Application Report

SWRA444–January 2014

Developing a ZigBee® System Using a CC2530-ZNP Approach

Abhed Misra — *WTBU-Low-Power Wireless*

ABSTRACT

This application note describes how to use the CC2530-ZNP approach for developing a ZigBee-compliant system. The CC2530-ZNP (ZigBee Network Processor) is a reliable, quick and simple approach for developing a ZigBee system. In this approach the ZigBee-compliant networking and communication is handled by TI Z-Stack™. The host MCU is responsible for all the Z-Stack configuration and data handling (command and response) over SPI or UART.

Contents

List of Figures

List of Tables

1 Introduction

Z-Stack is a ZigBee certified stack from TI available at the TI website for free download. It is running on the ZigBee system-on-chip radio-CC2530. TI Z-Stack with combination of CC2530 is a ZigBee certified and compliant platform by ZigBee alliance listed on the ZigBee organization website too. In this document we will discuss in detail the hardware and software setup for the ZigBee system development composing CC2530 and a host MCU. The reader is expected to have a basic fundamental understanding of ZigBee standard and ZigBee network entities. In this example I have used the CC2530 (ZigBee SOC) and CC2591 (Radio Front End-Power Amplifier) based LPRF module as ZigBee Radio and value line MSP430 as a Host MCU.

2 System Requirements

2.1 Hardware

Table 1. Hardware Tools and Functional Description

Device	Tool	Function
CC2530 and CC2591 Low Power RF Module	CC2530-CC2591EMK	This tool is flashed with TI Z-Stack. It takes care of the Physical, MAC and networking layer of the ZigBee Network.
MSP430G2553 value line MCU	MSP430G2553IPW20	This MCU acts as a HOST in the example and does the ZigBee network parameter configuration and data handling over the network too.
MSP430 value line Launchpad	MSP-EXP430G2	This hardware platform is used for host-side application development on MSP430G2553.
Hardware interface between CC2530 and MSP430	BOOST-CCEMADAPTER	This acts as the hardware interface between the CC2530 serial port and Host MCU.
Sniffer Tool	CC2531EMK	This tool is used along with the Packet sniffer software tool to sniff and analyze the communication over the air between the radios.

2.2 Software

Table 2. Software Tools and Functional Description

Software	Role	Function
CC2530 and CC2591 Low Power RF Module	TI Z-Stack	The Z-Stack is flashed on to the CC2530 SOC.
MSP430G2553 value line MCU	Stack configuration and Application Code	This software runs from Host MCU and configures the Z-Stack over UART. The Host configures Z-Stack for all the necessary and relevant parameters and also manages the data communication over the network.
Compiler IDE (IAR or CCS)	Development Environment	This IDE tool is used to develop, debug and compile the application code and Z-Stack code.
Sniffer Tool(TI-packet Sniffer or Ubiqua)	PC tool for sniffing the over-the-air ZigBee packets	This tool is used to sniff and analyze the communication packets over the air between the radios.

3 Design Procedure

3.1 Hardware Configuration

In the ZNP configuration/approach of Z-Stack, the connection interface between CC2530 and Host MCU can be SPI/UART/USB. We will be using the UART approach as an example. In Figure 1, the connection setup details are mentioned. The CCEMADAPTER mounts on the MSP430 Launchpad and the CC2530-CC2591EM can be mounted on the adapter connecting to MSP430. The pin out connections are shown in Table 3.

Table 3. Device Pin Out Description

Function	MSP430G2553IPW20			Function	CC2530	
UART-RXD	Port 1.1	Pin 2		UART-RXD	Port 0.2	Pin 17
UART-TXD	Port 1.2	Pin 3		UART-TXD	Port 0.3	Pin 16
Supply	VCC	Pin 1		Supply	VCC	Pin 10
Ground	GND	Pin 20		Ground	GND	Pin 41

The pin connection details of SPI/UART/USB are given in document *CC2530ZNP Interface Specification*. This document also has a comprehensive set of API commands defined for configuring/communicating between host MCU and CC2530.

