

SmartRF™ Studio 7

Hands-on User Guide and Tutorial

*Wireless Connectivity Development Tools from
Texas Instruments*

v1.2, August 2011



Goals of this Tutorial

- Understand how SmartRF Studio works
- Learn how to use SmartRF Studio for
 - Evaluating RF products from TI
 - Exporting register values
 - Testing and tuning RF performance
 - Functional testing of the radio
 - Advanced control of the radio



The main goal of this tutorial is to get familiar with SmartRF Studio 7: Learn how it works and what it can do.

The tutorial will walk you through some exercises showing how to use SmartRF Studio 7 in situations that system developers often run into.

You will learn how to use SmartRF Studio 7 to evaluate the various RF products from Texas Instruments on a general basis. This includes performing functional testing of the radio and running specific tests for measuring, and eventually tuning, the performance of the radio.

In addition, you will learn how the register values found in SmartRF Studio can be exported and used in either your own software or in other related tools.

The best way of learning is by doing, so we encourage you to go through the exercises as described and try to understand what's happening before skipping ahead to the next exercise.

How to use this Tutorial

- The first part has an introduction to SmartRF Studio, highlighting some of its most important features and benefits.

This part is useful for getting a quick overview of the tool.

- The second part of the tutorial contains practical examples showing how many of the tool's features works and provides step-by-step instructions for how to use them

This part can be used as reference material or as self-teach modules.

Throughout the tutorial, the notes section below each slide will contain detailed instructions and explanations.



You'll find a detailed description for each slide here.

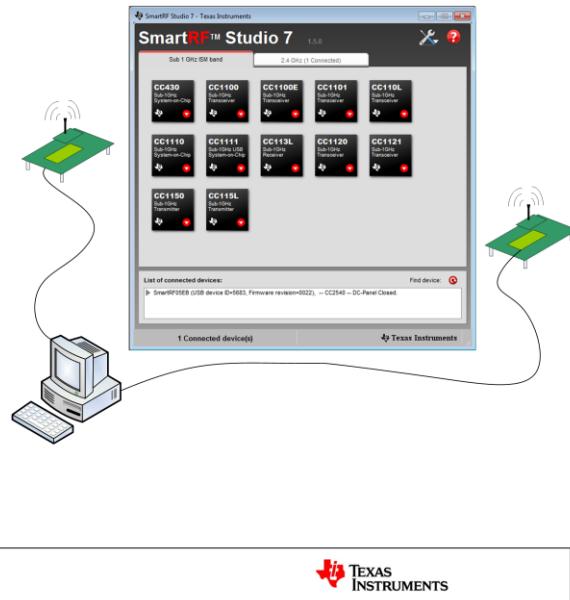
Overview

- **Introduction to SmartRF Studio 7**
- Installation and Hardware Setup
- Exercise 1: Measure Signal Strength
- Exercise 2: Measure Packet Error Rate
- Exercise 3: Register Export
- Exercise 4: XML Configuration
- Exercise 5: Command Strobe Panel



What is SmartRF Studio 7?

- SmartRF Studio 7 is a tool for evaluating TI Low Power RF ICs and for generating device register values
- The program can also be used to test RF performance and to test and tune customer specific hardware solutions



SmartRF Studio 7 is a PC application that can be used in combination with several development kits for Texas Instruments' CCxxxx and CC430 RF-ICs. It runs on Windows XP, Windows Vista and Windows 7 (32 and 64 bit) and uses USB (or parallel port for legacy boards) to communicate with the evaluation board (EB) which has an evaluation module (EM) with the RF chip mounted.

A radio on a custom board can also be tested with SmartRF Studio by wiring it to an EB board or a CC Debugger.

SmartRF Studio 7 helps designers of radio systems to easily evaluate the RF-IC at an early stage in the design process. The program provides an easy-to-operate PC interface to all of the chip's radio configuration registers, and it is very helpful for functional testing and for finding the appropriate radio settings.

SmartRF Studio 7 can also be used without any hardware, but then only to generate, edit and export radio register values.

SmartRF Studio 7 Features

- Quick and simple performance testing
 - Continuous TX for antenna testing and RF spectrum analysis
 - Continuous RX for radiation testing
 - Packet sending and receiving
- Generate and export register values from common RF parameters
 - Frequency, data rate, modulation, output power, etc.
 - Easy to export register values for use in SW.
- Read and write individual registers
 - For full control from the operator
- Advanced remote control and configuration of the radio from the PC
 - Configurable XML files
 - Perl script extension



Texas
INSTRUMENTS

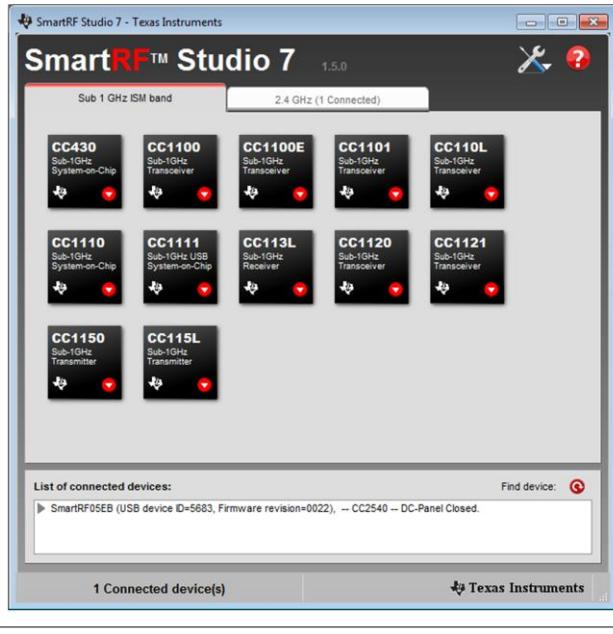
The main feature of SmartRF Studio 7 is that it has a control panel for direct access to the RF-IC's chip registers and radio-related features. It gives an overview of the many device specific features and provides full read and write access to the chip's radio registers.

SmartRF Studio 7 is a major update from SmartRF Studio 6.x.x (and earlier). The tool has been redesigned from ground up, focusing on flexibility and ease of use. The most apparent change is the new look and feel, aiming at making more information available and presenting it in an intuitive and easily understandable way.

The tool now has two main operating modes: In Easy Mode, the user can easily get started by using predefined register values and radio configuration. In Expert Mode, the user can fine tune all settings and is given more flexibility when configuring how the device should operate.

All of the device configuration and register settings are stored in XML files, making it possible to add custom configuration settings and to parse and reuse the files by external tools. The new script extension is particularly useful for customers who want to automate some of the test functions found in SmartRF Studio.

SmartRF Studio 7 – Startup Window



This is the startup window of SmartRF Studio 7.

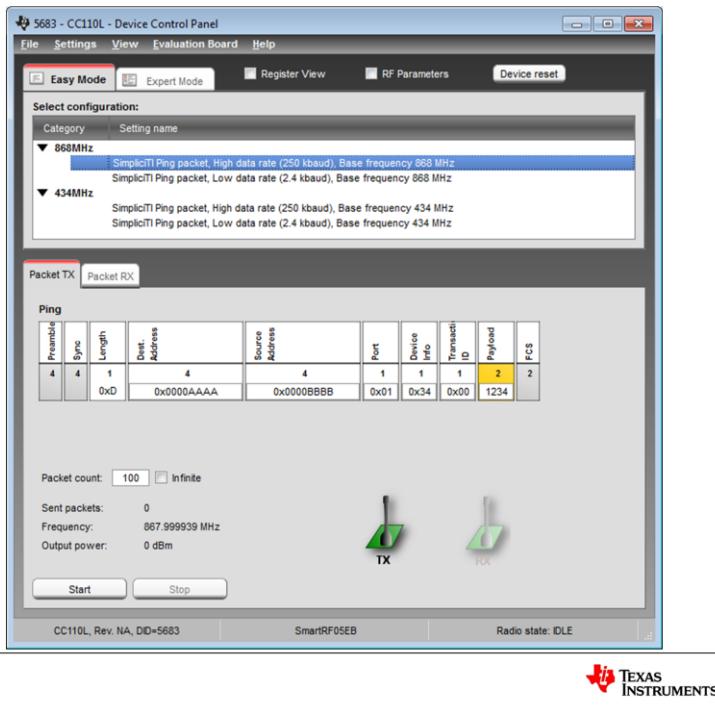
At the top, there are two tabs that lets you choose what product family you are working with: Either devices operating in the 2.4 GHz frequency band or devices in the sub-1 GHz band. The tab will show how many devices in each category are connected to the PC.

Each tab will have icons showing the supported devices. If there are any devices connected (either via a SmartRF Evaluation Board or a CC Debugger), the icon will be highlighted. If you double click one of the device icons, you will open the Device Control Panel for the device. The Device Control Panel is either in online or offline mode. In online mode, you have direct access to the connected device, giving full control of the device. In offline mode, you can set the various RF parameters and export register values. More options will be available by right-clicking on the device icon.

If you connect a SmartRF Evaluation Board with an Evaluation Module (or a CC Debugger connected to a supported device), the board will appear in the list of connected devices. The corresponding radio device icon will also be highlighted. The list of connected devices will, including to show which radio is on the board, show details about the evaluation board – in particular the board's firmware revision. If you double click on the board in the list, and the firmware is not the most recent version, SmartRF Studio 7 will ask the user whether the firmware should be updated or not.

Note that the current version does not yet support older devices like CC1020, CC2400 and CC2420. For these devices, use SmartRF Studio 6 which can still be downloaded from the web.

SmartRF Studio 7 – Device Control Panel

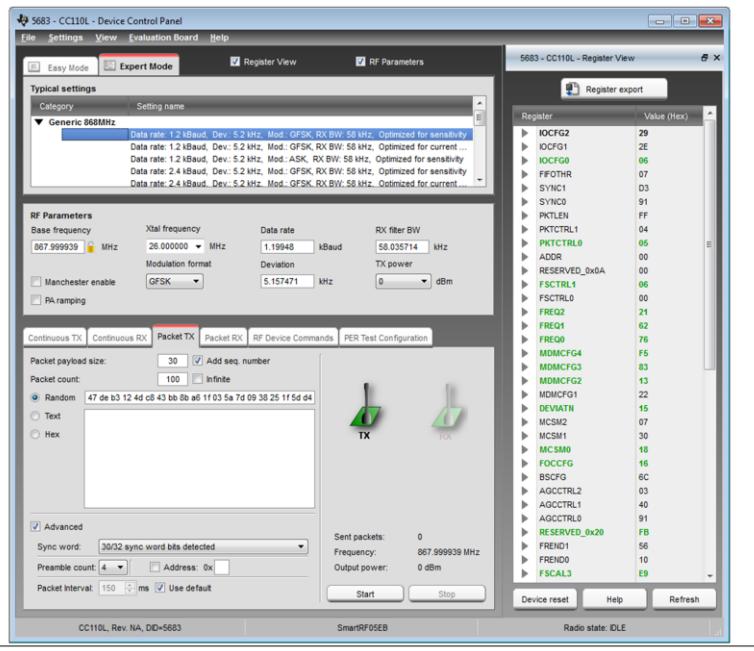


SmartRF Studio 7 has two main operating modes: Easy Mode and Expert Mode.

