MiWi Library

Microchip Libraries for Applications (MLA)

Table of Contents

MiWi Library

1 MiWi Library

1

1.1 Introduction

MiWi™ Development Environment with MiMAC and MiApp Interfaces

MiWi™ Development Environment (MiWi™ DE) is developed by Microchip to support a wide range of wireless applications. The backbone of MiWi™ DE is MiMAC and MiApp interfaces, which link the support of multiple RF transceivers as well as wireless communication protocols together as a well-defined simple but robust Microchip proprietary wireless development environment.

Within MiWi™ DE, application developers are able to switch between RF transceivers and wireless protocols with little or no modification in the application layer. By providing such easy migration capability in MiWi™ DE, as well as simple but robust interfaces, the firmware development risk has been reduced to a level that has never been observed in the industry before.

MiWi™ DE is defined in three layers: application layer, protocol layer and RF transceiver layer. The three layers are linked together by MiMAC and MiApp interfaces. Application layer uses MiApp interfaces to talk to the protocol layer. In protocol layer, there are implementations of MiWi™ P2P and MiWi wireless communication protocols available. The drivers for Microchip RF transceivers (MRF24J40, MRF49XA and MRF89XA for this release) are called by protocol layers via MiMAC interfaces. Configuration files are also presented in each layer. Following diagram shows the architecture of Microchip MiWi™ DE.

The details of the major modifications can be found in the Release Notes.

1.2 Legal Information

1.3 Release Notes

Microchip MiWi(TM) Development Environment Software Stack * Version 5.15.0, August 2015

Microchip MiWi™ Development Environment (MiWi™ DE) protocol stack provides simple wireless connectivity for short-range, low data rate and low power applications. Microchip MiWi™ DE protocol is royalty free as long as implemented on Microchip PIC microcontroller and radio frequency transceiver. Please refer to the attached license agreement for details.

The MiWi™ DE source code is released with applications to demonstrate communications between two RF devices. The source code for each device is located in individual directories under "apps/miwi". In addition, the directory "miwi" is for MiWi™ stack source code.

The main updates from earlier versions are:

- Support for latest versions of XC8, XC16 compilers.
- All the MiWi features in the last MiWi Release 5.10.0 are also supported in this Release

Resource Usage

Peripherals

Type/Use	Specific/Configurable	Limitations
UART for hyper-terminal/Teraterm output	Select via pin definitions in system_config.h	None
Timer for protocol timing	16bit Timer	Timer is preferred to be configured to represent 16us for one tick
SPI for RF transceiver	Select via pin definitions in system_config.h	Both hardware SPI or software big-bang can be used.
Digital I/O pins to RF transceiver	Select via pin definitions in system_config.h	Must be able to be configured as external interrupt pin or interrupt-on-change pin; must have a pull-up

Limitations

1. Microchip C18, C30, XC32 compilers are no longer supported with this release. If support is required for XC32, please contact us.
2. High data rate for MRF49XA may require MCU running at faster speed. This is due to the nature of 16-bit RX buffer used in MRF49XA.
3. MRF24XA under proprietary mode supports data rates up to 2 Mbps and while using higher data rates it is advised to run MCU at faster speed.

1.3.1 Online References and Resources

This section includes useful links to online MiWi development resources.

Description

Note: Newer versions of the documents below may be available. Please check www.microchip.com/miwi for the latest version.

MiWi Design Center

http://www.microchip.com/miwi

User Guides and Reference Documentation

MiWi® Demo Kit – User's Guide (2.4 GHz MRF24J40, 868 MHz MRF89XA, 915 MHz MRF89XA)

8-bit Wireless Development Kit User's Guide

Application Notes

AN1284 – Microchip Wireless (MiWi) Application Programming Interface - MiApp

AN1283 – Microchip Wireless (MiWi) Media Access Controller - MiMAC

AN1066 – MiWi Wireless Networking Protocol Stack

AN1371 – Microchip MiWi PRO Wireless Networking Protocol

AN1204 - Microchip MiWi P2P Wireless Protocol

1.4 Library Interface

1.4.1 Configuring the Library

MiWi(TM) Development Environment uses configuration files to regulate the behavior of the stack. There are three layers of configurations in application, protocol stack and RF transceivers respectively.

1.4.1.1 Application

Configuration in application layer defines the basic functionality of the wireless node. Usually, those configurations may differ among different wireless nodes in the same application, depending on the wireless node's role in the network and application.

Configurations in application layer include following categories:

* Choice of wireless protocol

* Choice of RF transceiver

* System Resources Definitions

* Enable/Disable functionalities according to application needs

* Application specific information

1.4.1.1.1 EUI_0

EUI_0 to EUI_7 defines the Extended Universal Identifier, or permanent address, for the wireless node. The length of the EUI is defined as MY_ADDRESS_LENGTH.

1.4.1.2 Wireless Protocol

Configurations in wireless protocol layer can be used to fine tune the behavior of wireless protocol. The possible configurations differ between different protocols.

1.4.1.2.1 MiWi(TM) P2P Communication Protocol

Following configurations can be used to fine tune the behavior of MiWi(TM) P2P wireless protocol.

1.4.1.2.2 MiWi(TM) Mesh Networking Protocol

Following configurations can be used to fine tune the behavior of MiWi(TM) mesh networking protocol.

1.4.1.2.3 MiWi(TM) PRO Networking Protocol

Following configurations can be used to fine tune the behavior of MiWi(TM) PRO networking protocol.

1.4.1.3 RF Transceivers

Configurations for RF transceivers specifies how RF transceiver work in MiMAC layer. The configurations in this layer may define frequency band, data rate and other RF related parameters. Those configurations differ between different RF transceivers.

1.4.1.3.1 MRF24J40 IEEE 802.15.4 Compliant 2.4GHz Transceiver

Following configurations can be used to regulate Microchip MRF24J40 IEEE 802.15.4 compliant 2.4GHz transceiver.

1.4.1.3.2 MRF24XA IEEE 802.15.4 Compliant 2.4GHz Transceiver

Following configurations can be used to regulate Microchip MRF24XA IEEE 802.15.4 compliant 2.4GHz transceiver.

1.4.1.3.3 SubGHz Transceivers

Microchip SubGHz transceivers share some common configurations.

1.4.1.3.3.1 MRF89XA SubGHz Transceiver

Following configurations can be used to regulate Microchip MRF89XA subGHz transceiver.

1.4.1.3.3.2 MRF49XA SubGHz Transceiver

Following configurations can be used to regulate Microchip MRF49XA subGHz transceiver.

1.4.2 Using API

In this section, MiMAC and MiApp interfaces are defined in detail.

1.4.2.1 MiApp Interfaces

1.4.2.1.1 Call Back Functions

Callback Functions *

MiApp callback functions are called from the protocol stack to application layer. In most of the cases, the callback functions are defined as macros. If developer choose to implement the function, the macro can be commented out and replaced by a function call in the application layer.

1.4.2.2 MiMAC Interfaces

1.5 Advanced Features

Advanced Features

MiWi DE supports advanced features , such as Network Freezer, Enhanced Data Request and Time Synchronization. Some of these features are not documented in the Application Notes. In this section, we describe those new features in detail to help users o understand them. All these features can be enabled in any demo, which is included in the release package.

1.5.1 Network Freezer

Motivation

Occasionally, a wireless network may lose power. After power is restored, in most of the cases, the wireless nodes might form a different network through different joining procedures. This is particularly obvious for MiWi protocol, which uses 16-bit network address in communication. After the power cycle, a wireless node in MiWi network may be assigned with a different network address. As the result, the application layer may have to dedicate more efforts to handle the power cycle scenario. It is important to develop a feature which can release the application layer from handling power cycle.

Solution

Network Freezer feature is developed to solve this problem. It saves critical network information into the Non-Volatile Memory (NVM) and restore them after power cycle. In this way, the application does not need to worry about the power cycle scenario and the network can be restored to the state before the power cycle without any message exchange after the power cycle.

Interface

Network Freezer feature can be enabled by defining ENABLE_NETWORK_FREEZER in configuration file of application layer: ConfigApp.h. In the demo, this feature has been enabled for feature demo.

Network Freezer feature is invoked by calling the MiApp function MiApp_ProtocolInit. The only input boolean parameter bNetworkFreezer indicates if Network Freezer feature should be invoked. When this parameter is TRUE, the network information will be restored from NVM; otherwise, the network information in NVM will be erased and the wireless node start from scratch.

Additional Notes

Network Freezer feature requires NVM to store the critical network information. NVM can be data EPROM in MCU, external EEPORM connected to MCU via SPI, or programming space, if enhanced flash is used in MCU. Choosing the correct form of NVM can be configured in hardware configuration file HardwareProfile.h. The possible options are:

- USE_DATA_EEPROM
- USE_EXTERNAL_EEPROM
- USE_PROGRAMMING_SPACE

For each selection, there are a few minor configurations which can be found in NVM configuration file NVM.h.

1.5.2 Enhanced Data Request

Motivation

In a lot of practical network, most of the devices are sleeping devices, which are connected to a few Full Function Devices (FFDs). Usually, the sleeping devices wake up periodically, asking data from the FFDs and report information back to the network.

For most of the applications, it is critical to provide long battery life for the sleeping devices. A majority portion of power is consumed when the sleeping device is active, asking for data and sending data in the duty cycle. Since the power consumption in active mode is around ten thousand times higher than in sleep mode, lower the total active time plays an important role to prolong the battery life.

Solution

According to IEEE 802.15.4 specification, there are typically three message exchanges after a sleeping device wakes up:

1. Data Request command from the sleeping device to FFD, asking for indirect message from FFD.
2. Indirect message from FFD to sleeping device
3. Message from sleeping device to FFD.

In order to save battery power, we can combine message 1 and 3 together, attach message 3 as payload of message 1. In this way, there are only two messages transmitted, saving the time in CSMA-CA detection and put the device into sleep earlier. Our tests show that the total active time could be lowered 20-30%.

Interface

To enable Enhanced Data Request feature, ENABLE_ENHANCED_DATA_REQUEST must be defined in configuration files for protocols: ConfigP2P.h or ConfigMiWi.h. The reason of enabling this feature in protocol layer instead of application layer is that both sleeping devices and FFDs must enable this feature at the same time. While configuration files in application layer is for each individual devices and configuration file in protocol layer is for every devices, it makes sense to enable/disable this feature in the protocol layer to avoid any mismatch in feature enabling.

There is no special function call for the Enhanced Data Request feature. However, the application function call procedure is different with or without Enhanced Data Request feature.

For applications **without** Enhanced Data Request feature, the procedure after MCU waking up is following:

1. Wake up the transceiver by calling *MiApp_TransceiverPowerState* with parameter POWER_STATE_WAKEUP_DR. It will wake up the transceiver as well as asking FFD for indirect message by sending out Data Request command.
2. Send data from sleeping device to FFD.

For application **with** Enhanced Data Request feature, the procedure after MCU waking up is revised, as shown below:

1. Send data from sleeping device to FFD. However, the data is just queue up in the memory. Actual data is not sent yet.
2. Wake up the transceiver by calling *MiApp_TransceiverPowerState* with parameter POWER_STATE_WAKEUP_DR. It will wake up the transceiver and send Data Request command for indirect message. The data that is sent in step 1 will be the payload of Data Request command. At FFD side, it will handle such message by spliting it into Data Request command as well as the individual message.

The implementation of Enhanced Data Request feature can be found in feature demo in the release package.

Additional Notes

Enhanced Data Request feature can be used to transmit unicast message from the sleeping device to the FFD, but broadcast message still depends on normal message delivery method, because broadcast message and Data Request command may have different destination address.

Because Enhanced Data Request is a brand new feature, ZENA sniffer program has not been updated to decode it. However, this limitation would not affect the operation of the stack.

1.5.3 Time Synchronization

Motivation

In a practical wireless network, large number of sleeping node may be connected to a single Full Function Device (FFD). All sleeping devices can sleep for a while and wake up and request indirect messages in a duty cycle. If multiple sleeping devices wake up around the same time and send Data Request to the FFD, some of those packets may collide and get lost, or have to try multiple times before a positive acknowledgement can be received. This scenario also put burden on the FFD to handle multiple requests almost at the same time.

Solution

To solve this kind of problem, requiring each sleeping device to report in a predefined interval is preferred. This approach is somewhat similar to beacon network which is defined in IEEE 802.15.4. However, beacon network in IEEE 802.15.4 only support star topology and require extensive hardware assistant. Our solution is simpler and requires far more less system resources. It is also suitable to be implemented in transceivers that is not IEEE 802.15.4 compliant.

Our solution have the FFD to control the timing of the sleeping device when to wake up and check in next time. As the result, the timing information will be attached to the indirect message response time. As the result, the indirect message has been changed to the following format:

Name	MAC Command	Rough Timing Info	Precise Timing Info	Indirect Message
Length (BYTE)	1	2	2	various
Description	Time Sync Data Packet Command (0x8A) for data indirect message. Time Sync Command Packet Command (0x8B) for command indirect message.	Timeout times on timers. Timers timeout roughly once per 16 seconds.	Timer ticks for precise timing control. One timer tick is configured to be around 244 us.	The indirect message itself.

