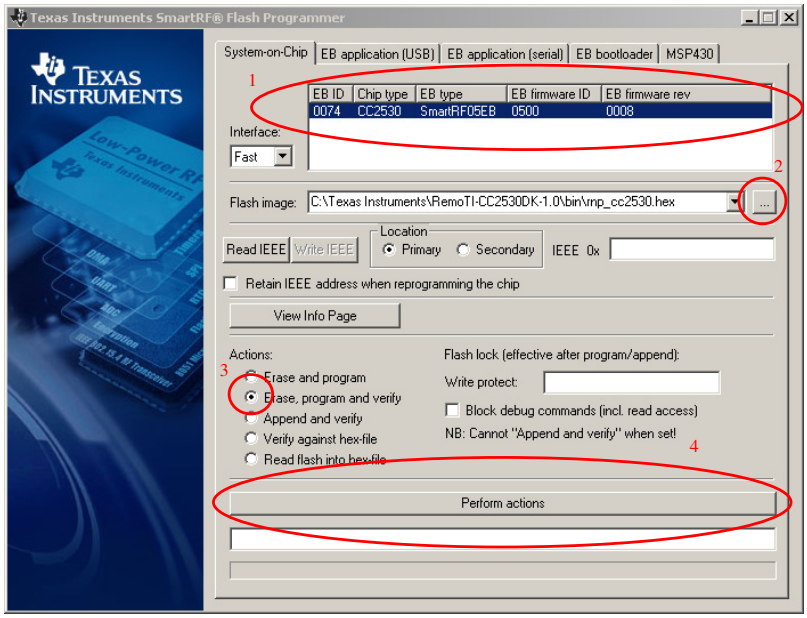


Figure 5. SmartRF Flash Programmer Interface

4.2 Network Processor (RNP)

The RemoTI Network Processor (RNP) sample application runs on the target board included in the RemoTI kit. The RNP sample application also runs on either the SmartRF05EB or the CC2531 USB dongle. For running sample application on the SmartRF05EB board, see the HAL_BOARD_CC253xRB, HAL_BOARD_CC253xEB_REV13 and HAL_BOARD_CC253xEB_REV17 compiler options explained in chapter 6. The sample application enables RTI API remote procedure calls from a host processor through a serial interface. The serial interface can be UART, SPI or USB. The RNP can be instantiated as a controller or a target node.

The Target Emulator application included with the RemoTI software release connects with the RNP over either a UART interface or a USB interface and provides a means to control the RNP and visualize the messages received from the RC. Target Emulator instantiates the RNP as a target node. See [3] for instructions for how to configure and control the RNP target node.

4.2.1 Building the Sample Application

Start the IAR workbench application, select '*File -> Open -> Workspace…*', browse to the *\Projects\RNP\CC2530EB* directory and select the *rnp_cc2530.eww* file. You should see the workspace as shown in Figure 6.

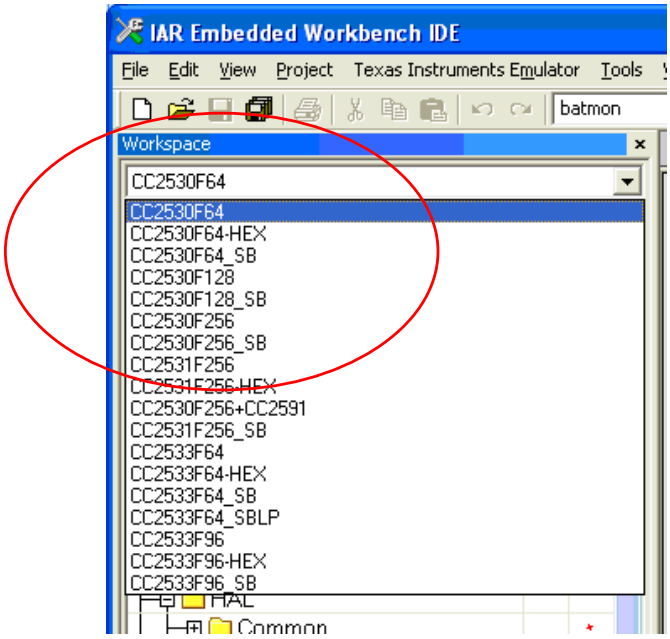


Figure 6. IAR Workbench for the RNP Sample Application

The workspace includes multiple project configurations as seen in the drop-down box in the workspace window of Figure 6. Choosing one of the project options will include the correct source and linker files, and set the project compile options accordingly. The available project options for a network processor without serial bootloader capabilities are listed below. See chapter 4.6 for instructions for how to build an RNP image with serial bootloader capabilities (configuration options with *_SB*).

- CC2530F64 – This configuration targets the CC2530 64KB flash device and is using non-banked code model.
- CC2530F128 – This configuration targets the CC2530 128KB flash device and is using banked code model.
- CC2530F256 – This configuration targets the CC2530 256KB flash device and is using banked code model.
- CC2531F256 – This configuration targets the CC2531 device and is using banked code model.
- CC2530F64-HEX – This configuration is used to generate a hex file for use by the SmartRF programmer, for CC2530 part.
- CC2531F256-HEX – This configuration is used to generate a hex file for use by the SmartRF programmer, for CC2531 part.
- CC2530F256+CC2591 – This configuration targets the CC2530 256 KB flash devices with a CC2591 RF front-end chip and is using the banked code model.

- CC2533F64 – This configuration targets the CC2533 64KB device and is using non-banked code model.
- CC2533F96 – This configuration targets the CC2533 96KB device and is using banked code model.
- CC2533F64-Hex – This configuration is used to generate a hex file for use by the SmartRF programmer, for CC2533F64 part.
- CC2533F96-Hex – This configuration is used to generate a hex file for use by the SmartRF programmer, for CC2533F96 part.

See chapter 5.1 for memory map details of these project configurations.

The CC2530 EM included with the RemoTI development kit is capable of supporting all above-listed project configurations for CC2530, CC2533EM supports the CC2533 while CC2531 USB dongle supports project configurations for CC2531. You can as such choose a corresponding configuration for use with the rest of the instructions in this chapter.

Choose '*Project -> Clean*' and then '*Project -> Rebuild All*' to build the basic remote sample application.

4.2.2 Programming the Target Board with IAR Workbench

Connect the CC debugger to the Target Board and the PC. Instructions for how to connect the cables correctly are described in [6]. Verify the LED on the CC debugger is 'green' to indicate a successful connection. If not, press the 'Reset' button.

You can now program the target board by choosing '*Project -> Debug*' from the IAR workbench. If you don't want the debugger connected while executing, terminate the debug session:

1. Clicking '*Debug->Stop Debugging'* from IAR
2. Remove the flat ribbon cable from the target board

4.2.3 Programming the Target Board with SmartRF Flash Programmer

Follow the instruction in section 4.1.4 to program the target board using the SmartRF Flash programmer.

Note that the RemoTI software release also includes a hex image called *rnp_cc2530.hex* in the *C:\Texas Instruments\RemoTI\bin* folder for the CC2530F64 project configuration.

4.2.4 Programming the CC2531 USB dongle with IAR workbench

Connect the CC debugger to the CC2531 USB dongle. Instructions for connecting debug cable to the dongle are described in [8]. Instead of using SmartRF05EB board, CC debugger can be used in the same way as it is used for a RemoTI Target Board. Verify the LED on CC debugger is 'green' to indicate a successful connection. If not, press the 'Reset' button.

You can now program the target board by choosing '*Project -> Debug*' from the IAR workbench. If you don't want the debugger connected while executing, terminate the debug session:

3. Clicking '*Debug->Stop Debugging'* from IAR
4. Remove the flat ribbon cable from the target board

4.2.5 Programming the CC2531 USB dongle with SmartRF Flash Programmer

Follow the instruction in section 4.1.4 to program the target board using the SmartRF Flash programmer.

Note that the RemoTI software release also includes a hex image called *rnp_cc2531.hex* in the *C:\Texas Instruments\RemoTI\bin* folder for the CC2531F256 project configuration.

4.3 HID Dongle

The HID dongle sample application runs on a CC2531 USB dongle included in the RemoTI kit. The sample application exercises the USB module of the CC2531 chip and provides an example USB HID class implementation. The sample dongle provides interfaces as a HID keyboard device as well as a HID consumer control device and supports USB HID standard reporting as such devices, when controlled by a CERC remote.

The Target Emulator application included with the RemoTI software release is used to trigger pairing and to see diagnostic messages from the dongle, but once the dongle is paired with a remote, it generates the standard keyboard device reports and consumer control device reports when corresponding commands are sent from the paired remote.

4.3.1 Building the Sample Application

Start the IAR workbench application, select '*File -> Open -> Workspace…*', browse to the *C:\Texas Instruments\RemoTI-CC2530DK-1.2\Projects\HidDongle\CC2531USB* directory and select the *hid_cc2531.eww* file. You should see the workspace as shown in Figure 6.