In Easy Mode (default), the user can select from a predefined set of radio configurations and packet types. The packet type matches one typical packet from a supported protocol, e.g. ZigBee or SimpliciTI. This selection determines what will be sent from the radio when using the packet TX and RX test functions in SmartRF Studio. The radio register values can be exported to an external file by opening the Register View and clicking the **Register export** button.

SmartRF Studio 7 has stored all configuration data and parameters in XML files. As a consequence, it is easy for users to add their own protocol/packet definitions, packet types and associated RF parameters.

SmartRF Studio 7 – Device Control Panel



In Expert Mode, the user will get access to more advanced features and test functions for the device.

The list of typical settings includes recommended register settings for some typically used parameter values. By selecting one of the typical settings the recommended register values for this combination of parameters will be programmed to the device.

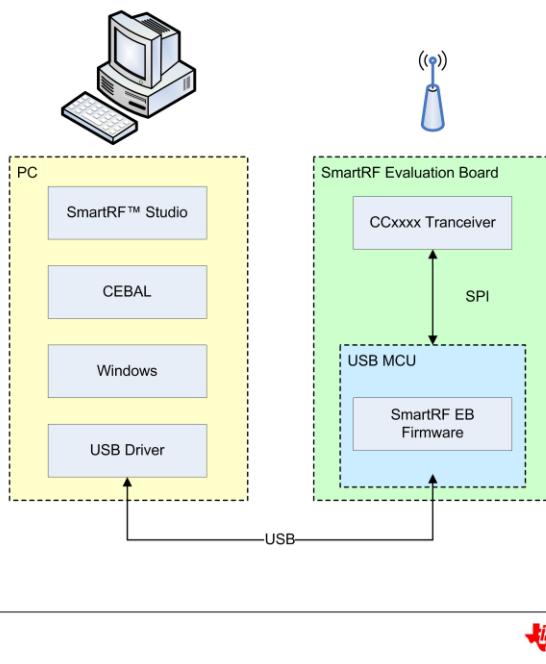
There are a number of test functions available:

- Continuous TX for transmitting an un-modulated or modulated carrier. This is useful for testing the antenna efficiency, output power, spectrum etc.
- Continuous RX for testing radiation in RX and for simple scanning of energy level on the given frequency.
- Receive and transmit packets
- Send individual strobe commands

In addition, the register view window will give an accurate representation of the register values currently on the chip. The register view can be opened by checking the Register View check box, by selecting “View → Register View” in the menu or simply by pressing F7. Color coding is used to identify the origin of the register value:

- | | |
|--------|---|
| Black: | Reset value |
| Green: | Typical/recommended setting or changed by the user in the RF Parameters panel |
| Blue: | Changed by the user in the Register View |
| Red: | Changed dynamically by the radio |

How does it work?



TEXAS
INSTRUMENTS

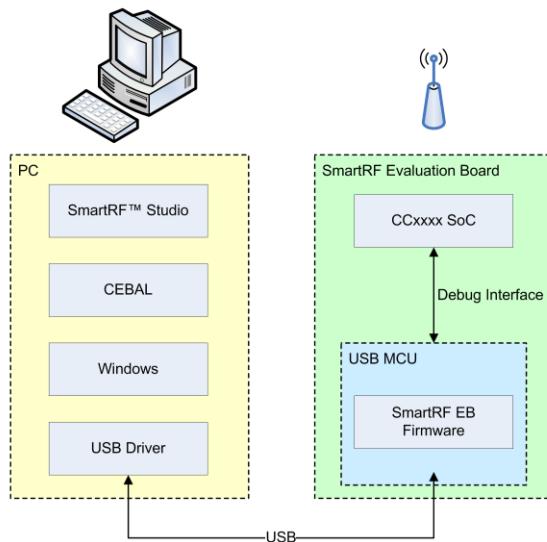
SmartRF Studio 7 communicates with the Evaluation Board over the USB interface via a library called CEBAL (Chipcon Evaluation Board Access Layer). This is a SW library developed to interface the USB driver and firmware running on the evaluation board, containing all the functions required to read/write data over the SPI interface between the USB MCU and the Transceiver or the debug interface in case of a System-on-Chip.

For proper operation of the applications using CEBAL, the board will need to have compatible firmware running on the USB MCU. If the firmware is out of date, SmartRF Studio 7 will propose that the user updates the firmware. The firmware update can be done directly in SmartRF Studio 7.

SmartRF Studio 7 will send requests to the USB MCU. The USB MCU will handle the request and apply the appropriate read/write commands on the SPI or debug interface.

The USB driver is a licensed driver delivered by Thesycon:
<http://www.thesycon.de/eng>

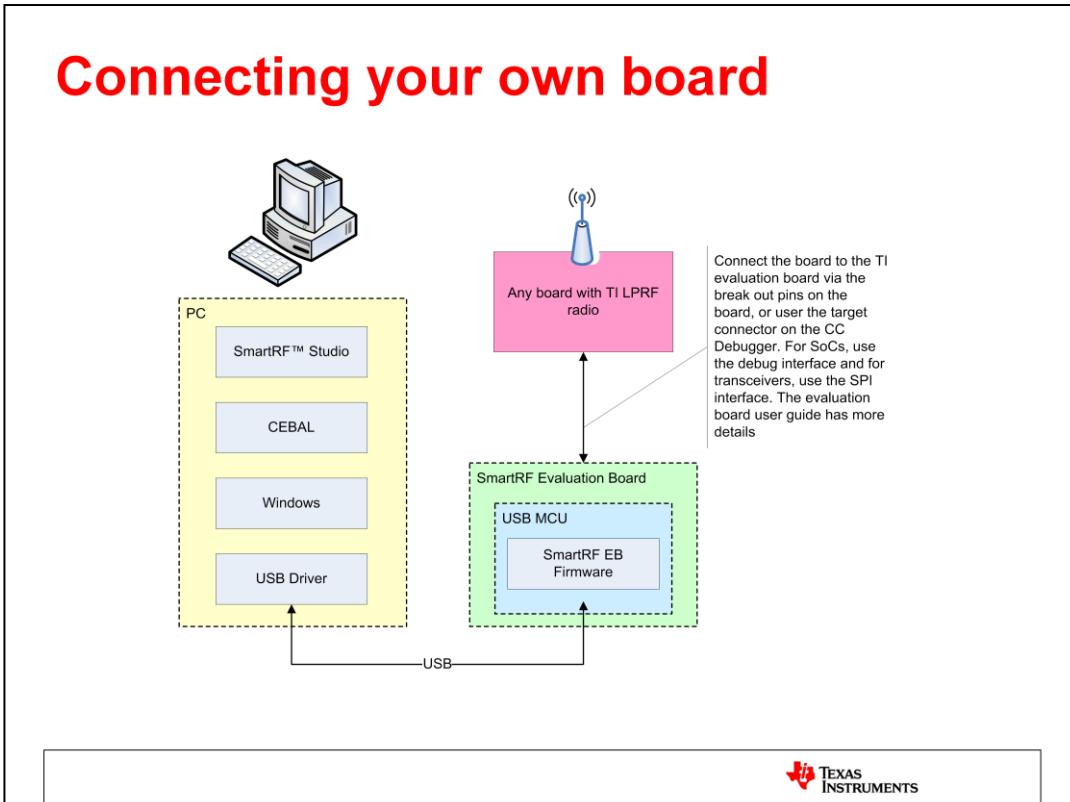
How does it work?



 TEXAS INSTRUMENTS

For the CCxxxx System-on-Chips (SoC) we use the same SW library on the PC side and the same FW on the USB MCU. The difference is that the USB MCU will use the debug interface to read/write data of the SoC.

Connecting your own board



It is of course possible to connect your own hardware to the SmartRF Evaluation Board or CC Debugger (preferred) to test your own radio design with SmartRF Studio.

Connect the board to the TI evaluation board via the break out pins on the board, or user the target connector on the CC Debugger. For SoCs, use the debug interface and for transceiver, use the SPI interface. Please make sure the boards are properly connected and that the voltage levels are correct – especially if you are not using level converters and the voltage level on your board is different from the voltage level on the EB board (usually 3.3 Volt).

There is more information in SmartRF05EB User's Guide (www.ti.com/lit/swru210) and CC Debugger User's Guide (www.ti.com/lit/swru197).

Overview

- Introduction to SmartRF Studio 7
- **Installation and Hardware Setup**
- Exercise 1: Measure Signal Strength
- Exercise 2: Measure Packet Error Rate
- Exercise 3: Register Export
- Exercise 4: XML Configuration
- Exercise 5: Command Strobe Panel



How to get it

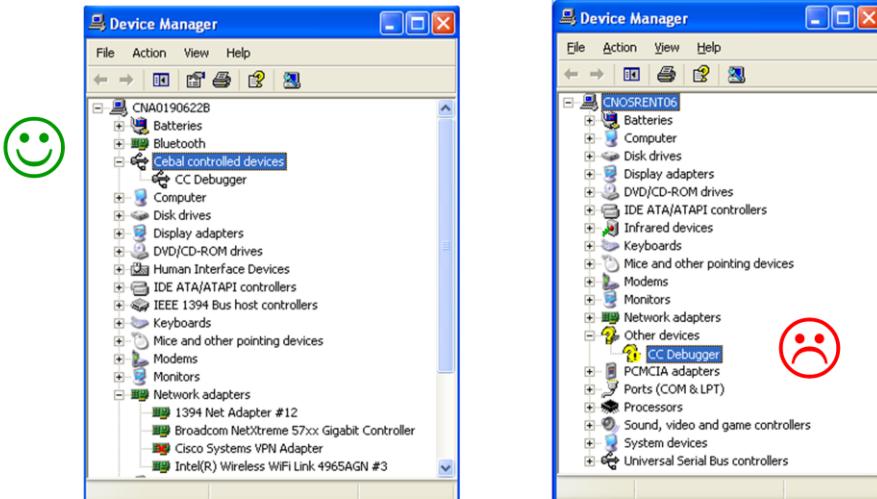
URL: www.ti.com/smartrfstudio

The screenshot shows the Texas Instruments website for the SmartRF Studio product. At the top, there's a navigation bar with links for Products, Applications, Design Support, Sample & Buy, and a search bar. Below the navigation, the product name "SmartRF Studio" is displayed with a status of "ACTIVE". There are tabs for Description/Features, Technical Documents, and Support & Community. A "Order Now" section shows a table with one row for "SMARTRFTM-STUDIO", which is ACTIVE and Free. A yellow box highlights the "Key Document" section, which contains a PDF link for "SmartRF Studio 7 v1.5.0 (Rev. 1)" (ZIP 34630 KB) from June 29, 2011, with 13,611 views. A link to "View all technical documents (8)" is also present. To the right, there's a small thumbnail image of the software interface. At the bottom, the Texas Instruments logo is visible.