Interfaces

To enable Time Synchronization feature, ENABLE_TIME_SYNC must be defined in configuration files for protocols: ConfigP2P.h or ConfigMiWi.h. The reason of enabling this feature in protocol layer instead of application layer is that both sleeping devices and FFDs must enable this feature at the same time. While configuration files in application layer is for each individual devices and configuration file in protocol layer is for every devices, it makes sense to enable/disable this feature in the protocol layer to avoid any mismatch in feature enabling.

Additional configuration for Time Synchronization is the total number of slots, TIME_SYNC_SLOTS, supported in the wake up interval of sleeping devices. **As the rule, the TIME_SYNC_SLOTS must be higher or equal to number of sleeping devices that connects to the FFD, so that every sleeping device can have at least one time slot.** Same as ENABLE_TIME_SYNC, TIME_SYNC_SLOTS is defined in protocol configuration files ConfigP2P.h or ConfigMiWi.h for the same reason above. Another configuration is the frequency for the external crystal that connects to the 16-bit asynchronized counter.

Apart from the configurations in protocol layer, there is no special requirement for function calls on application layer. There are additional hardware requirement for this feature. The details of additional hardware requirement can be found in the next section.

Additional Notes

Time Synchronization feature requires hardware support. The MCU needs a 16-bit timer working as asynchronized counter mode on a 32KHz external crystal. The timer will be able to run when the MCU is in sleep mode and wake up the MCU once it reaches the preset interval.

When Time Synchronization feature is enabled, the minimum time slot depends on the primary oscillator accuracy, 32 KHz external crystal accuracy as well as random time delay caused by CSMA-CA on the environment noise. On standard Microchip demo board, the time slot can be lowered to 100 millisecond.

1.6 Demos

1.6.1 Required Hardware

To run this project, you will need two wireless nodes. Each of the nodes can be any of setups of listed hardware.

1.6.1.1 Hardware Sets

Hardware Setups

Hardware Set 1:

- Demo Board:
  - PICDEM Z 2.4 Demo Kit (DM163027-4 OR DM163027-5) OR
  - PICDEM Z Mother Board (AC163027-1)
- RF Board:
  - MRF24J40
    - PICDEM Z MRF24J40 2.4GHz Daughter Card (AC163027-4) OR
    - MRF24J40MA PICDEM™ Z 2.4GHz RF Card (AC163028)

Hardware Set 2:

- Demo Board:
  - PIC18 Explorer with PIC18F87J11 PIM (DM183032)
- RF Board:
  - MRF24J40
    - MRF24J40MA PICtail (AC164134-1)
  - MRF49XA
    - MRF49XA PICtail 434MHz (AC164137-1) OR
    - MRF49XA PICtail 868/915MHz (AC164137-2)
  - MRF89XA
    - MRF89XA PICtail 868MHz OR
    - MRF89XA PICtail 915MHz
  - MRF24XA
    - MRF24XA PICtail 2.4GHz

Hardware Set 3:

- Demo Board:
  - Explorer 16 (DM240001)
  - PIC24FJ128GA010 Plug-In-Module (PIM) (MA240011)
- RF Borad
  - MRF24J40
    - PICDEM Z MRF24J40 2.4GHz Daughter Card (AC163027-4) OR

1

  - MRF24J40MA PICtail™ Plus 2.4GHz RF Card (AC164134) OR
  - MRF24J40MA PICtail (AC164134-1)
- MRF49XA
  - MRF49XA PICtail 434MHz (AC164137-1) OR
  - MRF49XA PICtail 868/915MHz (AC164137-2)
- MRF89XA
  - MRF89XA PICtail 868MHz OR
  - MRF89XA PICtail 915MHz
- MRF24XA
  - MRF24XA PICtail 2.4GHz

Hardware Set 4:

- Demo Board:
  - 8-bit Wireless Development Kit
- RF Board:
  - MRF24J40
    - MRF24J40MA PICtail (AC164134-1)
  - MRF49XA
    - MRF49XA PICtail 434MHz (AC164137-1) OR
    - MRF49XA PICtail 868/915MHz (AC164137-2)
  - MRF89XA
    - MRF89XA PICtail 868MHz OR
    - MRF89XA PICtail 915MHz
  - MRF24XA
    - MRF24XA PICtail 2.4GHz

1.6.2 Configuring the Hardware

This section describes how to set up the various configurations of hardware to run this demo:

Configuration 1: PICDEM Z Demo Kit

Configuration 2: PIC18 Explorer demo board

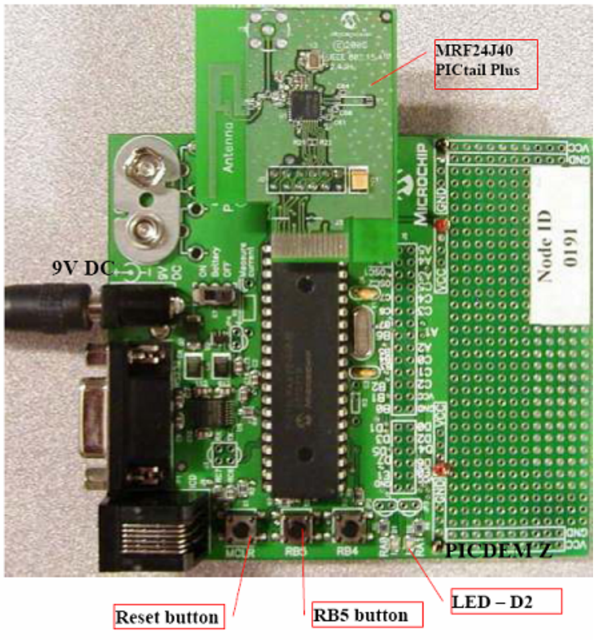
Configuration 3: Explorer 16 demo board, with PIC24FJ128GA010 or PIC32MX360F512L PIM

Configuration 4: 8-bit Wireless Development Kit

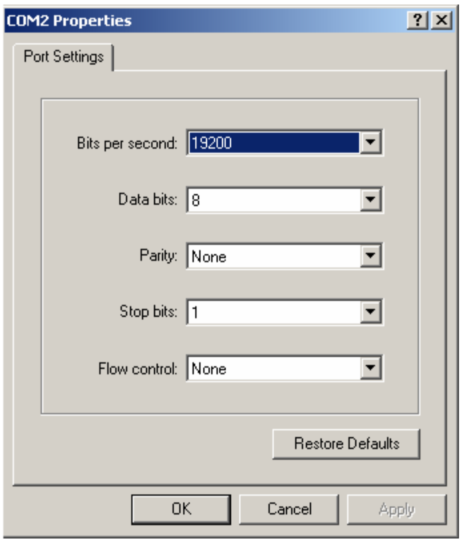
1.6.2.1 PICDEM Z

Configuration 1: PICDEM Z

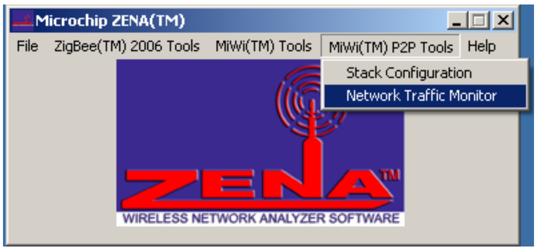
Connect the MRF24J40 2.4GHz RF Card tot he PICDEM Z demo board as shown in the picture



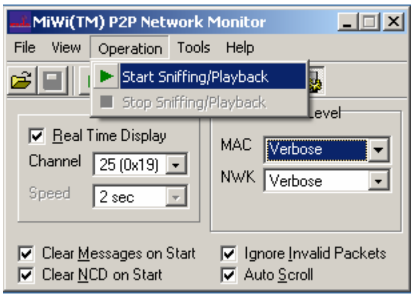
Before running the demos, it is highly recommended to connect a serial cable to both demo boards and connect the ZENA sniffer hardware to monitor the operating of the network. Once the serial cable is connected between the demo board and PC, launch a hyper terminal to display the information from the demo board. The hyper terminal configuration is baud rate 19200, Data bit 8, Parity None, Stop bits 1 and Flow control None, as shown below.



PICDEM Z demo board only support Microchip MRF24J40 transceiver, which is compliant with IEEE 802.15.4 specification. ZENA™ network analyzer can be used to monitor the network traffic of IEEE 802.15.4 network. To run ZENA™ sniffer, connect the ZENA board with PC through the USB interface, then launch ZENA software. The ZENA window will show up. Choose “MiWi™ P2P Tools” Menu or "MiWi(TM) Tools" Menu, depending on the protocol is used, and then click the menu item “Network Traffic Monitor” to launch Network Monitor window.



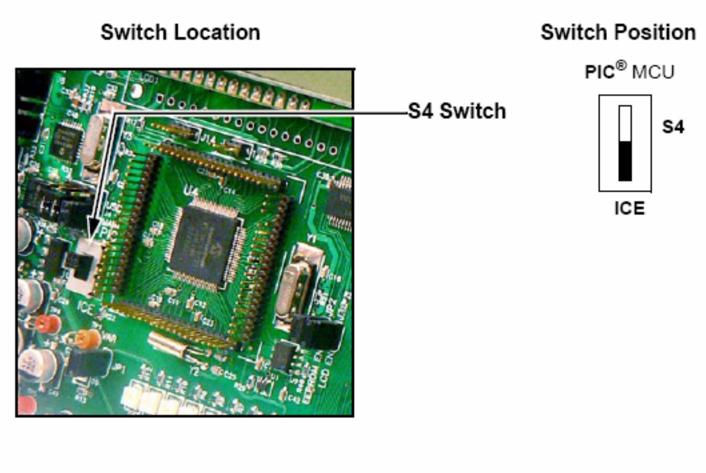
From the Network Monitor window, check “Real Time Display” box and choose proper channel. By default, this demo use channel 25. Choose “Operation” menu and click “Start Sniffing/Playback” menu item to launch the “ZENA™ Packet Sniffer” window to monitor the wireless traffic.



1.6.2.2 PIC18 Explorer

Configuration 2: PIC18 Explorer

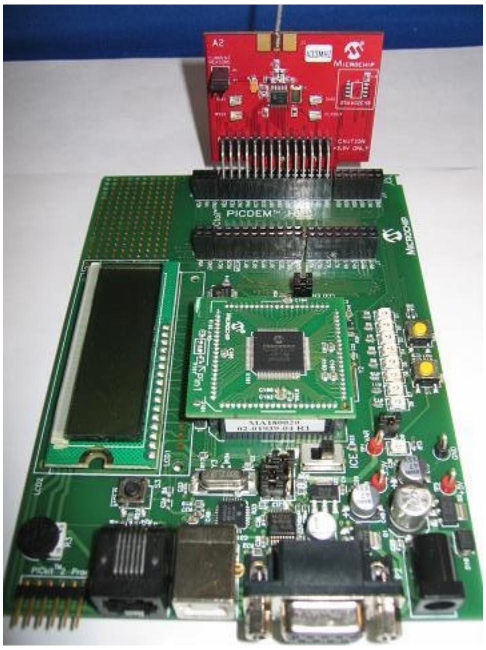
1) Set the S4 switch on PIC18 Explorer at the position of ICE.



2) Before connecting the PIM to the PIC18 Explorer board, remove all attached cables. Connect the PIM to the PIC18 Explorer board. Be careful when connecting the boards to insure that no pins are bent or damaged during the process. Also ensure that the PIM is not shifted in any direction and that all of the headers are properly aligned.

3) Connect the either the MRF24J40, MRF49XA or MRF89XA RF board to the PICTail connector. Be aware that the transceiver chip should face the PIM and the first pin should be plugged into the hole labeled "RE2".

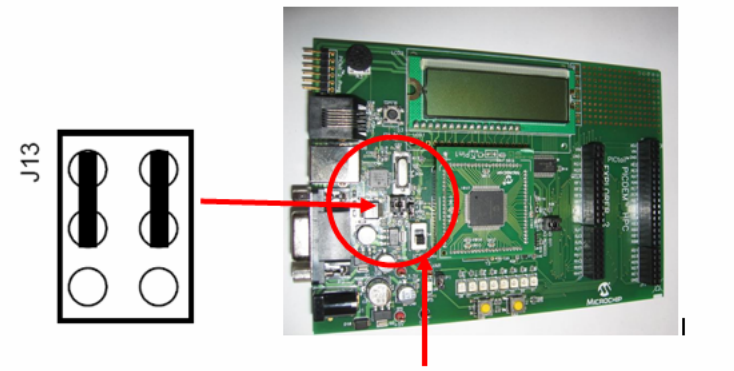
The configured hardware setup for PIC18 Explorer board should look like following picture.



PIC18 Explorer support both RS232 serial port and USB to connect to the PC for monitoring. PIC18 Explorer by default is configured to use the RS232 serial port to communicate with PC. Following steps setup the PIC18 Explorer to use USB port.

1. Hardware Setup

Configure the Explorer PIC18 demo board to use USB connection by setting jumper J13 according to the following diagram

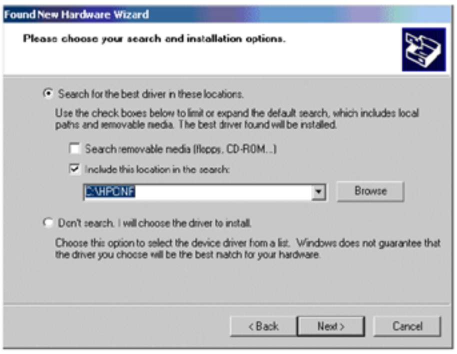


Make sure toggle switch is in the DOWN – ICE position. This switch activates the PIC18 on the PIM. DO NOT remove the PIM or the board Vdd will be 5V, which may damage the RF module.