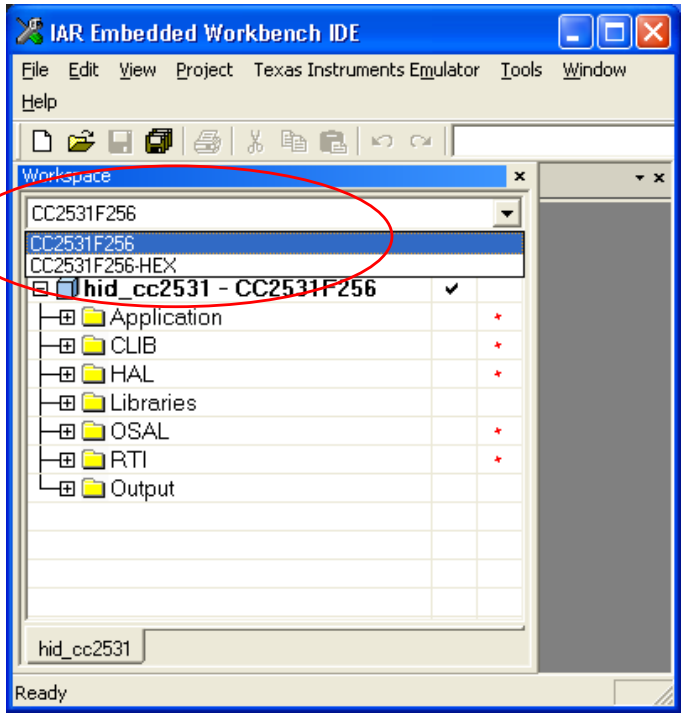


Figure 7. IAR Workbench for the HID Dongle Sample Application

The workspace includes multiple project configurations as seen in the drop-down box in the workspace window of Figure 6. Choosing one of the project options will include the correct source and linker files, and set the project compile options accordingly. The available project options are listed below.

- CC2531F256 – This configuration targets the CC2531 device and uses the banked code model and the configuration generates an image to be used with the IAR debugger.
- CC2531F256-HEX – This configuration targets the CC2531 device and uses banked code model. This configuration generates a hex image to be used with the SmartRF programmer tool.
- CC2531F256_SB – This configuration is used to generate a downloadable image for use with a serial boot loader.

Choose '*Project -> Clean*' and then '*Project -> Rebuild All*' to build the HID dongle sample application.

4.3.2 Programming the CC2531 USB dongle with IAR workbench

Connect the CC debugger to the CC2531 USB dongle. Instructions for connecting debug cable to the dongle are described in [8]. Instead of using SmartRF05EB board, CC debugger can be used in the same way as it is used for a RemoTI Target Board. Verify the LED on CC debugger is 'green' to indicate a successful connection. If not, press the 'Reset' button.

You can now program the target board by choosing '*Project -> Debug*' from the IAR workbench. If you don't want the debugger connected while executing, terminate the debug session:

1. Clicking '*Debug->Stop Debugging'* from IAR
2. Remove the flat ribbon cable from the target board

4.3.3 Programming the CC2531 USB dongle with SmartRF Flash Programmer

Follow the instruction in section 4.1.4 to program the target board using the SmartRF Flash programmer.

Note that the RemoTI software release also includes a hex image called *hid_cc2531.hex* in the *C:\Texas Instruments\RemoTI-CC2530DK-1.2\bin* folder.

4.3.4 HID dongle execution

When HID dongle is plugged into a PC and while the dongle is active, a red LED on the dongle is lit. A green LED displays status of pairing. If any remote was already paired, the green LED is turned on. Otherwise it is turned off. The green LED blinks when pairing procedure is in progress.

When there is no paired remote, i.e., when the green LED is not lit, you have to run the Target Emulator application on PC and operate the Target Emulator and the remote to pair the two devices. See [3] for the Target Emulator operation instructions.

Once a remote is paired, you can continue to use Target Emulator to monitor the reports generated by the USB dongle. With or without the Target Emulator, the USB host PC handles the reports. For instance, keyboard reports triggered by navigation keys on the paired remote (up, down, left, right arrows) is handled in the same way as normal keyboard arrow keys. Numeric keys (0 – 9), volume up/down/mute keys, channel up/down keys and play / pause / stop / rewind / fast forward / skip next / skip backward keys are handled by supporting applications (for example, Windows master volume control, media player, etc.). Note that other keys (e.g. green, red function keys) are not mapped to the HID report and hence pressing non-mapped key does not trigger report log display on Target Emulator.

4.4 RC and RNP System Setup and Program Execution

The sample RC system provided with the RemoTI release is comprised of the RC node, the target node and the target emulator running as an application on a PC as seen in Figure 8. The RC

sends commands to the target node which will work as a bridge to relay the messages up to the PC application over a serial (UART) connection. The target emulator application is used to control the network processor and to highlight which key was pressed by the RC. The target emulator is also used to configure the target node, control pairing etc. and to configure test parameters when in test mode. See chapter 4.5 for test mode details.

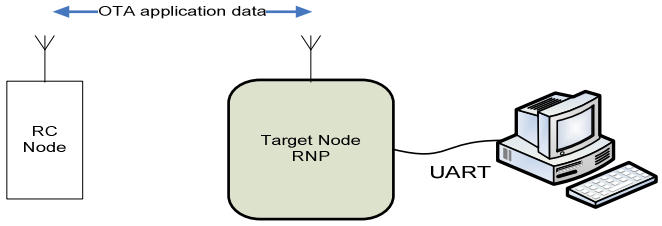


Figure 8. RC and RNP System Setup

CC2531 USB dongle is connected through USB port to the PC but a virtual serial connection is created and hence the overall connection is logically same as in Figure 8. When plugging in CC2531 USB dongle programmed with RNP project for the first time, or when executing the image through IAR for the first time, Windows asks for a driver installation. The driver information file (.inf) is located in Tools\Driver folder under the RemoTI kit installation root folder. Search the driver from the folder accordingly as in Figure 9.

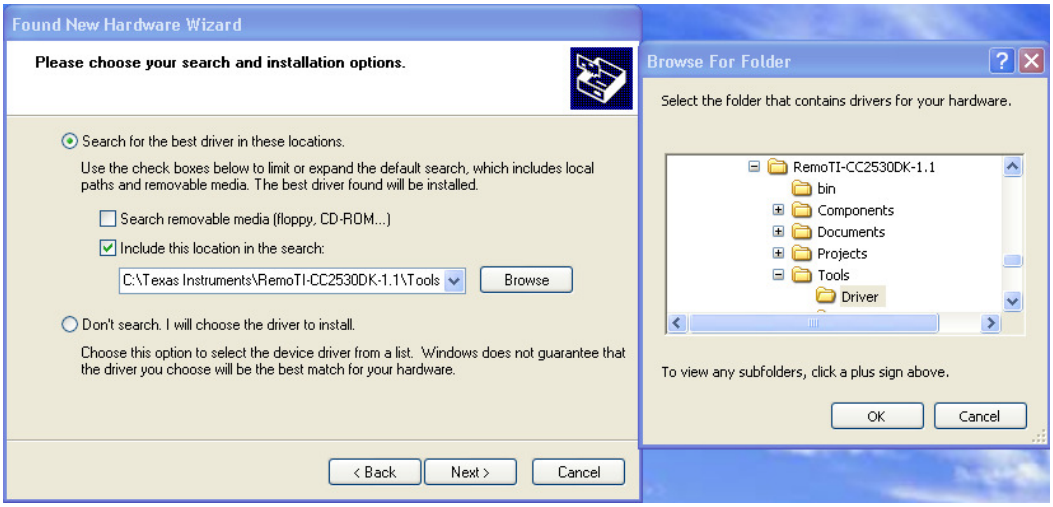


Figure 9. CC2531 RNP Driver Installation

The serial port number for the newly installed CC2531 RNP driver can be found from device manager by identifying RemoTI Network Processor label as in Figure 10.

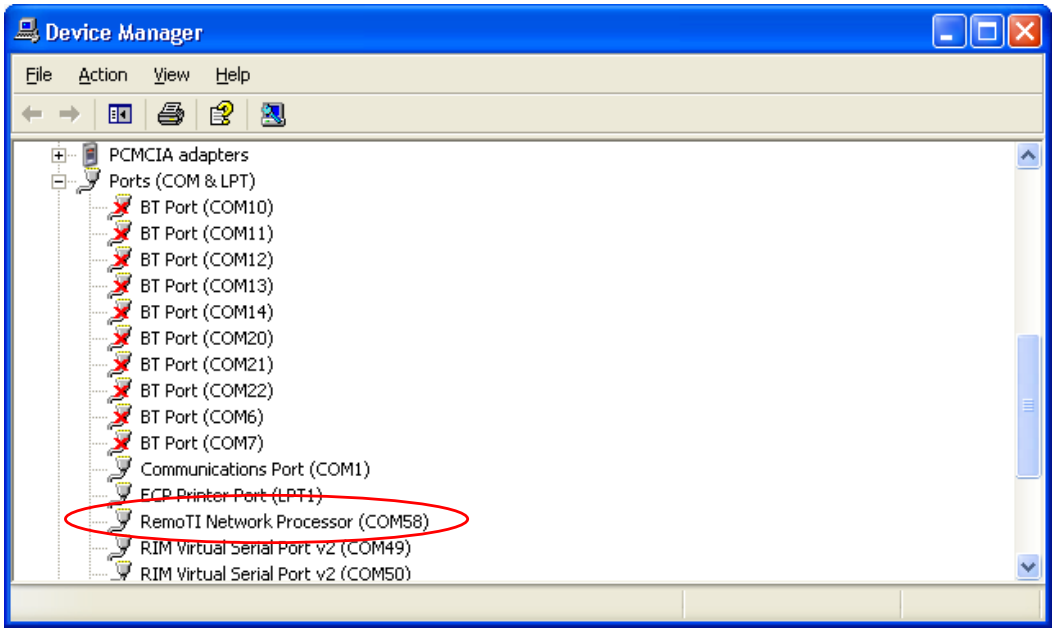


Figure 10. Serial Port number in Device Manager

The target emulator application is described in [3]. Please consult this document for detailed instructions of the actions described in the subsequent sections.

