WirelessUSB™ NX Demonstration Kit Guide

Doc. No. 001-90000 Rev. *A

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): 800.858.1810
Phone (Intnl): 408.943.2600
http://www.cypress.com

Copyrights

Trademarks

Source Code

Disclaimer

Contents

1. Introduction

The WirelessUSB™ NX Demonstration Kit allows you to develop wireless applications in the 2.4-GHz ISM band, using Cypress's WirelessUSB™ NX (Cypress's fourth-generation, ultra-low-power, 2.4 GHz, proprietary radio).

1.1 Kit Contents

The WirelessUSB NX (WUSB-NX) Demonstration Kit includes:

- Two CY3668 development boards with integrated LCD panels (also called WUSB-NX demonstration kit base boards)
- Two WirelessUSB NX modules
- Two enCoRe V modules
- Two 3.3V LCDs
- Two 12-V power adaptors
- Two USB Standard A to Mini-B cable
- One PSoC® MiniProg 1 programmer
- Ten jumper wires
- Quick Start Guide for data transfer demonstration

1.2 PSoC Designer™

Cypress's PSoC Designer is an easy-to-use software development Integrated Design Environment (IDE) that introduces a hardware and software codesign environment based on schematic entry and embedded design methodology.

With PSoC Designer, you can automatically place and route select components and integrate simple glue logic normally residing in discrete muxes.

1.3 Additional Learning Resources

Visit http://www.cypress.com for additional learning resources in the form of datasheets and application notes.

1.4 Document Conventions

Table 1-1. Documentation Conventions for User Guides

Convention	Usage
`Courier New`	Displays file locations and source code: `C:\ …cd\icc\.`
Italics	Displays file names and reference documentation: *sourcefile.hex*
[**Bracketed, bold**]	Displays keyboard commands in procedures: [**Enter**] or [**Ctrl**] [**C**]
File > Open	Represents menu paths: File > Open > New Project
Bold	Displays commands and selections, and icon names in procedures: Click the **Debugger** icon, and then click **Next**.

2. Software Prerequisites

This chapter lists the software and tools required for using the WUSB-NX Demonstration Kit.

2.1 PSoC Designer 5.4

The example project in Wireless USB demo kit needs to be built using PSoC Designer. PSoC Designer is an IDE used for firmware development on the enCoRe V MCU. You can download PSoC Designer 5.4 from the Cypress download site.

2.2 PSoC Programmer 3.20

PSoC Programmer is used to program the hex file generated by PSoC Designer, into enCoRe V MCU. You can download PSoC Programmer 3.20 from the Cypress download site.

2.3 .NET Framework 2.0

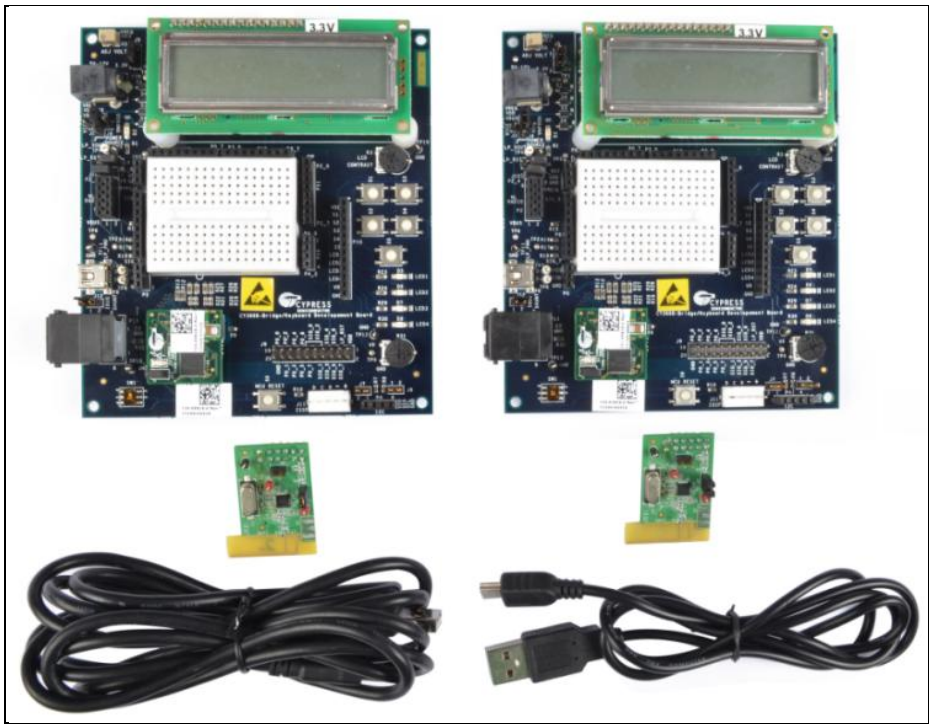
The Wireless USB-NX demo kit uses a Windows based application which requires .NET framework 2.0 to be installed on the system. The .NET Framework can be downloaded from the Microsoft downloads site.

3. Kit Operation

3.1 Install Hardware

WARNING: Static discharges from the human body can easily reach 20,000 volts. This can damage the components on the DVK. Make sure that any static charge is discharged before touching the hardware. Ensure the DVK is powered off before making any changes to the connections.

Figure 3-1. WirelessUSB NX Demonstration Kit Contents



3.1.1 Jumper Settings

The following are the required jumper settings for the WUSB-NX demonstration kit base board:

Table 3-1. Jumper Settings for WUSB-NX Demo Kit Base Board

Jumper	Setting
J2	Place a jumper between V_{DD} and V_{REG}
J5	Place a jumper between pins 2 and 3 (for 3.3-V operation)
J6	Place a jumper between pins 2 and 3
J10	Place a jumper between pin 1 and 2
J12	Place a jumper between pin 1 and 2

Leave other jumpers open.

The following are the required jumper settings for the WUSB-NX modules:

Table 3-2. Jumper Settings for WUSB-NX Modules

Jumper	Setting
J1	Leave open
J2	Place a jumper

3.2 Programming and Emulation Process

Note: You can program the kit using CY3217-MiniProg1, CY8CKIT-002 PSoC® MiniProg3 Program and Debug Kit, or ICE-Cube. However, the ICE Cube is required for emulation. ICE-Cube is available in the PSoC Development Kit (CY3215-DK) and can be purchased from Cypress's online store. MiniProg 3 is available in the PSoC MiniProg 3 Program and Debug Kit (CY8CKIT-002), which can be purchased from Cypress's online store.

3.2.1 Programming Steps

Important

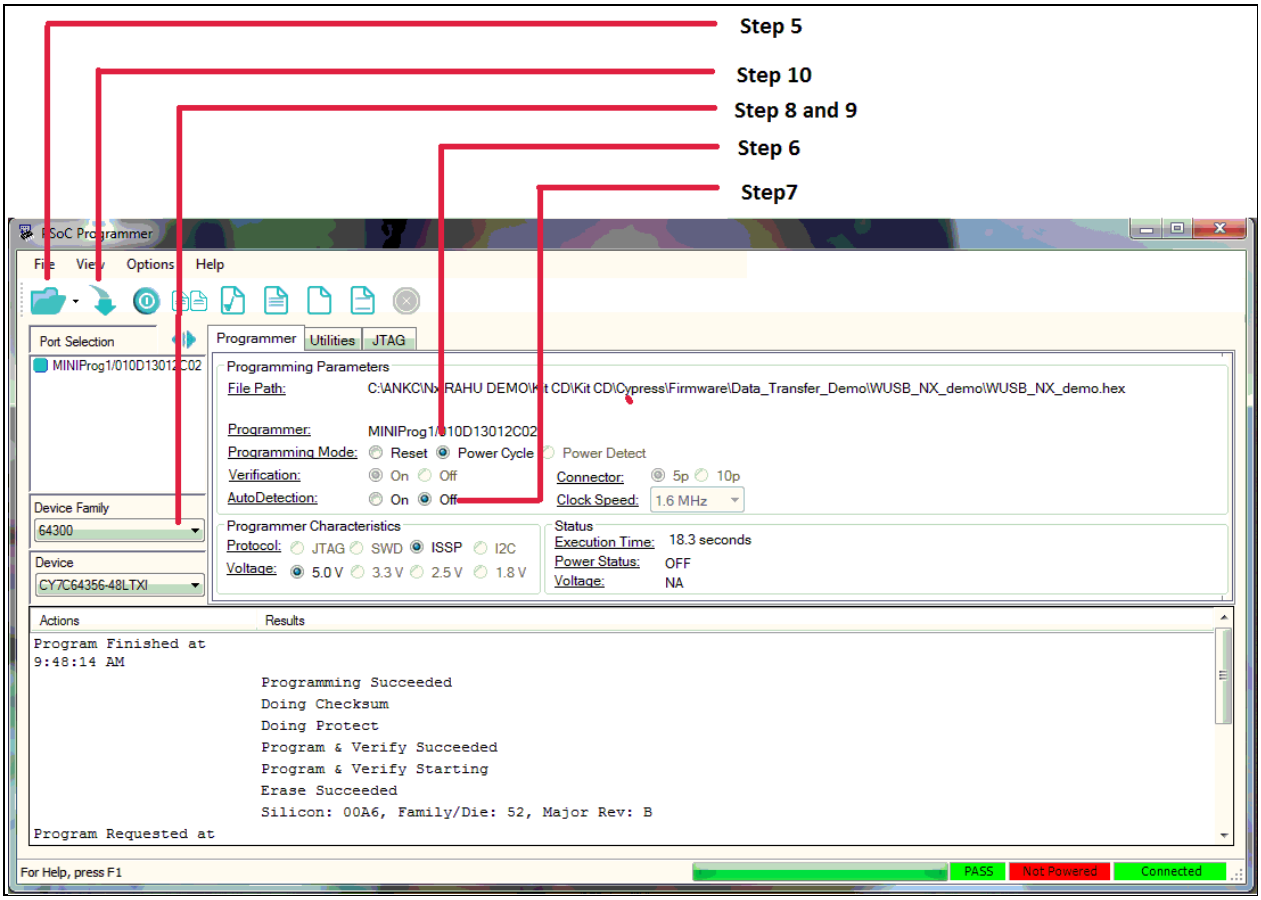
- Disconnect USB cable or 12-V power adapter from the board.
- Remove the WUSB-NX module from the development board before programming the firmware.
- Make sure to position the SW1 switch slider away from the LCD panel (see enCoRe V Programming Selector in Section 4.2.6).

Follow these steps to program the kit:

1. Connect one end of the USB cable to MiniProg 1, and the other end to the computer.
2. Connect MiniProg 1 to the 5-pin ISSP header on the CY3668 DVK board.
3. Launch PSoC Programmer.

   Note: If the **Firmware Update required at ...** dialog box is displayed, upgrade MiniProg 1's firmware by clicking **Utilities > Upgrade Firmware** before proceeding with the remaining steps.

