Getting Started with ZigBee using the CC2530ZNP Mini Kit

Morten Braathen & Suyash Jain
Version 1.01
September 2011

Important Notice

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, patent infringement, and limitation of liability.

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

Customers are responsible for their applications using TI components.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty or endorsement thereof.

Revisions

Version 1.0	August 2011	Original release
Version1.01	September2011	Lab errata

Mailing Address

Texas Instruments
Technical Training Organization
7839 Churchill Way, M/S 3984
Dallas, Texas 75251-1903

Texas Instruments Norway AS
Low Power RF
Gaustadalleen 21
N-0349 Oslo, Norway

Table of Contents

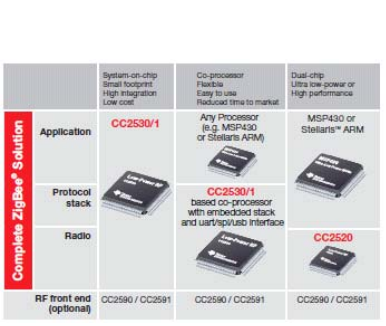
Abstract

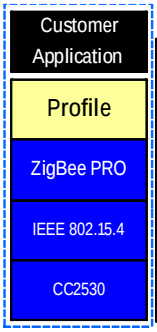
Abstract

This 3-hour workshop will introduce you to ZigBee and how to build a ZigBee Application by understanding the design process for a ZigBee Network Processor. You will come away from this workshop understanding how to set up a ZigBee Mesh Network using ZigBee coordinators, routers, and end devices. You will run Packet Sniffers and then observe the Personal Area Network (PAN) traffic over the network. Other features to be learned are Mesh Routing, Network Commissioning, PAN Formation.

Overview

ZigBee™ Overview





Application
ZigBee™ Stack
- Network functionality

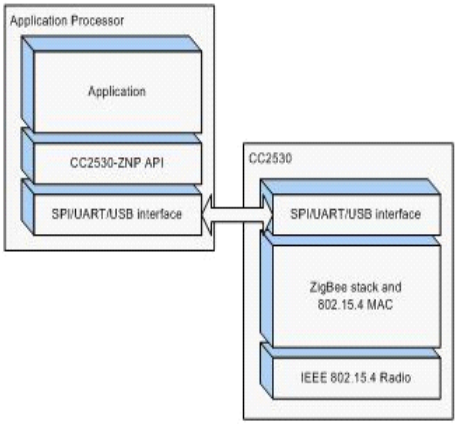
IEEE 802.15.4
- Media Access Layer
  - MAC
- Physical Layer
  - PHY
- Standardized point to point link

- Self Healing (Mesh Networks)
  - Routing Protocol
  - Redundancy
- Targeting Long Battery Life
- Basic Idea - One Channel, One Network
- End User (Customer) Responsible for Commissioning / Deployment Strategy
- Large Network Support - 100+
- Profiles
  - Public Profile (Interoperability)
  - Manufacture Specific Profile (MSP)
- Standardized Protocol
- MSP – Private Profile
  - Only for CC2530ZNP - "Out of the Box"

ZNP

ZigBee Network Processor - ZNP

- The ZigBee Network Processor (ZNP) is a CC2530/1 running the ZigBee Pro stack with API to communicate to the system's main processor through the Serial interface (SPI/UART)
- Partitioned Architecture – Two Chip Solution
- ZIgBee Add-On - Use Existing Application Processor
- CC2530 ZNP is not a "Pre-Programmed" device



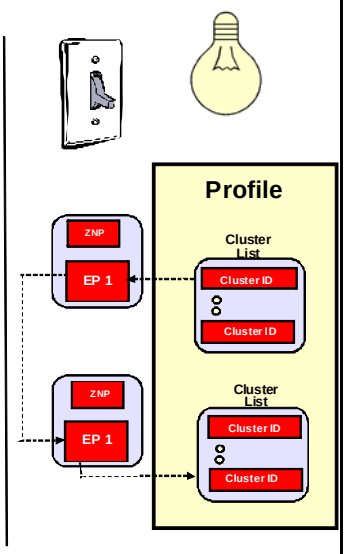
ZNP Based System Architecture

App Development

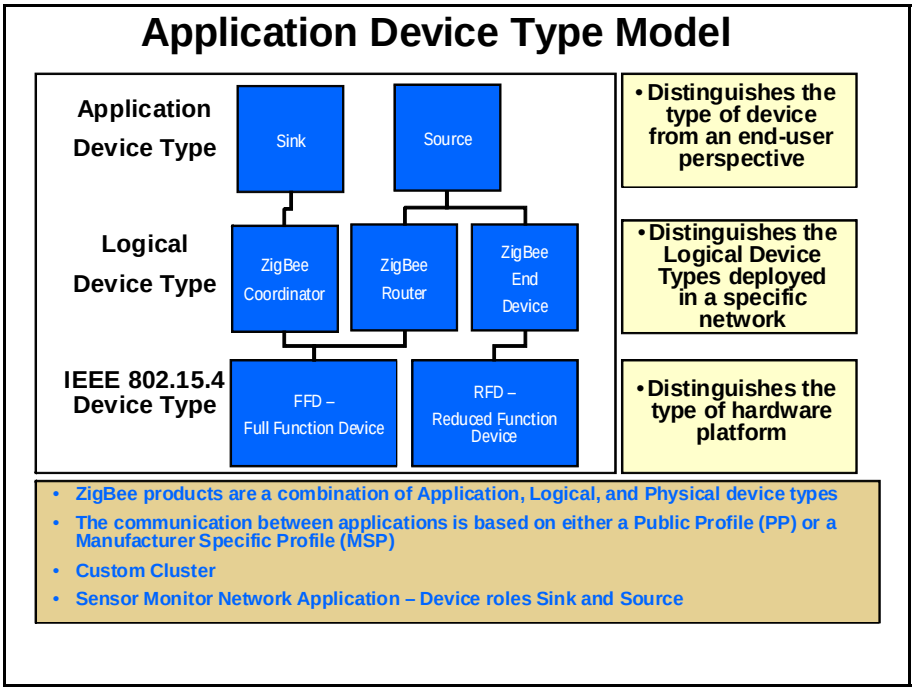
ZigBee Application Development - Things to do

A new application concept with no defined ZigBee Profile

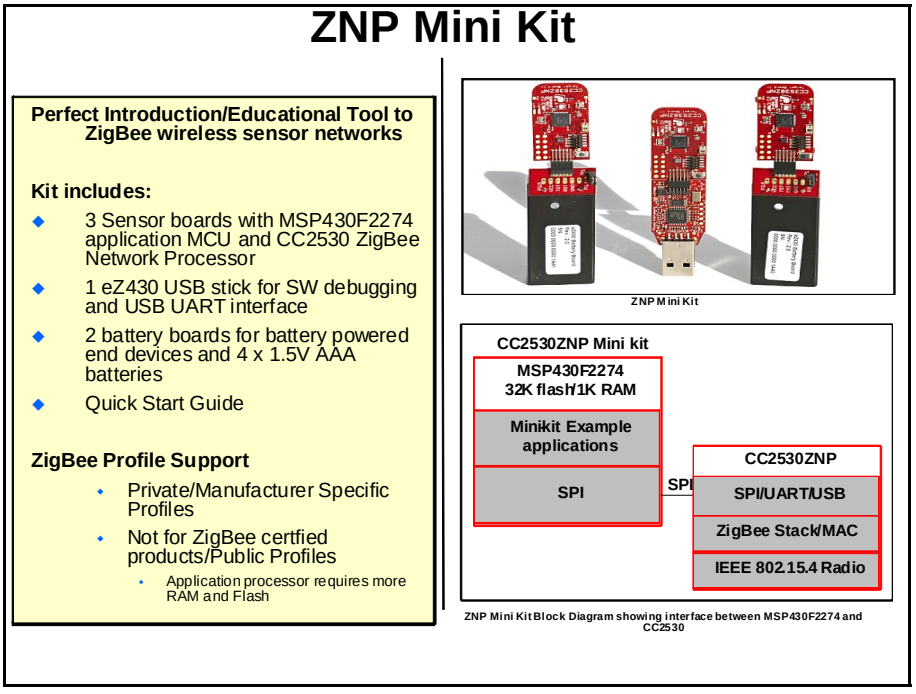
- Define the Profile
- Define devices in the system
- Map to Logical Devices
- Define the Device Descriptions within each Profile
- Define the descriptors, deploy the applications over endpoints
- Define the Clusters and indicate which are input and output from each Device Description
- Define the Security solution
- Define the Commissioning Process
- Package the solution
- Integrate and test the solution
- Solve Deployment issues



App Device Type Model



ZNP Mini Kit



TI Examples

Examples From TI

- TI provides extensive examples for developers to understand basic ZNP application development and start with custom development
- Download the examples from TI web page for ZNP Mini Kit
- Wiki Pages for ZNP Mini kit provide detailed information about ZNP Mini kit and application examples

Screen Shot of Wiki Page for ZNP mini Kit

Pin Configurations

Three Pin Configurations

- As explained in the ZNP interface specification documentation, ZNP builds from Texas instruments use three types of pin configuration
  - Main Pin Configuration – SPI Mode
  - Alternate Pin Configuration – UART Mode
  - ZNP Mini Kit Pin Configuration – SPI Mode
- **CFG0 and CFG1 pin on CC2530**
- **CC2590/CC2591**
  - To enable the CC2590 build the ZNP image with compile option HAL_LNA_PA_CC2590
  - To enable the CC2591 build the ZNP image with compile option HAL_LNA_PA

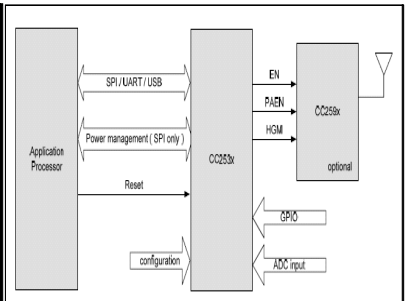


Figure 1 CC2530 Interface

ZNP Interface

Application Processor ZNP Interface

- CC2530ZNP has two sets of interfaces
  - Reduced Interface API (Simple Interface)
    - Fewer Commands (and Features)
    - Supports Single End-Point
  - Full – ZigBee API (AF-ZDO Interface)
    - More commands (and Features)
    - Supports Multiple Endpoints
- Refer to ZNP Interface Specification Documentation included with Z-Stack Install for complete details.

ZNP Interface – Frame Format

- **ZNP Command Types**
  - AREQ – Asynchronous Request
    - Callback Events
    - Function call with return value
  - POLL – Polling for Data
    - Retrieve Queued Data
  - SREQ – Synchronous Request
    - Immediate Response
  - SRSP – Synchronous Response
    - Response to a SREQ command
- **ZNP Subsystem**

? The Subsystem of the Command is described by bits 0-4 of CMD0

Bytes: 1	2	0-250
Length	Command	Data

Fig: General Frame Format

Cmd0		Cmd1
Bits: 7-5	4-0	7-0
Type	Subsystem	ID

Fig: Command Field

Type	CMD0 Value
POLL	0x00
SREQ	0x20
AREQ	0x40
SRSP	0x60

Subsystem Value	Subsystem Name
0	RPC Error interface
1	SYS interface
2	Reserved
3	Reserved
4	AF interface
5	ZDO interface
6	Simple API interface
7	UTIL interface
8-32	Reserved

ZNP Glossary

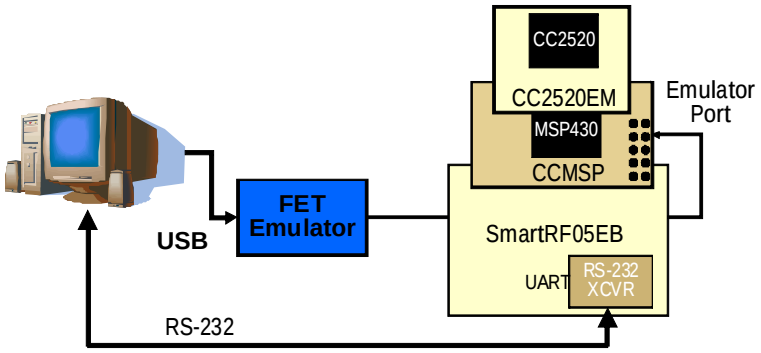
ZigBee® Glossary

- A Application Profile is an agreement on a series of messages defining an application space
  - Public Application Profile
    - Interoperable application profile developed by the ZigBee Alliance
  - Manufacturer Specific Profile
    - Private application profile developed by a company to operate a ZigBee Device
    - Limits application interoperability to devices that share this profile
- Profile ID a unique identifier assigned by the ZigBee Alliance
- A Cluster is a message identifier for exchange of information within an application profile
- Endpoint a communication entity within a device which permits support for a specific application through the exchange of Clusters
- Binding/s is a way of connecting devices between two endpoints
  - Each binding supporting a specific application profile
  - Each message type is represented by a cluster within the profile
- Commissioning the task of configuring devices and networks to achieve the needs of a specific installation
- Application Object is software at an endpoint that controls the ZigBee Device
  - A single ZigBee Device node supports up to 240 application objects
  - Each application objects supports endpoints numbered between 1 and 240
  - Endpoint 0 reserved for the ZigBee Device Object (ZDO)
- ZigBee Device Object (ZDO) defines the logical role of a device within the network
  - Device type: Coordinator, Router, End device
  - Initiates and/or responds to binding and discovery
  - CC2530ZNP is running the ZDO endpoint
- PAN ID – a unique Personal Area Network Identifier; Controlled by User and Application development
- Device Description is a description of a specific device within a profile
- Device Discovery can find the identity of devices on active channels within the PAN
- In a Mesh network the routing of messages is performed as a decentralized, cooperative process involving many peer devices routing on each others' behalf
- Service Discovery is the ability to determine supported services on given devices within the PAN (e.g. USB Enumeration)

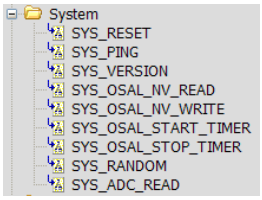
Z-Tool

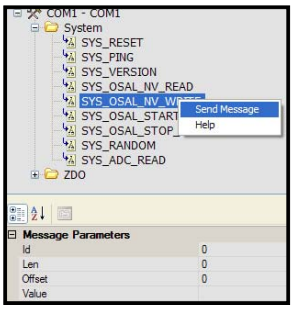
Z-Tool

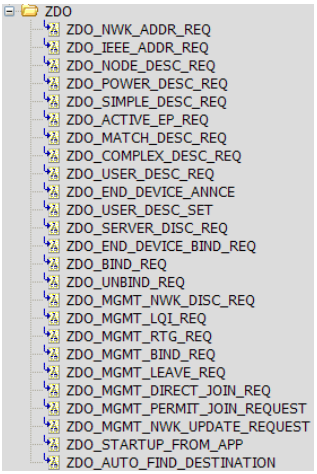
- A PC application for sending and receiving commands to/from programs on the target board using the RS-232 port (transceiver required)
- Requires MT_TASK Compiler option



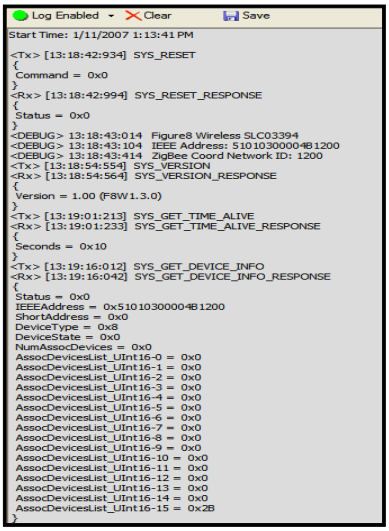
Sending Commands from Z-Tool







Receiving Messages in Z-Tool



- Z-Tool automatically formats and displays incoming and outgoing messages
- Z-stack is instrumented to automatically provide system status at startup (when enabled)

Joining a ZigBee Network

JOINING A ZIGBEE NETWORK

Ln.	Timestamp	Time Delta	Ch.	Stack	Packet Information	PAN Src.	PAN Dst.	MAC Src.	MAC Dst.
10	18:26:18.065806	00:00:11.303384	11	ZigBee	MAC: Beacon Request		0xFFFF		0xFFFF
28	18:26:18.069478	00:00:00.003672	11	ZigBee	Beacon	0xDB73		0x0000	
21	18:26:18.577766	00:00:00.508288	11	ZigBee	MAC: Association Request	0xFFFF	0xDB73	0x00124B00011F73C1	0x0000
5	18:26:18.578822	00:00:00.001056	11	ZigBee	Acknowledgment				
18	18:26:19.072454	00:00:00.493632	11	ZigBee	MAC: Data Request		0xDB73	0x00124B00011F73C1	0x0000
5	18:26:19.073414	00:00:00.000960	11	ZigBee	Acknowledgment				
27	18:26:19.076070	00:00:00.002656	11	ZigBee	MAC: Association Response		0xDB73	0x00124B00011F74E6	0x00124B000
5	18:26:19.077318	00:00:00.001248	11	ZigBee	Acknowledgment				
39	18:26:19.086926	00:00:00.009608	11	ZigBee	ZDP: Device_annce		0xDB73	0x91FF	0x0000
5	18:26:19.088558	00:00:00.001632	11	ZigBee	Acknowledgment				
39	18:26:19.095134	00:00:00.006576	11	ZigBee	ZDP: Device_annce		0xDB73	0x0000	0xFFFF

Active Scan to search for existing Networks

Beacon Response from Coordinator or Router in the Vicinity

Association Request to Join the network

Data Request Polling for Association Response

Association Response from the Coordinator or Router

Device Announces its address on the network to check if any other device has similar address.