When installing SmartRF Studio it will include the Windows drivers needed to communicate with the SmartRF04EB, SmartRF05EB and CC Debugger. It is therefore in general a good idea to install SmartRF Studio **before** you connect the evaluation board to the computer for the first time.

When the evaluation board is connected to the PC, the drivers will be installed automatically.

Verifying the driver installation



TEXAS
INSTRUMENTS

It is possible to verify that the driver has been installed correctly. By looking in Windows' Device Manager you can see the state of the connected hardware.

To the left the CC Debugger has been correctly associated as a CEBAL controlled device. CEBAL is the PC library used to communicate with the evaluation boards over USB.

To the right the CC Debugger is listed under *Other devices* and has an exclamation mark. This is typical if Windows is not able to associate any driver for it. In this situation the PC will not be able to communicate with the evaluation board and the driver should be manually (re)installed.

For troubleshooting and more detailed information about driver installation, please consult "DN304 -- CCxxxx Development Tools USB Driver Installation Guide" <http://www.ti.com/lit/swra366>

Required Hardware for the Tutorial

- The instructions in the tutorial is based on having access to the following development kits from Texas Instruments:
 - 1 x CC1110DK-MINI-868
 - 1 x CC Debugger
 - 1 x CC1111EMK-868-915
- Suggested alternate hardware
 - CC1120DK
 - CC1101DK-868-915 or CC1101DK-433
 - CC11XLDK-868-915
 - CC1110-CC1111DK
 - CC2510-CC2511DK

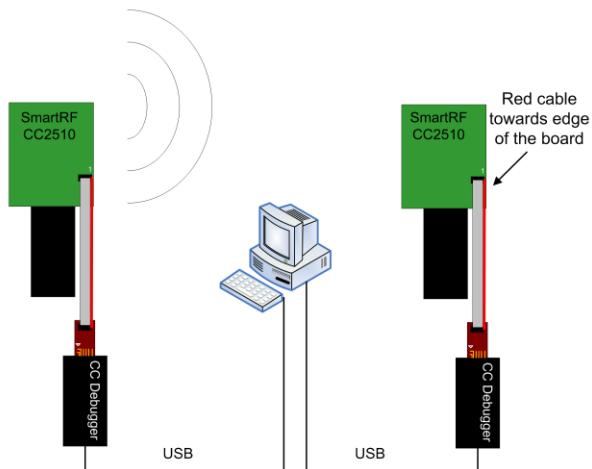


The tutorial section of this hands-on user guide shows how to use SmartRF Studio together with the hardware included in the CC1110 Mini Development Kit (CC1110DK-MINI-868, together with an additional CC Debugger and a CC1111 USB Dongle).

It is of course possible to use different hardware. In essence, you should have two boards that can be connected to the PC for setting up a simple RF link between two devices. You would also need a compatible packet capture device for Exercise 3. Recommended capture devices are CC2511 USB dongle (CC2511EMK) for 2.4 GHz proprietary radios, CC2531 USB dongle (CC2531EMK) for IEEE 802.15.4 compliant radios or a CC1111 USB dongle (CC1111EMK-868-915) for sub-1 GHz proprietary radios.

Hardware Setup

- Two CC1110 mini boards connected via CC Debuggers as shown in the figure to the right.
- The boards can be powered through the CC Debugger by setting a jumper on the board (see user guide), so there is no need for batteries.
- 1 CC1111 USB Dongle preprogrammed with packet sniffer firmware connected directly to the PC (only required in exercise 3)



The figure above shows how to connect the default development kits to the PC in order to go through the tutorial exercises. Other hardware can of course also be used, but then some of the details in the description of the exercises might not apply.

As long as you have two development boards with a radio that can be controlled by SmartRF Studio, you will be able to do most of the exercises.

Note that if you use hardware from e.g. the CC2530DK or CC2540DK, you have less options for setting RF parameters, as the physical layer (radio parameters) are more or less fixed for these devices. CC2530 is an IEEE 802.15.4 radio and CC2540 is a *Bluetooth* low energy device, where the physical layer is determined by the standard. In particular, there is no need for specific register settings when using these devices as capture devices in SmartRF Packet Sniffer, so most of exercise 3 is not relevant.

Overview

- Introduction to SmartRF Studio 7
- Installation and Hardware Setup
- **Exercise 1: Measure Signal Strength**
- Exercise 2: Measure Packet Error Rate
- Exercise 3: Register Export
- Exercise 4: XML Configuration
- Exercise 5: Command Strobe Panel



Exercise 1: Measure Signal Strength

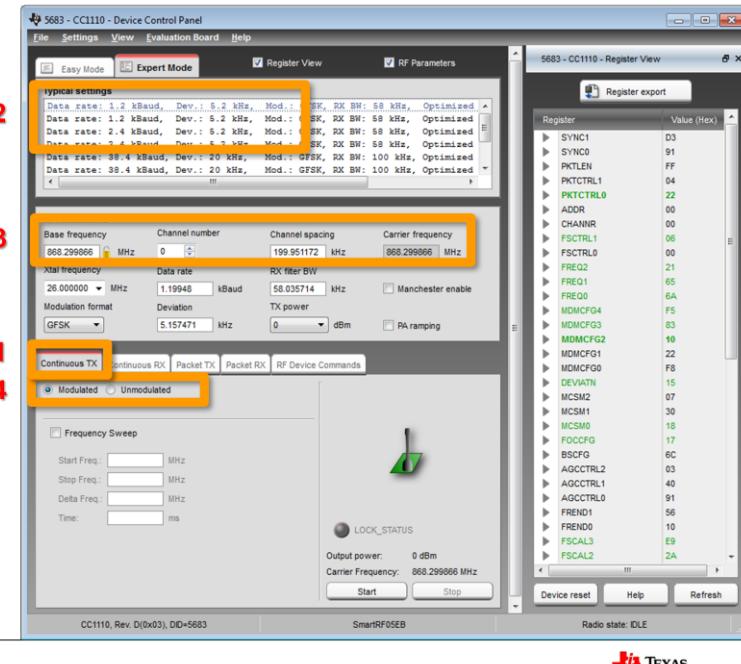
Purpose of the exercise:

- Use SmartRF Studio to measure the signal strength (RSSI) of a transmitted carrier
- Learn how to set the radio in Continuous TX
- Learn how to set the radio in Continuous RX



With SmartRF Studio it is easy to measure the signal strength (RSSI) on the receiver. This can be useful to find optimal RF settings or for testing basic functionality of your own radio system.

Exercise 1: Continuous TX



First, we will set up one of the devices to transmit a carrier signal.

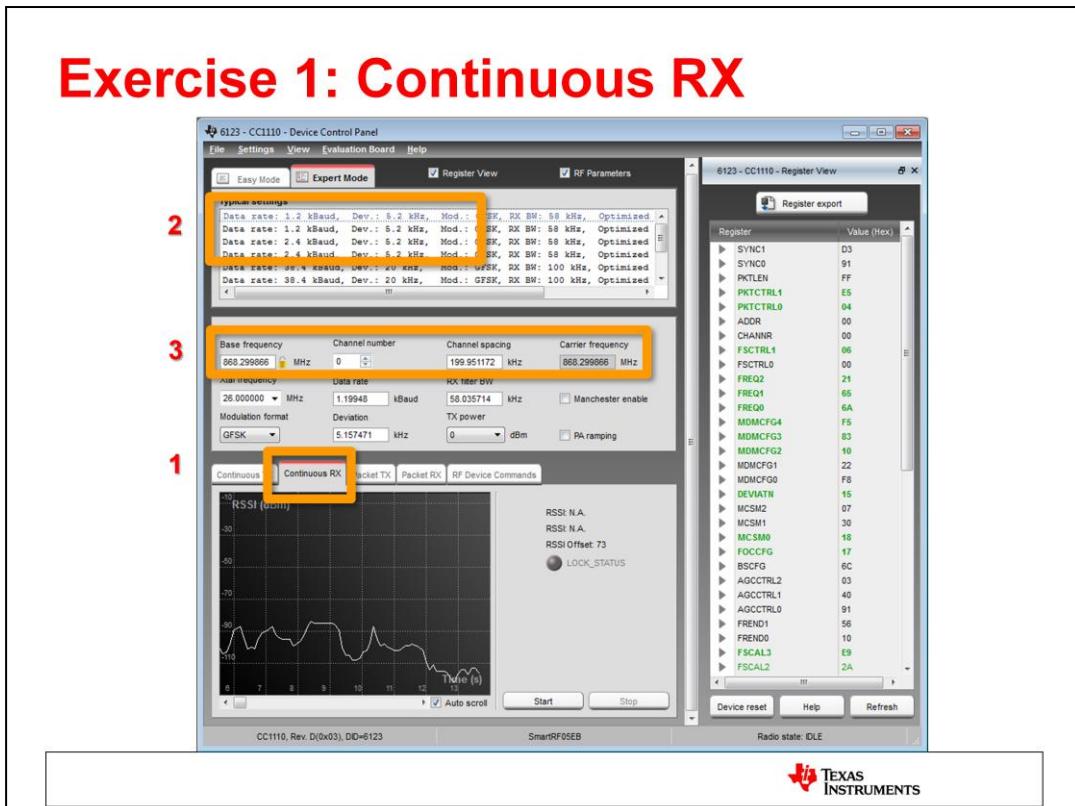
1. Start by navigating to the Expert Mode panel and select the Continuous TX panel.
2. Next, select the appropriate typical settings.
3. Select the frequency. Note that the actual carrier frequency is the base frequency + (channel number * channel spacing).
You can also select a different output power.
4. Select either a modulated or unmodulated carrier.

If you choose a modulated carrier, a pseudorandom signal will be transmitted using the modulation parameters given by the typical settings selected above.

If you choose an unmodulated carrier, a pure RF tone will be transmitted at the chosen frequency. The other RF settings will not have any impact on the signal.

Finally, click the **Start** button to start the transmission of the carrier by the radio.

Exercise 1: Continuous RX



1. On the other node (the device control panel for the other device), select Expert Mode and Continuous RX.
2. Select the same typical settings as for the transmitter
3. Select the same frequency as the transmitter

You can now start Continuous RX by clicking the **Start** button.

Observe the RSSI value, which is the strength of the signal received by the radio. Both the RSSI value in dBm and the raw value as returned by the chip is displayed. The RSSI in dBm will also be plotted vs time in the graph window.