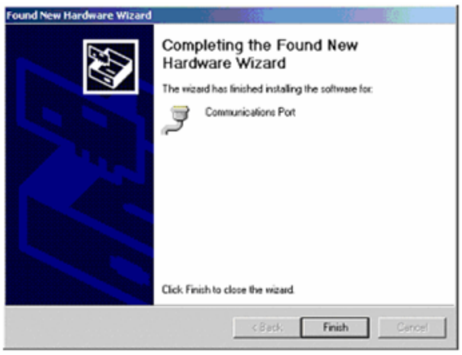
2. USB Driver Install

- Connect the PIC18 Explorer demo board to the PC using a USB cable.
- Power up PIC18 Explorer demo board, following pop up window will appear

- Select "Install from a list or specific location" option and click "Next". Following pop up window appear



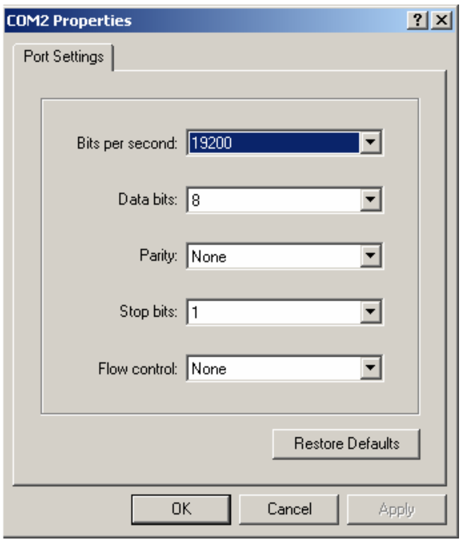
- Select the check box "Include this location in the search", in the text box, browse to "<Install Directory>\PC Software" folder. This is the location of the "mchpcdc.inf" driver.
- Click "Next". There may be warning from Windows operating system about installing a driver without digital signature. Please ignore that warning and continue. After the driver is installed properly, the following screen will appear



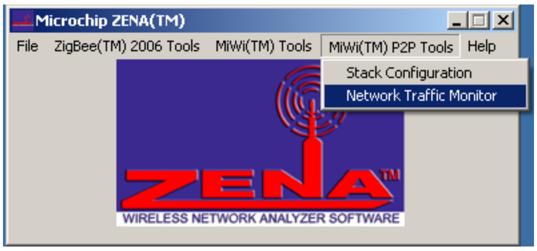
- Click "Finish". USB port is ready to be used.

3. Open Hyper Terminal

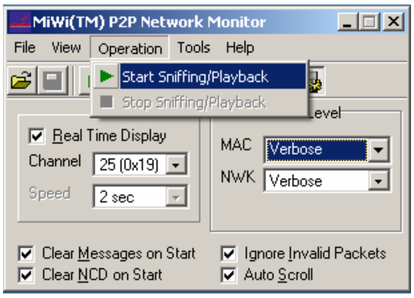
Once the RS232 serial or USB cable is connected between the demo board and PC, launch a hyper terminal to display the information from the demo board. The hyper terminal configuration is baud rate 19200, Data bit 8, Parity None, Stop bits 1 and Flow control None, as shown below.



In the case that Microchip MRF24J40 transceiver is used in the demo, ZENA™ network analyzer can be used to monitor the network traffic of IEEE 802.15.4 network. To run ZENA™ sniffer, connect the ZENA board with PC through the USB interface, then launch ZENA software. The ZENA window will show up. Choose “MiWi™ P2P Tools” Menu or "MiWi(TM) Tools" Menu, depending on the protocol is used, and then click the menu item “Network Traffic Monitor” to launch Network Monitor window.



From the Network Monitor window, check “Real Time Display” box and choose proper channel. By default, this demo use channel 25. Choose “Operation” menu and click “Start Sniffing/Playback” menu item to launch the “ZENA™ Packet Sniffer” window to monitor the wireless traffic.

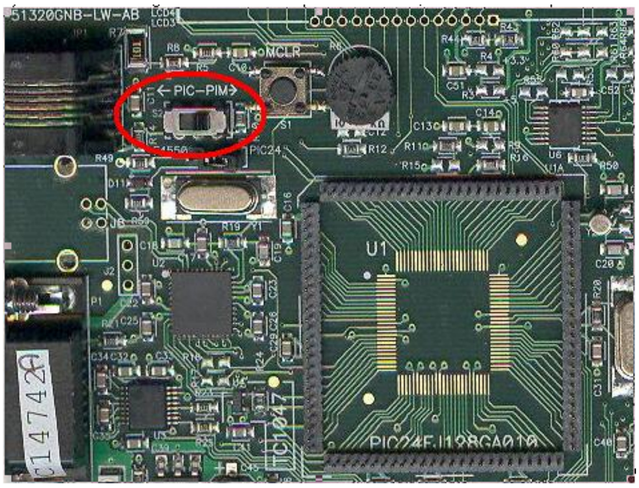


In the case Microchip MRF49XA or MRF89XA transceiver is used in the demo, setting the RF utility driver in the receiving mode can be used as the basic sniffer, though packet decoding is not supported.

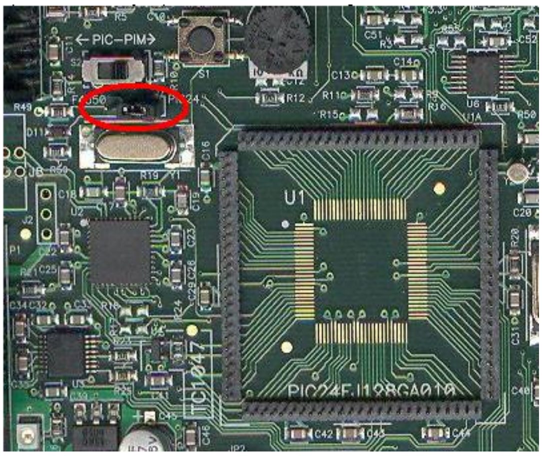
1.6.2.3 Explorer 16

Configuration 3: Explorer 16

1) Before attaching the PIM to the Explorer 16 board, ensure that the processor selector switch (S2) is in the “PIM” position as seen in the image below:

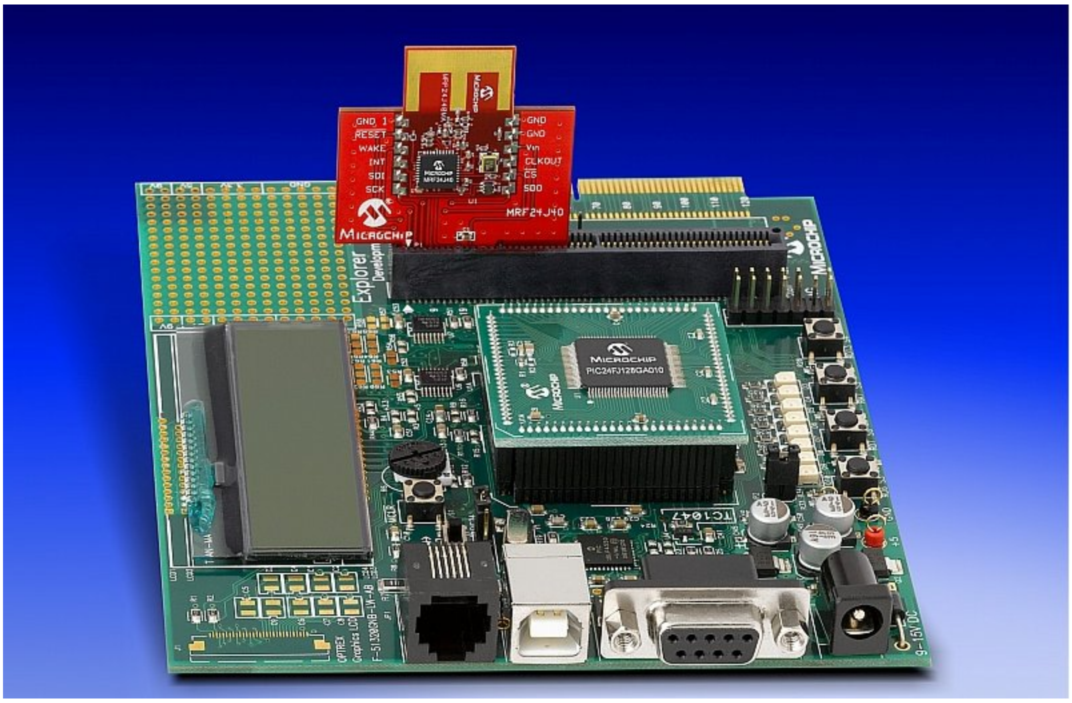


2) Short the J7 jumper to the "PIC24" setting

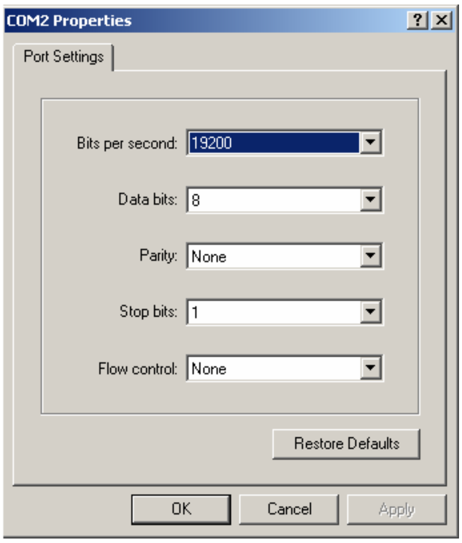


3) Before connecting the PIM to the Explorer 16 board, remove all attached cables. Connect the PIM to the Explorer 16 board. Be careful when connecting the boards to insure that no pins are bent or damaged during the process. Also ensure that the PIM is not shifted in any direction and that all of the headers are properly aligned.

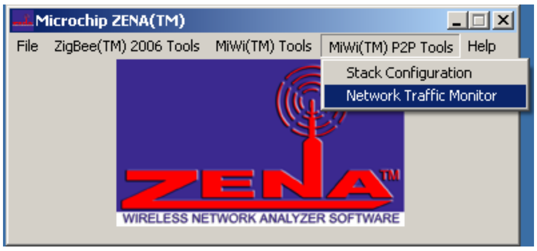
4) Connect the RF board for MRF24J40 or MRF49XA to the first slot of edge card connector, as shown in the following picture.



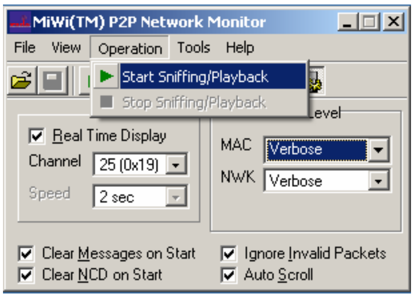
5) Before running the demos, it is highly recommended to connect a serial cable to both demo boards and connect the ZENA sniffer hardware to monitor the operating of the network. Once the serial cable is connected between the demo board and PC, launch a hyper terminal to display the information from the demo board. The hyper terminal configuration is baud rate 19200, Data bit 8, Parity None, Stop bits 1 and Flow control None, as shown below.



In the case that Microchip MRF24J40 transceiver is used in the demo, ZENA™ network analyzer can be used to monitor the network traffic of IEEE 802.15.4 network. To run ZENA™ sniffer, connect the ZENA board with PC through the USB interface, then launch ZENA software. The ZENA window will show up. Choose “MiWi™ P2P Tools” Menu or "MiWi(TM) Tools" Menu, depending on the protocol is used, and then click the menu item “Network Traffic Monitor” to launch Network Monitor window.



From the Network Monitor window, check “Real Time Display” box and choose proper channel. By default, this demo use channel 25. Choose “Operation” menu and click “Start Sniffing/Playback” menu item to launch the “ZENA™ Packet Sniffer” window to monitor the wireless traffic.



In the case Microchip MRF49XA or MRF89XA transceiver is used in the demo, setting the RF utility driver in the receiving mode can be used as the basic sniffer, though packet decoding is not supported.

1.6.2.4 8-bit Wireless Demo Board

Configuration 4: 8-bit Wireless Demo Board

Connect the 8-bit Wireless Demo Board in the following way that is shown in the picture. Be aware that the jumper "JP1" on the LCD daughter board should be removed to work with RS232 daughter board.

1.6.3 Firmware

Firmware

To run this project, you will need to load the corresponding firmware into the devices. There are two methods available for loading the demos: Precompiled demos and source code projects.