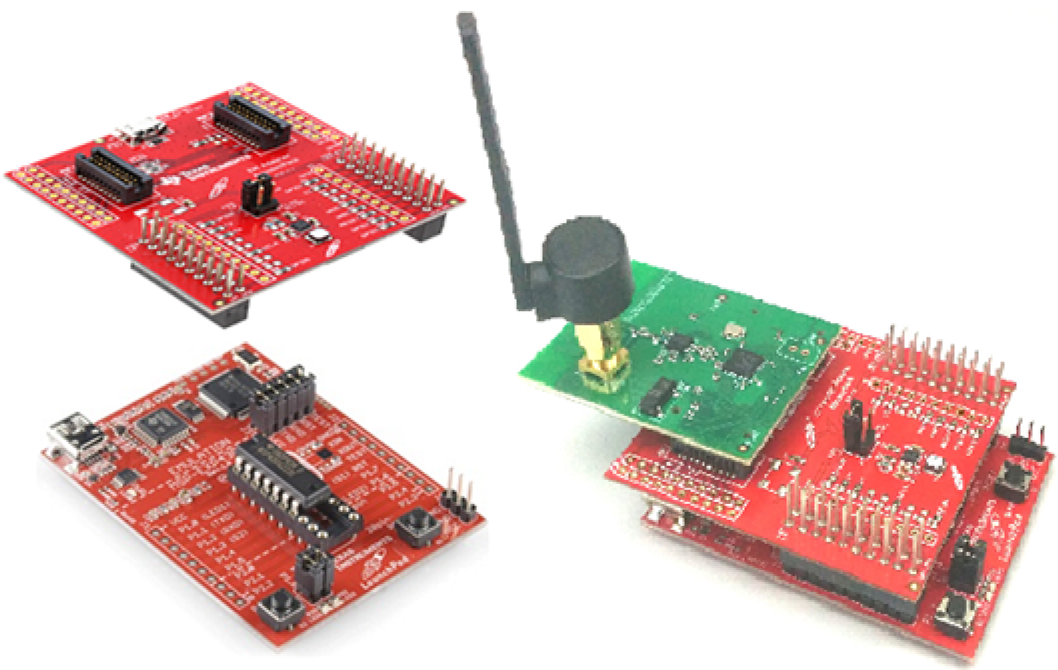

Figure 1. Hardware Connections Setup

The various steps along with command and response sets used for configuring the Z-Stack to construct a ZigBee System follow:

STEP 1: Connecting the CC2530 radio on UART of host MCU as per connections suggested in Table 3 and Figure 1.

STEP 2: For changing the baud rate and configuring other parameters in Z-Stack following steps need to be followed.

STEP 2.1: In the folder where you have installed the Z-Stack on your PC you can find the workspace of Z-Stack:

\Texas Instruments\ZSTACK-CC2530-2.5.0\Projects\ZSTACK\ZNP\CC253x

Open the workspace in IAR-8051, and select the respective project as per your device used. That is if you are using CC2530 then select the CC2530 workspace as shown in Figure 2.

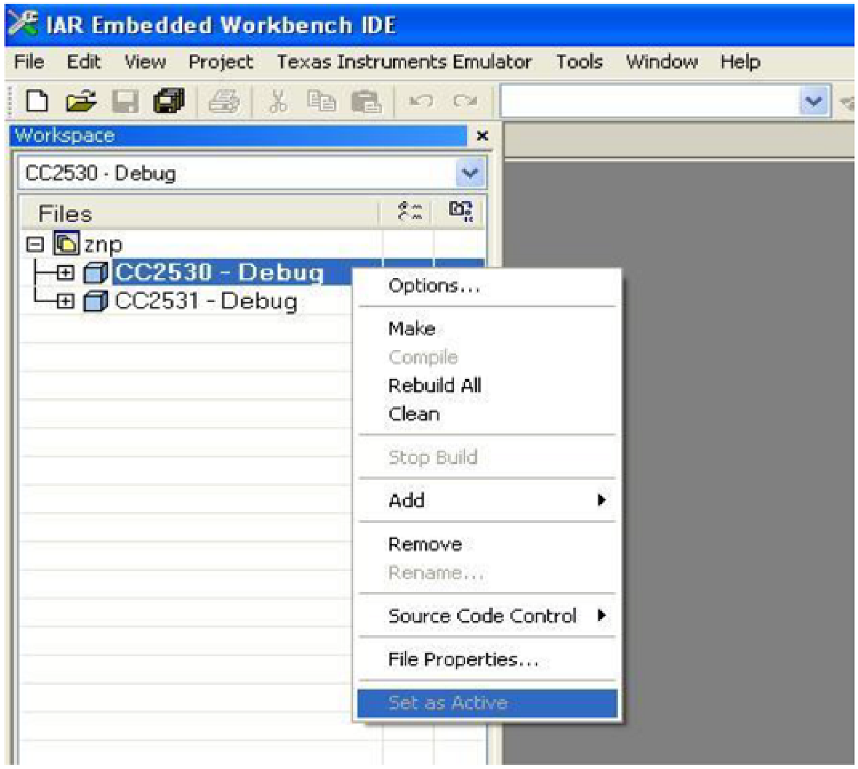

Figure 2. Workspace Lookup in IAR IDE

STEP 2.2: Open the file named “f8wconfig.cfg”. This is the configuration file for the . Ensure the following items in f8wconfig.cfg:

1. **DZIGBEEPRO** is enabled. This will enable the ZigBee Pro features in Z-Stack.
2. **DSECURE** is equal to 1. This enables the security in ZigBee. On enabling the security in ZigBee the network association, authentication and formation will only be possible if the Radio Device will have the correct TRUST CENTRE LINK KEY(TC LINK KEY) and NETWORK KEY.

   Note: The Security in Z-Stack can also be enabled thru compile options, by mentioning ‘SECURE=1’ in the preprocessor as shown in the following paragraph.

In Z-Stack the NETWORK KEY is defined in f8wconfig.cfg by name of -DDEFAULT_KEY. I have configured the security key as “ZIGBEE” thru the ZNP command. If needed you can change the DEFAULT_KEY in the “f8wconfig.cfg” also.

In Z-Stack the Trust Center LINK KEY is defined in “nwk_globals.h” as DEFAULT_TC_LINK_KEY.

STEP 2.3: Now for changing the baud rate of UART, open the file “znp.cfg”. You will see that the default value of ‘-DZNP_UART_BAUD’ is HAL_UART_BR_115200. For 9600 change this statement to ‘-DZNP_UART_BAUD=HAL_UART_BR_9600’.

STEP 2.4: To retain all the configurations made in network parameters of radio device during commissioning, you will have to compile the stack with one more compile option of ‘NV_RESTORE’. This compile option configures the stack for storing all the commissioned parameters of the radio device in non-volatile memory and reload at the time of initialization. This option can also be configured using the ZNP command as shown ahead in the document.

STEP 3: Now compile this Z-Stack and program the CC253x device on your application board.

STEP 4: For configuration of network radio device, the following set of commands in the respective order will be sent by HOST MCU on UART to CC2530. All these commands are described in detail in the CC2530 ZNP Interface Specification document.

NOTE: The DATA section is made GREEN, and the COMMAND section is made white.

Table 4. Abbreviations

Abbreviations for Command and Response	
SOF	Start of Flag
Len	Length
Cmd-0	Command ID 0
Cmd-1	Command ID 1
CRC	Cyclic Redundancy Check
S/W	Software
H/W	Hardware
Rev	Revision
NOIC	Number of Input Clusters
NOOC	Number of Output Clusters
APID	Application Profile Identification
EP	End Point
Ack	Acknowledgment
Nack	Non-Acknowledgment

COMMAND-1: SYS_RESET_REQ

FE	01	41	00	00	40
SOF	**Len**	**Cmd-0**	**Cmd-1**	**Type**	**CRC**

RESPONSE-1:

FE	06	41	80	02	02	00	02	05	00	C0
SOF	**Len**	**Cmd-0**	**Cmd-1**	**Reason**	**Transport ID**	**Product ID**	**Major Release**	**Minor Release**	**H/W Rev**	**CRC**

REMARK: This command ensures the proper reset of the radio and brings the radio in configuration mode.

COMMAND-2: ZB_WRITE_CONFIGURATION -> ZCD_NV_STARTUP_OPTION ->
STARTOPT_CLEAR_STATE

FE	03	26	05	0.3	01	02	20

RESPONSE-2:

FE	01	66	05	00	62

REMARK: The CC2530-ZNP device has two kinds of information stored in non-volatile memory: The configuration parameters (listed in this section) and network state information. The configuration parameters are user configured before starting the ZigBee operation. The network state information is collected by the device after it joins a network and creates bindings, and so forth (at runtime). This is not set by the application processor. This information is stored so that if the device were to reset accidentally, it can restore itself without going through all the network joining and binding process again.

We have configured the radio to *clear the network state at every power up*.

COMMAND-3: SYS_RESET_REQ

FE	01	41	00	00	CRC

RESPONSE-3:

FE	06	41	80	02	02	00	02	05	00	**CRC**

REMARK: This command ensures the proper reset of the radio and brings the radio in configuration mode.