Figure 3-2. PSoC Programmer



4. Click **File Load** and browse for the desired binary (.hex) file.

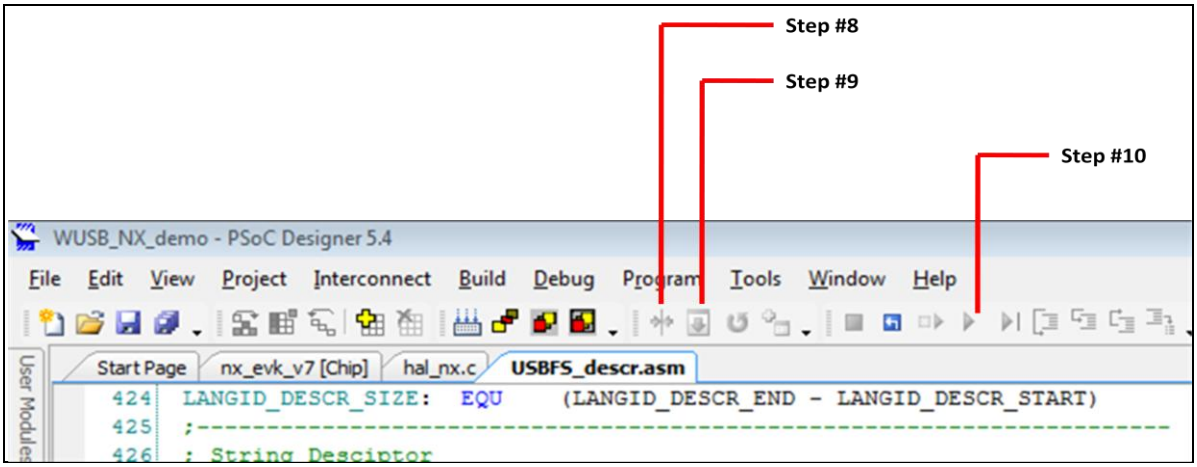
   Note: The .hex file for every PSoC Designer project is located at *<project directory>\<project name>\output\<project name>.hex*.
5. For Programming Mode, select **Power Cycle**.
6. Turn off Auto-Detection.
7. Select **64300** from the **Device Family** list.
8. Select **CY7C64300-48LTXI** part from the **Device** list.
9. Click the Program button. PSoC Programmer goes through programming and verification steps. Results are displayed in the bottom half of the screen.
10. When programming is complete, disconnect the MiniProg 1 from the WUSB-NX demo base board.

3.2.2 Emulation Steps

1. Launch the desired .APP project file project from *<project directory>\<project name>.app.*
2. Power ICE-Cube with a 12 V adaptor and connect it to the computer via a USB cable.
3. Connect one end of the blue CAT 5e cable to the ICE-Cube and the other end of the CAT 5e cable to the base board.
4. Select Build > Generate Configuration Files for <Project name> Project.
5. Select **Build > Build** <Project name> **Project** or press **[F7]**.
6. Configure the debug port setting. Select **Project > Settings > Debugger**, and then select ICE-Cube.

7. Select the Pod Power Source and Pod Supply Voltage:
   - If External only is selected, use the jumper setting as mentioned in the Jumper Settings section on page 8.
   - If ICE may power pod is selected, choose 3.3 V as Pod Supply Voltage, and place a jumper between VDD and VICE at J3.

Figure 3-3. Debugging with PSoC Designer



8. Select **Debug > Connect**, or click the **Connect** icon.
9. Select **Debug > Download to Emulator…**, or click the **Download to Emulator…** icon.
10. Select **Debug > Go**, press **[F5]**, or click the **Go** icon.

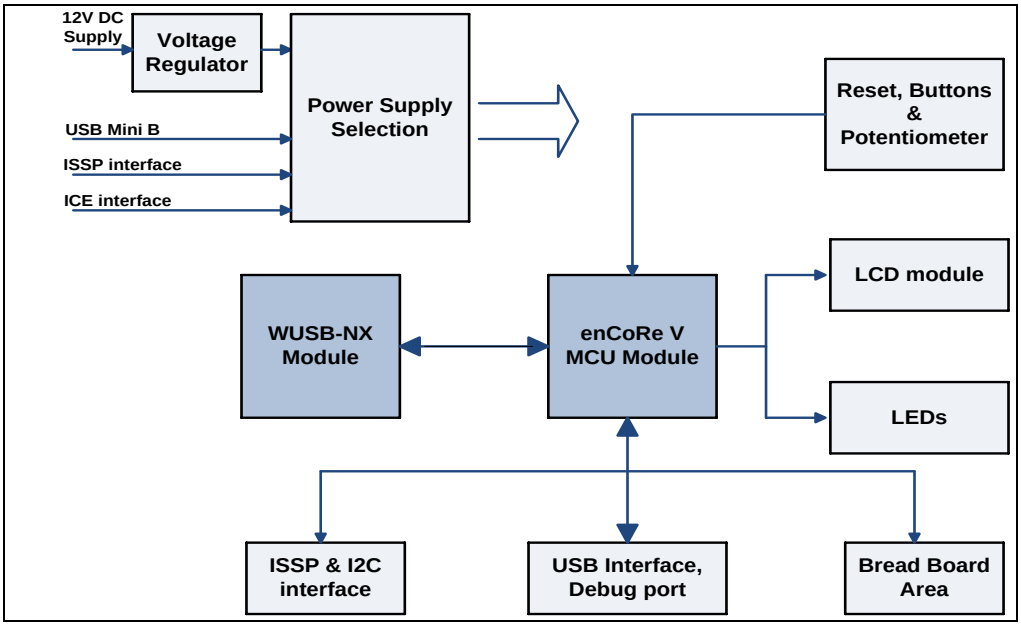
4. Hardware

4.1 System Block Diagram

The WUSB-NX demonstration kit base board consists of the following blocks:

- Voltage Regulator and Power Supply Selection
- MCU header
- Module header (for WUSB-NX radio)
- LCD module
- LEDs
- Reset, push buttons, and potentiometer
- Breadboard area
- USB Mini-B interface
- Debug port
- ISSP interface
- I^2C interface

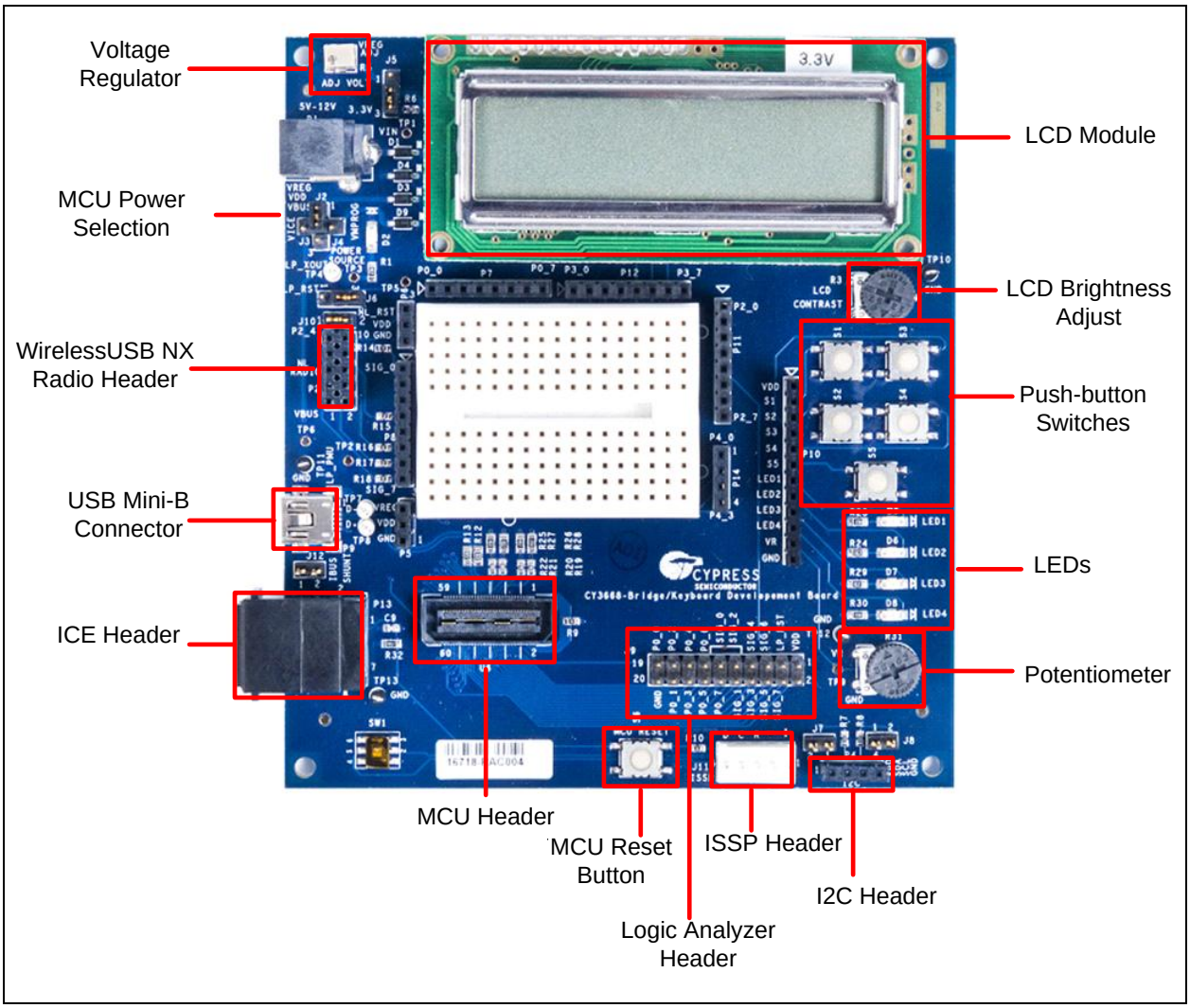
Figure 4-1. System Block Diagram



4.2 Functional Description

Figure 4-2 shows the functional blocks on the WUSB-NX demonstration kit base board.

