Contents | Index

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. <u>Application</u> layer uses MiApp interfaces to talk to the protocol layer. In protocol layer, there are implementations of MiWi[™] P2P, MiWi and MiWi PRO 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.



MiWiTM DE version 4.x is built on top of earlier version 3.1.4. The details of the major modifications can be found in the Release Notes.

Introduction

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index

SW License Agreement

MICROCHIP IS WILLING TO LICENSE THE ACCOMPANYING SOFTWARE AND DOCUMENTATION TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF THE FOLLOWING TERMS. TO ACCEPT THE TERMS OF THIS LICENSE, CLICK "I ACCEPT" AND PROCEED WITH THE DOWNLOAD OR INSTALL. IF YOU DO NOT ACCEPT THESE LICENSE TERMS, CLICK "I DO NOT ACCEPT," AND DO NOT DOWNLOAD OR INSTALL THIS SOFTWARE.

NON-EXCLUSIVE SOFTWARE LICENSE AGREEMENT

This Nonexclusive Software License Agreement ("Agreement") is a contract between you, your heirs, successors and assigns ("Licensee") and Microchip Technology Incorporated, a Delaware corporation, with a principal place of business at 2355 W. Chandler Blvd., Chandler, AZ 85224-6199, and its subsidiary, Microchip Technology (Barbados) II Incorporated (collectively, "Microchip") for the accompanying Microchip software including, but not limited to, Graphics Library Software, IrDA Stack Software, MCHPFSUSB Stack Software, Memory Disk Drive File System Software, mTouch(TM) Capacitive Library Software, Smart Card Library Software, TCP/IP Stack Software, MiWi(TM) DE Software, Security Package Software, and/or any PC programs and any updates thereto (collectively, the "Software"), and accompanying documentation, including images and any other graphic resources provided by Microchip ("Documentation").

1. <u>Definitions</u>. As used in this Agreement, the following capitalized terms will have the meanings defined below:

a. "Microchip Products" means Microchip microcontrollers and Microchip digital signal controllers.

b. "Licensee Products" means Licensee products that use or incorporate Microchip Products.

c. "Object Code" means the Software computer programming code that is in binary form (including related documentation, if any), and error corrections, improvements, modifications, and updates.

d. "Source Code" means the Software computer programming code that may be printed out or displayed in human readable form (including related programmer comments and documentation, if any), and error corrections, improvements, modifications, and updates.

e. "Third Party" means Licensee's agents, representatives, consultants, clients, customers, or contract manufacturers.

f. "Third Party Products" means Third Party products that use or incorporate Microchip Products.

2. Software License Grant. Microchip grants strictly to Licensee a non-exclusive, non-transferable, worldwide license to:

a. use the Software in connection with Licensee Products and/or Third Party Products;

b. if Source Code is provided, modify the Software; provided that Licensee clearly notifies Third Parties regarding the source of such modifications;

c. distribute the Software to Third Parties for use in Third Party Products, so long as such Third Party agrees to be bound by this Agreement (in writing or by "click to accept") and this Agreement accompanies such distribution;

d. sublicense to a Third Party to use the Software, so long as such Third Party agrees to be bound by this Agreement (in writing or by "click to accept");

e. with respect to the TCP/IP Stack Software, Licensee may port the ENC28J60.c, ENC28J60.h, ENCX24J600.c, and ENCX24J600.h driver source files to a non-Microchip Product used in conjunction with a Microchip ethernet controller;

f. with respect to the MiWi (TM) DE Software, Licensee may only exercise its rights when the Software is embedded on a Microchip Product and used with a Microchip radio frequency transceiver or UBEC UZ2400 radio frequency transceiver which are integrated into Licensee Products or Third Party Products.

For purposes of clarity, Licensee may NOT embed the Software on a non-Microchip Product, except as described in this Section.

3. Documentation License Grant. Microchip grants strictly to Licensee a non-exclusive, non-transferable, worldwide license to use the Documentation in support of Licensee's authorized use of the Software

4. Third Party Requirements. Licensee acknowledges that it is Licensee's responsibility to comply with any third party license terms or requirements applicable to the use of such third party software, specifications, systems, or tools. This includes, by way of example but not as a limitation, any standards setting organizations requirements and, particularly with respect to the Security Package Software, local encryption laws and requirements. Microchip is not responsible and will not be held responsible in any manner for Licensee's failure to comply with such applicable terms or requirements. 5. Open Source Components. Notwithstanding the license grant in Section 1 above, Licensee further acknowledges that certain components of the Software may be covered by so-called "open source" software licenses ("Open Source Components"). Open Source Components means any software licenses approved as open source licenses by the Open Source Initiative or any substantially similar licenses, including without limitation any license that, as a condition of distribution of the software licensed under such license, requires that the distributor make the software available in source code format. To the extent required by the licenses covering Open Source Components, the terms of such license will apply in lieu of the terms of this Agreement. To the extent the terms of the licenses applicable to Open Source Components prohibit any of the restrictions in this Agreement with respect to such Open Source Components, such restrictions will not apply to such Open Source Component.

6. Licensee Obligations. Licensee will not: (a) engage in unauthorized use, modification, disclosure or distribution of Software or Documentation, or its derivatives; (b) use all or any portion of the Software, Documentation, or its derivatives except in conjunction with Microchip Products, Licensee Products or Third Party Products; or (c) reverse engineer (by disassembly, decompilation or otherwise) Software or any portion thereof. Licensee may not remove or alter any Microchip copyright or other proprietary rights notice posted in any portion of the Software or Documentation. Licensee will defend, indemnify and hold Microchip and its subsidiaries harmless from and against any and all claims, costs, damages, expenses (including reasonable attorney's fees), liabilities, and losses, including without limitation: (x) any claims directly or indirectly arising from or related to the use, modification, disclosure or distribution of the Software, Documentation, or any intellectual property rights related thereto; (y) the use, sale and distribution of Licensee Products or Third Party Products; and (z) breach of this

Agreement.

7. Confidentiality. Licensee agrees that the Software (including but not limited to the Source Code, Object Code and library files) and its derivatives, Documentation and underlying inventions, algorithms, know-how and ideas relating to the Software and the Documentation are proprietary information belonging to Microchip and its licensors ("Proprietary Information"). Except as expressly and unambiguously allowed herein, Licensee will hold in confidence and not use or disclose any Proprietary Information and will similarly bind its employees and Third Party(ies) in writing. Proprietary Information will not include information that: (i) is in or enters the public domain without breach of this Agreement and through no fault of the receiving party; (ii) the receiving party was legally in possession of prior to receiving it; (iii) the receiving party can demonstrate was developed by the receiving party independently and without use of or reference to the disclosing party's Proprietary Information; or (iv) the receiving party receives from a third party without restriction on disclosure. If Licensee is required to disclose Proprietary Information by law, court order, or government agency, License will give Microchip prompt notice of such requirement in order to allow Microchip to object or limit such disclosure. Licensee agrees that the provisions of this Agreement regarding unauthorized use and nondisclosure of the Software. Documentation and related Proprietary Rights are necessary to protect the legitimate business interests of Microchip and its licensors and that monetary damage alone cannot adequately compensate Microchip or its licensors if such provisions are violated. Licensee, therefore, agrees that if Microchip alleges that Licensee or Third Party has breached or violated such provision then Microchip will have the right to injunctive relief, without the requirement for the posting of a bond, in addition to all other remedies at law or in equity.

8. Ownership of Proprietary Rights. Microchip and its licensors retain all right, title and interest in and to the Software and Documentation including, but not limited to all patent, copyright, trade secret and other intellectual property rights in the Software, Documentation, and underlying technology and all copies and derivative works thereof (by whomever produced). Licensee and Third Party use of such modifications and derivatives is limited to the license rights described in this Agreement.

9. Termination of Agreement. Without prejudice to any other rights, this Agreement terminates immediately, without notice by Microchip, upon a failure by Licensee or Third Party to comply with any provision of this Agreement. Upon termination, Licensee and Third Party will immediately stop using the Software, Documentation, and derivatives thereof, and immediately destroy all such copies.

10. Warranty Disclaimers. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE. MICROCHIP AND ITS LICENSORS ASSUME NO RESPONSIBILITY FOR THE ACCURACY, RELIABILITY OR APPLICATION OF THE SOFTWARE OR DOCUMENTATION. MICROCHIP AND ITS LICENSORS DO NOT WARRANT THAT THE SOFTWARE WILL MEET REQUIREMENTS OF LICENSEE OR THIRD PARTY, BE UNINTERRUPTED OR ERROR-FREE. MICROCHIP AND ITS LICENSORS HAVE NO OBLIGATION TO CORRECT ANY DEFECTS IN THE SOFTWARE.

11. Limited Liability. IN NO EVENT WILL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER ANY LEGAL OR EQUITABLE THEORY FOR ANY DIRECT OR INDIRECT DAMAGES OR EXPENSES INCLUDING BUT NOT LIMITED TO INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS. The aggregate and cumulative liability of Microchip and its licensors for damages hereunder will in no event exceed \$1000 or the amount Licensee paid Microchip for the Software and Documentation, whichever is greater. Licensee acknowledges that the foregoing limitations are reasonable and an essential part of this Agreement.

12. General, THIS AGREEMENT WILL BE GOVERNED BY AND CONSTRUED UNDER THE LAWS OF THE STATE OF ARIZONA AND THE UNITED STATES WITHOUT REGARD TO CONFLICTS OF LAWS PROVISIONS. Licensee agrees that any disputes arising out of or related to this Agreement, Software or Documentation will be brought exclusively in either the U.S. District Court for the District of Arizona, Phoenix Division, or the Superior Court of Arizona located in Maricopa County, Arizona. This Agreement will constitute the entire agreement between the parties with respect to the subject matter hereof. It will not be modified except by a written agreement signed by an authorized representative of Microchip. If any provision of this Agreement will be held by a court of competent jurisdiction to be illegal, invalid or unenforceable, that provision will be limited or eliminated to the minimum extent necessary so that this Agreement will otherwise remain in full force and effect and enforceable. No waiver of any breach of any provision of this Agreement will constitute a waiver of any prior, concurrent or subsequent breach of the same or any other provisions hereof, and no waiver will be effective unless made in writing and signed by an authorized representative of the waiving party. Licensee agrees to comply with all import and export laws and restrictions

and regulations of the Department of Commerce or other United States or foreign agency or authority. The indemnities, obligations of confidentiality, and limitations on liability described herein, and any right of action for breach of this Agreement prior to termination, will survive any termination of this Agreement. Any prohibited assignment will be null and void. Use, duplication or disclosure by the United States Government is subject to restrictions set forth in subparagraphs (a) through (d) of the Commercial Computer-Restricted Rights clause of FAR 52.227-19 when applicable, or in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, and in similar clauses in the NASA FAR Supplement. Contractor/manufacturer is Microchip Technology Inc., 2355 W. Chandler Blvd., Chandler, AZ 85224-6199.

If Licensee has any questions about this Agreement, please write to Microchip Technology Inc., 2355 W. Chandler Blvd., Chandler, AZ 85224-6199 USA. ATTN: Marketing.

Copyright (c) 2012 Microchip Technology Inc. All rights reserved.

License Rev. No. 05-012412

SW License Agreement

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Release Notes

Microchip MiWi(TM) Development Environment Software Stack

Version 4.x, June 8 2011

Microchip MiWi[™] Development Environment (MiWi[™] DE) protocol stack provides simple wireless connectivity for shortrange, 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 MiWi[™] DE 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 "MiWi DE Demo". In addition, the directory "Microchip" is for MiWi[™] DE stack source code.

Micorchip MiWi^m DE version 4.x is updated from version 3.1.4 released earlier. The main updates from earlier versions are:

- Support MiWi[™] PRO networking protocol
- Support Microchip MPLAB X
- Provide demo source code for Microchip Wireless Development Kit

- Provide testing interface for MiWi PRO protocol
- Bug fixes

For all new features, please refer to the section New Features.

Peripherals

Type/Use	Specific/Configurable	Limitations
UART for hyper terminal output	Select via console configuration in ConfigApp.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 configurations in ConfigApp.h	Both hardware SPI or software bit-bang can be used.
Digital I/O pins to RF transceiver	Select via pin configurations in ConfigApp.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 compiler version 3.30 or earlier has a mismatch in the memory mapping of linker script for PIC18F87J11 family. It is highly recommended to use the linker script in the project file when compile the stack for PIC18 Explorer demo board.
- 2. Due to C30 (version 3.11) compiler limitation, using optimization

level 2 or above in compilation may reset the MCU when decrypt a message, when MRF49XA is chosen as the RF transceiver. Please use no optimization or optimization level 1 to compiler the MiWi[™] P2P code as MRF49XA is used.

- 3. High data rate for <u>MRF49XA</u> may require MCU running at faster speed. This is due to the nature of 16-bit RX buffer used in <u>MRF49XA</u>.
- When PIC32 MCU family is used, the demo needs to be compiled with optimization on to track the delay timing for LCD. The MiWi[™] P2P stack itself does not have such requirement.
- 5. In the feature demo, only button 1 can wake up the end device to transmit message on PIC18 Explorer demo board, due to the demo board hardware design.

Release Notes

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

New Features

New Features

This release of MiWi[™] DE has included a few new features, such as Network Freezer, Enhanced Data Request and Time Synchronization. Those features are so new that we don't have a chance to modify the application note to document them. In this section, we describe those new features in details to help our user understanding and using them. All new features can be enabled in the feature demo, which is included in the release package.

Topics

Name	Description
Network Freezer	
<u>Enhanced Data</u> <u>Request</u>	
Time Synchronization	

New Features

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

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/MiWi PRO protocols, which use 16bit network address in communication. After the power cycle, a wireless node in MiWi/MiWi PRO 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 <u>ENABLE_NETWORK_FREEZER</u> in configuration file of application layer: ConfigApp.h. In the demo, this feature has been enabled.

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.

<u>New Features > Network Freezer</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

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/protocol header transmission and put the device into sleep earlier. Our tests show that the total active time could be lowered up to 20-30% in certain usage case.

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 <u>MiApp_TransceiverPowerState</u> with parameter <u>POWER_STATE_WAKEUP_DR</u>. 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 <u>MiApp_TransceiverPowerState</u> with parameter <u>POWER_STATE_WAKEUP_DR</u>. 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 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.

<u>New Features > Enhanced Data Request</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

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, each sleeping device is required to report in a predefined interval. 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. The indirect message has been changed to the following format:

Name M Com	IAC Rough Imand Timing Info	Precise Timing Info	Indirect Message
---------------	-----------------------------------	---------------------------	---------------------

Length (<mark>BYTE</mark>)	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, ConfigMiWi.h or ConfigMiWiPRO.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, <u>TIME_SYNC_SLOTS</u>, supported in the wake up interval of sleeping devices. As the rule, the <u>TIME_SYNC_SLOTS</u> 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 <u>ENABLE_TIME_SYNC</u>, <u>TIME_SYNC_SLOTS</u> is defined in protocol configuration files ConfigP2P.h, ConfigMiWi.h or ConfigMiWiPRO.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.

New Features > Time Synchronization

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Demos

Topics

Name	Description
Required Hardware	To run this project, you will need two wireless nodes. Each of the nodes can be any of setups of listed hardware.
<u>Configuring the</u> <u>Hardware</u>	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, RF Card and PIC24FJ128GA010 or PIC32MX360F512L PIM Configuration 4: 8-bit Wireless Development Kit
<u>Firmware</u>	
Running Demos	

Demos

Microchip My Application xx.yy - [Jan 1, 2009] Copyright O 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

Required Hardware

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

Topics

Name

Description

Hardware Sets

Demos > Required Hardware

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Hardware Sets

Hardware Setups

Hardware Set 1:

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

Hardware Set 2:

- Demo Board:
 - PIC18 Explorer with PIC19F87J11 PIM (DM183032)
- RF Board:
 - <u>MRF24J40</u>
 - MRF24J40MA PICtail (AC164134-1)
 - <u>MRF49XA</u>
 - MRF49XA PICtail 434MHz (AC164137-1) OR
 - MRF49XA PICtail 868/915MHz (AC164137-2)
 - <u>MRF89XA</u>
 - MRF89XA PICtail 868MHz OR
 - MRF89XA PICtail 915MHz

Hardware Set 3:

- Demo Board:
 - Explorer 16 (DM240001)
 - PIC24FJ128GA010 Plug-In-Module (PIM) (MA240011)
- RF Borad
 - <u>MRF24J40</u>
 - <u>PICDEM Z MRF24J40 2.4GHz Daughter Card</u> (AC163027-4) OR
 - ¬ <u>MRF24J40MA PICtail™ Plus 2.4GHz RF Card</u> <u>(AC164134)</u> OR
 - MRF24J40MA PICtail (AC164134-1)
 - <u>MRF49XA</u>
 - <u>MRF49XA PICtail 434MHz (AC164137-1)</u> OR
 - MRF49XA PICtail 868/915MHz (AC164137-2)
 - <u>MRF89XA</u>
 - MRF89XA PICtail 868MHz OR
 - MRF89XA PICtail 915MHz

Hardware Set 4:

- Demo Board:
 - Explorer 16 (DM240001)
 - PIC32MX360F512L Plug-In-Module (PIM) (MA320001)
- RF Board:
 - <u>MRF24J40</u>
 - PICDEM Z MRF24J40 2.4GHz Daughter Card (AC163027-4) OR
 - ¬<u>MRF24J40MA PICtail™ Plus 2.4GHz RF Card</u> <u>(AC164134)</u> OR
 - <u>AMRF24J40MA PICtail (AC164134-1)</u>
 - <u>MRF49XA</u>
 - MRF49XA PICtail 434MHz (AC164137-1) OR
 - MRF49XA PICtail 868/915MHz (AC164137-2)
 - <u>MRF89XA</u>
 - MRF89XA PICtail 868MHz OR
 - MRF89XA PICtail 915MHz

Hardware Set 5:

- Demo Board:
 - 8-bit Wireless Development Board
- RF Board:
 - <u>MRF24J40</u>
 - MRF24J40MA PICtail (AC164134-1)
 - <u>MRF49XA</u>
 - MRF49XA PICtail 434MHz (AC164137-1) OR
 - MRF49XA PICtail 868/915MHz (AC164137-2)
 - <u>MRF89XA</u>
 - MRF89XA PICtail 868MHz OR
 - <u>MRF89XA</u> PICtail 915MHz

<u>Demos</u> > <u>Required Hardware</u> > <u>Hardware Sets</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

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, RF Card and PIC24FJ128GA010 or PIC32MX360F512L PIM

Configuration 4: 8-bit Wireless Development Kit

Topics

Name	Description
PICDEM Z	
PIC18 Explorer	
Explorer 16	
<u>8-bit Wireless</u> Development Kit	

Demos > Configuring the Hardware

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

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.

COM	2 Properties			? ×
Po	rt Settings			
	Bits per second:	19200		•
	Data bits:	8		•
	Parity:	None		•
	Stop bits:	1		•
	Flow control:	None		
			Restore D)efaults
_	0	К	Cancel	Apply

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.



Demos > Configuring the Hardware > PICDEM Z

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.



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 <u>MRF24J40</u>, <u>MRF49XA</u> or <u>MRF89XA</u> 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

Found New Hardware Wizard	
	Completing the Found New Hardware Wizard The wizard has finished installing the software for:
	<back cencel<="" finish="" td=""></back>

- 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.

COM2	Properties			<u>? ×</u>
Por	t Settings			
Г				
	Bits per second:	19200		
	Data bits:	8		•
	Parity:	None		•
	Stop bits:	1		•
	Flow control:	None		
L			Restore	Defaults
	0	к	Cancel	Apply

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.

<u>Demos</u> > <u>Configuring the Hardware</u> > <u>PIC18 Explorer</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

Explorer 16

Configuration 3: Explorer 16

1) 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 <u>MRF24J40</u> or <u>MRF49XA</u> 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.

COM	12 Properties			? ×
Po	ort Settings			
	Bits per second:	19200		
	Data bits:	8		•
	Parity:	None		•
	Stop bits:	1		•
	Flow control:	None		•
			Restor	e Defaults
	0	K	Cancel	Apply

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.

<u>Demos</u> > <u>Configuring the Hardware</u> > <u>Explorer 16</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

8-bit Wireless Development Kit

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.





<u>Demos</u> > <u>Configuring the Hardware</u> > <u>8-bit Wireless Development Kit</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright O 2009 Microchip Technology, Inc. All rights reserved.

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.

Topics

Name	Description
Precompiled HEX Files	
Demo Source Code Project for MPLAB 8.x	
Demo Source Code Project for MPLAB X	 From this release, MiWi DE starts to support Microchip IDE MPLAB X. Users have the option to convert a MPLAB 8.x project to MPLAB X, or use the MPLAB X project directly. To load project from MPLAB X, first select File -> Open Project. The pop up window will show up and you can browse to the demo directory. Choose MPLAB.X directory and open the MPLAB X project directly, as shown below.



<u>Demos</u> > <u>Firmware</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Precompiled HEX Files

Precompiled HEX Files

Precompiled **Demos** are available in the following directories:

- <Install Directory>\MiWi DE Demo\Basic <u>Demos</u>\Simple Example\Node 1\Precompiled HEX
- <Install Directory>\MiWi DE Demo\Basic <u>Demos</u>\Simple Example\Node 2\Precompiled HEX
- <Install Directory>\MiWi DE Demo\Basic <u>Demos</u>\Feature Demo\Node 1\Precompiled HEX
- <Install Directory>\MiWi DE Demo\Basic <u>Demos</u>\Feature Demo\Node 2\Precompiled HEX

A hex file is provided for each hardware configuration and each available RF transceiver under the above directories. Import the corresponding hex file and then program the hex to the demo board.

<u>Demos</u> > <u>Firmware</u> > <u>Precompiled HEX Files</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Demo Source Code Project for MPLAB 8.x

Demo Source Code Modification

The source code for this demo is available in the following directories:

- <Install Directory>\MiWi DE Demo\Basic <u>Demos</u>\Simple Example\Node 1
- <Install Directory>\MiWi DE Demo\Basic <u>Demos</u>\Simple Example\Node 2
- <Install Directory>\MiWi DE Demo\Basic <u>Demos</u>\Feature Demo\Node 1
- <Install Directory>\MiWi DE Demo\Basic <u>Demos</u>\Feature Demo\Node 2
- <Install Directory>\MiWi DE Demo\MiWi PRO Test Interface

In above 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:

MPLAB IDE v8.10	Open Project	? ×
File Edit View Project Debugger Project Project Wizard New New Open Close Set Active Project	Dogrammer Toc Look in: È P2P Node 1 Image: Strate Demo P2P - P1C18 - C18 Not Petture Demo P2P - P1C24-dsP1C33 Image: Strate Demo P2P - P1C24 - C12 Not Simple Example P2P - P1C34 - C13 Not Image: Strate Demo P2P - P1C34 - C13 Not Simple Example P2P - P1C34 - C13 Not Image: Strate Demo P2P - P1C34 - C13 Not Simple Example P2P - P1C34 - C13 Not Image: Strate Demo P2P - P1C34 - C13 Not Simple Example P2P - P1C34 - C13 Not Image: Strate Demo P2P - P1C34 - C13 Not Simple Example P2P - P1C34 - C13 Not Image: Strate Demo P2P - P1C34 - C13 Not Simple Example P2P - P1C34 - C13 Not Image: Strate Demo P2P - P1C34 - C13 Not Simple Example P2P - P1C34 - C13 Not	C30 Node 1.mcp le 1.mcp de 1.mcp de 1.mcp de 1.mcp de 1.mcp
	File name: Simple Example P2P - PII Files of type: MPLAB IDE Project Files	C18 - C18 Node 1.mcp Open (".mcp) Cancel

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 8-bit Wireless Development Kit 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 initially 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.

Topics

Name	Description
MiWi P2P	
MiWi Mesh	
<u>MiWi PRO</u>	test

Demos > Firmware > Demo Source Code Project for MPLAB 8.x

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

MiWi P2P

Topics

Name	Description
<u>PICDEM Z Demo</u> Board for MiWi P2P	
<u>PIC18 Explorer Demo</u> <u>Board for MiWi P2P</u>	
<u>8-bit Wireless</u> <u>Development Kit for</u> <u>MiWi P2P</u>	
Explorer 16 Demo Board for MiWi P2P	

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi P2P

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Only <u>MRF24J40</u> is supported for this demo board. Uncomment "#define <u>MRF24J40</u>" and comment out all other transceiver definition.

From the project window, choose to edit file "HardwareProfile.h"

under the directory "Header Files -> <u>Application</u>", 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.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi

P2P > PICDEM Z Demo Board for MiWi P2P

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi P2P > PIC18 Explorer Demo Board for MiWi P2P

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

8-bit Wireless Development Kit for MiWi P2P

8-bit Wireless Development Kit 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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

<u>Demos</u> > <u>Firmware</u> > <u>Demo Source Code Project for MPLAB 8.x</u> > <u>MiWi</u> <u>P2P</u> > <u>8-bit Wireless Development Kit for MiWi P2P</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

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.

Topics

Name	Description	
PIC24 or dsPIC33 for MiWi P2P	PIC24 or dsPIC33 on Explorer 16 Demo Board for MiWi [™] P2PExplorer 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. Source Files Header Files Header Files SystemProfile.h SystemProfile.h SystemProfile.h Protocols In file "ConfigApp.h", first elect to use MiWi P2P protocol, make sure to uncomment "#define PROTOCOL_P2P" and comment	

	out <u>more</u>
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. Source Files Header Files SystemProfile.h SystemProfile.h Protocols In file "ConfigApp.h", first elect to use MiWi P2P protocol, make sure to uncomment "#define PROTOCOL_P2P" and comment out "#define PROTOCOL_MIWI" and

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi P2P > Explorer 16 Demo Board for MiWi P2P

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi P2P > Explorer 16 Demo Board for MiWi P2P > PIC24 or dsPIC33 for MiWi P2P

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

<u>Demos</u> > <u>Firmware</u> > <u>Demo Source Code Project for MPLAB 8.x</u> > <u>MiWi</u> <u>P2P</u> > <u>Explorer 16 Demo Board for MiWi P2P</u> > <u>PIC32 for MiWi P2P</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

MiWi Mesh

Topics

Name	Description
<u>PICDEM Z Demo</u> <u>Board for MiWi</u>	
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. Source Files Header Files GorfigApp.h", first elect to use MiWi more
8-bit Wireless	8-bit Wireless Development Kit for

Development Kit for MiWi	 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. Source Files → Application as shown below. Source Files → Application as shown below. From the group of the state of the state
Explorer 16 Demo Board for MiWi	Explorer 16 Demo Board for MiWi™ Explorer 16 demo board support development for PIC24, dsPIC33 and PIC32.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi Mesh

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Only <u>MRF24J40</u> is supported for this demo board. Uncomment "#define <u>MRF24J40</u>" and comment out all other transceiver definition.