1.6.3.1 Precompiled HEX Files

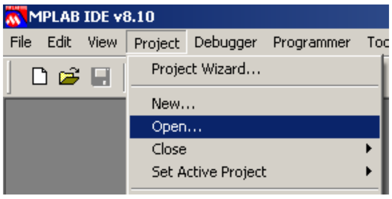
Precompiled HEX Files

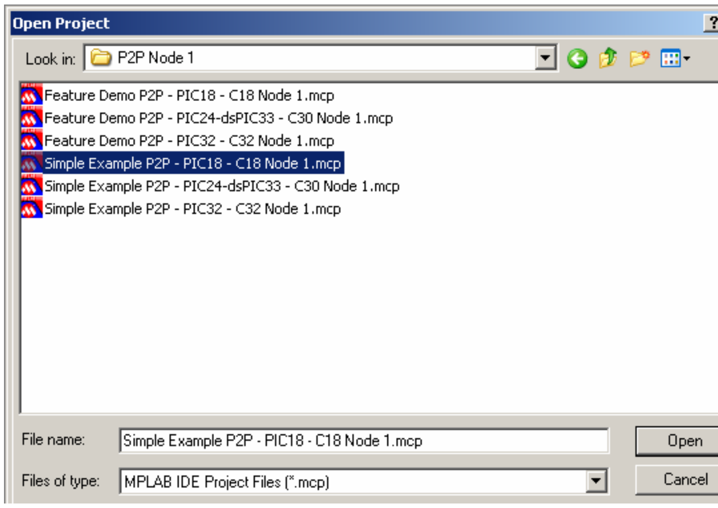
Precompiled Demos are available in the “<Install Directory>\MiWi DE Demo\Node 1\HEX” and “<Install Directory>\MiWi DE Demo\Node 2\HEX” folders. A hex file is provided for each hardware configuration and each available RF transceiver under the two directories. Import the corresponding hex file and then program the hex to the demo board.

1.6.3.2 Demo Source Code Project

Demo Source Code Modification

The source code for this demo is available in the “<Install Directory\P2P\P2P Node 1” and “<Install Directory\P2P\P2P Node 2” directories. In these directories, you will find all of the application level source and header files as well as project files for each of the hardware platforms. To open the project, select “Project” from the main menu of MPLAB IDE and then “Open…” option. A window will pop out and request the project file. The snap-shot of the opening project process can be found below:





To run the demo, both nodes must be configured to use the same RF transceiver with the same settings and the same wireless protocol. However, both nodes do not have to be the same demo board.

To compile the demo, following working environment must be established:

• PICDEM Z, PIC18 Explorer and RFD Board Configuration: C18 v3.20 or higher

• Explorer 16 Configuration: C30 v3.10 or higher for PIC24 and dsPIC33; C32 v1.02 or higher for PIC32.

Compile and program the demo code into the hardware platform. For more help on how to compile and program projects, please refer to the MPLAB® IDE help available through the help menu of MPLAB IDE (Help->Topics…->MPLAB IDE).

By default, there are three projects provided for each demo. They are for PIC18, PIC24 and PIC32 MCUs respectively. Each demo is configured to use MRF24J40 RF transceiver to run MiWi™ P2P stack. With minor configuration modification, user can compile and run the demo on any Microchip RF transceivers, any Microchip proprietary wireless protocols on any supported standard demo boards. This section demonstrates how to migrate the demo among all supported options.

1.6.3.2.1 MiWi P2P

1.6.3.2.1.1 PICDEM Z Demo Board for MiWi P2P

PICDEM Z Demo Board for MiWi™ P2P

PICDEM Z Demo board use PIC18F4620 MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC18F4620" as the device and then click "OK".

Open either the simple example or feature demo project for PIC18.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi P2P protocol, make sure to uncomment "#define PROTOCOL_P2P" and comment out "#define PROTOCOL_MIWI".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following defintions:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the PICDEM Z board by uncomment "#define PICDEMZ" and comment out all other demo boards definitions.

Remove any linker script in the project, as shown below.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1.6.3.2.1.2 PIC18 Explorer Demo Board for MiWi P2P

PIC18 Explorer Demo Board for MiWi™ P2P

By default, PIC18 Explorer Demo board use PIC18F87J11 MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC18F87J11" as the device and then click "OK". If you use a different PIM other than the default PIC18F87J11, please select the corresponding MCU accordingly.

Open either the simple example or feature demo project for PIC18.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi P2P protocol, make sure to uncomment "#define PROTOCOL_P2P" and comment out "#define PROTOCOL_MIWI".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following definitions. The other two definition of RF transceiver must be commented out:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the PIC18 Explorer board by uncomment "#define PIC18_EXPLORER" and comment out all other demo boards definitions.

Due to a bug in the earlier version of C compiler, the memory map of PIC18F87J11 may be incorrect. The latest version of C18 compiler has fixed this problem. In the case that you are not sure if you have the latest C18 compiler, it is highly recommended to add the linker script "18f87j11_e.lkr" within the project if PIC18 Explorer demo board and default PIC18F87J11 PIM are used in this demo.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1.6.3.2.1.3 Explorer 16 Demo Board for MiWi P2P

Explorer 16 Demo Board for MiWi™ P2P

Explorer 16 demo board support development for PIC24, dsPIC33 and PIC32.

1.6.3.2.1.3.1 PIC24 or dsPIC33 for MiWi P2P

PIC24 or dsPIC33 on Explorer 16 Demo Board for MiWi™ P2P

Explorer 16 Demo board use PIC24FJ128GA010 MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC24FJ128GA010" as the device and then click "OK".

Open either the simple example or feature demo project for PIC24 and dsPIC33.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi P2P protocol, make sure to uncomment "#define PROTOCOL_P2P" and comment out "#define PROTOCOL_MIWI".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following defintions:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the Explorer 16 board by uncomment "#define EXPLORER16" and comment out all other demo boards definitions.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1.6.3.2.1.3.2 PIC32 for MiWi P2P

PIC32 on Explorer 16 Demo Board for MiWi™ P2P

Explorer 16 Demo board use PIC32MX360F512L MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC32MX360F512L" as the device and then click "OK".

Open either the simple example or feature demo project for PIC32.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi P2P protocol, make sure to uncomment "#define PROTOCOL_P2P" and comment out "#define PROTOCOL_MIWI".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following defintions:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the Explorer 16 board by uncomment "#define EXPLORER16" and comment out all other demo boards definitions.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1.6.3.2.1.4 8-bit Wireless Demo Board for MiWi P2P

8-bit Wireless Demo Board for MiWi™ P2P

8-bit Wireless Demo board use PIC18F46J50 MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC18F46J50" as the device and then click "OK".

Open either the simple example or feature demo project for PIC18.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi P2P protocol, make sure to uncomment "#define PROTOCOL_P2P" and comment out "#define PROTOCOL_MIWI".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following defintions:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the 8-bit wireless demo board by uncomment "#define EIGHT_BIT_WIRELESS_BOARD" and comment out all other demo boards definitions.

Remove any linker script in the project, as shown below.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1.6.3.2.2 MiWi

1.6.3.2.2.1 PICDEM Z Demo Board for MiWi

PICDEM Z Demo Board for MiWi™

PICDEM Z Demo board use PIC18F4620 MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC18F4620" as the device and then click "OK".

Open either the simple example or feature demo project for PIC18.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi protocol, make sure to uncomment "#define PROTOCOL_MIWI" and comment out "#define PROTOCOL_P2P".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following defintions:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the PICDEM Z board by uncomment "#define PICDEMZ" and comment out all other demo boards definitions.

Remove any linker script in the project, as shown below.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1.6.3.2.2.2 PIC18 Explorer Demo Board for MiWi

PIC18 Explorer Demo Board for MiWi™

By default, PIC18 Explorer Demo board use PIC18F87J11 MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC18F87J11" as the device and then click "OK". If you use a different PIM other than the default PIC18F87J11, please select the corresponding MCU accordingly.

Open either the simple example or feature demo project for PIC18.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi protocol, make sure to uncomment "#define PROTOCOL_MIWI" and comment out "#define PROTOCOL_P2P".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following definitions. The other two definition of RF transceiver must be commented out:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the PIC18 Explorer board by uncomment "#define PIC18_EXPLORER" and comment out all other demo boards definitions.

Due to a bug in the earlier version of C compiler, the memory map of PIC18F87J11 may be incorrect. The latest version of C18 compiler has fixed this problem. In the case that you are not sure if you have the latest C18 compiler, it is highly recommended to add the linker script "18f87j11_e.lkr" within the project if PIC18 Explorer demo board and default PIC18F87J11 PIM are used in this demo.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1

1.6.3.2.2.3 8-bit Wireless Demo Board for MiWi

8-bit Wireless Demo Board for MiWi™

8-bit wireless demo board use PIC18F46J50 MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC18F46J50" as the device and then click "OK".

Open either the simple example or feature demo project for PIC18.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi protocol, make sure to uncomment "#define PROTOCOL_MIWI" and comment out "#define PROTOCOL_P2P".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following definitions:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the 8-bit wireless board by uncomment "#define EIGHT_BIT_WIRELESS_BOARD" and comment out all other demo boards definitions.

Remove any linker script in the project, as shown below.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1.6.3.2.2.4 Explorer 16 Demo Board for MiWi

Explorer 16 Demo Board for MiWi™

Explorer 16 demo board support development for PIC24, dsPIC33 and PIC32.

1.6.3.2.2.4.1 PIC24 or dsPIC33 for MiWi

PIC24 or dsPIC33 on Explorer 16 Demo Board for MiWi™

Explorer 16 Demo board use PIC24FJ128GA010 MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC24FJ128GA010" as the device and then click "OK".

Open either the simple example or feature demo project for PIC24 and dsPIC33.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi protocol, make sure to uncomment "#define PROTOCOL_MIWI" and comment out "#define PROTOCOL_P2P".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following definitions:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the Explorer 16 board by uncomment "#define EXPLORER16" and comment out all other demo boards definitions.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1.6.3.2.2.4.2 PIC32 for MiWi

PIC32 on Explorer 16 Demo Board for MiWi™

Explorer 16 Demo board use PIC32MX360F512L MCU as the host controller. Select "Configure" from the MPLAB menu and then choose "Select Device...". From the pop up menu, choose "PIC32MX360F512L" as the device and then click "OK".

Open either the simple example or feature demo project for PIC32.

From the project window, choose to edit file "ConfigApp.h" under the directory "Header Files -> Application", as shown below.

In file "ConfigApp.h", first elect to use MiWi P2P protocol, make sure to uncomment "#define PROTOCOL_MIWI" and comment out "#define PROTOCOL_P2P".

Second step, choose the RF transceiver to be used. Three RF transceivers: MRF24J40, MRF49XA and MRF89XA are supported in this release. Support of RF transceiver is enabled by uncomment one and only one following defintions:

#define MRF24J40

#define MRF49XA

#define MRF89XA

From the project window, choose to edit file "HardwareProfile.h" under the directory "Header Files -> Application", as shown below

In the file "HardwareProfile.h", choose the Explorer 16 board by uncomment "#define EXPLORER16" and comment out all other demo boards definitions.

Compile the project and then load the hex file to the MCU through a programmer or debugger.

1.6.4 Running Demos

Running Demos

Two demos are provided to demonstrate the simplicity and functionalities of MiWi™ Development Environment.

1.6.4.1 Simple Example

Simple Example

The simple example application code focuses on the simplicity of the MiWi™ DE protocol stack application programming interfaces. It provides a clean and straightforward wireless communication between two devices with less than 30 lines of effective C code to run the stack in application layer for both devices. In this application, following features of MiWi™ DE protocol stack have been demonstrated:

• Establish connection automatically between two devices

• Broadcast a packet

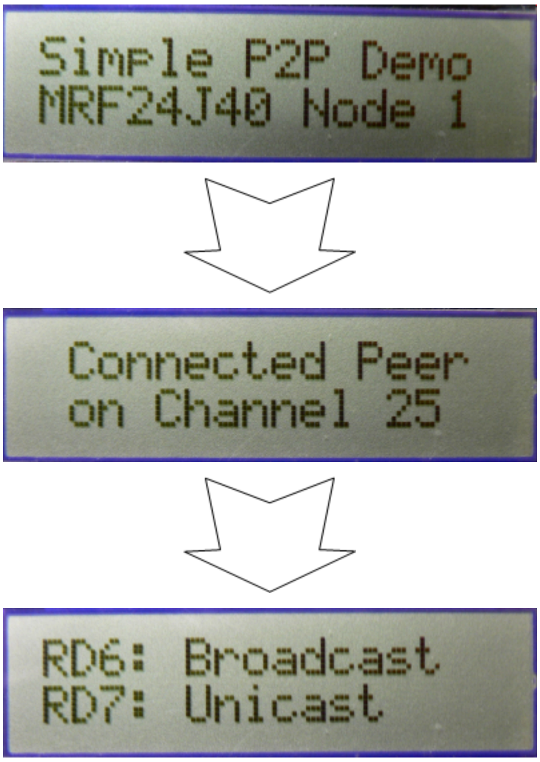
• Unicast a packet

• Apply security to the transmitted packet

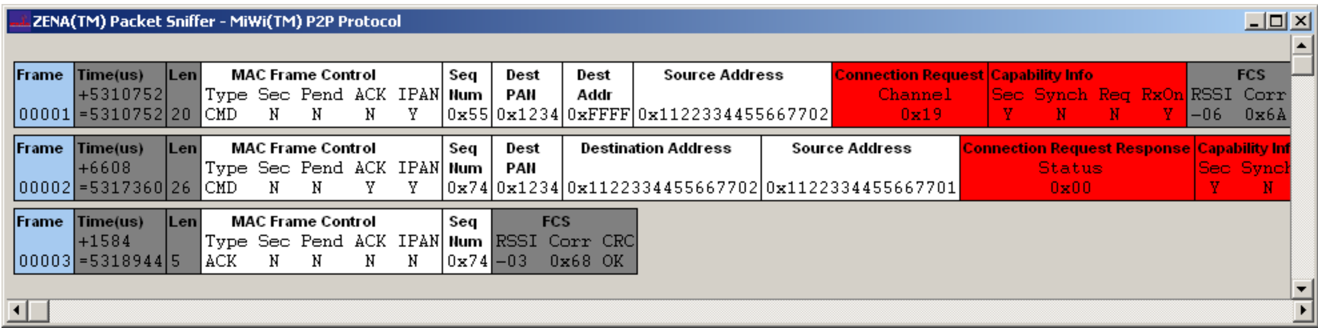
To run the simple example application, following is the instruction:

1. Program node 1 and node 2 with proper firmware. We assume that the users are familiar with Microchip tool chain and have no problem compile and program the firmware to the demo boards.
2. Power on node 1 and node 2 respectively
3. Wait a few seconds, until the first LED (RA0 on PICDEM Z, D8 on PIC18 Explorer or D10 on Explorer 16) on both nodes light up. These are the steps to establish connections between two devices.
    - This means a connection has been established automatically. For the details of connection establishment, please refer to section “VARIATIONS FOR HANDSHAKING” in application note AN1204 “Microchip MiWi™ P2P Wireless Protocol" if MiWi™ P2P protocol is used, or section "MAC Function Description" in IEEE 802.15.4 specification if MiWi™ protocol is used.