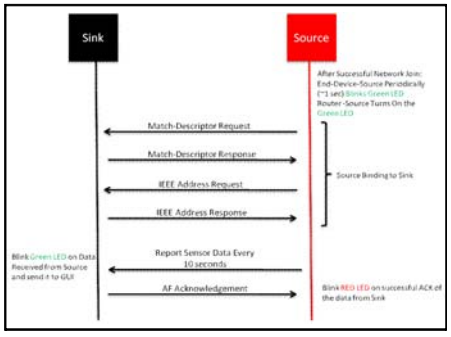
If yes the parent of joining node assigns new address and process is repeated until a unique address is assigned to the device

Binding

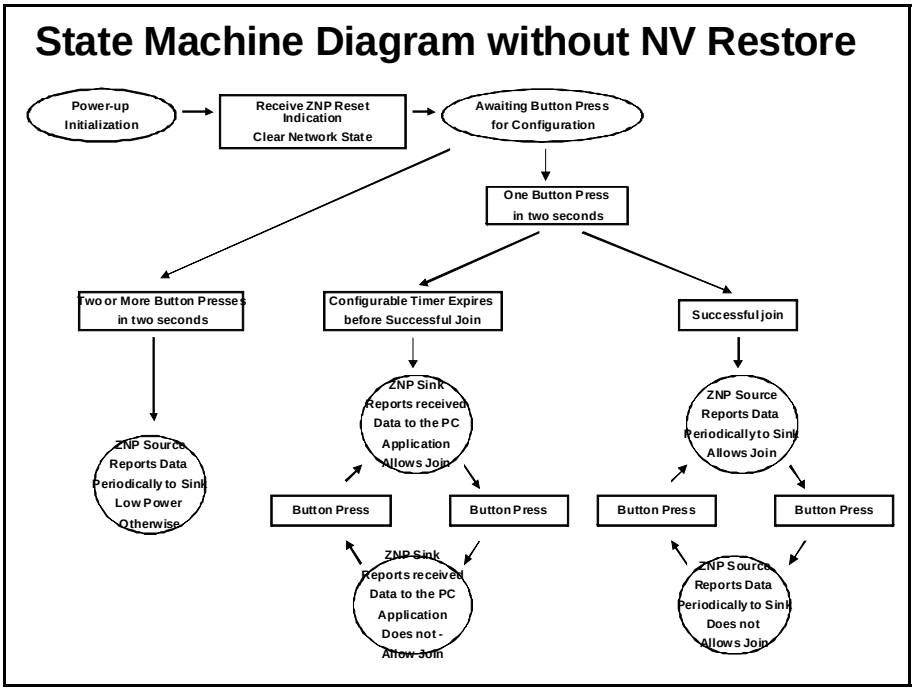
BINDING – SENSOR MONITOR NETWORK SAMPLE APP

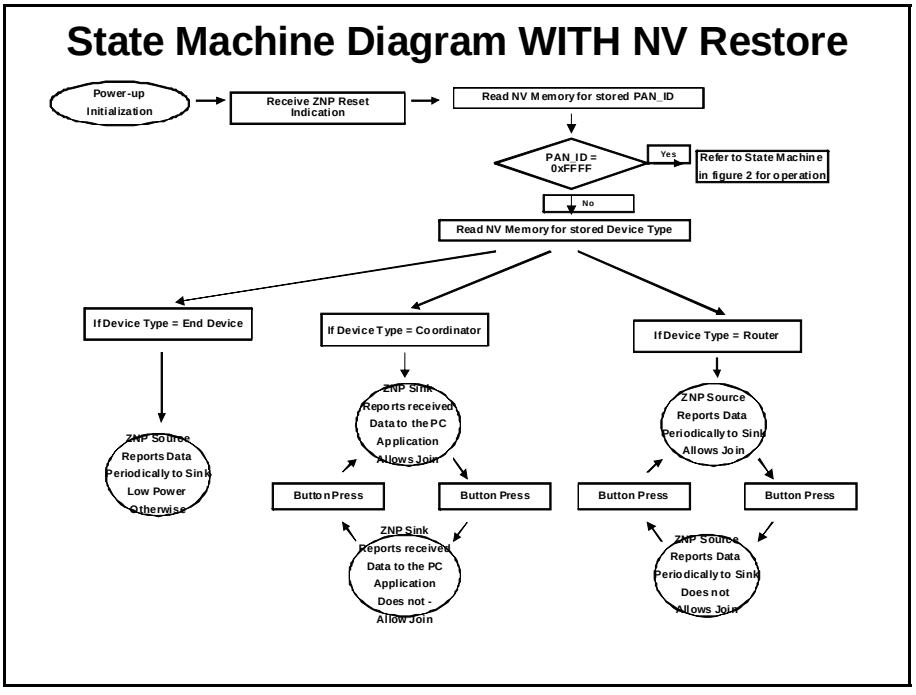
36	17:58:44.218625	00:00:00.010776	11	ZigBee	ZDP: Match_Desc_req		0x2AB0	0x077F
36	17:58:44.225577	00:00:00.006952	11	ZigBee	ZDP: Match_Desc_req		0x2AB0	0x0000
33	17:58:44.234777	00:00:00.009200	11	ZigBee	ZDP: Match_Desc_rsp		0x2AB0	0x0000
5	17:58:44.236217	00:00:00.001440	11	ZigBee	Acknowledgment			
27	17:58:44.243369	00:00:00.007152	11	ZigBee	APS Acknowledgment		0x2AB0	0x077F
5	17:58:44.244617	00:00:00.001248	11	ZigBee	Acknowledgment			
32	17:58:44.261297	00:00:00.016680	11	ZigBee	ZDP: IEEE_addr_req		0x2AB0	0x077F
5	17:58:44.262705	00:00:00.001408	11	ZigBee	Acknowledgment			
31	17:58:44.266345	00:00:00.003640	11	ZigBee	Private: 0x0001		0x2AB0	0x077F
5	17:58:44.267721	00:00:00.001376	11	ZigBee	Acknowledgment			
39	17:58:44.276865	00:00:00.009144	11	ZigBee	ZDP: IEEE_addr_rsp		0x2AB0	0x0000

View network operation and over the air messages (after network join operation) illustrating binding and then Source devices periodically reporting the data and receiving acknowledgement from the Sink device



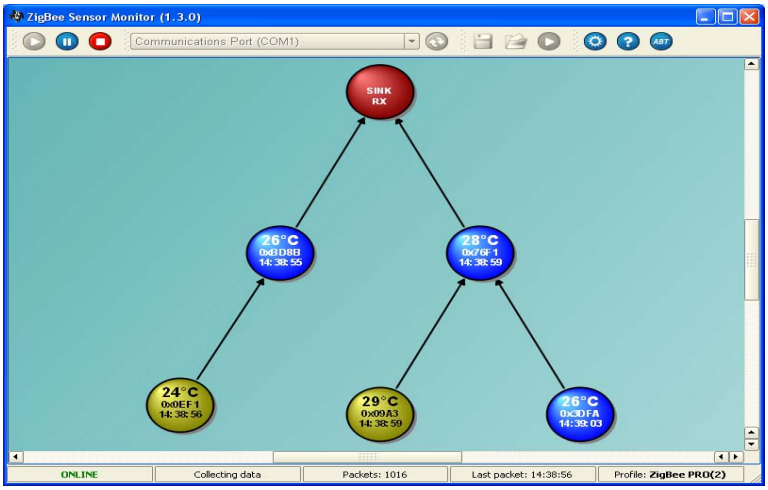
State Machine Diagrams





Sensor Monitor GUI

Sensor Monitor GUI



Descriptor

Descriptor

- Assign a Device ID & Version
- Specify Profile ID
- Specify Endpoint ID
- Initialized in AppReset()
- Must be registered with CC2530ZNP

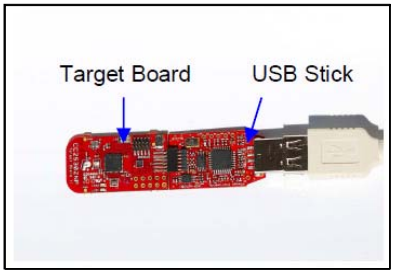
- Simple (Device)

- Endpoint Descriptor provide information about the application like:
  - Simple Descriptor (Number of defined Clusters)
  - Endpoint ID
- zb_AppRegisterRequest(srceEP)
  - Allows CC2530ZNP to know the cluster information for ONLY one endpoint
  - Simplifies handling of messages to the node by differentiating by endpoint

Labs

Lab Overview

- Lab1
  - A: Get MAC address example
  - B: RF Tester example
- Lab2: Sensor Monitor Network sample application
- Lab3: Sensor Monitor Network sample application – NV Restore
- Lab4: Form a large Zigbee network



Introduction

This workshop consists of four Labs. These labs will introduce you to ZigBee and how to build a ZigBee Application by understanding the design process for a ZigBee Network Processor (ZNP) using the CC2530ZNP Mini Kit. The workshop will help in understanding how to set up a ZigBee Mesh Network using ZigBee coordinators, routers, and end devices. The workshop will run Packet Sniffers to observe the Personal Area Network (PAN) traffic over the network. Other features to be learned are Mesh Routing, Network Commissioning, PAN Formation, etc.

Learning Objectives

- Understand ZigBee Network basics
- Develop familiarity with the development kits
- Develop familiarity with Zigbee development tools
- Learn to download code and start applications
  - Use ZNP Mini Kit – Interface Examples
- Understand the Interface between the ZNP and the Host Processor

Hardware List

- CC2511Dongle– A 2.4GHz Transceiver, 8051 MCU, 32kB Flash Memory and Full Speed USB interface
- CC-Debugger
- CC2530ZNP Mini DK

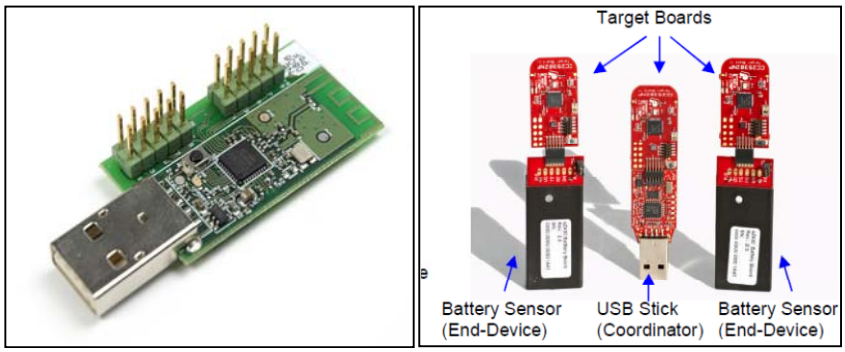


Figure: (a) CC2511 Dongle (b) CC2530ZNP Mini Kit

The ZigBee Network Processor development kit is designed to give a simple introduction to ZigBee wireless networks. The hardware consists of a CC2530 ZigBee device pre-programmed with ZigBee software and an MSP430 microcontroller that controls the ZigBee device. The development kit demonstrates simple examples of typical sensor networks with temperature, light, voltage, and movement sensors.

The target board connected to the USB stick is programmed with a coordinator sample application. The coordinator sets up the network and configures the ZigBee network parameters. A ZigBee system can only have one coordinator.

The battery powered sensors are end-devices that periodically report their key data to the coordinator. The devices can also be programmed as routers. Routers are typically used to extend the ZigBee network since they can route messages from other devices. Note that hardware for all the target boards is identical. Each of the boards can be programmed to be coordinator, router, or end-device.

Software List

- IAR Embedded Workbench - EW430 Evaluation Version
- HyperTerminal
- Spectrum Analyzer Utility

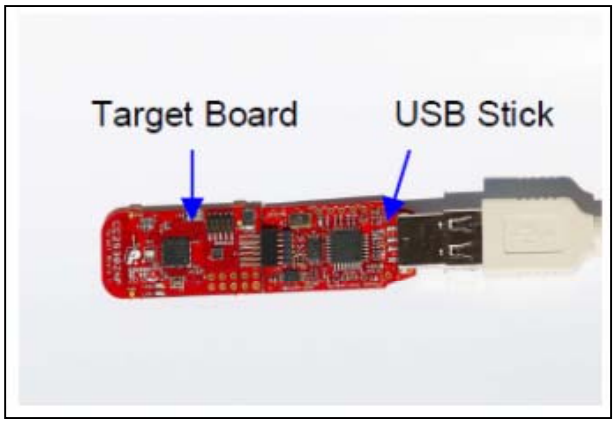
LAB 1a: Get MAC Address Example

Objective

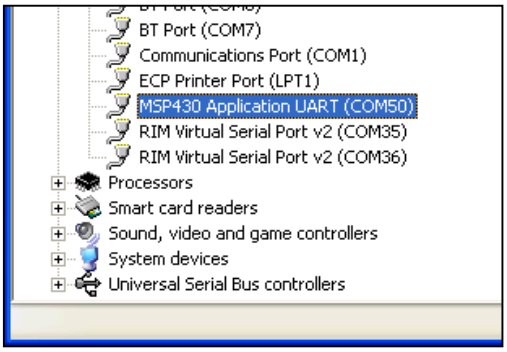
- This lab will introduce the basic concepts of starting, downloading and running applications on the ZNP Mini Kit using the IAR Embedded Work bench.
- The Lab will provide the attendees the understanding of the ZNP Interface API and how to use the interface specification document and perform debugging while developing the applications.

1. Arrange and turn on the hardware

- Connect the USB stick to a USB port on the PC. The Windows new device driver will request a new device driver to be installed. Normally the driver should be detected automatically. If the driver is not detected it can be found at the default location: **C:\Texas Instruments\CC2530ZNP Mini Kit\Drivers**



- After installation a new COM port will be installed on the PC. To see the COM port number, open the Windows Control panel – System – Hardware – Device Manager and check the COM port number under Ports (COM and LPT).

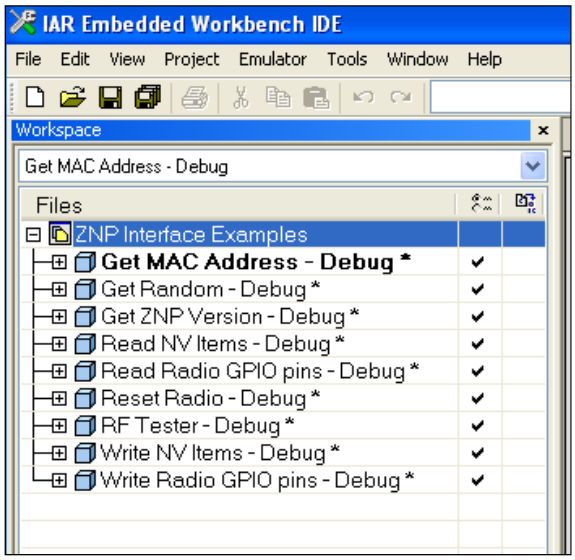


- The USB stick serves both as a virtual serial port interface and as debugger/programmer. This means that during normal operation the USB stick should be connected to the coordinator target board.

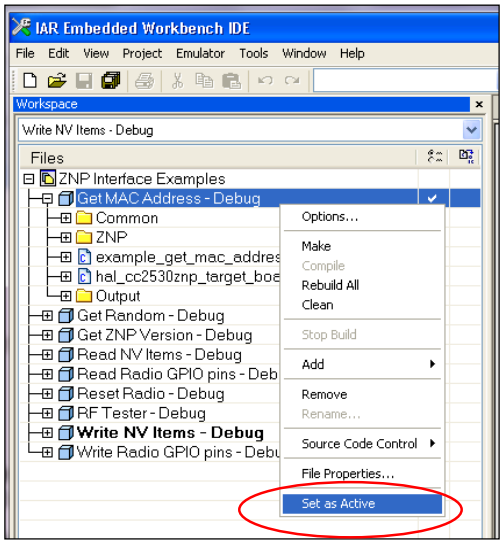
- The USB stick can also be used to program and debug the router and end-device target boards using the IAR Embedded Workbench.

2. **Start IAR Embedded Workbench (EW430) and select the "Get MAC Address – Debug" Project**

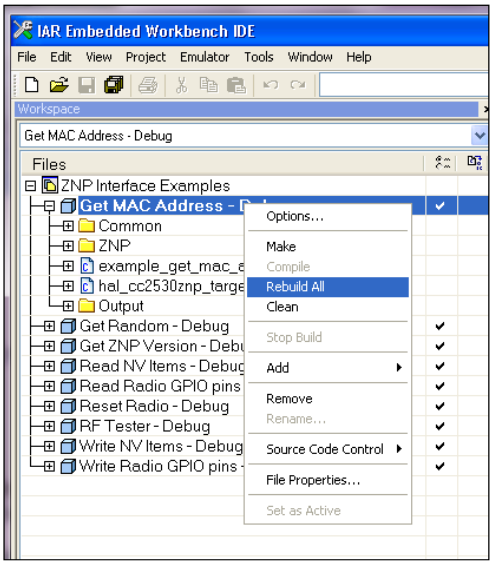
- Start IAR EW430 and open the "**ZNP Interface Examples.eww"** workspace from the following folder: **[C:\Texas Instruments\CC2530ZNP Mini Kit\ZNP Examples\IAR\]**
- In IAR EW, select the Get MAC Address - Debug from the pull down as shown below:



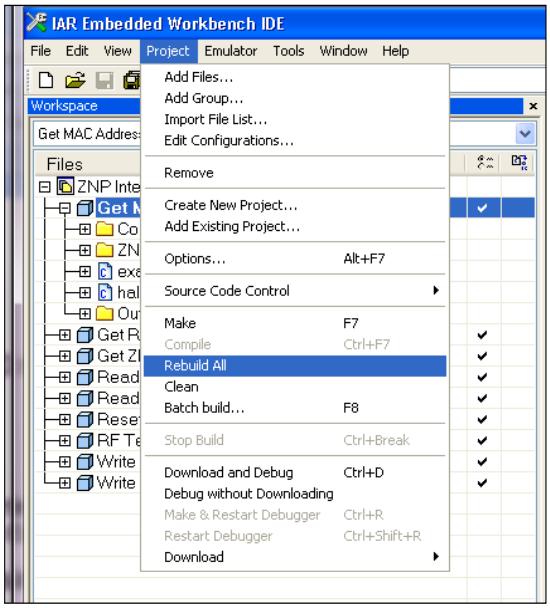
- In IAR EW, select and right click on the "Get MAC Address – Debug" Project. Select the option "Set as Active".