COMMAND-4: ZB_WRITE_CONFIGURATION ->ZCD_NV_PANID

FE	04	26	05	83	02	FF	FF	CRC

RESPONSE-4:

FE	01	66	05	00	CRC

REMARK: This command configures the PAND ID in the **Z-Stack** to be used by the radio device. The device in ROUTER configuration starts the scan of this PAN ID at power up and keeps searching till it joins the network with this PAN ID.

We have configured the Z-Stack with OPEN PAN ID, that is, 0xFFFF.

COMMAND-5: ZB_WRITE_CONFIGURATION ->ZCD_NV_EXTPANID

FE	0A	26	05	2D	08	DD	DD	DD	DD	DD	DD	DD	DD	CRC

RESPONSE-5:

FE	01	66	05	00	CRC

REMARK: This command configures the EXTENDED PAN ID in Z-Stack. The extended pan id is used to further segregate the sub network(s) among a bigger PAN network. We have configured Z-Stack with extended PAN ID of 0xDD, 0xDD, 0xDD, 0xDD, 0xDD, 0xDD, 0xDD, 0xDD.

COMMAND-6: ZB_WRITE_CONFIGURATION -> ZCD_NV_CHANLIST

FE	06	26	05	84	04	03	FF	F8	00	CRC

RESPONSE-6:

FE	01	66	05	00	CRC

REMARK: This command configures the stack for the list of channels to be used. We have configured the Z-Stack to use first 15 channels only; hence the channel mask of 0x03FFF800 is used.

COMMAND-7: ZB_WRITE_CONFIGURATION -> ZCD_NV_LOGICAL_TYPE

FE	03	26	05	87	01	01	CRC

RESPONSE-7:

FE	01	66	05	00	CRC

REMARK: This command configures the LOGICAL TYPE of the radio device. When the device will start the application then will emerge as ROUTER, and will join the network as ROUTER, only. The device can be configured in any of the 3 LOGICAL TYPES: Coordinator, Router, and END Device.

COMMAND-8: ZB_WRITE_CONFIGURATION -> ZCD_NV_PRECFGKEY

FE	12	26	05	62	10	16 bytes Long Network Key	CRC

RESPONSE-8:

FE	01	66	05	00	CRC

REMARK: This command is used to change the network key of the radio device. This network key can be configured thru ‘f8wconfig.cfg’ file also.

COMMAND-9: ZB_WRITE_CONFIGURATION -> ZCD_NV_PRECFGKEYS_ENABLE

FE	03	26	05	63	01	01	CRC

RESPONSE-9:

FE	01	66	05	00	CRC

COMMAND-10: ZB_WRITE_CONFIGURATION -> ZCD_NV_TC_LINK_KEY

FE	24	21	09	01	01	00	20	8 bytes of 0xFF	16 bytes Long Trust Center Link Key	8 bytes of 0x00	CRC

RESPONSE-10:

FE	01	61	09	00	CRC

COMMAND-11: AF_REGISTER

FE	11	24	00	08	0D	BF	01	05	01
SOF	**Len**	**Cmd-0**	**Cmd-1**	**EP**	**AP ID-0**	**AP ID-1**	**App Device ID-0**	**App Device ID-1**	**App Device Version**

00	04	00	00	15	00	02	07	XX	XX	00	CRC
Latency	**NOIC**	**Basic Cluster**		**Commissioning Cluster**		**Simple Metering Cluster**		**Manufacturer Specific Cluster**		**NOOC**	

RESPONSE-11:

FE	01	64	00	00	CRC

REMARK: This command is used by the Router to register the application with the coordinator, basically indicating what clusters it supports. Here we are registering

a. Basic Cluster.(0x00)
b. Commissioning Cluster.(0x0015)
c. Simple Metering Cluster.(0x0702)
d. User Specific Cluster.(XXXX)

COMMAND-12: ZDO_STARTUP_FROM_APP

FE	02	25	40	00	00	**CRC**

RESPONSE-12: Part-1

FE	01	66	40	01	25

RESPONSE-12: Part 2: ZDO_STATE_CHANGE_IND

FE	01	45	C0	02	86

REMARK: The value 0x02 indicates that the device is discovering PAN's to join.

RESPONSE-12: Part 3: ZDO_STATE_CHANGE_IND

FE	01	45	C0	05	86

REMARK: The value 0x05 indicates that the device has joined but not yet authenticated by the trust center.