Follow the steps in chapter 4.1 to program the RC and the step in chapter 4.2 to program the RNP.

The basic RC and RNP sample applications implements the CERC profile. As such a one-to-one button press is used to trigger auto-discovery and pairing. Follow these steps to pair the RC with the RNP target node;

1. Star the target node and initialize the UART connection
2. Select' TV', 'Video recorder/player', 'Set-Top-Box' or 'Media Center PC' as the device type
3. Click the target emulator '*pair*' button to enable the RNP node to accept CERC discovery and pairing requests. You know have 30 seconds to complete the discovery and pairing.
4. Press the '*Zoom'* key with the 30 seconds to send the CERC discovery and pair request to the RNP node. Note that the order of step 3 and 4 can be revered. The important point is that the both will be in discovery and pairing 'mode' for 30 seconds.
5. A successful pairing is made when '*Paired controllers*' is incremented by one.
6. You can know send commands to the target device by pressing any key on the RC except the keys listed in chapter 4.1.1. Note, however, that the TV, DVD, STB and AUX keys

are used to select the target node recipient of command when the RC is paired with multiple targets.

7. Press a key and observe the key is highlighted in the remote controller simulator. Note also the 'Received packet count' increases in the target emulator and the packet content is displayed in the log window. Observe also the RC backlight LED is turned on when the command is sent. The command is sent using the unicast, multi-channel w/ACK transmit option and the LED is turned off when an ACK is received or after 1 sec. See [1] for details of this transmit option.
8. You can delete the RC pairing entry by holding the '□' (stop) key for 5 seconds and all RNP node pairing entries by selecting the 'Pairing -> Clear Pairing Info' in the target emulator.

Note that the RC can pair with multiple RNP target nodes and a RNP target node can be paired with multiple RC nodes.

4.5 RC Test Mode

The test mode enables execution of the latency and PER test. The results of the test will be displayed in the target emulator.

The 'FRZ' key is mapped to toggle in/out of test mode to enable the RC to send CERC vendor specific commands and to enable the target to configure test parameters. Figure 11 illustrates the test mode states and how to transition between the different states.

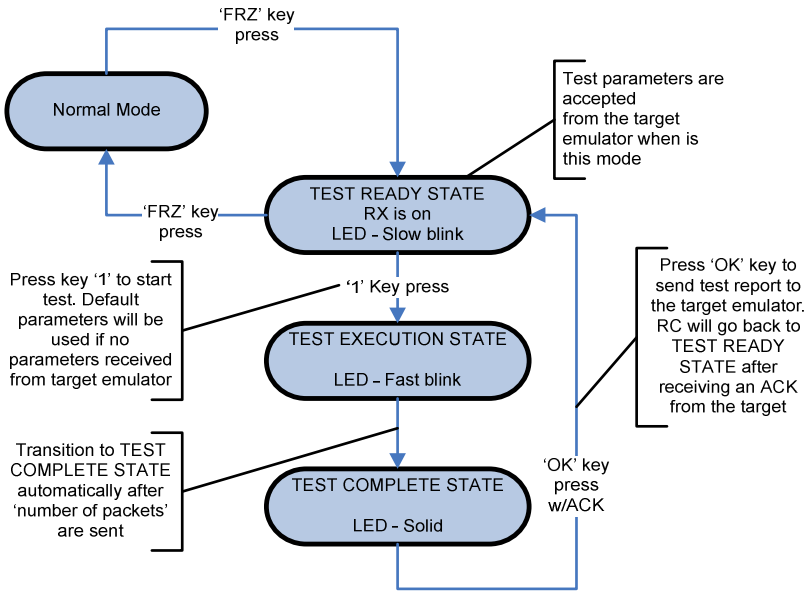


Figure 11. RC Test Mode State Diagram

Note that the RC must be paired with the RNP target node to enable test mode.

4.5.1 Test Mode Parameters

The latency and PER test configuration parameters can be configures in the target emulator and sent to the RC when in test ready state. See [3] for information for how to set these parameters. The RC sample application includes a default set of parameters that will be used if no parameters are received from the target emulator. The parameters and their default values are described in Table 1 below.

Parameter		Description	Default value
numPackets		Number of packets to be sent from RC to target node	1000
maxBackoffDuration		A random delay in ms between successful ACK of a packet to send of next packet	64 ms
txOptions	*Broadcast*	Broadcast packets	-
	IEEE address	IEEE addressing mode	-

	Acknowledged	Packets ACKed	√
	Security	Encrypted packets	-
	Single Channel	Single channel transmission	-
	Channel Designator	Channel normalization	-
	Vendor specific	Vendor specific packet format	√
userDataSize		Number of bytes of the application payload. Effective application payload will be this size plus 1 byte used as a command identifier	8 bytes

Table 1. Test Mode Parameters

4.5.2 Test Execution

Follow these steps to execute the latency and PER test;

1. Pair the RC with the target node. The target emulator device type is not important for the test.
2. Press the 'FRZ' key to transition to TEST READY STATE, and observe the RC LED is slow blinking.
3. Optional: Use the target emulator to configure and send the test parameters to the RC.
4. Press the number '1' key to enter TEST EXECUTION STATE and observe the RC LED is fast blinking. The default parameters in Table 1 are used if step 3 is omitted. Also observe the packets in the target emulator log window as seen in Figure 12 below.

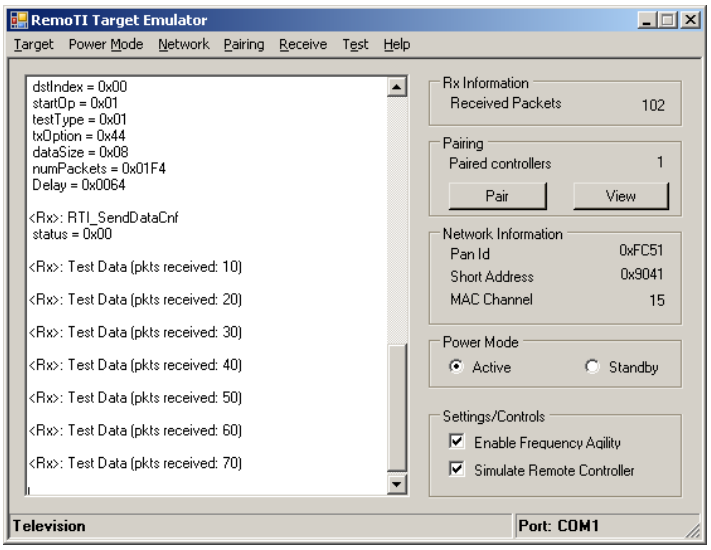


Figure 12. Target Emulator Test Execution Stage

5. The RC is in TEST COMPLETE STATE when the RC LED is solid on. Press the 'OK' key to send the test report to the target emulator. After RC receives an ACK from the target node, it enters TEST READY STATE. Figure 13 below shows an example test report.

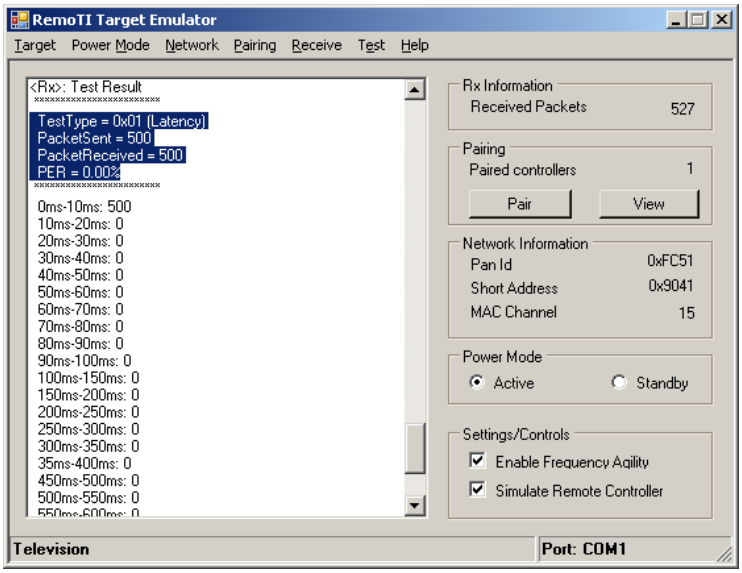


Figure 13. Target Emulator Example Test Report.

4.5.3 Test Results Interpretation

The test report will provide the following information as seen in Figure 13:

- *TestType* : 0x01 is the latency and PER test described in this document
- *PacketSent* : This is the number of packets the RC sent to the target node during execution of the test.
- *PacketReceived* : This is the number of packets the target node received from the RC during execution of the test. This number does not include the test report.
- *X* ms – *Y* ms : This is the number of ACKs the RC received from the target node in the time-span from *X* ms to *Y* ms. In other words, it is the round-trip time elapsed from the RC sends a packet at the application level until an ACK is received at the RC application level.
- *Failed* – This is the number of packets that was sent by the RC and the time elapse before ACK was received is greater than 1 second or the packet wasn't ACKed as all.