Figure 4-2. WUSB-NX Demonstration Kit Base Board Functional Blocks

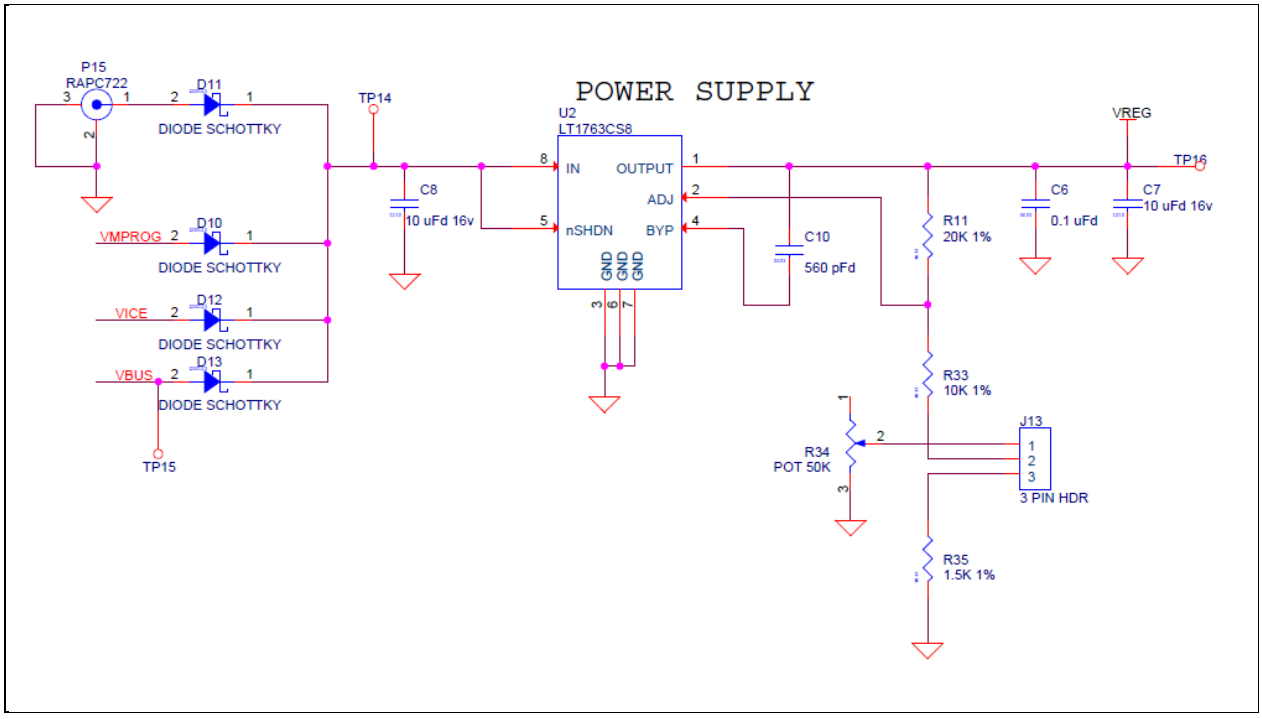


4.2.1 Voltage Regulator

Figure 4-3 shows the kit power supply design. On the kit, the regulator is present under the LCD module (you can remove this module from the kit). There are four input sources to the voltage regulator:

- Power jack Mini 0.08-inch RA PCMT connector
- MiniProg
- PSoC ICE
- USB mini-B connector

Figure 4-3. Voltage Regulator Circuit Diagram



The voltage regulator regulates the inputs and provides a fixed 3.3 V output or an adjustable voltage in the range 3.66 V to 1.63 V, depending on the J5 jumper and R5 (potentiometer) configuration. Table 4-1 gives details about how the different configurations can be achieved.

Table 4-1. Voltage Regulator Configuration

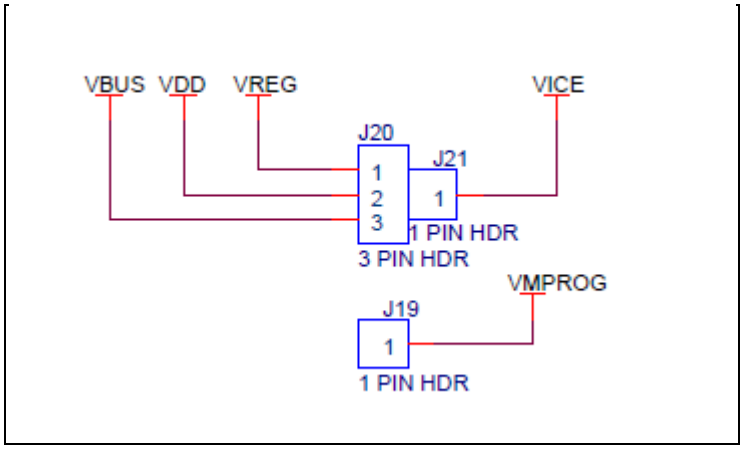
Configuration	Setting
3.3 V fixed	J5: Short pins 2 and 3
Adjustable voltage from 3.66 V to 1.63 V	J5: Short pins 1 and 2 Adjust the R5 potentiometer

Note: When using the adjustable power supply configuration of the kit, use a multimeter at TP3 to monitor the output of the regulator.

Important: Before changing any power supply configuration on the kit, first power off the kit and remove any MCU modules or WirelessUSB NX modules on the kit. When all the power configurations are fixed, remount all the necessary modules to resume operation.

4.2.2 MCU Power Selection

Figure 4-4. MCU Power Selection Circuit Diagram



As shown in Figure 4-4, the MCU module can be powered directly from any of the following voltage sources:

- Voltage regulator
- USB
- PSoC ICE
- MiniProg

Table 4-2 shows the configuration to select the different voltage sources.

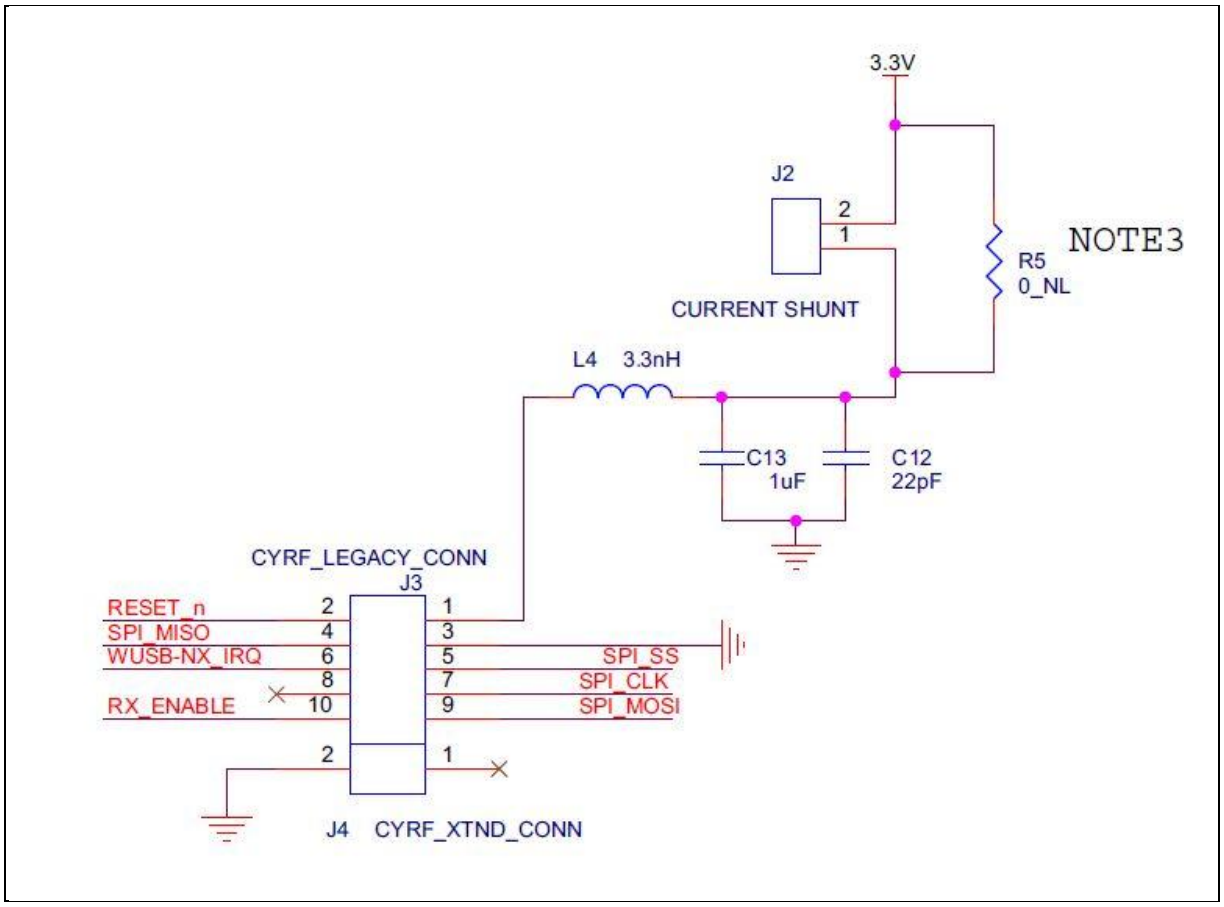
Table 4-2. Voltage Source Configuration

Voltage Source	Jumper Setting
Voltage regulator	J2: Short pins 1 and 2
USB bus	Short J2: Pins 2 and 3 J12: Pins 1 and 2
PSoC ICE	Short J3: Pin 1 J2: Pin 2
MiniProg	Short J4: Pin 1 J2: Pin 2

4.2.3 WirelessUSB NX Radio Header

The WUSB-NX demonstration kit reuses the base board and other hardware from the CY3668 WUSB-NL kit. Therefore, both WUSB-NX and WUSB-NL (Cypress's third-generation, low-power, 2.4 GHz, proprietary radio) use the same radio header on the base board.

Figure 4-5. 10-Pin Header on the Radio Module for WUSB-NX



The following table provides the mapping of the 10-pin header for both WUSB-NL and WUSB-NX modules:

Table 4-3. 10-Pin Header Pin Mapping for WUSB-NX and WUSB-NL

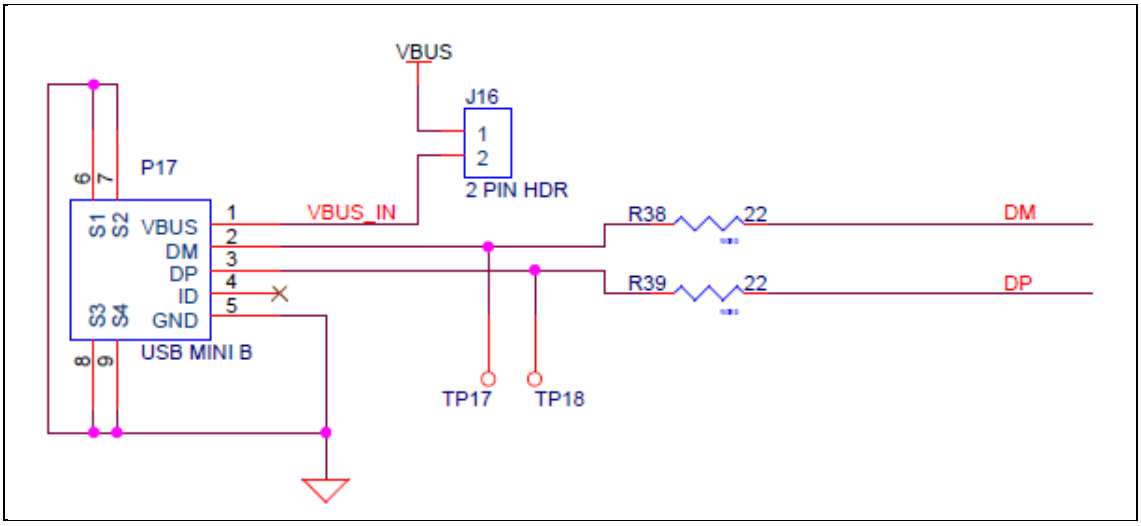
Pin No.	WUSB-NL Pin Name	WUSB-NX Pin Name
1	VDD	VDD
2	TP2	NX_RST
3	GND	GND
4	NL_MISO	NX_MISO
5	NL_Nss	NX_SS
6	NL_PKT	NX_IRQ
7	NL_SCK	NX_SCLK
8	NL_BRCLK	No Connect
9	NL_MOSI	NX_MOSI
10	NL_RST	NX_RX_ENABLE

You can use the jumper J2 to measure current.