From the project window, choose to edit file "HardwareProfile.h"

under the directory "Header Files -> <u>Application</u>", 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.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi

Mesh > PICDEM Z Demo Board for MiWi

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.
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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi Mesh > PIC18 Explorer Demo Board for MiWi

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

8-bit Wireless Development Kit for MiWi

8-bit Wireless Development Kit 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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

<u>Demos</u> > <u>Firmware</u> > <u>Demo Source Code Project for MPLAB 8.x</u> > <u>MiWi</u> <u>Mesh</u> > <u>8-bit Wireless Development Kit for MiWi</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Explorer 16 Demo Board for MiWi

Explorer 16 Demo Board for MiWi™

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

Topics

Name	Description	
<u>PIC24 or dsPIC33 for</u> <u>MiWi</u>	 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. Source Files Header Files SystemProfile.h SystemProfile.h Protocols In file "ConfigApp.h", first elect to use MiWi protocol, make sure to uncomment "#define 	

	#define PROTOCOL_P2P more	
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. Source Files Header Files SystemProfile.h SystemProfile.h In file "ConfigApp.h", first elect to use MiWi protocol, make sure to uncomment "#define PROTOCOL_MIWI" and comment out "#define PROTOCOL_P2P" and "#define PROTOCOL_MIWI PRO more	
μ	μ	

<u>Demos</u> > <u>Firmware</u> > <u>Demo Source Code Project for MPLAB 8.x</u> > <u>MiWi</u> <u>Mesh</u> > <u>Explorer 16 Demo Board for MiWi</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

<u>Demos</u> > <u>Firmware</u> > <u>Demo Source Code Project for MPLAB 8.x</u> > <u>MiWi</u> <u>Mesh</u> > <u>Explorer 16 Demo Board for MiWi</u> > <u>PIC24 or dsPIC33 for MiWi</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

<u>Demos</u> > <u>Firmware</u> > <u>Demo Source Code Project for MPLAB 8.x</u> > <u>MiWi</u> <u>Mesh</u> > <u>Explorer 16 Demo Board for MiWi</u> > <u>PIC32 for MiWi</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

MiWi PRO

test

Topics

Name	Description
<u>PICDEM Z Demo</u> Board for MiWi PRO	
PIC18 Explorer Demo Board for MiWi PRO	
8-bit Wireless Development Kit for MiWi PRO	 8-bit Wireless Development Kit for MiWi™ PRO8-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. Source Files Header Files Header Files Header Files Header Files In file "ConfigApp.h", first elect to use MiWi PRO protocol, make sure to uncomment "#define PROTOCOL MIWL PRO" and

	comment out "#define <u>PROTOCOL_P2P</u> " and "#define <u>PROTOCOL_MIWI</u> more
Explorer 16 Demo Board for MiWi PRO	Explorer 16 Demo Board for MiWi™ Explorer 16 demo board support development for PIC24, dsPIC33 and PIC32.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi PRO

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

PICDEM Z Demo Board for MiWi PRO

PICDEM Z Demo Board for MiWi[™] PRO

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 -> <u>Application</u>", as shown below.



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

Second step, choose the RF transceiver to be used. Only <u>MRF24J40</u> is supported for this demo board. Uncomment "#define <u>MRF24J40</u>" and comment out all other transceiver definition.

From the project window, choose to edit file "HardwareProfile.h"

under the directory "Header Files -> <u>Application</u>", 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.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi

PRO > PICDEM Z Demo Board for MiWi PRO

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

PIC18 Explorer Demo Board for MiWi PRO

PIC18 Explorer Demo Board for MiWi™ PRO

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 -> <u>Application</u>", as shown below.



In file "ConfigApp.h", first elect to use MiWi PRO protocol, make sure to uncomment "#define PROTOCOL_PRO" and comment out "#define <u>PROTOCOL_MIWI</u>" and "#define <u>PROTOCOL_P2P</u>".

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi PRO > PIC18 Explorer Demo Board for MiWi PRO

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

8-bit Wireless Development Kit for MiWi PRO

8-bit Wireless Development Kit for MiWi[™] PRO

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 -> <u>Application</u>", as shown below.



In file "ConfigApp.h", first elect to use MiWi PRO protocol, make sure to uncomment "#define <u>PROTOCOL_MIWI_PRO</u>" and comment out "#define <u>PROTOCOL_P2P</u>" and "#define <u>PROTOCOL_MIWI</u>".

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

<u>Demos</u> > <u>Firmware</u> > <u>Demo Source Code Project for MPLAB 8.x</u> > <u>MiWi</u> <u>PRO</u> > <u>8-bit Wireless Development Kit for MiWi PRO</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Explorer 16 Demo Board for MiWi PRO

Explorer 16 Demo Board for MiWi™

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

Topics

Name	Description	
PIC24 or dsPIC33 for MiWi PRO	 PIC24 or dsPIC33 on Explorer 16 Demo Board for MiWi[™] PROExplorer 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. Source Files Header Files SystemProfile.h SystemProfile.h In file "ConfigApp.h", first elect to use MiWi PRO protocol, make sure to uncomment 	

	comment out more	
PIC32 for MiWi PRO	PIC32 on Explorer 16 Demo Board for MiWi™ PRO 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. Source Files Header Files Header Files Header Files Header Files Notice Protocols In file "ConfigApp.h", first elect to use MiWi PRO protocol, make sure to uncomment "#define PROTOCOL_MIWI_PRO" and comment out "#define PROTOCOL_P2P" and more	

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi PRO > Explorer 16 Demo Board for MiWi PRO

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

PIC24 or dsPIC33 for MiWi PRO

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

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 -> <u>Application</u>", as shown below.



In file "ConfigApp.h", first elect to use MiWi PRO protocol, make sure to uncomment "#define <u>PROTOCOL_MIWI_PRO</u>" and comment out "#define <u>PROTOCOL_P2P</u>" and "#define <u>PROTOCOL_MIWI</u>".

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

Demos > Firmware > Demo Source Code Project for MPLAB 8.x > MiWi PRO > Explorer 16 Demo Board for MiWi PRO > PIC24 or dsPIC33 for MiWi PRO

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

PIC32 for MiWi PRO

PIC32 on Explorer 16 Demo Board for MiWi[™] PRO

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 -> <u>Application</u>", as shown below.



In file "ConfigApp.h", first elect to use MiWi PRO protocol, make sure to uncomment "#define <u>PROTOCOL_MIWI_PRO</u>" and comment out "#define <u>PROTOCOL_P2P</u>" and "#define <u>PROTOCOL_MIWI</u>".

Second step, choose the RF transceiver to be used. Three RF transceivers: <u>MRF24J40</u>, <u>MRF49XA</u> and <u>MRF89XA</u> 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 -> <u>Application</u>", 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.

<u>Demos</u> > <u>Firmware</u> > <u>Demo Source Code Project for MPLAB 8.x</u> > <u>MiWi</u> <u>PRO</u> > <u>Explorer 16 Demo Board for MiWi PRO</u> > <u>PIC32 for MiWi PRO</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Previous | Up | Next

Demo Source Code Project for MPLAB X

From this release, MiWi DE starts to support Microchip IDE MPLAB X. Users have the option to convert a MPLAB 8.x project to MPLAB X, or use the MPLAB X project directly.

To load project from MPLAB X, first select File -> Open Project. The pop up window will show up and you can browse to the demo directory. Choose MPLAB.X directory and open the MPLAB X project directly, as shown below.



The single MPLAB X project supports multiple Microchip demo hardware. From the customization list, user can choose one of the hardware platforms that have been configured within MPLAB X project, as shown below

×	MPLAB X IDE Beta6.0		
File	Edit View Navigate Source	Run Debug Team Tools Window	Help
4	9 🚰 😫 🖳 🐚 🍘	MPLAB : PIC18_Explorer	18
1		MPLAB: PIC18_Explorer	
GS	Projects I X Classes	MPLAB : PICDEM_Z	Hard
	Feature Demo Node 1 Feature Demo Node 2	MPLAB : Eight_bit_Wireless_Development_Kit MPLAB : Explorer_16PIC24_or_dsPIC33 MPLAB : Explorer_16PIC32	2 😎
	🗈 🚍 Simple Example Node 1	Customize	ame -

In the source code, user still needs to open application configuration and choose the right transceiver as well as wireless protocol to fit their needs. User alsol needs to open hardware configuration file HardwareProfile.h to choose the correct demo board before a successful compilation can be done, as shown in following snapshot



Demos > Firmware > Demo Source Code Project for MPLAB X

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Running Demos

Running Demos

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

Topics

Name	Description
Basic Demos	
MiWi PRO Test Interface	MiWi PRO Test Interface A new project, "MiWi PRO Test Interface" has been added to this release of MiWi DE. This new project, driven by testing menus on hyper terminal, is the main interface for Microchip developer to verify the functionalities of MiWi PRO protocol stack. Users, on the other hand, may benefit in evaluating the capability of MiWi PRO protocol stack with this existing interface. Majority of the testing is done on Coordinators, which have routing capability. Project "MiWi PRO Test Interface" is configured to use only MiWi PRO protocol with Coordinator capability. For testing, user needs to change more
<u>8 bit Wireless</u> Development Kit	A set of demos has been developed for 8-bit Wireless Development Kit (8WDK). Please

Demos r	efer to "8-bit Wireless Development Kit
ι	Jser Guide" (DS70654A) for details.

<u>Demos</u> > <u>Running Demos</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

	MiWi(TM) Development Environment Help	Contents Index Home	Previous Up Next
В	Basic Demos		
Т	opics		
	Name	Description	
	Simple Example		

<u>Demos</u> > <u>Running Demos</u> > <u>Basic Demos</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright C 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Feature Demo

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

Following two diagrams show the MiApp flow chart of two nodes respectively.

Simple Node 1 Flow Chart



Simple Node 2 Flow Chart


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
- 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.

ZENA((TM) Packet	Sniff	er - Mi\	₩i(TM) P2P P	rotoc	ol											
Frame	Time(us)	Len	MA	C Fra	me Con	trol	TDIN	Seq	Dest	Dest	Source Addr	ess	Connection Requ	est Capa	bility Info		F	CS
00001	=5310752	20	CMD	N	N	N	Y	0x55	0x1234	0xFFFF	0x112233445	5667702	0x19	Y	N N	q Reon	-06	Ox6A
Frame	Time(us)	Len	MA	C Fra	me Con	trol	TDIN	Seq	Dest	Destina	ation Address	Sour	ce Address	Connecti	on Request F	lesponse	Capab	dity Inf
00002	=5317360	26	CMD	N	N	Y	Y	0x74	0x1234	0x1122	334455667702	0x1122	334455667701		0x00		Y	N
Frame	Time(us)	Len	MA	C Fra	me Con	trol		Seq	FC	s	1							
00003	+1584 =5318944	5	Туре АСК	Sec N	Pend N	ACK N	IPAN N	Num 0x74	RSSI C -03 0	orr CRC x68 OK								
																		-

• 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.

🛄 Tera Term Web 3.1 - COM2 ¥T	
File Edit Setup Web Control Window Help	
Starting Node 2 of Simple Demo for MiWi(TM) P2P Stack RF Transceiver: MRF24J40	-
Input Configuration: Button 1: RD6 on Explorer 16 RB5 on PICDEM Z RB0 on PIC18 Explorer Button 2: RD7 on Explorer 16 RB4 on PICDEM Z	
RAS on PIC18 Explorer Output Configuration:	
ES232 port USB on PIC18 Explorer and Explorer 16 LED 1: D10 on Explorer 16 RA0 on PICDEM Z D8 on PIC18 Explorer LED 2: D9 on Explorer 16 D1 or DICDEM Z	
D7 on PICDEM Z D7 on PIC18 Explorer	
Power on the board until LED 1 lights up to indicate connecting with peer. Push Button 1 to broadcast message. Push Button 2 to unicast encrypted message. LED 2 will be toggled upon receiving messages.	
2	
Connection PeerLongAddress PeerInfo 00 1122334455667701 41	-

- 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.

ZENA	(TM) Packet	: Sniff	er - Mil	₩i(TM) P2P P	rotoc	ol														_	
		_								1					_	_						
Frame	Time(us)	Len	MA	C Fra	me Con	trol	TDIN	Seq	Dest	Dest	s	ource A	ddress	Paylo	ad poor	FCS	000					
00001	+2681328	8 19	lype DATA	N	N	ACK N	IPAN Y	Num 0x4F	PAN 0x1234	Addr 0xFFFF	0x11	122334	455667702	0x0D 0x0A	-06	l Cor Ox6	9 OK					
Frame	Time(us)	Len	MA	C Fra	me Con	trol		Seq	Dest	Dest	s	ource A	ddress	Paylo	nd							
	+1178288	3	Type	Sec	Pend	ACK	IPAN	Num	PAN	Addr				0xB2	0x20	0xB2	0x20	0xB2	0xB2	0xB2	0x20	0xE
00002	=3859616	37	DATA	N	N	N	Y	0x50	0x1234	0xFFFF	0x11	122334	1455667702	0x20	0xB2	0x20	0x20	0x20	0xB2	0xB2	0x20	0x0
Frame	Time(us)	Len	MA	C Fra	me Con	trol		Seq	Dest	Dest	S	ource A	ddress	Paylo	nd							
	+1074256	5	Type	Sec	Pend	ACK	IPAN	Num	PAN	Addr				0xB2	0x20	0xB2	0x20	0xB2	0x20	0x20	0x20	0xE
00003	=4933872	2 37	DATA	N	N	N	Y	0x51	0x1234	0xFFFF	0x11	122334	1455667702	0x20	0xB2	0x20	0x20	0xB2	0x20	0x20	0xB2	0x0
																						-
•																						Þ

 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 "HELLO". Press the button 1 continuously on one end will display the complete bit map of "HELLO".

4	Simple Example - HyperTerminal	
F	le Edit View Call Transfer Help	
L		
	RA0 on PICDEM Z D8 on PIC18 Explorer LED 2: D9 on Explorer 16 RA1 on PICDEM Z D7 on PIC18 Explorer RF Transceiver: MRF24J40 Demo Instruction: Power on the board until LED 1 lights up to indicate connecting with peer. Push Button 1 to broadcast message. Push Button 2 to unicast encrypted message. LED 2 will be toggled upon receiving messages.	
	Connection PeerLongAddress PeerInfo 00 1122334455667702 41 Receive Packet Broadcast with RSSI D3 from 1122334455667702: Receive Packet Broadcast with RSSI D1 from 1122334455667702: Receive Packet Broadcast with RSSI D0 from 1122334455667702: Receive Packet Broadcast with RSSI D3 from 1122334455667702:	-
Co	nnected 0:02:28 Auto detect 19200 8-N-1 SCROLL CAPS NUM Capture Print echo	1.

- 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.



• 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.

ZENA	(TM) Packet Sn	iffer -	MiWi(TM) P	2P Prot	ocol												- 🗆 🗵
Frame	Time(us)	Len	MA	C Fra	me Con	trol		Sea	Dest	Destinat	ion Address	Source Address	Enervo	ted Dat	a			^
00022	+2891584 =119454496	34	Type DATA	Sec Y	Pend N	ACK Y	IPAN Y	Num 0x32	PAN 0x1234	4 0x11223	34455667701	0x1122334455667702	0x07 0xDA	0x00 0xB6	0x00 0xD5	0x00 0x1D	0x00 0xCC	0xBD
Frame 00023	Time(us) +2000 =119456496	Len 5	МА Туре АСК	AC Fra Sec N	me Con Pend N	trol ACK N	IPAN N	Seq Num Ox32	F(RSSI (+00 (CS Corr CRC Dx6A OK								
Frame 00024	Time(us) +1344080 =120800576	Len 45	MA Type DATA	AC Fra Sec Y	me Con Pend N	trol ACK Y	IPAN Y	Seq Num Ox33	Dest PAN 0x1234	Destinat 4 0x11223	ion Address 34455667701	Source Address 0x1122334455667702	Encryp 0x08 0xFE	oted Data 0x00 0xE4	a 0x00 0x8F	0x00 0x1F	0x00 0x7F	0x85 0x62
Frame 00025	Time(us) +2560 =120803136	Len 5	МА Туре АСК	AC Fra Sec N	me Con Pend N	trol ACK N	IPAN N	Seq Num Ox33	F(RSSI (+00 (CS Corr CRC Dx6A OK								
Frame 00026	Time(us) +1361168 =122164304	Len 45	МА Туре DATA	AC Fran Sec Y	me Con Pend N	trol ACK Y	IPAN Y	Seq Num 0x34	Dest PAN 0x1234	Destinat 4 0x11223	ion Address 34455667701	Source Address 0x1122334455667702	Encryp 0x09 0xB0	oted Data 0x00 0x8B	a 0x00 0x2D	0x00 0xC9	0x00 0x47	0xC6 0xF5
Frame 00027	Time(us) +2576 =122166880	Len 5	МА Туре АСК	AC Fra Sec N	me Con Pend N	trol ACK N	IPAN N	Seq Num Ox34	F(RSSI (+00 (C S Corr CRC Dx6B OK								
Frame 00028	Time(us) +1280208 =123447088	Len 45	MA Type DATA	AC Fra Sec Y	me Con Pend N	trol ACK Y	IPAN Y	Seq Num Ox35	Dest PAN 0x1234	Destinat 4 0x11223	ion Address 34455667701	Source Address 0x1122334455667702	Encryp OxOA Ox74	oted Data 0x00 0xF2	a 0x00 0xE6	0x00 0x12	0x00 0xC8	0xFC 0x46
•																		•

• 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.

MiWi(TM) P2P Network	Monitor	
File View Operation Tools	; Help	
<u></u>		
Eeal Time Display Channel 26 (0x1A) Speed 2 sec	Verboseness Level	
✓ Clear <u>M</u> essages on Start ✓ Clear <u>N</u> CD on Start	✓ Ignore Invalid Packets ✓ Auto Scroll	
Security Network Key OF OE O Security Level AES-CCM-	DD OC OB OA O9 O8 O7 O6 O5 O4 32	03 02 01 00 ecurity Parameters

 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 "P2P". Press the button 2 continuously on one end will display the complete bit map of "P2P".

🏀 Simple Example - HyperTerminal	
File Edit View Call Transfer Help	
Demo Instruction: Power on the board until LED 1 lights up to indicate connecting with peer. Push Button 1 to broadcast message. Push Button 2 to unicast encrypted message. LED 2 will be toggled upon receiving messages.	
Connection 00PeerLongAddress 1122334455667702PeerInfo 41Receive Packet Receive PacketBroadcast with RSSI D1 from 1122334455667702: Broadcast with RSSI D4 from 1122334455667702: Receive Packet Broadcast with RSSI D3 from 1122334455667702: Receive Packet Broadcast with RSSI D3 from 1122334455667702: Receive Packet Broadcast with RSSI D5 from 1122334455667702: Receive Packet Broadcast with RSSI D5 from 1122334455667702: Receive Packet Broadcast with RSSI D5 from 1122334455667702: Receive Packet Secured Unicast with RSSI D5 from 1122334455667702: Receive Packet Secured Unicast with RSSI D3 from 1122334455667702: Receive Packet Secured Unicast with RSSI D3 from 1122334455667702: Receive Packet Secured Unicast with RSSI D3 from 1122334455667702: 	
Connected 0:18:03 Auto detect 19200 8-N-1 SCROLL CAPS NUM Capture Print echo	11.

<u>Demos</u> > <u>Running Demos</u> > <u>Basic Demos</u> > <u>Simple Example</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

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

Following two diagrams show the MiApp flow chart of two nodes respectively.

Feature Demo Node 1 Flow Chart



Feature Demo Node 2 Flow Chart



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.

🛄 Tera Term Web 3.1 - COM2 VT	
File Edit Setup Web Control Window Help	
Starting Node 1 of Feature Demo for MiWi(TM) P2P Stack Input Configuration: Button 1: RD6 on Explorer 16 RB5 on PICDEM Z RB0 on PIC18 Explorer Button 2: RD7 on Explorer 16 RB4 on PICDEM Z RA5 on PIC18 Explorer Output Configuration: RS232 port USB on PIC18 Explorer and Explorer 16 IED 1: D10 on Explorer 16 RA0 on PICDEM Z D8 on PIC18 Explorer IED 2: D9 on Explorer 16 RA1 on PICDEM Z	
D7 on PIC18 Explorer	
Power on the board until LED 1 lights up to indicate it is ready to establish new connections. Push Button 1 to perform frequency agility procedure. Push Button 2 to unicast encrypted message. LED 2 will be toggled upon receiving messages. Starting Active Scan Scan Channel 11 Scan Channel 12 Scan Channel 13 Scan Channel 15 Scan Channel 16 Scan Channel 17 Scan Channel 18 Scan Channel 19 Scan Channel 20 Scan Channel 21 Scan Channel 22	
	-

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.

ZENA(TM) Packet Sniffer - MiWi(TM) P2P Protocol	_ 🗆 🗙
	-
Frame Timetus) Len MAC Frame Control Seq Dest Dest Source Source Address Active Scan FCS +11387664 Type Sec Pend AC(I TPAN Ibum PAH Addr PAH	
00001 =11387664 21 CMD N N N N 0xC3 0xFFFF 0xFFFF 0x1234 0x1122334455667702 0x19 +06 0x6A OK	
Frame Time(us) Len MAC Frame Control Seq Dest Destination Address Source Address Active Scan Response FCS	
1970 19pe Sec Fend Ack Fax Hum PAn 00002 =11394448 25 CMD N Y V 0xEC 0x122334455667702 0x1122334455667701 0x02 +00 0x6A OK	
Frame Time(us) Len MAC Frame Control Seq FCS	
+1520 Type Sec Fend ACK IFAN Num RSSI Corr CRC 00003 =11395968 ACK N N N 0xBC +06 0x6A 0K	
Frame Time(us) Len MAC Frame Control Seq Dest Source Address Connection Request Capability Info FCS	
+5015008 Type Sec Fend ACK IFAN Num PAN Addr Channel Sec Synch Reg RX0n RSSI Corr CRC 00004 =16410976 20 CMD N N Y 0xC5 0x1223 0x1122334455667702 0x19 Y N N +05 0x6B 0K	
Frame Time(us) Len MAC Frame Control Seq Dest Destination Address Source Address Connection Request Response Capability Info	
+5055 Type Sec Fend ACK IFAN Num PAN PAN Status Status Sec Synch Reg N 00005 =16416032 26 CMD N N Y 0xED 0x1223 0x1122334455667701 0x201 Y N N	Y 4
Frame Time(us) Len MAC Frame Control Seq FCS	
+1568 Type Sec Pend ACK IPAN Num RSSI Corr CRC 00006 =16417600 5 ACK N N N 0xED +05 0x69 0K	
Frame Time(us) Len MAC Frame Control Seq Dest Destination Address Source Address FCS	
+8068688 Type See Pend ACK IPAN PAN Data Request RSSI Corr CRC 00007 =24486288 24 CMD N N Y V 0xC6 0x1122334455667701 0x1122334455667702 +06 0x6B OK	
Frame Time(us) Len MAC Frame Control Seq FCS	
+1472 Type Sec Pend ACK IPAN Num RSSI Corr CRC 00008 =24487760 5 ACK N N N 0xC6 +02 0x6B 0K	
Frame Time(us) Len MAC Frame Control Seq Dest Destination Address Source Address FCS	
+5312 Type Sec Fend ACK IPAN N PAN RSSI Corr CRC 00009 =24493072 23 DATA N N Y Y 0xBE 0x12234 0x1122334455667701 +02 0x6B OK	
Frame Time(us) Len MAC Frame Control Seq FCS	
+1424 Type Sec Fend ACK IFAN Num RSSI Corr CRC	-

 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.

🛄 Tera Term Web 3.1 - COM2 ¥T	
File Edit Setup Web Control Window Help	
Button 1: RD6 on Explorer 16 RB5 on PICDEM Z RB0 on PIC18 Explorer Button 2: RD7 on Explorer 16 RB4 on PICDEM Z RA5 on PIC18 Explorer	
Output Configuration:	
USB on PIC18 Explorer and Explorer 16 LED 1: D10 on Explorer 16 RAO on PICDEM Z D8 on PIC18 Explorer	
LED 2: D9 on Explorer 16 RA1 on PICDEM Z D7 on PIC18 Explorer	
RF Transceiver: MRF24J40 Demo Instruction:	
Power on the board until LED 1 lights up to indicate it is connected to the peer. Push Button 1 to broadcast a message. Push Button 2 to unicast encrypted message on PICDEM Z or Explorer 16 demo boards. LED 2 will be toggled upon receiving messages.	
Starting Active Scan Scan Channel 11 Scan Channel 12 Scan Channel 13 Scan Channel 14 Scan Channel 15 Scan Channel 16 Scan Channel 17 Scan Channel 18 Scan Channel 19 Scan Channel 20 Scan Channel 20 Scan Channel 21 Scan Channel 22 Scan Channel 23 Scan Channel 23 Scan Channel 25 Scan Channel 26 Active Scan Results: Channel: 25 RSSI: FF	
Connection PeerLongAddress PeerInfo 00 1122334455667701 41	•

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 <u>MRF24J40</u> 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.

ZENA(TM) Packet Sniffer - MiWi(TM) P2P Protocol	- D ×
	-
Frame Time(us) Len MAC Frame Control Seq Dest Destination Address Source Address FCS	
+/513761 Type Sec Fend ACK IFAN Num PAN 00001 = 781376124 (MD N N Y V 0x100x1234)0x1122334455667701 0x1122334455667702 +02 0x6A 0K	
Frame Time(up) Len MAC Frame Control Sea FCS	
+1472 Type Sec Pend ACK IPAN Hum RSSI Corr CRC	
00002 =782848 5 ACK N Y N N 0x10 +02 0x6B OK	
Frame Time(us) Len MAC Frame Control Seq Dest Destination Address Source Address FCS	
+4/32 Type Sec Fend ACK IPAN Num PAN 00003 = 768680123 DATA N N Y V 0x8080x12340x1122334455667702 0x1122334455667701 +0.2 0x6E 0K	
Tennes Tenerge MAC Forme Control Con	
+1408 Type Sec Pend ACK IPAN Num RSSI Corr CRC	
00004 =788288 5 ACK N N N N 0x08 +03 0x6C OK	
Frame Time(us) Len MAC Frame Control Seq Dest Destination Address Source Address FCS	
+8003408 Type Sec Pend ACK IFAN Hum PAI 00005 #871566124 CMD N N Y V 0x110x12340x1122334455667701 0x1122334455667702 +04 0x64 0K	
+1472 Type See Pend ACK IPAN Num RSI Corr CRC	
00006 =8793168 5 ACK N Y N N 0x11 +00 0x6B OK	
Frame Time(us) Len MAC Frame Control Seq Dest Destination Address Source Address Encrypted Data FCS	
+16912 Type Sec Pend ACK IPAN Num PAI 00007_9810080134 DATA V N V V 0000001234 0x1122334455667202 0x1122334455667201 0x01 0x50 0x50 0x50 0x50 0x53 4 RSSI Co:	rr CF
	UN OF
+1394 Type See Pend ACK IPAN Num RSSI Corr CRC	
00008 =8812064 5 ACK N N N N N N 0x09+03 0x6B OK	-
	Þ

• 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.

Feature Demo - HyperTerminal	
File Edit View Call Transfer Help	
	1
Scan Channel 17Scan Channel 18Scan Channel 19Scan Channel 20Scan Channel 21Scan Channel 22Scan Channel 23Scan Channel 24Scan Channel 25Scan Channel 26Active Scan Results:Channel: 26RSSI: FF	
Connection Created	
Connection PeerLongAddress PeerInfo 00 1122334455667701 41 Receive Packet Secured Unicast with RSSI FF from 1122334455667701: Receive Packet Secured VIII FF from 1122334455667701	
Connected 0:11:14 Auto detect 19200 8-N-1 SCROLL CAPS NUM Capture Print echo	11.