- *PER* : Packet Error Rate. This indicated the number of packet the RC sent to the target node that was not ACKed or ACK was received after more that 1 second. The calculation is done on the target emulator using the following formula: *Failed / PacketReceived * 100%*

The RC starts a timer immediately before it sends a packet to the target node. See the *RSA_ProcessEvent()* function and the *RSA_EVT_RANDOM_BACKOFF_TIMER* event in the *rsa_basic.c* file for details of how this is implemented. An excerpt is also shown below.

```
rsaSendReqTimeStamp = osal_GetSystemClock();
RTI_SendDataReq(…)
```

The RC will record the time elapsed when the ACK was received successfully. See the *RTI_SendDataCnf()* function in the *rsa_basic.c* file for details for how this is implemented. An excerpt is also shown below.

```
rsaSendCnfTimeStamp = osal_GetSystemClock();
time = rsaSendCnfTimeStamp - rsaSendReqTimeStamp;
```

4.6 Serial Bootloader (SB)

The serial bootloader sample application runs on the target board included with the RemoTI kit, and is used to demonstrate new stack and application image download over a serial interface. This is a powerful feature used to upgrade the RNP image in the field, e.g. upload a new image to the network processor connected to a STB. The serial bootloader sample application is communicating with the RemoTI serial bootloader demo application on the PC via UART.

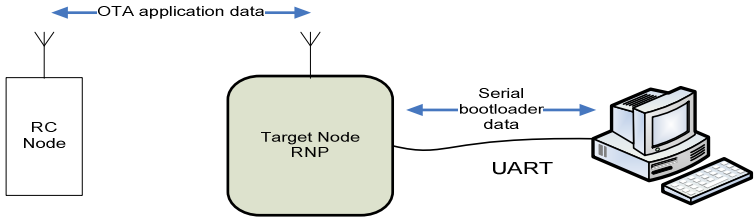


Figure 14. Serial Bootloader System Setup

4.6.1 Building the Sample Application

The serial bootloader and the application image must be built separately. The target node must be programmed with the serial bootloader to be capable of receiving an application image over the serial interface. The serial bootloader does not care if an application image is programmed or not since the serial bootloader itself manages the UART communication for control and new application image payload data.

4.6.1.1 Bootloader

Start the IAR workbench application, select '*File -> Open -> Workspace…*', browse to the *\Projects\RemoTI\SerialBoot\CC2530RC_Src* directory and select the *SerialBoot.eww* file. You should see the workspace as shown in Figure 15.

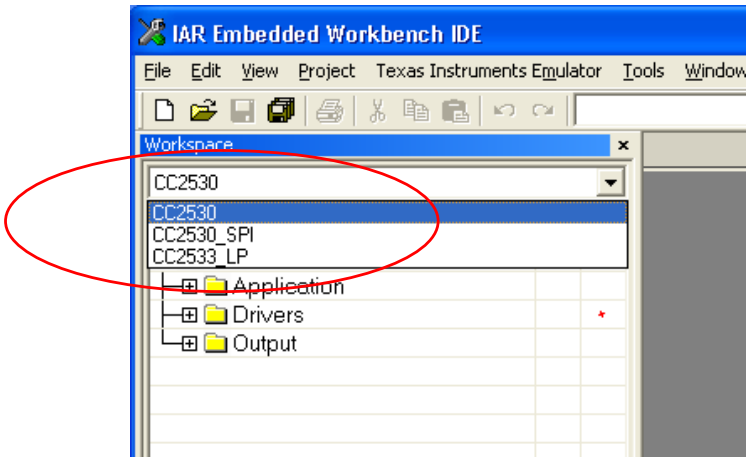


Figure 15. IAR Workbench for the SerialBoot Sample Application

The workspace includes multiple project configurations as seen in the drop-down box in the workspace window of Figure 15. Choosing one of the project options will include the correct source and linker files, and set the project compile options accordingly. The available project options are the following;

- CC2530 – This configuration targets all the CC2530 flash devices and is using UART to communicate with the host processor. The serial bootloader resides on the first flash page and occupies 2KB of code space.
- CC2530_SPI – This configuration targets all the CC2530 flash devices and is using SPI to communicate with the host processor. The serial bootloader resides on the first flash page and occupies 2KB of code space.
- CC2533F64_LP – This configuration targets the CC2533F64 flash device and is using UART to communicate with the host processor. The serial boot loader resides on the last

two flash pages and occupies 2KB of code space. This configuration is not recommended.

- CC2533F64 – This configuration targets the CC2533F64 flash device and is using UART to communicate with the host processor. The serial boot loader resides on the first flash page and occupies 2KB of code space. This configuration is to be used for both the CC2533F64 and CC2533F96 parts.

The target board included with the RemoTI development kit is capable of supporting all project configurations listed above. However, the PC demonstration application included in the RemoTI installer is connecting to the target board via UART interface. As such, you should choose the CC2530 configuration for use with the rest of the instructions in this document.

Choose '*Project -> Clean*' and then '*Project -> Rebuild All*' to build the basic remote sample application.

The serial bootloader image must be programmed onto a target platform before downloading a new application image using the RemoTI serial bootloader demo application.

CC2531 USB dongle serial boot loading is also supported but the project file is located in *\Projects\SerialBoot\CC2531USB* directory instead.

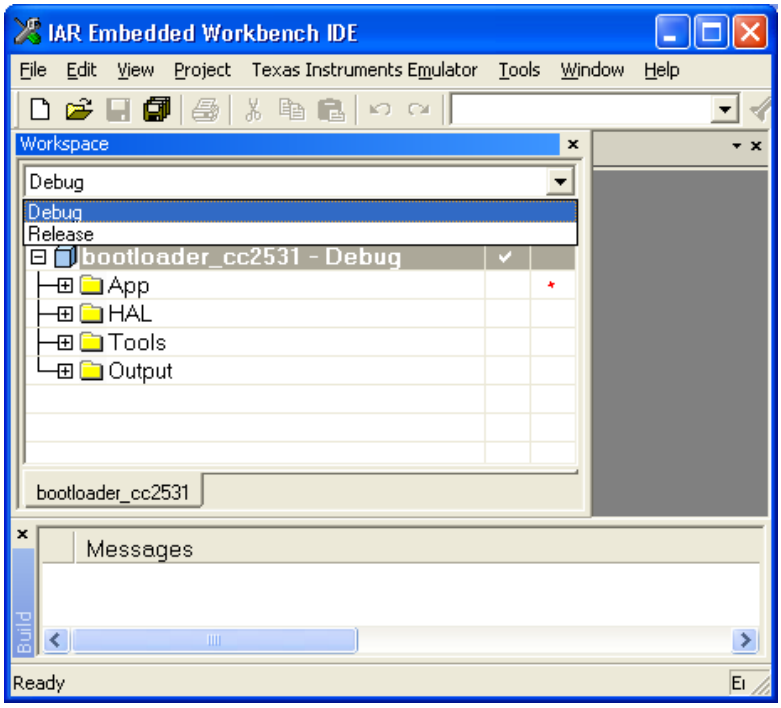


Figure 16. IAR Workbench for the CC2531 SerialBoot Sample Application

This project has two configurations, Debug and Release as illustrated in Figure 16. The Debug configuration is used to generate a debug image which can be downloaded through IAR (see section 4.6.2) while the Release configuration is used to generate a hex image which can be used with a SmartRF programmer (see section 4.6.3). Unlike the CC2530 project, the CC2531 project required two separate configurations to generate the debug image and the hex image because the two use different linker command options as the serial boot loader is compiled as banked code in order to use the last page in 256KB device and not waste the page. In order to build an additional hex file which combines both the serial boot loader and the downloadable image as described in section 4.6.4, the Release configuration must have been built beforehand.

4.6.2 Programming the Target Board with IAR Workbench

Connect the CC debugger to the target board and the PC. Instructions for how to connect the cables correctly are described in [6]. Verify the LED on the CC debugger is 'green' to indicate a successful connection. If not, press the 'Reset;' button.

You can now program the target board by choosing '*Project -> Debug*' from the IAR workbench. You also need to terminate the debug session:

1. Clicking '*Debug->Stop Debugging'* from IAR
2. Remove the RemoTI target board mini USB cable from PC
3. Remove ribbon cable from the target board
4. Re-connect mini USB cable to power cycle the target board.

4.6.3 Programming the Target Board with SmartRF Flash Programmer

Follow the instruction in section 4.1.4 to configure the workbench to generate a hex file that can be used to program the target board using the SmartRF Flash programmer.

4.6.4 Application image

Start the IAR workbench application, select '*File -> Open -> Workspace…*', browse to the *\Projects\RemoTI\RNP\CC2530EB* directory and select the *rnp_cc2530.eww* file. You should see the workspace as shown in Figure 6. For the CC2531, the HidDongle project may be used as well.

The workspace includes multiple project configurations as seen in the drop-down box in the workspace window of Figure 6. Choosing one of the project options will include the correct source and linker files, and set the project compile options accordingly. The available project options for use with a serial bootloader capable network processor are the following;

- CC2530F64_SB - This configuration targets the CC2530 64KB flash device and includes support for Serial Bootloader (SB).