4.2.4 USB Mini-B Connector

The kit can be powered using the USB power by shorting the pins 1 and 2 of the jumper J12.

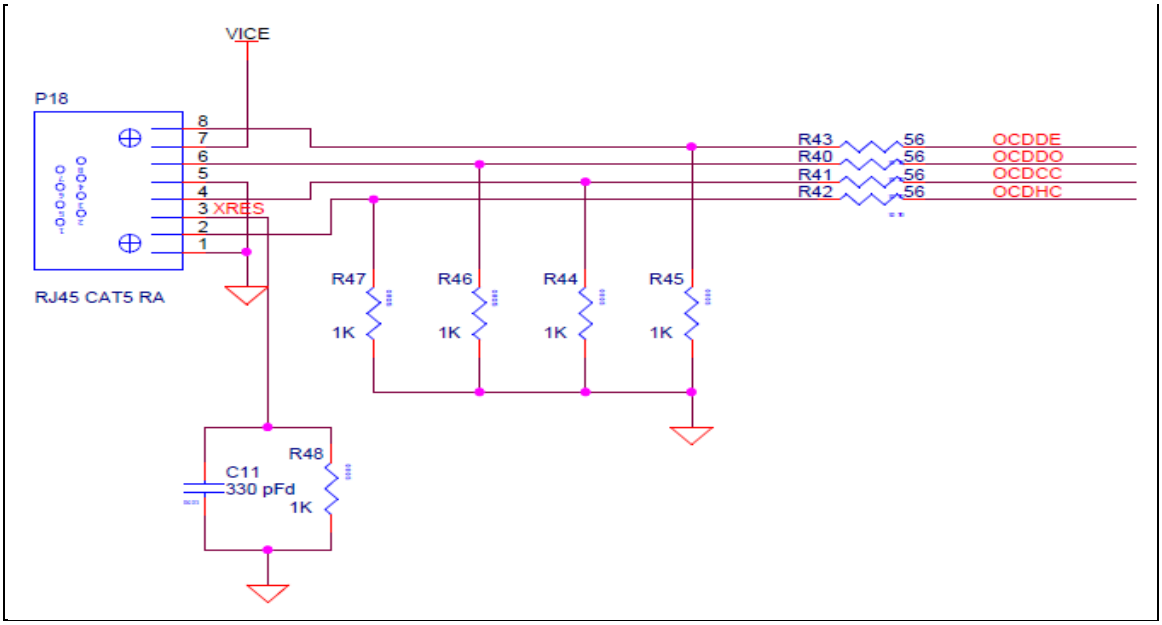
Figure 4-6. USB Connector and Power Jumper Circuit Diagram



4.2.5 ICE Header

The ICE-Cube debugger allows debugging and viewing the contents of specific memory locations. To use the on-chip debugger (OCD) capability of the MCU, connect the ICE-Cube debugger to the RJ45 connector on the base board.

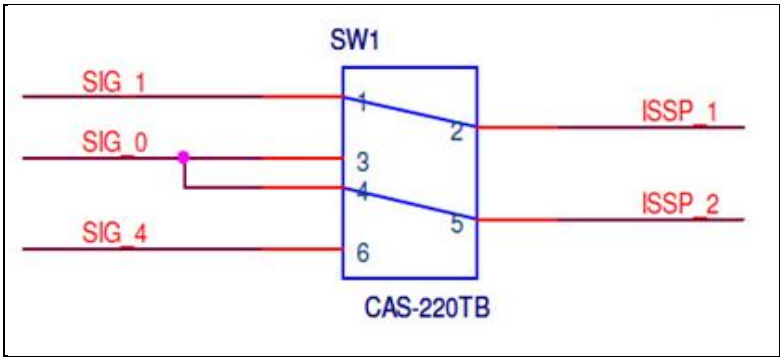
Figure 4-7. ICE Debug Header Circuit Diagram



4.2.6 enCoRe V Programming Selector

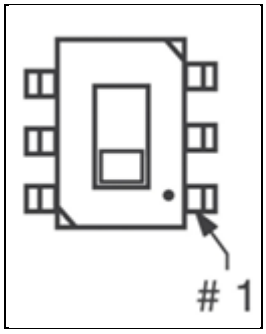
Set the selector switch (SW1), to program enCoRe V modules (see Figure 4-8).

Figure 4-8. MCU Programming Selector Switch Circuit Diagram



To program an enCoRe V module, position the slider of the switch towards pin 1 of the switch as indicated in Figure 4-9. Because enCoRe V is the only MCU supported on this kit, ensure that SW1 is always configured for enCoRe V.

Figure 4-9. Slider Orientation in Relation to Pin 1 for enCoRe V MCU



4.2.7 MCU Header

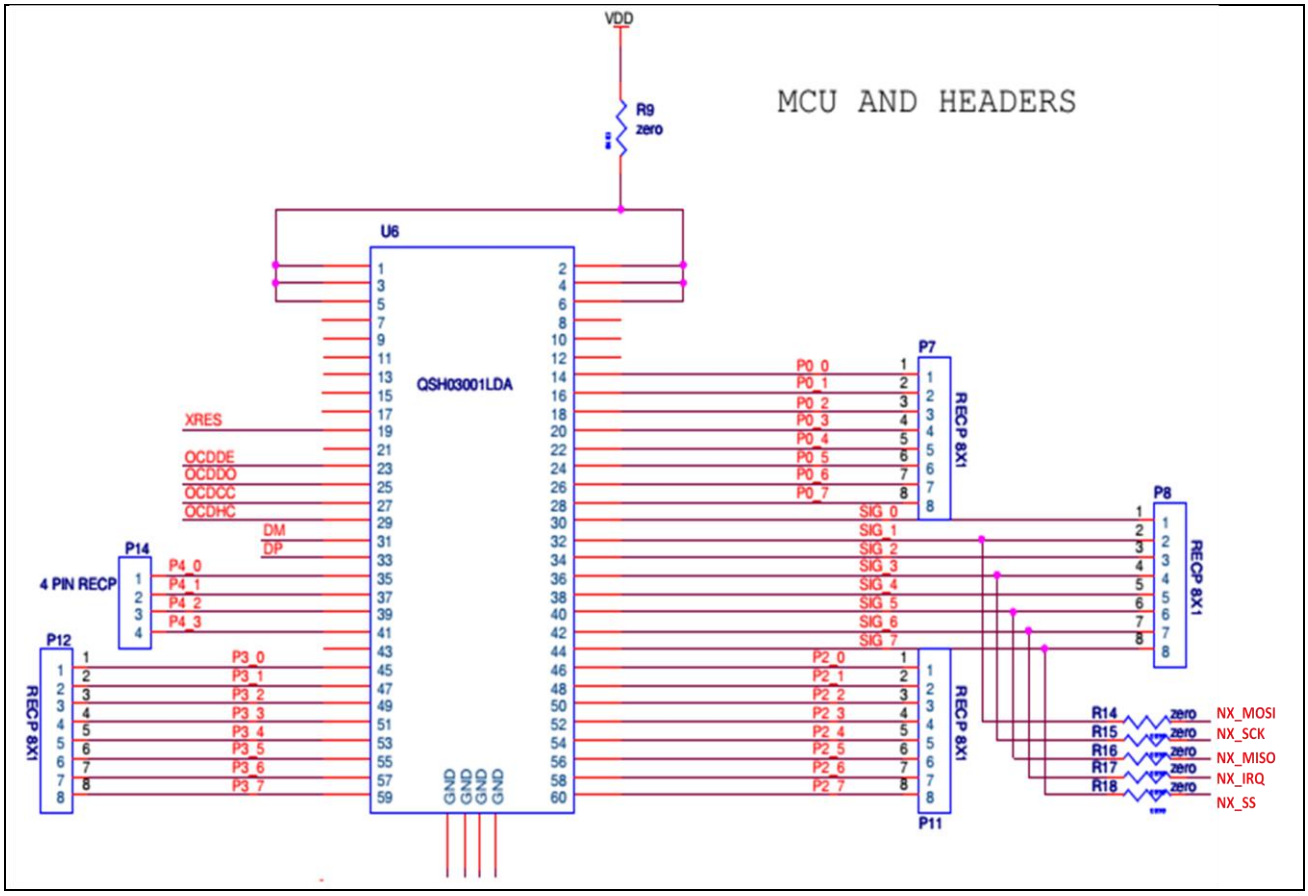
Figure 4-10 shows the MCU header and connections. This kit support Cypress's enCoRe V MCUs via replaceable modules. Cypress's enCoRe V Full-Speed USB peripheral microcontroller provides the following capabilities:

- Up to 32 KB of flash memory
- Three 16-bit timers
- Up to 36 general-purpose I/Os (GPIOs) to accommodate enhanced multimedia features in human interface devices (HIDs)

The in-system reprogrammable enCoRe V and enCoRe V LV MCUs provide design flexibility with a 10-bit ADC, flash memory with EEPROM emulation, and small form-factor packages. The enCoRe V devices also include eight USB endpoints. The enCoRe V LV family operates in the full range of 1.7 V to 3.6 V, with low power consumption for prolonged battery life in wireless applications, especially when paired with Cypress's low-power WirelessUSB NX 2.4-GHz radio. All enCoRe MCUs include Cypress's patented crystal-less oscillator and integrated pull-up resistor to reduce component count and bill of material (BOM) cost.

For more details on the MCUs please refer to the datasheet of enCoRe V and enCoRe V LV.

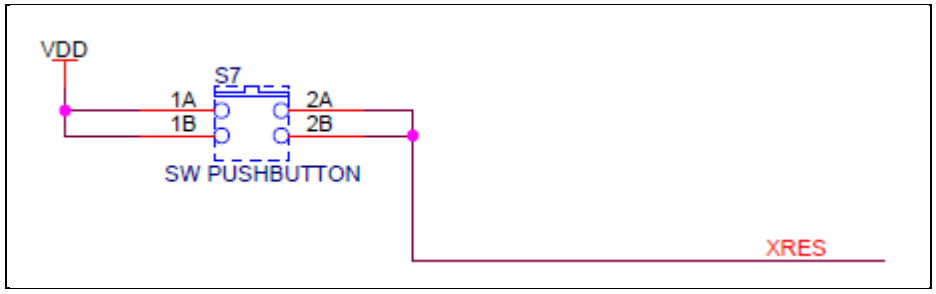
Figure 4-10. MCU Header Circuit Diagram



4.2.8 MCU Reset Button

Use the MCU reset button to reset the MCU (see Figure 4-11).