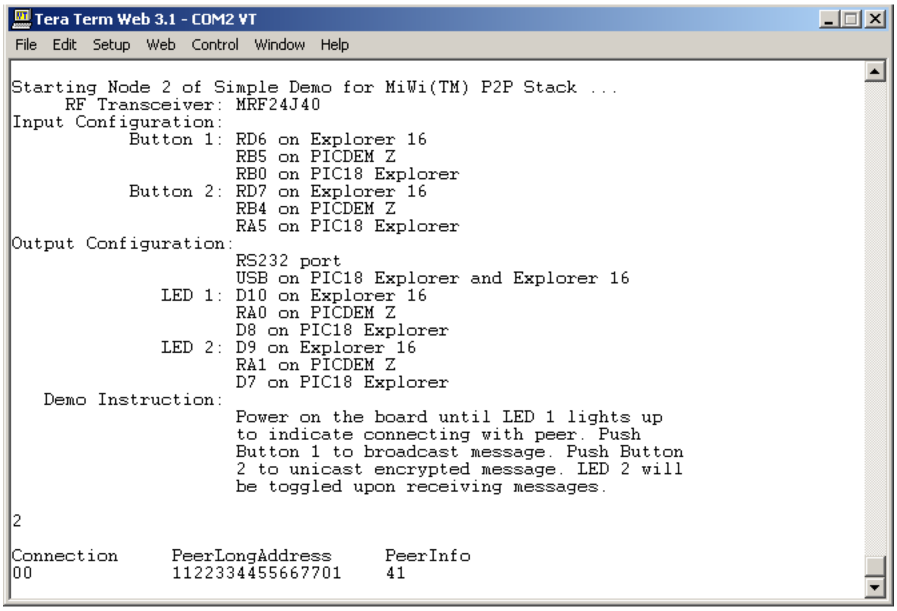
- If the demo is running on PIC18 Explorer, 8-bit wireless demo board or Explorer 16 demo boards, critical information will be shown on the LCD of the demo board. It first shows the demo name, RF transceiver and node number, then connecting information and channel information will be shown before the LCD shows the demo instruction: button 1 for broadcast and button 2 for unicast.



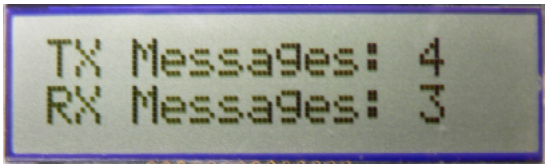
- If MRF24J40 transceiver is demonstrated and ZENA network analyzer is used, the default channel is 25. You should be able to see the hand-shaking procedure with exchanged packets between two nodes.



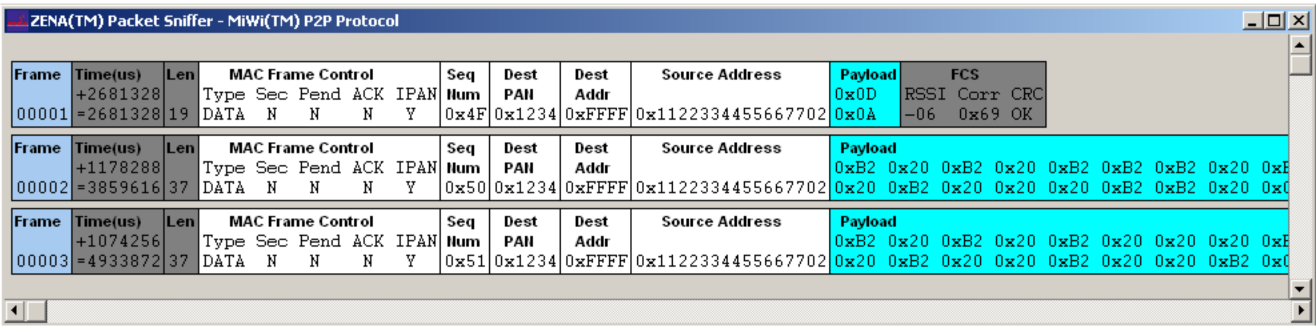
- If a hyper terminal has been opened to monitor firmware output, you should be able to see the information about the peer device printed out from both nodes.



4. Press button 1 (RB5 on PICDEM Z, RB0 on PIC18 Explorer or RD6 on Explorer 16) on one node will toggle the second LED (RA1 on PICDEM Z, D7 on PIC18 Explorer or D9 on Explorer 16) on the other node
   - This shows how a broadcast packet has been transmitted.
   - If the demo is running on PIC18 Explorer or Explorer 16 demo board, the total number of transmitted and received messages will be shown on the LCD.



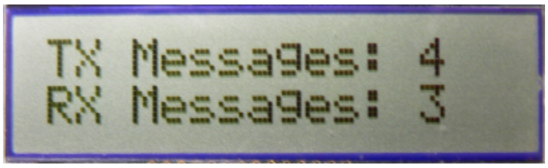
- If ZENA network analyzer is used, you should be able to see that a broadcast packet with various bytes has been sent out.



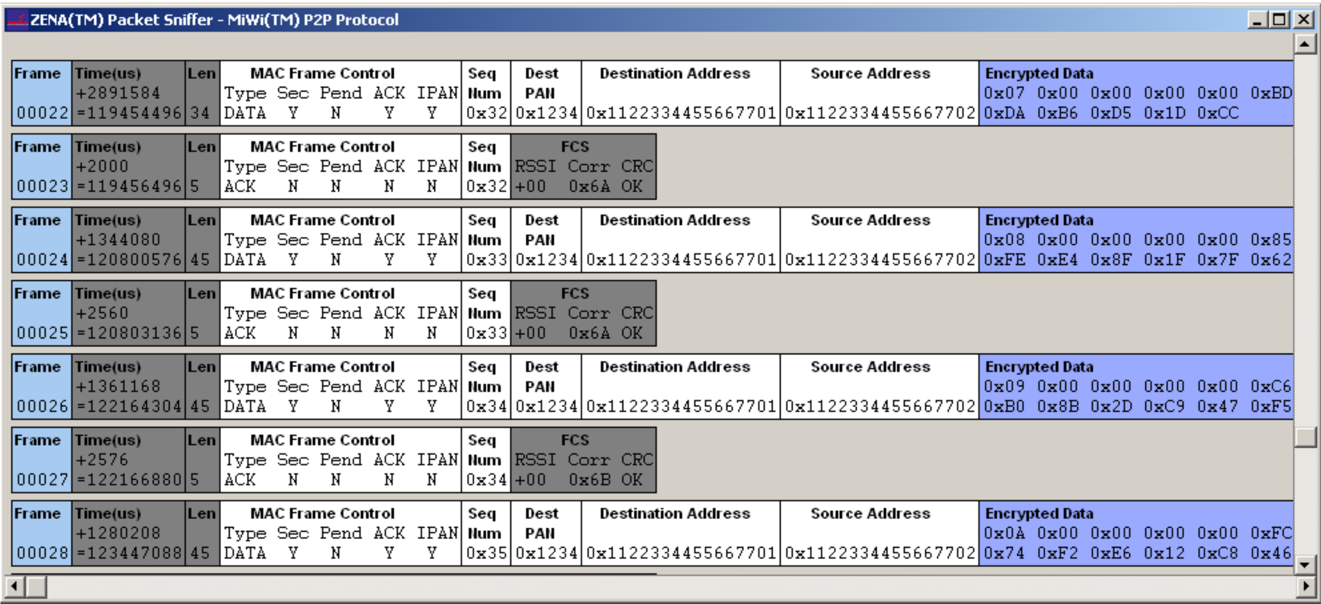
- If hyper terminal has been used, on the receiving end (the device that has LED2 toggled), you should be able to see the print out of broadcast packet source address, signal strength and the packet payload. The packet payload is the one line of bit map of “MiWi”. Press the button 1 continuously on one end will display the complete bit map of "MiWi".

5. Press button 2 (RB4 on PICDEM Z, RA5 on PIC18 Explorer or RD7 on Explorer 16) on one node will toggle the second LED (RA1 on PICDEM Z, D7 on PIC18 Explorer or D9 on Explorer 16) on the other node.
   - This shows how an encrypted unicast packet has been transmitted and decrypted by the radio after it is received. For the details of how MiWi™ P2P handles encryption, please refer to section “Security Features” in application note AN1204 “Microchip MiWi™ P2P Wireless Protocol”.
   - If the demo is running on PIC18 Explorer or Explorer 16 demo board, the total number of transmitted and received messages will be shown on the LCD.

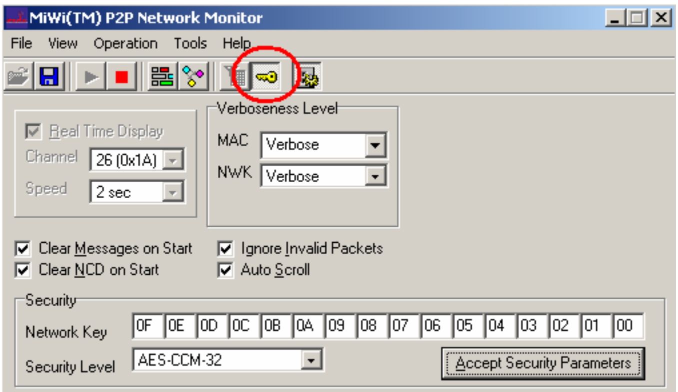
1



- If ZENA network analyzer is used, you should be able to see that a unicast packet with various bytes has been transmitted from one device and the acknowledgement packet with 5 bytes transmitted from the other device. You will also notice that the unicast packet is encrypted.



- By pressing the button with a key icon on the MiWi™ Network Monitor window, the encrypted packet can be decrypted with correct security setting. For the security setting, the key is 0x0F0E0D0C0B0A09080706050403020100 and the security level is AES-CCM-32. Press the button of “Accept Security Parameters” will apply the security setting to decrypt the packets. If your ZENA software disables the security feature, you need to order a full version ZENA through Microchip agent to be compliant with US export control regulation.



- If hyper terminal has been used, on the receiving end (the device that has LED2 toggled), you should be able to see the print out of secured unicast packet source address, signal strength and the packet payload. The packet payload should have been decrypted by the receiving device. The packet payload is the one line of bit map of “DE”. Press the button 2 continuously on one end will display the complete bit map of "DE".

1.6.4.2 Feature Demo

Feature Demo

The feature demo application code demonstrates the rich features of MiWi™ DE protocol stack. It shows how the stack manages to operate on optimal condition as well as robustness of the stack that can recover from operating environment change. In this application, in addition to features shown in simple example application, following features of MiWi™ DE protocol stack have been demonstrated:

- Network Freezer feature that restores network after power cycle.
- Active Scan to locate existing PAN in the neighborhood
- Energy Scan to find channel with least noise
- Sleeping device to conserve energy to be able to be powered by battery
- Indirect message to be able to deliver message to sleeping device
- Frequency agility capability that is able to change operating channel in case operating environment changes
- Resynchronization capability that is able to resynchronize with the original PAN in case operating channel was changed

Optionally, the feature demo can also demonstrate following new features when they are enabled in the protocol configuration.

- Enhanced Data Request feature that lowers the total active time for sleeping device
- Time Synchronization feature that enable large number of sleeping devices to connect with one FFD without communication confliction.