- CC2530F128_SB - This configuration targets the CC2530 128KB flash device and includes support for Serial Bootloader (SB).
- CC2530F256_SB - This configuration targets the CC2530 256KB flash device and includes support for Serial Bootloader (SB).
- CC2531F256_SB - This configuration targets the CC2531 256KB flash device and includes support for Serial Bootloader (SB).
- CC2533F64_SB – This configuration targets the CC2533 64KB flash device and includes support for Serial Bootloader (SB).
- CC2533F64_SBLP – This configuration targets the CC2533 64KB flash device, programmed with a serial boot loader on the last flash page.
- CC2533F96_SB – This configuration targets the CC2533 96KB flash device and includes support for Serial Bootloader (SB).

See chapter 5.2 for memory map details of these project configurations.

Choose '*Project -> Clean*' and then '*Project -> Rebuild All*' to build the basic remote sample application.

The output of the build will be located in *Projects\RemoTI\RNP\CC2530EB\<Project configuration>\Exe* and is named *rnp_cc2530.bin*. This file will be loaded onto the target board using the RemoTI serial bootloader demo application.

If this build is performed after the successful build of the corresponding serial boot loader build (section 4.6.1.1), an additional output file can be found in the same location with ".hex" extension. This hex file includes a combined image of both serial boot loader and the downloadable image pre-configured as an active image.

4.6.5 Serial Bootloader Demo Application and Execution

Programming the RNP or HID Dongle with a new application image is performed using RemoTI serial boot loader demo tool. The tool has the dialog GUI shown in Figure 17.

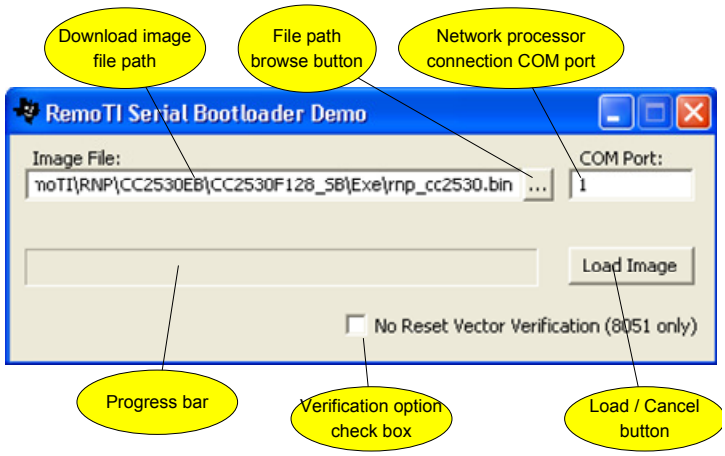


Figure 17. Serial Bootloader Demo Application User Interface

The demo tool connects to the target board through selected COM port and downloads a particular image to the selected target platform.

Follow the steps below to run the serial bootloader demo.

1. Run *bin\RemoTI_SBDemo.exe*.
2. CC2531 dongle only – disconnect the USB dongle and press and hold the toggle button furthest from the USB connector (SW1) while re-inserting it.
3. Type in PC COM port number to where the target board is connected.
4. Click the '…' button and select a downloadable image that you have built in chapter 4.6.4
5. Check the 'No Reset Vector Verification' option check box in case the serial boot loader on the connected target board is located on the last flash page. Otherwise, leave the check box unchecked. When checked, the tool does not verify the reset vector match with the downloaded image.
6. Click the '*Load Image*' button. The '*Load Image*' button will change its label to '*Cancel*' when downloading. The progress bar will display progress and a text line will appear right above progress bar describing current status. Once downloading a new application image completes, a message window will pop up to notify the completion.
7. Use your downloaded image. You can close the serial boot loader demo application and follow the steps in chapter 4.4 to verify its operation. The target board does not require power cycling before connecting the target emulator.
8. Try loading image again after playing with downloaded image. Note that non-volatile data including pairing table is retained when downloading a new image. You can verify this by simply close the target emulator application and return to step 1 to download the image again.

Formatted: Bullets and Numbering

4.6.6 Serial Bootloader System Overview

This chapter briefly describes how the sample code works, as a reference for serial bootloader application developers. The chapter does not apply to the configuration where the serial boot loader is located on the last page of the flash. The serial boot loader located on the last flash page is not recommended due to the risk of unrecoverable state caused by download discontinuation.

Serial bootloader sample code is located in lowest address page of the code flash and hence executes first when the target platform (CC253x) resets. The boot loader checks the wake-up condition and determines whether it needs to download application image or jump to an already downloaded image. NV data pages are located in the highest address and this area is not overwritten through serial boot loading in order to retain the data. See chapter 5.2 for more information about the memory map.

Downloaded image is built to fit 64K code space but it is also implementation specific depending on how much code space is available in the target hardware platform.

Once the image is downloaded and downloaded image is running, host processor (in the demo, PC with serial boot loader demo application) triggers download of new image by sending a command using the same network processor interface packet format which downloaded image understands as a command to restart serial boot loader and this time with the intention of downloading a new image. See [4] for the packet format. Notification of new image download trigger is done by simply updating a shared variable in a specific memory location and resetting to start serial boot loader in the sample software. Downloaded image in this demo has been specifically configured to accept the command and to notify such trigger.

The new image download trigger does not necessarily have to follow the same mechanism as in the sample code. It could be triggered by host processor asserting a certain general purpose I/O line while resetting CC253x with RESET_N assertion. In such a case, serial bootloader can be designed to read the IO port and determine whether to start downloading a new image or to jump to previously downloaded image. This is more robust in the sense that the host processor can recover network processor from a prior bad or buggy image, if there is a concern of downloading buggy image.

4.6.7 Serial Boot Loader Image Compatibility

The RemoTI-1.1 serial boot loader downloadable image and RemoTI-1.2 serial boot loader downloadable image uses different non-volatile memory system and hence they are not compatible with each other. For CC2530 parts, the RemoTI-1.2 sample application includes code to erase RemoTI-1.1 non-volatile memory pages, so that the application could at least run without erratic behavior after updating a RemoTI-1.1 image with a RemoTI-1.2 image through the serial boot loader. Note that the previous non-volatile memory storage is completely erased and hence the pairing information is removed altogether and users have to pair the devices again after updating a RemoTI-1.1 image with a RemoTI-1.2 image.

Also note that neither CC2533 downloadable image nor CC2531 downloadable image of RemoTI-1.2 includes such a code to erase RemoTI-1.1 non-volatile memory pages, for an obvious reason that there was no such downloadable image included in RemoTI-1.1.

4.7 Over The Air Download (OAD)

The OAD sample application runs on the RC included with the RemoTI kit and is used to demonstrate new stack and application image download OTA. This is a powerful feature used to upgrade the RC image in the field, e.g. upload a new image to the RC communicating with a STB. The RemoTI OAD demo tool communicates with a target board programmed with the RNP image using the UART interface.

Note that the RC must be programmed with the OAD bootloader and an application image containing the RemoTI stack to be capable of uploading a new application image OTA. The reason for this is that the OAD protocol is using TI specific vendor extension to the CERC profile for control and image payload transfer.

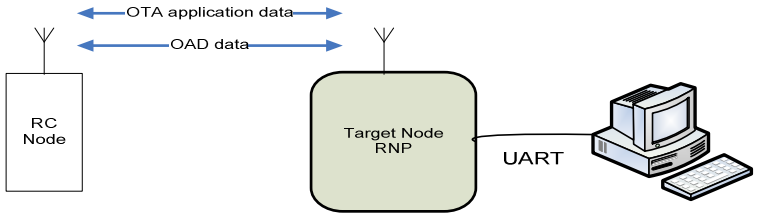


Figure 18. OAD Bootloader System Setup

4.7.1 Building and Programming the Sample Application

The OAD bootloader and the application image must be built separately. The IAR project configurations are set up such that the application image must be downloaded before downloading the bootloader. More specifically, the OAD bootloader is configured not to erase flash before programming the code as seen in Figure 19. This will prevent the application image from being deleted.

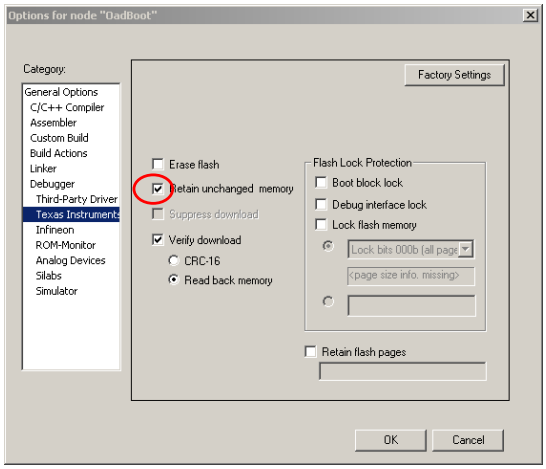


Figure 19. OAD bootloader Debugger Configuration

4.7.1.1 Application Image

Start the IAR workbench application, select '*File -> Open -> Workspace…*', browse to the *\Projects\BasicRemote\CC2530RC* directory and select the *rsa_cc2530.eww* file. You should see the workspace as shown in Figure 3.