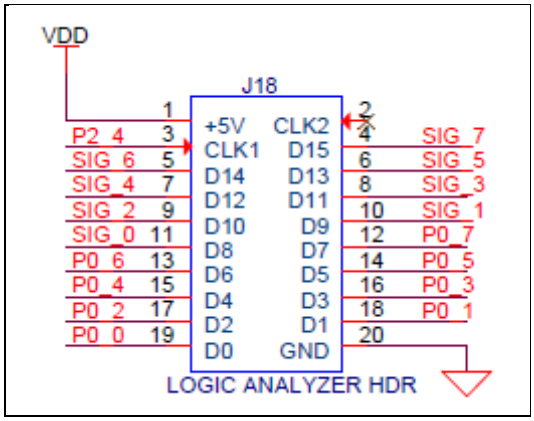
Figure 4-11: MCU Reset Button



4.2.9 Logic Analyzer Header

Use the logic analyzer header to view different signals on an oscilloscope or a logic analyzer. This interface enables hardware level debugging. Figure 4-12 shows the circuit diagram for the header.

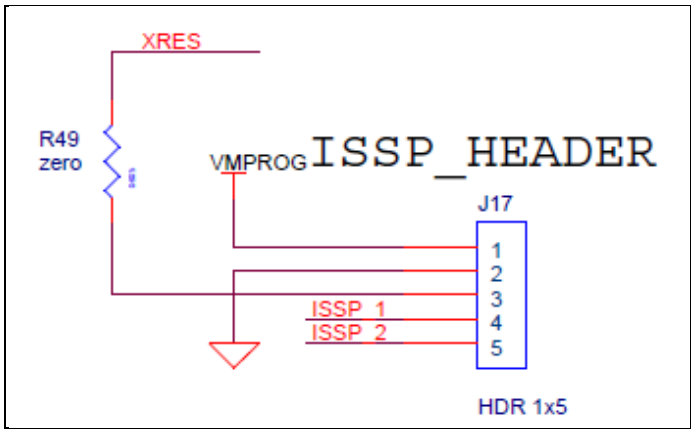
Figure 4-12: Logic Analyzer Header Details



4.2.10 ISSP Header

Use the in-system serial programmer (ISSP) to program the device along with the MiniProg programmer device and USB cable. ISSP/I^{2}C programming is done through the 5-pin connector.

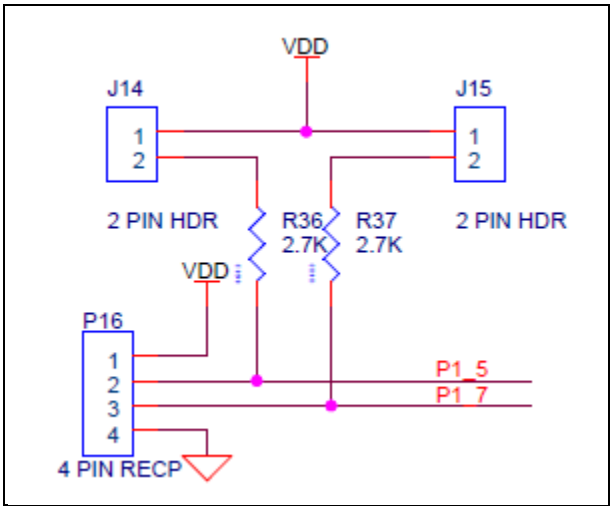
Figure 4-13: Pin Mapping for the ISSP Connector



4.2.11 I²C Header

The WUSB-NX demonstration kit base board is provided with an I²C connector (See Figure 4-14).

Figure 4-14: I²C Header



4.2.12 Potentiometer

Use the potentiometer to vary the voltage to the ADC input. The output of the potentiometer can be accessed from P10, pin 11. See Figure 4-15 for the circuit diagram.

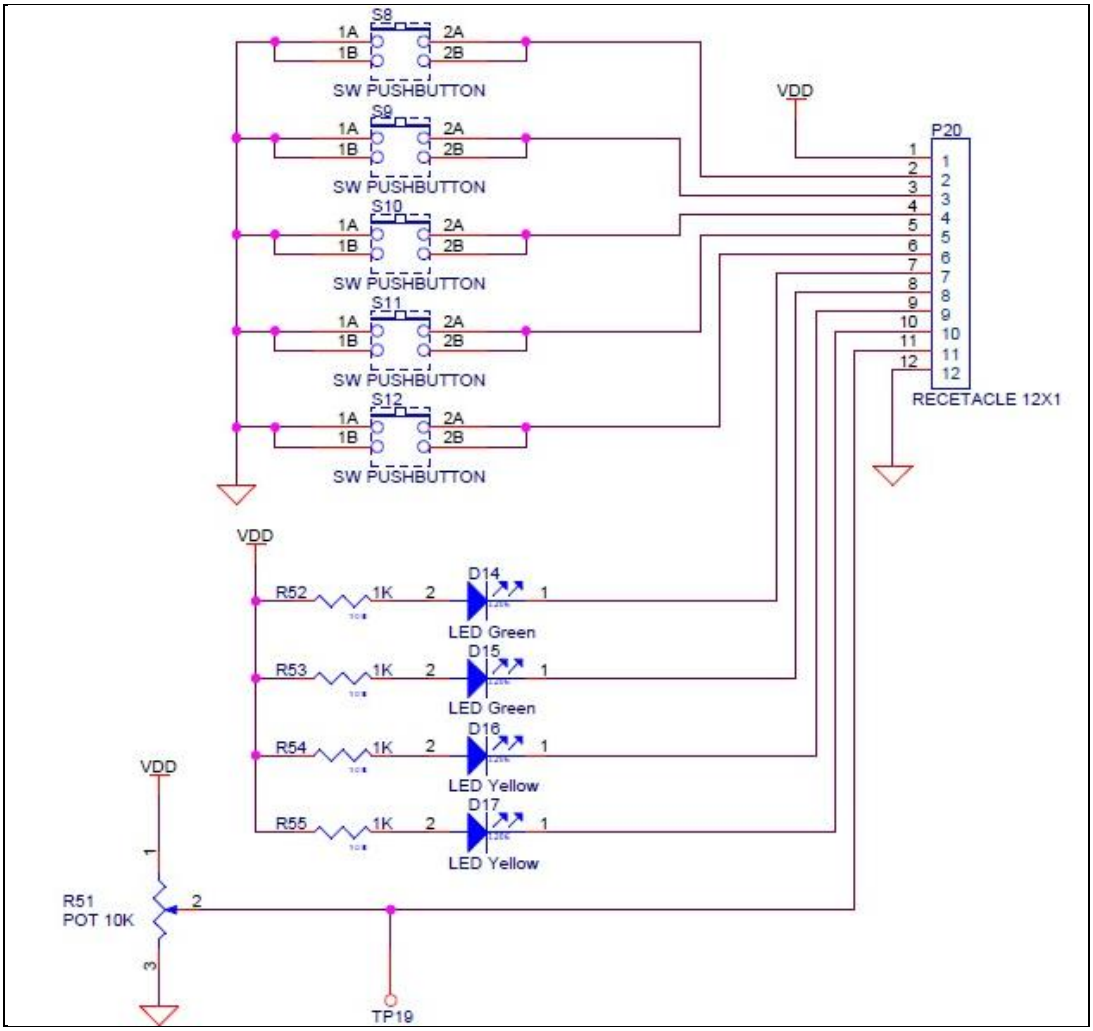
4.2.13 LEDs

The LEDs provide a mechanism to indicate the firmware status. They are exposed via P10, pins 7 through 10. See Figure 4-15 for the circuit diagram.

4.2.14 Push-Button Switches

The switches provide firmware inputs to users. They are exposed via P10 pins 1 through 6. See Figure 4-15 for the circuit diagram.

Figure 4-15: Potentiometer, LEDs, and Switches



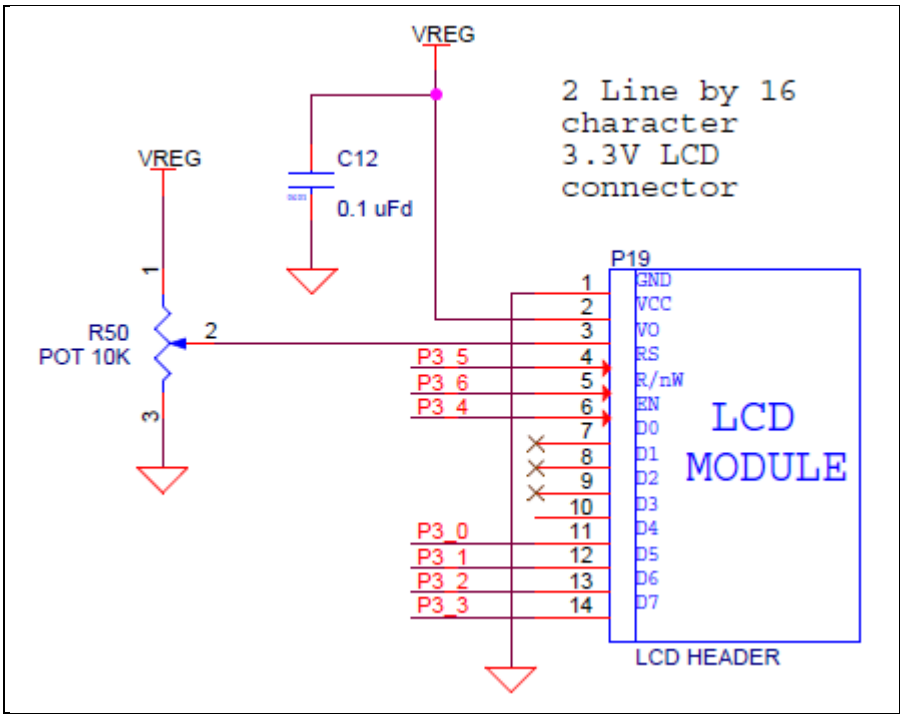
4.2.15 LCD Brightness Adjust

The contrast of the LCD can be varied using the LCD contrast potentiometer. See Figure 4-16 for the circuit diagram.

4.2.16 16×2 LCD Module

The LCD interface connector is used to connect an LCD. It supports a two-line, 16-character, 3.3-V LCD module.

Figure 4-16: LCD Interface and Brightness Control



5. Example Project

The example project provided with the kit demonstrates the basic functionality of WirelessUSB NX radio. The following sections describe the theory of operation and the procedure to exercise the example project and the Windows application provided with the kit. Two WUSB-NX demonstration kit base boards are used, both of which can be configured as either a transmitter or a receiver.