Exercise 1: Measure Signal Strength

Try to modify some of the RF parameters on the receiver side and observe the effect on the signal strength:

1. Set a carrier frequency that is slightly different from the transmitted carrier frequency (e.g. +/- 500 kHz). What happens?
2. With the frequency offset above, try to increase the RX filter bandwidth (e.g. 800 kHz).
3. Increase or reduce the output power from the transmitter



1. Change the frequency on the receiver about 500 kHz off the frequency of the sender and see that the signal strength decrease.
2. Change the RX filter bandwidth on the receiver to about 800 kHz and see that the signal strength increase.
3. Reduce the output power from the transmitter and see that the received signal is reduced accordingly. E.g. a 3 dBm reduction of the output power from the transmitter should be reflected in a 3 dBm lower RSSI value on the receiver. The result could be influenced by the distance between the two devices and other RF sources/noise. If the devices are too close to each other, the differences at the transceiver and the receiver side could be less accurate.

Overview

- Introduction to SmartRF Studio 7
- Installation and Hardware Setup
- Exercise 1: Measure Signal Strength
- **Exercise 2: Measure Packet Error Rate**
- Exercise 3: Register Export
- Exercise 4: XML Configuration
- Exercise 5: Command Strobe Panel



Exercise 2: Packet Error Rate Test

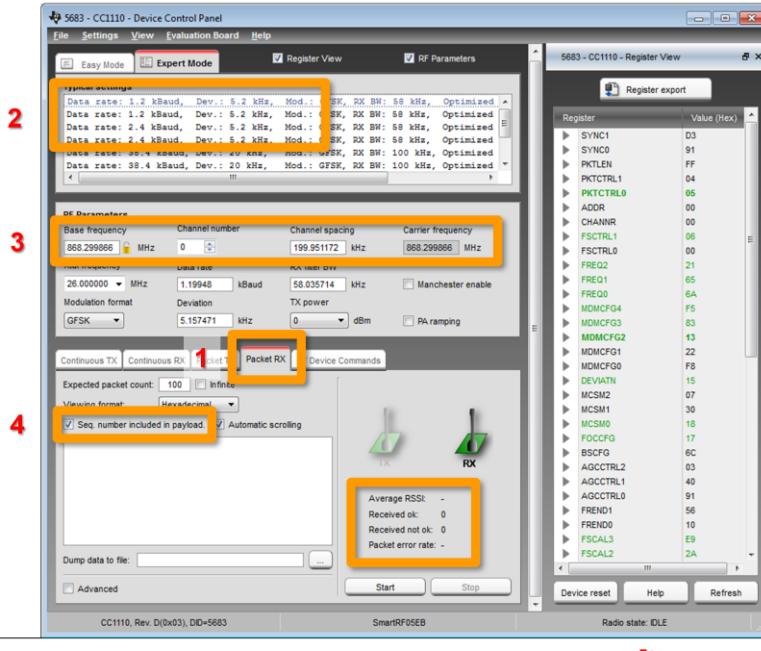
Purpose of the exercise:

- Check the radio link quality
 - The Packet Error Rate is an easy way to do link testing.
 - The RF settings can be customized to meet the system requirements.



The Packet TX and Packet RX tests can be used to check the link quality between two devices. The user can select preferred settings given by SmartRF Studio 7 or use customized register values to do the packet error rate (PER) testing.

Exercise 2: Packet RX



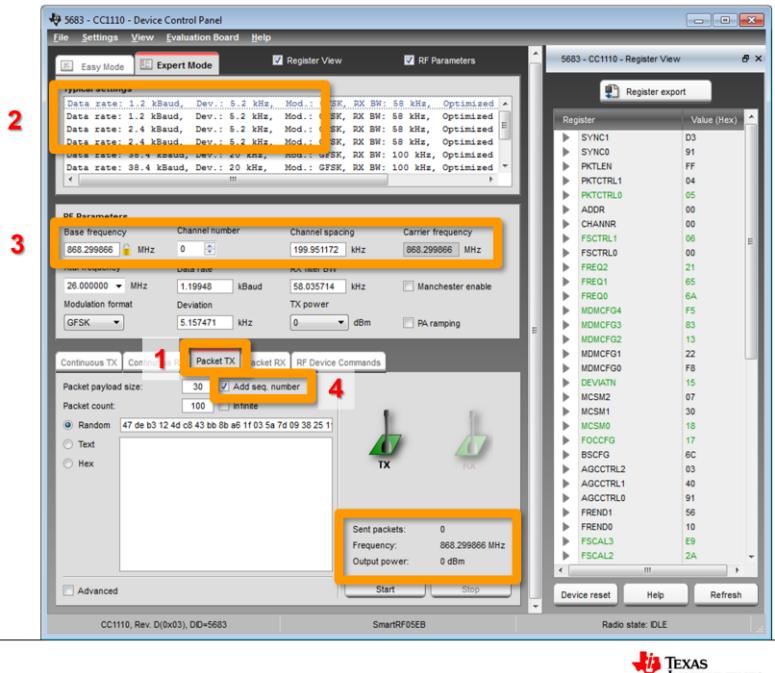
1. Select the Packet RX panel.
2. Select any of the typical settings from the list.
3. Select an appropriate frequency.
4. Select the option "Seq[ue]nce number included in payload". The receiver will use this number to check if packets are lost or not.

Click the **Start** button to start receiving packets.

When the other device has started sending packets (see next slide), observe the number of packets received OK and not OK at the right bottom of the Packet RX panel. The NOK counter will be incremented if the receiver detects that packets are lost (requires that the transmitter includes a sequence number in the payload) or if a received packet has a CRC error. The PER value is calculated using the formula:

$$\text{PER} = \# \text{ packets NOK} / (\# \text{ packets OK} + \# \text{ packets NOK})$$

Exercise 2: Packet TX



1. On the second device, navigate to the Packet TX panel.
2. Select the same typical settings as for the transmitter
3. Select the same frequency as the transmitter
4. The packet length and payload can be configured in the Packet TX panel. Make sure that the "Add seq[ue]nce" number" option is turned on. This will make it possible for the receiver to calculate the packet error rate more accurately.

Click the **Start** button at the right bottom of the panel.

The control panel for the receiver will now show the number of received packets and calculate the PER value based on the number of OK and NOK packets. The transmitter will display the number of packets currently sent.

Exercise 2: Packet Error Rate

Try to modify some of the RF parameters on either the receiver or transmitter side of the link.

1. See how output power/modulation/deviation/RX filter bandwidth affects the results
2. Tweak parameters the same way as was done in Exercise 1 and observe the effect.



Overview

- Introduction to SmartRF Studio 7
- Installation and Hardware Setup
- Exercise 1: Measure Signal Strength
- Exercise 2: Measure Packet Error Rate
- **Exercise 3: Register Export**
- Exercise 4: XML Configuration
- Exercise 5: Command Strobe Panel



Exercise 3: Register Export

Purpose of the exercise:

- Learn how to export register values
 - For direct use in SW development
 - To easily configure the SmartRF Packet Sniffer



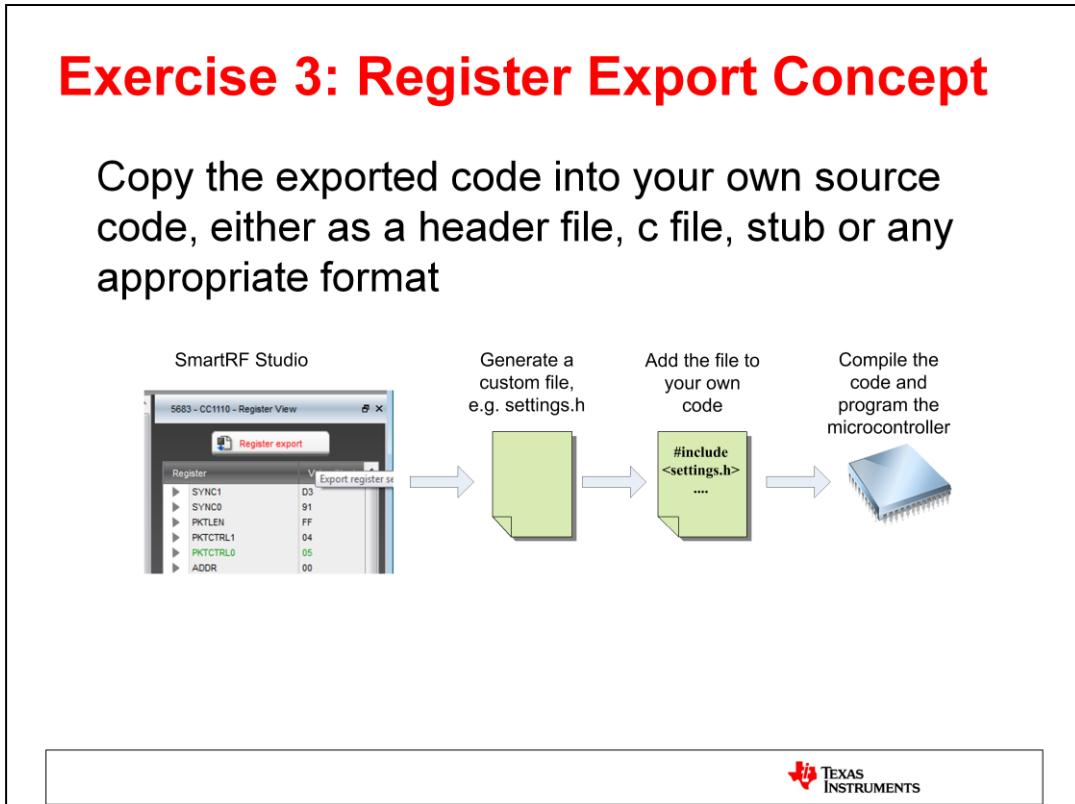
Register export, or code export, is a useful feature that simplifies the process of taking the register values found in SmartRF Studio and applying them in your own software or using it for other radio configuration purposes.

The register export applet lets you define whatever format of the exported registers that you like, so it can be made compatible with any thinkable software environment. A number of predefined formats (we call them templates) are already defined in the export applet, and you can of course create your own.

The exported register values can also be imported by the SmartRF Packet Sniffer in order to configure the packet capture device. This functionality will be demonstrated later in this exercise.