RESPONSE-12: Part 4: ZDO_STATE_CHANGE_IND

FE	01	45	C0	07	86

REMARK: The value 0x07 indicated that the device has joined, authenticated and is a Router.

In case of Coordinator, the response is as follows:

RESPONSE-12: Part 1: ZDO_STATE_CHANGE_IND

FE	01	45	C0	09	86

REMARK: The value 0x02 indicates that the device has started the PAN successfully.

The point to be noted here is that there can be multiple *ZDO_STATE_CHANGE_IND* responses from ZNP on UART in case of coordinator functionality, till the PAN is not formed which is ultimately confirmed by 0x09 in *ZDO_STATE_CHANGE_IN*. In the case of router functionality the *ZDO_STATE_CHANGE_IND* will continue to send the 0x02 or 0x05 or even RESET indication on UART until it gets associated and authenticated by a coordinator in a network.

After the successful configuration and startup of the device and forming and joining the network, the nodes can exchange the data on the network. To send the data command over the network and receive the data response thru network, the ZNP approach provided in the following mentions command and responses.

COMMAND: AF_DATA_REQUEST

0x0F	24	01	27	C0	08	08	34	12	00
Len	**Cmd-0**	**Cmd-1**	**Destination Short Address**		**Destination EP**	**Source EP**	**Cluster ID**		**Trans ID**

80	0F	05	00	02	00	00	00	CRC
Ack Request Type	**Radius**	**Data Length**	**Data Packet/Payload**					

RESPONSE:

01	64	01	00	CRC

REMARK: In this the host MCU of coordinator (Source Short Address: ‘0x0000’) is instructing the radio to send data (’0x00, 0x02, 0x00, 0x00, 0x00’) to the Destination router (Short Address: ‘0xC027’). In case of router pinging the parent, as done in the workspace also the destination address will be 0x0000 (short address of the coordinator).

COMMAND: AF_DATA_INCOMING

0x0F	44	81	xx	xx	34	12	00	00	08	08
Len	**Cmd-0**	**Cmd-1**	**Group ID**		**Cluster ID**		**Source Address**		**Source EP**	**Destination EP**

xx	xx	xx	xx	xx	xx	xx	xx	xx	**Data Packet/Payload**	CRC
Was Broadcast	**LQI**	**Security Use**	**Time Stamp**				**Len**	Seq No		

RESPONSE:

01	64	01	00	CRC

REMARK: In this the host MCU of router (Source Short Address: ‘0x0000’) is getting an incoming message from ZNP which was sent by coordinator over the network as a response of the data command sent by the router.

Figure 3 shows the synopsis of entire UART and over the air communication.