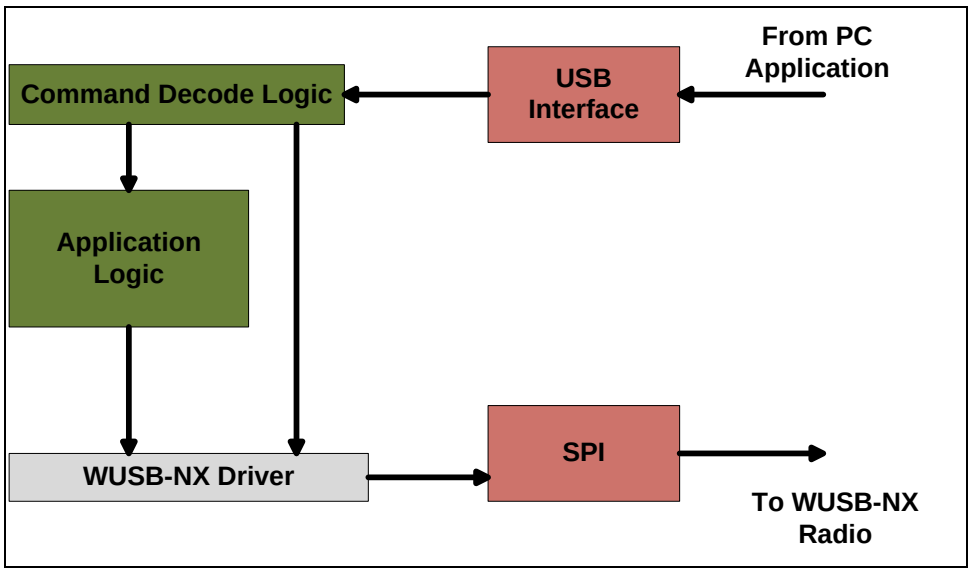
5.1 Theory of Operation

This section explains the design and implementation of the example project. It includes the application block diagram and details about the key functions that implement the application logic. These details are helpful in building customized applications.

5.1.1 Firmware

5.1.1.1 Firmware Block Diagram

Figure 5-1: Different Layers of Firmware



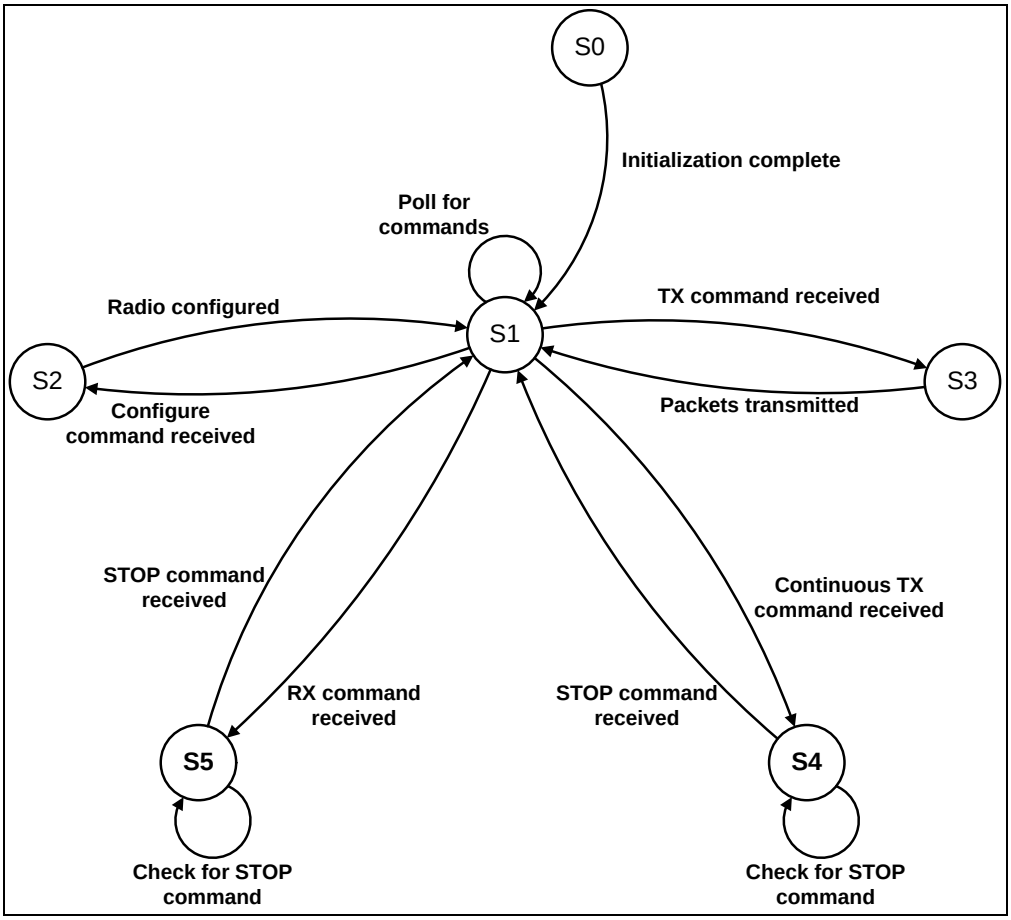
The firmware runs on an enCoRe V MCU, which is interfaced to a WirelessUSB NX radio over SPI protocol. The firmware consists of the following layers apart from the standard PSoC Designer modules such as SPI, USBFS, and LCD:

- WirelessUSB NX Driver: This layer interfaces with the WirelessUSB NX radio and provides APIs for configuring parameters, enabling or disabling features, and controlling transmission and reception.
- Command Decode Logic: This layer interfaces with the USB interface of enCoRe V. It decodes the commands from the PC application (received over USB). Based on the command type, this layer either provides inputs to the application logic or directly interacts with WirelessUSB NX driver layer.
- Application logic: This layer is responsible for the initialization of both the WirelessUSB NX radio and the enCoRe V MCU. It also takes actions such as data transmission and reception based on the inputs from the Command decode logic.

5.1.1.2 MCU States

The following figure shows the details of different states of the MCU firmware:

Figure 5-2: State Transition Diagram of Firmware



- **S0: Initialization.** The device enters this state at power-up. In this state, the firmware initializes the radio and enCoRe V peripherals such as LCD, USB, and GPIOs. During this state, "Enumerating…" is displayed on the LCD.
- **S1: Poll for command.** After initialization, the firmware continuously polls for data over USB. If the firmware jumps from S0 to S1, "Ready for communication" is displayed on the LCD. If firmware jumps from any other state to S0, the LCD panel retains the previously displayed text on the LCD.
- **S2: Radio configuration.** The firmware jumps to this state if any parameter configuration command is received over USB. It jumps back to S1 after the radio configuration is complete.
- **S3: Transmit per count.** The firmware jumps to this state when it gets a command to send some payload for a fixed number of times. It jumps back to S1 after sending the payload for the specified number of times. In this state, the LCD displays "TX Pkt Count" followed by the actual number of packets transmitted.
- **S4: Transmit repeatedly.** The firmware jumps to this state when it gets a command to send payload repeatedly. An incremental count i.e., 0x00 to 0xFF is continuously transmitted in this state. While in S4, "Tx Pkt Count" followed by the actual number of packets transmitted is displayed on the LCD. The firmware jumps to S0 only after getting a 'stop transmit' command over USB.
- **S5: Receive.** The firmware jumps to this state when it gets a command to switch to receive mode. In this state, the LCD displays RX Pkt Count followed by the number of packets received. The firmware jumps to S0 only after getting a 'stop receive' command over USB.

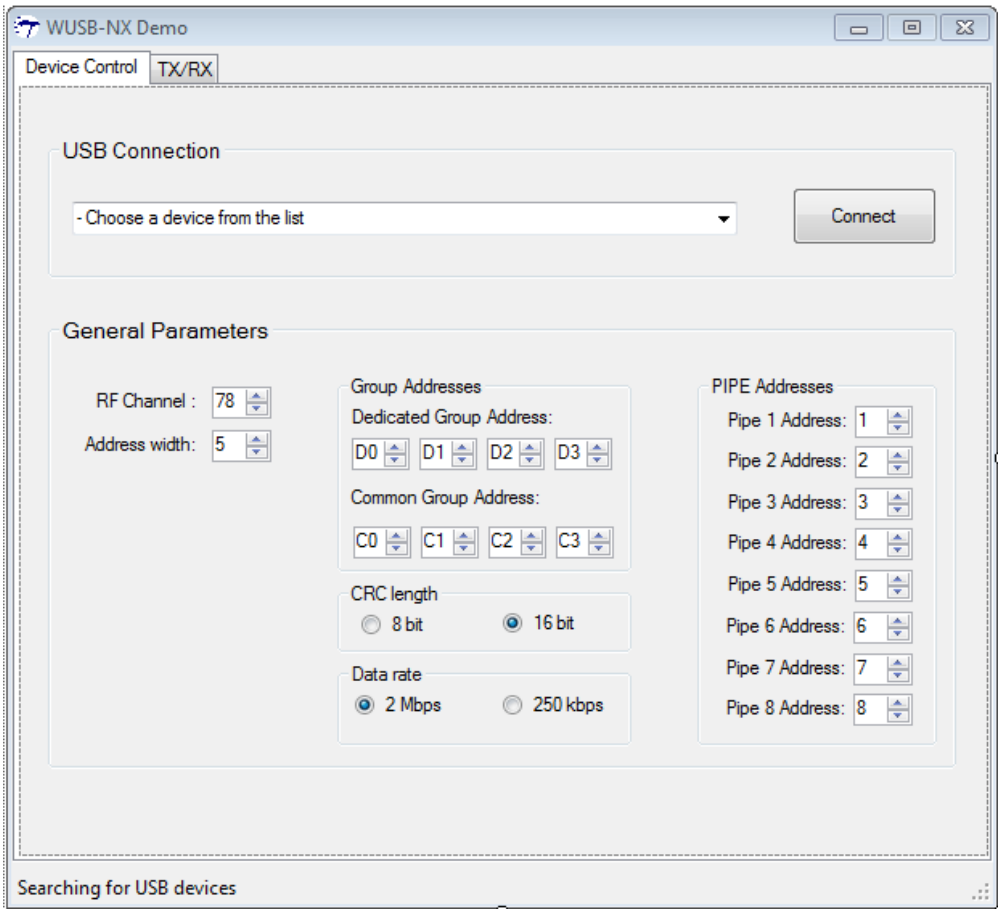
5.1.1.3 Code Details

The WirelessUSB NX drivers, command decode logic and application logic is implemented in the files *hal_nx.c*, *command.c* and *radio.c* respectively. Refer to the function headers in the code for more details on every driver function.

5.1.2 PC Application

The PC application is built using Cypress's own library for USB drivers, *CyUSB.dll*. The application provides a user-friendly interface for configuring the parameters of the WirelessUSB NX radio.

Figure 5-3.Device Control Tab of the PC Application



The PC application has the following sections:

- USB connection section (under the Device Control tab, shown in Figure 5-3) contains a dropdown list of all the WUSB-NX demonstration kits connected to the PC. It also has a Connect /Disconnect button which is used to connect or disconnect the device in the list.