- 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.



• If <u>MRF24J40</u> 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.

ZENA(TM) Packet Sniffer - MiWi(TM) P2P Protocol	
Frame Time(us) Len MAC Frame Control Seq Destination Address Source Address Data Request FCS 47451440 Type Sec Fend ACK IPAN PAH Destination Address Source Address Data Request RSSI Corr CRC 00057 =112316160 24 CMD N N Y Y 0x122334455667703 0x1122334455667702 Destination Address PAH PAH <td></td>	
Frame Time(us) Len MAC Frame Control Seq Destination Address Source Address Bata Request FCS 00058 =112319040 24 CMD N N Y Y Outside Control 0x122334455667703 0x1122334455667703 0x1122334455667703 <td></td>	
Frame Time(us) Len MAC Frame Control Seq Destination Address Source Address Data Request FCS 00059 =112323520 24 CMD N Y Y Y Destination Address Source Address Data Request RSSI Corr CRC +4480 Image: Sec Fend ACK IPAN N Y Y V Destination Address Source Address Data Request RSSI Corr CRC +00059 =112323520 24 CMD N Y Y V <t< td=""><td></td></t<>	
Frame Time(us) Len MAC Frame Control Seq Destination Address Source Address Data Request FCS 00000 =112326080 24 CMD N N Y Y Ox5E Destination Address Source Address Data Request RSSI Corr CRC 00000 =112326080 24 CMD N Y Y V Data Request Data Request RSSI Corr CRC 00000 =112326080 24 CMD N Y Y V	
Frame Time(us) Len MAC Frame Control Seq Dest Dest Source Address Channel Hopping FCS 41191104 Type Sec Fend ACK IPAN Addr Outlet Current Nev RSSI Corr CRC 113517184 20 ChD N N N Y Y Seq Dest Source Address Channel Hopping Current Nev RSSI Corr CRC	
Frame Time(us) Len MAC Frame Control Seq Dest Dest Source Address Chamel Hopping FCS 4251344 Type Sec Fend ACK IPAN Num PAN Addr Outlot Source Address Chamel Hopping FCS 00002 =113768528 20 ChD N N N Y Outlot Source Address Chamel Hopping Current Nev RSSI Corr CRC	
Frame Time(us) Len MAC Frame Control Seq Dest Dest Source Address Channel Hopping FCS 4249712 Type Sec Fend ACK IPAN N Y Out23 Out234 OxFFFF 0x1122334455667703 Ox80 Ox00 Hopping FCS 00003 =114018240 20 ChD N N Y Ox21 Ox12234 OxFFFF 0x1122334455667703 Ox80 Ox00 HO2 Ox6B OK	
Frame Time(us) Len MAC Frame Control Seq Destination Address Source Address FCS Image: Ima	•

• 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.

ZENA(TM) Packet Sniffer - MiWi(TM) P2P Protocol			
Frame Time(us) Len MAC Frame Control Seq Destination Address Source Address Active Scan FCS 00003 =12785024 25 CMD N Y Y 0x348 0x122334455667703 0x1122334455667702 0x1122334455667702 0x10 0x10 0x69 0K			
Frame +1520 Len +1520 MAC Frame Control Type Sec Pend ACK IPAN ACK N N N N N Seq N N FCS 00004 =12786544 5 ACK N N N N N N 0xA8 +04 0xA8 +04 0xA8 +04			
Frame Time(us) Len MAC Frame Control Seq Destination Address Source Address Active Scan Response FCS 00005 =12792864 25 CMD N V V 0x122334455667702 0x1122334455667703 0x02 +04 0x6B K			
Frame +1520 Len +1520 MAC Frame Control Type Sec Pend ACK IPAN ACK N N N N Seq Hum N FCS RSSI Corr 0x45 00006 =12794384 5 ACK N N N N N 0x45 +06 0x6A OK			
Frame Time(us) Len MAC Frame Control Seq Destination Address Source Address Data Request FCS 00007 =20257296 24 CHD N V V 0x122334455667703 0x1122334455667702 0x1122334455667702 Data Request FCS			
Frame +1472 Time(us) +1472 Len Type MAC Frame Control Type Seq Sec FCS 00008 =20258768 5 ACK N Y N N 0xA9 +04 0x6E OK			
Frame Time/us) I en MAC Frame Control Sea Dest Destination Address Source Address FCS			

• 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.

🏀 P2P - HyperTerminal	
File Edit View Call Transfer Help	
Energy Scan Results: Channel 11:	
Connected 0:00:44 Auto detect 19200 8-N-1 SCROLL CAPS NUM Capture Print echo	11.

• 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 21 Checking Channel 22 Checking Channel 23 Checking Channel 24	
Checking Channel 25	
Resynchronized Connection to Channel 26	
ICONNECTED U:24:UZ AUTO detect 19200 8-N-1 SCRULL CAPS NUM Capture Print echo	11.

<u>Demos</u> > <u>Running Demos</u> > <u>Basic Demos</u> > <u>Feature Demo</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiWi PRO Test Interface

MiWi PRO Test Interface

A new project, "MiWi PRO Test Interface" has been added to this release of MiWi DE. This new project, driven by testing menus on hyper terminal, is the main interface for Microchip developer to verify the functionalities of MiWi PRO protocol stack. Users, on the other hand, may benefit in evaluating the capability of MiWi PRO protocol stack with this existing interface.

Majority of the testing is done on Coordinators, which have routing capability. Project "MiWi PRO Test Interface" is configured to use only MiWi PRO protocol with Coordinator capability. For testing, user needs to change the EUI address defined file "ConfigApp.h" for each nodes. When powering up, the node will first try to find a possible Coordinator to join. If no Coordinator is found, it will start a new network and act as a PAN Cooridnator; otherwise, it will join the network. As described in MiWi PRO application note, it is highly recommended to introduce an interval longer than 30 seconds between each Coordinator joining the network.

By pressing "Enter" key on the PC keyboard, a menu system can be brought up on the hyper terminal that is connected to the RS232 port on the standard demo boards. The hyper terminal settings are the same as other MiWi DE demos: 19200 baud rate, 8 bit data, none parity, 1 bit stop and no flow control. The root menu looks like following: _____

- | 1: Enable/Disable Join
- | 2: Show Family Tree
- | 3: Show Routing Table
- 4: Send Message
- | 5: Set Family Tree
- | 6: Set Routing Table
- | 7: Set Neighbor Routing Table
- | 8: Start Frequency Agility
- | 9: Socket
- | z: Dump Connection

With option 1, user can enable/disable a node to accept new connections. When multiple Coordinators allow join, they all will respond to an active scan. The joining device, usually tries to join the first one to respond. As the result, the new device may join the network at random points and user has no control over network topology. By enable/disable join, user can effectively configure only one Coordinator to enable join, thus control where the next Coordinator to join the network, ultimately control the

network topology.

Option 2 and 3 allow the tester to see the family tree information as well as the routing table, including neighbor routing table information. Those information are essential for routing. User can check those two options and verify if the routing is as planned by the routing mechanism according to available routing information.

Option 4 may be used by the user to send a message by different ways. This option triggers data transmission to verify routing capabilities.

Option 5, 6 and 7 may be used by the tester to modify the family tree and routing table. By manually setting the family tree as well as routing information, user can stage the node in a particular network condition and verify its routing capabilities under such condition. These interfaces are heavily used by the developers. Tester needs to be very sure of the meaning of the his/her modifications, otherwise, the node may not be able to send any message due to invalid routing information.

Option 8 may be used by the tester to initiate frequency agility functionalities. Only the PAN Coordinator can use this option. All other Coordinators will discard this option.

Option 9 may be used to establish a socket connection (or indirect connection in MiApp's term). When two nodes on the

network choose option 9 at the same time (within 3 seconds by default setting), a socket should be established and node information should be exchanged and inserted into connection table respectively. The peer information can be verified by option Z.

Option Z may be used to dump connection table contents.

Demos > Running Demos > MiWi PRO Test Interface

Contents | Index | Home

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

8 bit Wireless Development Kit Demos

A set of demos has been developed for 8-bit Wireless Development Kit (8WDK). Please refer to "8-bit Wireless Development Kit User Guide" (DS70654A) for details.

<u>Demos</u> > <u>Running Demos</u> > <u>8 bit Wireless Development Kit Demos</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

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.

Topics

Name	Description
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
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.
<u>RF Transceivers</u>	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.

Configuring the Library

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

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

Macros

	Name	Description
~	ADDITIONAL_NODE_ID_SIZE	ADDITIONAL_NODE_ID_SIZ defines the size of additional payload will be attached to the P2P Connection Request. Additional payload is the information that the devices w to share with their peers on th P2P connection. The addition payload will be defined by the application and defined in mai
~	CONNECTION_SIZE	P2P_CONNECTION_SIZE defines the maximum P2P

		connections that this device allowes at the same time.
Ŷ	ENABLE_ACTIVE_SCAN	ENABLE_ACTIVE_SCAN will enable the device to do an ac scan to to detect current exist connection.
ę	ENABLE_BROADCAST	ENABLE_BROADCAST will enable the device to broadcas messages for the sleeping devices until they wake up an ask for the messages
Ŷ	ENABLE_ED_SCAN	ENABLE_ED_SCAN will enat the device to do an energy detection scan to find out the channel with least noise and operate on that channel
Ŷ	ENABLE_FREQUENCY_AGILITY	ENABLE_FREQUENCY_AGI will enable the device to chan operating channel to bypass t sudden change of noise
\$	ENABLE_HAND_SHAKE	ENABLE_HAND_SHAKE ena the protocol stack to hand-sha before communicating with ea other. Without a handshake process, RF transceivers can broadcast, or hardcoded the destination address to perforn unicast.
ş	ENABLE_PA_LNA	ENABLE_PA_LNA enable the external power amplifier and I noise amplifier on the RF boa achieve longer radio

		communication range. To ena PA/LNA on RF board without power amplifier and low noise amplifier may be harmful to th transceiver.
ę	ENABLE_INDIRECT_MESSAGE	ENABLE_INDIRECT_MESSA will enable the device to store packets for the sleeping devic temporily until they wake up a ask for the messages
\$	ENABLE_NETWORK_FREEZER	ENABLE_NETWORK_FREE2 enables the network freezer feature, which stores critical network information into non- volatile memory, so that the protocol stack can recover fro power loss gracefully. Networl freezer feature needs definitio NVM kind to be used, which is specified in HardwareProfile.h
ę	ENABLE_SECURITY	ENABLE_SECURITY will ena the device to encrypt and dec information transferred
Ŷ	ENABLE_SLEEP	ENABLE_SLEEP will enable t device to go to sleep and wak from the sleep
Ŷ	<u>EUI_0</u>	EUI_0 to <u>EUI_7</u> defines the Extended Universal Identifier, permanent address, for the wireless node. The length of t EUI is defined as <u>MY_ADDRESS_LENGTH</u> .
Ş	HARDWARE_SPI	HARDWARE_SPI enables the hardware SPI implementation MCU silicon. If HARDWARE_ is not defined, digital I/O pins be used to bit-bang the RF transceiver
---	----------------------	--
Ŷ	<u>MRF24J40</u>	Definition of MRF24J40 enabl the application to use Microch MRF24J40 2.4GHz IEEE 802.15.4 compliant RF transceiver. Only one RF transceiver can be defined.
Ŷ	<u>MRF49XA</u>	Definition of MRF49XA enable the application to use Microch MRF49XA subGHz proprietar transceiver. Only one RF transceiver can be defined.
Ŷ	MRF89XA	Definition of MRF89XA enable the application to use Microch MRF89XA subGHz proprietar transceiver
ş	MY_ADDRESS_LENGTH	MY_ADDRESS_LENGTH def the size of wireless node permanent address in byte. TI definition is not valid for IEEE 802.15.4 compliant RF transceivers.
Ŷ	<u>MY_PAN_ID</u>	MY_PAN_ID defines the PAN identifier. Use 0xFFFF if prefe random PAN ID.
Ŷ	NWK_ROLE_COORDINATOR	NWK_ROLE_COORDINATOF not valid if <u>PROTOCOL_P2P</u>

		defined. It specified that the net has the capability to be coordinator or PAN coordinato This definition cannot be defin with <u>NWK_ROLE_END_DEVICE</u> .
~	NWK_ROLE_END_DEVICE	NWK_ROLE_END_DEVICE is not valid if <u>PROTOCOL_P2P</u> defined. It specified that the n has the capability to be an end device. This definition cannot defined with <u>NWK_ROLE_COORDINATOF</u>
~~	PROTOCOL_MIWI	PROTOCOL_MIWI enables th application to use MiWi mesh networking stack. This definition cannot be defined with PROTOCOL_P2P.
~	PROTOCOL_MIWI_PRO	PROTOCOL_MIWI_PRO ena the application to use MiWi Pf stack. This definition cannot b defined with <u>PROTOCOL_P2I</u> <u>PROTOCOL_MIWI</u> .
~	PROTOCOL_P2P	Definition of Protocol Stack. ONLY ONE PROTOCOL STA CAN BE CHOSEN
~	RFD_WAKEUP_INTERVAL	RFD_WAKEUP_INTERVAL defines the wake up interval for RFDs in second. This definitic for the FFD devices to calcula various timeout. RFD depends the setting of the watchdog tin to wake up, thus this definition

		not used.
\$	RX_BUFFER_SIZE	RX_BUFFER_SIZE defines th maximum size of application payload which is to be receive
Ŷ	TARGET_SMALL	TARGET_SMALL will remove support of inter PAN communication and other min features to save programming space
Ŷ	TX_BUFFER_SIZE	TX_BUFFER_SIZE defines th maximum size of application payload which is to be transmitted

Configuring the Library > Application

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

ADDITIONAL_NODE_ID_SIZE Macro

С

#define ADDITIONAL_NODE_ID_SIZE 1

Description

ADDITIONAL_NODE_ID_SIZE defines the size of additional payload will be attached to the P2P Connection Request. Additional payload is the information that the devices what to share with their peers on the P2P connection. The additional payload will be defined by the application and defined in main.c

<u>Configuring the Library</u> > <u>Application</u> > <u>ADDITIONAL_NODE_ID_SIZE</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

CONNECTION_SIZE Macro

С

#define CONNECTION_SIZE 16

Description

P2P_CONNECTION_SIZE defines the maximum P2P connections that this device allowes at the same time.

<u>Configuring the Library > Application > CONNECTION_SIZE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

ENABLE_ACTIVE_SCAN Macro

С

#define ENABLE_ACTIVE_SCAN

Description

ENABLE_ACTIVE_SCAN will enable the device to do an active scan to to detect current existing connection.

Configuring the Library > Application > ENABLE_ACTIVE_SCAN Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ENABLE_BROADCAST Macro

С

#define ENABLE_BROADCAST

Description

ENABLE_BROADCAST will enable the device to broadcast messages for the sleeping devices until they wake up and ask for the messages

<u>Configuring the Library > Application > ENABLE_BROADCAST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ENABLE_ED_SCAN Macro

С

#define ENABLE_ED_SCAN

Description

ENABLE_ED_SCAN will enable the device to do an energy detection scan to find out the channel with least noise and operate on that channel

<u>Configuring the Library > Application > ENABLE_ED_SCAN Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ENABLE_FREQUENCY_AGILITY Macro

С

#define ENABLE_FREQUENCY_AGILITY

Description

ENABLE_FREQUENCY_AGILITY will enable the device to change operating channel to bypass the sudden change of noise

<u>Configuring the Library > Application > ENABLE_FREQUENCY_AGILITY</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ENABLE_HAND_SHAKE Macro

С

#define ENABLE_HAND_SHAKE

Description

ENABLE_HAND_SHAKE enables the protocol stack to handshake before communicating with each other. Without a handshake process, RF transceivers can only broadcast, or hardcoded the destination address to perform unicast.

<u>Configuring the Library > Application > ENABLE_HAND_SHAKE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ENABLE_PA_LNA Macro

С

#define ENABLE_PA_LNA

Description

ENABLE_PA_LNA enable the external power amplifier and low noise amplifier on the RF board to achieve longer radio communication range. To enable PA/LNA on RF board without power amplifier and low noise amplifier may be harmful to the transceiver.

<u>Configuring the Library > Application > ENABLE_PA_LNA Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

ENABLE_INDIRECT_MESSAGE Macro

С

#define ENABLE_INDIRECT_MESSAGE

Description

ENABLE_INDIRECT_MESSAGE will enable the device to store the packets for the sleeping devices temporily until they wake up and ask for the messages

<u>Configuring the Library > Application > ENABLE_INDIRECT_MESSAGE</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ENABLE_NETWORK_FREEZER Macro

С

#define ENABLE_NETWORK_FREEZER

Description

ENABLE_NETWORK_FREEZER enables the network freezer feature, which stores critical network information into non-volatile memory, so that the protocol stack can recover from power loss gracefully. Network freezer feature needs definition of NVM kind to be used, which is specified in HardwareProfile.h

<u>Configuring the Library > Application > ENABLE_NETWORK_FREEZER</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ENABLE_SECURITY Macro

С

#define ENABLE_SECURITY

Description

ENABLE_SECURITY will enable the device to encrypt and decrypt information transferred

<u>Configuring the Library > Application > ENABLE_SECURITY Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ENABLE_SLEEP Macro

С

#define ENABLE_SLEEP

Description

ENABLE_SLEEP will enable the device to go to sleep and wake up from the sleep

<u>Configuring the Library > Application > ENABLE_SLEEP Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 EUI_0 Macro
 C
 #define EUI_0 0x01

Description

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

Configuring the Library > Application > EUI_0 Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

HARDWARE_SPI Macro

С

#define HARDWARE_SPI

Description

HARDWARE_SPI enables the hardware SPI implementation on MCU silicon. If HARDWARE_SPI is not defined, digital I/O pins will be used to bit-bang the RF transceiver

<u>Configuring the Library > Application > HARDWARE_SPI Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 MRF24J40 Macro
 C

 // C
 #define MRF24J40

Description

Definition of MRF24J40 enables the application to use Microchip MRF24J40 2.4GHz IEEE 802.15.4 compliant RF transceiver. Only one RF transceiver can be defined.

Configuring the Library > Application > MRF24J40 Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Description

Definition of MRF49XA enables the application to use Microchip MRF49XA subGHz proprietary RF transceiver. Only one RF transceiver can be defined.

<u>Configuring the Library > Application > MRF49XA Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help	Contents Index Home	Previous Up Next
MRF89XA Macro		
С		
#define MRF89XA		

Description

Definition of MRF89XA enables the application to use Microchip MRF89XA subGHz proprietary RF transceiver

Configuring the Library > Application > MRF89XA Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MY_ADDRESS_LENGTH Macro

С

#define MY_ADDRESS_LENGTH 8

Description

MY_ADDRESS_LENGTH defines the size of wireless node permanent address in byte. This definition is not valid for IEEE 802.15.4 compliant RF transceivers.

<u>Configuring the Library > Application > MY_ADDRESS_LENGTH Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MY_PAN_ID Macro

С

#define MY_PAN_ID 0x1234

Description

MY_PAN_ID defines the PAN identifier. Use 0xFFFF if prefer a random PAN ID.

<u>Configuring the Library > Application > MY_PAN_ID Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

NWK_ROLE_COORDINATOR Macro

С

#define NWK_ROLE_COORDINATOR

Description

NWK_ROLE_COORDINATOR is not valid if <u>PROTOCOL_P2P</u> is defined. It specified that the node has the capability to be coordinator or PAN coordinator. This definition cannot be defined with <u>NWK_ROLE_END_DEVICE</u>.

<u>Configuring the Library > Application > NWK_ROLE_COORDINATOR</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

NWK_ROLE_END_DEVICE Macro

С

#define NWK_ROLE_END_DEVICE

Description

NWK_ROLE_END_DEVICE is not valid if <u>PROTOCOL_P2P</u> is defined. It specified that the node has the capability to be an end device. This definition cannot be defined with <u>NWK_ROLE_COORDINATOR</u>.

<u>Configuring the Library</u> > <u>Application</u> > <u>NWK_ROLE_END_DEVICE</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

PROTOCOL_MIWI Macro

С

#define PROTOCOL_MIWI

Description

PROTOCOL_MIWI enables the application to use MiWi mesh networking stack. This definition cannot be defined with PROTOCOL_P2P.

Configuring the Library > Application > PROTOCOL_MIWI Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

PROTOCOL_MIWI_PRO Macro

С

#define PROTOCOL_MIWI_PRO

Description

PROTOCOL_MIWI_PRO enables the application to use MiWi PRO stack. This definition cannot be defined with <u>PROTOCOL_P2P</u> or <u>PROTOCOL_MIWI</u>.

<u>Configuring the Library > Application > PROTOCOL_MIWI_PRO Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

PROTOCOL_P2P Macro

С

#define PROTOCOL_P2P

Description

Definition of Protocol Stack. ONLY ONE PROTOCOL STACK CAN BE CHOSEN

PROTOCOL_P2P enables the application to use MiWi P2P stack. This definition cannot be defined with <u>PROTOCOL_MIWI</u>.

<u>Configuring the Library > Application > PROTOCOL_P2P Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

RFD_WAKEUP_INTERVAL Macro

С

#define RFD_WAKEUP_INTERVAL 8

Description

RFD_WAKEUP_INTERVAL defines the wake up interval for RFDs in second. This definition is for the FFD devices to calculated various timeout. RFD depends on the setting of the watchdog timer to wake up, thus this definition is not used.

<u>Configuring the Library > Application > RFD_WAKEUP_INTERVAL</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

RX_BUFFER_SIZE Macro

С

#define RX_BUFFER_SIZE 40

Description

RX_BUFFER_SIZE defines the maximum size of application payload which is to be received

<u>Configuring the Library > Application > RX_BUFFER_SIZE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TARGET_SMALL Macro

С

#define TARGET_SMALL

Description

TARGET_SMALL will remove the support of inter PAN communication and other minor features to save programming space

<u>Configuring the Library > Application > TARGET_SMALL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TX_BUFFER_SIZE Macro

С

#define TX_BUFFER_SIZE 40

Description

TX_BUFFER_SIZE defines the maximum size of application payload which is to be transmitted

<u>Configuring the Library > Application > TX_BUFFER_SIZE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

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.

Macros

	Name	Description
¢.	ACTIVE_SCAN_RESULT_SIZE	ACTIVE_SCAN_RESI the maximum active s stack can hold. If activ received exceed the d ACTIVE_SCAN_RESI later active scan respo discarded
¢.	CONNECTION_RETRY_TIMES	CONNECTION_RETF maximum time that the try to establish a conn retry times are exhaus return to application la to do next
Ŷ	COUNTER_CRYSTAL_FREQ	COUNTER_CRYSTAL frequency of the crysta to the MCU counter to functionality when MC
Ş	ENABLE_DUMP	ENABLE_DUMP will e be able to print out the connection entry. It is debugging process
÷	ENABLE_ENHANCED_DATA_REQUEST	ENABLE_ENHANCE

		enables the Enhanced feature of P2P stack. I message that is send device with Data Requ wakeup, to save 20% for sleeping device, th battery life.
~~	ENABLE_TIME_SYNC	ENABLE_TIME_SYNG Synchronizaiton feature allows the FFD to coo- interval of sleeping de FFD to connect to man Once Time Synchroni: enabled, following par required to be defined <u>TIME_SYNC_SLOTS</u> <u>COUNTER_CRYSTAL</u>
~~	FA_BROADCAST_TIME	FA_BROADCAST_TIN number of times to bro hopping message to the before the Frequency to the new channel
~~	INDIRECT_MESSAGE_SIZE	INDIRECT_MESSAGI maximum number of r device can store for th device(s)
~~	INDIRECT_MESSAGE_TIMEOUT	INDIRECT_MESSAGI defines the timeout int the stored packets for
Ŷ	RESYNC_TIMES	RESYNC_TIMES defi number of times to try in all available channe the control to the appli

~~	RFD_DATA_WAIT	RFD_DATA_WAIT is t for sleeping device to from the associate dev Request. After this tim device can continue to go to sleep to conserv
~~0	TIME_SYNC_SLOTS	TIME_SYNC_SLOTS number of time slot av duty cycle. As a rule, t slot must be equal or I number of sleeping de connected to the FFD, sleeping device can be slot. The time slot peri following formula: Tim <u>RFD_WAKEUP_INTE</u> TIME_SYNC_SLOTS slot period depends of oscillator accuracy on the 32KHz crystal acc devices. The definitior TIME_SYNC_SLOTS <u>more</u>

Topics

Name	Description
MiWi(TM) P2P Communication Protocol	Following configurations can be used to fine tune the behavior of MiWi(TM) P2P wireless protocol.
MiWi and MiWi PRO Networking Protocols	Following configurations can be used to fine tune the behavior of MiWi(TM) mesh networking protocol.

Configuring the Library > Wireless Protocol

Microchip My Application xx.yy - [Jan 1, 2009] Copyright O 2009 Microchip Technology, Inc. All rights reserved.