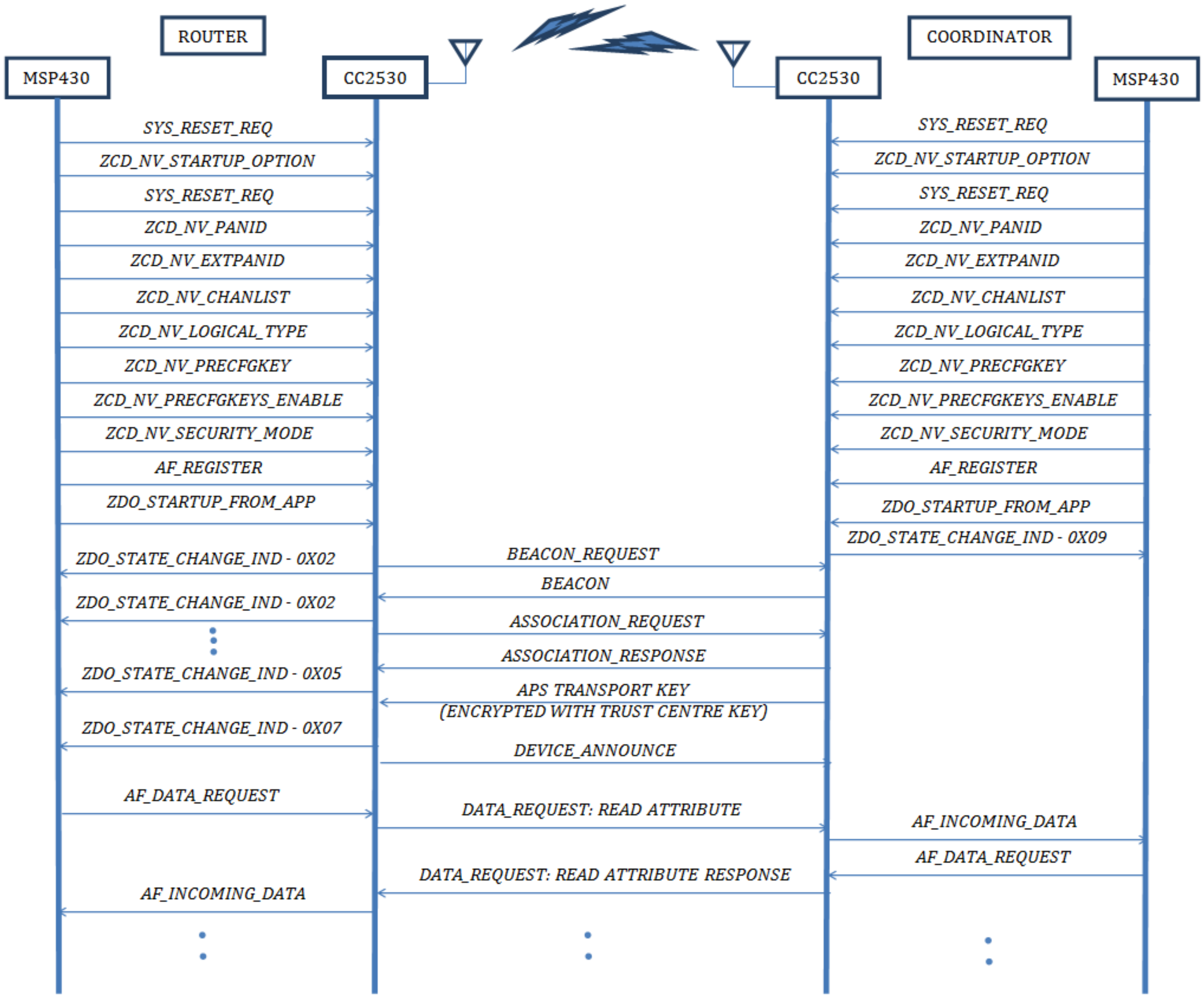

Figure 3. Communication Between Host-Radio and Coordinator-Router

4 Application

The application code on MSP430 (Host) performs the following functions:

1. Configures the Z-Stack on CC2530 thru UART for all the ZigBee parameters.
2. Exchanges DATA packets over the network thru CC2530 radio.
3. Pings parent (coordinator) of the network for its presence periodically. If the parent doesn't respond it restarts itself and again tries for a parent/network to join. When no parents are available, the Host enters in a fault state.

During configuration of Z-Stack on CC2530, the application blinks LED1 on the MSP430 Launchpad as an indication. On successfully joining the parent or network, LED1 becomes still as shown in Figure 4.