- In IAR EW, select and right click on the "Get MAC Address – Debug" Project. Select the option "Rebuild All".



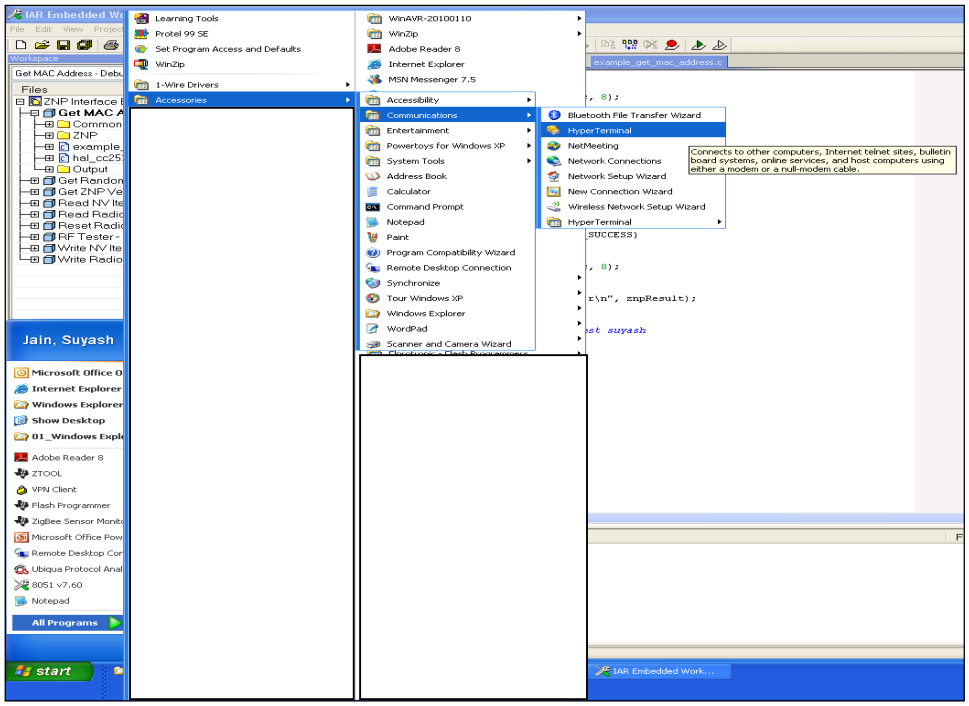
- **Another method to Build:** In IAR EW, select Project Menu -> Rebuild All



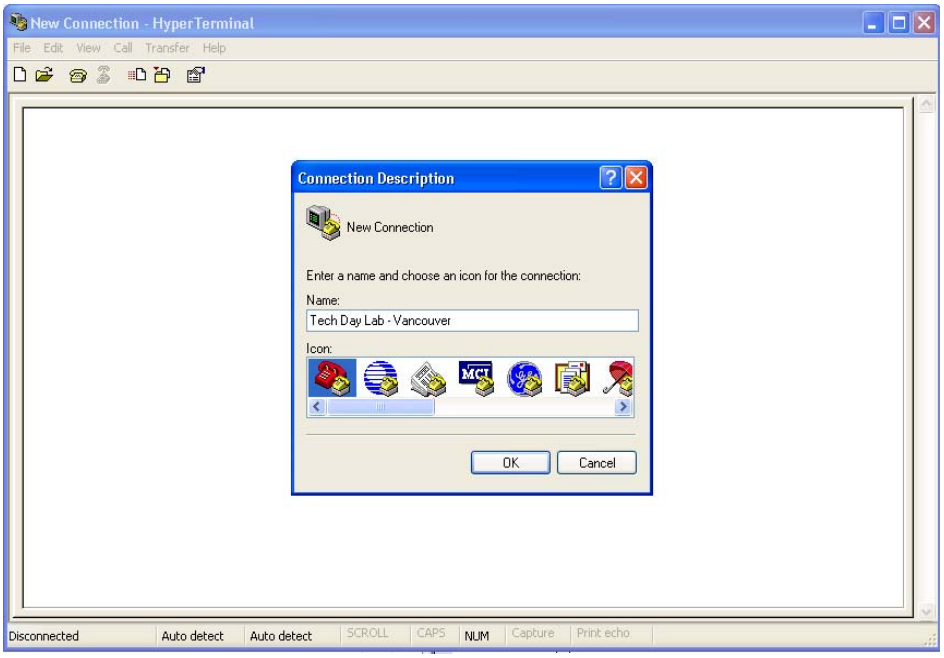
- **Alternatively**, you can use the hotkey – F7

3. Start and configure the Hyper-terminal Application

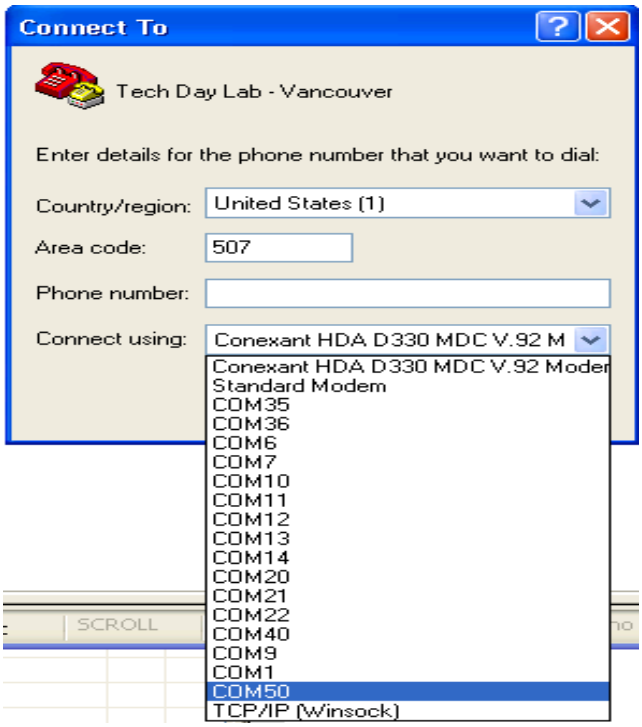
The USB stick connects to the PC using a virtual USB serial port. The serial port interface can be accessed through a terminal program. The Windows default terminal interface can be found on the Windows [Start menu] -> [Accessories] -> [Communication] –> [HyperTerminal].



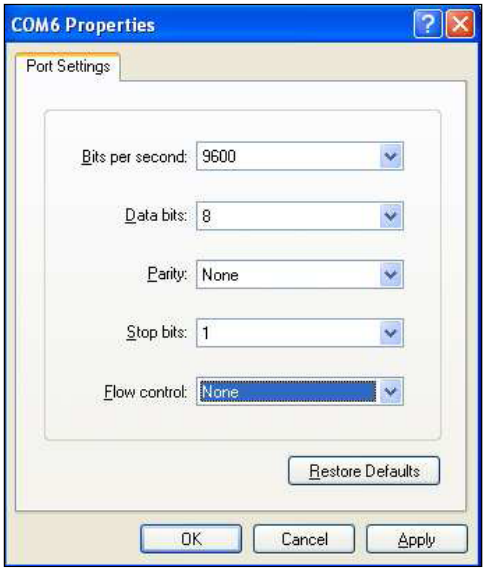
After starting HyperTerminal, give a name to the connection ex- Tech Day Lab, etc



- Next, select the COM port used by the USB stick.



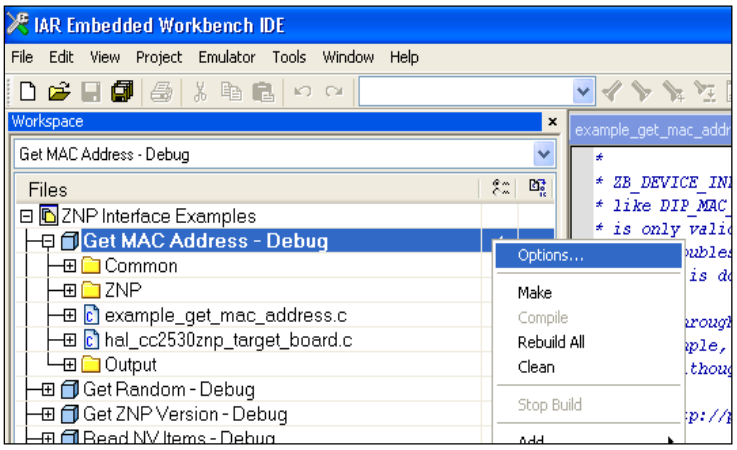
- In the COM port properties, select 9600 baud, 8 data bits, no parity, 1 stop bit, no flow control, see window below and select Apply and Ok.



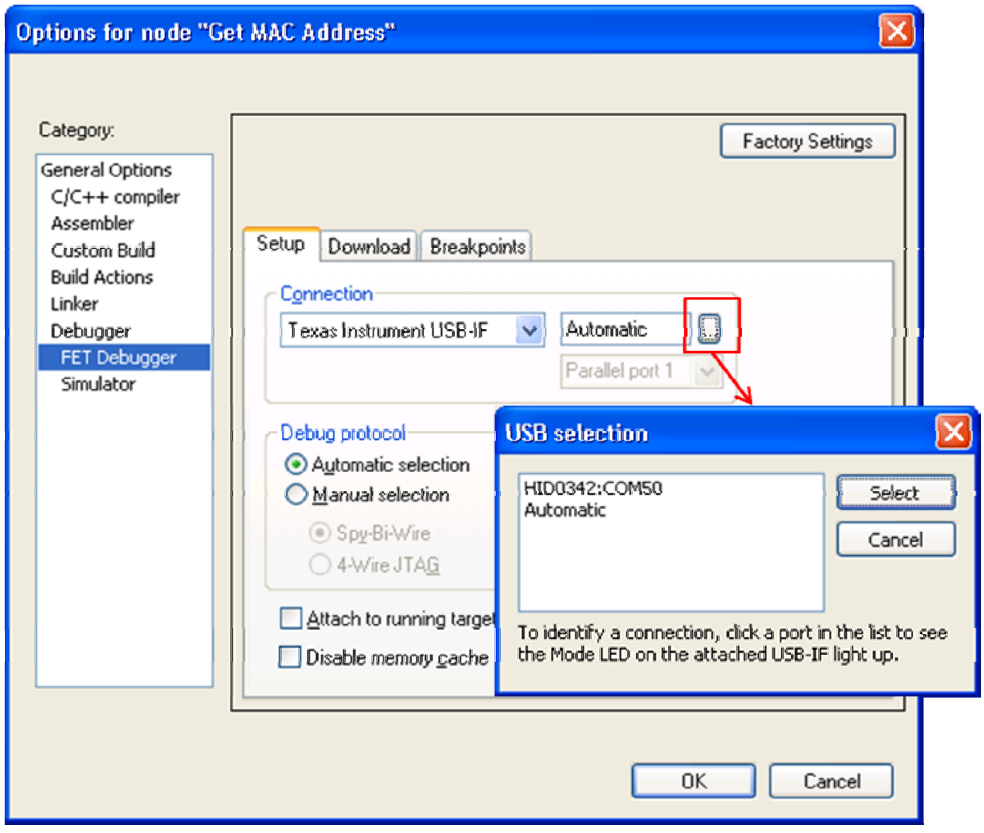
4. Download the Code to the MSP430F2274

Step 1: If multiple CC2530 ZNP Mini Kit's are connected to the PC's COM ports, you can select the device to which the code will be downloaded by selecting its COM port as described below

- ***Right click*** the project "Get MAC Address – debug"

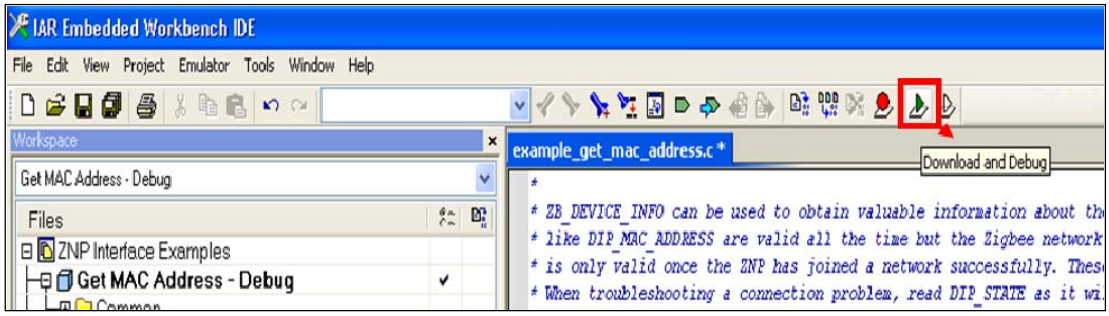


- Select the Option "***FET Debugger***" in the Category window as shown below. Then under the ***tab Setup***, select the highlighted button and ***select the desired COM port***

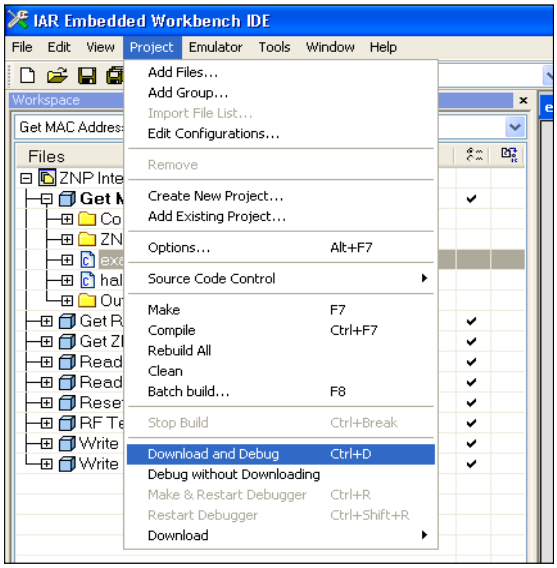


Step 2: Download the Code to the MSP430

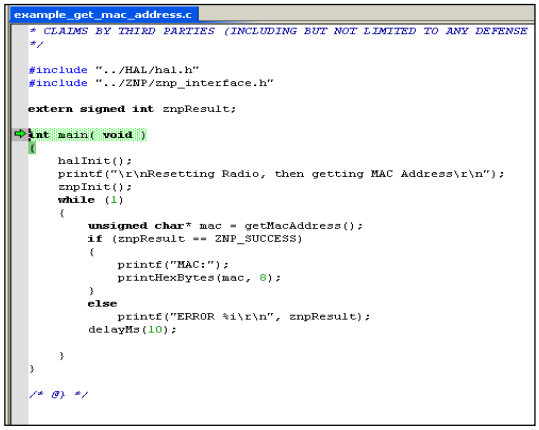
- Use the highlighted Play icon to download the code to the MSP430.



- **Other method to download the code:** In IAR EW, select Project Menu -> Download and Debug

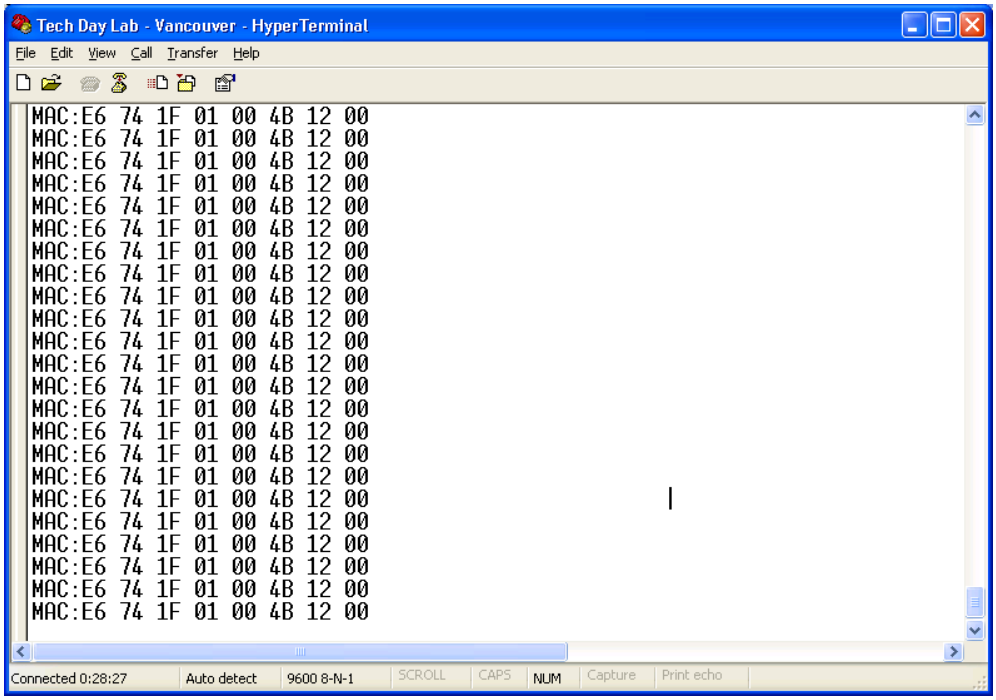


- **Alternatively**, you can use the hotkey – Shift+D
- After downloading the code, IAR will wait for the run command at the main



5. Run the application and observe the output on the HyperTerminal

- Hit Hot key F5, and example will start to run
- Observe the output of the example at the HyperTerminal – continuously reads out the MAC address/ IEEE address of the ZNP displayed as LSB first as shown in figure below.