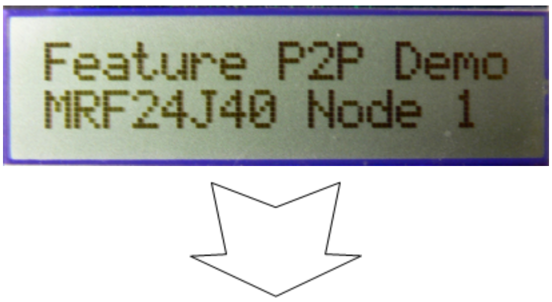
To run the feature demo application, following is the instruction:

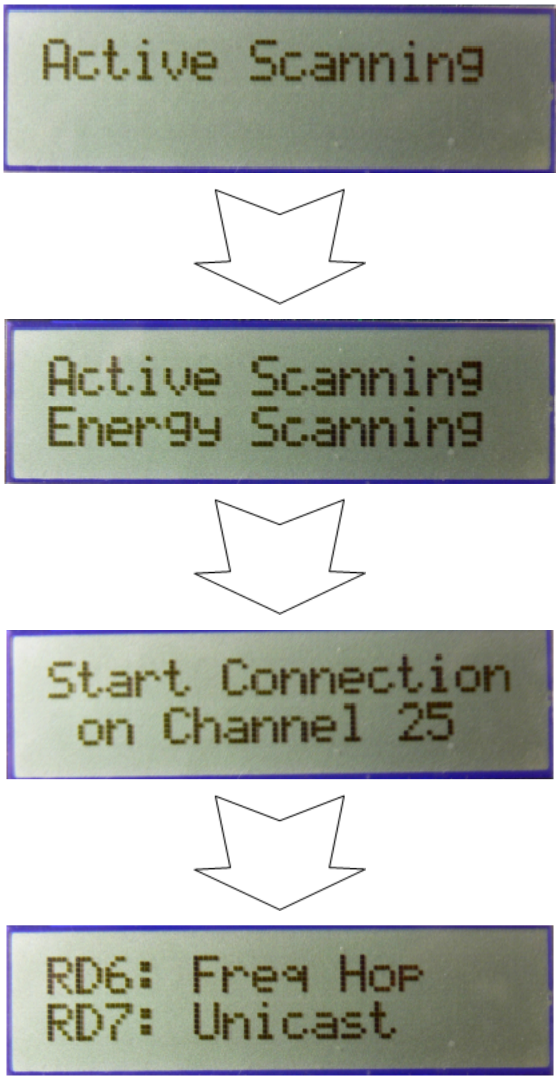
1. Program the proper firmware to node 1 and node 2 respectively. We assume that the users are familiar with Microchip tool chain and have no problem to compile and program the firmware to the demo boards.
2. Power on Node 1

If this device has been part of network before and you would like to restore the previous network configuration, follow steps below to restore network configuration in node 1.

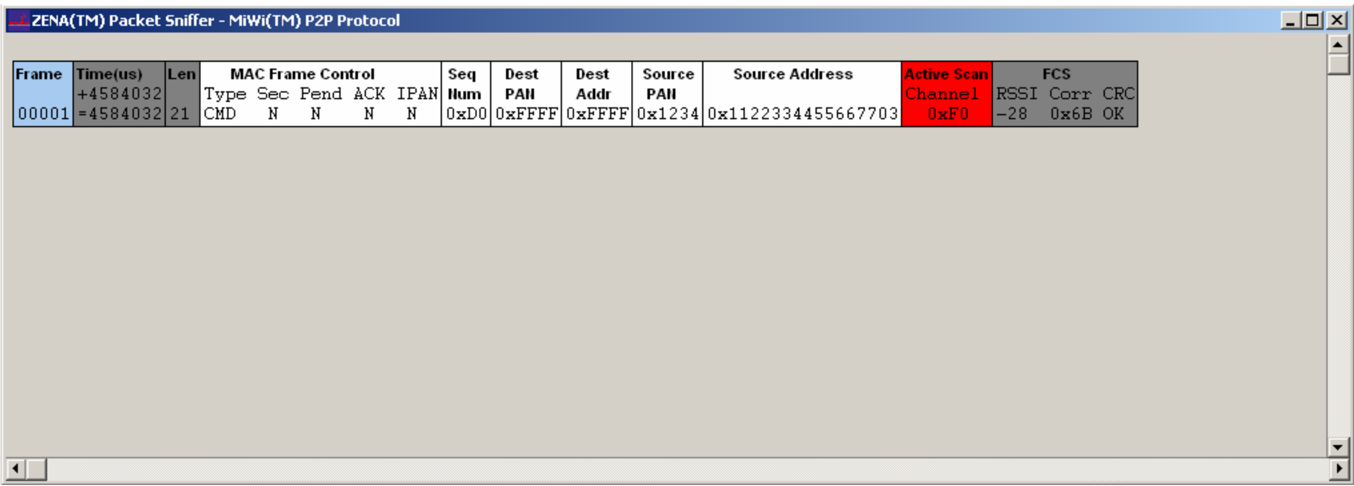
- Press and hold button 1 on node 1 before powering on
  - Power on node 1 for 5 seconds.
  - Release button 1 on node 1. At this time, the network has been recovered and you could get to step 4 to power on node 2; otherwise, continue on step 3 to start the network.

3. Wait a while until the LED1 lights up on node 1.
   - In this step, an active scan and a possible energy scan has been done by node 1. The PAN has been established on the channel with least noise. For details of active scan and energy scan, please refer to sections “Active Scan” and “Energy Scan” in Microchip application note AN1204 “Microchip MiWi™ P2P Wireless Protocol” if MiWi™ P2P is used as the protocol, or section "MAC Function Description" in IEEE 802.15.4 specification if MiWi™ protocol is used.
   - If PIC18 Explorer, 8-bit Wireless demo board or Explorer 16 demo boards are used in the demo, the demo name, RF transceiver and node number will be displayed on the LCD first. Then the message of active scan and energy scan will be displayed respectively. Finally, the demo instruction will be displayed on the LCD: button 1 for frequency hopping and button 2 for indirect unicast to sleeping device.

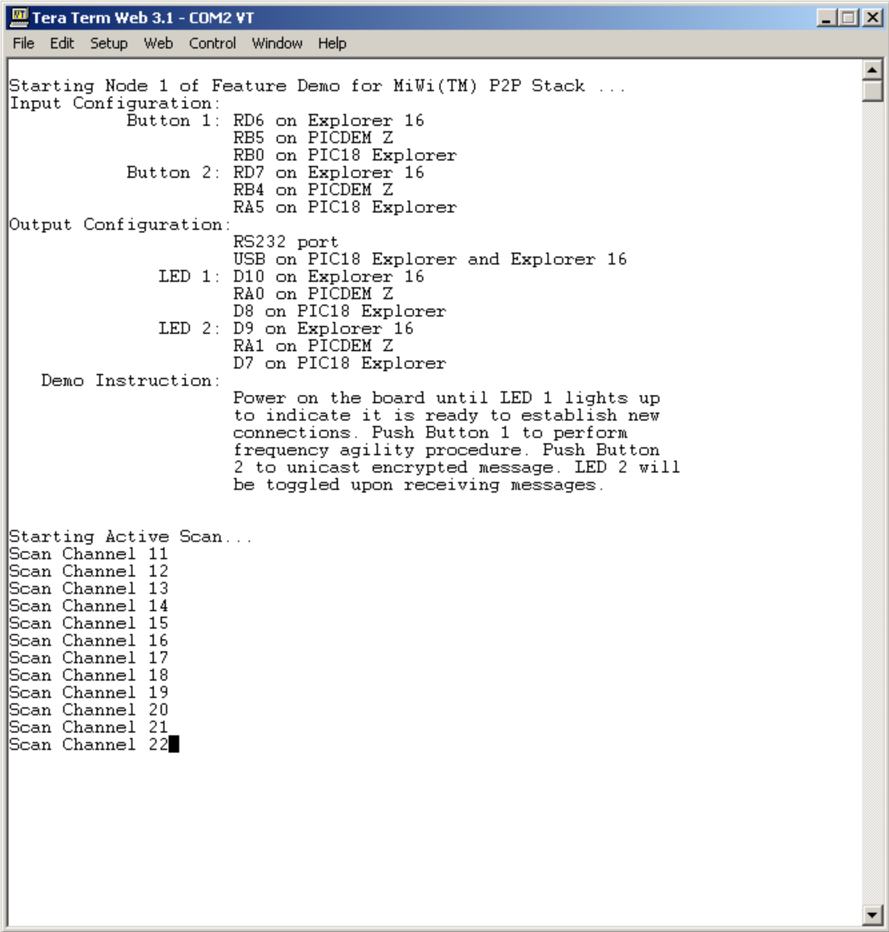




- If MRF24J40 is demonstarted and ZENA network analyzer is used, no matter what channel you are monitoring, you should be able to see a broadcast command. That is the active scan from node 1. After the LED1 lights up, if hyper terminal is used to monitor firmware output, please change the ZENA monitor channel to where the PAN has established, based on the print out on the hyper terminal.



- If hyper terminal has been connected between PC and demo board, you should see from the hyper terminal that node 1 does an active scan first. If there is any MiWi™ P2P or MiWi™ PAN established in the neighborhood, you should be able to see the printout of the list of available PANs. If one of the PANs has the same PAN identifier as the desired one, node 1 will try to establish a connection with that PAN. Otherwise, node 1 will do an energy scan. You should be able to see the energy reading on each channel printed out on the hyper terminal. At the end, you will see that node 1 establishes the PAN on the channel with least noise, or energy reading.

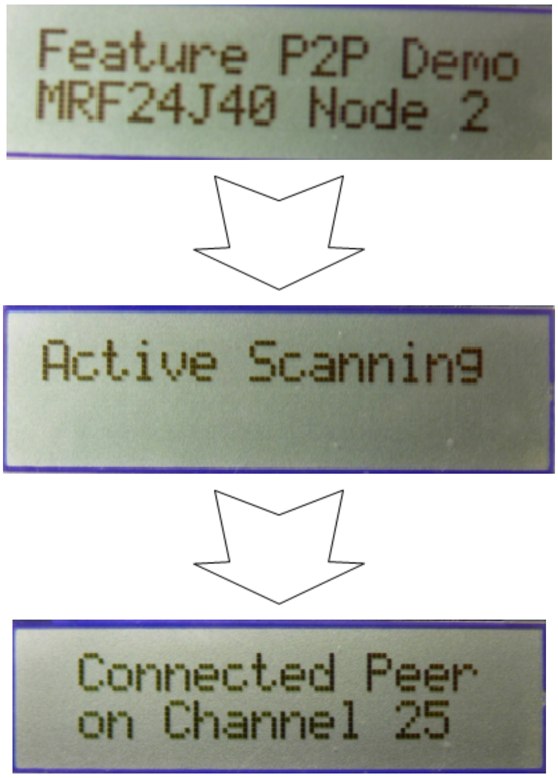


4. Power on node 2

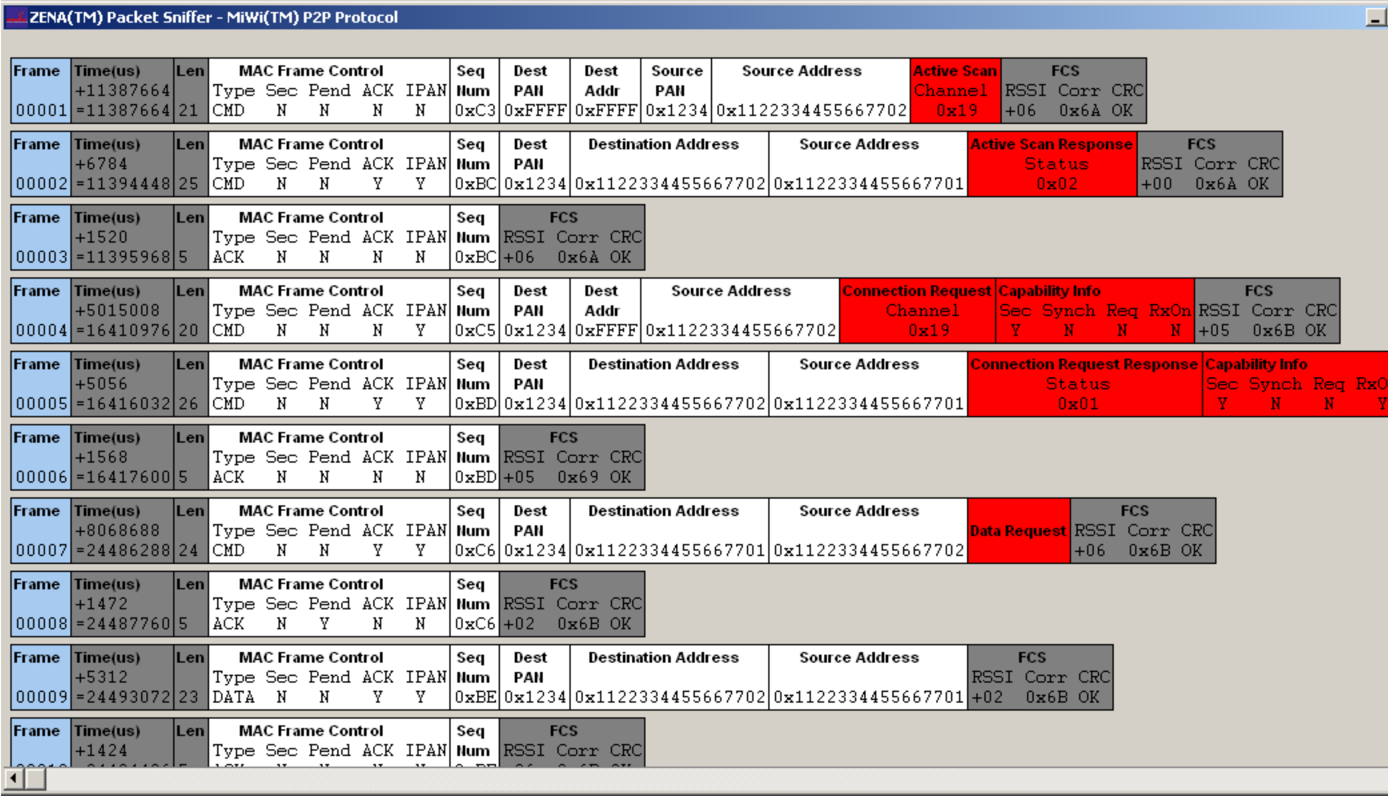
If this device has been part of network before and you would like to restore the previous network configuration, follow steps below to restore network configuration in node 2.