Exercise 3: Register Export Concept

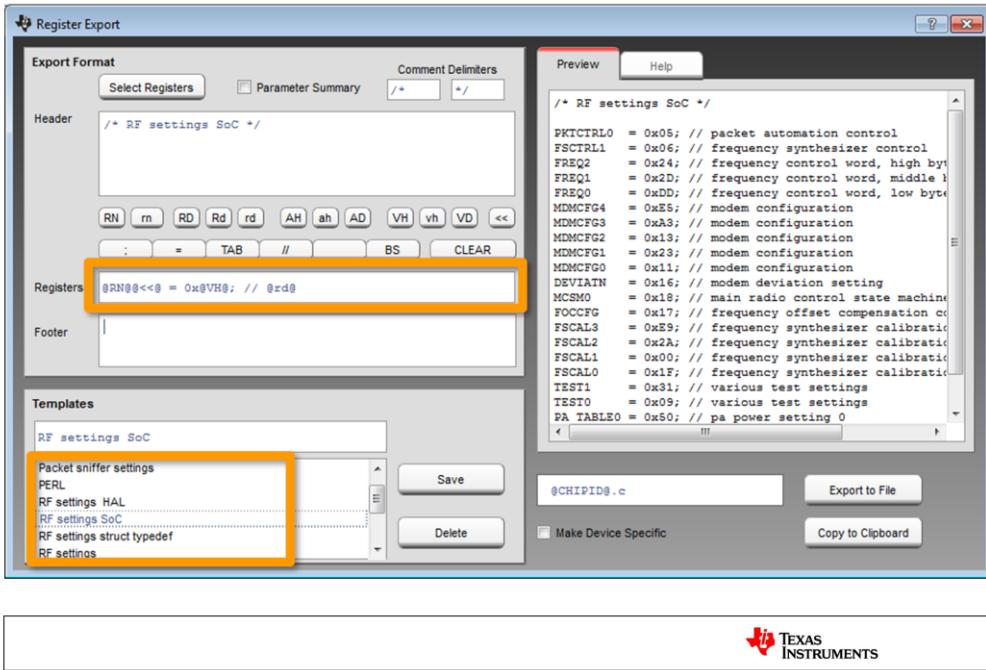
Copy the exported code into your own source code, either as a header file, c file, stub or any appropriate format



The concept of the register export functionality is straight forward:

Once you have a working set of register settings, click the **Register export** button in the register view. A small widget will let you select exactly the registers you are interested in and export them in any format you like to an external file (or simply directly to the Windows clipboard). Next, you can import those settings in your own software, recompile and download to the microcontroller on your own target board or on the development kit. This process makes it easy to quickly change settings and test the effect on the system.

Exercise 3: Register Export



To start the Register Export widget, click on the **Register export** button in the Register View panel or select Register Export from the menu ("File → Register Export"). Note that it is the currently active register values that will be used by the exporter.

In the export widget, you have the option to select which registers to export. The default selection includes all registers with a value different from the reset value.

Next, you can choose the format of the output. You have the option of selecting the format from one of the many predefined templates or create your own.

A new template can be defined by writing the export format to the Registers field. Next, provide a name for the template and press the **Enter** key on the keyboard to add the new template to the list. The template file will be stored in <installation dir>\config\codeexport. For any modifications done to the template, click the **Save** button for simple reuse at a later stage.

The "Make Device Specific" check box can be used if the template only applies to one particular chip. The template will then only be visible when you work with that chip.

The existing templates can be seen in the bottom left part of the window. Double clicking on one of them generates the code in the output window on the right side. The code can now be copied to the clipboard or written to a file.

Description of the export format fields

Header

The header will be added to the beginning of the exported code.

Registers

This field defines the style for each of the selected registers. You can select

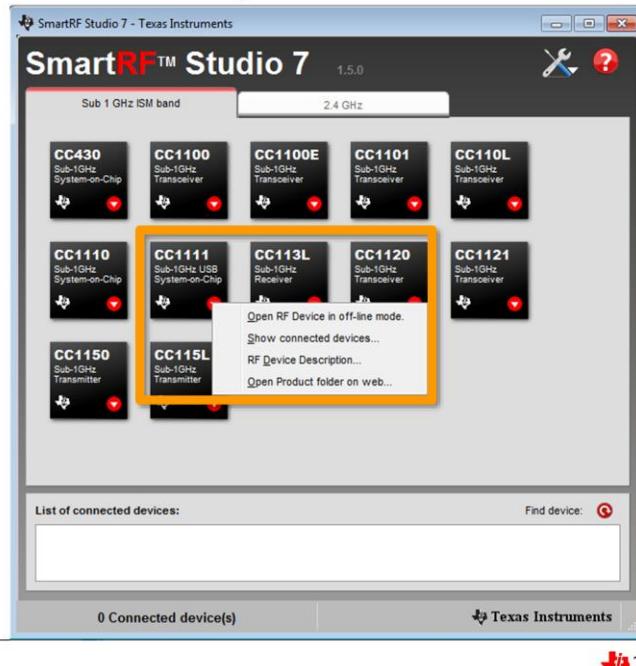
- Register address (AH, ah or AD) (modifiers for upper/lower case and/or hexa-/decimal format)
- Register name (RN, rn)
- Register value (VH, vh, VD)
- Register description (RD, Rd, rd)

The syntax to be used for the exported register values can be found in the help system. Most of the tags can be inserted using the buttons above the input field.

Footer

The footer will be appended to the end of the exported code.

Exercise 3: Export to Packet Sniffer



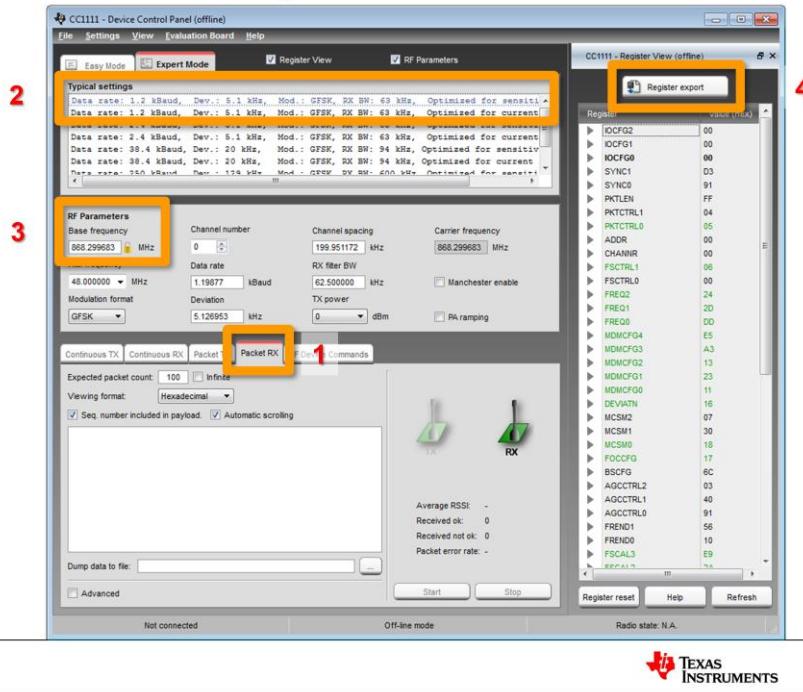
In the rest of this exercise we will look at a practical use case of the register export functionality.

We will export registers from SmartRF Studio to configure a capture device used by the SmartRF Packet Sniffer application. The goal is to use the CC1111 USB Dongle to capture, or sniff, packets transmitted by a CC1110 device.

Although the radio devices are very similar, please remember to use CC1111 register values for the CC1111 device and CC1110 register values for the CC1110 device. Also note that the fundamental RF parameters (frequency, data rate and modulation) for the two devices must be the same to make it possible for the CC1111 to receive the packets sent by the CC1110 device.

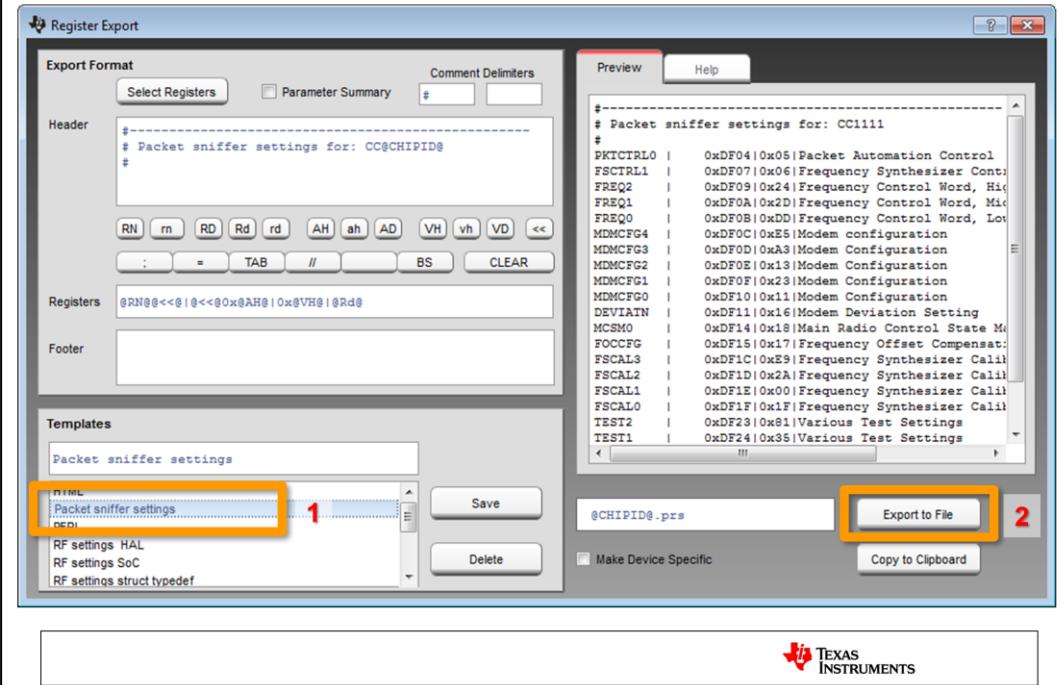
Start by opening the Device Control Panel for CC1111 in offline mode. Simply double click on the device icon (if no device is connected) or right-click on the CC1111 icon and select “Open RF Device in off-line mode”.