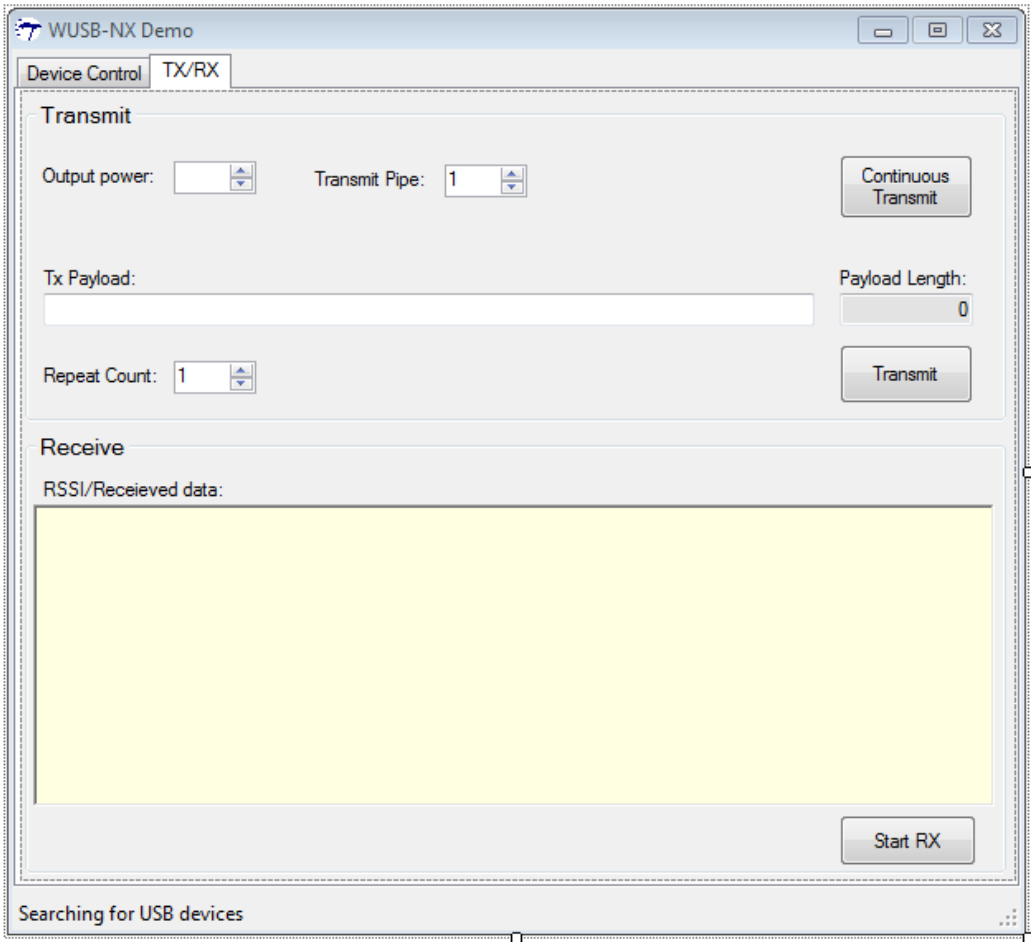
  Note: The string followed by "WUSB-NX Device:" in the dropdown list is extracted from the USB device path provided by the OS. The string is based on the USB port that is used for connecting the kit.

- The General Parameters section (under the Device Control tab) of the interface has configuration parameters common to both the transmitter and the receiver:

  - **RF Channel** sets the frequency which will be used by the WUSB-NX radio. A total of 79 channels (each of 1 MHz) are available. Possible values for this parameter are 0 to 78 mapped to 2.402 GHz to 2.480 GHz in the worldwide ISM band.
  - **Address width** sets the width of the address field in the WUSB-NX radio. Possible values are 3, 4, and 5.
  - **Dedicated Group Address** sets the MSB used in the address field of the WUSB-NX radio packet while communicating over pipe 1. The width of the group address is always one less than the value of the address width.
  - **Common Group Address** sets the MSB used in the address field of the WUSB-NX radio packet while communicating over pipe 2 to pipe 8. The width of the group address is always one less than the value of the address width.
  - **CRC length** sets the length of the CRC field used in the WUSB-NX radio packet.

- **Data rate** sets the on- air data rate of the WUSB-NX radio.
- **Pipe Address** sets the LSB of the address field of the WUSB-NX radio while communicating over the respective pipe.

Figure 5-4: TX/RX Tab of the PC Application



The **Transmit** section (under the **TX/RX** tab, shown in Figure 5-4) has the following fields:

- **Tx Payload.** You can enter a custom payload (up to 32 bytes). If you choose the **Continuous Transmit** mode for communication, you cannot enter a value in this field and any existing content in the field is not used
- **Payload length** is a read-only field which displays the length of the payload entered in the **Tx Payload** field.
- **Output power** field sets the output power of the transmitter. The possible values are 0 dbm, -6 dbm, -12 dbm, and -18 dbm.
- **Transmit Pipe** sets the pipe number that will be used for communication. Possible values are 1 to 8.
- **Repeat Count** sets the number of times the payload in the **Tx Payload** field should be sent.
- **Transmit** button sends a command to the kit instructing it to transmit the payload in the **Tx Payload** field. The number of times the payload is transmitted is decided by the *Repeat Count* field.
- **Continuous Transmit** button, when clicked, puts the kit in continuous mode where it sends a single byte payload repeatedly. The payload data starts with 0x00 and is incremented by 1 over each packet transmission. The payload data cycles back to 0x00 on reaching 0xFF.

The Receive section (under the **TX/RX** tab) has the following fields:

- The **Start RX** button puts the device in the receive mode.

- **RSSI/Received data** displays the data received by the WUSB-NX radio when in the receive mode. It also indicates the RSSI (Received Signal Strength Indicator) value of the packet received and the pipe number on which the packet is received.
- **Status Bar** (common to both the tabs) serves multiple purposes. On the left hand side, it displays the USB connection status. On the right hand side, it shows the Packet Error Rate (**PER**), that is, the ratio of number of packets lost (that is, for which an ACK is not received) to the total number of packets sent, when in TX mode. In RX mode, the left hand side of the bar displays the number of packets being displayed (**RX Packet Count**) in the field **RSSI/Received data**. The **PER** and **RX Packet Count** resets whenever the device is put to the TX and RX mode again respectively.

Implementation details of the PC application are not necessary for the demonstration and are not provided with the kit. Contact Cypress Technical Support or email to wusbnx@cypress.com for help on any issues or any requirement related to the PC application.

5.2 Setting Up and Exercising the Example Project

5.2.1 Setting Up the Hardware

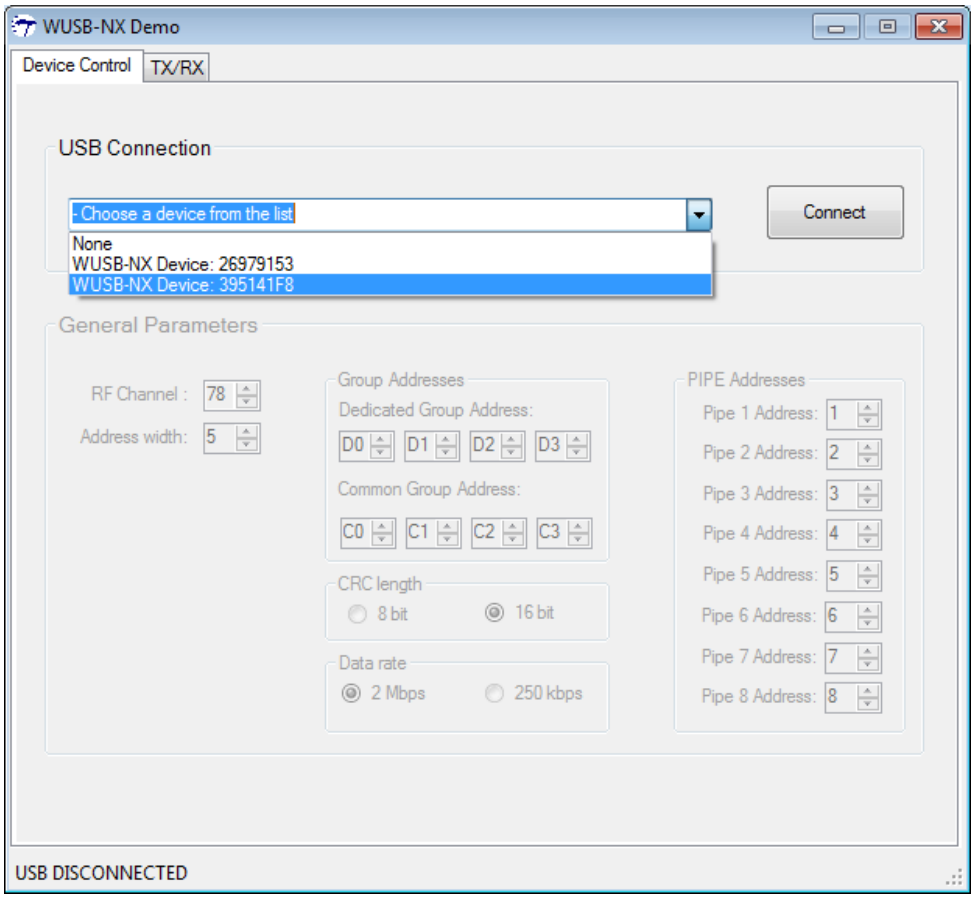
1. Create a local copy of the project named *WUSB_NX_demo* from *<Flash Drive name>\Cypress\Firmware\Data_Transfer_Demo* to your computer. This is to make sure that the project in the flash drive can be used as a backup.
2. Open the WUSB_NX_demo project from your local directory in PSoC Designer. The project file is located at the *<Local Directory>\WUSB_NX_demo\WUSB_NX_demo.app* directory.
3. Make sure that the switch SW1 is positioned away from the LCD panel.
4. Go to **Project** > **Settings** > **Compiler** and select the **Imagecraft** compiler.
5. Select **Build** > **Generate Configuration Files for ' WUSB_NX_demo' Project**.
6. Build the project. Select **Build** > **Build 'WUSB_NX_demo' Project** or press [**F6**]. After the build process is **successful, the output hex file is** located at *<Local Directory>\WUSB_NX_demo\WUSB_NX_demo\output\WUSB_NX_demo.hex.*
7. Download the built firmware (hex file) on the demonstration kit base board by following the Programming Steps given in Section 3.2 Programming and Emulation Process on page 9.

   Note: Make sure to remove the power source (12 V power adaptor or USB cable) from the demonstration kit base board before proceeding with the remaining steps.
8. Attach the WirelessUSB NX Radio module to P2 and LCD module to P3. Make sure that the jumper settings are the same as mentioned in above section.
9. Connect the mini-B side of the USB Standard A/Mini-B cable to the demonstration kit base board.
10. Connect the Standard A side of the USB Standard A/Mini-B cable to the PC.
11. The WUSB-NX demonstration kit base board should enumerate on the PC as a HID compliant device and the LCD should display "**Ready for communication**". If the observations are different refer to Section Troubleshooting on page 32.