- So the IEEE Address of the Device = 0x00124B00011F74E6
- This is the Primary Address of the CC2530 programmed in the information memory at the time of production. This is provided by the Texas Instruments. 1st 3 Bytes are the Texas Instruments OUI – Organizations Unique Identifier [i.e. 0x00, 0x12, 0x4B.]. The last 5 bytes are added by the organization to identify the unique devices.

6. Modifying and Debugging the Example

By modifying the code we will do the following

- Add the Application Menu – display additional information on start-up
- Understand Modifying printf statements and adding compile flag
- Will obtain byte aligned printout of IEEE address
- Understand the debugging and matching command with CC2530ZNP Interface Specification

Task 1: Understand - Adding the Compile Options, Application Menu, and Additional code for byte aligned output of IEEE address

- Add the following code to the "example_get_mac_address.c for application menu and modifying the printf statements

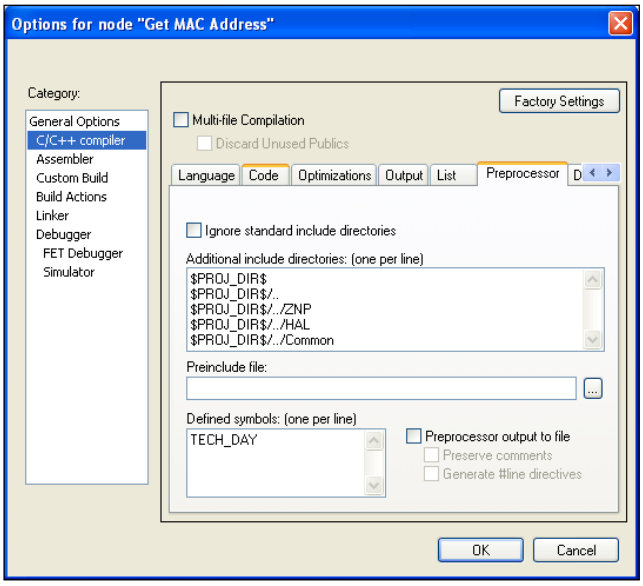
```
int main( void )
{
   halInit();
   printf("\r\n-------------------------------");
   printf("\r\nTech Day - Application Example");
   printf("\r\nCC2530ZNP");
   printf("\r\n-------------------------------");

   printf("\r\nResetting Radio, then getting MAC Address\r\n");
   znpInit();
   while (1)
   {
      unsigned char* mac = getMacAddress();
      if (znpResult == ZNP_SUCCESS)
      {
    #ifdef TECH_DAY
            printf("MAC: 0x");
    #else
            printf("MAC:");
    #endif
        printHexBytes(mac, 8);
      }
      else
         printf("ERROR %i\r\n", znpResult);
      delayMs(100);

   }
}
```

Figure: Text in Red Shows the changes to be made in the main code in the file example_get_mac_address.c

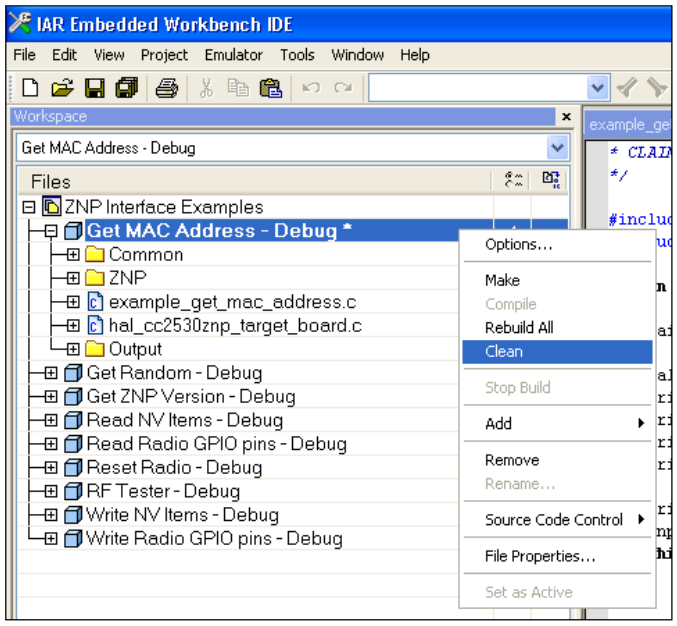
```
void printHexBytes(unsigned char* toPrint, unsigned char numBytes)
{
#ifdef TECH_DAY
  for (int i=(numBytes-1); i>=0; toPrint[--i])
       printf("%02X ", toPrint[i]);
    printf("\r\n");
#else
  for (int i=0; i<numBytes; i++)
        printf("%02X ", toPrint[i]);
    printf("\r\n");
#endif
}
```

Figure: Text in Red shows the changes to be made in the function printHexBytes() in the file utilities.c

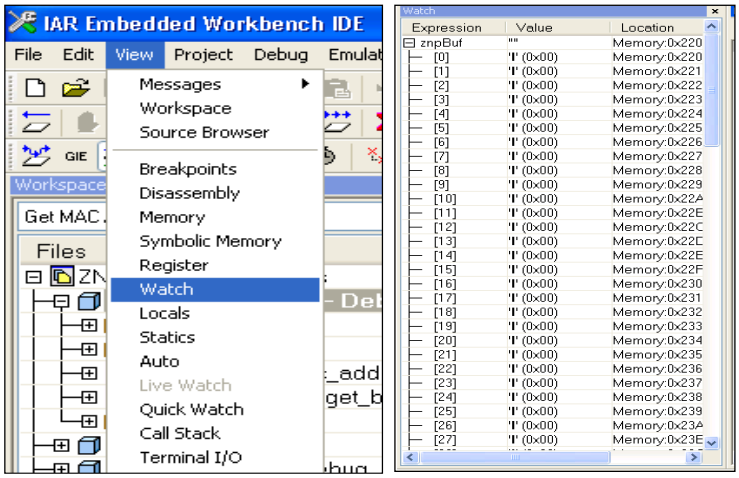
- **Add the compile option** TECH_DAY to the IAR EW project
- Hit Hot Keys Alt+F7 to open the Project Options Dialog Window
- Select C/C++ Compiler and the Preprocessor tab
- Add TECH_DAY to the Defined symbols window
- Select OK button



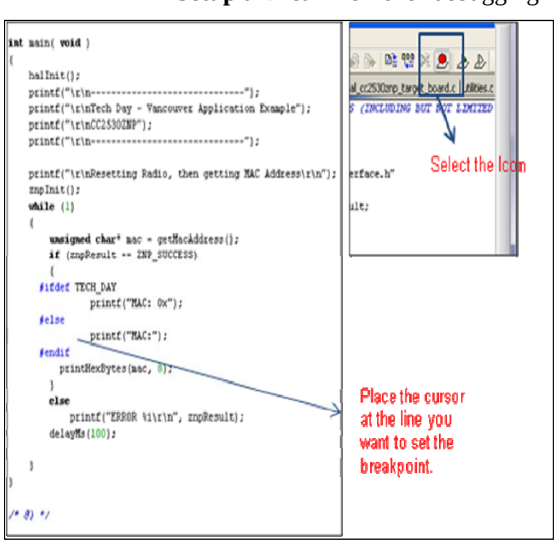
- In IAR, **Clean and Rebuild the Project**
  - Before building and debugging it is always a good procedure to clean the projects before re-building. To Clean: Right Click on the project "Get MAC Address – debug" and select the option Clean, as shown in the figure below
  - To Rebuild: Follow the steps described earlier



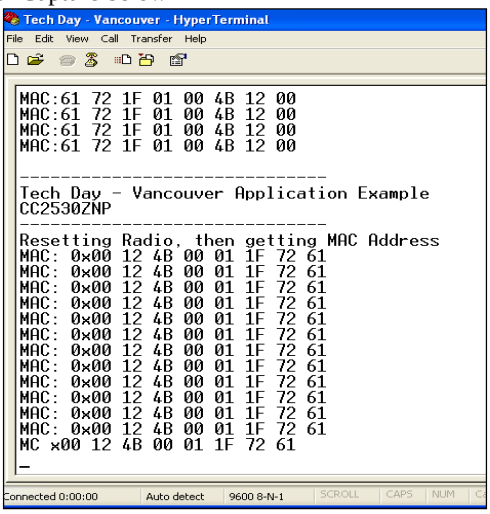
- **Set up a Debug Watch Window** to view various variables. For this example we will see the znpBuf array which is used to send and receive the commands over the SPI interface between the MSP430 and CC2530ZNP in the example code.
  - Select Menu option -> View -> Watch
  - In the watch window as shown in figure below, type the name of the variable (**znpBuf**) to monitor



- **Setup a Break Point** for debugging

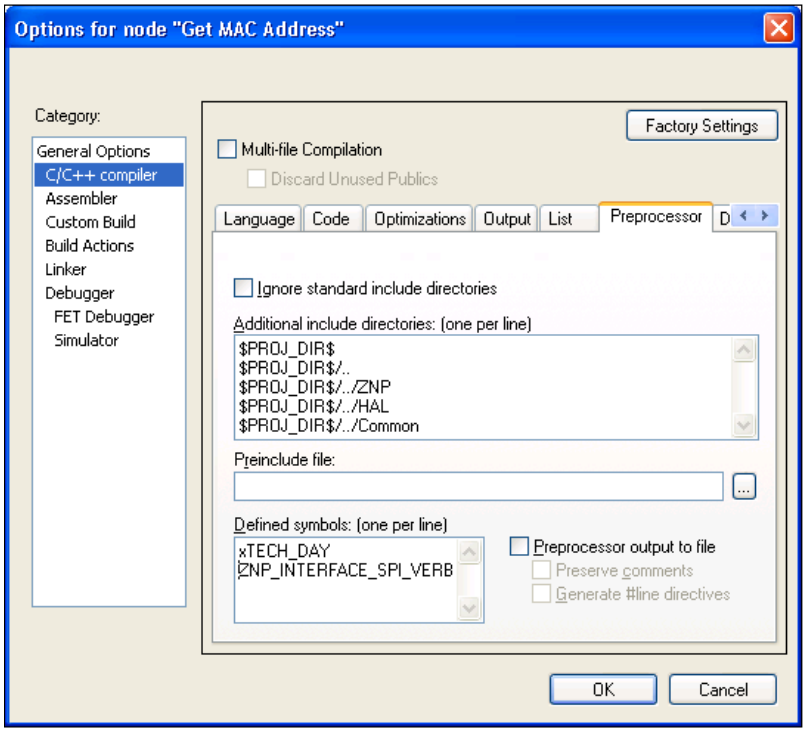


- **Download and run** program to see the effects of changes in the print output. See HyperTerminal Capture below



- **Task 2: Understanding the communication over the ZNP Interface using an example command ZB_GET_DEVICE_INFO.**

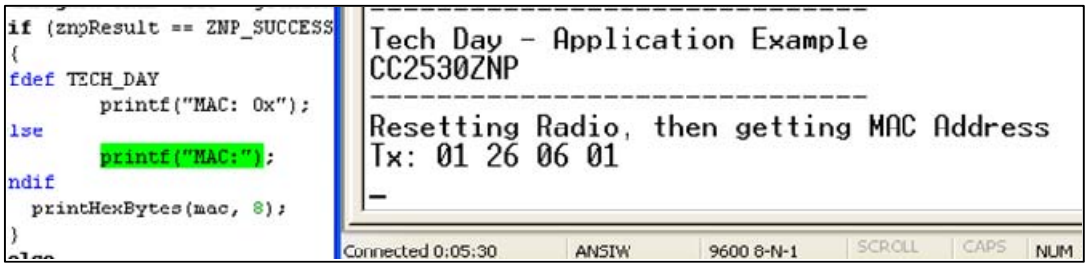
- Turn off the compile option TECH_DAY and add a compile option ZNP_INTERFACE_SPI_VERBOSE (Adding a small case 'x' in front of the compile option disables it.)



- Download the code as described earlier to the MSP430

- Run the program

- Program will then stop at the set breakpoint as seen in the figure below. The figure also shows the capture of the HyperTerminal window showing the Command sent over the SPI Interface to the CC2530ZNP



- **Referring to the Page 40 of the ZNP Interface Specification document**, the byte sequence of the command sent to the CC2530ZNP is shown in the figure below

SREQ:

1	1	1	1
Length = 0x01	Cmd0 = 0x26	Cmd1 = 0x06	Param

- Where Param is selected by the value in the table below, for this example we request IEEE address, i.e. Param = 1.

Parameter	Size	Description
0	1 byte	Device state – See 4.5.50
1	8 bytes	Device IEEE address
2	2 bytes	Device short address
3	2 bytes	Short address of the parent device
4	8 bytes	IEEE address of the parent device
5	1 byte	Channel on which the ZigBee network is operating
6	2 bytes	PAN ID of the ZigBee network
7	8 bytes	Extended PAN Id of the ZigBee network

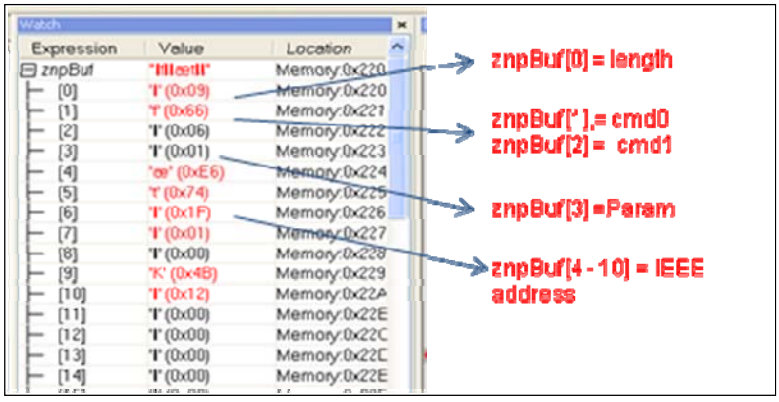
- The HyperTerminal window shown above shows the command sent to the CC2530NP

- ZNP then processes the command and sends the SRSP (Synchronous Request Synchronous Response) back in the format shown below

SRSP:

1	1	1	1	8
Length = 0x09	Cmd0 = 0x66	Cmd1 = 0x06	Param	Value

- Where Param is the value sent the command and value is requested value for the Param.
- For this example, znpBuf should look like -> [0x09, 0x66, 0x06, 0x01, IEEE address the next 8 bytes] at where you have your breakpoint set. Figure below shows the capture of the znpBuf in the watch window in IAR.



7. Conclusion

- This example exercise should provide attendees a familiarity with the IAR Embedded workbench – How to open and select a project, how to build the project and download it to the target device of choice. And how to setup breakpoints for development and debugging.
- Attendees should be familiar with the command Interface ZNP provides to the Host Application processor
- Attendees should understand how to use the ZNP Interface specification document to develop and debug Zigbee applications.

8. Questions:

1. What is a globally unique identifier for a Zigbee Device?
2. What are the different parts of an IEEE address?

LAB1B: RF Tester Example

The RF Tester example enables simple RF output testing with the ZNP. It offers a convenient command line interface to change channels, set output to modulated/un-modulated, etc. The main menu explains the options:

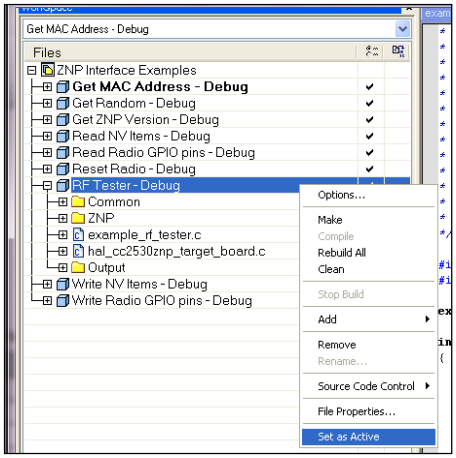
RF Tester Commands:
] Increment RF Test Channel
[Decrement RF Test Channel
M Change RF Test Mode to modulated output
U Change RF Test Mode to UNmodulated output
R Change RF Test Mode to receive
! Reset Radio
? Print this menu

- Features of this example

- The application outputs the RF signal on IEEE 802.15.4 channels- 11 to 26.
- Transmits both modulated and Un-modulated signals
  - The Modulated output will output random data modulated on the carrier.
  - The Un-modulated output option will cause the ZNP to output a clean carrier at the desired frequency.
- Allows to set the radio in RF Receive mode (transmission stops).
- Note: when set to receive, the ZNP does not report any information about what it has received; it is just muting the transmitter.
- To change the output power modify the last parameter to the method, writeRfTestParam().

1. Start IAR Embedded Workbench (EW430) and select the "RF Tester – Debug" Project

- Set the project "RF Tester – Debug" as Active project in the IAR workspace
  - Right click on the RF Tester Project and select the option "**Set as Active**"



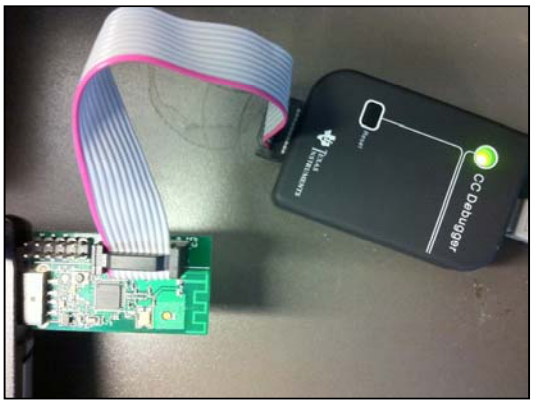
- In IAR EW, select and right click on the "RF Tester – Debug" Project. Select the option "Rebuild All". Or select Project Menu -> Rebuild All Or hit the HotKey – F7 to rebuild the project.