ACTIVE_SCAN_RESULT_SIZE Macro

С

#define ACTIVE_SCAN_RESULT_SIZE 4

Description

ACTIVE_SCAN_RESULT_SIZE defines the maximum active scan result that the stack can hold. If active scan responses received exceed the definition of ACTIVE_SCAN_RESULT_SIZE, those later active scan responses will be discarded

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>ACTIVE_SCAN_RESULT_SIZE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.
CONNECTION_RETRY_TIMES Macro

С

#define CONNECTION_RETRY_TIMES 3

Description

CONNECTION_RETRY_TIMES is the maximum time that the wireless node can try to establish a connection. Once the retry times are exhausted control will be return to application layer to decide what to do next

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>CONNECTION_RETRY_TIMES Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

COUNTER_CRYSTAL_FREQ Macro

С

#define COUNTER_CRYSTAL_FREQ 32768

Description

COUNTER_CRYSTAL_FREQ defines the frequency of the crystal that is connected to the MCU counter to perform timing functionality when MCU is in sleep.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>COUNTER_CRYSTAL_FREQ Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

ENABLE_DUMP Macro

С

#define ENABLE_DUMP

Description

ENABLE_DUMP will enable the stack to be able to print out the content of the P2P connection entry. It is useful in the debugging process

<u>Configuring the Library > Wireless Protocol > ENABLE_DUMP Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ENABLE_ENHANCED_DATA_REQUEST Macro

С

#define ENABLE_ENHANCED_DATA_REQUEST

Description

ENABLE_ENHANCED_DATA_REQUEST enables the Enhanced Data Request feature of P2P stack. It combines the message that is send from the sleeping device with Data Request command upon wakeup, to save 20% - 30% active time for sleeping device, thus prolong the battery life.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>ENABLE_ENHANCED_DATA_REQUEST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ENABLE_TIME_SYNC Macro

С

#define ENABLE_TIME_SYNC

Description

ENABLE_TIME_SYNC enables the Time Synchronizaiton feature of P2P stack. It allows the FFD to coordinate the check-in interval of sleeping device, thus allow one FFD to connect to many sleeping device. Once Time Synchronization feature is enabled, following parameters are also required to be defined: <u>TIME_SYNC_SLOTS COUNTER_CRYSTAL_FREQ</u>

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>ENABLE_TIME_SYNC</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

FA_BROADCAST_TIME Macro

С

#define FA_BROADCAST_TIME 0x03

Description

FA_BROADCAST_TIME defines the total number of times to broadcast the channel hopping message to the rest of PAN, before the Frequency Agility initiator jump to the new channel

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>FA_BROADCAST_TIME</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

INDIRECT_MESSAGE_SIZE Macro

С

#define INDIRECT_MESSAGE_SIZE 2

Description

INDIRECT_MESSAGE_SIZE defines the maximum number of packets that the device can store for the sleeping device(s)

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>INDIRECT_MESSAGE_SIZE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

INDIRECT_MESSAGE_TIMEOUT Macro

С

#define INDIRECT_MESSAGE_TIMEOUT (ONE_SECOND * RFD W/

Description

INDIRECT_MESSAGE_TIMEOUT defines the timeout interval in seconds for the stored packets for sleeping devices

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>INDIRECT_MESSAGE_TIMEOUT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

RESYNC_TIMES Macro

С

#define RESYNC_TIMES 0x03

Description

RESYNC_TIMES defines the maximum number of times to try resynchronization in all available channels before hand over the control to the application layer

<u>Configuring the Library > Wireless Protocol > RESYNC_TIMES Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

RFD_DATA_WAIT Macro

С

#define RFD_DATA_WAIT 0x00003FFF

Description

RFD_DATA_WAIT is the timeout defined for sleeping device to receive a message from the associate device after Data Request. After this timeout, the RFD device can continue to operate and then go to sleep to conserve battery power.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>RFD_DATA_WAIT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

TIME_SYNC_SLOTS Macro

С

#define TIME_SYNC_SLOTS 10

Description

TIME_SYNC_SLOTS defines the total number of time slot available within one duty cycle. As a rule, the number of time slot must be equal or higher than the total number of sleeping devices that are connected to the FFD, so that each sleeping device can be assigned to a time slot. The time slot period is calcualted by following formula: Time Slot Period = <u>RFD_WAKEUP_INTERVAL</u> / TIME_SYNC_SLOTS The length of time slot period depends on the primary oscillator accuracy on the FFD as well as the 32KHz crystal accuracy on sleeping devices. The definition of TIME_SYNC_SLOTS is only valid if <u>ENABLE_TIME_SYNC</u> is defined

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>TIME_SYNC_SLOTS</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

MiWi(TM) P2P Communication Protocol

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

Macros

	Name	Description
~0	CONNECTION_INTERVAL	CONNECTION_INTERVAL defines the interval in second between two connection request.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi(TM) P2P</u> <u>Communication Protocol</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

CONNECTION_INTERVAL Macro

С

#define CONNECTION_INTERVAL 2

Description

CONNECTION_INTERVAL defines the interval in second between two connection request.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi(TM) P2P</u> <u>Communication Protocol</u> > <u>CONNECTION_INTERVAL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

MiWi and MiWi PRO Networking Protocols

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

Macros

	Name	Description
~	INDIRECT_MESSAGE_TIMEOUT_CYCLE	When broadcasting t enabled, it is hard for track which end devi broadcast message. tracking is provided, receive the same brc MiWi PRO solves thi the broadcast messa side. INDIRECT_MESSAC defines the total num receives before the t out. It is hard for a sl timing, so tracking th received is a simpler
~~0	MAX_ROUTING_FAILURE	MAX_ROUTING_FA of failures of routing before such route is decision of message definition of this para the available routes of definition is only valid
~	OPEN_SOCKET_POLL_INTERVAL	For a sleeping device an indirect connectio

		be desirable to poll the interval, which can bound the OPEN_SOCKET_TH is to poll the data at a OPEN_SOCKET_TH OPEN_SOCKET_PC polling interval in syn device to acquire dat process of establishiconnection. This para sleeping device.
Ŷ	OPEN_SOCKET_TIMEOUT	OPEN_SOCKET_TII period in symbols for attempt to establish a or in MiApp term, an

Topics

Name	Description
MiWi Mesh Networking Protocol	
MiWi PRO Networking Protocol	

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

INDIRECT_MESSAGE_TIMEOUT_CYCLE Macro

С

#define INDIRECT_MESSAGE_TIMEOUT_CYCLE 2

Description

When broadcasting to a sleeping device is enabled, it is hard for a parent node to track which end device has received the broadcast message. However, if no tracking is provided, the end device may receive the same broadcast multiple times. MiWi PRO solves this problem by tracking the broadcast message on sleeping device side. INDIRECT_MESSAGE_TIMEOUT_CYCLE defines the total number of messages receives before the broadcast record times out. It is hard for a sleeping node to track timing, so tracking the number of message received is a simpler way.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > INDIRECT_MESSAGE_TIMEOUT_CYCLE Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MAX_ROUTING_FAILURE Macro

С

#define MAX_ROUTING_FAILURE 3

Description

MAX_ROUTING_FAILURE is the number of failures of routing between coordinators before such route is disabled in the decision of message route. Proper definition of this parameter helps to update the available routes dynamically. This definition is only valid for a coordinator.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>MAX_ROUTING_FAILURE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

OPEN_SOCKET_POLL_INTERVAL Macro

С

#define OPEN_SOCKET_POLL_INTERVAL (ONE_SECOND)

Description

For a sleeping device, when establishing an indirect connection (socket), it may not be desirable to poll the data at the normal interval, which can be longer than <u>OPEN_SOCKET_TIMEOUT</u>, the solution is to poll the data at a fast rate, lower than <u>OPEN_SOCKET_TIMEOUT</u>.

OPEN_SOCKET_POLL_INTERVAL is the polling interval in symbols for a sleeping device to acquire data from its parent in the process of establishing indirect (socket) connection. This parameter is only valid for sleeping device.

<u>Configuring the Library > Wireless Protocol > MiWi and MiWi PRO</u> <u>Networking Protocols > OPEN_SOCKET_POLL_INTERVAL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

OPEN_SOCKET_TIMEOUT Macro

С

#define OPEN_SOCKET_TIMEOUT (ONE_SECOND * 3)

Description

OPEN_SOCKET_TIMEOUT is the timeout period in symbols for a node to abandon attempt to establish a socket connection, or in MiApp term, an indrect connection

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>OPEN_SOCKET_TIMEOUT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi Mesh Networking Protocol

Macros

	Name	Description
~~	BROADCAST_RECORD_SIZE	BROADCAST_RECORD_S the parameter that specifies maximum number of broad record available. Broadcast is used to track the broadca messages so that the wirele knows if the same broadcas been received before.
~	BROADCAST_RECORD_TIMEOUT	BROADCAST_RECORD_T defines the timeout in symb node to expire its broadcast The broadcast record is use track the received broadcas message and to prevent rec duplicate broadcast messag definition is only valid for a sleeping device.
~0	MIWI_ACK_TIMEOUT	MIWI_ACK_TIMEOUT is th timeout period in symbols for to receive a MiWi network la acknowledgement. This par is for MiWi network layer, no MAC layer. MAC layer acknowledgement timeout i handled in MiMAC layer.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO

Networking Protocols > MiWi Mesh Networking Protocol

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

BROADCAST_RECORD_SIZE Macro

С

#define BROADCAST_RECORD_SIZE 4

Description

BROADCAST_RECORD_SIZE is the parameter that specifies the maximum number of broadcast record available. Broadcast record is used to track the broadcast messages so that the wireless node knows if the same broadcast has been received before.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>MiWi Mesh Networking Protocol</u> > <u>BROADCAST_RECORD_SIZE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

BROADCAST_RECORD_TIMEOUT Macro

С

#define BROADCAST_RECORD_TIMEOUT (ONE_SECOND)

Description

BROADCAST_RECORD_TIMEOUT defines the timeout in symbols for a node to expire its broadcast record. The broadcast record is used to track the received broadcast message and to prevent receiving duplicate broadcast message. This definition is only valid for a non-sleeping device.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>MiWi Mesh Networking Protocol</u> > <u>BROADCAST_RECORD_TIMEOUT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

Previous | Up | Next

MIWI_ACK_TIMEOUT Macro

С

#define MIWI_ACK_TIMEOUT (ONE_SECOND)

Description

MIWI_ACK_TIMEOUT is the timeout period in symbols for a node to receive a MiWi network layer acknowledgement. This parameter is for MiWi network layer, not for MAC layer. MAC layer acknowledgement timeout is handled in MiMAC layer.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi Mesh Networking Protocol > MIWI_ACK_TIMEOUT Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi PRO Networking Protocol

Macros

	Name	Description
~~	<u>COMM_INTERVAL</u>	COMM_INTERV, interval in symbo communications. services that use FAMILY_TREE_I ROUTING_TABL CHANNLE_HOP
~~	COMM_RSSI_THRESHOLD	COMM_RSSI_TI minimum signal s a Coordinator to neighbor Coordir that is higher tha neighbor to be al setting usually is interpretation for This value is repu data width, thus s
~0	ENABLE_MIWI_PRO_ACKNOWLEDGEMENT	ENABLE_MIWI_ enables the MiW acknowledgemer data packet for a process will be h stack without app MiWi PRO acknc layer. It may be s should be differie message is sent

		PRO acknowledç and notified by ca <u>MiApp_CB_RFD</u> Otherwise, wheth acknowledgemen application layer
Ŷ	ENABLE_ROUTING_UPDATE	ENABLE_ROUT Coordinator capt out Routing Table table. The interva is defined as <u>RO</u>
Ş	ENABLE_BROADCAST_TO_SLEEP_DEVICE	ENABLE_BROA enables message device.
Ş.	FA_COMM_INTERVAL	FA_COMM_INTE interval between agility related me FA_BROADCAS granularity for the recommended to half second.
\$	FA_MAX_NOISE_THRESHOLD	FA_MAX_NOISE maximum noise I accepts for a cha Frequency Agility service when a n PAN Coordinator agility operation. to the signal stree specific RF trans represented by <u>B</u> thus should be le
~	FA_WAIT_TIMEOUT	FA_WAIT_TIME(

		symbols during th Those timeouts in Coordinators to s the PAN Coordin Agility Against Ch channel is sugge devices to jump t receiving Freque protocol service. protocol runs smo set this timeout h
Ş	FAMILY_TREE_BROADCAST	FAMILY_TREE_I number of broad protocol service a network. To ensu table, it is recomi higher than 1.
, Ç	MIWI_PRO_ACK_TIMEOUT	MIWI_PRO_ACK period in symbols PRO network lay parameter is for I for MAC layer. M timeout is handle
Ŷ	NUM_COORDINATOR	NUM_COORDIN number of Coord support. The pos and 64. NUM_CC closedly associat resources
ç	PACKET_RECORD_SIZE	PACKET_RECO that specifies the record available. the broadcast me node knows if the

		received before.
Ŷ	PACKET_RECORD_TIMEOUT	PACKET_RECO timeout in symbo packet record. Th track the received prevent receiving message. This do sleeping device.
Ŷ	RANDOM_DELAY_RANGE	RANDOM_DELA delay range in m a message, it is r Coordinator shou to avoid multiple collide the messa actual random de between 0 and R This setting is rea than 20. The valu with 8-bit data wi 256.
Ŷ	ROUTING_UPDATE_INTERVAL	ROUTING_UPD, interval in symbo capable device to Report. This define ENABLE_ROUT
¢.	ROUTING_UPDATE_EXPIRATION	ROUTING_UPD/ valid link expired <u>ROUTING_UPD/</u> receiving Routing parameter is defi invalid after (3 * <u>ROUTING_UPD/</u> no Routing Table other side of the

ROUTING_TABLE_BROADCAST

ROUTING_TABL number of broad Report. To ensurcoordinators, it is value higher thar

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>MiWi PRO Networking Protocol</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help

Contents | Index | Home

COMM_INTERVAL Macro

С

#define COMM_INTERVAL ONE_SECOND

Description

COMM_INTERVAL defines communication interval in symbols of protocol services communications. An incomplete list of protocol services that use COMM_INTERVAL are FAMILY_TREE_REPORT, ROUTING_TABLE_REPORT and CHANNLE_HOPPING_REQUEST.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > COMM_INTERVAL Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

COMM_RSSI_THRESHOLD Macro

С

#define COMM_RSSI_THRESHOLD 0x01

Description

COMM_RSSI_THRESHOLD defines the minimum signal strength that is acceptable for a Coordinator to route messages through. A neighbor Coordinator that has signal strength that is higher than this threshold can be list as neighbor to be able to route message. This setting usually is related to the signal strength interpretation for the specific RF transceiver. This value is represented by <u>BYTE</u> with 8-bit data width, thus should be less than 256.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > COMM_RSSI_THRESHOLD Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

ENABLE_MIWI_PRO_ACKNOWLEDGEMENT

Macro

С

#define ENABLE_MIWI_PRO_ACKNOWLEDGEMENT

Description

ENABLE_MIWI_PRO_ACKNOWLEDGEMENT enables the MiWi PRO stack to send back an acknowledgement packet when a MiWi PRO data packet for application is received. This process will be handled automatically in the stack without application layer involvement. MiWi PRO acknowledgement is ack in network layer. It may be sent across multiple hops and should be differiented with MAC ack. When message is sent to a sleeping device, the MiWi PRO acknowledgemnt will be received later and notified by call back function MiApp_CB_RFDAcknowledgement. Otherwise, whether receive MiWi PRO acknowledgement will be used to notify application layer if transmission is successful.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > ENABLE_MIWI_PRO_ACKNOWLEDGEMENT Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ENABLE_ROUTING_UPDATE Macro

С

#define ENABLE_ROUTING_UPDATE

Description

ENABLE_ROUTING_UPDATE enables the Coordinator capble device to periodically send out Routing Table Report to update the routing table. The interval of sending out routing table is defined as ROUTING_UPDATE_INTERVAL

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>MiWi PRO Networking Protocol</u> > <u>ENABLE_ROUTING_UPDATE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

ENABLE_BROADCAST_TO_SLEEP_DEVICE Macro

С

#define ENABLE_BROADCAST_TO_SLEEP_DEVICE

Description

ENABLE_BROADCAST_TO_SLEEP_DEVICE enables messages broadcast to a sleeping device.

<u>Configuring the Library > Wireless Protocol > MiWi and MiWi PRO</u> <u>Networking Protocols > MiWi PRO Networking Protocol ></u> <u>ENABLE_BROADCAST_TO_SLEEP_DEVICE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

FA_COMM_INTERVAL Macro

С

#define FA_COMM_INTERVAL (ONE_SECOND)

Description

FA_COMM_INTERVAL defines the time interval between broadcasting the frequency agility related message for FA_BROADCAST_TIME times. To leave granularity for the protocol runs smoothly, it is recommended to set this timeout higher than a half second.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > FA_COMM_INTERVAL Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

FA_MAX_NOISE_THRESHOLD Macro

С

#define FA_MAX_NOISE_THRESHOLD 0x80

Description

FA_MAX_NOISE_THRESHOLD defines the maximum noise level that a Coordinator accepts for a channel to avoid sending out Frequency Agility Against Channel protocol service when a new channel is proposed by PAN Coordinator to hop to during frequency agility operation. This setting usually is related to the signal strength interpretation for the specific RF transceiver. This value is represented by <u>BYTE</u> with 8-bit data width, thus should be less than 256.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > FA_MAX_NOISE_THRESHOLD Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.
FA_WAIT_TIMEOUT Macro

С

#define FA_WAIT_TIMEOUT ((ONE_SECOND) * 2)

Description

FA_WAIT_TIMEOUT defines the timeouts in symbols during the frequency agility process. Those timeouts include timeout for all Coordinators to start energy scan; timeout for the PAN Coordinator to receive Frequency Agility Against Channel protocol service after a channel is suggested and timeout for all devices to jump to the new channel after receiving Frequency Agility Change Channel protocol service. To leave granularity for the protocol runs smoothly, it is recommended to set this timeout higher than 1 second.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > FA_WAIT_TIMEOUT Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

FAMILY_TREE_BROADCAST Macro

С

#define FAMILY_TREE_BROADCAST 3

Description

FAMILY_TREE_BROADCAST defines the number of broadcasts for Family Tree Report protocol service after a Coordinator joins the network. To ensure delivery of the Family Tree table, it is recommended to set this value higher than 1.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > FAMILY_TREE_BROADCAST Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MIWI_PRO_ACK_TIMEOUT Macro

С

#define MIWI_PRO_ACK_TIMEOUT (ONE_SECOND)

Description

MIWI_PRO_ACK_TIMEOUT is the timeout period in symbols for a node to receive a MiWi PRO network layer acknowledgement. This parameter is for MiWi PRO network layer, not for MAC layer. MAC layer acknowledgement timeout is handled in MiMAC layer.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>MiWi PRO Networking Protocol</u> > <u>MIWI_PRO_ACK_TIMEOUT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

NUM_COORDINATOR Macro

С

#define NUM_COORDINATOR 16

Description

NUM_COORDINATOR defines the maximum number of Coordinators that the network can support. The possible numbers are 8, 16, 32 and 64. NUM_COORDINAOTR definition is closedly associated with RAM and NVM resources

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > NUM_COORDINATOR Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

PACKET_RECORD_SIZE Macro

С

#define PACKET_RECORD_SIZE 5

Description

PACKET_RECORD_SIZE is the parameter that specifies the maximum number of packet record available. Packet record is used to track the broadcast messages so that the wireless node knows if the same packet has been received before.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > PACKET_RECORD_SIZE Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

PACKET_RECORD_TIMEOUT Macro

С

#define PACKET_RECORD_TIMEOUT (ONE_SECOND/2)

Description

PACKET_RECORD_TIMEOUT defines the timeout in symbols for a node to expire its packet record. The packet record is used to track the received broadcast message and to prevent receiving duplicate broadcast message. This definition is only valid for a non-sleeping device.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>MiWi PRO Networking Protocol</u> > <u>PACKET_RECORD_TIMEOUT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

RANDOM_DELAY_RANGE Macro

С

#define RANDOM_DELAY_RANGE 200

Description

RANDOM_DELAY_RANGE defines random delay range in milliseconds. When rebroadcast a message, it is recommended that a Coordinator should introduce a random delay to avoid multiple Coordinators rebroadcast and collide the messages at the same time. The actual random delay will be randomly selected between 0 and RANDOM_DELAY_RANGE. This setting is recommended to be set higher than 20. The value is represented by BYTE with 8-bit data width, thus should be less than 256.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > RANDOM DELAY RANGE Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

ROUTING_UPDATE_INTERVAL Macro

С

#define ROUTING_UPDATE_INTERVAL (ONE_HOUR)

Description

ROUTING_UPDATE_INTERVAL defines the interval in symbols that the Coordinator capable device to send out Routing Table Report. This definition is only effective if <u>ENABLE_ROUTING_UPDATE</u> is defined.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>MiWi PRO Networking Protocol</u> > <u>ROUTING_UPDATE_INTERVAL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ROUTING_UPDATE_EXPIRATION Macro

С

#define ROUTING_UPDATE_EXPIRATION 3

Description

ROUTING_UPDATE_EXPIRATION defines the valid link expired after times of <u>ROUTING_UPDATE_INTERVAL</u> without receiving Routing Table Report. If this parameter is defined as 3, a link will becomes invalid after (3 * <u>ROUTING_UPDATE_INTERVAL</u>) symbols if no Routing Table Report is received from the other side of the link.

<u>Configuring the Library</u> > <u>Wireless Protocol</u> > <u>MiWi and MiWi PRO</u> <u>Networking Protocols</u> > <u>MiWi PRO Networking Protocol</u> > <u>ROUTING_UPDATE_EXPIRATION Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

ROUTING_TABLE_BROADCAST Macro

С

#define ROUTING_TABLE_BROADCAST 3

Description

ROUTING_TABLE_BROADCAST defines the number of broadcasts for Routing Table Report. To ensure routing table accuracy on all coordinators, it is recommended to set this value higher than 1.

Configuring the Library > Wireless Protocol > MiWi and MiWi PRO Networking Protocols > MiWi PRO Networking Protocol > ROUTING_TABLE_BROADCAST Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

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.

Macros

	Name	Description
<i>⊶</i> 0	BANK_SIZE	BANK_SIZE defines the number of packet can be received and stored to wait for handling in MiMAC layer.
~~	KEY_SEQUENCE_NUMBER	KEY_SEQUENCE_NUMBER defines the sequence number that is used to identify the key. Different key should have different sequence number, if multiple security keys are used in the application.
~0	SECURITY_LEVEL	SECURITY_LEVEL defines the security mode used in the application.
0 0	SECURITY_KEY_00	SECURITY_KEY_xx defines xxth byte of security key used in the block cipher. The length of the key depends on the key size of the block cipher.

Topics

Name	Description
MRF24J40 IEEE 802.15.4 Compliant 2.4GHz Transceiver	Following configurations can be used to regulate Microchip <u>MRF24J40</u> IEEE 802.15.4 compliant 2.4GHz transceiver.
SubGHz Transceivers	Microchip SubGHz transceivers share some common configurations.

<u>Configuring the Library</u> > <u>RF Transceivers</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help		Contents Index Home	Previous Up Next
BANK_SIZE Macro			
	С		
	#define BANK_SIZE :	2	
	Þ		

Description

BANK_SIZE defines the number of packet can be received and stored to wait for handling in MiMAC layer.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>BANK_SIZE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

KEY_SEQUENCE_NUMBER Macro

С

#define KEY_SEQUENCE_NUMBER 0×00

Description

KEY_SEQUENCE_NUMBER defines the sequence number that is used to identify the key. Different key should have different sequence number, if multiple security keys are used in the application.

<u>Configuring the Library > RF Transceivers ></u> <u>KEY_SEQUENCE_NUMBER Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

SECURITY_LEVEL Macro

С

#define SECURITY_LEVEL SEC_LEVEL_CCM_16

Description

SECURITY_LEVEL defines the security mode used in the application.

<u>Configuring the Library > RF Transceivers > SECURITY_LEVEL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

SECURITY_KEY_00 Macro

С

#define SECURITY_KEY_00 0x00

Description

SECURITY_KEY_xx defines xxth byte of security key used in the block cipher. The length of the key depends on the key size of the block cipher.

<u>Configuring the Library > RF Transceivers > SECURITY_KEY_00 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MRF24J40 IEEE 802.15.4 Compliant 2.4GHz Transceiver

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

Macros

	Name	Description
~~0	TURBO_MODE	TURBO_MODE enables <u>MRF24J40</u> transceiver to perform the communication in proprietary modulation, which is not compliant to IEEE 802.15.4 specification. The data rate at turbo mode is up to 625Kbps.
→0	VERIFY_TRANSMIT	VERIFY_TRANSMIT configures the MRF24J40 transceiver to transmit data in a block procedure, which ensures finish transmission before continue other task. This block procedure ensures the delivery state of transmitting known to the upper protocol layer, thus may be necessary to detect transmission failure. However, this block procedure slightly lower the throughput

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>MRF24J40 IEEE 802.15.4</u> <u>Compliant 2.4GHz Transceiver</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TURBO_MODE Macro

С

#define TURBO_MODE

Description

TURBO_MODE enables <u>MRF24J40</u> transceiver to perform the communication in proprietary modulation, which is not compliant to IEEE 802.15.4 specification. The data rate at turbo mode is up to 625Kbps.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>MRF24J40 IEEE 802.15.4</u> <u>Compliant 2.4GHz Transceiver</u> > <u>TURBO_MODE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

VERIFY_TRANSMIT Macro

С

#define VERIFY_TRANSMIT

Description

VERIFY_TRANSMIT configures the <u>MRF24J40</u> transceiver to transmit data in a block procedure, which ensures finish transmission before continue other task. This block procedure ensures the delivery state of transmitting known to the upper protocol layer, thus may be necessary to detect transmission failure. However, this block procedure slightly lower the throughput

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>MRF24J40 IEEE 802.15.4</u> <u>Compliant 2.4GHz Transceiver</u> > <u>VERIFY_TRANSMIT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright o 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

SubGHz Transceivers

Microchip SubGHz transceivers share some common configurations.

Macros

	Name	Description
Ŷ	ACK_INFO_SIZE	ACK_INFO_SIZE def number of acknowled information structure stored to avoid duplic to the protocol layer.
~	<u>CCA_RETRIES</u>	CCA_RETRIES defin maximum retries can performed in the case Channel Assessment the CCA procedure. (procedure perform Co <u>CCA_TIMES</u> and che times of CCA failure to number defined in <u>CCA_THRESHOLD</u> . case that CCA failure beyond <u>CCA_THRES</u> the whole procedure repeated up to CCA_ times before transmis failure can be flaggec
ş	CCA_THRESHOLD	-65dB limit for CCA tł values (as sampling a data/preamble) can u

		values for preamble (802.11 standard)
~	CCA_TIMES	CCA_TIMES defines number of Clear Cha Assessment in the CO procedure. CCA proc perform CCA for CCA and check if the times failure beyond the nu defined in <u>CCA_THR</u> In the case that CCA times is beyond <u>CCA_THRESHOLD</u> , procedure must be re to <u>CCA_RETRIES</u> tin transmission failure c flagged.
~	ENABLE_ACK	ENABLE_ACK enable MRF89XA to automation send back an acknowledgement pa MiMAC layer after rec packet, when such acknowledgement is by the packet sender.
\$	ENABLE_CCA	ENABLE_CCA enable MRF89XA to perform Channel Assessemer transmitting data in N layer.
Ŷ	ENABLE_RETRANSMISSION	ENABLE_RETRANS enables <u>MRF89XA</u> tc retransmit the packet <u>RETRANSMISSION</u>

		ENABLE_ACK is defi proper acknowledgen packet is not received sender in predefined period.
ę	FRAME_COUNTER_UPDATE_INTERVAL	The interval to update counter in the NVM
Ŷ	LNA_GAIN	LNA_GAIN defines th IF gain for <u>MRF89XA</u> transceiver.
\$	RETRANSMISSION_TIMES	RETRANSMISSION_ defines the maximum that can be performed proper acknowledgen packet is not received predefined time perio <u>ENABLE_RETRANS</u> is defined.
~	SOURCE_ADDRESS_ABSENT	SOURCE_ADDRESS disable the stack to tr source address in the layer, if the destinatio care where the messa from. This feature is r application dependen feature is only availat transceivers that supp MiMAC frame format.
Ŷ	TX_POWER	TX_POWER defines power for <u>MRF89XA</u>

Topics

Name	Description
<u>MRF49XA SubGHz</u> <u>Transceiver</u>	Following configurations can be used to regulate Microchip <u>MRF49XA</u> subGHz transceiver.
<u>MRF89XA SubGHz</u> <u>Transceiver</u>	Following configurations can be used to regulate Microchip <u>MRF89XA</u> subGHz transceiver.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

MRF49XA SubGHz Transceiver

Following configurations can be used to regulate Microchip <u>MRF49XA</u> subGHz transceiver.