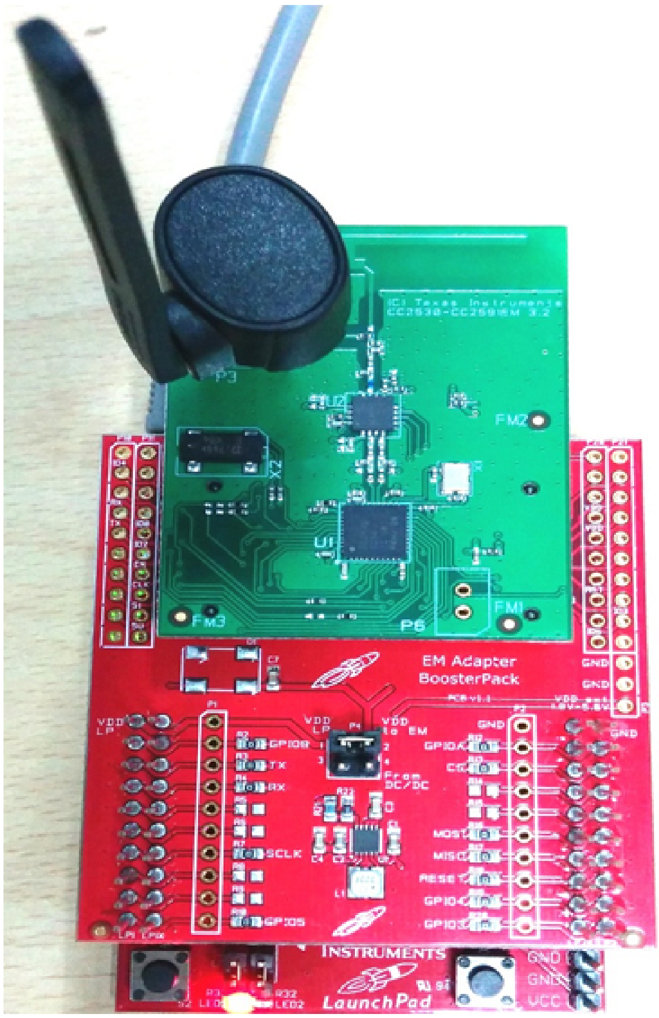

Figure 4. LED1 Indicating the Device Joined the Network

LED2 starts blinking slowly to indicate the parent pinging by the router. In case the parent goes OFF or disappears the application again restarts the configuration of Z-Stack after resetting the CC2530. If the router is not able to find any parent then LED1 is switched OFF and LED2 starts blinking fast indicating the FAULT state, as shown in Figure 5. The application is written in such a way that the user can integrate their own user-specific cluster and have different attribute data exchanged over the air.

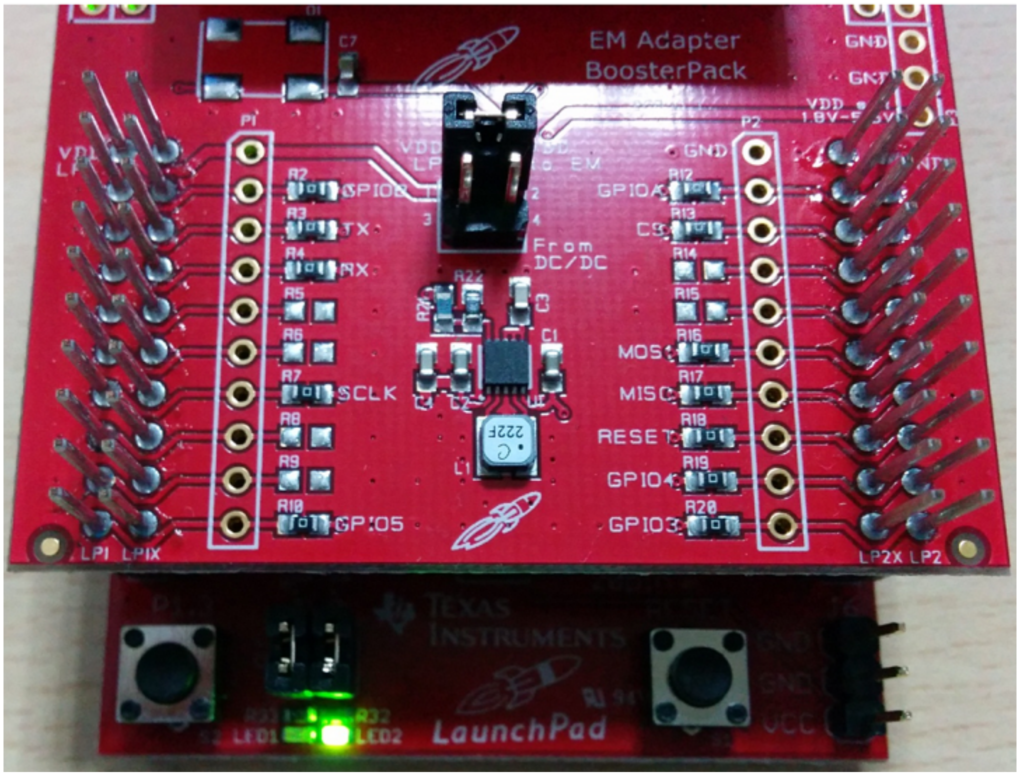

Figure 5. LED2 Indicating the Fault in Router Joining the Network

The application code can be completely ported on any general purpose MCU.

5 Test Tools

To test the desired configuration and communication between the Host MCU and CC2530 we used the third party serial communication port sniffers. The following tools were used for the over-the-air communication:

1. Texas Instruments: Smart-RF Packet Sniffer: Packet Sniffer
2. Ubilogix: Ubiqua Protocol Sniffer

As an example, a screenshot of Ubiqua is illustrated in Figure 6, showing the data communication happening between Coordinator and Router.