2. **Configure and Start the Hyper-terminal Application**

   - As done in the LAB1 Example 1 Step 3, configure and start the HyperTerminal

3. **Load the Firmware on CC2511 dongle**

   - Connect the CC-debugger to the Debug Port 3 on the CC2511 dongle as shown in the figure below



   - Start SmartRF Flash Programmer Start>All Programs>Texas Instruments>SmartRF Flash Programmer>Flash Programmer

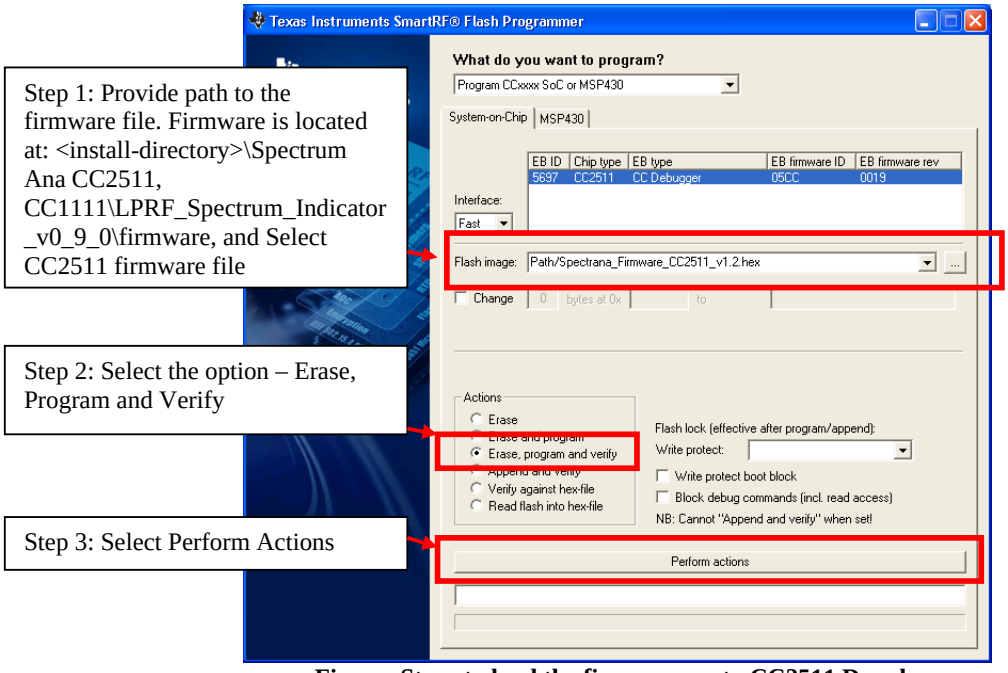


Figure: Steps to load the firmware on to CC2511 Dongle

4. Configure the Spectrum Analyzer

The LPRF spectrum analyzer is a utility developed by Texas Instruments to work with CC2511 or the CC1111 dongles, which allows sniffing the air for RF energy and displays it on the GUI as shown below.

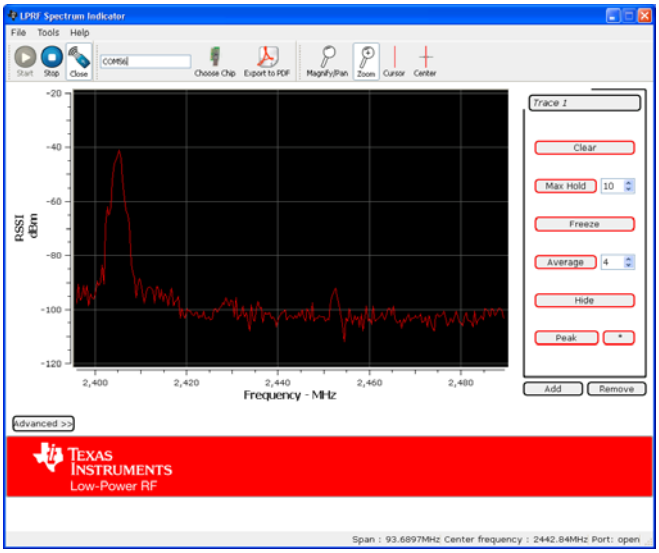


Figure: LPRF Spectrum Analyzer Utility

- As Pre-work, attendees will have the Spectrum Analyzer installed on their machines.
- Appendix A Lists the Steps to install the drivers to work with this utility.

To get started:

- Start the Spectrum Analyzer Utility and select the Chip -> CC2511

- Find out at which COM port your device is located. If it is not COM8 (default for the utility), change it to whatever is correct.

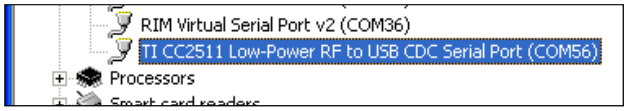
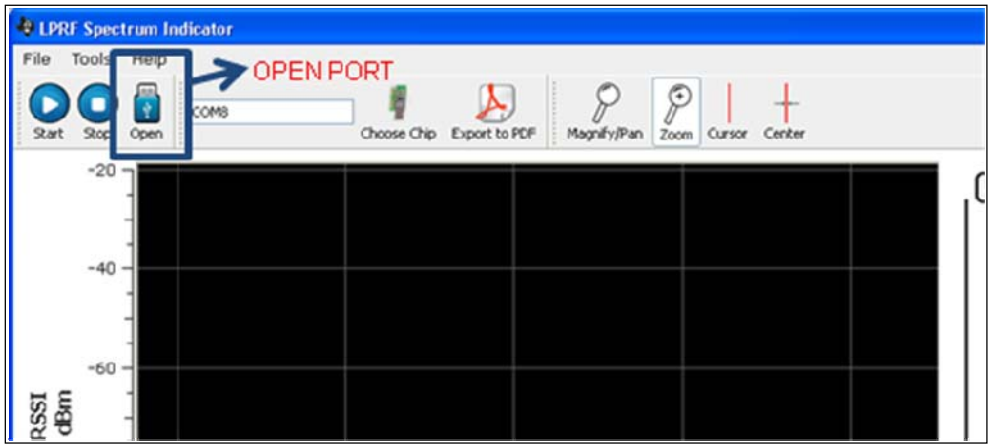
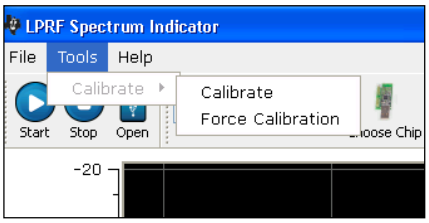


Figure: Device Manager window of PC showing CC2511 Dongle Connected on COM56

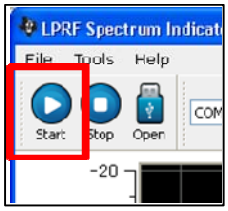
- Push the “Open” port button. The status field on the bottom right should now have changed to “Open”.



- Select Tools-> Calibrate -> Force Calibration



- Try to start the application by pressing the “Start” button.



- If nothing happens other than a message in the bottom left corner: “Calibration not performed!”. Calibrate the device by pressing the “Calibrate” button in the “Tools” menu. You should now read “ACK calibrated received” in the status field in the bottom left corner, this might take a few seconds.

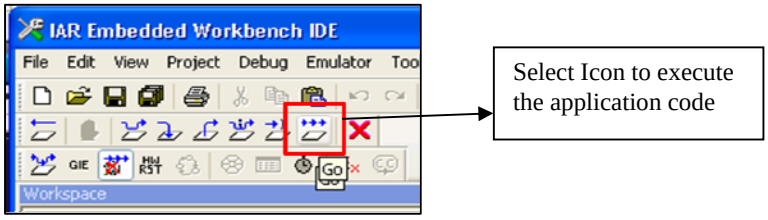
- Try to start the application by pressing the “Start” button, again.

5. Download the Code to MSP430F2274

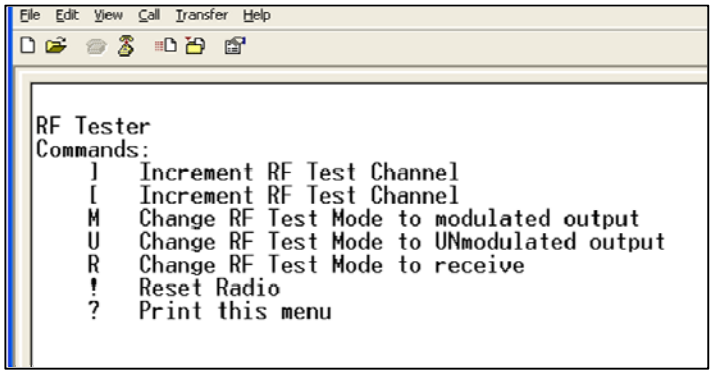
- As done in the LAB1 Example 1 Step 4, download the code to the MSP430F2274

6. Run the application and observe the output on the HyperTerminal

- After downloading the code, IAR will wait for the run command to execute the program
- Hit the Run icon or press the Hotkey – F5.



- Observe the HyperTerminal window which will display the options available and the corresponding action that will be taken.



- For this lab exercise we will Start with
  - "**U**" – Observe the RF Output on the Spectrum Analyzer GUI as shown below

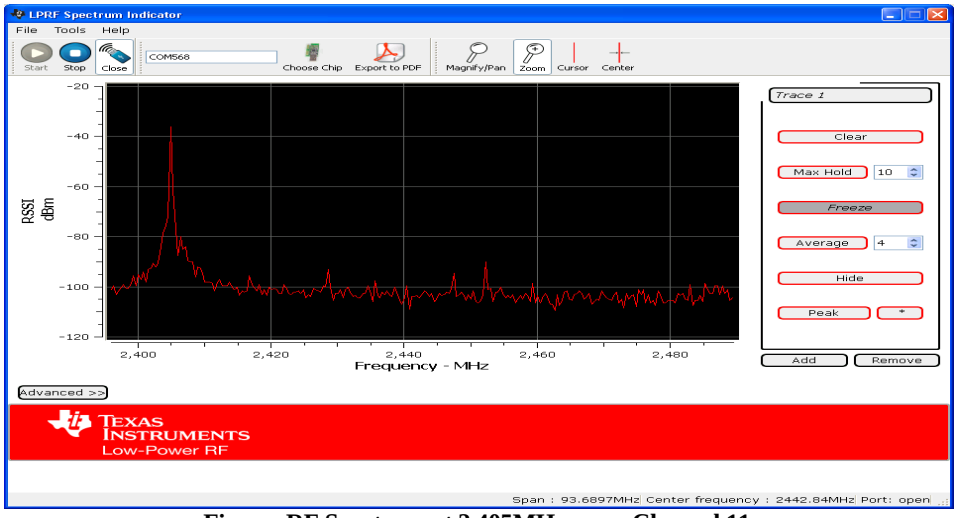


Figure: RF Spectrum at 2.405MHz or on Channel 11

- "]" – Increase the RF Test Channel
  - The RF Spectrum Should shift by 5MHz for each time '] ' is pressed in the HyperTerminal window up to channel 26.
  - Test and observe the transmitted spectrum up to Channel 26.
- "[" – Decrease the RF Test Channel
  - Observe the transmitted spectrum move by 5MHz each time "[" is pressed.

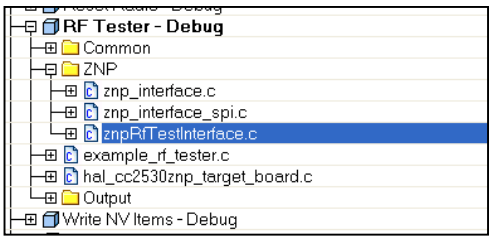
7. Modify Output Power

By default the output power in transmit mode from the RF-Tester – Debug Example is set to 4.5dBm. Now we will change the output power to -10dBm and observe the change in the spectrum.

- In the file znpRfTestInterface.c, in the function rfTest() change the last parameter RF_OUTPUT_POWER_PLUS_4_5_DBM to the function writeRfTestParam as shown below

  writeRfTestParam(mode, channel, RF_OUTPUT_POWER_PLUS_4_5_DBM);

  to **RF_OUTPUT_POWER_MINUS_10_0_DBM**



- Rebuild and download the project to the MSP430F2274
- Run the code by pressing the Run Icon or the Hotkey F5
- Select the option “U” and observe the Output power

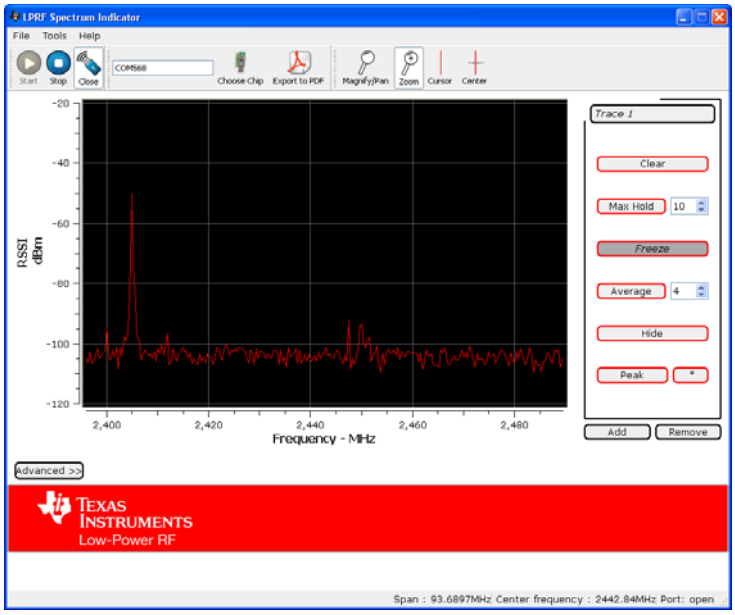


Figure: Received Power by the CC2511 dongle is lower.

APPENDIX A

This sections lists the steps for installing the drivers for the Spectrum Analyzer utility

- Install the provided driver for the virtual serial port. (*by executing the "install_driver.bat" found in the install folder of the utility)
- Use CC Debugger or an EB, and SmartRF Flash Programmer to flash your CC2511/CC111 SoC USB-dongle with the provided firmware; "firmware/Spectrana_Firmware_CC2511_v1.2.hex"/"firmware/Spectrana_Firmware_CC1111_v1.2.hex". See "http://focus.ti.com/lit/ug/swru069d/swru069d.pdf" part 4.1
- If you don't have Microsoft Visual C++ 2008 installed on your computer, install the runtime components of Visual C++ by downloading and executing vcredist_x86.exe from http://www.microsoft.com/downloads/details.aspx?familyid=9B2DA534-3E03-4391-8A4D-074B9F2BC1BF&displaylang=en
- Double-click the "LPRF Spectrum Indicator" shortcut. Follow on-screen pop-up, although the same information is rewritten next.

LAB2: Sensor Monitor Network Sample Application

Objective

In this lab you will learn

- How to form a ZigBee Network
- How to join an existing ZigBee Network
- How to observe network behavior over the air using TI packet sniffer
- How to program CC2530 using SmartRF Flash Programmer
- Key Zigbee Concepts and terminologies.
- Basics of Zigbee Network Commissioning – Channel Allocation, PANID configuration, device binding.

Following are the tasks and steps to do those for the Lab1

1. Upgrading the CC2530ZNP Flash Image

The sensor monitor network sample application has been written using the CC2530ZNP image build from the Z-Stack-2.5.0 release, it is recommended to use the image built using Z-Stack-2.5.0 for running this sample application. This section illustrates how to use the SmartRF® Flash programmer and CC-Debugger to program the CC2530 on the ZNP Mini Kit with a new ZNP Image

Following are the steps to program a new image into CC2530 on the ZNP Mini Kit.

1. Connect the CC-debugger 10 pin header to the 10-pin debug header as shown in the figure below

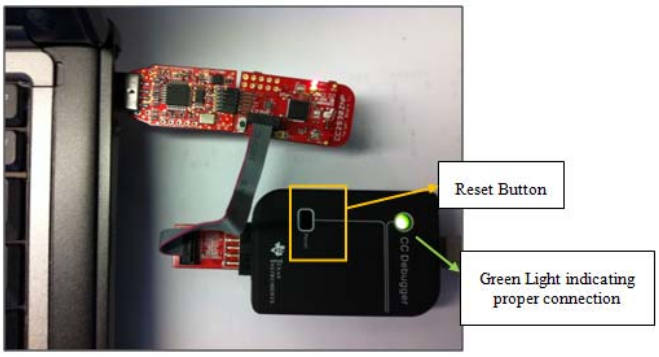


Figure: CC-Debugger Connected to CC2530ZNP Mini Kit board. Green Light indicates proper connection

2. Press the reset button on the CC-Debugger. The light on the debugger must turn green indicating the device has been correctly connected to the debugger. If the Red light remains ON, it indicates that the device is not connected properly. Ensure that the CC-Debugger's 10-pin connector is not reversed and pin 1 on the connector is connected to the pin-1 on the header and so on.
3. Start the SmartRF Flash Programmer. The device will be recognized in the SmartRF flash programmer window as shown in the figure below.

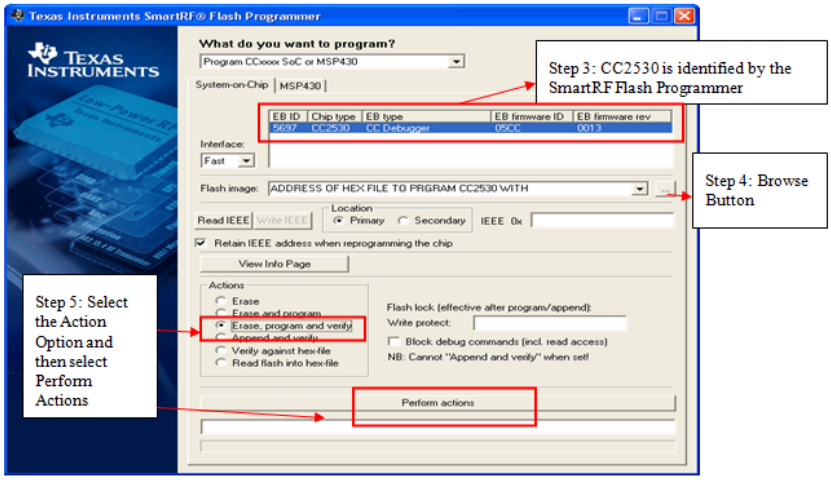


Figure: SmartRF Flash Programmer window

4. Select the desired ZNP image by pointing the SmartRF Flash Programmer to a destination on your PC, by selecting the browse option see figure above.
5. Download the image by selecting the option – Erase, program and Verify as shown in the figure 16 above.
6. Erase, program and verify success message will then appear as shown in figure below.