5.2.2 Working With the PC Application

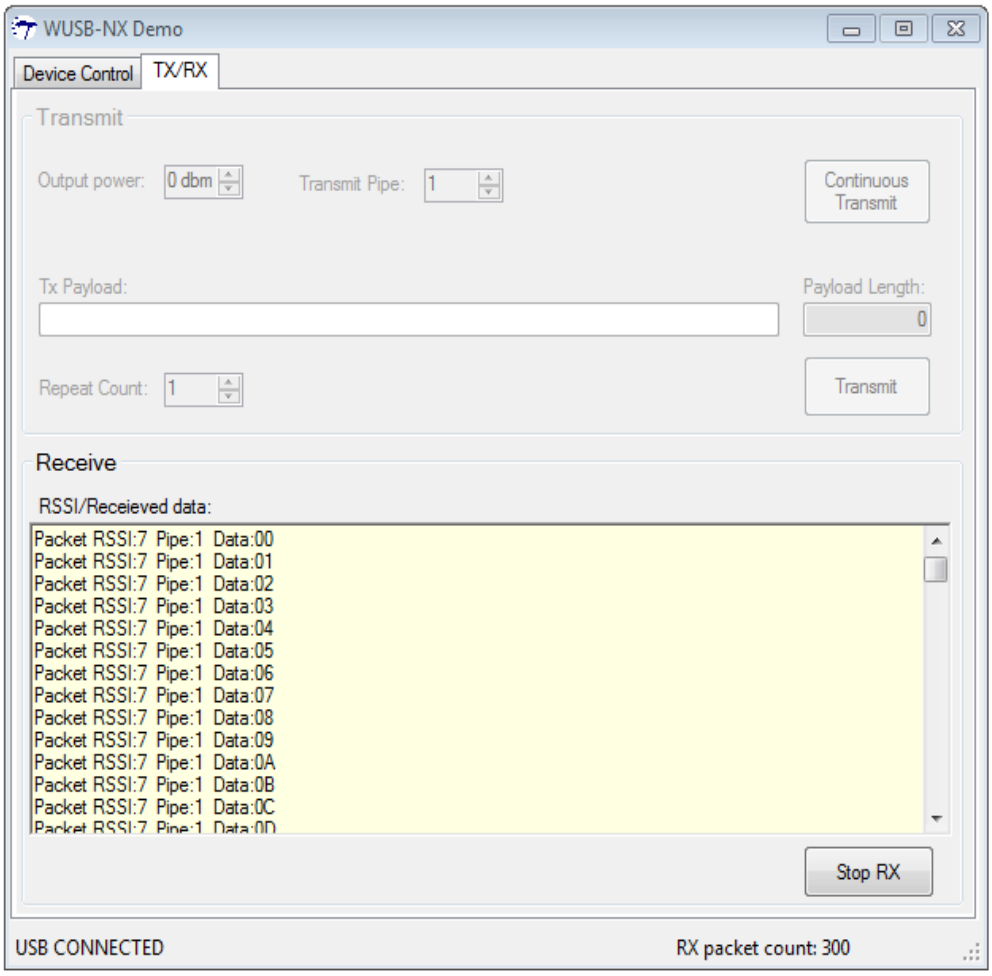
1. To launch the PC application, run *WUSB_NX_Demo.exe* from the following location: *<Drive name>\Cypress\Application*. You can either run one instance of the application on different PCs or two instances on the same PC.
2. Select the desired device from the **USB Connection** box (Figure 5-5) and click **Connect**. If you are running two instances of the application on the same PC, make sure that one device (demonstration kit base board) is bound to only one application.

Figure 5-5. Device Selection in the PC Application



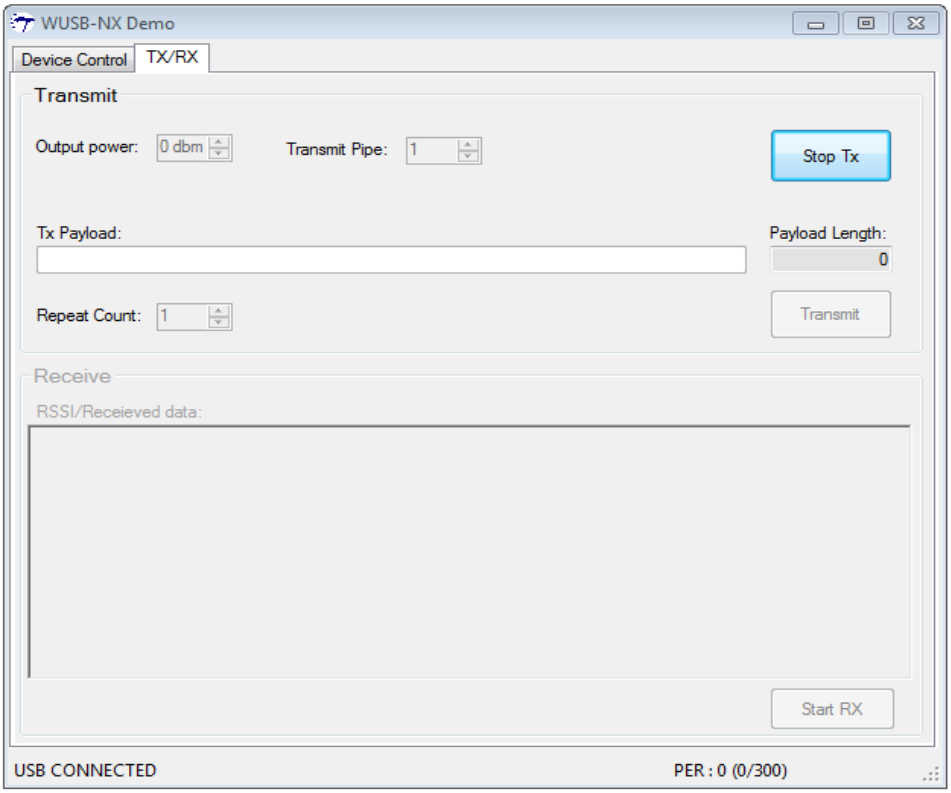
3. Configure the required parameters from the **General** Parameters section.
4. Switch to **TX/RX** tab of the user interface.
5. Click **Start RX** to put one device in the RX mode (Figure 5-6). The status bar at the bottom right starts showing the RX packet count.

Figure 5-6: RX Operation in the PC Application



6. For the other device, configure the required parameters in the Transmit section of the **TX/RX** tab.
7. Start transmitting by clicking either "**Transmit**" button or the "**Continuous Transmit**" button. If Continuous Transmit is used, the status bar at the bottom right should show the PER (Packet Error Rate) value (Figure 5-7).

Figure 5-7: TX Operation in the PC Application



8. When the receiver receives any packet, it is displayed in the **Receive** section. The number of packets successfully received is displayed on the status bar at the bottom.
9. To change any other configuration parameter on the device, click **Stop TX** (for transmitter) and **Stop RX** (for receiver) first, and then make the changes.

5.2.3 Modifying the Project for Custom Application

You can use the example project in the <*Drive name*>\ *Cypress\Firmware\Data_Transfer_Demo\WUSB_NX_demo* with the PC application only. If you want to make a custom project with WUSB-NX radio, follow the steps given below:

1. Create a local duplicate copy of the project on your computer. Make sure that all the changes are made in your local copy so that the project in the flash drive can be used as a backup copy.
2. In the *main.h* file, change the macro “CUSTOM_MAIN” to 1.
3. Write your custom code in *main.c* under the section provided.
4. If you do not want to reuse functions from the files *port.c*, *usb.c*, *radio.c,* and *command.c*, exclude them from the project. This can be done by right clicking on file name in workspace explorer and selecting “Exclude from Project”.
5. Make sure that the files *hal_nx_hw.c* and *hal_nx.c* are not excluded as they contain the driver functions for the WUSB-NX radio.

6. Troubleshooting

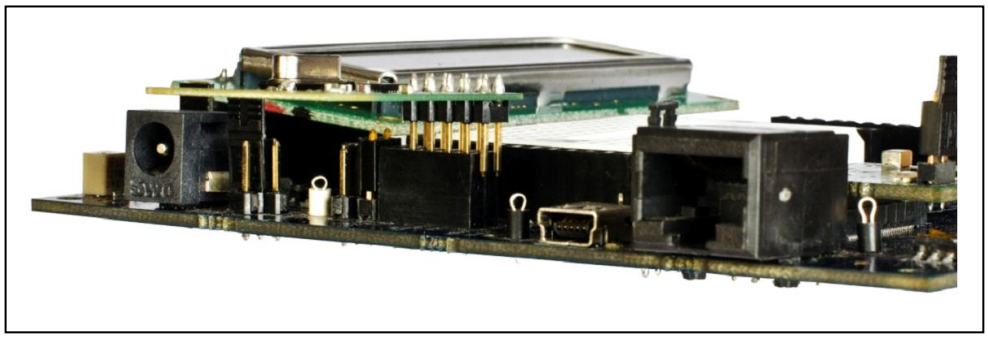
- **Issue:** Unable to download the firmware to the enCoRe V module after plugging it on the development board.

  Possible Solution: The switch SW1 determines the MCU type to be used on a given development board. This demo uses the enCoRe V module. Ensure that the switch is pushed away from the LCD panel.

- **Issue:** The firmware does not come up after plugging in the WUSB-NX module.

  Possible Solution: Make sure all 10 pins of the WUSB-NX module are plugged on to the 10 pins of the socket located on the development board and are properly aligned.

Figure 6-1: WirelessUSB NX Module Plugged-in Incorrectly



- **Issue:** Text displayed on the LCD display area is not clear or is not visible.

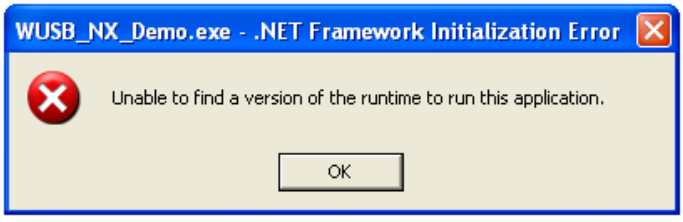
  Possible Solution: The rotary knob located below the LCD controls the contrast of the LCD display (Figure 4-2 for knob). Turn the knob in the clockwise direction to increase the contrast of the LCD screen. Set the contrast to a level of your choice.

- **Issue:** The development board is not powered using the USB cable.

  Possible Solution: The position of jumpers at J2, J3, and J4 determine the power source for the development board. To power it using the USB cable, short pins 2 and 3 of J2 and pins 1 and 2 of J12. See MCU Power Selection in Section 4.2.2 for details.

- **Issue:** The PC application throws an error (as shown in Figure 6-2) and does not start.

Figure 6-2. Error Due to .NET Framework Version Mismatch



  Possible Solution: The system doesn't have the correct version of Microsoft .NET Framework. To resolve the issue, download and install.NET framework 2.0 from the following link: http://www.microsoft.com/en-in/download/details.aspx?id=1639

Revision History

Document Title: WirelessUSB™ NX Demonstration Kit Guide Document Number: 001-90000			
Revision	**Issue Date**	**Origin of Change**	**Description of Change**
**	03/26/2014	ANKC	New Document
*A	04/16/2014	ANKC	Updated Figure 4.5. Updated Table 4.3.