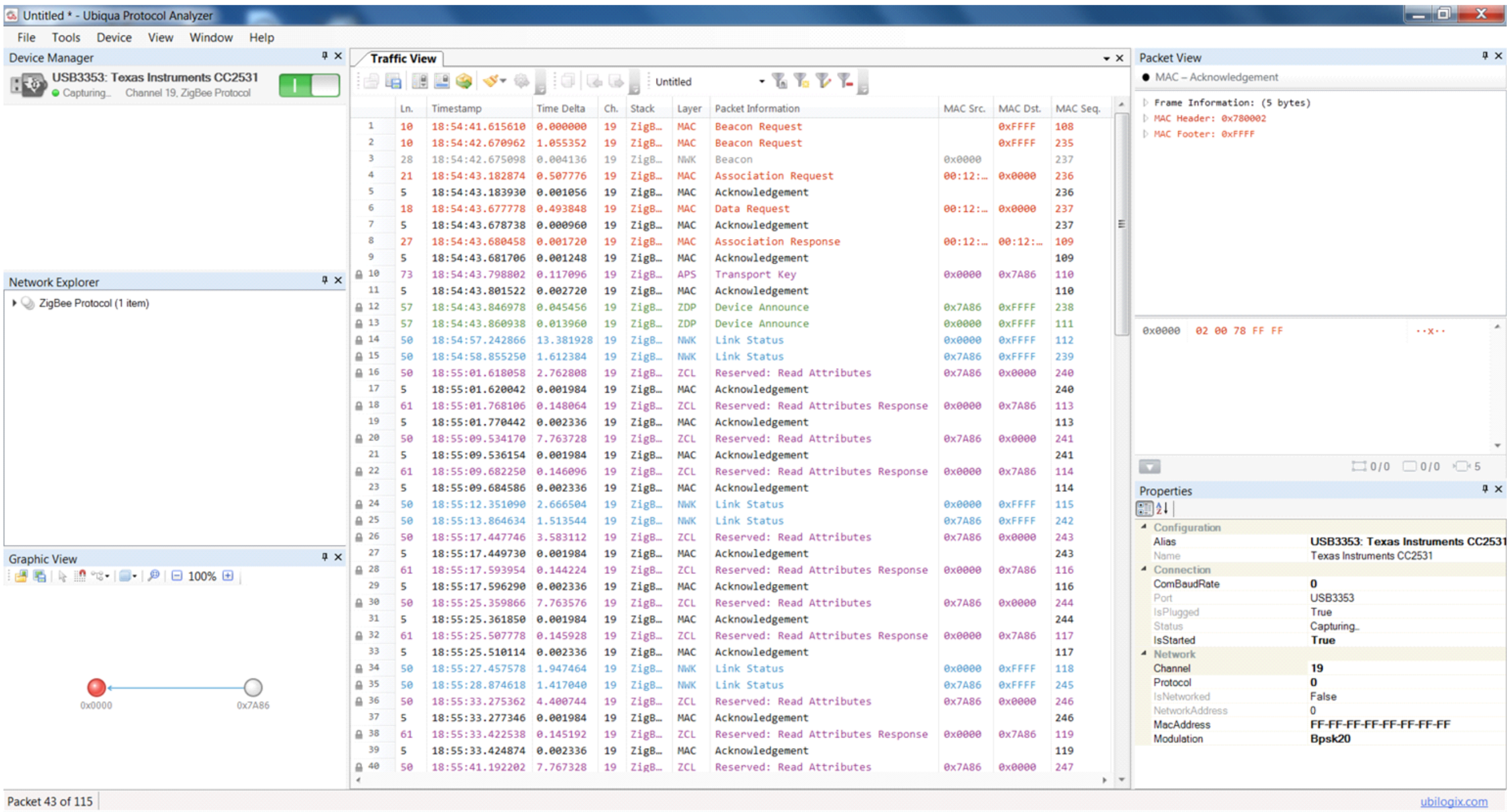

Figure 6. UBIQUA Sniffer Log of the ZigBee N/W Communication

6 References

1. CC2530: Second Generation System-on-Chip Solution for 2.4 GHz IEEE 802.15.4 / RF4CE / ZigBee Data Sheet (SWRS081B).
2. MSP430 Value Line Launchpad Development Kit White Paper (SLAY017).
3. EM Adapter Booster Pack User's Guide. (SWRU338A).
4. Ubiqua: Your toolbox for sensor networks (www.ubilogix.com).
5. CC2530ZNP Interface Specification document.
6. IEEE std. 802.15.4 – 2006: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs) (http://standards.ieee.org/findstds/standard/802.15.4-2006.html)
7. ZigBee Standard Specification. (https://www.zigbee.org/Standards/Downloads.aspx)