The workspace includes multiple project configurations as seen in the drop-down box in the workspace window of Figure 3. Choosing the appropriate project options will include the correct source and linker files, and set the project compile options accordingly. The available project options for use with the OAD bootloader capable RC are the following;

- CC2533F96_OAD – This configuration targets the CC2533 96KB flash device and supports OAD.
- CC2530F128_OAD – This configuration targets the CC2530 128KB flash device and supports OAD.
- CC2530F256_OAD – This configuration targets the CC2530 256KB flash device and supports OAD.

See chapter 5.3 for memory map details of these project configurations.

Choose '*Project -> Clean*' and then '*Project -> Rebuild All*' to build the basic remote sample application.

Download the image onto the RC by following the instructions in chapter 4.1.3, but do not shut down the IAR workbench application and follow steps in the subsequent section 4.7.1.2.

4.7.1.2 Bootloader

Start the IAR workbench application, select '*File -> Open -> Workspace…*', browse to the *\Projects\OadBoot\CC2530RC_Src* directory and select the *OadBoot.eww* file.

The workspace includes multiple project configurations as seen in the drop-down box in the workspace window. Choosing one of the project options will include the correct source and linker files, and set the project compile options accordingly. The available project options are the following;

- CC2533F96 – This configuration targets the CC2533 96KB flash devices. The OAD bootloader resides on the first NV page and occupies 2KB of code space.
- CC2530F128 – This configuration targets the CC2530 128KB flash devices. The OAD bootloader resides on the first NV page and occupies 2KB of code space.
- CC2530F256 - This configuration targets the CC2530 256KB flash devices. The OAD bootloader resides on the first NV page and occupies 2KB of code space.

Choose '*Project -> Clean*' and then '*Project -> Rebuild All*' to build the OAD bootloader sample application.

Download the image onto the RC by following the instructions in chapter 4.1.3. Terminate the debug session and restart the RC:

1. Select *Debug -> Stop Debugging* from the menu bar.
2. Disconnect cables and power cycle remote.
    a. Disconnect USB cable of CC debugger from the PC.
    b. Disconnect the ribbon cable connected to the remote.
    c. Remove batteries of the remote and place the batteries back into the battery bay.

The RC is now programmed with the OAD bootloader and the application image, and is a fully functional RC. Follow the steps in chapter 4.4 to pair the RC with the target emulator.

4.7.2 Alternative Downloading through the SmartRF programmer

In order to use SmartRF programmer to download an OAD image, a hex image file which combines both the OAD boot loader and the application image is required. In order to build such a hex file, simply build the boot loader first and then build the application image. Upon building the application image after successfully building the boot loader, an additional hex file can be found in the same directory where the application image binary file resides. This hex file is the one that combines both the boot loader and the application image.
The hex file can be downloaded to the remote controller hardware in the same way as described in section 4.1.4.

4.7.3 OAD Demo Application and Execution

Programming the RC with a new application image is performed using RemoTI OAD demo tool. The tool has the dialog GUI show in Figure 20.

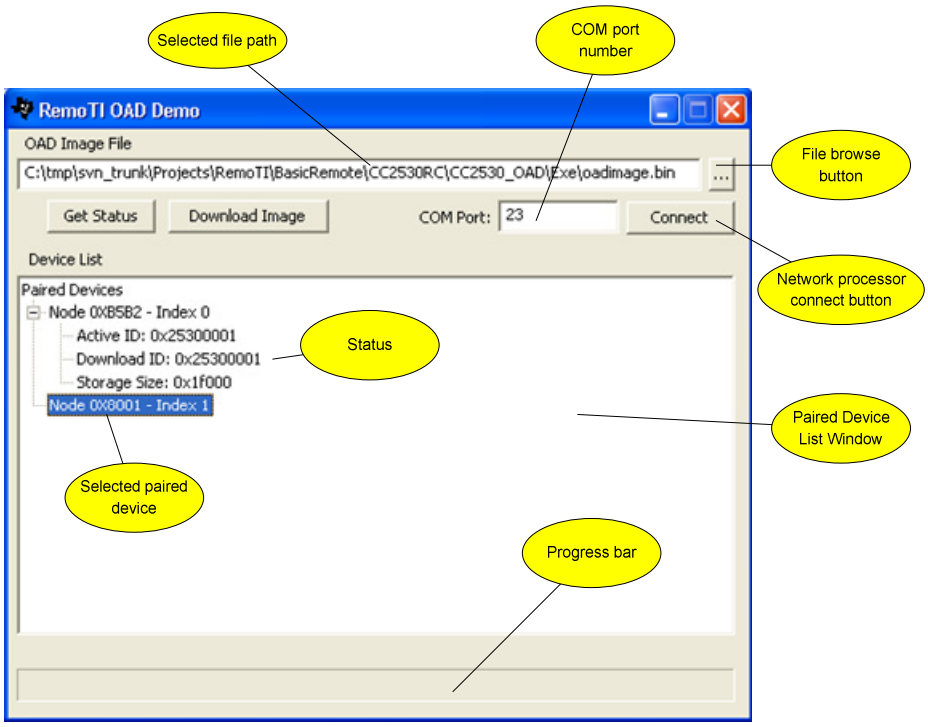


Figure 20. RemoTI OAD Demo Application UI

The demo tool connects to the target board through selected COM port and downloads a particular image to the selected target platform.

In order to verify that a new application image is downloaded onto the RC, it is helpful to create a new basic RC image with e.g. two keys swapped. The key mapping is defined in *rsa_basic.c* file as an array variable called *rsaKeyMap*. Change *RTI_CERC_NUM_1* with *RTI_CERC_NUM_2* as shown in Figure 21 below, and follow the instructions in chapter 4.7.1.1, except downloading the image onto the RC, to create the image to be downloaded OTA.

```
RTI_CERC_NUM_2,                 // 0b00 101 000   - 1
RTI_CERC_RESERVED_1,            // 0b00 101 001
RTI_CERC_VOLUME_UP,             // 0b00 101 010   - VOL+
RTI_CERC_NUM_3,                 // 0b00 101 011   - 3
RTI_CERC_PREVIOUS_CHANNEL,      // 0b00 101 100   - LAST
RTI_CERC_RESERVED_1,            // 0b00 101 101
RTI_CERC_NUM_1,                 // 0b00 101 110   - 2
RTI_CERC_CHANNEL_UP,            // 0b00 101 111   - CH+
```

Figure 21. BasicRemote Key swap

The output of the OAD application build will produce a binary file located in *Projects\RemoTI\BasicRemote\CC2530RC\<Project configuration>\Exe* and is named *rsa_cc2530.bin*. This file will be loaded onto the RC using the RemoTI OAD demo tool.

Follow the steps below to run the OAD demo.

1. Follow the instruction in [3] to connect the target emulator and pair with the RC.
2. Close the target emulator application.
3. Run *bin\RemoTI_OadDemo.exe*. Initially, paired device list window is empty.
4. Type in PC COM port number to where network processor (target board) is connected, and press the '*Connect'* button. The device list window will show devices (as *Node 0X*….) that are paired with the network processor. It should include the RC you paired with in step 1. At this stage, status information will not be shown.
5. Click on the node name in the device list you want to upload with a new image. Node name must be highlighted after selection.
6. Click the '…' button and navigate and select a downloadable image (*rsa_cc2530.bin*).
7. Click the '*Download Image'* button. The button will change its label to 'Cancel'. Nothing will happen at this stage other than button label change.
8. Press "Poll" key from the remote. Poll key is the key which has action "*RSA_ACT_POLL*" mapped in the key map file of the application file. Default poll key is the *'Live'* key. You should see progress bar starting to move to the right and right above the status bar there will be displayed some text information about the progress. By pressing '*Cancel*' button (which was "*Download Image*" button before), the download process can be aborted. When download is complete, a pop up message will indicate completion of download.
9. Close RemoTI OAD Demo application and re-open target emulator application to see number keys 1 and 2 are swapped. A number key 1 button press should now highlight number key 2 in the target emulator and vice versa

Note that the polling key on the RC is just one way of triggering OAD. Another typical way of triggering download will be for the RC to periodically wake up and poll the target node for any download initialization.

Note also that the OAD will not overwrite the current active image until the new image is completely downloaded and verified. This ensures the RC is always operational.

4.7.4 OAD System Overview

This chapter briefly describes how the over-the-air download demo works, as a reference for over-the-air download application developers.

The download of a new image over the air is performed by a previously downloaded and commissioned image, or default image itself in the sample application. Such currently active image is located right on top of a boot code. Downloaded image is temporarily stored into a free space (Downloaded image area). See chapter 5.3 for more information about the memory map.

Once the downloaded image is commissioned after successful download, the software resets into the boot code and boot code checks whether image in the downloaded area is a commissioned one. In such a case, boot code will copy the newly downloaded image over the active image area and then jump to the newly active image.

Location of where active image is, where to store a new downloaded, where to place non-volatile data pages, whether or not to retain non-volatile data after commissioning a new image, etc. are all application / implementation specific decisions. You do not have to follow the sample code mechanism as is. Also, the sample code performs download of the entire image except for boot code. That is also application specific decision. You might want to allow download of only upper layer applications and prohibit download of RemoTI stack software itself, or further allow download of only application data such as new key command map, for a certain product.