- Press and hold button 1 on node 2 before powering on
  - Power on node 2 for 5 seconds.
  - Release button 1 on node 2. At this time, the network has been recovered and you could get to step 6 to perform transmission and receiving; otherwise, continue on step 5 to join the network.

5. Wait a few seconds until the LED1 lights up on node 2.
   - In this step, an active scan has been done by node 2; node 1 has responded to node 2's active scan and node 2 established a P2P connection with node 1. For details of active scan, please refer to section "Active Scan" in Microchip application note AN1204 "Microchip MiWi™ P2P Wireless Protocol" if MiWi™ P2P protocol is used, or section "MAC Function Description" in IEEE 802.15.4 specification if MiWi™ protocol is used.
   - If PIC18 Explorer, 8-bit Wireless demo board or Explorer 16 demo board is used in the demo, critical information will be displayed on the LCD of the demo board. The demo name, RF transceiver and node number will be displayed first, followed by the information of active scan during the process. Once the active scan is finished and the device has been connected to the peer, the connected information and the channel number will be displayed for a short time before the demo instruction being displayed: button 1 for broadcast. For Explorer 16 demo board, button 2 can also be used to wake up the device and send encrypted unicast message.

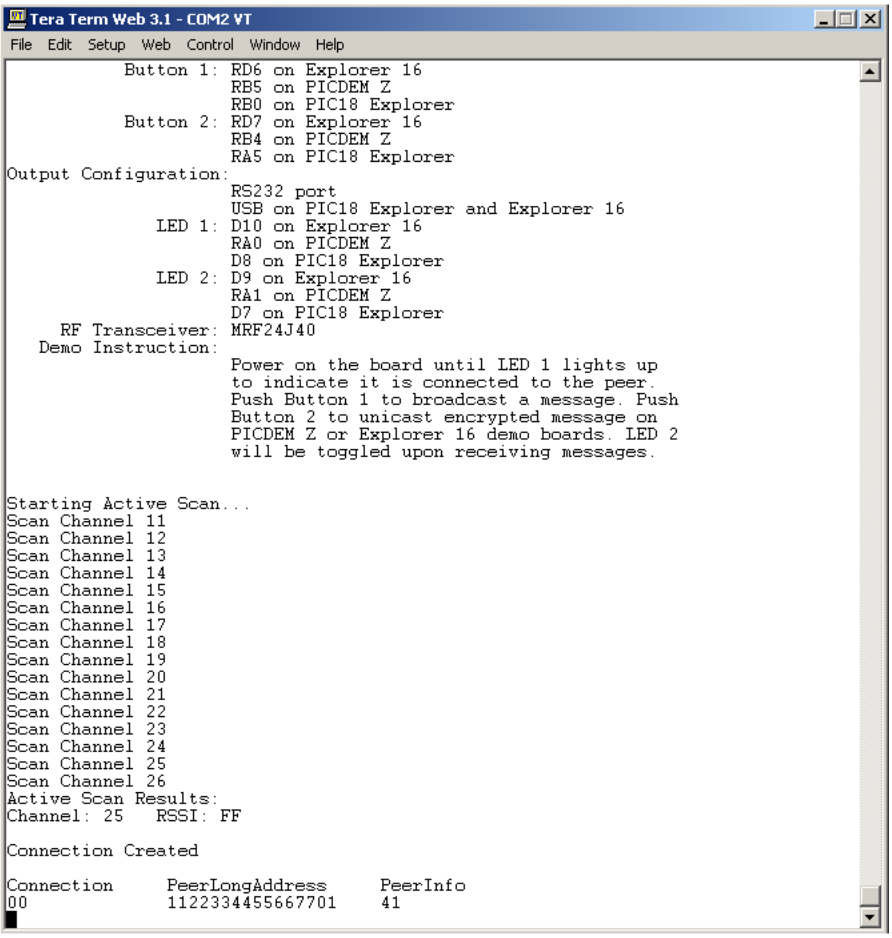


- If MRF24J40 transceiver is demonstrated and ZENA network analyzer is used and the monitor channel is the operating channel of the PAN, you should be able to see a broadcast packet. That is an active scan from node 2. Then node 1 responds to the active scan by unicast a packet to node 2. You will then see the 5-byte acknowledgement from node 1 to node 2. A few seconds later, a broadcast packet from node 2 initiates the hand-shaking process with node 1. You will see that node 1 responds with a unicast message to node 2 and node 2 acknowledge the unicast packet from node 1. After the hand-shaking procedure is done, a connection has been established. You will then see that node 2 sends out a Data Request command to node 1 about every 8 seconds to retrieve possible message from node 1 to node 2. Node 1 will acknowledge the Data Request command and later unicast an empty packet, if there is no message to node 2.



- If RS232 serial port or USB cable has been connected, you should see from the hyper terminal that node 2 does an active scan first and then prints out all operating MiWi™ P2P or MiWi™ PAN on the screen. Later, node 2 will choose the PAN that matches its own PAN identifier and starts the hand-shaking procedure. After the hand-shaking procedure has been completed, the LED1 lights up and the information about its peer device, node 1, will be printed out on the hyper terminal.

1



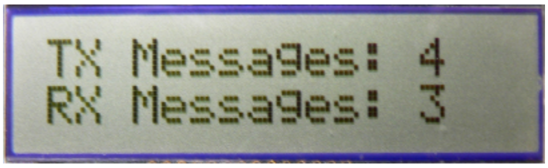
6. Press button 1 or button 2 on node 2 will wake up node 2 immediately from sleeping mode and send a broadcast or encrypted unicast message, just as the way that has been demonstrated in the simple example application. Because of demo board hardware design, for PIC18 Explorer demo board, only button 1 is able to wake up the the node 2 device and send out data. Pressing button 2 on PIC18 Explorer demo board will not be able to wake up the node 2 and sending out data.

- If PIC18 Explorer, 8-bit Wireless demo board or Explorer 16 demo boards are used, the total number of transmitted and received messages will be displayed on the LCD of the demo board.

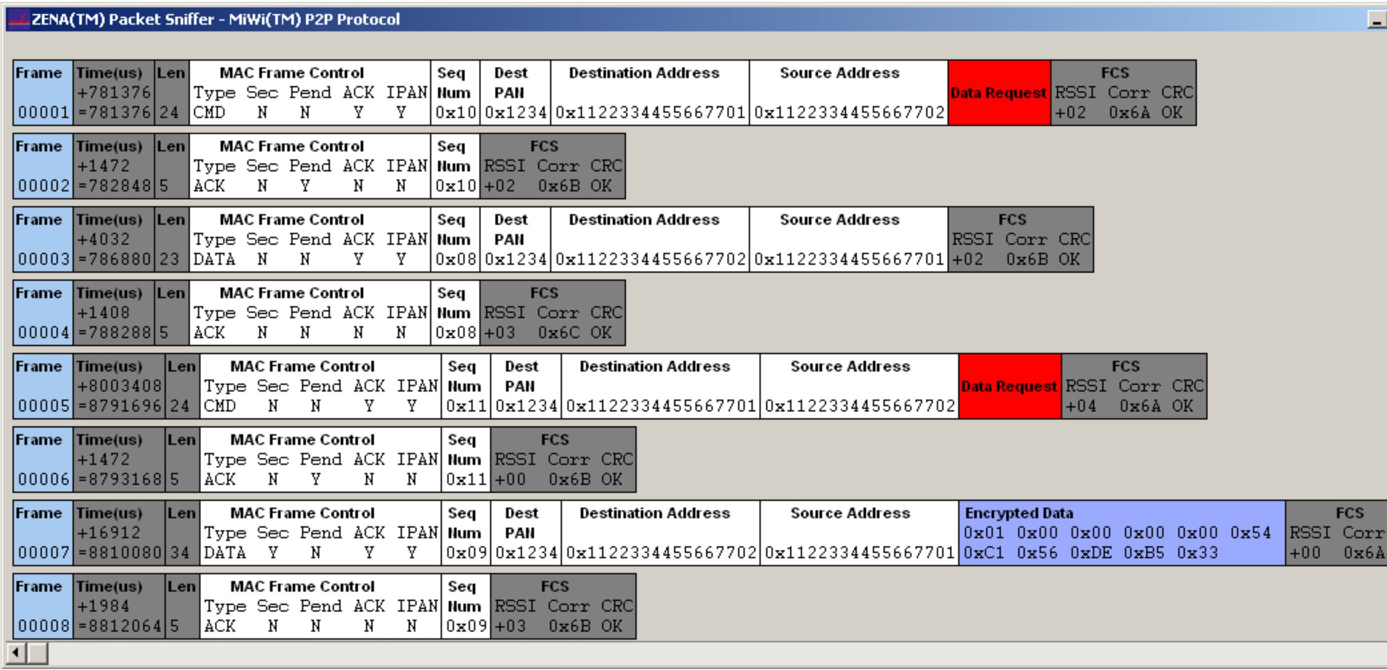
7. Press button 2 on node 1 will send a secured unicast message to node 2.

- Since node 2 is a RFD device that sleeps during idle, the message will not be delivered immediately. This kind of message is defined as indirect message. When node 2 wakes up next time within about 8 seconds, the indirect message will be delivered after the node 2 sends out Data Request command. After node 2 receives the indirect message, it will toggle its LED2. For details of indirect message, please refer to sections “Idle Devices Turning Off Radios” in Microchip application note AN1204 “Microchip MiWi™ P2P Wireless Protocol” if MiWi™ P2P protocol is used, or section "MAC Function Description" in IEEE 802.15.4 specification if MiWi™ protocol is used.

- If PIC18 Explorer, 8-bit Wireless demo board or Explorer 16 demo board is used in the demo, the delivery of indirect message to the sleeping device will be displayed on the LCD on both demo boards. Be aware that due to the delay in delivery indirect message, there is also a delay between the sender update number of the transmitted messages and receiver update the number of received messages.

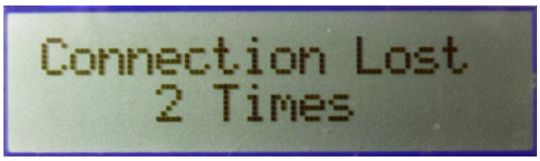


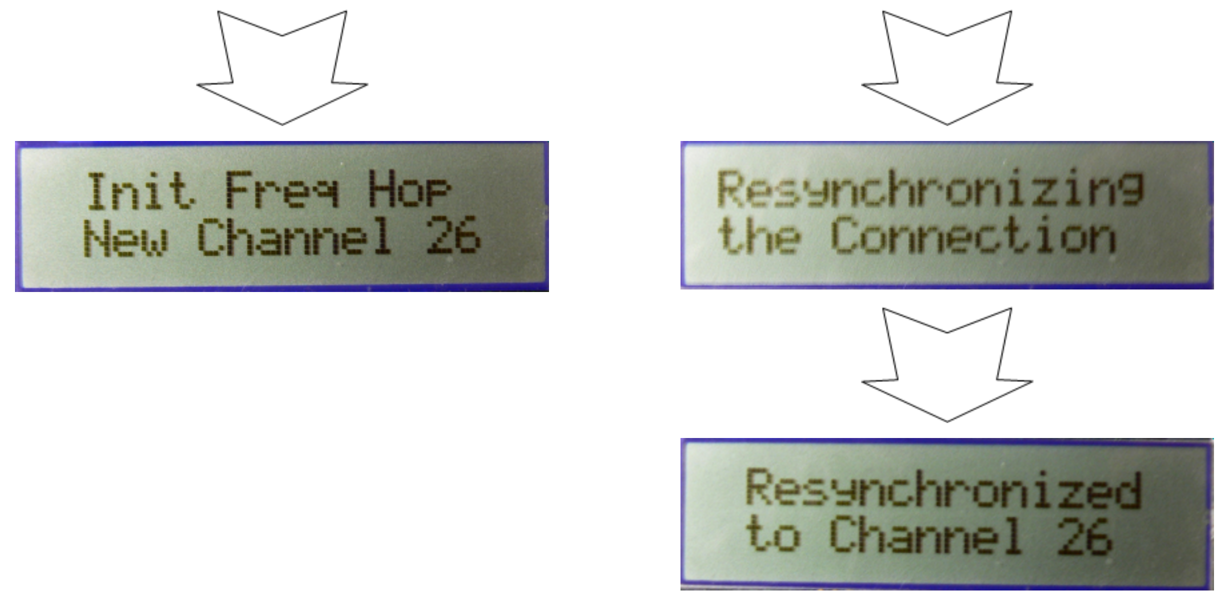
- If MRF24J40 transceiver is demonstrated and ZENA network analyzer is used, you should be able to see that the message is delivered to node 2 followed by a Data Request command.