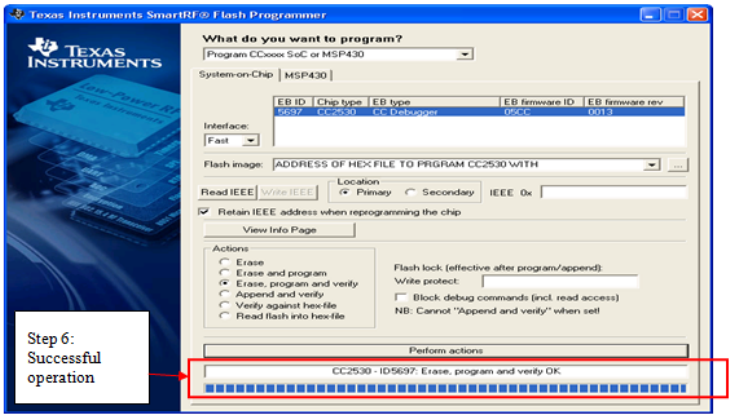
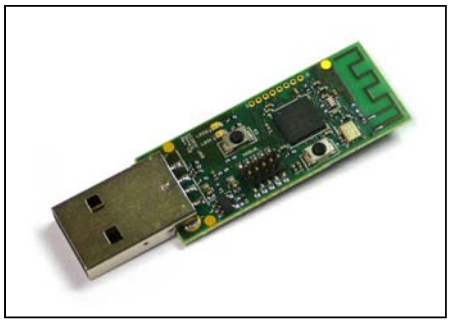


Figure: SmartRF Flash Programmer window with status message showing successful operation status message

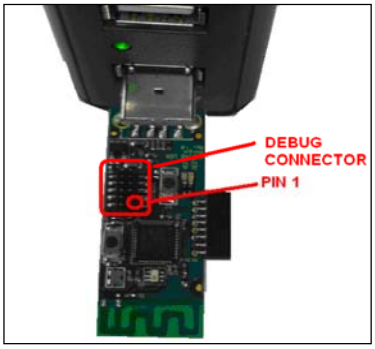
2. Prepare for Packet Sniffing - program or upgrade the CC2531 USB Dongles

- First setup and program the CC2531 dongles as packet sniffers for each group of two people



- The CC Debugger allows for debugging using IAR Embedded Workbench, as well as for reading and writing hex files to the CC253x flash memory using the SmartRF Flash Programmer software. The SmartRF Flash Programmer also has the capability to change the IEEE address of the CC253x devices.

- To program the CC2531 on the USB dongle, first plug the USB Dongle into a PC USB port (or USB hub)

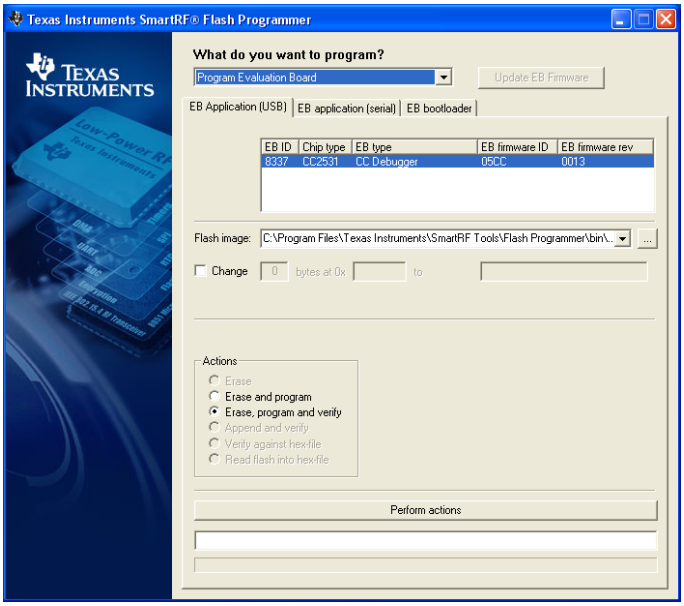


- Connect the CC Debugger to the CC2531 USB Dongle as shown below. Be sure that the ribbon cable is oriented properly, with the red stripe connected to pin 1:

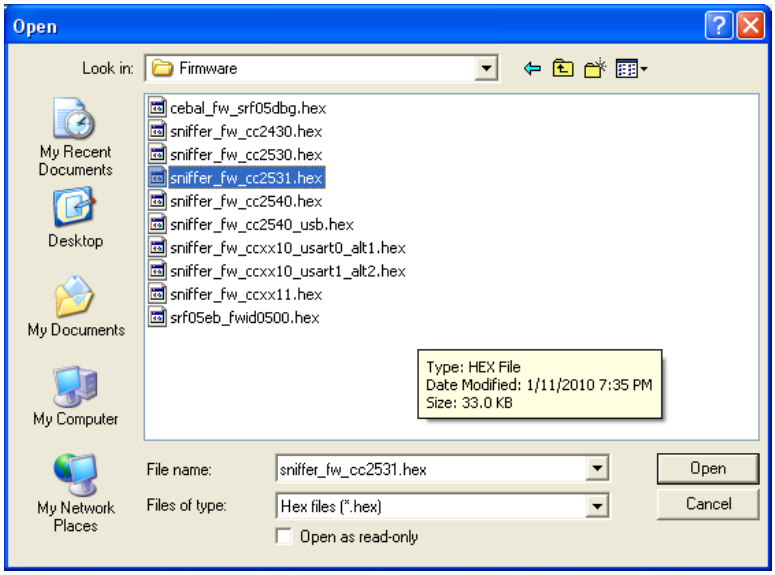
- Connect the CC Debugger to the PC USB port. The status indicator LED on the CC Debugger should turn on. If the LED is red, that means no CC2531 device was detected. If it is green, then a CC2531 device has been detected. If the USB Dongle is connected and the LED is red, try pressing the reset button on the CC Debugger. This resets the debugger and re-checks for a CC2531 device. If the LED still does not turn green, re-check that all cables are securely connected.

- Once the CC Debugger status LED is showing green, you are ready to use IAR to debug or to read or write a hex file from/to the USB Dongle.

- To start the SmartRF Flash Programmer application go into your programs by choosing Start > All Programs > Texas Instruments > SmartRF Flash Programmer > SmartRF Flash Programmer. The program should open up the following window:



- The sniffer firmware to be programmed is located at the following folder: **[C:\Program Files\Texas Instruments\Packet Sniffer\General\Firmware\ sniffer_fw_cc2531.hex]**



- After a successful programming the green LED on the CC2531 USB dongle will be on, indicating successful programming of the Packet Sniffer firmware.

3. Network Commissioning – Channel and PANID Allocation

- Table below specifies the channel number and PanId for each group in the workshop

Group Number	Channel Number	PANID
1	11 (0x0B)	0x0001
2	12 (0x0C)	0x0002
3	13 (0x0D)	0x0003
4	14 (0x0E)	0x0004
5	15 (0x0F)	0x0005
6	16 (0x10)	0x0006
7	17 (0x11)	0x0007
8	18 (0x12)	0x0008
9	19 (0x13)	0x0009
10	20 (0x14)	0x0010
11	21 (0x15)	0x0011
12	22 (0x16)	0x0012
13	23 (0x17)	0x0013
14	24 (0x18)	0x0014
15	25 (0x19)	0x0015
16	26 (0x1A)	0x0016
17	11 (0x0B)	0x0017
18	12 (0x0C)	0x0018
19	13(0x0D)	0x0019
20	14 (0x0E)	0x0020
21	15 (0x0F)	0x0021
22	16 (0x10)	0x0022
23	17 (0x11)	0x0023
24	18 (0x12)	0x0024
25	19 (0x13)	0x0025
26	20 (0x14)	0x0026
27	21 (0x15)	0x0027
28	22 (0x16)	0x0028
29	23 (0x17)	0x0029
30	24 (0x18)	0x0030

- Modify the following two parameters in the file **application_configuration.h** to match the assigned Channel Number and the PANId.

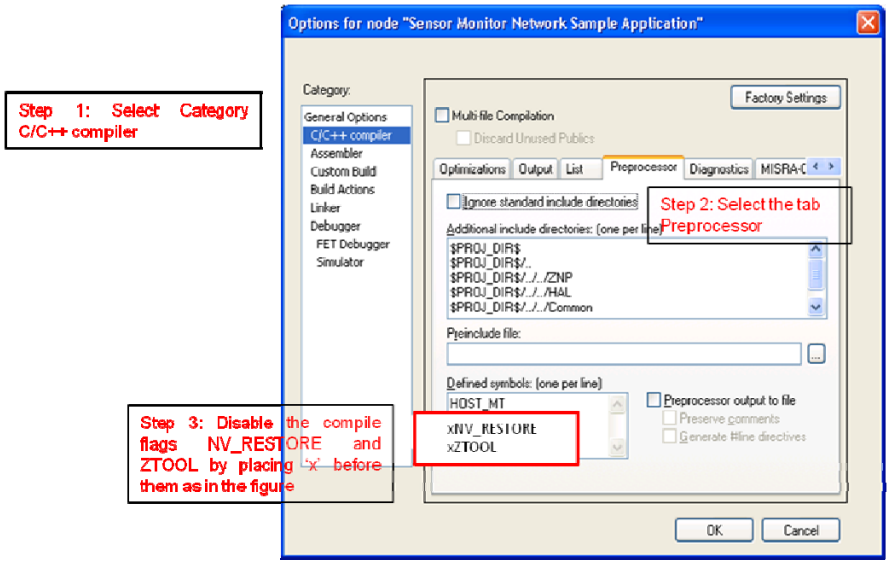
```
/*channel for network operation*/
#define DEFAULT_CHANNEL 11

/*panID for the Network */
#define MYPANID 0x0001
```

- Add the command to set pan Id in the function appStartRequest() in file sensor_demo_app.c as shown in the figure below

```
static void appStartRequest(uint8 rtrType)
{

  unsigned char deviceType ;
  if (rtrType)
  {
    deviceType = ROUTER;
  }
  else
  {
    deviceType = END_DEVICE;
    appFlags |= (appLowPwrF | appSetPollF);  /* flags set for end devic
  }

  /*Set Pan ID */
  setPanId(MYPANID);

  /*Set ZigBee Device Type */
  setZigbeeDeviceType(deviceType);

  /*Register application endpoint for either router or end device*/
  sapiRegisterApplication(srceEP());

  /*Start Application*/
  sapiStartApplication();
```

4. Download the code on the MSP430F2274 Using IAR Embedded Workbench

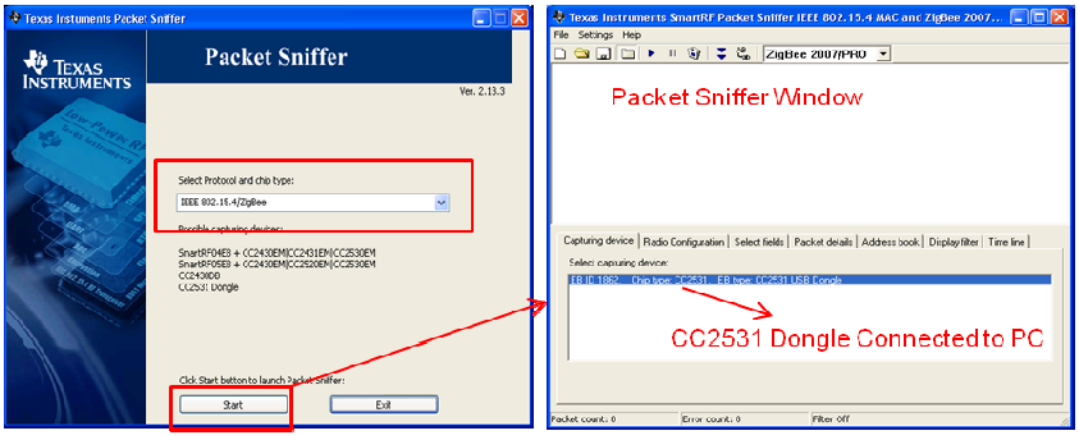
- Ensure that only HOST_MT compile flag is enabled.



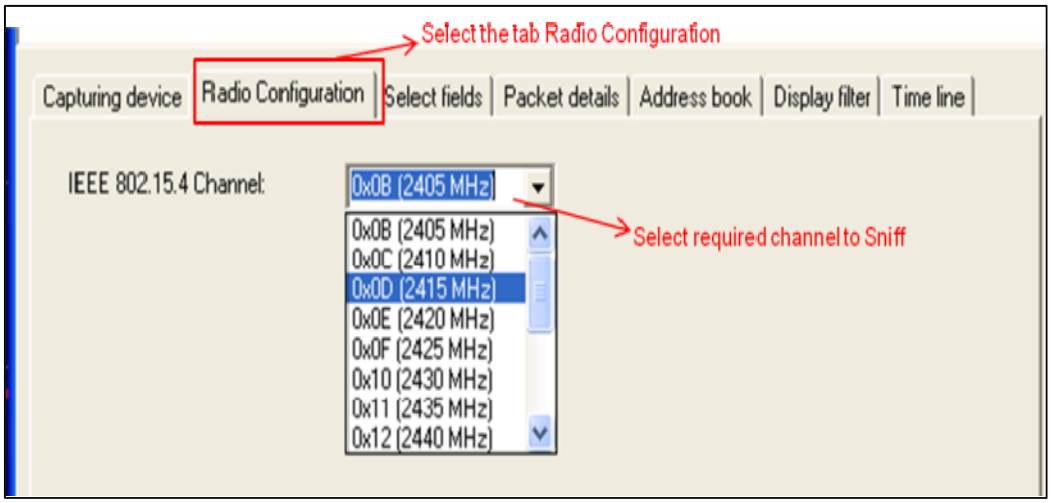
- Using IAR Embedded workbench, download the code on all the three targets boards on the ZNP Mini kit one by one. After successful download both LED's on the board will start to blink at the interval of ~1sec.

5. Start the TI Packet Sniffer

- Select ZigBee and the start to open an instance.

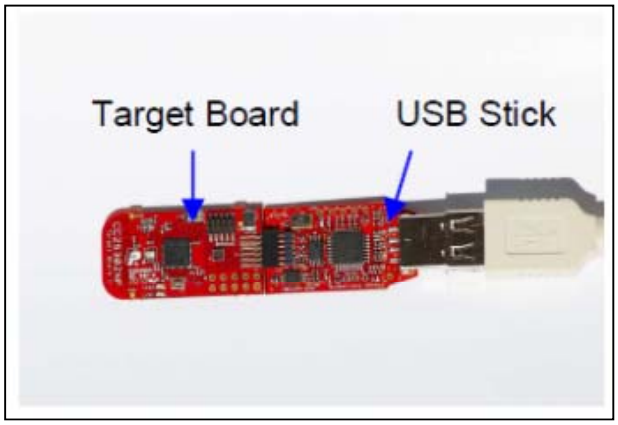


- Start the Packet Sniffer on the channel assigned to the Group. Figure below shows how to select the required channel for the selected USB dongle.



6. Start the Network

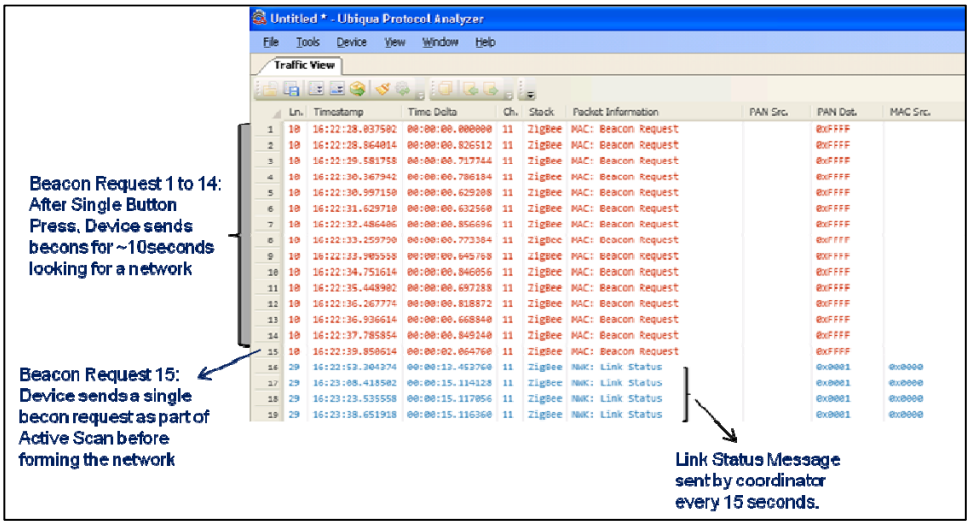
- Arrange and turn on the hardware: Connect the USB stick to a USB port on the PC. The Windows new device driver will request a new device driver to be installed. Normally the driver should be detected automatically. If the driver is not detected it can be found at the default location: C:\Texas Instruments\CC2530ZNP Mini Kit\Drivers



- Rebuild and download the code modified in the step 3 above on the MSP430430F2274 on the ZNP Mini Kit target boards.

- If the code is downloaded properly, you will see both the LED's on the target board blinking at an interval of 1 second.