Exercise 3: Export to Packet Sniffer



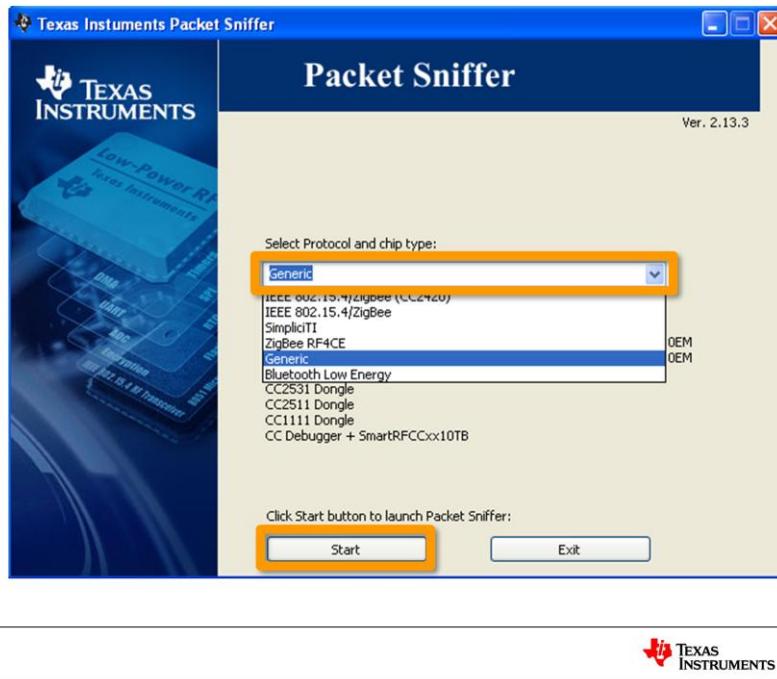
1. Go to the Expert Mode panel and select the Packet RX panel.
2. Select the first typical setting from the list (for the sake of this example).
3. Select an appropriate frequency.
4. Click the **Register export** button

Exercise 3: Export to Packet Sniffer



1. In the templates panel, select the template called "Packet sniffer settings" by double clicking. The register values can now be seen in the Preview window.
2. Click the **Export to File** button. A file dialog window will open. Give the file a name, e.g. CC1111_sniff_settings.prs. Make sure you use the extension .prs for the file name.

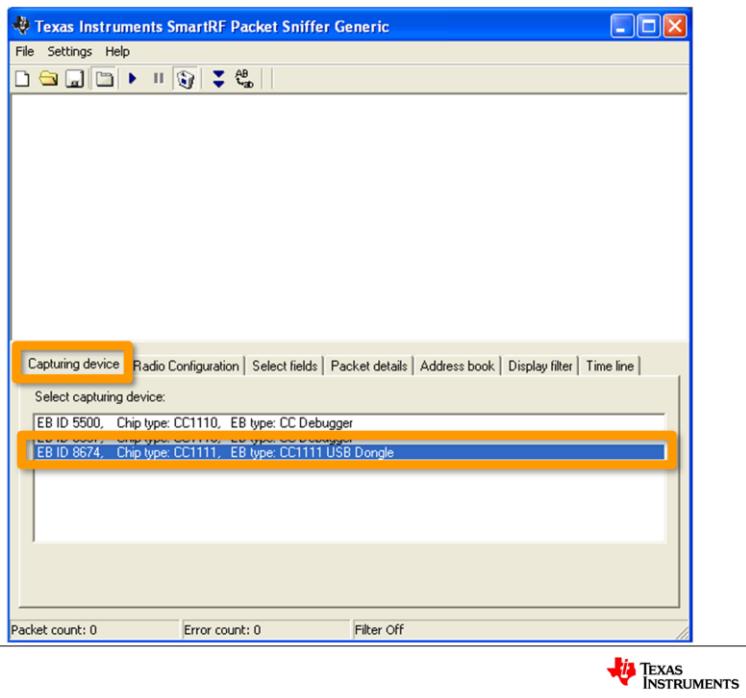
Exercise 3: Packet Sniffer



The register settings exported from SmartRF Studio 7 can now be used to configure the Packet Sniffer (or more correctly the capture device) with the same settings.

- Start the Packet Sniffer application. Select "Generic" protocol as shown in the picture.
- Click the **Start** button to launch the Packet Sniffer application.

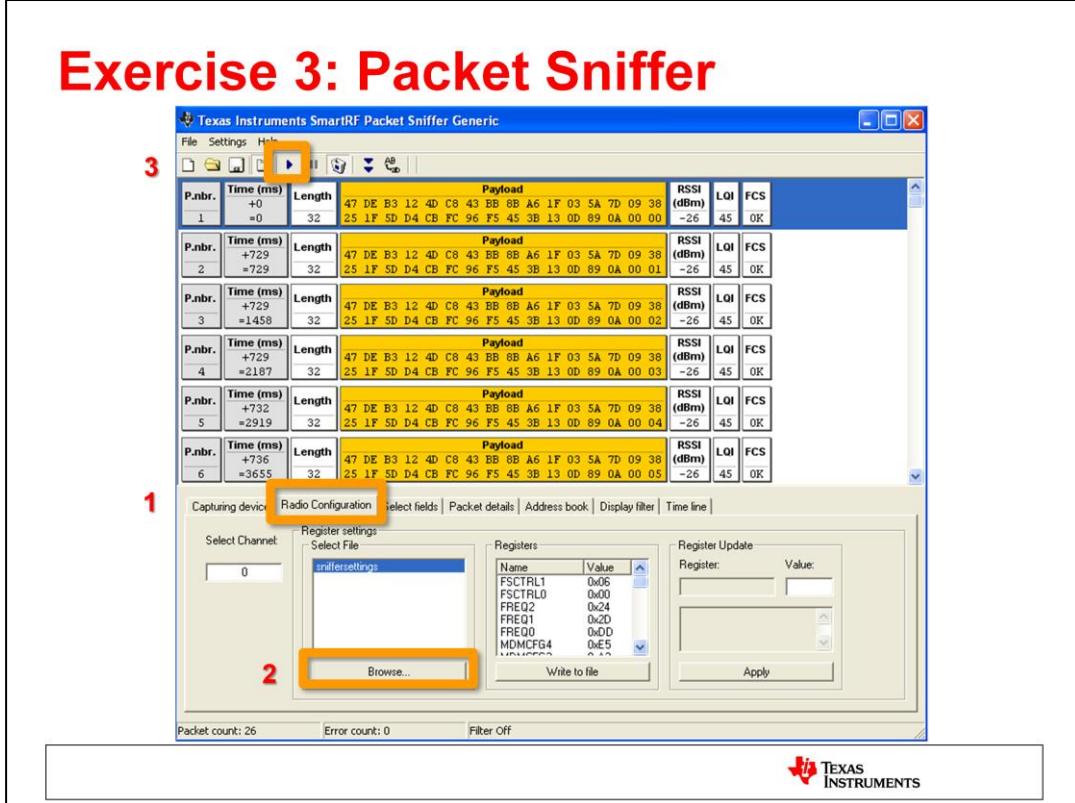
Exercise 3: Packet Sniffer



Connect the CC1111 USB dongle to the PC. Make sure the dongle has been programmed with the Packet Sniffer firmware. The SmartRF Packet Sniffer User Manual describes how this can be done. In short, locate the file C:\Program Files (x86)\Texas Instruments\SmartRF Tools\Packet Sniffer\bin\general\firmware\sniffer_fw_ccxx11.hex and program it on the dongle using SmartRF Flash Programmer and a CC Debugger.

If everything is done correctly, you should be able to see the CC1111 dongle in the Capturing device panel. Select the dongle as the capture device by clicking on it in the list.

Exercise 3: Packet Sniffer



The exported register settings from SmartRF Studio 7 will now be imported to the Packet Sniffer:

1. Select the Radio Configuration panel
2. Click **Browse** and locate the file that was exported from SmartRF Studio (the *.prs file). The register values from the selected file can be seen in the Registers panel. Note that it is possible to modify the value of individual registers directly (these changes will not be saved to the *.prs file).
3. Press the **play** button (marked with a small arrow) at the top to start capturing packets.
4. Use SmartRF Studio 7 and one of the CC1110 devices to send packets. Open the device control panel for the connected CC1110 device and go to the Packet TX panel. Make sure that the same RF parameters as for CC1111 is used, so again for the sake of this example, select the first typical setting in the list of predefined settings. Press the **Start** button to start sending packets.

The transmitted packets from the CC1110 device can now be observed in the Packet Sniffer application.

Overview

- Introduction to SmartRF Studio 7
- Installation and Hardware Setup
- Exercise 1: Measure Signal Strength
- Exercise 2: Measure Packet Error Rate
- Exercise 3: Register Export
- **Exercise 4: XML Configuration**
- Exercise 5: Command Strobe Panel



Exercise 4: XML Configuration

Purpose of the exercise:

Learn how to make your own customized settings for use directly in SmartRF Studio.

- SmartRF Studio 7 use XML for configuration files:
 - Register definitions and descriptions
 - Strobe commands
 - Typical settings
 - Device description
 - Packet format description (Easy Mode)
- Motivation for using XML
 - Configurability, scalability and maintainability
 - Text based database
 - User can easily extend and modify already existing files or add their own



SmartRF Studio uses XML to store chip specific information and configuration settings. Examples of the data stored in XML are register definitions and descriptions, strobe commands, typical settings, power table settings (mapping between output power in dBm and register values), packet format in Easy Mode and device descriptions.

The XML configuration files are found in the folder \SmartRF Studio 7\config\xml under the installation directory for SmartRF Studio 7.

Exercise 4: XML File Example

- The example on the right shows an excerpt from a file defining typical register settings for a device.
- The combination of register values gives specific radio parameters.
- Adding your own register settings is easy

```
<Registersetting>
  <Name>868MHz, 1.2kbps, 2-FSK</Name>
  <Description>Narrow band</Description>
  <Register>
    <Name>SYNC_CFG0</Name>
    <Value>0x17</Value>
  </Register>
  <Register>
    <Name>DEVIATION_M</Name>
    <Value>0x6</Value>
  </Register>
  <Register>
    <Name>MODCFG_DEV_E</Name>
    <Value>0x3</Value>
  </Register>
  <Register>
    <Name>DCFILT_CFG</Name>
    <Value>0x1C</Value>
  </Register>
```



The example above shows the essence of an XML file defining typical settings for a device. A "register setting" is a collection of registers, where each register has an identifier (`<name>`) and value (`<value>`). The name of the register collection is what will appear in SmartRF Studio. The description field can be used for additional information about the specific configuration.

It is possible to add your own `<Registersetting>` by setting up a collection of registers as shown in the example.

One XML file can contain multiple sets of register settings.

Exercise 4: XML Configuration

- The content of the highlighted areas are defined in XML files.
- Modifying the XML files will modify what you see in SmartRF Studio



In this exercise, we will look at the specific XML files used in the Easy Mode panel for CC1110 and show how they can be modified to fit your own system.

In Easy Mode the configurations (settings) and packet formats are defined by XML files:

```
<installation_dir>\config\xml\cc1110\easyemode_settings.xml  
<installation_dir>\config\xml\packet\packet_simpliciti_ping.xml
```

Note that for some other devices, there are multiple files for the settings. Instead of defining all the various sets of register settings in one file, each set is defined in individual files. The settings shown in the Easy Mode panel are prefixed with "em_" (easy mode). The settings shown in the Expert Mode panel are prefixed with "ts_" (typical).

Exercise 4: XML Configuration

Tasks for this exercise:

- Create a new packet format for Easy Mode
- Create a custom easy mode setting using the new Easy Mode packet format
- Send packets between two devices using the customized XML files



In order to learn exactly how the XML file works and looks like, we will modify some of the XML files and observe the effect.