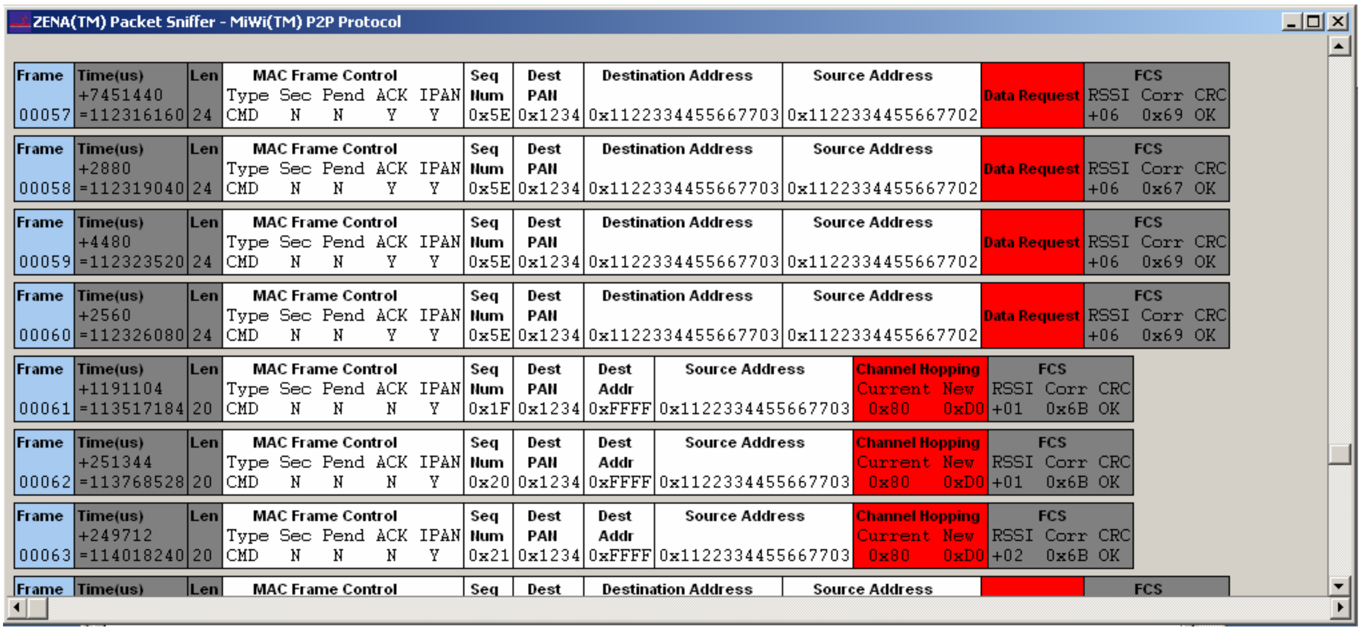
- If RS232 serial port or USB cable has been connected on node 2, you should able to see the printout of the received packet, as we have discussed in simple example application.

8. Press button 1 (RB5 on PICDEM Z, RB0 on PIC18 Explorer or RD6 on Explorer 16) on node 1 will initiate the frequency agility feature of MiWi™ P2P or MiWi™ stack. For details of frequency agility, please refer to sections “Frequency Agility” in Microchip application note AN1204 “Microchip MiWi™ P2P Wireless Protocol” if MiWi™ P2P protocol is used.
    - In this step, node 1 will start the frequency agility procedure. In this demo, node 1 is the frequency agility starter and node 2 is the frequency agility follower. Node 1 will first do an energy scan on all channels other than the current operating one to ensure a channel hopping. Then node 1 will choose the channel with least noise as the next operating channel for the PAN and broadcast the channel hopping command with the next operating channel to all devices that can hear. For those devices that do not hear the channel hopping command, such as a RFD like node 2 with its radio off during idle, a resynchronization procedure will start to re-establish the connection after transmission failures occur predefined times,.
    - If PIC18 Explorer, 8-bit Wireless demo board or Explorer 16 demo board is used in the demo, node 1 will display the frequency hopping process and the new channel that node 1 has hopped to. Node 2, on the other hand, will display the increasing number of losing connection with peer until it starts to re-synchronize the connection and its hop to the new channel of the peer.

Frequency Hopping Initiator	Frequency Hopping Follower
	

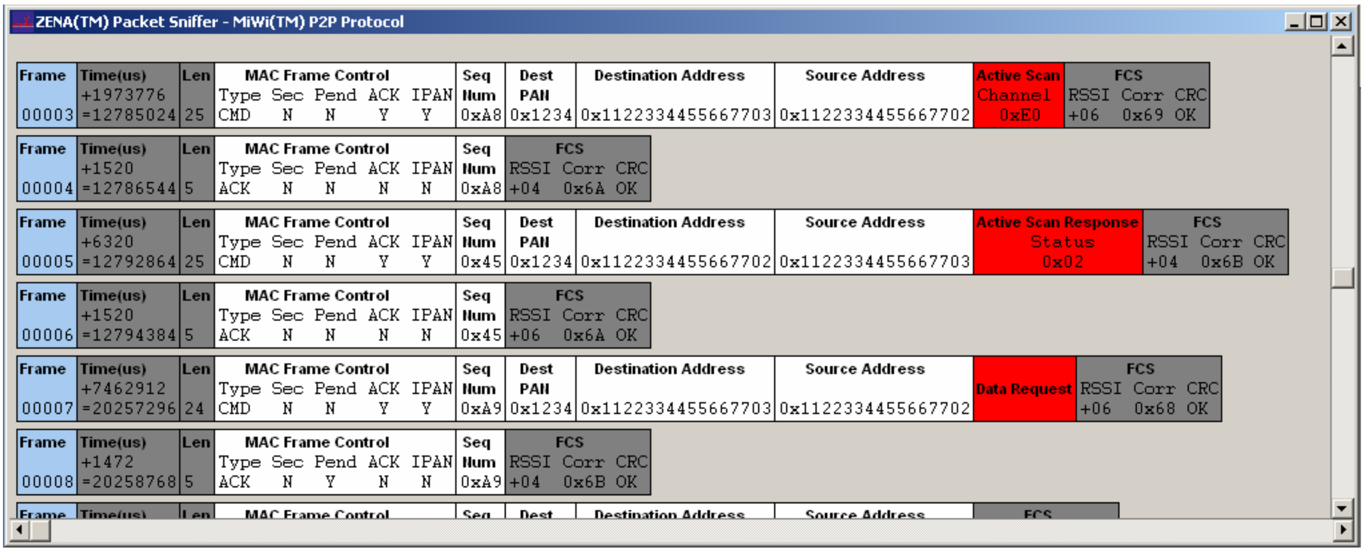


- If MRF24J40 transceiver is demonstrated and ZENA network analyzer is used, you should be able to see that the Data Request command from node 2 fails after button 1 of node 1 is pressed. Later, you will later see a broadcast channel hopping command from node 1.

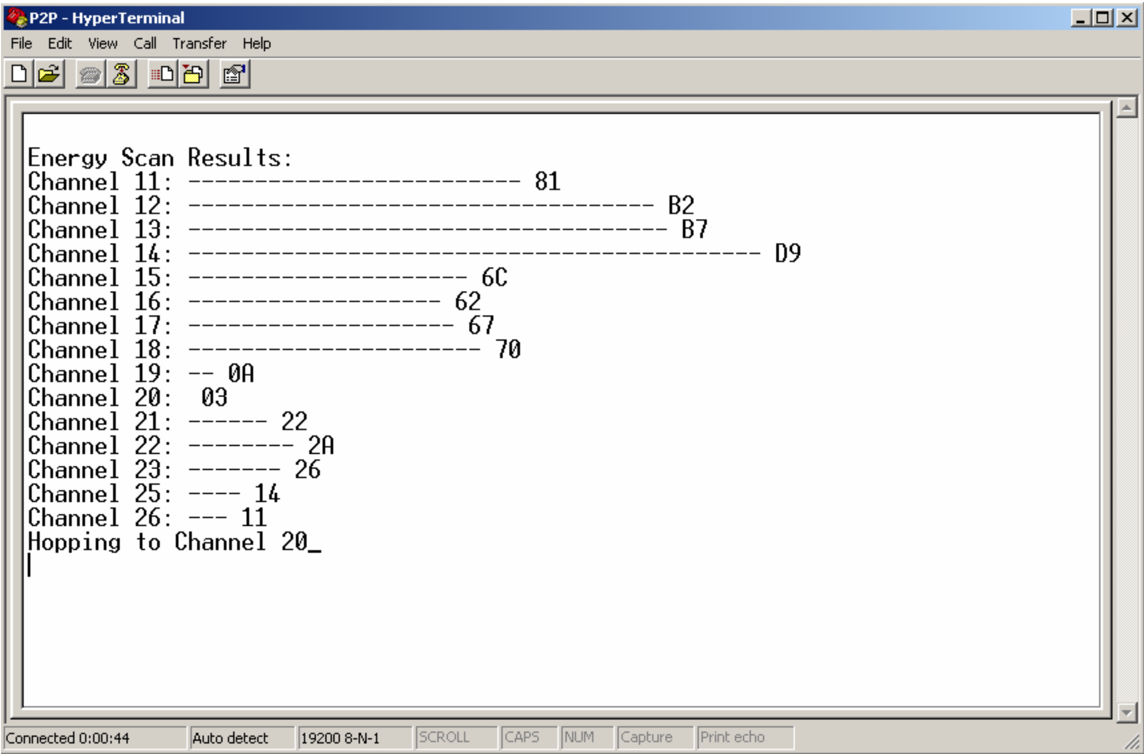


- On the new operating channel, a resynchronization command will be sent from node 2. Node 1 will respond to the resynchronization command. After that, you can see normal Data Request procedure going on at the new operating channel.

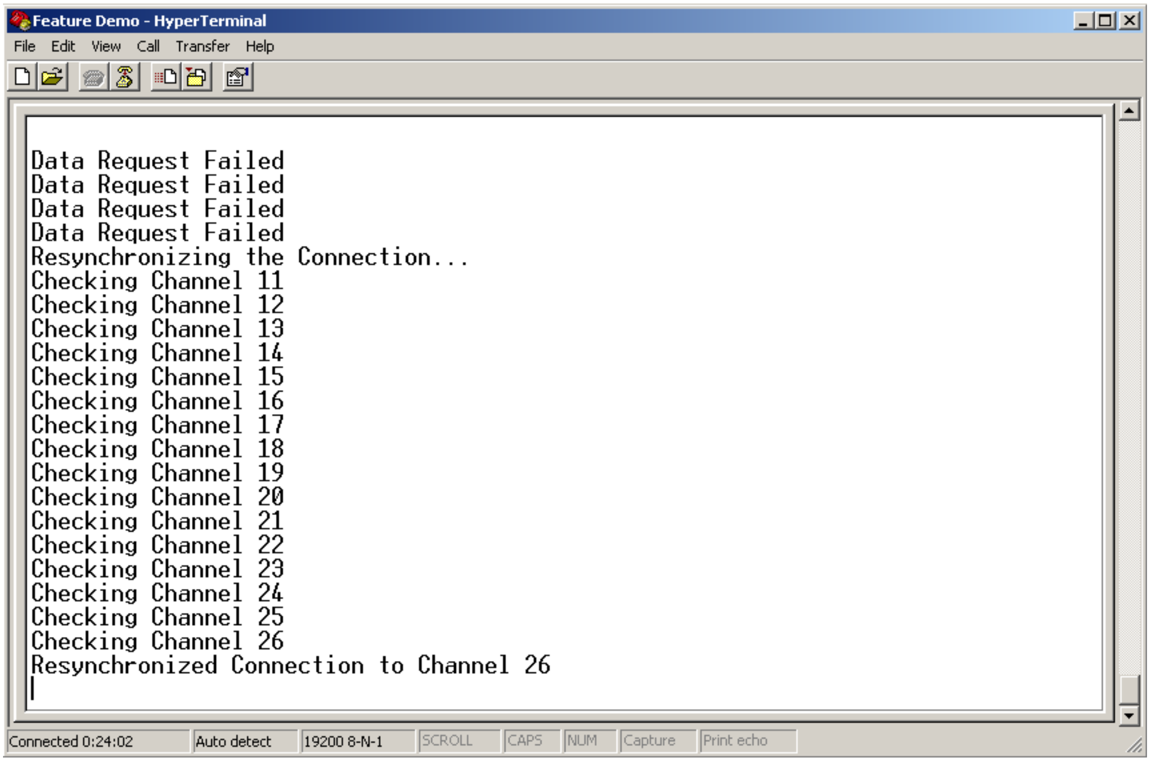
1



- If RS232 serial port or USB has been connected on node 1, you should able to see the energy reading on each channel. Finally, node 1 will change to the channel with lowest energy reading.



- If RS232 serial port or USB has been connected on node 2, you should be able to see the printout of Data Request failure for three times. After the failure for the fourth time, the resynchronization procedure will start and finally the node 2 will be resynchronized to the channel that node 1 chose before.


Feature Demo - HyperTerminal
File Edit View Call Transfer Help
Data Request Failed
Data Request Failed
Data Request Failed
Data Request Failed
Resynchronizing the Connection...
Checking Channel 11
Checking Channel 12
Checking Channel 13
Checking Channel 14
Checking Channel 15
Checking Channel 16
Checking Channel 17
Checking Channel 18
Checking Channel 19
Checking Channel 20
Checking Channel 21
Checking Channel 22
Checking Channel 23
Checking Channel 24
Checking Channel 25
Checking Channel 26
Resynchronized Connection to Channel 26
Connected 0:24:02
Auto detect
19200 8-N-1
SCROLL
CAPS
NUM
Capture
Print echo

Index

S

T

U

W