- **Zigbee Coordinator** is the first device to come in the network. It is responsible to form the network.In this sample application the ZigBee Coordinator connects to the PC using a virtual USB serial port. We will configure the device connected to the PC using the USB port as coordinator -> **press the button on the target board once.**



7. **Start the Sensor Monitor GUI**

- Sensor Monitor GUI is Start > All Programs > Texas Instruments > Zigbee Sensor Monitor > Zigbee Sensor Monitor

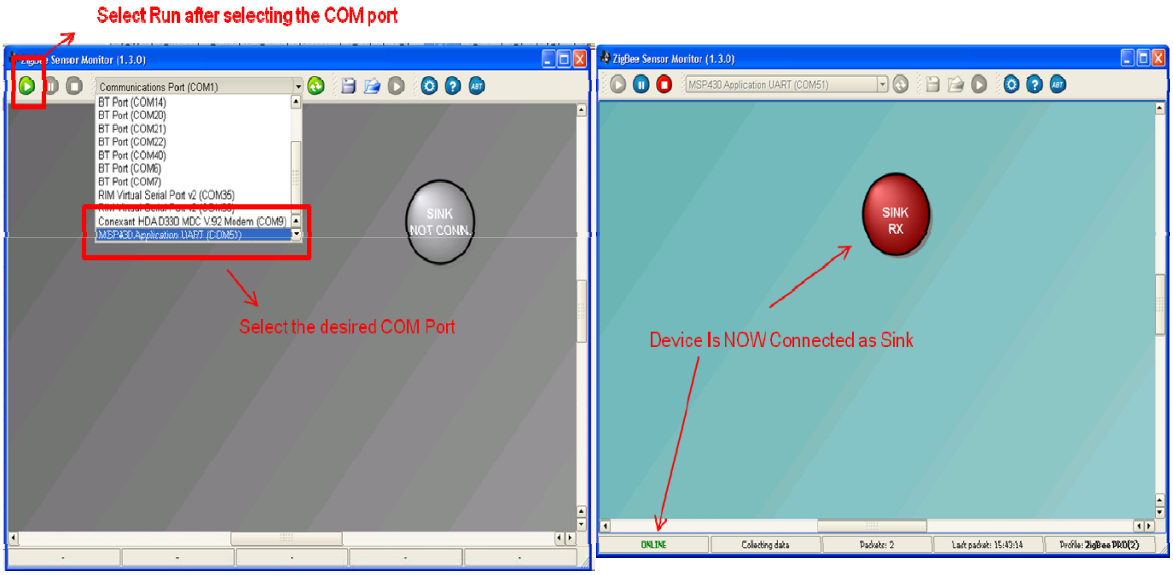
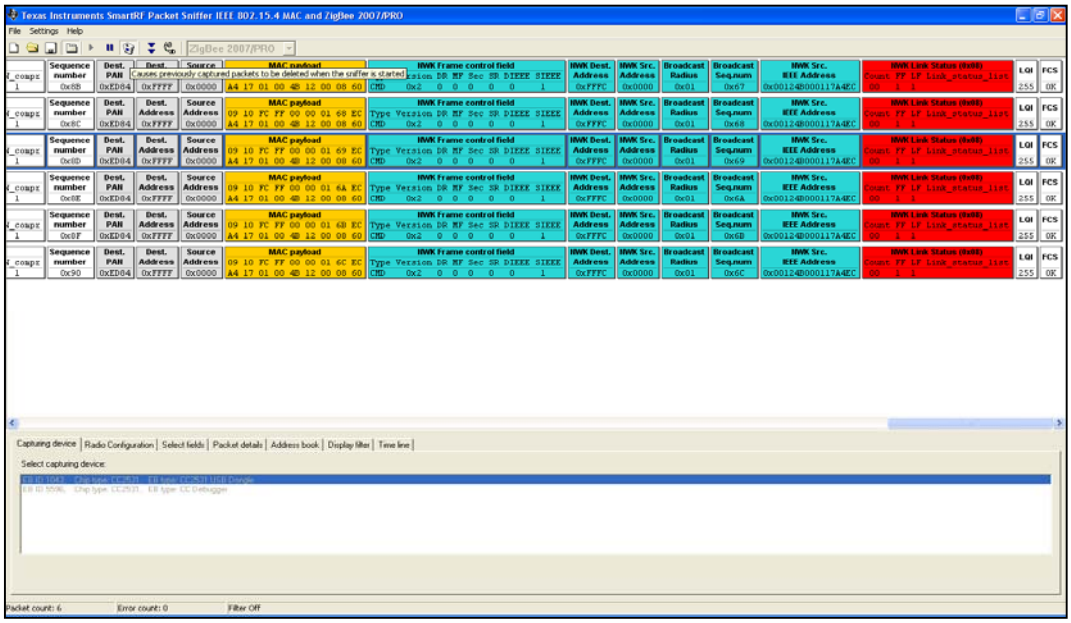


Figure: The Sink (coordinator) turns on to be red in the Sensor Monitor GUI. The network is now formed and ready to receive connection from other devices (Sources) on the network.

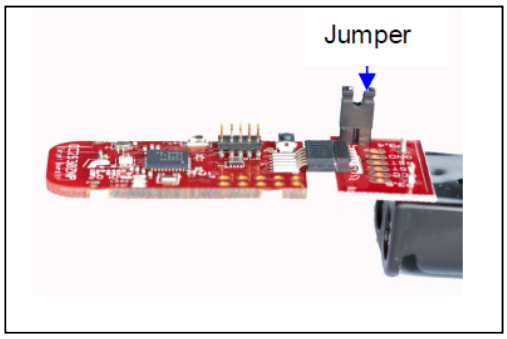
- Observe the Packet Sniffer Window for Link Status messages:

Verify also with the Packet sniffer that you see the Link Status message coming from your Coordinator. Observe that the PAN ID from Sensor Monitor (Remember byte order, 0xED84) matches a PAN ID in the Packet Sniffer.



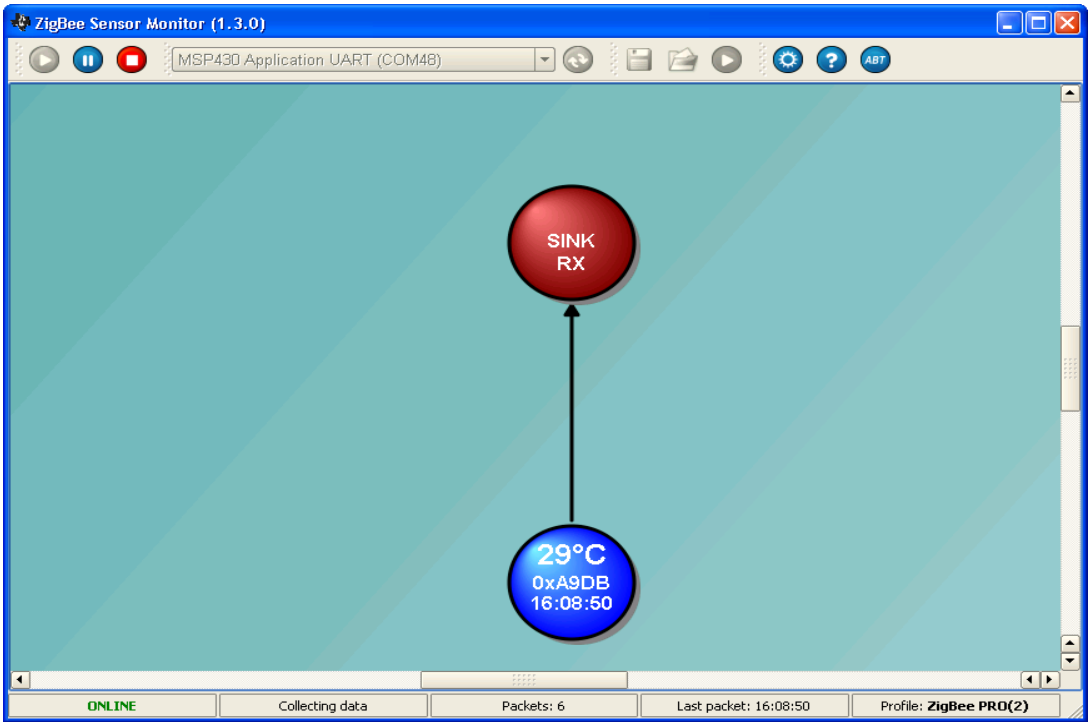
8. Prepare to add New Devices to the network

- Insert batteries in one or more of the battery boards and make sure the jumper is connected to power the end-device.



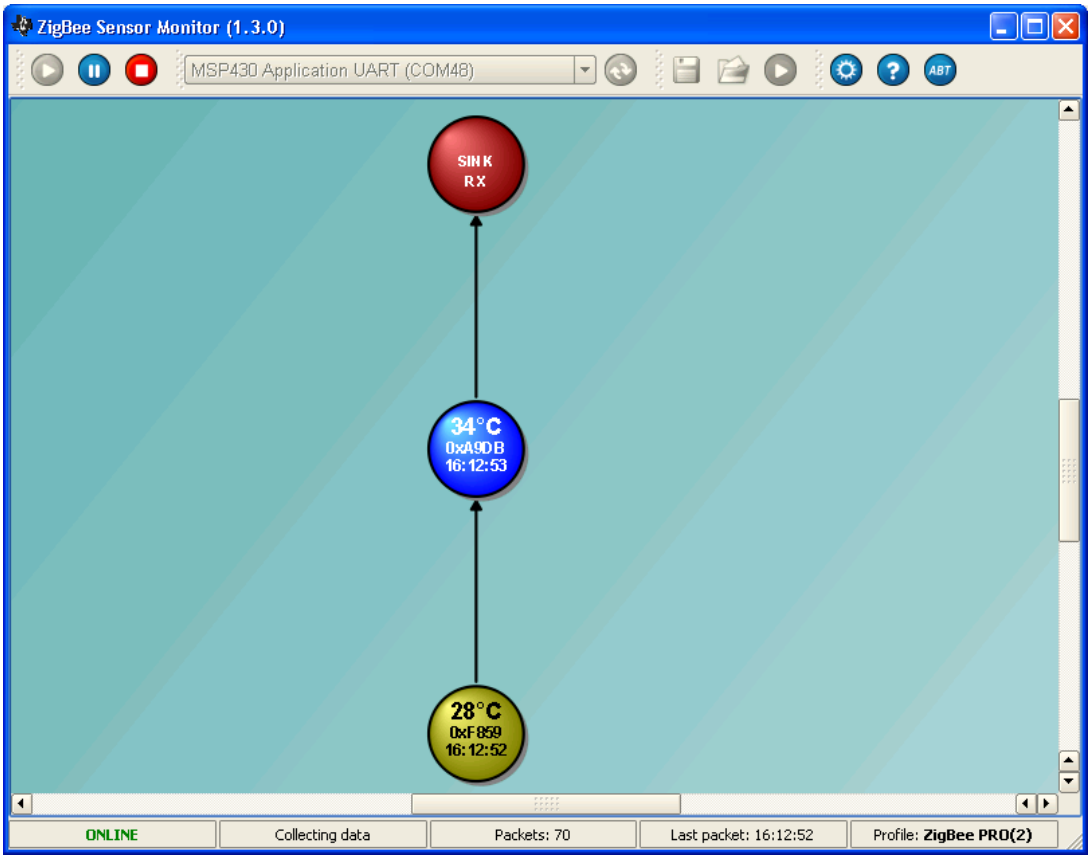
9. Add the Source Device - Router to your network

- Follow the steps used to configure the device as the Coordinator. This time since the network formed above already exists the device will join that network as a router.
- You will see on the Sensor monitor GUI display the router as a blue circle. Showing Temperature, Device shot address and time stamp when the message was received.



10. Add the Source Device - End Device to your network.

- Press the button on the target board twice to configure the device as an end-device.
- You will see on the Sensor monitor GUI display the end-device as a yellow circle, showing Temperature, Device short address and time stamp when the message was received.



- When the end-device sends information to the coordinator
  - An RX-message will be indicated as a device circle blink on the sensor monitor GUI.
  - The green LED on the end-devices will blink when they are sending a message.

NOTE: If the end device loses connection to the coordinator the green LED will be on constantly.

- The sensor messages from the source devices are sent every 10 seconds to the Sink node.

11. Questions

1. How does a ZigBee network get formed?
2. What is a Link Status Message?
3. How would you verify that every end device is connected to the network?
4. What happens to the End Device if you remove the Coordinator, check Packet sniffer?
5. What happens if you bring the Coordinator back up on the network? What do you need to do to re-establish your network?

.

LAB3: Sensor Monitor Network Sample Application – NV Restore

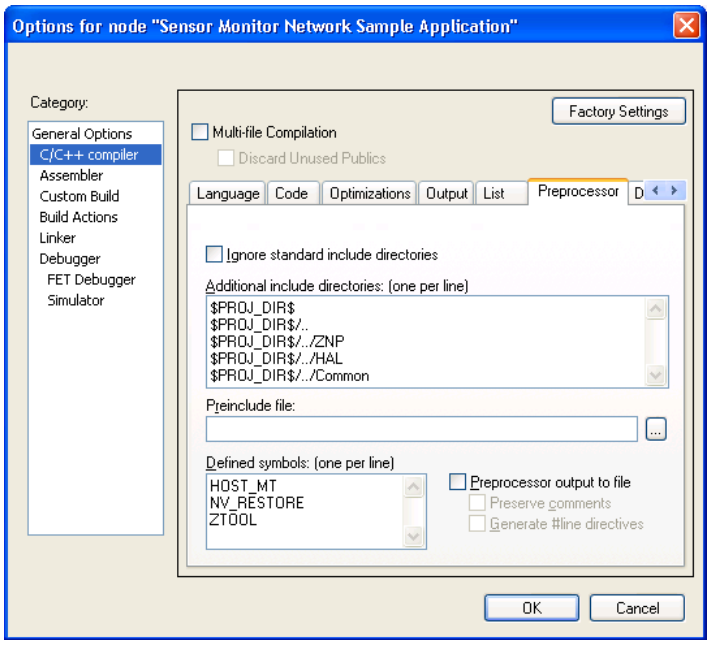
Objective
You will learn

- Restarting/Rejoining the network silently in case of power loss (battery run out, power cycle, etc)
  - (using the Non-Volatile memory for Zigbee Network parameter storage)
- Understand Parameters required for Zigbee Network Startup – Network Specific and Device Specific Parameters

Following are the steps for this Lab 2 Example.

1. Modify the Sample application

- Add Compile Flag NV_RESTORE and ZTOOL



- Remove the command to set pan Id in the function appStartRequest() by commenting it out as shown in the figure below

```
static void appStartRequest(uint8 rtrType)
{

  unsigned char deviceType ;
  if (rtrType)
  {
    deviceType = ROUTER;
  }
  else
  {
    deviceType = END_DEVICE;
    appFlags |= (appLowPwrF | appSetPollF);  /* flags set for end device type*/
  }

  /*Set Pan ID */
  //setPanId(MYPANID);

  /*Set ZigBee Device Type */
  setZigbeeDeviceType(deviceType);

  /*Register application endpoint for either router or end device*/
  sapiRegisterApplication(srceEP());

}
```

2. **Re-build and download the Sensor Monitor Network Sample Application**

3. **Start the Network**
   - Follow the steps described in Lab2 Example 1 step 5

4. **Add the Source Device – Router to your Network**
   - Follow the steps described in Lab2 Example 1 Step 6

5. **Add the Source Device – End Device to your Network**
   - Follow the steps described in Lab2 Example 1 Step 7

6. **Observe the network behavior and note down network parameters**
   - For this exercise note the following two parameters
     - PANID
     - Device short Address
   - Also Refer to the **ZNP Interface Specification document** (included in Z-Stack Install at <install - directory>\Texas Instruments\ZStack-CC2530-2.5.0\Documents\CC2530) **Section 4.2 (4.2.3 and 4.2.4)** for understanding the configuration parameters for Zigbee Network – Device Specific and Network Specific.

7. **Power Cycle all the devices**
   - Power-up the coordinator device first
   - Then bring up the other network devices one by one

8. **Observe the network behavior**
   - The devices after power cycle restore their network parameters and report the sensor data to the sink periodically as if they never left the network.
   - Binding is not required after power cycle with NV_RESTORE enabled.
   - Route requests commands can be seen as the network tries to find the path to the sink from each device.

9. **Questions**
   a. Compare the network behavior between the two labs – Lab 1 and Lab2
   b. What commands do we see over the air in this example?
   c. What are device specific and network specific configuration parameters?

LAB 4: Form a Large Zigbee Network

Objective

- In this lab you will learn important Zigbee Concepts related to building and commissioning Large ZigBee Networks
- Attendees will learn how to use Z-Tool for application Development and Debugging

Following are the steps for this Lab 3 Example.

1. **Assign channel and Pan1d**

2. **Build ZigBee Nodes**
   - Each group will build two nodes
   - Before building and downloading the code on to MSP430F2274, **Compile Option ZTool and HOST_MT have to be enabled**.
   - While assembling the hardware one node – Router will be connected to the USB stick, while the end device be battery powered

3. **Instructor will be the coordinator**

4. **Configure and Start ZTOOL**

Follow the steps below to interface the Z-Tool with CC2530ZNP Mini Kit.