Macros

	Name	Description
Ŷ	BAND_915	BAND_915, BAND_868 or BAND_434 are three supported frequency band for Microchip <u>MRF49XA</u> . One and only one of the frequency band must be defined
~	<u>CRYSTAL_PPM</u>	CRYSTAL_PPM defines the accuracy of the external crystal in PPM
~	DATA_RATE_9600	define DATA_RATE_1200 define DATA_RATE_19200 define DATA_RATE_38400 define DATA_RATE_57600 define DATA_RATE_115200
\$	INFER_DEST_ADDRESS	INFER_DEST_ADDRESS enables inferred destination address mode, which does not transmit the destination address, but depends on the software CRC to infer the destination address. Infer destination address applies to only transceivers that support MiMAC frame format and the

		CRC engine that supports this feature.
\$	MAX_ALLOWED_TX_FAILURE	MAX_ALLOWED_TX_FAILURE defines the maximum number of tries to transmit a packet before a transmission failure can be issued to the upper protocol layer. Transmission failure under this condition usually due to timeout from MRF89XA pin switch.
Ŷ	RSSI_THRESHOLD	RSSI_THRESHOLD defines the threshold for the RSSI digital output
Ŷ	XTAL_LD_CAP	XTAL_LD_CAP defines the capacitor load on the external crystal as the clock to MRF49XA transceiver

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>MRF49XA SubGHz Transceiver</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

BAND_915 Macro

С

#define BAND_915

Description

BAND_915, BAND_868 or BAND_434 are three supported frequency band for Microchip <u>MRF49XA</u>. One and only one of the frequency band must be defined

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>MRF49XA SubGHz Transceiver > BAND_915 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CRYSTAL_PPM Macro

С

#define CRYSTAL_PPM 10

Description

CRYSTAL_PPM defines the accuracy of the external crystal in PPM

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>MRF49XA SubGHz Transceiver</u> > <u>CRYSTAL_PPM Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

DATA_RATE_9600 Macro

С

#define DATA_RATE_9600

Description

define DATA_RATE_1200 define DATA_RATE_19200 define DATA_RATE_38400 define DATA_RATE_57600 define DATA_RATE_115200

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>MRF49XA SubGHz Transceiver > DATA_RATE_9600 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

INFER_DEST_ADDRESS Macro

С

#define INFER_DEST_ADDRESS

Description

INFER_DEST_ADDRESS enables inferred destination address mode, which does not transmit the destination address, but depends on the software CRC to infer the destination address. Infer destination address applies to only transceivers that support MiMAC frame format and the CRC engine that supports this feature.

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>MRF49XA SubGHz Transceiver > INFER_DEST_ADDRESS Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MAX_ALLOWED_TX_FAILURE Macro

С

#define MAX_ALLOWED_TX_FAILURE 20

Description

MAX_ALLOWED_TX_FAILURE defines the maximum number of tries to transmit a packet before a transmission failure can be issued to the upper protocol layer. Transmission failure under this condition usually due to timeout from <u>MRF89XA</u> pin switch.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>MRF49XA SubGHz Transceiver</u> > <u>MAX_ALLOWED_TX_FAILURE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

RSSI_THRESHOLD Macro

С

#define RSSI_THRESHOLD RSSI_THRESHOLD_79

Description

RSSI_THRESHOLD defines the threshold for the RSSI digital output

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>MRF49XA SubGHz Transceiver > RSSI_THRESHOLD Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 XTAL_LD_CAP Macro
 C

 #define XTAL_LD_CAP XTAL LD CAP 10

Description

XTAL_LD_CAP defines the capacitor load on the external crystal as the clock to <u>MRF49XA</u> transceiver

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>MRF49XA SubGHz Transceiver > XTAL_LD_CAP Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MRF89XA SubGHz Transceiver

Following configurations can be used to regulate Microchip <u>MRF89XA</u> subGHz transceiver.

Macros

	Name	Description
~	BAND_902	BAND_902, <u>BAND_915</u> or BAND_863 (or BAND_950 - circuit dependent) are three supported frequency band for Microchip <u>MRF89XA</u> . One and only one of the frequency band must be defined
~	DATA_RATE_20	DATA_RATE_5, DATA_RATE_10, DATA_RATE_20, DATA_RATE_25, DATA_RATE_40, DATA_RATE_50, DATA_RATE_66, DATA_RATE_100 and DATA_RATE_200 are 10 data rates supported by Microchip <u>MRF89XA</u> transceivers in MiMAC interface. One and only one of the data rate must be defined

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>MRF89XA SubGHz Transceiver</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

BAND_902 Macro

С

#define BAND_902

Description

BAND_902, <u>BAND_915</u> or BAND_863 (or BAND_950 - circuit dependent) are three supported frequency band for Microchip <u>MRF89XA</u>. One and only one of the frequency band must be defined

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>MRF89XA SubGHz Transceiver</u> > <u>BAND_902 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

DATA_RATE_20 Macro

С

#define DATA_RATE_20

Description

DATA_RATE_5, DATA_RATE_10, DATA_RATE_20, DATA_RATE_25, DATA_RATE_40, DATA_RATE_50, DATA_RATE_66, DATA_RATE_100 and DATA_RATE_200 are 10 data rates supported by Microchip <u>MRF89XA</u> transceivers in MiMAC interface. One and only one of the data rate must be defined

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>MRF89XA SubGHz Transceiver > DATA_RATE_20 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ACK_INFO_SIZE Macro

С

#define ACK_INFO_SIZE 5

Description

ACK_INFO_SIZE defines the number of acknowledgement information structure can be stored to avoid duplicate packet to the protocol layer.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>ACK_INFO_SIZE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.
Contents | Index | Home

Previous | Up | Next

CCA_RETRIES Macro

С

#define CCA_RETRIES 4

Description

CCA_RETRIES defines the maximum retries can be performed in the case of Clear Channel Assessment failure in the CCA procedure. CCA procedure perform CCA for <u>CCA_TIMES</u> and check if the times of CCA failure beyond the number defined in <u>CCA_THRESHOLD</u>. In the case that CCA failure times is beyond <u>CCA_THRESHOLD</u>, the whole procedure must be repeated up to CCA_RETRIES times before transmission failure can be flagged.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>CCA_RETRIES Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.



Description

-65dB limit for CCA threshold values (as sampling at data/preamble) can use higher values for preamble (refer 802.11 standard)

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>CCA_THRESHOLD Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CCA_TIMES Macro

С

#define CCA_TIMES 5

Description

CCA_TIMES defines the total number of Clear Channel Assessment in the CCA procedure. CCA procedure perform CCA for CCA_TIMES and check if the times of CCA failure beyond the number defined in <u>CCA_THRESHOLD</u>. In the case that CCA failure times is beyond <u>CCA_THRESHOLD</u>, the whole procedure must be repeated up to <u>CCA_RETRIES</u> times before transmission failure can be flagged.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>CCA_TIMES Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ENABLE_ACK Macro

С

#define ENABLE_ACK

Description

ENABLE_ACK enables <u>MRF89XA</u> to automatically send back an acknowledgement packet in MiMAC layer after receiving a packet, when such acknowledgement is requested by the packet sender.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>ENABLE_ACK Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ENABLE_CCA Macro

С

#define ENABLE_CCA

Description

ENABLE_CCA enables <u>MRF89XA</u> to perform Clear Channel Assessement before transmitting data in MiMAC layer.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>ENABLE_CCA Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

ENABLE_RETRANSMISSION Macro

С

#define ENABLE_RETRANSMISSION

Description

ENABLE_RETRANSMISSION enables <u>MRF89XA</u> to retransmit the packet up to <u>RETRANSMISSION_TIMES</u>, if <u>ENABLE_ACK</u> is defined, and a proper acknowledgement packet is not received by the sender in predefined time period.

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>ENABLE_RETRANSMISSION Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

FRAME_COUNTER_UPDATE_INTERVAL Macro

С

#define FRAME_COUNTER_UPDATE_INTERVAL 1024

Description

The interval to update the frame counter in the NVM

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>FRAME_COUNTER_UPDATE_INTERVAL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help	Contents Index Home	Previous Up Next
LNA_GAIN Macro		
С		
#define LNA_GAIN LNA_GAIN_0_DB		
2		

Description

LNA_GAIN defines the internal IF gain for MRF89XA transceiver.

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>LNA_GAIN Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

RETRANSMISSION_TIMES Macro

С

#define RETRANSMISSION_TIMES 3

Description

RETRANSMISSION_TIMES defines the maximum retries that can be performed if a proper acknowledgement packet is not received in predefined time period, if <u>ENABLE_RETRANSMISSION</u> is defined.

<u>Configuring the Library > RF Transceivers > SubGHz Transceivers ></u> <u>RETRANSMISSION_TIMES Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

SOURCE_ADDRESS_ABSENT Macro

С

#define SOURCE_ADDRESS_ABSENT

Description

SOURCE_ADDRESS_ABSENT disable the stack to transmit the source address in the MAC layer, if the destination does not care where the message comes from. This feature is highly application dependent. This feature is only available for transceivers that support MiMAC frame format.

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>SOURCE_ADDRESS_ABSENT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 TX_POWER Macro
 C
 #define TX_POWER TX_POWER_1_DB

Description

TX_POWER defines the output power for MRF89XA

<u>Configuring the Library</u> > <u>RF Transceivers</u> > <u>SubGHz Transceivers</u> > <u>TX_POWER Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Library API

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

Topics

Name	Description
MiApp Interfaces	
MiMAC Interfaces	

Library API

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiApp Interfaces

Functions

	Name	Description
≡∳	MiApp_BroadcastPacket	This function broadcast a message in the <u>TxBuffer</u> .
=∳	MiApp_ConnectionMode	This function set the current connection mode.
≡∳	MiApp_DiscardMessage	This function discard the current message for the application and notify the protocol layer that it is ready to receive the next message.
=∳	MiApp_EstablishConnection	This function establish a connection with one or more nodes in an existing PAN.
≓ ∳	MiApp_InitChannelHopping	This function tries to start a channel hopping (frequency agility) procedure
=♦	MiApp_MessageAvailable	This function return a boolean if a message is available for the application
≓∳	MiApp_NoiseDetection	This function perform a noise scan and returns the channel with least noise
≡∳	MiApp_RemoveConnection	This function remove connection(s) in connection

		table
=•	MiApp_ResyncConnection	This function tries to resynchronize the lost connection with peers, probably due to channel hopping
≡∳	MiApp_SearchConnection	This function perform an active scan to locate operating PANs in the neighborhood.
≡\$	MiApp_SetChannel	This function set the operating channel for the RF transceiver
≡ ∳	MiApp_StartConnection	This function start a PAN without connected to any other devices
-	MiApp_TransceiverPowerState	This function put the RF transceiver into different power state. i.e. Put the RF transceiver into sleep or wake it up. This function put the RF transceiver into different power state. i.e. Put the RF transceiver into sleep or wake it up.
- :	MiApp_UnicastAddress	This function unicast a message in the <u>TxBuffer</u> to the device with DestinationAddress
=\$	MiApp_UnicastConnection	This function unicast a message in the TxBuffer to

	the device with the input
	ConnectionIndex in the
	connection table.

Macros

	Name	Description
~~O	<u>MiApp_FlushTx</u>	This macro reset the pointer of the TX buffer. This function is usually called before filling application payload.
Ŷ	MiApp_WriteData	This macro writes one byte of application payload to the TX buffer.

Topics

Name	Description
Call Back Functions	

Library API > MiApp Interfaces

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_BroadcastPacket Function

```
C

<u>BOOL</u> MiApp_BroadcastPacket(

<u>BOOL</u> SecEn

);
```

Description

This is the primary user interface function for the application layer to broadcast a message. The application payload is filled in the global char array <u>TxBuffer</u>.

Preconditions

Protocol initialization has been done.

Parameters

Parameters	Description
BOOL SecEn	The boolean indicates if the application payload needs to be secured before transmission.

Returns

A boolean to indicates if the broadcast procedure is successful.

Remarks

None

Example

// Secure and then broadcast the message stored in MiApp_BroadcastPacket(TRUE);

Library API > MiApp Interfaces > MiApp_BroadcastPacket Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_ConnectionMode Function

```
C
void MiApp_ConnectionMode(
BYTE Mode
);
```

Description

This is the primary user interface function for the application layer to configure the way that the host device accept connection request.

Preconditions

Protocol initialization has been done.

Parameters

Parameters	Description	
BYTE Mode	The mode to accept connection request. The privilege for those modes decreases gradually as defined. The higher privilege mode has all the rights of the lower privilege modes. The possible modes are	
	 ENABLE_ALL_CONN Enable response to all connection request ENABLE_PREV_CONN Enable response to connection request from device already in the connection table. ENABLE_ACTIVE_SCAN_RSP Enable response to active scan only DISABLE_ALL_CONN Disable 	

response to connection request, including an acitve scan request.

Returns

None

Remarks

None

Example

Copy Code

// Enable all connection request
MiApp_ConnectionMode(ENABLE ALL CONN);

Library API > MiApp Interfaces > MiApp_ConnectionMode Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiApp_DiscardMessage Function

С

void MiApp_DiscardMessage();

Description

This is the primary user interface functions for the application layer to discard the current active message, release the system resources used and ready to receive the next message. It is must be called after finish handling the message, otherwise, no further message can be received.

Preconditions

Protocol initialization has been done. A message has been received by the application layer.

Returns

None

Remarks

None

Example

```
Copy Code
if( TRUE == MiApp MessageAvailable() )
{
    // handle the received message in global variab
    // discard the received message after processin
```

<u>Library API > MiApp Interfaces > MiApp_DiscardMessage Function</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

}

Contents | Index | Home

MiApp_EstablishConnection Function

```
C

BYTE MiApp_EstablishConnection(

BYTE ActiveScanIndex,

BYTE Mode

);
```

Description

This is the primary user interface function for the application layer to start communication with an existing PAN. For P2P protocol, this function call can establish one or more connections. For network protocol, this function can be used to join the network, or establish a virtual socket connection with a node out of the radio range. There are multiple ways to establish connection(s), all depends on the input parameters.

Preconditions

Protocol initialization has been done. If only to establish connection with a predefined device, an active scan must be performed before and valid active scan result has been saved.

Parameters

Parameters	Description
BYTE ActiveScanIndex	The index of the target device in the ActiveScanResults array, if a predefined device is targeted. If the value of ActiveScanIndex is 0xFF, the protocol stack will try to establish a connection with any device.

BYTE Mode	The mode to establish a connection. This parameter is generally valid in
a network protocol. The possible modes are	 <u>CONN_MODE_DIRECT</u> Establish a connection within radio range. <u>CONN_MODE_INDIRECT</u> Establish a virtual connection with a device that may be in or out of the radio range. This mode sometimes is called cluster socket, which is only valid for network protocol. The PAN Coordinator will be involved to establish a virtual indirect socket connection.

Returns

The index of the peer device on the connection table.

Remarks

If more than one connections have been established through this function call, the return value points to the index of one of the peer devices.

Example

Copy Code

// Establish one or more connections with any devic
PeerIndex = MiApp_EstablishConnection(0xFF, CONN_MO

Library API > MiApp Interfaces > MiApp_EstablishConnection Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright o 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

MiApp_FlushTx Macro

С

#define MiApp_FlushTx {TxData = PAYLOAD_START;}

Description

This macro reset the pointer of the TX buffer. This function is usually called before filling application payload.

Preconditions

Protocol initialization has been done.

Returns

None

Remarks

None

Example

Copy Code

```
MiApp_FlushTx();
<u>MiApp_WriteData</u>(AppPayload[0]);
<u>MiApp_WriteData</u>(AppPayload[1]);
```

Library API > MiApp Interfaces > MiApp_FlushTx Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_InitChannelHopping Function



Description

This is the primary user interface function for the application to do energy scan to locate the channel with least noise. If the channel is not current operating channel, process of channel hopping will be started.

Preconditions

Transceiver has been initialized

Parameters

Parameters	Description
DWORD ChannelMap	The bit map of the candicate channels which can be hopped to

Returns

a boolean to indicate if channel hopping is initiated

Example

Copy Code

```
// if condition meets, scan all possible channels a
// to the one with least noise
```

MiApp_InitChannelHopping(0xFFFFFFF);

Remark: The operating channel will change to the optimal channel with least noise

<u>Library API > MiApp Interfaces > MiApp_InitChannelHopping Function</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_MessageAvailable Function

С

BOOL MiApp_MessageAvailable();

Description

This is the primary user interface functions for the application layer to call the Microchip proprietary protocol stack to check if a message is available for the application. The function will call the protocol stack state machine to keep the stack running. It is expected that this function should be called periodically in the application. In case a message is available, all information related to the recevied message will be stored in the global variable RxMessage in the format of <u>RECEIVED MESSAGE</u>.

Preconditions

Protocol initialization has been done.

Returns

A boolean to indicates if a message is available for application.

Remarks

None

Example

<u>Copy Code</u> **if**(TRUE == MiApp_MessageAvailable()) { // handle the received message in global variab

// discard the received message after processin MiApp DiscardMessage();

<u>Library API > MiApp Interfaces > MiApp_MessageAvailable Function</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright O 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

}

Contents | Index | Home

MiApp_NoiseDetection Function

```
C

BYTE MiApp_NoiseDetection(

DWORD ChannelMap,

BYTE ScanDuration,

BYTE DetectionMode,

OUTPUT BYTE * NoiseLevel

);
```

Description

This is the primary user interface functions for the application layer to perform noise detection on multiple channels.

Preconditions

Protocol initialization has been done.

Parameters

Parameters	Description
DWORD ChannelMap	The bit map of channels to perform noise scan. The 32-bit double word parameter use one bit to represent corresponding channels from 0 to 31. For instance, 0x0000003 represent to scan channel 0 and channel 1.
BYTE ScanDuration	The maximum time to perform scan on single channel. The value is from 5 to 14. The real time to perform scan can be calculated in following formula from IEEE 802.15.4 specification 960 * (2^ScanDuration + 1) * 10^(-6) second

BYTE DetectionMode	 The noise detection mode to perform the scan. The two possible scan modes are NOISE_DETECT_ENERGY Energy detection scan mode NOISE_DETECT_CS Carrier sense detection scan mode
BYTE *NoiseLevel	The noise level at the channel with least noise level

Returns

The channel that has the lowest noise level

Remarks

None

Example

Copy Code

BYTE NoiseLevel;
OptimalChannel = MiApp_NoiseDetection(0xFFFFFFF, 1

Library API > MiApp Interfaces > MiApp_NoiseDetection Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiApp_RemoveConnection Function

```
C
void MiApp_RemoveConnection(
BYTE ConnectionIndex
);
```

Description

This is the primary user interface function to disconnect connection(s). For a P2P protocol, it simply remove the connection. For a network protocol, if the device referred by the input parameter is the parent of the device calling this function, the calling device will get out of network along with its children. If the device referred by the input parameter is children of the device calling this function, the target device will get out of network.

Preconditions

Transceiver has been initialized. Node has establish one or more connections

Parameters

Parameters	Description
BYTE ConnectionIndex	The index of the connection in the connection table to be removed

Returns

None

Remarks

None

Example

Copy Code

MiApp_RemoveConnection(0x00);

<u>Library API > MiApp Interfaces > MiApp_RemoveConnection Function</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_ResyncConnection Function

```
C

<u>BOOL</u> MiApp_ResyncConnection(

<u>BYTE</u> ConnectionIndex,

<u>DWORD</u> ChannelMap

);
```

Description

This is the primary user interface function for the application to resynchronize a lost connection. For a RFD device that goes to sleep periodically, it may not receive the channel hopping command that is sent when it is sleep. The sleeping RFD device depends on this function to hop to the channel that the rest of the PAN has jumped to. This function call is usually triggered by continously communication failure with the peers.

Preconditions

Transceiver has been initialized

Parameters

Parameters	Description
DWORD ChannelMap	The bit map of channels to perform noise scan. The 32-bit double word parameter use one bit to represent corresponding channels from 0 to 31. For instance, 0x0000003 represent to scan channel 0 and channel 1.

Returns

a boolean to indicate if resynchronization of connection is successful

Example

Copy Code

// Sleeping RFD device resync with its associated d
// in the connection table
MiApp_ResyncConnection(0, 0xFFFFFFF);

Remark: If operation is successful, the wireless node will be hopped to the channel that the rest of the PAN is operating on.

<u>Library API > MiApp Interfaces > MiApp_ResyncConnection Function</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_SearchConnection Function

```
C

<u>BYTE</u> MiApp_SearchConnection(

<u>BYTE</u> ScanDuration,

<u>DWORD</u> ChannelMap

);
```

Description

This is the primary user interface function for the application layer to perform an active scan. After this function call, all active scan response will be stored in the global variable <u>ActiveScanResults</u> in the format of structure <u>ACTIVE_SCAN_RESULT</u>. The return value indicates the total number of valid active scan response in the active scan result array.

Preconditions

Protocol initialization has been done.

Parameters

Parameters	Description
BYTE ScanDuration	The maximum time to perform scan on single channel. The value is from 5 to 14. The real time to perform scan can be calculated in following formula from IEEE 802.15.4 specification 960 * (2^ScanDuration + 1) * 10^(-6) second
DWORD ChannelMap	The bit map of channels to perform noise

scan. The 32-bit double word parameter use
one bit to represent corresponding channels
from 0 to 31. For instance, 0x00000003
represent to scan channel 0 and channel 1.

Returns

The number of valid active scan response stored in the global variable <u>ActiveScanResults</u>.

Remarks

None

Example

Copy Code

// Perform an active scan on all possible channels
NumOfActiveScanResponse = MiApp_SearchConnection(10

<u>Library API > MiApp Interfaces > MiApp_SearchConnection Function</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.
MiApp_SetChannel Function

```
C
BOOL MiApp_SetChannel(
BYTE channel
);
```

Description

This is the primary user interface function to set the operating channel for the RF transceiver. Valid channels are from 0 to 31. Depends on the RF transceiver, its frequency band, data rate and other settings, not all channels are available. If input channel is not available under the current condition, the current operating channel will not be change, and the return value will be FALSE. Otherwise, the return value will be TRUE.

Preconditions

Protocol initialization has been done.

Parameters

Parameters	Description
BYTE Channel	The channel to as the future operating channel. Channels higher than 31 are invalid. Valid channels between 0-31 depends on a lot of factors

Returns

a boolean to indicate if channel change has been performed successfully

Remarks

None

Example

Copy Code
if(TRUE == MiApp_SetChannel(15))
{
 // channel changes successfully
}

<u>Library API > MiApp Interfaces > MiApp_SetChannel Function</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_StartConnection Function

```
C

BOOL MiApp_StartConnection(

BYTE Mode,

BYTE ScanDuration,

DWORD ChannelMap

);
```

Description

This is the primary user interface function for the application layer to a PAN. Usually, this fucntion is called by the PAN Coordinator who is the first in the PAN. The PAN Coordinator may start the PAN after a noise scan if specified in the input mode.

Preconditions

Protocol initialization has been done.

Parameters

Parameters	Description	
BYTE Mode	Whether to start a PAN after a noise scan. Possible modes are	
	 <u>START_CONN_DIRECT</u> Start PAN directly without noise scan <u>START_CONN_ENERGY_SCN</u> Perform an energy scan first, then start the PAN on the channel with least noise. <u>START_CONN_CS_SCN</u> Perform a 	

	carrier-sense scan first, then start the PAN on the channel with least noise.
BYTE ScanDuration	The maximum time to perform scan on single channel. The value is from 5 to 14. The real time to perform scan can be calculated in following formula from IEEE 802.15.4
specification	960 * (2^ScanDuration + 1) * 10^(-6) second ScanDuration is discarded if the connection mode is <u>START_CONN_DIRECT</u> .
DWORD ChannelMap	The bit map of channels to perform noise scan. The 32-bit double word parameter use one bit to represent corresponding channels from 0 to 31. For instance, 0x0000003 represent to scan channel 0 and channel 1. ChannelMap is discarded if the connection mode is <u>START_CONN_DIRECT</u> .

Returns

a boolean to indicate if PAN has been started successfully.

Remarks

None

Example

Copy Code

// start the PAN on the least noisy channel after s
MiApp_StartConnection(START CONN ENERGY SCN, 10, 0x

Library API > MiApp Interfaces > MiApp_StartConnection Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_TransceiverPowerState Function



Description

This is the primary user interface functions for the application layer to put RF transceiver into sleep or wake it up. This function is only available to those wireless nodes that may have to disable the transceiver to save battery power.

This is the primary user interface functions for the application layer to put RF transceiver into sleep or wake it up. This function is only available to those wireless nodes that may have to disable the transceiver to save battery power.

Preconditions

Protocol initialization has been done.

Protocol initialization has been done.

Parameters

Parameters	Description	
BYTE Mode	The mode of power state for the RF transceiver to be set. The possible power states are following	
	 <u>POWER_STATE_SLEEP</u> The deep sleep mode for RF transceiver 	

 POWER_STATE_WAKEUP Wake up state, or operating state for RF transceiver POWER_STATE_WAKEUP_DR Put device into wakeup mode and then transmit a data request to the device's associated device
The mode of power state for the RF transceiver to be set. The possible power states are following
 POWER_STATE_SLEEP The deep sleep mode for RF transceiver POWER_STATE_WAKEUP Wake up state, or operating state for RF transceiver POWER_STATE_WAKEUP_DR Put device into wakeup mode and then transmit a data request to the device's associated device

Returns

The status of the operation. The following are the possible status

- <u>SUCCESS</u> Operation successful
- ERR_TRX_FAIL Transceiver fails to go to sleep or wake up
- <u>ERR_TX_FAIL</u> Transmission of Data Request command failed. Only available if the input mode is <u>POWER_STATE_WAKEUP_DR</u>.
- <u>ERR_RX_FAIL</u> Failed to receive any response to Data Request command. Only available if input mode is <u>POWER_STATE_WAKEUP_DR</u>.
- ERR_INVLAID_INPUT Invalid input mode.

The status of the operation. The following are the possible status

- <u>SUCCESS</u> Operation successful
- ERR_TRX_FAIL Transceiver fails to go to sleep or wake up
- <u>ERR_TX_FAIL</u> Transmission of Data Request command failed. Only available if the input mode is <u>POWER_STATE_WAKEUP_DR</u>.
- <u>ERR_RX_FAIL</u> Failed to receive any response to Data Request command. Only available if input mode is <u>POWER_STATE_WAKEUP_DR</u>.
- ERR_INVLAID_INPUT Invalid input mode.

Remarks

None

None

Example 1

```
Copy Code

// put RF transceiver into sleep

MiApp_TransceiverPowerState(POWER STATE SLEEP;

// Put the MCU into sleep

Sleep();

// wakes up the MCU by WDT, external interrupt or a

// make sure that RF transceiver to wake up and sen

MiApp_TransceiverPowerState(POWER_STATE_WAKEUP_DR);
```

Example 2

```
<u>Copy Code</u>
// put RF transceiver into sleep
MiApp_TransceiverPowerState(<u>POWER STATE SLEEP;</u>
// Put the MCU into sleep
```

Sleep();

// wakes up the MCU by WDT, external interrupt or a

// make sure that RF transceiver to wake up and sen
MiApp_TransceiverPowerState(POWER_STATE_WAKEUP_DR);

<u>Library API > MiApp Interfaces > MiApp_TransceiverPowerState</u> <u>Function</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright o 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_UnicastAddress Function

```
C

BOOL MiApp_UnicastAddress(

BYTE * DestAddress,

BOOL PermanentAddr,

BOOL SecEn

);
```

Description

This is one of the primary user interface functions for the application layer to unicast a message. The destination device is specified by the input parameter DestinationAddress. The application payload is filled in the global char array TxBuffer.

Preconditions

Protocol initialization has been done.

Parameters

Parameters	Description
BYTE * DestinationAddress	The destination address of the unicast
BOOL PermanentAddr	The boolean to indicate if the destination address above is a permanent address or alternative network address. This parameter is only used in a network protocol.
BOOL SecEn	The boolean indicates if the application payload needs to be secured before transmission.

R	et		r	n	ς
	Cι	.u			Э

Ш

A boolean to indicates if the unicast procedure is successful.