4.7.5 OAD Image Compatibility

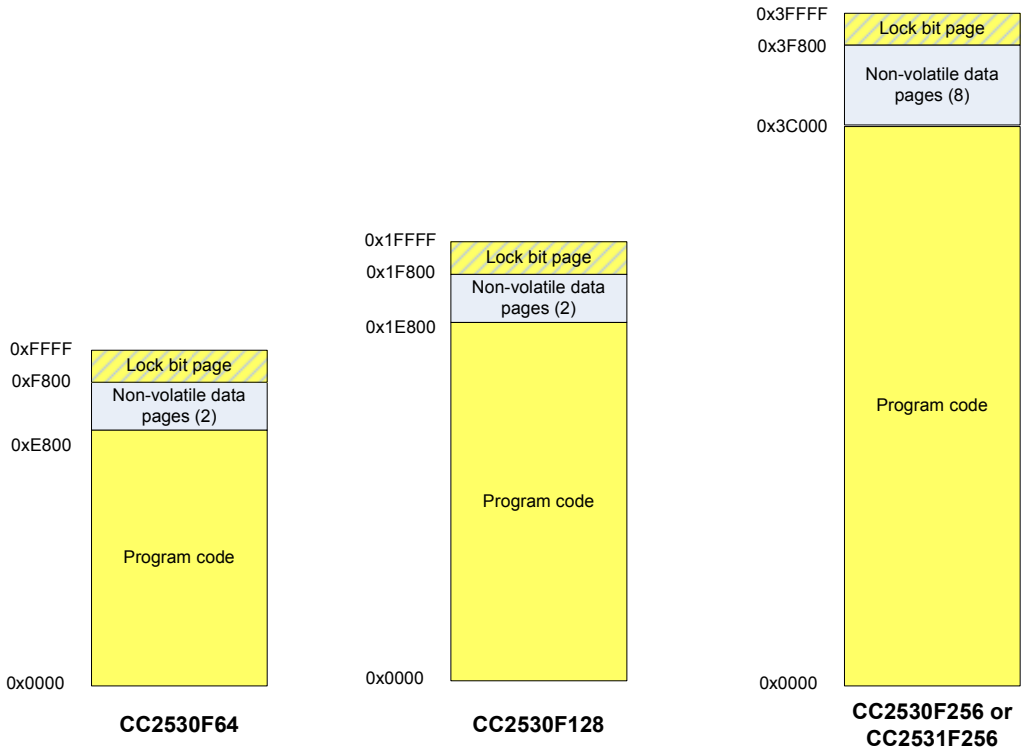
The RemoTI-1.1 OAD downloadable image and RemoTI-1.2 OAD downloadable image uses different non-volatile memory system and hence they are not compatible with each other. The RemoTI-1.2 sample application includes code to erase RemoTI-1.1 non-volatile memory pages, so that the application could at least run without erratic behavior after updating a RemoTI-1.1 image with a RemoTI-1.2 image through OAD. Note that the previous non-volatile memory storage is completely erased and hence the pairing information is removed altogether and users have to pair the devices again after updating a RemoTI-1.1 image with a RemoTI-1.2 image.

5 Memory Maps

This chapter describes the memory maps for the sample application configurations.

5.1 Basic Remote Controller (RC) and Network Processor (RNP)

Figure 22 below shows the memory maps for the CC2533F64, CC2533F96, CC2530F64, CC2530F128 and CC2530F256 *BasicRemote* and CC2533F64, CC2533F96, CC2530F64, CC2530F128, CC2530F256 and CC2531F256 RNP project configurations. Note that the each configuration is configured with a different number of NV data pages. The lock bit page contains the 16 byte flash lock bit structure as explained in [7]. The project configurations also enable programming of the 8 bytes commissioned IEEE address on the lock bit page. The rest of the lock bit page is used for program code.



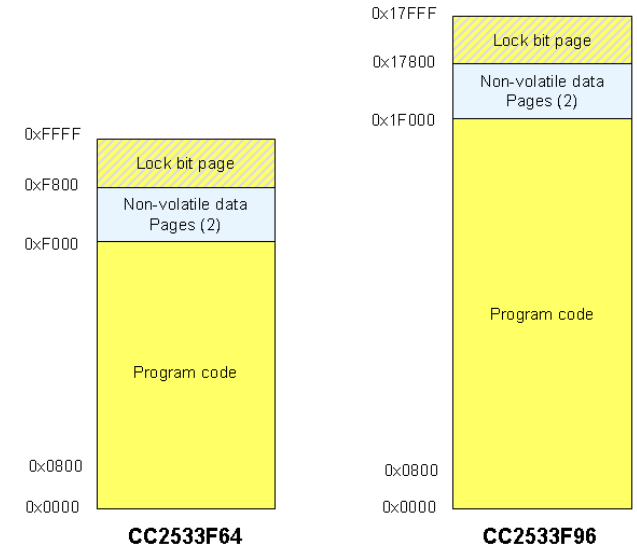


Figure 22. RC and RNP Memory Maps

5.2 RNP with Serial Bootloader (SB)

Figure 23 below shows the memory maps for the CC2530F64_SB, CC2530F128_SB and CC2530F256_SB RNP project configurations. Note that the each configuration is configured with a different number of NV data pages. The lock bit page is only accessible via the debug interface as explained in [7] and can as such not be used for program code downloaded by the serial bootloader (SB). The project configurations also enable programming of the 8 bytes commissioned IEEE address on the lock bit page.

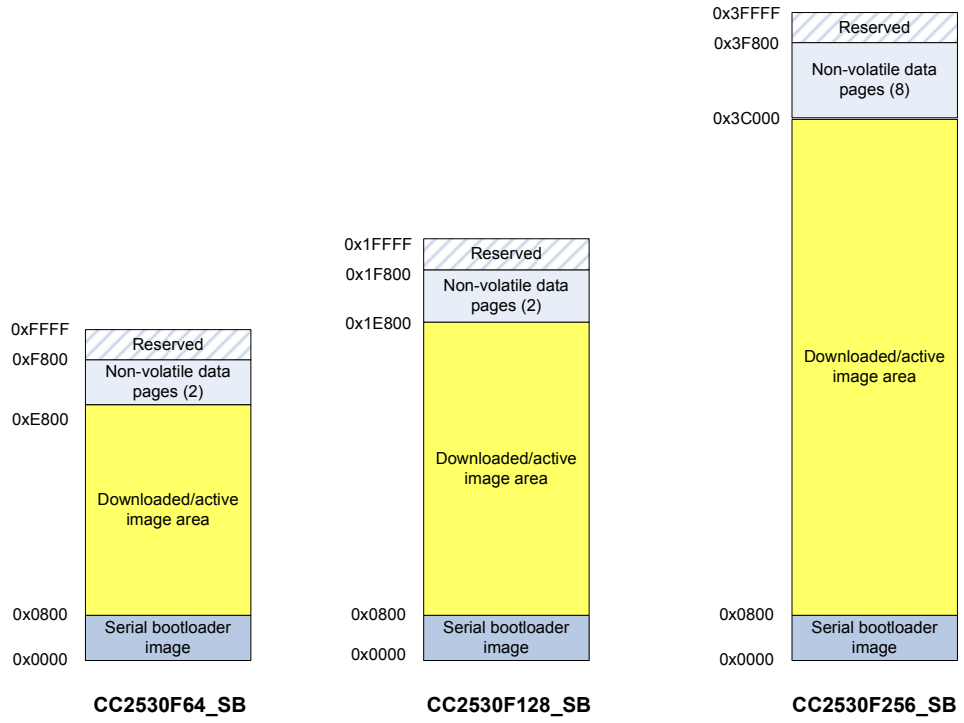


Figure 23. Serial Bootloader (SB) Memory Maps

The recommended configuration for the CC2533F64_SB is to locate the serial boot loader image is located in the first flash page. An alternative configuration CC2533F64_SBLP locates the bootloader image in the last flash page and the downloaded and active image from the start of memory except for the first three bytes of the reset vector which keeps pointing to the serial boot loader in the last page. The configuration is illustrated in Figure 24.

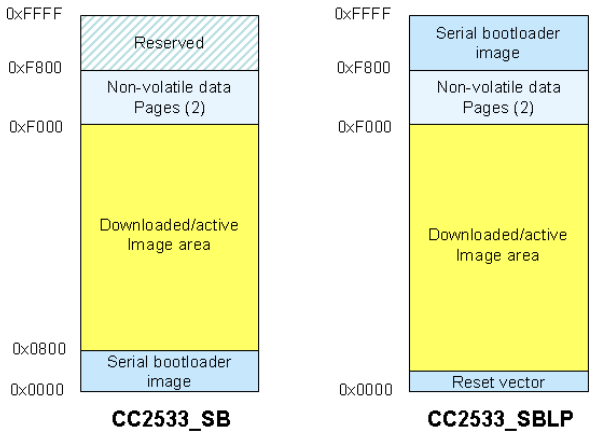


Figure 24. Serial Boot Loader for CC2533

5.3 RC with Over-The-Air Download (OAD)

Figure 25 below shows the memory maps for the CC2530F128_OAD and CC2530F256_OAD *BasicRemote* project configurations. Note that the each configuration is configured with a different number of NV data pages. The lock bit page is only accessible via the debug interface as explained in [7] and can as such not be used for program code downloaded by the OAD bootloader. The project configurations also enable programming of the 8 bytes commissioned IEEE address on the lock bit page.