1. Connect the Target board connected to the eZ430 USB stick to a USB port on your PC

Figure 1: eZ430-CC2530ZNP Mini Kit board connected to the PC

2. Identify the COM port number on the PC of the connected eZ430-CC2530ZNP Mini Kit.
   - Open -> Control Panel -> System -> (Tab) Hardware -> Device Manager

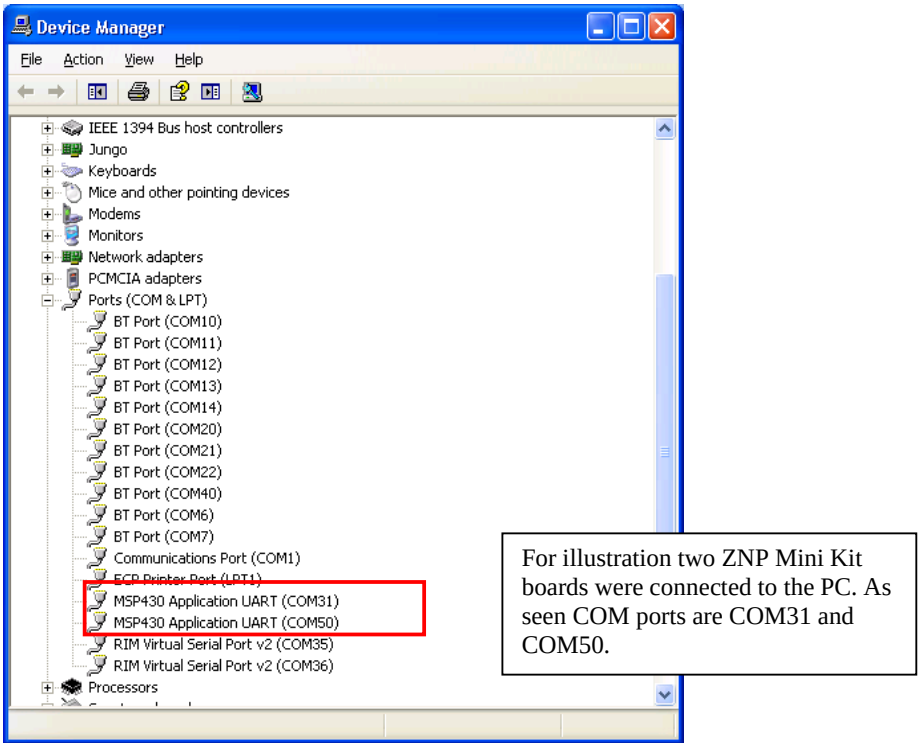
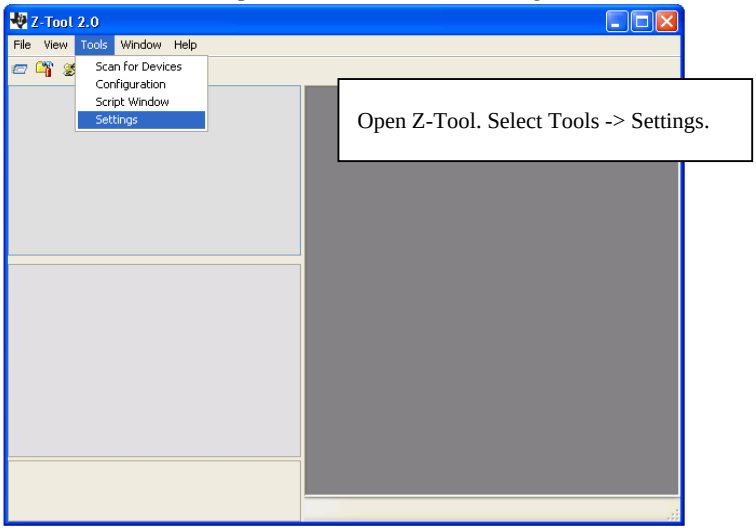
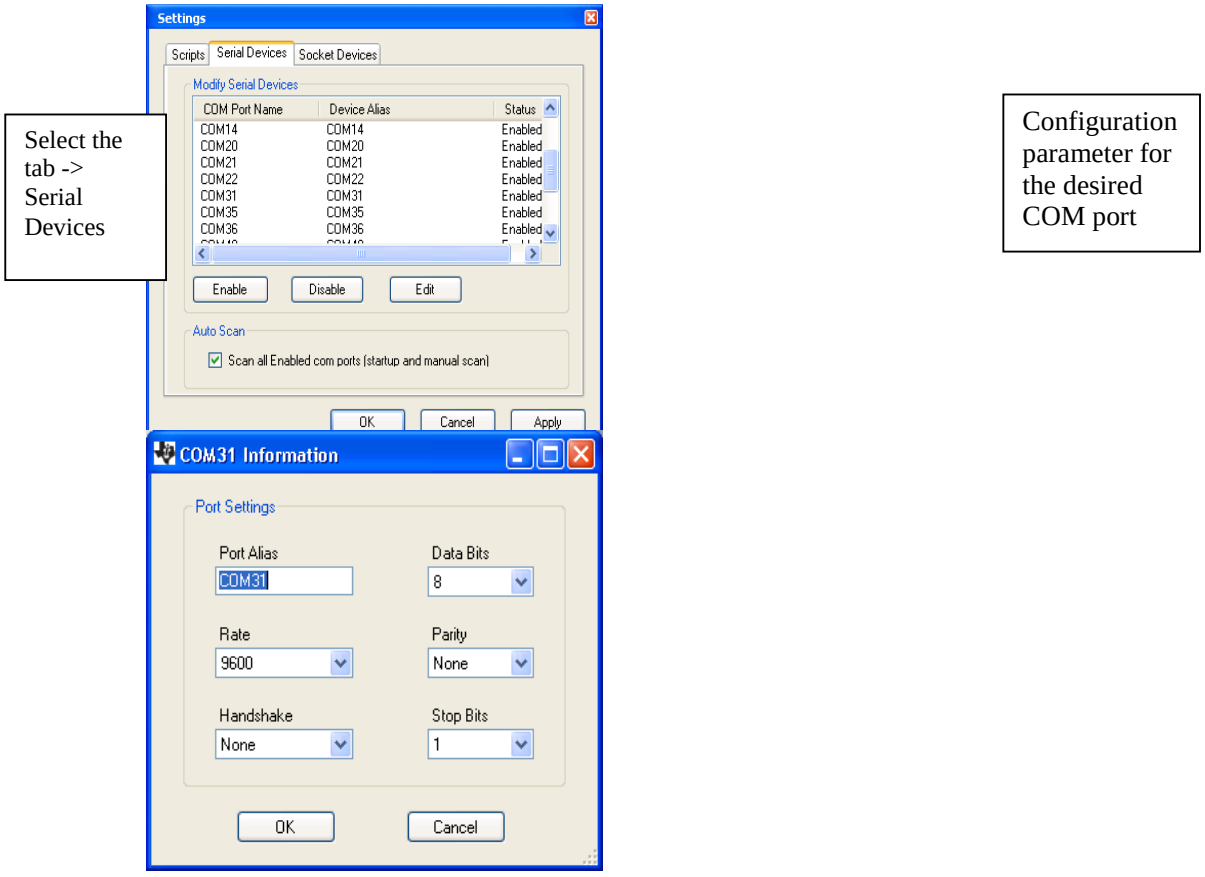


Figure 2: Device Manager View of the PC to identify COM port number of connected devices

3. Open Z-Tool. For the identified devices (ex- devices on COM31 and COM50 in this example) configure the communication interface parameters as shown in the figure 3 below



(a)



(b) (c)

Figure 3: Steps illustrating the how to configure the configuration parameters for the COM port for communication with Z-Tool application.

4. Close Z-Tool and start again
5. Z-Tool should now be able to recognize the CC2530ZNP and window as shown in figure 10 should appear.

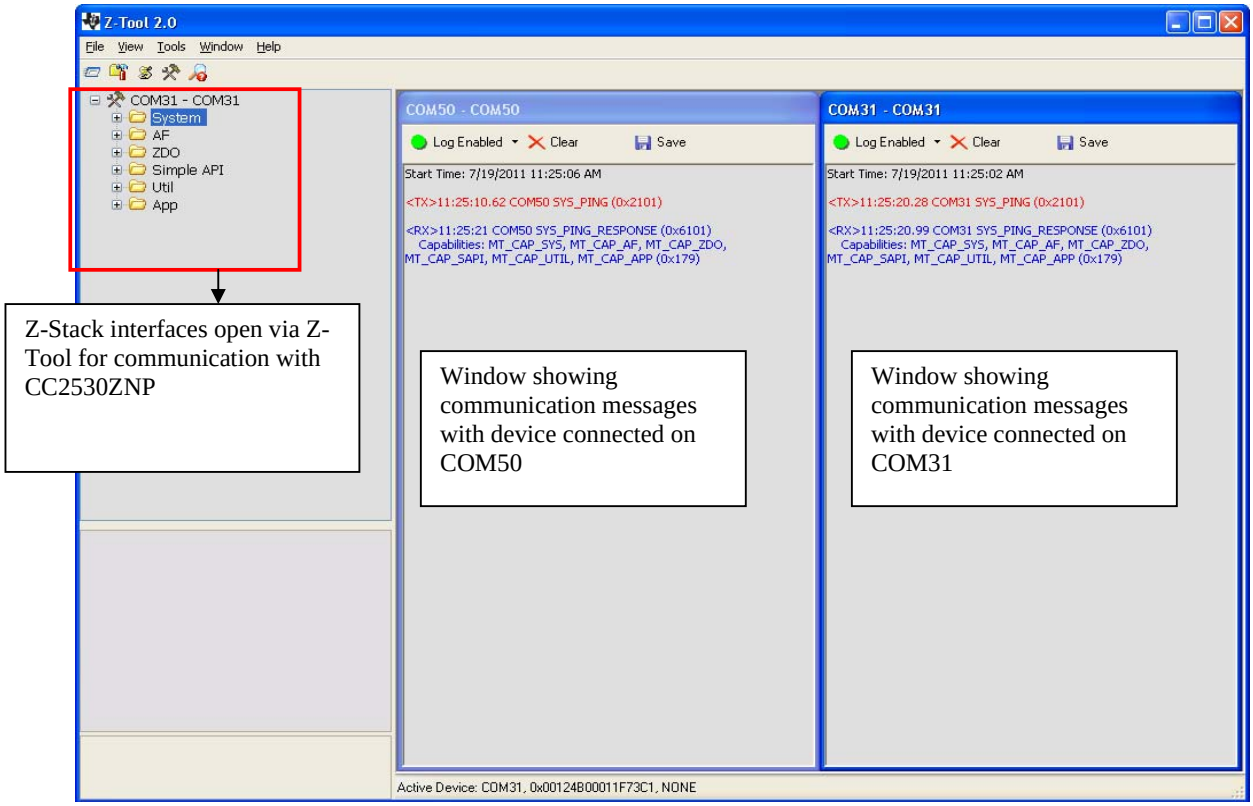


Figure 4: Z-Tool Application showing two devices connected and exposed interfaces – System, AF, ZDO, Simple API, Util, and App on the CC2530ZNP on the CC2530ZNP Mini Kit.

5. Start the nodes one by one

- Observe the network on the instructors screen.
- Understand the process of:
  - Network joins
  - Routing
  - Device Associations/Binding

6. Use ZTOOL to issue commands

- **Exercise 1:** IEEE Address request – Query coordinator for its IEEE Address
  - Device address 0x0000 for coordinator is sent to the ZNP which returns the SRSP indicating success. This implies the command was successfully received by the ZNP. The ZNP then sends the command OTA and when it receives the response message back from the queried device, it sets the SRDY low to interrupt the MSP430. MSP430 then receives the ZDO_IEEE_ADDR_RSP message and sends it to Z-Tool application as seen in figure 5 below.

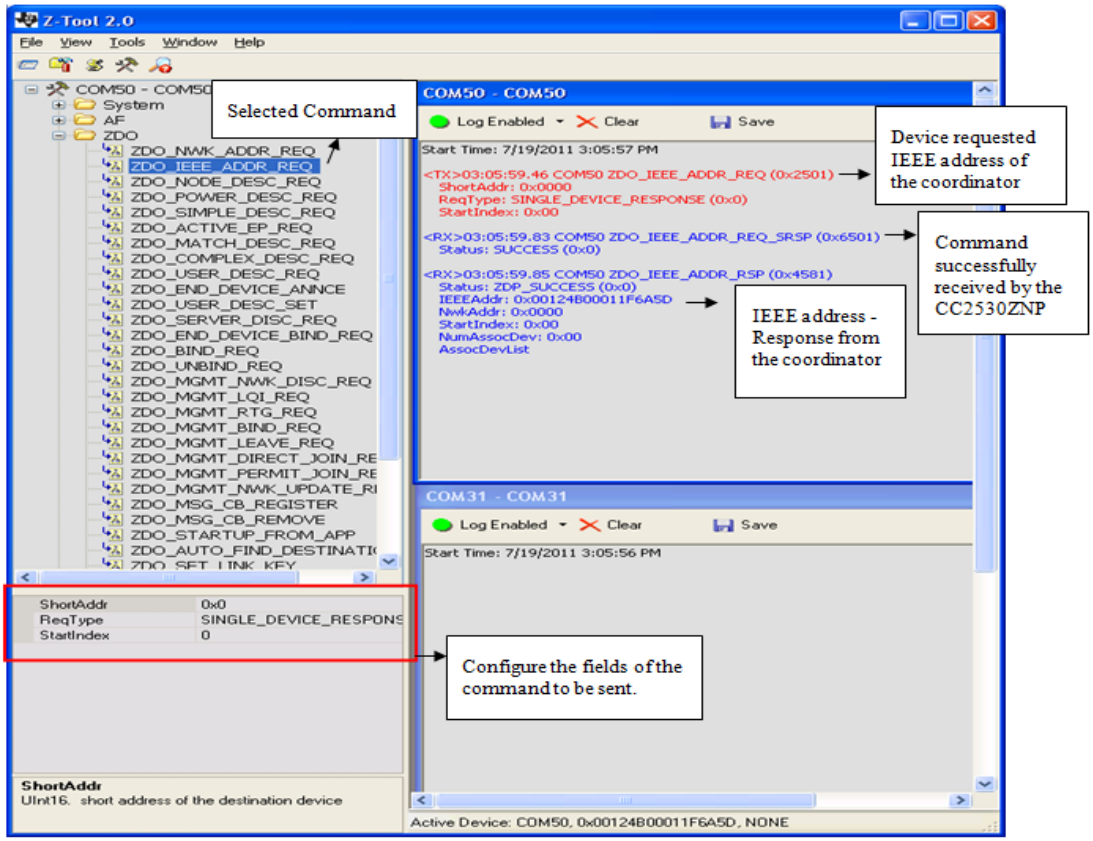


Figure 5: Z-Tool Application window with two connected devices on COM50 and COM31. Device on COM50 requested IEEE address from the device at address 0x0000 (coordinator) and gets the response back of type IEEE_ADDR_REQ_SRSP.

- **Exercise 2:**
  - Short Address – find the short address of the end device in your group,
  - Send a three byte message [11, 12,13] to the End Device

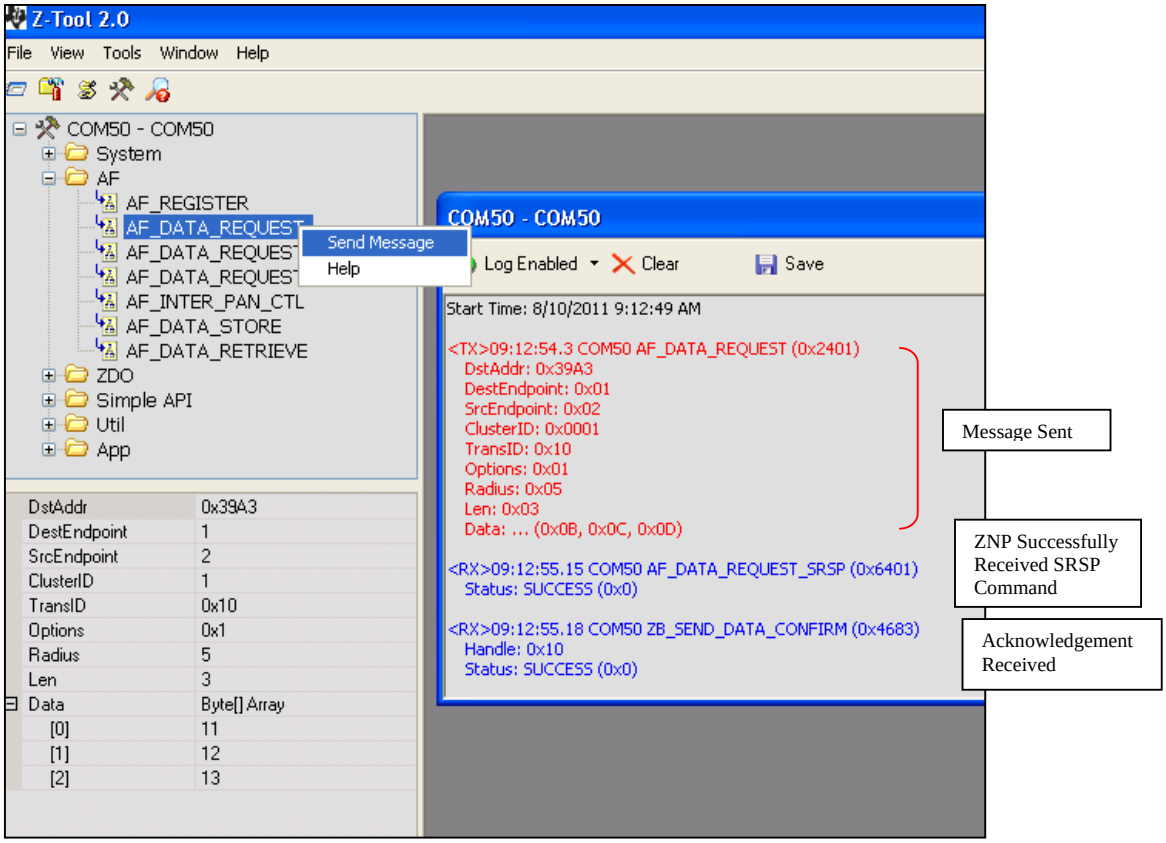


Figure 6: Z-Tool Screen Capture After sending a 3 byte long message to the device with Short Address = 0x39A3.

7. Questions

1. What commands do we see over the air in this example?
2. How can we make a node and all its children leave the network?

You're done.

Document History

Revision	Date	Description/Changes
1.0	8/11/2011	Initial release.