Remarks

None

Example

Copy Code

// Secure and then broadcast the message stored in
// specified in the input parameter.
MiApp_UnicastAddress(DestAddress, TRUE, TRUE);

Library API > MiApp Interfaces > MiApp_UnicastAddress Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiApp_UnicastConnection Function

```
C

BOOL MiApp_UnicastConnection(

BYTE ConnectionIndex,

BOOL SecEn

);
```

Description

This is one of the primary user interface functions for the application layer to unicast a message. The destination device is in the connection table specified by the input parameter ConnectionIndex. The application payload is filled in the global char array <u>TxBuffer</u>.

Preconditions

Protocol initialization has been done. The input parameter ConnectionIndex points to a valid peer device in the connection table.

Parameters

Parameters	Description
BYTE ConnectionIndex	The index of the destination device in the connection table.
BOOL SecEn	The boolean indicates if the application payload needs to be secured before transmission.

Returns

A boolean to indicates if the unicast procedure is successful.

Remarks

None

Example

Copy Code

// Secure and then unicast the message stored in Tx
// the connection table
MiApp_UnicastConnection(0, TRUE);

<u>Library API > MiApp Interfaces > MiApp_UnicastConnection Function</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiApp_WriteData Macro

С

```
#define MiApp_WriteData(a) TxBuffer[TxData++] = a
```

Description

This macro writes one byte of application payload to the TX buffer.

Preconditions

Protocol initialization has been done. <u>MiApp_FlushTx()</u> has been called before calling the first MiApp_WriteData for the first time.

Parameters

Parameters	Description	
BYTE a	One byte of application payload to be written to the TX buffer	

Returns

None

Remarks

None

Example

Copy Code

MiApp_FlushTx();

MiApp_WriteData(AppPayload[0]); MiApp_WriteData(AppPayload[1]);

Library API > MiApp Interfaces > MiApp_WriteData Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

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.

Macros

	Name	Description
~	MiApp_CB_AllowConnection	MiApp_CB_AllowConnection called by the stack to the application layer to notify the application that a node is tryi join the network. <u>Application</u> return a boolean to indicate in joining is allowed or denied.
~~	MiApp_CB_RFDAcknowledgement	MiApp_CB_RFDAcknowledg is called by the MiWi or MiWi stack to notify the application a network layer acknowledge is received from a sleeping d MiApp function <u>MiApp_UnicastAddress</u> or <u>MiApp_UnicastConnection</u> ca wait for the acknowledgemer a sleep device, so call back function has to be used to no the application layer that

acknowledgement has been received.

Library API > MiApp Interfaces > Call Back Functions

Microchip My Application xx.yy - [Jan 1, 2009] Copyright O 2009 Microchip Technology, Inc. All rights reserved.

MiApp_CB_AllowConnection Macro

С

#define MiApp_CB_AllowConnection(handleInConnectionTa

Description

MiApp_CB_AllowConnection is called by the stack to the application layer to notify the application that a node is trying to join the network. <u>Application</u> may return a boolean to indicate if joining is allowed or denied.

<u>Library API > MiApp Interfaces > Call Back Functions ></u> <u>MiApp_CB_AllowConnection Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

MiApp_CB_RFDAcknowledgement Macro

С

#define MiApp_CB_RFDAcknowledgement(SourceShortAddres

Description

MiApp_CB_RFDAcknowledgement is called by the MiWi or MiWi PRO stack to notify the application that a network layer acknowledgement is received from a sleeping device. MiApp function <u>MiApp_UnicastAddress</u> or <u>MiApp_UnicastConnection</u> cannot wait for the acknowledgement from a sleep device, so call back function has to be used to notify the application layer that acknowledgement has been received.

<u>Library API > MiApp Interfaces > Call Back Functions ></u> <u>MiApp_CB_RFDAcknowledgement Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

MiMAC Interfaces

Functions

	Name	Description
= \$	MiMAC_ChannelAssessment	This function perform the noise detection on current operating channel This function perform the noise detection on current operating channel
-	MiMAC_DiscardPacket	This function discard the current packet received from the RF transceiver This function discard the current packet received from the RF transceiver
÷	MiMAC_Init	This function initialize MiMAC layer
≓ ∳	MiMAC_PowerState	This function puts the RF transceiver into sleep or wake it up
= \$	MiMAC_ReceivedPacket	This function check if a new packet has been received by the RF transceiver This function check if a new packet has been received by the RF transceiver
≡∳	MiMAC_SendPacket	This function transmit a packet

≡\$	MiMAC_SetAltAddress	This function set the alternative network address and PAN identifier if applicable
- =	MiMAC_SetChannel	This function set the operating channel for the RF transceiver
÷	MiMAC_SetPower	This function set the output power for the RF transceiver

Library API > MiMAC Interfaces

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiMAC_ChannelAssessment Function



Description

This is the primary MiMAC interface for the protocol layer to perform the noise detection scan. Not all assessment modes are supported for all RF transceivers.

This is the primary MiMAC interface for the protocol layer to perform the noise detection scan. Not all assessment modes are supported for all RF transceivers.

Preconditions

MiMAC initialization has been done.

MiMAC initialization has been done.

Parameters

Parameters	Description
BYTE AssessmentMode	The mode to perform noise assessment. The possible assessment modes are
	 CHANNEL_ASSESSMENT_CARRIER_SENS Carrier sense detection mode CHANNEL_ASSESSMENT_ENERGY_DETE Energy detection mode The mode to perform noise assessment. The

possible assessment modes are
<u>CHANNEL_ASSESSMENT_CARRIER_SENS</u>
Carrier sense detection mode
 CHANNEL_ASSESSMENT_ENERGY_DETE
Energy detection mode

Returns

A byte to indicate the noise level at current channel.

A byte to indicate the noise level at current channel.

Remarks

None

None

Example 1

Copy Code

NoiseLevel = MiMAC_ChannelAssessment(CHANNEL_ASSESS

Example 2

Copy Code

NoiseLevel = MiMAC_ChannelAssessment(<u>CHANNEL_ASSESS</u>

Library API > MiMAC Interfaces > MiMAC_ChannelAssessment Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiMAC_DiscardPacket Function

С

```
void MiMAC_DiscardPacket();
```

Description

This is the primary MiMAC interface for the protocol layer to discard the current packet received from the RF transceiver.

This is the primary MiMAC interface for the protocol layer to discard the current packet received from the RF transceiver.

Preconditions

MiMAC initialization has been done.

MiMAC initialization has been done.

Returns

None

None

Remarks

None

None

Example 1



Example 2



Library API > MiMAC Interfaces > MiMAC_DiscardPacket Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 MiMAC_Init Function
 C
 BOOL MiMAC_Init(MACINIT_PARAM initValue);
 Mimac_Init(

Description

This is the primary MiMAC interface for the protocol layer to initialize the MiMAC layer. The initialization parameter is assigned in the format of structure MACINIT_PARAM.

Preconditions

MCU initialization has been done.

Parameters

Parameters	Description
MACINIT_PARAM initValue	Initialization value for MiMAC layer

Returns

A boolean to indicates if initialization is successful.

Remarks

None

Example

Copy Code

MiMAC_Init(initParameter);

Library API > MiMAC Interfaces > MiMAC_Init Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright O 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiMAC_PowerState Function

```
C

<u>BOOL</u> MiMAC_PowerState(

<u>BYTE</u> PowerState

);
```

Description

This is the primary MiMAC interface for the protocol layer to set different power state for the RF transceiver. There are minimal power states defined as deep sleep and operating mode. Additional power states can be defined for individual RF transceiver depends on hardware design.

Preconditions

MiMAC initialization has been done.

Parameters

Parameters	Description
BYTE PowerState	The power state of the RF transceiver to be set to. The minimum definitions for all RF transceivers are
	 <u>POWER_STATE_DEEP_SLEEP</u> RF transceiver deep sleep mode. <u>POWER_STATE_OPERATE</u> RF transceiver operating mode.

Returns

A boolean to indicate if chaning power state of RF transceiver is successful.

Remarks

None

Example

Copy Code // Put RF transceiver into sleep MiMAC_PowerState(POWER_STATE_DEEP_SLEEP); // Put MCU to sleep Sleep(); // Wake up the MCU by WDT, external interrupt or an // Wake up the RF transceiver MiMAC_PowerState(POWER_STATE_OPERATE);

Library API > MiMAC Interfaces > MiMAC_PowerState Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright o 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiMAC_ReceivedPacket Function

С

BOOL MiMAC_ReceivedPacket();

Description

This is the primary MiMAC interface for the protocol layer to check if a packet has been received by the RF transceiver. When a packet has been received, all information will be stored in the global variable MACRxPacket in the format of <u>MAC_RECEIVED_PACKET</u>;

This is the primary MiMAC interface for the protocol layer to check if a packet has been received by the RF transceiver. When a packet has been received, all information will be stored in the global variable MACRxPacket in the format of <u>MAC_RECEIVED_PACKET</u>;

Preconditions

MiMAC initialization has been done.

MiMAC initialization has been done.

Returns

A boolean to indicate if a packet has been received by the RF transceiver.

A boolean to indicate if a packet has been received by the RF transceiver.

Remarks

None

None

Example 1

```
Copy Code
if( TRUE == MiMAC_ReceivedPacket() )
{
    // handle the raw data from RF transceiver
    // discard the current packet
    MiMAC_DiscardPacket();
}
```

Example 2



Library API > MiMAC Interfaces > MiMAC_ReceivedPacket Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiMAC_SendPacket Function

```
C
BOOL MiMAC_SendPacket(
<u>MAC_TRANS_PARAM</u> transParam,
BYTE * MACPayload,
BYTE MACPayloadLen
);
```

Description

This is the primary MiMAC interface for the protocol layer to send a packet. Input parameter transParam configure the way to transmit the packet.

Preconditions

MiMAC initialization has been done.

Parameters

Parameters	Description
MAC_TRANS_PARAM transParam	The struture to configure the transmission way
BYTE * MACPaylaod	Pointer to the buffer of MAC payload
BYTE MACPayloadLen	The size of the MAC payload

Returns

A boolean to indicate if a packet has been received by the RF

transceiver.

Remarks

None

Example

Copy Code

MiMAC_SendPacket(transParam, MACPayload, MACPayload

Library API > MiMAC Interfaces > MiMAC_SendPacket Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 MiMAC_SetAltAddress Function
 C
 C
 BOOL MiMAC_SetAltAddress(BYTE * Address, BYTE * PANID
 Function

Description

This is the primary MiMAC interface for the protocol layer to set alternative network address and/or PAN identifier. This function call applies to only IEEE 802.15.4 compliant RF transceivers. In case alternative network address is not supported, this function will return FALSE.

Preconditions

MiMAC initialization has been done.

Parameters

Parameters	Description
BYTE * Address	The alternative network address of the host device.
BYTE * PANID	The PAN identifier of the host device

Returns

A boolean to indicates if setting alternative network address is successful.

Remarks

None

Example

```
<u>Copy Code</u>

<u>WORD</u> NetworkAddress = 0x0000;

<u>WORD</u> PANID = 0x1234;

MiMAC_SetAltAddress(&NetworkAddress, &PANID);
```

Library API > MiMAC Interfaces > MiMAC_SetAltAddress Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiMAC_SetChannel Function

```
C

<u>BOOL</u> MiMAC_SetChannel(

<u>BYTE</u> channel,

<u>BYTE</u> offsetFreq

);
```

Description

This is the primary MiMAC interface for the protocol layer to set the operating frequency of the RF transceiver. Valid channel number are from 0 to 31. For different frequency band, data rate and other RF settings, some channels from 0 to 31 might be unavailable. Paramater offsetFreq is used to fine tune the center frequency across the frequency band. For transceivers that follow strict definition of channels, this parameter may be discarded. The center frequency is calculated as (LowestFrequency + Channel * ChannelGap + offsetFreq)

Preconditions

Hardware initialization on MCU has been done.

Parameters

Parameters	Description
BYTE channel	Channel number. Range from 0 to 31. Not all channels are available under all conditions.
BYTE offsetFreq	Offset frequency used to fine tune the center frequency. May not apply to all RF
	transceivers
--	--------------
--	--------------

Returns

A boolean to indicates if channel setting is successful.

Remarks

None

Example

Copy Code

// Set center frequency to be exactly channel 12
MiMAC_SetChannel(12, 0);

Library API > MiMAC Interfaces > MiMAC_SetChannel Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 MiMAC_SetPower Function
 C
 SetPower (BOOL MiMAC_SetPower (BYTE outputPower);
 Environment Help

Description

This is the primary MiMAC interface for the protocol layer to set the output power for the RF transceiver. Whether the RF transceiver can adjust output power depends on the hardware implementation.

Preconditions

MiMAC initialization has been done.

Parameters

Parameters	Description
BYTE outputPower	RF transceiver output power.

Returns

A boolean to indicates if setting output power is successful.

Remarks

None

Example

Copy Code

// Set output power to be OdBm MiMAC_SetPower(TX_POWER_0_DB);

Library API > MiMAC Interfaces > MiMAC_SetPower Function

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Types

Structures

	Name	Description
*	CONNECTION_ENTRY	Peer Device Information in Connection Table This structure contains device information about the peer device of current node. It is the element structure for connection table. Due to the bank limitation in PIC18 MCU architecture, the size of CONNECTION_ENTRY must be dividable by 256 in case the array is across the bank. In this case, the user need to make sure that there is no problem
*	RECEIVED_MESSAGE	Received Message information This structure contains information about the received application message.
*	MAC_RECEIVED_PACKET	Content of the Received Message This structure contains all information of the received message
*	MAC_TRANS_PARAM	Parameters to Transmit a Packet This structure contains configurations to transmit a packet

U U

IJ

Types

	Name	Description
>	BOOL	Undefined size
>	<u>CHAR</u>	8-bit signed
>	<u>SHORT</u>	16-bit signed
>	<u>BYTE</u>	8-bit unsigned
*	<u>LONG</u>	32-bit signed MPLAB C Compiler for PIC18 does not support 64-bit integers
\$	<u>WORD</u>	16-bit unsigned
*	DWORD	32-bit unsigned MPLAB C Compiler for PIC18 does not support 64-bit integers

Unions

	Name	Description
*	WORD_VAL	This is type WORD_VAL.
\$ >	DWORD_VAL	This is type DWORD_VAL.

Symbol Reference > <u>Types</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 BOOL Type
 Image: Content _ BOOL BOOL;
 Image: Content _ BOOL BOOL;

Description

Undefined size

Symbol Reference > <u>Types</u> > <u>BOOL Type</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 CHAR Type
 C

typedef signed char CHAR;

Description

8-bit signed

Symbol Reference > <u>Types</u> > <u>CHAR Type</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help	Contents Index Home	Previous Up Next
SHORT Type		
С		
typedef signed	short int SHORT;	
Description		

16-bit signed

Symbol Reference > <u>Types</u> > <u>SHORT Type</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 BYTE Type
 C
 C
 Vertical state
 Vertical state

 C
 typedef unsigned char BYTE;
 Vertical state
 Vertical state
 Vertical state

Description

8-bit unsigned

Symbol Reference > <u>Types</u> > <u>BYTE Type</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help	Contents Index Home	Previous Up Next
LONG Type		
С		
typedef signed l	.ong LONG;	

Description

32-bit signed MPLAB C Compiler for PIC18 does not support 64bit integers

Symbol Reference > <u>Types</u> > <u>LONG Type</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 WORD Type
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C
 C

16-bit unsigned

Symbol Reference > <u>Types</u> > <u>WORD Type</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

WORD_VAL Union

С			
<pre>typedef union {</pre>			
<u>WORD</u> Val;			
structPACK	ED {		
EXTENSION	<u>BYTE</u>	b0 :	1;
EXTENSION	<u>BYTE</u>	b1 :	1;
EXTENSION	<u>BYTE</u>	b2 :	1;
EXTENSION	<u>BYTE</u>	b3 :	1;
EXTENSION	<u>BYTE</u>	b4 :	1;
EXTENSION	<u>BYTE</u>	b5 :	1;
EXTENSION	<u>BYTE</u>	b6 :	1;
EXTENSION	<u>BYTE</u>	b7 :	1;
EXTENSION	<u>BYTE</u>	b8 :	1;
EXTENSION	<u>BYTE</u>	b9 :	1;
EXTENSION	<u>BYTE</u>	b10	: 1;
EXTENSION	<u>BYTE</u>	b11	: 1;
EXTENSION	<u>BYTE</u>	b12	: 1;
EXTENSION	<u>BYTE</u>	b13	: 1;
EXTENSION	<u>BYTE</u>	b14	: 1;
EXTENSION	<u>BYTE</u>	b15	: 1;
<pre>} byte;</pre>			
structPACK	ED {		
EXTENSION	<u>BYTE</u>	b0 :	1;
EXTENSION	<u>BYTE</u>	b1 :	1;
EXTENSION	<u>BYTE</u>	b2 :	1;
EXTENSION	<u>BYTE</u>	b3 :	1;
EXTENSION	<u>BYTE</u>	b4 :	1;
EXTENSION	<u>BYTE</u>	b5 :	1;
EXTENSION	<u>BYTE</u>	b6 :	1;
EXTENSION	<u>BYTE</u>	b7 :	1;
EXTENSION	<u>BYTE</u>	b8 :	1;
EXTENSION	<u>BYTE</u>	b9 :	1;
EXTENSION	<u>BYTE</u>	b10	: 1;



Description

This is type WORD_VAL.

Symbol Reference > <u>Types</u> > <u>WORD_VAL Union</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help	Contents Index Home	Previous Up Next
DWORD Type		
С		
typedef unsigned	d long DWORD;	
Description		

32-bit unsigned MPLAB C Compiler for PIC18 does not support 64-bit integers

Symbol Reference > <u>Types</u> > <u>DWORD Type</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

DWORD_VAL Union

С

<pre>typedef union {</pre>				
<u>DWORD</u> Val;				
structPACKE	ED {			
EXTENSION	<u>BYTE</u>	b0 :		1;
EXTENSION	<u>BYTE</u>	b1 :		1;
EXTENSION	<u>BYTE</u>	b2 :		1;
EXTENSION	<u>BYTE</u>	b3 :		1;
EXTENSION	<u>BYTE</u>	b4 :		1;
EXTENSION	<u>BYTE</u>	b5 :		1;
EXTENSION	<u>BYTE</u>	b6 :		1;
EXTENSION	<u>BYTE</u>	b7 :		1;
EXTENSION	<u>BYTE</u>	b8 :		1;
EXTENSION	<u>BYTE</u>	b9 :		1;
EXTENSION	<u>BYTE</u>	b10	:	1;
EXTENSION	<u>BYTE</u>	b11	:	1;
EXTENSION	<u>BYTE</u>	b12	:	1;
EXTENSION	<u>BYTE</u>	b13	:	1;
EXTENSION	<u>BYTE</u>	b14	:	1;
EXTENSION	<u>BYTE</u>	b15	:	1;
EXTENSION	<u>BYTE</u>	b16	:	1;
EXTENSION	<u>BYTE</u>	b17	:	1;
EXTENSION	<u>BYTE</u>	b18	:	1;
EXTENSION	<u>BYTE</u>	b19	:	1;
EXTENSION	<u>BYTE</u>	b20	:	1;
EXTENSION	<u>BYTE</u>	b21	:	1;
EXTENSION	<u>BYTE</u>	b22	:	1;
EXTENSION	<u>BYTE</u>	b23	:	1;
EXTENSION	<u>BYTE</u>	b24	:	1;
EXTENSION	<u>BYTE</u>	b25	:	1;
EXTENSION	<u>BYTE</u>	b26	:	1;
EXTENSION	<u>BYTE</u>	b27	:	1;
EXTENSION	<u>BYTE</u>	b28	:	1;

EXTENSION	<u>BYTE</u>	b29 : 1;
EXTENSION	<u>BYTE</u>	b30 : 1;
EXTENSION	<u>BYTE</u>	b31 : 1;
} word;		
structPACKE	ED {	
EXTENSION	<u>BYTE</u>	b0 : 1;
EXTENSION	<u>BYTE</u>	b1 : 1;
EXTENSION	<u>BYTE</u>	b2 : 1;
EXTENSION	<u>BYTE</u>	b3 : 1;
EXTENSION	<u>BYTE</u>	b4 : 1;
EXTENSION	<u>BYTE</u>	b5 : 1;
EXTENSION	<u>BYTE</u>	b6 : 1;
EXTENSION	<u>BYTE</u>	b7 : 1;
EXTENSION	<u>BYTE</u>	b8 : 1;
EXTENSION	<u>BYTE</u>	b9 : 1;
EXTENSION	BYTE	b10 : 1;
EXTENSION	BYTE	b11 : 1;
EXTENSION	BYIE	b12 : 1;
EXTENSION	BYIE	b13 : 1;
EXTENSION	BYIE	b14 : 1;
	BYIE	D15 : 1;
	BYIE	D16 : 1;
	BYIE	D17 : 1;
	BILE	b18 : 1;
	DYTE	$b19 \cdot 1$
	DIIE	$b20 \cdot 1,$
		$b21 \cdot 1$
		$b22 \cdot 1$
	BVTE	$h24 \cdot 1$
	BVTE	$b24 \cdot 1,$
	BYTE	b25 : 1, b26 : 1.
	BYTE	b27 · 1·
EXTENSION	BYTE	b28 : 1:
EXTENSION	BYTE	b29 : 1:
EXTENSION	BYTE	b30 : 1:
EXTENSION	BYTE	b31 : 1:

} byte;		
structPACK	ED {	
EXTENSION	<u>BYTE</u>	b0 : 1;
EXTENSION	<u>BYTE</u>	b1 : 1;
EXTENSION	<u>BYTE</u>	b2 : 1;
EXTENSION	<u>BYTE</u>	b3 : 1;
EXTENSION	<u>BYTE</u>	b4 : 1;
EXTENSION	<u>BYTE</u>	b5 : 1;
EXTENSION	<u>BYTE</u>	b6 : 1;
EXTENSION	<u>BYTE</u>	b7 : 1;
EXTENSION	<u>BYTE</u>	b8 : 1;
EXTENSION	<u>BYTE</u>	b9 : 1;
EXTENSION	<u>BYTE</u>	b10 : 1;
EXTENSION	<u>BYTE</u>	b11 : 1;
EXTENSION	<u>BYTE</u>	b12 : 1;
EXTENSION	<u>BYTE</u>	b13 : 1;
EXTENSION	<u>BYTE</u>	b14 : 1;
EXTENSION	<u>BYTE</u>	b15 : 1;
EXTENSION	<u>BYTE</u>	b16 : 1;
EXTENSION	<u>BYTE</u>	b17 : 1;
EXTENSION	<u>BYTE</u>	b18 : 1;
EXTENSION	<u>BYTE</u>	b19 : 1;
EXTENSION	<u>BYTE</u>	b20 : 1;
EXTENSION	<u>BYTE</u>	b21 : 1;
EXTENSION	<u>BYTE</u>	b22 : 1;
EXTENSION	<u>BYTE</u>	b23 : 1;
EXTENSION	<u>BYTE</u>	b24 : 1;
EXTENSION	<u>BYTE</u>	b25 : 1;
EXTENSION	<u>BYTE</u>	b26 : 1;
EXTENSION	<u>BYTE</u>	b27 : 1;
EXTENSION	<u>BYTE</u>	b28 : 1;
EXTENSION	<u>BYTE</u>	b29 : 1;
EXTENSION	<u>BYTE</u>	b30 : 1;
EXTENSION	<u>BYTE</u>	b31 : 1;
} wordUnion;		
structPACK	ED {	
EXTENSION	<u>BYTE</u>	b0 : 1;

	EXTENSION	<u>BYTE</u>	b1 : 1;
	EXTENSION	BYTE	b2 : 1;
	EXTENSION	<u>BYTE</u>	b3 : 1;
	EXTENSION	<u>BYTE</u>	b4 : 1;
	EXTENSION	<u>BYTE</u>	b5 : 1;
	EXTENSION	<u>BYTE</u>	b6 : 1;
	EXTENSION	<u>BYTE</u>	b7 : 1;
	EXTENSION	<u>BYTE</u>	b8 : 1;
	EXTENSION	<u>BYTE</u>	b9 : 1;
	EXTENSION	<u>BYTE</u>	b10 : 1;
	EXTENSION	<u>BYTE</u>	b11 : 1;
	EXTENSION	<u>BYTE</u>	b12 : 1;
	EXTENSION	<u>BYTE</u>	b13 : 1;
	EXTENSION	<u>BYTE</u>	b14 : 1;
	EXTENSION	<u>BYTE</u>	b15 : 1;
	EXTENSION	<u>BYTE</u>	b16 : 1;
	EXTENSION	<u>BYTE</u>	b17 : 1;
	EXTENSION	<u>BYTE</u>	b18 : 1;
	EXTENSION	<u>BYTE</u>	b19 : 1;
	EXTENSION	<u>BYTE</u>	b20 : 1;
	EXTENSION	<u>BYTE</u>	b21 : 1;
	EXTENSION	<u>BYTE</u>	b22 : 1;
	EXTENSION	<u>BYTE</u>	b23 : 1;
	EXTENSION	<u>BYTE</u>	b24 : 1;
	EXTENSION	<u>BYTE</u>	b25 : 1;
	EXTENSION	<u>BYTE</u>	b26 : 1;
	EXTENSION	<u>BYTE</u>	b27 : 1;
	EXTENSION	<u>BYTE</u>	b28 : 1;
	EXTENSION	<u>BYTE</u>	b29 : 1;
	EXTENSION	<u>BYTE</u>	b30 : 1;
	EXTENSION	<u>BYTE</u>	b31 : 1;
	} bits;		
}	DWORD_VAL;		

Description

This is type DWORD_VAL.

Symbol Reference > <u>Types</u> > <u>DWORD_VAL Union</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright O 2009 Microchip Technology, Inc. All rights reserved.

CONNECTION_ENTRY Structure

```
C

typedef struct __CONNECTION_ENTRY {

    WORD_VAL PANID;

    WORD_VAL AltAddress;

    BYTE Address[MY_ADDRESS_LENGTH];

    CONNECTION_STATUS status;

    BYTE PeerInfo[ADDITIONAL_NODE ID_SIZE];

} CONNECTION_ENTRY;
```

Description

Peer Device Information in Connection Table

This structure contains device information about the peer device of current node. It is the element structure for connection table. Due to the bank limitation in PIC18 MCU architecture, the size of CONNECTION_ENTRY must be dividable by 256 in case the array is across the bank. In this case, the user need to make sure that there is no problem

Members

Members	Description
WORD_VAL PANID;	PAN Identifier of the peer device. May not necessary in P2P protocol
WORD_VAL AltAddress;	Alternative address of the peer device. Not necessary in P2P protocol

BYTE Address[MY_ADDRESS_LENGTH];	Permanent address of peer device
BYTE PeerInfo[ADDITIONAL_NODE_ID_SIZE];	Additional Node ID information, if defined in application layer

п

Symbol Reference > <u>Types</u> > <u>CONNECTION_ENTRY Structure</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

ш

Contents | Index | Home

RECEIVED_MESSAGE Structure

```
С
typedef struct {
  union {
    BYTE Val;
    struct {
      BYTE broadcast : 2;
      BYTE ackReq : 1;
      BYTE secEn : 1;
      <u>BYTE</u> repeat : 1;
      BYTE command : 1;
      BYTE srcPrsnt : 1;
      BYTE altSrcAddr : 1;
    } bits;
  } flags;
 WORD_VAL SourcePANID;
  BYTE * SourceAddress;
  BYTE * Payload;
  BYTE PayloadSize;
  BYTE PacketRSSI;
  BYTE PacketLQI;
} RECEIVED_MESSAGE;
```

Description

Received Message information

This structure contains information about the received application message.