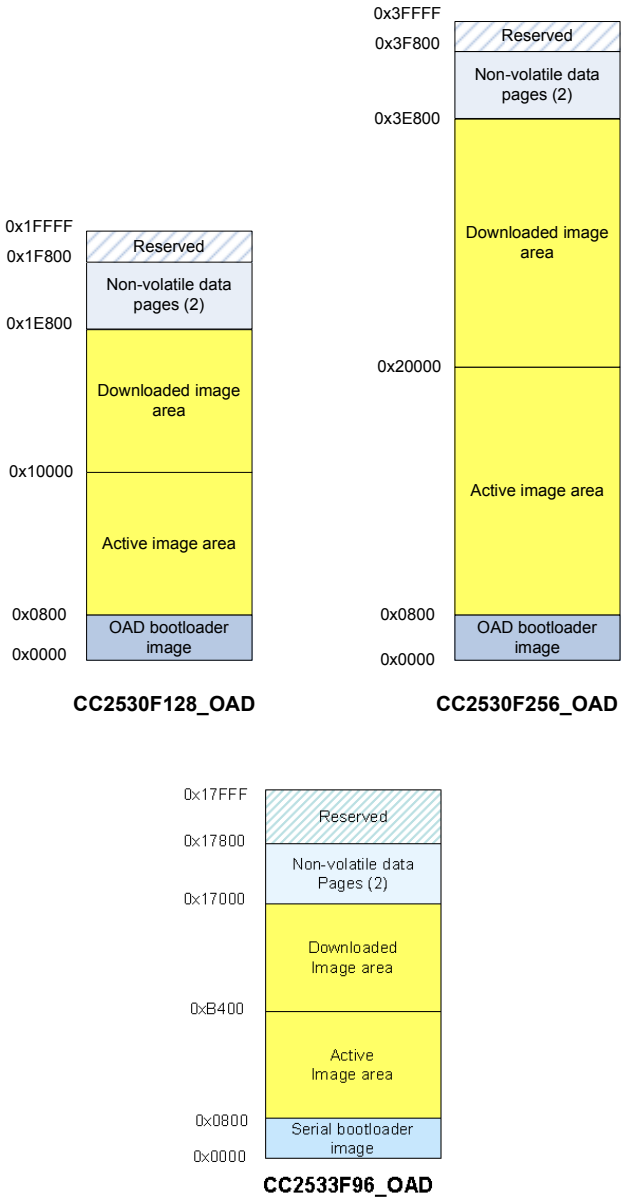


Figure 25. Over-The-Air Download (OAD) Memory Maps

6 Compile Options

This chapter provides a briefly description of the compile options used in sample application projects for information purpose.

Note that compile options are preset with each sample application project configurations and such pre-configured options are not meant to be modified.

Compile Flag	Description
POWER_SAVING	When defined, power saving modes are enabled. Without the compile flag, CC2530 PM2 and PM3 are not exercised. The compile flag affects HAL sleep module, OSAL power management module, RemoTI application framework (RTI) and network processor module. The CC2531 RNP does not use PM2 or PM3 and hence this compile flag is not defined in project configurations for the CC2531 device.
HAL_BOARD_CC2530RB	Platform board selection for RemoTI network processor project. This compile flag selects RemoTI Target Board as the hardware platform. This compile flag is defined by default when no other platform selection compile flag is defined.
HAL_BOARD_CC2530EB_REV13	Platform board selection for RemoTI network processor project. This compile flag selects SmartRF05 revision 1.3 board with CC2530EM as the hardware platform - DEPRECATED.
HAL_BOARD_CC2530EB_REV17	Platform board selection for RemoTI network processor project. This compile flag selects SmartRF05 revision 1.7 board with CC2530EM as the hardware platform.
CC2530F64	Non-volatile memory configuration selection for CC2530F64
CC2530F128	Non-volatile memory configuration selection for CC2530F128
CC2530F256OAD	Non-volatile memory configuration selection for CC2530F256 when over the air downloading is enabled. Note that without CC2530F64, CC2530F128 or CC2530F256OAD, default non-volatile memory configuration is set for CC2530F256 without over the air downloading feature. A notable difference between the default non-volatile memory configuration and the one set by CC2530F256OAD compile flag is the size of non-volatile memory pages.
CC2533F64	Non-volatile memory configuration selection for CC2533F64
CC2533F96	Non-volatile memory configuration selection for CC2533F96
CC2533F96OAD	OAD configuration for the CC2533F96

Compile Flag	**Description**
OAD_IMAGE_ID=*value*	Image identifier value for application image used by over-the-air download can be set with this compile flag value. For example, OAD_IMAGE_ID=0x25300002 will set image identifier value to be 0x25300002.
OAD_KEEP_NV_PAGES	This compile flag, when defined, suppresses adding non-volatile memory pages (OSAL NV module) into linker code. When FEATURE_OAD is defined, this compile flag also has to be defined if OSAL NV is used.
SB_TRIGGER_BY_GPIO	This compile flag directs serial boot loader sample code to poll an IO pin to determine whether or not to go into boot loader mode. The compile flag is used for SPI configuration.
FEATURE_TEST_MODE	RTI test mode API functions shall be enabled with this compile flag. The compile flag affects RTI and RTI surrogate.
FEATURE_CONTROLLER_ONLY	This compile flag, when defined, reduces RTI code size when RTI is compiled for remote controller functionality only.
FEATURE_OAD	This compile flag, when defined, enables over the air download feature in basic remote controller application code. Note that defining this compile flag alone does not enable over the air download for the basic remote controller sample application. Choose the proper configuration (with OAD tag) in the provided project to enable over the air download feature. Such configuration includes this compile flag definition in the project settings.
FEATURE_SERIAL_BOOT	This compile flag, when defined, enables serial boot loading feature in RTI surrogate for RemoTI network processor. Note that defining this compile flag alone does not enable serial boot loader for the whole RemoTI network processor (RNP) sample application. Choose the proper configuration (with SB tag) in the provided project to enable over the air download feature. Such configuration includes this compile flag definition in the project settings.
SB_LOC_LAST_PAGE	This compile flag directs serial boot loader project and RNP project to locate the serial boot loader into the last flash page of CC2530F64 device. Note that such location of the serial boot loader is not recommended.
GENERIC=__generic	This compile flag shall always be defined as GENERIC=__generic to be compatible with the RemoTI library files. The compile flag was devised to add IAR specific compiler keyword to certain function parameters.

7 General Information

7.1 Document History

Revision	Date	Description/Changes
1.0	2009-04-13	Initial release.
swru201a	2009-09-18	Update to RemoTI-1.1 release. Added pairing abort key usage. Added CC2531 USB dongle sample applications usage. Section 4.6.2 correction (RC → target board).
swru201b	2010-07-07	Update to RemoTI-1.2 release. Added new configurations of sample application projects included in RemoTI-1.2 release. Added serial boot loader demo update due to a new configuration (serial boot loader in the last flash page and the serial boot loader for CC2531 dongle) and additional output files (combined hex file). Added a section for alternative downloading of OAD image through the SmartRF programmer. Added sections about RemoTI-1.1 upgrade to RemoTI-1.2 through the serial boot loader or the over-the-air downloading

8 Address Information

Texas Instruments Norway AS
Gaustadalléen 21
N-0349 Oslo
NORWAY
Tel: +47 22 95 85 44
Fax: +47 22 95 85 46
Web site: http://www.ti.com/lpw

9 TI Worldwide Technical Support

Internet

TI Semiconductor Product Information Center Home Page: support.ti.com

TI Semiconductor Knowledge Base Home Page: support.ti.com/sc/knowledgebase

Product Information Centers

Americas

Phone: +1(972) 644-5580

Fax: +1(972) 927-6377

Internet/Email: support.ti.com/sc/pic/americas.htm

Europe, Middle East and Africa

Phone:

Belgium (English)	+32 (0) 27 45 54 32
Finland (English)	+358 (0) 9 25173948
France	+33 (0) 1 30 70 11 64
Germany	+49 (0) 8161 80 33 11
Israel (English)	180 949 0107
Italy	800 79 11 37
Netherlands (English)	+31 (0) 546 87 95 45
Russia	+7 (0) 95 363 4824
Spain	+34 902 35 40 28
Sweden (English)	+46 (0) 8587 555 22
United Kingdom	+44 (0) 1604 66 33 99

Fax:	+49 (0) 8161 80 2045	
Internet:	support.ti.com/sc/pic/euro.htm	

Japan

Fax	International	+81-3-3344-5317
	Domestic	0120-81-0036
Internet/Email	International	support.ti.com/sc/pic/japan.htm
	Domestic	www.tij.co.jp/pic

Asia

Phone	International	+886-2-23786800
	Domestic	Toll-Free Number
	Australia	1-800-999-084
	China	800-820-8682
	Hong Kong	800-96-5941
	India	+91-80-51381665 (Toll)
	Indonesia	001-803-8861-1006
	Korea	080-551-2804
	Malaysia	1-800-80-3973
	New Zealand	0800-446-934
	Philippines	1-800-765-7404
	Singapore	800-886-1028
	Taiwan	0800-006800
	Thailand	001-800-886-0010
Fax		+886-2-2378-6808
Email		tiasia@ti.com or ti-china@ti.com
Internet		support.ti.com/sc/pic/asia.htm

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI 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 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. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		Applications	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

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