The modifications will consist of:

- Creating a new packet format for Easy Mode
- Create a custom easy mode setting using the new Easy Mode packet format

Finally, we will use the new configuration files in practice by sending packets between two devices.

Exercise 4: Create new Packet in XML

- Locate the packet_simpliciti_ping.xml file
- Make your own copy based on that file.
- Modify the values in the fields defined by the packet.

```
<packet>
  <Name>Ping</Name>
  <Description>SimpliciTl packet</Description>
  <Field>
    ...
  </Field>
  <Field>
    <Name>Source Address</Name>
    <Description>Source Address</Description>
    <Type>Hex</Type>
    <Length>4</Length>
    <TextLength>10</TextLength>
    <Min>0</Min>
    <Max>0xFFFFFFFF</Max>
    <BgColor>#ffffff</BgColor>
    <TextColor>#000000</TextColor>
    <Default>0x0000BBBB</Default>
  </Field>
  ...
</packet>
```



Create a new packet format for Easy Mode:

- Locate the file <installation_dir>\config\xml\packet\packet_simpliciti_ping.xml
- Copy of the file and rename it to e.g. my_packet.xml
- Open the my_packet.xml file with your preferred text editor (Notepad will do, but try out e.g. XML Notepad from Microsoft).
- Directly under the <packet> tag, change the value for the <Name> tag to something different, e.g. “Pong”.
- Find the <Field> tag with the <Name> set to “Source Address”
- Change the value of <Default> to a different value. This will change the value of the Source Address in the packet.
- You can do similar modifications of the other fields.
- Save the XML file

The new packet format is now ready for use.

NB! Note that you will need administrator rights on your PC to modify files located under C:\Program Files.

Exercise 4: Create new Settings

```
<registersettings>
<Setting>
    <Name>My own custom settings</Name>
    <Description>...</Description>
    <Packet>my_packet</Packet>
<Register>
    <Name>FSCTRL1</Name>
    <Value>0x0C</Value>
</Register>
<Register>
    <Name>FSCTRL0</Name>
    <Value>0x00</Value>
</Register>
<Register>
    <Name>FREQ2</Name>
    <Value>0x22</Value>
</Register>
<Register>
    <Name>FREQ1</Name>
    <Value>0xB1</Value>
</Register>
...
...
```

Change the value of
<Name> and <Packet>



Create a custom easy mode setting using the new Easy Mode packet format:

- Locate the file <installation_dir>\config\xml\cc1110\easy.mode_settings.xml
- Create a backup of the file before starting to modify the file
- Open the file with your preferred text editor
- In the beginning of the file, locate the first <Setting> tag.
- Change the value of the <Name> tag to something else, for instance "My setting".
- Change the value of the <Packet> tag to the file name (without extension) of the new packet created in the previous step
- Save the file and restart SmartRF Studio 7. Restarting is required for Studio to reload the XML files.
- Observe the changes in the Device Control Panel (Easy Mode) for CC1110

Exercise 4: XML Configuration

- Restart SmartRF Studio
- Observe changes in Easy Mode
- Send packets between two devices using the modified Easy Mode configuration



The modifications in the XML files will take effect the next time you start SmartRF Studio.

Open the Device Control Panel for CC1110, go to the Easy Mode panel and verify that the new setting is listed together with the other predefined settings. The packet should also have changed.

Test the new configuration by sending packets with CC1110 and see that the new packet format is being used.

Feel free to do additional modifications to the packet format and register settings to get a set of configuration files that matches your own system. This will make it easier when using SmartRF Studio for testing your own software (or hardware) against known good software (Studio) and hardware (boards from the development kits).

Overview

- Introduction to SmartRF Studio 7
- Installation and Hardware Setup
- Exercise 1: Measure Signal Strength
- Exercise 2: Measure Packet Error Rate
- Exercise 3: Register Export
- Exercise 4: XML Configuration
- **Exercise 5: Command Strobe Panel**



Exercise 5: Command Strobe Panel

Purpose of the exercise:

- Learn how to use SmartRF Studio for detailed control of the device
 - Send individual command strobes to the radio
 - Learn how the radio's packet engine works



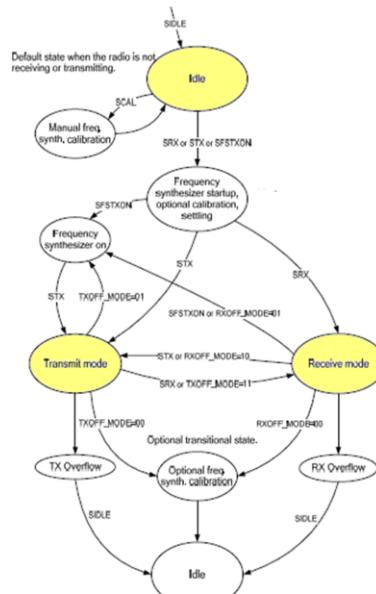
In this exercise, we will explore some of the more advanced features of SmartRF Studio.

With SmartRF Studio it is possible to send and receive packets manually - in the sense that each action required on the chip to send and receive packets is initiated manually from SmartRF Studio. With this feature it is possible to test or verify the same radio configuration and command sequence which is used in the software that controls the chip.

To get the most out of this exercise, it is worthwhile having a basic understanding of what "command strobes" are and what a "packet engine" is.

Exercise 5: Command Strobe Basics

- The figure on the right shows a simplified state diagram of the CC1110 radio.
 - Other radios have a very similar state machine. Refer to the datasheet for details
- A command strobe is used to change the state of the radio
- Strobes are needed to enter the active states for transmitting and receiving packets



TEXAS INSTRUMENTS

A command strobe is a single byte instruction, or command, to the radio. The command is used to change the state of the radio from e.g. an IDLE state to an ACTIVE state. Examples of active states are TX (transmit) or RX (RX). Similarly, a strobe can be used to set the radio back to IDLE mode or put it in SLEEP mode.

Strobe commands for CC1110:

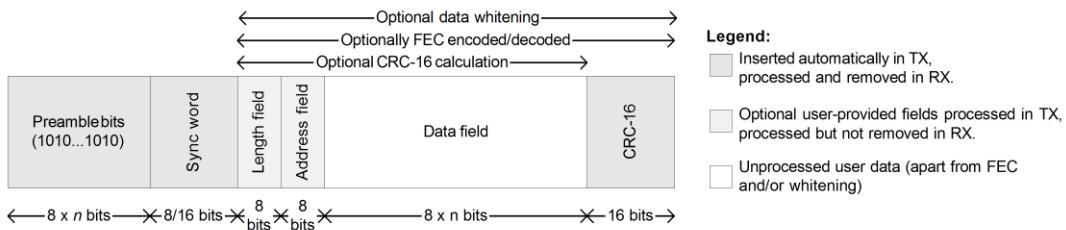
STX	Enable TX
SRX	Enable RX
SIDLE	Go to IDLE state (frequency synthesizer is turned off)
SCAL	Calibrate the frequency synthesizer and turn it off.
SFXON	Enable and calibrate the frequency synthesizer (for quick transition to TX)

Other radios have other commands, for instance commands to flush the receive and transmit FIFOs. You'll find these for e.g. the CC1120 and CC1101 devices.

One thing to note about how the radio works is what happens when a packet has been transmitted or received. Should the radio stay in the current ACTIVE state or should it do something else? Most of our radios have a couple of registers to control the state the radio shall enter when a packet has been received. On the CC1110, this would be the MCSM1 register (Main Radio Control State Machine configuration register 1). It has two fields, RX_OFF and TX_OFF that can be used to control what the next state will be. This feature can be used to simplify the software needed to control the radio.

The default OFF mode is IDLE.

Exercise 5: Packet Engine Basics



- Some of the services provided by the packet engine
 - Preamble generation
 - Automatic sync word insertion and detection
 - Automatic CRC insertion and verification
 - Address filtering
 - Flexible packet length (fixed, variable, or infinite)
 - Interrupts to the MCU on predefined events

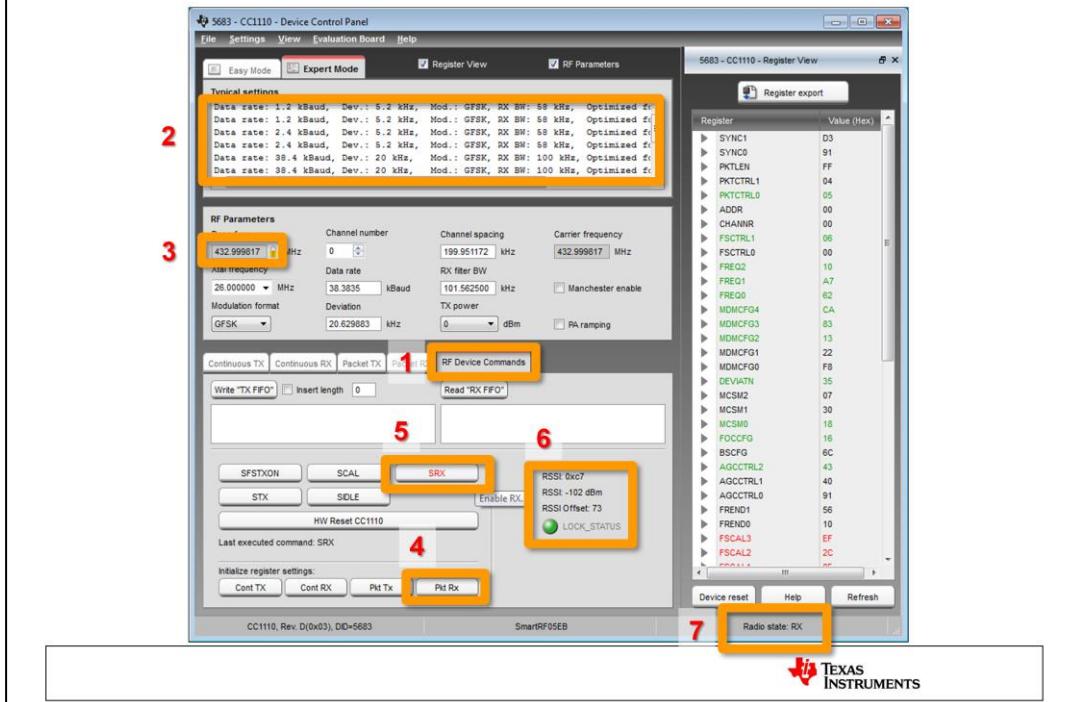


Most of the new CCxxxx radios have a built in packet engine that offloads the MCU with common tasks related to handling of a packet. The behavior of the packet engine is determined by several configuration registers and it is flexible in order to support a wide variety of packets.

Using the packet engine reduces the complexity of the software since the radio will automatically do some checking and verification that you otherwise would have to implement in software.