Members

Members	Description

BYTE broadcast : 2;	1: broadcast message
BYTE ackReq : 1;	1: sender request acknowledgement in MAC.
BYTE secEn : 1;	1: application payload has been secured
BYTE repeat : 1;	1: message received through a repeater
BYTE command : 1;	1: message is a command frame
BYTE srcPrsnt : 1;	1: source address present in the packet
BYTE altSrcAddr : 1;	1: source address is alternative network address
WORD_VAL SourcePANID;	PAN Identifier of the sender
BYTE * SourceAddress;	pointer to the source address
BYTE * Payload;	pointer to the application payload
BYTE PayloadSize;	application payload length
BYTE PacketRSSI;	RSSI value of the receive message
BYTE PacketLQI;	LQI value of the received message

Symbol Reference > <u>Types</u> > <u>RECEIVED_MESSAGE Structure</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MAC_RECEIVED_PACKET Structure

```
С
typedef struct {
  union {
    BYTE Val;
    struct {
      BYTE packetType : 2;
      BYTE broadcast : 1;
      BYTE secEn : 1;
      BYTE repeat : 1;
      BYTE ackReq : 1;
      BYTE destPrsnt : 1;
      BYTE sourcePrsnt : 1;
    } bits;
  } flags;
  BYTE * SourceAddress;
  BYTE * Payload;
  BYTE PayloadLen;
  BYTE RSSIValue;
  BYTE LQIValue;
  BOOL altSourceAddress;
 WORD VAL SourcePANID;
} MAC_RECEIVED_PACKET;
```

Description

Content of the Received Message

This structure contains all information of the received message

Members

Members	Description

BYTE packetType : 2;	type of packet. Possible types are
	 PACKET_TYPE_DATA - Data type PACKET_TYPE_COMMAND - Command type PACKET_TYPE_ACK - Acknowledgement type PACKET_TYPE_RESERVE - Reserved type
BYTE broadcast : 1;	1: broadcast, 0: unicast
BYTE secEn : 1;	1: secure the MAC payload, 0: send plain text
BYTE repeat : 1;	1: allow repeaters to forward the message, 0: send message directly
BYTE ackReq : 1;	1: acknowledgement required, 0: no acknowldgement
BYTE destPrsnt : 1;	1: destination address in the packet, 0: destination address not in the packet
BYTE sourcePrsnt : 1;	1: source address in the packet, 0: source address not in the packet
BYTE * SourceAddress;	Address of the Sender
BYTE * Payload;	Pointer to the payload
BYTE PayloadLen;	Payload size
BYTE RSSIValue;	RSSI value for the received packet
BYTE LQIValue;	LQI value for the received packet

BOOL altSourceAddress;	Source address is the alternative network address
WORD_VAL SourcePANID;	PAN ID of the sender

Symbol Reference > <u>Types</u> > <u>MAC_RECEIVED_PACKET Structure</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MAC_TRANS_PARAM Structure

```
С
typedef struct {
  union {
    BYTE Val;
    struct {
      BYTE packetType : 2;
      BYTE broadcast : 1;
      BYTE secEn : 1;
      BYTE repeat : 1;
      BYTE ackReq : 1;
      BYTE destPrsnt : 1;
      BYTE sourcePrsnt : 1;
    } bits;
  } flags;
  BYTE * DestAddress;
  BOOL altDestAddr;
  BOOL altSrcAddr;
  WORD_VAL DestPANID;
} MAC_TRANS_PARAM;
```

Description

Parameters to Transmit a Packet

This structure contains configurations to transmit a packet

Members

Members	Description	
BYTE packetType : 2;	type of packet. Possible types are	
	 <u>PACKET_TYPE_DATA</u> - Data type <u>PACKET_TYPE_COMMAND</u> - 	

	Command type • <u>PACKET_TYPE_ACK</u> - Acknowledgement type • <u>PACKET_TYPE_RESERVE</u> - Reserved type
BYTE broadcast : 1;	1: broadcast, 0: unicast
BYTE secEn : 1;	1: secure the MAC payload, 0: send plain text
BYTE repeat : 1;	1: allow repeaters to forward the message, 0: send message directly
BYTE ackReq : 1;	1: acknowledgement required, 0: no acknowldgement
BYTE destPrsnt : 1;	1: destination address in the packet, 0: destination address not in the packet
BYTE sourcePrsnt : 1;	1: source address in the packet, 0: source address not in the packet
BYTE * DestAddress;	destination address
BOOL altDestAddr;	use the alternative network address as destination in the packet
BOOL altSrcAddr;	use the alternative network address as source in the packet
WORD_VAL DestPANID;	PAN identifier of the destination

Symbol Reference > <u>Types</u> > <u>MAC_TRANS_PARAM Structure</u>

Copyright @ 2009 Microchip Technology, Inc. All rights reserved. $\underline{Contents \mid \underline{Index} \mid \underline{Home}$

Structs, Records, Enums

Structures

	Name	Description
*	ACTIVE_SCAN_RESULT	Active Scan result This structure contains information from active scan. <u>Application</u> layer will depend on this information to decide the way to establish connections.
*	<u>_CONNECTION_ENTRY</u>	Peer Device Information in Connection Table This structure contains device information about the peer device of current node. It is the element structure for connection table. Due to the bank limitation in PIC18 MCU architecture, the size of CONNECTION_ENTRY must be dividable by 256 in case the array is across the bank. In this case, the user need to make sure that there is no problem

Unions

	Name	Description
*	<u>CONNECTION_STATUS</u>	Status information of the connected peer information This structure contains the

		information regarding the status of the connected peer device.
*	CONNECTION_STATUS	Status information of the connected peer information This structure contains the information regarding the status of the connected peer device.
\$	BYTE_BITS	This is type BYTE_BITS.
\$	WORD_BITS	This is type WORD_BITS.

Symbol Reference > <u>Structs, Records, Enums</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

ACTIVE_SCAN_RESULT Structure

```
С
typedef struct {
  BYTE Channel;
  BYTE Address[MY_ADDRESS_LENGTH];
  WORD VAL PANID;
  BYTE RSSIValue;
  <u>BYTE</u> LQIValue;
  union {
    BYTE Val;
    struct {
      BYTE Role : 2;
      BYTE Sleep : 1;
      BYTE SecurityEn : 1;
      BYTE RepeatEn : 1;
      BYTE AllowJoin : 1;
      BYTE Direct : 1;
      BYTE altSrcAddr : 1;
    } bits;
  } Capability;
  BYTE PeerInfo[ADDITIONAL NODE ID SIZE];
} ACTIVE_SCAN_RESULT;
```

Description

Active Scan result

This structure contains information from active scan. <u>Application</u> layer will depend on this information to decide the way to establish connections.

Members

BYTE Channel;	Operating Channel of the PAN
BYTE Address[MY_ADDRESS_LENGTH];	Responding device address
WORD_VAL PANID;	PAN Identifier
BYTE RSSIValue;	RSSI value for the response
BYTE LQIValue;	LQI value for the response
BYTE Role : 2;	Role of the responding device in the PAN
BYTE Sleep : 1;	Whether the responding device goes to sleep when idle
BYTE SecurityEn : 1;	Whether the responding device is capable of securing the data
BYTE RepeatEn : 1;	Whether the responding device allow repeat
BYTE AllowJoin : 1;	Whether the responding device allows other device to join
BYTE Direct : 1;	Whether the responding device in radio range or through a repeater
BYTE altSrcAddr : 1;	Whether the Address is alternative network

	address or permanent address
BYTE PeerInfo[ADDITIONAL_NODE_ID_SIZE];	Additional Node ID information, if defined in application layer

Symbol Reference > <u>Structs, Records, Enums</u> > <u>ACTIVE_SCAN_RESULT Structure</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright O 2009 Microchip Technology, Inc. All rights reserved.

CONNECTION_STATUS Union

```
C
typedef union __CONNECTION_STATUS {
   BYTE Val;
   struct _CONNECTION_STAUTS_bits {
    BYTE RXOnWhenIdle : 1;
   BYTE directConnection : 1;
   BYTE longAddressValid : 1;
   BYTE shortAddressValid : 1;
   BYTE shortAddressValid : 1;
   BYTE isFamily : 1;
   BYTE filler : 1;
   BYTE filler : 1;
   BYTE isValid : 1;
   } bits;
} CONNECTION_STATUS;
```

Description

Status information of the connected peer information

This structure contains the information regarding the status of the connected peer device.

Members

Members	Description
BYTE RXOnWhenIdle : 1;	1 = transceiver always on, 0 = transceiver sleeps when idle
BYTE directConnection : 1;	1 = can talk to this device directly, 0 = must route to this device
BYTE	1 = long address valid, 0 = long address

longAddressValid : 1;	unknown
BYTE shortAddressValid : 1;	1 = short address valid, 0 = short address unknown
BYTE FinishJoin : 1;	1 = already finish joining procedure, 0 = in the process of join
BYTE isFamily : 1;	1 = family member (parent/child), 0 = not family
BYTE isValid : 1;	1 = this entry is valid, 0 = this entry is not valid

Symbol Reference > <u>Structs, Records, Enums</u> > <u>CONNECTION_STATUS Union</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright O 2009 Microchip Technology, Inc. All rights reserved.
Contents | Index | Home

Previous | Up | Next

BYTE_BITS Union

```
С
typedef union {
  BYTE Val;
  struct ___PACKED {
       EXTENSION <u>BYTE</u> b0
                                1;
                             1
       EXTENSION <u>BYTE</u> b1
                                1;
      _EXTENSION <u>BYTE</u> b2
                                1;
      EXTENSION BYTE b3
                                1;
                             1
      EXTENSION BYTE b4
                                1;
                             1
      _EXTENSION <u>BYTE</u> b5
                                1;
      _EXTENSION <u>BYTE</u> b6
                                1;
      EXTENSION BYTE b7
                                1;
  } bits;
  BYTE_BITS;
}
```

Description

This is type BYTE_BITS.

Symbol Reference > Structs, Records, Enums > BYTE_BITS Union

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

WORD_BITS Union

С

<pre>typedef union {</pre>		
WORD Val;		
structPACKE	ED {	
EXTENSION	<u>BYTE</u>	b0 : 1;
EXTENSION	<u>BYTE</u>	b1 : 1;
EXTENSION	<u>BYTE</u>	b2 : 1;
EXTENSION	<u>BYTE</u>	b3 : 1;
EXTENSION	<u>BYTE</u>	b4 : 1;
EXTENSION	<u>BYTE</u>	b5 : 1;
EXTENSION	<u>BYTE</u>	b6 : 1;
EXTENSION	<u>BYTE</u>	b7 : 1;
EXTENSION	<u>BYTE</u>	b8 : 1;
EXTENSION	<u>BYTE</u>	b9 : 1;
EXTENSION	<u>BYTE</u>	b10 : 1;
EXTENSION	<u>BYTE</u>	b11 : 1;
EXTENSION	<u>BYTE</u>	b12 : 1;
EXTENSION	<u>BYTE</u>	b13 : 1;
EXTENSION	<u>BYTE</u>	b14 : 1;
EXTENSION	<u>BYTE</u>	b15 : 1;
} byte;		
structPACKE	ED {	
EXTENSION	<u>BYTE</u>	b0 : 1;
EXTENSION	<u>BYTE</u>	b1 : 1;
EXTENSION	<u>BYTE</u>	b2 : 1;
EXTENSION	<u>BYTE</u>	b3 : 1;
EXTENSION	<u>BYTE</u>	b4 : 1;
EXTENSION	<u>BYTE</u>	b5 : 1;
EXTENSION	<u>BYTE</u>	b6 : 1;
EXTENSION	<u>BYTE</u>	b7 : 1;
EXTENSION	<u>BYTE</u>	b8 : 1;
EXTENSION	<u>BYTE</u>	b9 : 1;
EXTENSION	<u>BYTE</u>	b10 : 1;



Description

This is type WORD_BITS.

Symbol Reference > <u>Structs, Records, Enums</u> > <u>WORD_BITS Union</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Variables

Variables

	Name	Description
<i>~</i>	<u>ActiveScanResults</u>	The results for active scan, including the PAN identifier, signal strength and operating channel
~	AdditionalNodeID	AdditionalConnectionPayload variable array defines the additional information to identify a device on a P2P connection. This array will be transmitted with the P2P_CONNECTION_REQUEST command to initiate the connection between the two devices. Along with the long address of this device, this variable array will be stored in the P2P Connection Entry structure of the partner device. The size of this array is ADDITIONAL_CONNECTION_PAYLOAD, defined in P2PDefs.h. In this demo, this variable array is set to be empty.
	<u>ConnectionTable</u>	The peer device records for P2P connections
~	<u>rxMessage</u>	structure to store information for the received packet
<i>~</i>	TxBuffer	EXTERNAL VARIABLES
<i>•</i>	myLongAddress	



Symbol Reference > <u>Variables</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

ActiveScanResults Variable

С

ACTIVE SCAN RESULT ActiveScanResults[ACTIVE SCAN RES

Description

The results for active scan, including the PAN identifier, signal strength and operating channel

Symbol Reference > <u>Variables</u> > <u>ActiveScanResults Variable</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

AdditionalNodeID Variable

С

BYTE AdditionalNodeID[ADDITIONAL NODE ID SIZE] = {0x:

Description

AdditionalConnectionPayload variable array defines the additional information to identify a device on a P2P connection. This array will be transmitted with the P2P_CONNECTION_REQUEST command to initiate the connection between the two devices. Along with the long address of this device, this variable array will be stored in the P2P Connection Entry structure of the partner device. The size of this array is ADDITIONAL_CONNECTION_PAYLOAD, defined in P2PDefs.h. In this demo, this variable array is set to be empty.

Symbol Reference > <u>Variables</u> > <u>AdditionalNodeID Variable</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

ConnectionTable Variable

С

CONNECTION ENTRY ConnectionTable[CONNECTION SIZE];

Description

The peer device records for P2P connections

Symbol Reference > <u>Variables</u> > <u>ConnectionTable Variable</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 rxMessage Variable
 C

RECEIVED MESSAGE rxMessage;

Description

structure to store information for the received packet

Symbol Reference > <u>Variables</u> > <u>rxMessage Variable</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TxBuffer Variable

С

BYTE TxBuffer[TX_BUFFER_SIZE];

Description

EXTERNAL VARIABLES

Symbol Reference > <u>Variables</u> > <u>TxBuffer Variable</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

myLongAddress Variable

С

BYTE myLongAddress[MY_ADDRESS_LENGTH] = {EUI 0, EUI 1

Description

permanent address definition

Symbol Reference > <u>Variables</u> > <u>myLongAddress Variable</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

mySecurityKey Variable

С

ROM const unsigned char mySecurityKey[8] = {<u>SECURITY</u>

Description

security module.

Symbol Reference > <u>Variables</u> > <u>mySecurityKey Variable</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Definitions

Macros

	Name	Descri
~0	ASSOCIATION_ACCESS_DENIED	This is
~0	ASSOCIATION_PAN_FULL	This is
~0	ASSOCIATION_SUCCESSFUL	This is
~~	CHANNEL_ASSESSMENT_CARRIER_SENSE	This is CHANI
~~	CHANNEL_ASSESSMENT_ENERGY_DETECT	This is CHANI
~0	CMD_CHANNEL_HOPPING	This is
~0	CMD_DATA_REQUEST	This is
~0	CMD_MAC_DATA_REQUEST	This is
~0	CMD_P2P_CONNECTION_REMOVAL_REQUEST	This is CMD_I
~0	CMD_P2P_CONNECTION_REMOVAL_RESPONSE	This is CMD_I
~0	CMD_P2P_CONNECTION_REQUEST	This is
~0	CMD_P2P_CONNECTION_RESPONSE	This is
~0	CONN_MODE_DIRECT	This is

~	CONN_MODE_INDIRECT	This is
~	DISABLE_ALL_CONN	This is
~~	ENABLE_ALL_CONN	This is
~	ENABLE_CONSOLE	ENABL termina proces
~	ENABLE_PREV_CONN	This is
÷	ERR_INVALID_INPUT	This is
~	ERR_RX_FAIL	This is
~	ERR_TRX_FAIL	This is
÷	ERR_TX_FAIL	This is
~	IEEE_802_15_4	This is
~	KEY_SIZE	This is
~	LNA_GAIN_0_DB	00 [1:0
÷	LNA_GAIN_N_14_DB	This is
÷	LNA_GAIN_N_20_DB	This is
~	LNA_GAIN_N_6_DB	This is
~	NOISE_DETECT_CS	This is
~	NOISE_DETECT_ENERGY	This is
~	PACKET_TYPE_ACK	This is
~	PACKET_TYPE_COMMAND	This is
11		11

~	PACKET_TYPE_DATA	This is
~	PACKET_TYPE_MASK	This is
~	PACKET_TYPE_RESERVE	This is
~	POWER_STATE_DEEP_SLEEP	This is
~	POWER_STATE_OPERATE	This is
~	POWER_STATE_SLEEP	This is
~~	POWER_STATE_WAKEUP	This is
~	POWER_STATE_WAKEUP_DR	This is
~~	RSSI_THRESHOLD_103	This is
~	RSSI_THRESHOLD_73	This is
~~	RSSI_THRESHOLD_79	This is
~0	RSSI_THRESHOLD_85	This is
~~	RSSI_THRESHOLD_91	This is
~~	RSSI_THRESHOLD_97	This is
~	SECURITY_MASK	This is
~0	SOFTWARE_CRC	This is
~~	START_CONN_CS_SCN	This is
~	START_CONN_DIRECT	This is
~~	START_CONN_ENERGY_SCN	This is
~	SUCCESS	This is
1		

÷	TX_POWER_0_DB	This is
~	TX_POWER_N_10_DB	This is
~	TX_POWER_N_12_5_DB	This is
~	TX_POWER_N_15_DB	This is
~	TX_POWER_N_17_5_DB	This is
~	TX_POWER_N_2_5_DB	This is
~	TX_POWER_N_5_DB	This is
~	TX_POWER_N_7_5_DB	This is
~	XTAL_LD_CAP_10	This is
~	XTAL_LD_CAP_105	This is
~	XTAL_LD_CAP_11	This is
~~	XTAL_LD_CAP_115	This is
~	XTAL_LD_CAP_12	This is
~	XTAL_LD_CAP_125	This is
~	XTAL_LD_CAP_13	This is
~	XTAL_LD_CAP_135	This is
~~	XTAL_LD_CAP_14	This is
~	XTAL_LD_CAP_145	This is
~	XTAL_LD_CAP_15	This is
~	XTAL_LD_CAP_155	This is
1	II III III III III III III III III III	1

followir This is This is This is CMD_
This is This is This is CMD_
This is This is This is CMD_
This is This is CMD_
This is CMD_
This is
This is DATA
This is
EUI_x wireles
This is

~	FRAME_TYPE_COMMAND	This is
~	FRAME_TYPE_DATA	This is
~	FREQ_BAND	915M⊦
~~	MAC_COMMAND_ASSOCIATION_REQUEST	This is MAC_(
~0	MAC_COMMAND_ASSOCIATION_RESPONSE	This is MAC_(
~	MAC_COMMAND_BEACON_REQUEST	This is
~~	MAC_COMMAND_COORDINATOR_REALIGNMENT	This is MAC_(
~	MAC_COMMAND_DATA_REQUEST	This is
~	MAC_COMMAND_DISASSOCIATION_NOTIFICATION	This is MAC_(
Ŷ	MAC_COMMAND_ORPHAN_NOTIFICATION	This is MAC_(
~	MAC_COMMAND_PAN_ID_CONFLICT_NOTIFICATION	This is MAC_(
~0	STATUS_ACTIVE_SCAN	This is
~	STATUS_ENTRY_NOT_EXIST	This is
~	STATUS_EXISTS	This is
~	STATUS_NOT_ENOUGH_SPACE	This is
~	STATUS_NOT_SAME_PAN	This is
~	STATUS_SUCCESS	This is

ACK_REPORT_TYPETHROLE_COORDINATORTHROLE_FFD_END_DEVICETHROLE_PAN_COORDINATORTHOPEN_SOCKET_REQUESTTHOPEN_SOCKET_RESPONSETHOPEN_SOCKET_RESPONSETHCHANNEL_HOPPING_REQUESTTHEUI_ADDRESS_SEARCH_REQUESTTHMIWI_ACK_REQTHRESYNCHRONIZATION_REQUESTTH	μ	U Construction of the second sec	μ
Image: width of the synchround constraints of the	~	ACK_REPORT_TYPE	This is
Image: width of the synthesis of the synt	÷	ROLE_COORDINATOR	This is
Image: width of the sector o	~	ROLE_FFD_END_DEVICE	This is
Image: width of the synchronic synchron	÷	ROLE_PAN_COORDINATOR	This is
 OPEN_SOCKET_RESPONSE CHANNEL_HOPPING_REQUEST EUI_ADDRESS_SEARCH_REQUEST EUI_ADDRESS_SEARCH_RESPONSE MIWI_ACK_REQ RESYNCHRONIZATION_REQUEST 	~	OPEN_SOCKET_REQUEST	This is
 CHANNEL_HOPPING_REQUEST EUI_ADDRESS_SEARCH_REQUEST EUI_ADDRESS_SEARCH_RESPONSE MIWI_ACK_REQ RESYNCHRONIZATION_REQUEST 	~	OPEN_SOCKET_RESPONSE	This is
✓ EUI_ADDRESS_SEARCH_REQUEST Th ✓ EUI_ADDRESS_SEARCH_RESPONSE Th ✓ MIWI_ACK_REQ Th ✓ RESYNCHRONIZATION_REQUEST Th	÷	CHANNEL_HOPPING_REQUEST	This is
✓ EUI_ADDRESS_SEARCH_RESPONSE Th ✓ MIWI_ACK_REQ Th ✓ RESYNCHRONIZATION_REQUEST Th	~	EUI_ADDRESS_SEARCH_REQUEST	This is
MIWI_ACK_REQ The synchronization request	~0	EUI_ADDRESS_SEARCH_RESPONSE	This is
	~0	MIWI_ACK_REQ	This is
	~	RESYNCHRONIZATION_REQUEST	This is
RESYNCHRONIZATION_RESPONSE	~0	RESYNCHRONIZATION_RESPONSE	This is

Symbol Reference > Definitions

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

ASSOCIATION_ACCESS_DENIED Macro

С

#define ASSOCIATION_ACCESS_DENIED 0x02

Description

This is macro ASSOCIATION_ACCESS_DENIED.

Symbol Reference > <u>Definitions</u> > <u>ASSOCIATION_ACCESS_DENIED</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

ASSOCIATION_PAN_FULL Macro

С

#define ASSOCIATION_PAN_FULL 0x01

Description

This is macro ASSOCIATION_PAN_FULL.

Symbol Reference > <u>Definitions</u> > <u>ASSOCIATION_PAN_FULL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

ASSOCIATION_SUCCESSFUL Macro

С

#define ASSOCIATION_SUCCESSFUL 0×00

Description

This is macro ASSOCIATION_SUCCESSFUL.

Symbol Reference > <u>Definitions</u> > <u>ASSOCIATION_SUCCESSFUL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

CHANNEL_ASSESSMENT_CARRIER_SENSE Macro

С

#define CHANNEL_ASSESSMENT_CARRIER_SENSE 0x00

Description

This is macro CHANNEL_ASSESSMENT_CARRIER_SENSE.

Symbol Reference > <u>Definitions</u> > <u>CHANNEL_ASSESSMENT_CARRIER_SENSE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

CHANNEL_ASSESSMENT_ENERGY_DETECT Macro

С

#define CHANNEL_ASSESSMENT_ENERGY_DETECT 0x01

Description

This is macro CHANNEL_ASSESSMENT_ENERGY_DETECT.

Symbol Reference > <u>Definitions</u> > <u>CHANNEL_ASSESSMENT_ENERGY_DETECT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CMD_CHANNEL_HOPPING Macro

С

#define CMD_CHANNEL_HOPPING 0x84

Description

This is macro CMD_CHANNEL_HOPPING.

Symbol Reference > <u>Definitions</u> > <u>CMD_CHANNEL_HOPPING Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CMD_DATA_REQUEST Macro

С

#define CMD_DATA_REQUEST 0x83

Description

This is macro CMD_DATA_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>CMD_DATA_REQUEST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CMD_MAC_DATA_REQUEST Macro

С

#define CMD_MAC_DATA_REQUEST 0x04

Description

This is macro CMD_MAC_DATA_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>CMD_MAC_DATA_REQUEST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CMD_P2P_CONNECTION_REMOVAL_REQUEST Macro

С

#define CMD_P2P_CONNECTION_REMOVAL_REQUEST 0x82

Description

This is macro CMD_P2P_CONNECTION_REMOVAL_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>CMD_P2P_CONNECTION_REMOVAL_REQUEST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CMD_P2P_CONNECTION_REMOVAL_RESPONSE Macro

С

#define CMD_P2P_CONNECTION_REMOVAL_RESPONSE 0x92

Description

This is macro CMD_P2P_CONNECTION_REMOVAL_RESPONSE.

Symbol Reference > <u>Definitions</u> > <u>CMD_P2P_CONNECTION_REMOVAL_RESPONSE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

CMD_P2P_CONNECTION_REQUEST Macro

С

#define CMD_P2P_CONNECTION_REQUEST 0x81

Description

This is macro CMD_P2P_CONNECTION_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>CMD_P2P_CONNECTION_REQUEST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

CMD_P2P_CONNECTION_RESPONSE Macro

С

#define CMD_P2P_CONNECTION_RESPONSE 0x91

Description

This is macro CMD_P2P_CONNECTION_RESPONSE.

Symbol Reference > <u>Definitions</u> > <u>CMD_P2P_CONNECTION_RESPONSE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CONN_MODE_DIRECT Macro

С

#define CONN_MODE_DIRECT 0x00

Description

This is macro CONN_MODE_DIRECT.

Symbol Reference > <u>Definitions</u> > <u>CONN_MODE_DIRECT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CONN_MODE_INDIRECT Macro

С

#define CONN_MODE_INDIRECT 0x01

Description

This is macro CONN_MODE_INDIRECT.

Symbol Reference > <u>Definitions</u> > <u>CONN_MODE_INDIRECT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

DISABLE_ALL_CONN Macro

С

#define DISABLE_ALL_CONN 0x03

Description

This is macro DISABLE_ALL_CONN.

Symbol Reference > <u>Definitions</u> > <u>DISABLE_ALL_CONN Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ENABLE_ALL_CONN Macro

С

#define ENABLE_ALL_CONN 0×00

Description

This is macro ENABLE_ALL_CONN.

Symbol Reference > <u>Definitions</u> > <u>ENABLE_ALL_CONN Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

ENABLE_CONSOLE Macro

С

#define ENABLE_CONSOLE

Description

ENABLE_CONSOLE will enable the print out on the hyper terminal this definition is very helpful in the debugging process

Symbol Reference > <u>Definitions</u> > <u>ENABLE_CONSOLE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.
Contents | Index | Home

Previous | Up | Next

ENABLE_PREV_CONN Macro

С

#define ENABLE_PREV_CONN 0x01

Description

This is macro ENABLE_PREV_CONN.