Knowing what the packet engine can and cannot do is important for writing efficient code for the radio. We will see on the next pages how you can use SmartRF Studio to explore and possibly modify the behaviour of the packet engine. With the addition of being able to send individual command strobes, SmartRF Studio can be used to mimic the behaviour of your own low level radio control software .

Exercise 5: Command Strobe Panel



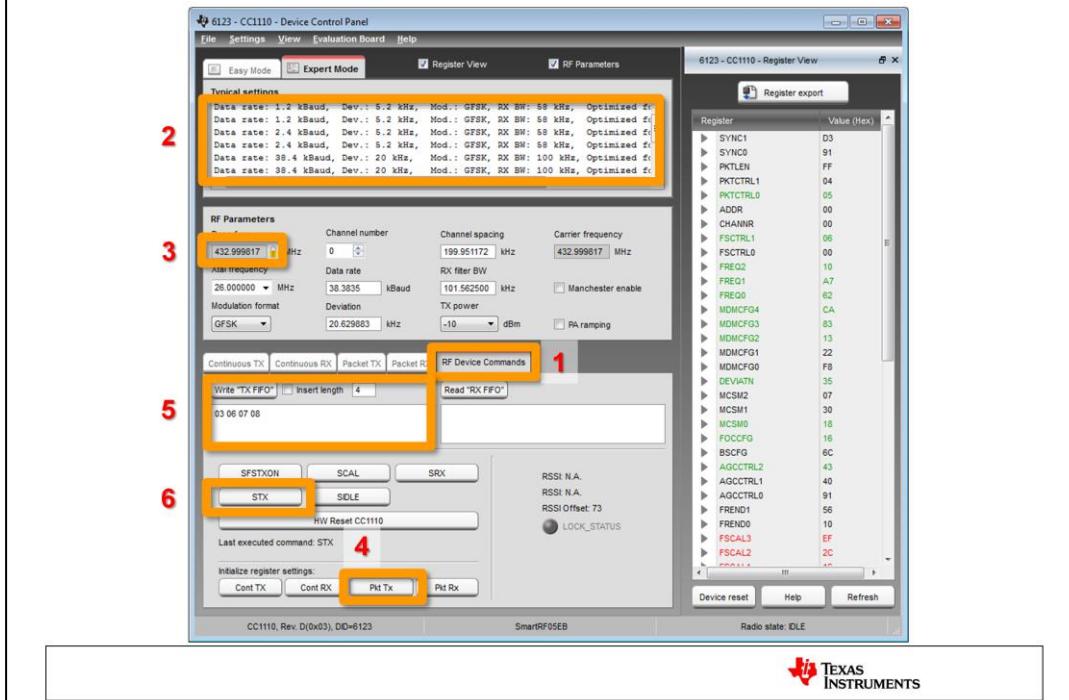
In this exercise, we will use the RF Device Commands panel to send and receive packets manually, i.e. perform all the individual actions required to

- Configure the radio correctly
- Write the correct packet payload to the radio
- Set the radio in the correct state for sending and receiving the packet
- Read and interpret the contents of the received packet

Let's start with preparing the receiver.

1. Open the RF Device Commands panel.
2. Select any radio configuration from the list of typical settings.
3. Select a frequency.
4. Click on the **Pkt Rx** button in the lower part of the screen. This will configure the registers that controls the behavior of the packet engine. If you skip this step, you should write the appropriate register values manually via the register view. This will allow detailed control of how the radio should operate when receiving a packet. It might be useful to start with the default configuration and modify individual settings afterwards.
5. Click the **SRX** command strobe button to set the device in RX.
6. The status part of the window will show that the device is in lock (frequency synthesizer calibrated and active) and it will show the strength of any RF signal observed by the radio.
7. The status bar of the window will also display the current status of the chip. While waiting for a packet, it should be in RX state. If you use the default configuration, the packet engine will enter IDLE once a packet has been received.

Exercise 5: Command Strobe Panel



The following steps describes how to transmit a packet. Do this for the other device while the first is in RX.

1. Open the RF Device Commands panel.
2. Select the same radio configuration as the receiver from the list of typical settings.
3. Select the same frequency as the receiver.
4. Click on the **Pkt Tx** button in the lower part of the screen. This will configure the registers that controls the behavior of the packet engine. If you skip this step, you should write the appropriate register values manually via the register view. This will allow detailed control of how the radio should operate when transmitting a packet. It might be useful to start with the default configuration and modify individual settings afterwards.
5. Write some bytes in the TX FIFO panel. This is the payload of the packet that will be sent over the air. As described previously, the packet engine can be configured to send fixed or variable length packets. The default configuration is variable length, meaning that the first byte of the payload should indicate the length of the packet (not including the packet length byte itself). By writing “03 06 07 08” you indicate that the packet length is 3 and that the packet payload is “06 07 08”.

Note the checkbox called “Insert length”. If this is checked, the length byte will be inserted automatically by Studio (i.e. you do not have to add it yourself in the TX FIFO window. In that case, it is sufficient to enter the payload, e.g. “06 07 08”.

Click the **Write TX FIFO** button. This will write whatever you have entered in the TX FIFO text field to the appropriate memory area of the chip (*).

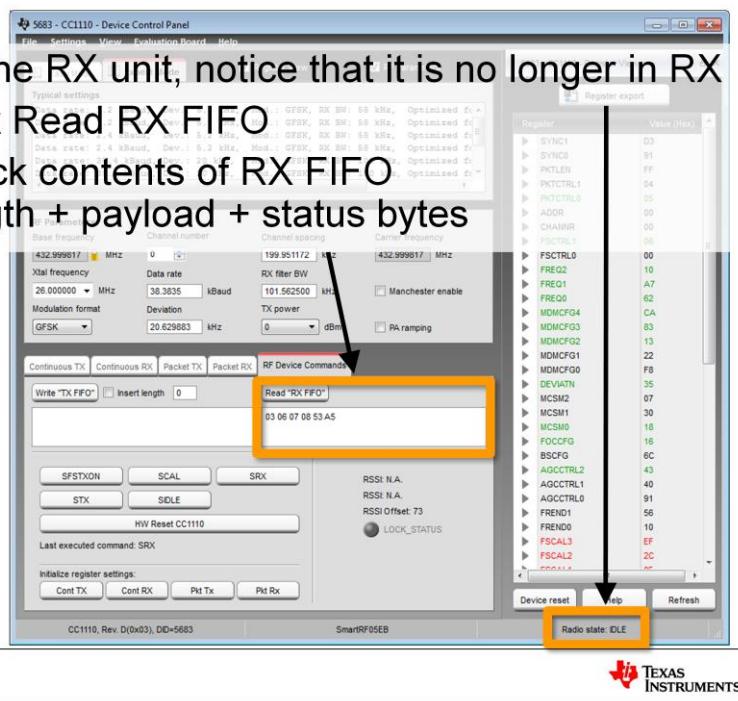
6. Click the **STX** command strobe button to set the device in TX. The packet in the FIFO will now be sent over the air.

(*) The CC1110 does not have a traditional TX FIFO. However, we mimic the behavior of a FIFO by setting up a memory area in RAM on the SoC to store the contents of the packet. When you click the **Write TX FIFO** button, the packet is written to RAM. When clicking the **STX** button, we set up a DMA channel to transfer the packet from RAM to the radio’s data input register (RFD). On a transceiver, like the CC1101 or C1120, we use the real FIFO in the chip.

Exercise 5: Command Strobe Panel

1. On the RX unit, notice that it is no longer in RX
2. Click Read RX FIFO
3. Check contents of RX FIFO

Length + payload + status bytes



Go back to the control panel of the receiver and see what has happened.

1. The device state is now IDLE, indicating that the radio has received something.
2. Click the **Read RX FIFO** button. The contents of the FIFO (i.e. the data received by the radio) will be displayed in the text area under the button.
3. If everything went well, you should see something similar to "03 06 07 08 53 A5". So what does this mean:

03: This is the packet length. When operating with variable length packets, the radio will use the first byte in the packet to determine how many [additional] bytes it should expect to receive. In this case, it will receive 3 more bytes over the air.

06 07 08: This is the packet payload

53 A5: These two status bytes are appended automatically by the receiver when the packet is received. This feature is optional and is controlled by PKTCTRL1.APPEND_STATUS.

The first byte is the raw RSSI value (in hex) from the chip. The coding of the RSSI value is different from one chip to another. For CC1110, the raw value is a signed 8 bit value with 0.5 dBm resolution. To get the real RSSI value in dBm, you need to divide the value by two and subtract the RSSI offset value (see datasheet for details). In this case, the offset value is 73 (decimal). Knowing that 53 in hex is 83 in decimal notation, we can calculate the signal strength of this packet:

$$83/2 - 73 = -31.5 \text{ dBm}$$

The most significant bit of the second byte is the CRC valid bit. If the bit is 1, CRC is OK. If it is 0, there is a CRC error in the received packet (i.e. one or more bit errors). The CRC check is done automatically if this feature is enabled (for this to work as expected, it also requires that the transmitter is configured to append a CRC to the packet it sends over the air).

The 7 least significant bits in the second status byte is the Link Quality Indicator, which is an estimate of how easily a received signal can be demodulated. Higher values are better. See the datasheet for more details about LQI.

You can now experiment with the packets transmitted over the air and see how modifications of the registers controlling the packet engine affects the receiver and transmitter. Try for instance to change to fixed packet length (relevant registers are PKTLEN and PKTCTRL0.LENGTH_CONFIG).

**Thank you for going through this
tutorial!**

**For more information and support,
visit the TI E2E Forum at
www.ti.com/lprf-forum**



Thank you for going through this tutorial!

For more information and support, visit the TI E2E Forum at www.ti.com/lprf-forum

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

Audio	www.ti.com/audio
Amplifiers	amplifier.ti.com
Data Converters	dataconverter.ti.com
DLP® Products	www.dlp.com
DSP	dsp.ti.com
Clocks and Timers	www.ti.com/clocks
Interface	interface.ti.com
Logic	logic.ti.com
Power Mgmt	power.ti.com
Microcontrollers	microcontroller.ti.com
RFID	www.ti-rfid.com
RF/IF and ZigBee® Solutions	www.ti.com/lprf

Applications

Communications and Telecom	www.ti.com/communications
Computers and Peripherals	www.ti.com/computers
Consumer Electronics	www.ti.com/consumer-apps
Energy and Lighting	www.ti.com/energy
Industrial	www.ti.com/industrial
Medical	www.ti.com/medical
Security	www.ti.com/security
Space, Avionics and Defense	www.ti.com/space-avionics-defense
Transportation and Automotive	www.ti.com/automotive
Video and Imaging	www.ti.com/video
Wireless	www.ti.com/wireless-apps

[TI E2E Community Home Page](#)

[e2e.ti.com](#)

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2011, Texas Instruments Incorporated