Symbol Reference > <u>Definitions</u> > <u>ENABLE_PREV_CONN Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ERR_INVALID_INPUT Macro

С

#define ERR_INVALID_INPUT 0xFF

Description

This is macro ERR_INVALID_INPUT.

Symbol Reference > <u>Definitions</u> > <u>ERR_INVALID_INPUT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ERR_RX_FAIL Macro

С

#define ERR_RX_FAIL 0x03

Description

This is macro ERR_RX_FAIL.

Symbol Reference > <u>Definitions</u> > <u>ERR_RX_FAIL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ERR_TRX_FAIL Macro

С

#define ERR_TRX_FAIL 0x01

Description

This is macro ERR_TRX_FAIL.

Symbol Reference > <u>Definitions</u> > <u>ERR_TRX_FAIL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ERR_TX_FAIL Macro

С

#define ERR_TX_FAIL 0x02

Description

This is macro ERR_TX_FAIL.

Symbol Reference > <u>Definitions</u> > <u>ERR_TX_FAIL Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

IEEE_802_15_4 Macro

С

#define IEEE_802_15_4

Description

This is macro IEEE_802_15_4.

Symbol Reference > <u>Definitions</u> > <u>IEEE_802_15_4 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help	Contents Index Home	Previous Up Next
KEY_SIZE Macro		
С		
#define KEY_SIZE	8	

Description

This is macro KEY_SIZE.

Symbol Reference > <u>Definitions</u> > <u>KEY_SIZE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.



00 [1:0] 0dB ;default (This is IF Filter gain)

Symbol Reference > <u>Definitions</u> > <u>LNA_GAIN_0_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

LNA_GAIN_N_14_DB Macro

С

#define LNA_GAIN_N_14_DB 0×0010

Description

This is macro LNA_GAIN_N_14_DB.

Symbol Reference > <u>Definitions</u> > <u>LNA_GAIN_N_14_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

LNA_GAIN_N_20_DB Macro

С

#define LNA_GAIN_N_20_DB 0x0018

Description

This is macro LNA_GAIN_N_20_DB.

Symbol Reference > <u>Definitions</u> > <u>LNA_GAIN_N_20_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

LNA_GAIN_N_6_DB Macro

С

#define LNA_GAIN_N_6_DB 0x0008

Description

This is macro LNA_GAIN_N_6_DB.

Symbol Reference > <u>Definitions</u> > <u>LNA_GAIN_N_6_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

NOISE_DETECT_CS Macro

С

#define NOISE_DETECT_CS 0x01

Description

This is macro NOISE_DETECT_CS.

Symbol Reference > <u>Definitions</u> > <u>NOISE_DETECT_CS Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

NOISE_DETECT_ENERGY Macro

С

#define NOISE_DETECT_ENERGY 0×00

Description

This is macro NOISE_DETECT_ENERGY.

Symbol Reference > <u>Definitions</u> > <u>NOISE_DETECT_ENERGY Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

PACKET_TYPE_ACK Macro

С

#define PACKET_TYPE_ACK 0x02

Description

This is macro PACKET_TYPE_ACK.

Symbol Reference > <u>Definitions</u> > <u>PACKET_TYPE_ACK Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

PACKET_TYPE_COMMAND Macro

С

#define PACKET_TYPE_COMMAND 0x01

Description

This is macro PACKET_TYPE_COMMAND.

Symbol Reference > <u>Definitions</u> > <u>PACKET_TYPE_COMMAND Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

PACKET_TYPE_DATA Macro

С

#define PACKET_TYPE_DATA 0x00

Description

This is macro PACKET_TYPE_DATA.

Symbol Reference > <u>Definitions</u> > <u>PACKET_TYPE_DATA Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

PACKET_TYPE_MASK Macro

С

#define PACKET_TYPE_MASK 0x03

Description

This is macro PACKET_TYPE_MASK.

Symbol Reference > <u>Definitions</u> > <u>PACKET_TYPE_MASK Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

PACKET_TYPE_RESERVE Macro

С

#define PACKET_TYPE_RESERVE 0x03

Description

This is macro PACKET_TYPE_RESERVE.

Symbol Reference > <u>Definitions</u> > <u>PACKET_TYPE_RESERVE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

POWER_STATE_DEEP_SLEEP Macro

С

#define POWER_STATE_DEEP_SLEEP 0x00

Description

This is macro POWER_STATE_DEEP_SLEEP.

Symbol Reference > <u>Definitions</u> > <u>POWER_STATE_DEEP_SLEEP</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

POWER_STATE_OPERATE Macro

С

#define POWER_STATE_OPERATE 0xFF

Description

This is macro POWER_STATE_OPERATE.

Symbol Reference > <u>Definitions</u> > <u>POWER_STATE_OPERATE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

POWER_STATE_SLEEP Macro

С

#define POWER_STATE_SLEEP 0x00

Description

This is macro POWER_STATE_SLEEP.

Symbol Reference > <u>Definitions</u> > <u>POWER_STATE_SLEEP Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

POWER_STATE_WAKEUP Macro

С

#define POWER_STATE_WAKEUP 0x01

Description

This is macro POWER_STATE_WAKEUP.

Symbol Reference > <u>Definitions</u> > <u>POWER_STATE_WAKEUP Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

POWER_STATE_WAKEUP_DR Macro

С

#define POWER_STATE_WAKEUP_DR 0x02

Description

This is macro POWER_STATE_WAKEUP_DR.

Symbol Reference > <u>Definitions</u> > <u>POWER_STATE_WAKEUP_DR Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

RSSI_THRESHOLD_103 Macro

С

#define RSSI_THRESHOLD_103 0×0000

Description

This is macro RSSI_THRESHOLD_103.

Symbol Reference > <u>Definitions</u> > <u>RSSI_THRESHOLD_103 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

RSSI_THRESHOLD_73 Macro

С

#define RSSI_THRESHOLD_73 0x0005

Description

This is macro RSSI_THRESHOLD_73.

Symbol Reference > <u>Definitions</u> > <u>RSSI_THRESHOLD_73 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

RSSI_THRESHOLD_79 Macro

С

#define RSSI_THRESHOLD_79 0x0004

Description

This is macro RSSI_THRESHOLD_79.

Symbol Reference > <u>Definitions</u> > <u>RSSI_THRESHOLD_79 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

RSSI_THRESHOLD_85 Macro

С

#define RSSI_THRESHOLD_85 0x0003

Description

This is macro RSSI_THRESHOLD_85.

Symbol Reference > <u>Definitions</u> > <u>RSSI_THRESHOLD_85 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

RSSI_THRESHOLD_91 Macro

С

#define RSSI_THRESHOLD_91 0x0002

Description

This is macro RSSI_THRESHOLD_91.

Symbol Reference > <u>Definitions</u> > <u>RSSI_THRESHOLD_91 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

RSSI_THRESHOLD_97 Macro

С

#define RSSI_THRESHOLD_97 0x0001

Description

This is macro RSSI_THRESHOLD_97.

Symbol Reference > <u>Definitions</u> > <u>RSSI_THRESHOLD_97 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

SECURITY_MASK Macro

С

#define SECURITY_MASK 0x08

Description

This is macro SECURITY_MASK.

Symbol Reference > <u>Definitions</u> > <u>SECURITY_MASK Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

SOFTWARE_CRC Macro

С

#define SOFTWARE_CRC

Description

This is macro SOFTWARE_CRC.

Symbol Reference > <u>Definitions</u> > <u>SOFTWARE_CRC Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

START_CONN_CS_SCN Macro

С

#define START_CONN_CS_SCN 0x02

Description

This is macro START_CONN_CS_SCN.

Symbol Reference > <u>Definitions</u> > <u>START_CONN_CS_SCN Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

START_CONN_DIRECT Macro

С

#define START_CONN_DIRECT 0x00

Description

This is macro START_CONN_DIRECT.

Symbol Reference > <u>Definitions</u> > <u>START_CONN_DIRECT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

START_CONN_ENERGY_SCN Macro

С

#define START_CONN_ENERGY_SCN 0x01

Description

This is macro START_CONN_ENERGY_SCN.

Symbol Reference > <u>Definitions</u> > <u>START_CONN_ENERGY_SCN Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

SUCCESS Macro

С

#define SUCCESS 0×00

Description

This is macro SUCCESS.

Symbol Reference > <u>Definitions</u> > <u>SUCCESS Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TX_POWER_0_DB Macro

С

#define TX_POWER_0_DB 0x0000

Description

This is macro TX_POWER_0_DB.

Symbol Reference > <u>Definitions</u> > <u>TX_POWER_0_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.
Contents | Index | Home

Previous | Up | Next

TX_POWER_N_10_DB Macro

С

#define TX_POWER_N_10_DB 0x0004

Description

This is macro TX_POWER_N_10_DB.

Symbol Reference > <u>Definitions</u> > <u>TX_POWER_N_10_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TX_POWER_N_12_5_DB Macro

С

#define TX_POWER_N_12_5_DB 0x0005

Description

This is macro TX_POWER_N_12_5_DB.

Symbol Reference > <u>Definitions</u> > <u>TX_POWER_N_12_5_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TX_POWER_N_15_DB Macro

С

#define TX_POWER_N_15_DB 0x0006

Description

This is macro TX_POWER_N_15_DB.

Symbol Reference > <u>Definitions</u> > <u>TX_POWER_N_15_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TX_POWER_N_17_5_DB Macro

С

#define TX_POWER_N_17_5_DB 0x0007

Description

This is macro TX_POWER_N_17_5_DB.

Symbol Reference > <u>Definitions</u> > <u>TX_POWER_N_17_5_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TX_POWER_N_2_5_DB Macro

С

#define TX_POWER_N_2_5_DB 0x0001

Description

This is macro TX_POWER_N_2_5_DB.

Symbol Reference > <u>Definitions</u> > <u>TX_POWER_N_2_5_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TX_POWER_N_5_DB Macro

С

#define TX_POWER_N_5_DB 0x0002

Description

This is macro TX_POWER_N_5_DB.

Symbol Reference > <u>Definitions</u> > <u>TX_POWER_N_5_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

TX_POWER_N_7_5_DB Macro

С

#define TX_POWER_N_7_5_DB 0x0003

Description

This is macro TX_POWER_N_7_5_DB.

Symbol Reference > <u>Definitions</u> > <u>TX_POWER_N_7_5_DB Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_10 Macro

С

#define XTAL_LD_CAP_10 0x0003

Description

This is macro XTAL_LD_CAP_10.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_10 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_105 Macro

С

#define XTAL_LD_CAP_105 0x0004

Description

This is macro XTAL_LD_CAP_105.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_105 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_11 Macro

С

#define XTAL_LD_CAP_11 0x0005

Description

This is macro XTAL_LD_CAP_11.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_11 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_115 Macro

С

#define XTAL_LD_CAP_115 0x0006

Description

This is macro XTAL_LD_CAP_115.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_115 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_12 Macro

С

#define XTAL_LD_CAP_12 0x0007

Description

This is macro XTAL_LD_CAP_12.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_12 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_125 Macro

С

#define XTAL_LD_CAP_125 0x0008

Description

This is macro XTAL_LD_CAP_125.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_125 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_13 Macro

С

#define XTAL_LD_CAP_13 0x0009

Description

This is macro XTAL_LD_CAP_13.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_13 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_135 Macro

С

#define XTAL_LD_CAP_135 0x000A

Description

This is macro XTAL_LD_CAP_135.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_135 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_14 Macro

С

#define XTAL_LD_CAP_14 0x000B

Description

This is macro XTAL_LD_CAP_14.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_14 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_145 Macro

С

#define XTAL_LD_CAP_145 0×000C

Description

This is macro XTAL_LD_CAP_145.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_145 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_15 Macro

С

#define XTAL_LD_CAP_15 0x000D

Description

This is macro XTAL_LD_CAP_15.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_15 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_155 Macro

С

#define XTAL_LD_CAP_155 0×000E

Description

This is macro XTAL_LD_CAP_155.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_155 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_16 Macro

С

#define XTAL_LD_CAP_16 0x000F

Description

This is macro XTAL_LD_CAP_16.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_16 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_85 Macro

С

#define XTAL_LD_CAP_85 0x0000

Description

following should be in the def file

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_85 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_9 Macro

С

#define XTAL_LD_CAP_9 0x0001

Description

This is macro XTAL_LD_CAP_9.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_9 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTAL_LD_CAP_95 Macro

С

#define XTAL_LD_CAP_95 0x0002

Description

This is macro XTAL_LD_CAP_95.

Symbol Reference > <u>Definitions</u> > <u>XTAL_LD_CAP_95 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

XTEA_ROUND Macro

С

#define XTEA_ROUND 32

Description

This is macro XTEA_ROUND.

Symbol Reference > <u>Definitions</u> > <u>XTEA_ROUND Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CMD_TIME_SYNCHRONIZATION_NOTIFICATION Macro

С

#define CMD_TIME_SYNCHRONIZATION_NOTIFICATION 0x86

Description

This is macro CMD_TIME_SYNCHRONIZATION_NOTIFICATION.

Symbol Reference > <u>Definitions</u> > <u>CMD_TIME_SYNCHRONIZATION_NOTIFICATION Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CMD_TIME_SYNCHRONIZATION_REQUEST Macro

С

#define CMD_TIME_SYNCHRONIZATION_REQUEST 0x85

Description

This is macro CMD_TIME_SYNCHRONIZATION_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>CMD_TIME_SYNCHRONIZATION_REQUEST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

DATA_REQUEST_ASSOCIATION_RESPONSE Macro

С

#define DATA_REQUEST_ASSOCIATION_RESPONSE 0×00

Description

This is macro DATA_REQUEST_ASSOCIATION_RESPONSE.

Symbol Reference > <u>Definitions</u> > <u>DATA_REQUEST_ASSOCIATION_RESPONSE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

DATA_REQUEST_SHORT_ADDRESSES Macro

С

#define DATA_REQUEST_SHORT_ADDRESSES 0x01

Description

This is macro DATA_REQUEST_SHORT_ADDRESSES.

Symbol Reference > <u>Definitions</u> > <u>DATA_REQUEST_SHORT_ADDRESSES Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 EUI_1 Macro
 C
 #define EUI_1 0x77

Description

This is macro EUI_1.

Symbol Reference > <u>Definitions</u> > <u>EUI_1 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Description

This is macro EUI_2.

Symbol Reference > <u>Definitions</u> > <u>EUI_2 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 EUI_3 Macro
 C
 #define EUI_3 0x55

Description

This is macro EUI_3.

Symbol Reference > <u>Definitions</u> > <u>EUI_3 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 EUI_4 Macro
 C
 #define EUI_4 0x44

Description

This is macro EUI_4.

Symbol Reference > <u>Definitions</u> > <u>EUI_4 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

 MiWi(TM) Development Environment Help
 Contents | Index | Home
 Previous | Up | Next

 EUI_5 Macro
 C
 #define EUI_5 0x33

Description

This is macro EUI_5.

Symbol Reference > <u>Definitions</u> > <u>EUI_5 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help	Contents Index Home	Previous Up Next
EUI_6 Macro		
С		
#define EUI_6 0×	22	

Description

This is macro EUI_6.

Symbol Reference > <u>Definitions</u> > <u>EUI_6 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MiWi(TM) Development Environment Help	Contents Index Home	Previous Up Next
EUI_7 Macro		
С		
#define EUI_7 0×	(11	

Description

EUI_x defines the xth byte of permanent address for the wireless node

Symbol Reference > <u>Definitions</u> > <u>EUI_7 Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

FRAME_TYPE_ACK Macro

С

#define FRAME_TYPE_ACK 0x02

Description

This is macro FRAME_TYPE_ACK.

Symbol Reference > <u>Definitions</u> > <u>FRAME_TYPE_ACK Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.
Contents | Index | Home

Previous | Up | Next

FRAME_TYPE_BEACON Macro

С

#define FRAME_TYPE_BEACON 0×00

Description

DEFINITIONS

Symbol Reference > <u>Definitions</u> > <u>FRAME_TYPE_BEACON Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

FRAME_TYPE_COMMAND Macro

С

#define FRAME_TYPE_COMMAND 0x03

Description

This is macro FRAME_TYPE_COMMAND.

Symbol Reference > <u>Definitions</u> > <u>FRAME_TYPE_COMMAND Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

FRAME_TYPE_DATA Macro

С

#define FRAME_TYPE_DATA 0x01

Description

This is macro FRAME_TYPE_DATA.

Symbol Reference > <u>Definitions</u> > <u>FRAME_TYPE_DATA Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.



915MHz

Symbol Reference > <u>Definitions</u> > <u>FREQ_BAND Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

MAC_COMMAND_ASSOCIATION_REQUEST Macro

С

#define MAC_COMMAND_ASSOCIATION_REQUEST 0x01

Description

This is macro MAC_COMMAND_ASSOCIATION_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>MAC_COMMAND_ASSOCIATION_REQUEST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

MAC_COMMAND_ASSOCIATION_RESPONSE Macro

С

#define MAC_COMMAND_ASSOCIATION_RESPONSE 0x02

Description

This is macro MAC_COMMAND_ASSOCIATION_RESPONSE.

Symbol Reference > <u>Definitions</u> > <u>MAC_COMMAND_ASSOCIATION_RESPONSE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

MAC_COMMAND_BEACON_REQUEST Macro

С

#define MAC_COMMAND_BEACON_REQUEST 0×07

Description

This is macro MAC_COMMAND_BEACON_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>MAC_COMMAND_BEACON_REQUEST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

MAC_COMMAND_COORDINATOR_REALIGNMEN Macro

С

#define MAC_COMMAND_COORDINATOR_REALIGNMENT 0x08

Description

This is macro MAC_COMMAND_COORDINATOR_REALIGNMENT.

Symbol Reference > <u>Definitions</u> > <u>MAC_COMMAND_COORDINATOR_REALIGNMENT Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

MAC_COMMAND_DATA_REQUEST Macro

С

#define MAC_COMMAND_DATA_REQUEST 0x04

Description

This is macro MAC_COMMAND_DATA_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>MAC_COMMAND_DATA_REQUEST</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

MAC_COMMAND_DISASSOCIATION_NOTIFICATI Macro

С

#define MAC_COMMAND_DISASSOCIATION_NOTIFICATION 0×03

Description

This is macro MAC_COMMAND_DISASSOCIATION_NOTIFICATION.

Symbol Reference > <u>Definitions</u> > <u>MAC_COMMAND_DISASSOCIATION_NOTIFICATION Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

MAC_COMMAND_ORPHAN_NOTIFICATION Macro

С

#define MAC_COMMAND_ORPHAN_NOTIFICATION 0×06

Description

This is macro MAC_COMMAND_ORPHAN_NOTIFICATION.

Symbol Reference > <u>Definitions</u> > <u>MAC_COMMAND_ORPHAN_NOTIFICATION Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright 0 2009 Microchip Technology, Inc. All rights reserved.

MAC_COMMAND_PAN_ID_CONFLICT_NOTIFICA[®] Macro

С

#define MAC_COMMAND_PAN_ID_CONFLICT_NOTIFICATION $0 \times 0!$

Description

This is macro MAC_COMMAND_PAN_ID_CONFLICT_NOTIFICATION.

Symbol Reference > <u>Definitions</u> > <u>MAC_COMMAND_PAN_ID_CONFLICT_NOTIFICATION Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

STATUS_ACTIVE_SCAN Macro

С

#define STATUS_ACTIVE_SCAN 0x02

Description

This is macro STATUS_ACTIVE_SCAN.

Symbol Reference > <u>Definitions</u> > <u>STATUS_ACTIVE_SCAN Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

STATUS_ENTRY_NOT_EXIST Macro

С

#define STATUS_ENTRY_NOT_EXIST 0xF0

Description

This is macro STATUS_ENTRY_NOT_EXIST.

Symbol Reference > <u>Definitions</u> > <u>STATUS_ENTRY_NOT_EXIST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

STATUS_EXISTS Macro

С

#define STATUS_EXISTS 0x01

Description

This is macro STATUS_EXISTS.

Symbol Reference > <u>Definitions</u> > <u>STATUS_EXISTS Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

STATUS_NOT_ENOUGH_SPACE Macro

С

#define STATUS_NOT_ENOUGH_SPACE 0xF1

Description

This is macro STATUS_NOT_ENOUGH_SPACE.

Symbol Reference > <u>Definitions</u> > <u>STATUS_NOT_ENOUGH_SPACE</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

STATUS_NOT_SAME_PAN Macro

С

#define STATUS_NOT_SAME_PAN 0xF2

Description

This is macro STATUS_NOT_SAME_PAN.

Symbol Reference > <u>Definitions</u> > <u>STATUS_NOT_SAME_PAN Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

STATUS_SUCCESS Macro

С

#define STATUS_SUCCESS 0x00

Description

This is macro STATUS_SUCCESS.

Symbol Reference > <u>Definitions</u> > <u>STATUS_SUCCESS Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ACK_REPORT_TYPE Macro

С

#define ACK_REPORT_TYPE 0x30

Description

This is macro ACK_REPORT_TYPE.

Symbol Reference > <u>Definitions</u> > <u>ACK_REPORT_TYPE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ROLE_COORDINATOR Macro

С

#define ROLE_COORDINATOR 0x01

Description

This is macro ROLE_COORDINATOR.

Symbol Reference > <u>Definitions</u> > <u>ROLE_COORDINATOR Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ROLE_FFD_END_DEVICE Macro

С

#define ROLE_FFD_END_DEVICE 0x00

Description

This is macro ROLE_FFD_END_DEVICE.

Symbol Reference > <u>Definitions</u> > <u>ROLE_FFD_END_DEVICE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

ROLE_PAN_COORDINATOR Macro

С

#define ROLE_PAN_COORDINATOR 0x02

Description

This is macro ROLE_PAN_COORDINATOR.

Symbol Reference > <u>Definitions</u> > <u>ROLE_PAN_COORDINATOR Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

OPEN_SOCKET_REQUEST Macro

С

#define OPEN_SOCKET_REQUEST 0x10

Description

This is macro OPEN_SOCKET_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>OPEN_SOCKET_REQUEST Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

OPEN_SOCKET_RESPONSE Macro

С

#define OPEN_SOCKET_RESPONSE 0x11

Description

This is macro OPEN_SOCKET_RESPONSE.

Symbol Reference > <u>Definitions</u> > <u>OPEN_SOCKET_RESPONSE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

CHANNEL_HOPPING_REQUEST Macro

С

#define CHANNEL_HOPPING_REQUEST 0x40

Description

This is macro CHANNEL_HOPPING_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>CHANNEL_HOPPING_REQUEST</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

EUI_ADDRESS_SEARCH_REQUEST Macro

С

#define EUI_ADDRESS_SEARCH_REQUEST 0x20

Description

This is macro EUI_ADDRESS_SEARCH_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>EUI_ADDRESS_SEARCH_REQUEST</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

EUI_ADDRESS_SEARCH_RESPONSE Macro

С

#define EUI_ADDRESS_SEARCH_RESPONSE 0x21

Description

This is macro EUI_ADDRESS_SEARCH_RESPONSE.

Symbol Reference > <u>Definitions</u> > <u>EUI_ADDRESS_SEARCH_RESPONSE Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Previous | Up | Next

MIWI_ACK_REQ Macro

С

#define MIWI_ACK_REQ 0x04

Description

This is macro MIWI_ACK_REQ.

Symbol Reference > <u>Definitions</u> > <u>MIWI_ACK_REQ Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up | Next

RESYNCHRONIZATION_REQUEST Macro

С

#define RESYNCHRONIZATION_REQUEST 0x41

Description

This is macro RESYNCHRONIZATION_REQUEST.

Symbol Reference > <u>Definitions</u> > <u>RESYNCHRONIZATION_REQUEST</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Previous | Up

RESYNCHRONIZATION_RESPONSE Macro

С

#define RESYNCHRONIZATION_RESPONSE 0x42

Description

This is macro RESYNCHRONIZATION_RESPONSE.

Symbol Reference > <u>Definitions</u> > <u>RESYNCHRONIZATION_RESPONSE</u> <u>Macro</u>

Microchip My Application xx.yy - [Jan 1, 2009] Copyright @ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

Contents

Introduction

SW License Agreement

Release Notes

New Features

Network Freezer Enhanced Data Request

Time Synchronization

Demos

Required Hardware

Hardware Sets

Configuring the Hardware

PICDEM Z

PIC18 Explorer

Explorer 16

8-bit Wireless Development Kit

Firmware

Precompiled HEX Files

Demo Source Code Project for MPLAB 8.x MiWi P2P

PICDEM Z Demo Board for MiWi P2P PIC18 Explorer Demo Board for MiWi P2P 8-bit Wireless Development Kit for MiWi P2P

Explorer 16 Demo Board for MiWi P2P

PIC24 or dsPIC33 for MiWi P2P PIC32 for MiWi P2P

MiWi Mesh

PICDEM Z Demo Board for MiWi PIC18 Explorer Demo Board for MiWi 8-bit Wireless Development Kit for MiWi

Explorer 16 Demo Board for MiWi

PIC24 or dsPIC33 for MiWi

PIC32 for MiWi

MiWi PRO

PICDEM Z Demo Board for MiWi PRO PIC18 Explorer Demo Board for MiWi PRO 8-bit Wireless Development Kit for MiWi PRO

Explorer 16 Demo Board for MiWi PRO

PIC24 or dsPIC33 for MiWi PRO PIC32 for MiWi PRO

PIC32 IUI MIWI PRO

Demo Source Code Project for MPLAB X

Running Demos

Basic Demos

Simple Example

Feature Demo

MiWi PRO Test Interface

8 bit Wireless Development Kit Demos

Configuring the Library

Application

ADDITIONAL_NODE_ID_SIZE Macro CONNECTION_SIZE Macro ENABLE_ACTIVE_SCAN Macro ENABLE_BROADCAST Macro ENABLE_ED_SCAN Macro ENABLE_FREQUENCY_AGILITY Macro ENABLE_HAND_SHAKE Macro ENABLE_PA_LNA Macro ENABLE_INDIRECT_MESSAGE Macro ENABLE_NETWORK_FREEZER Macro ENABLE_SECURITY Macro ENABLE_SLEEP Macro EUI_0 Macro HARDWARE_SPI Macro MRF24J40 Macro MRF49XA Macro MRF89XA Macro MY_ADDRESS_LENGTH Macro MY_ADDRESS_LENGTH Macro MY_PAN_ID Macro NWK_ROLE_COORDINATOR Macro NWK_ROLE_END_DEVICE Macro NWK_ROLE_END_DEVICE Macro PROTOCOL_MIWI_PRO Macro PROTOCOL_MIWI_PRO Macro PROTOCOL_P2P Macro RFD_WAKEUP_INTERVAL Macro RX_BUFFER_SIZE Macro TARGET_SMALL Macro TX_BUFFER_SIZE Macro

Wireless Protocol

ACTIVE SCAN_RESULT_SIZE Macro CONNECTION RETRY TIMES Macro COUNTER CRYSTAL FREQ Macro ENABLE DUMP Macro ENABLE ENHANCED DATA REQUEST Macro ENABLE TIME SYNC Macro FA BROADCAST TIME Macro INDIRECT MESSAGE SIZE Macro INDIRECT MESSAGE TIMEOUT Macro **RESYNC TIMES Macro RFD DATA WAIT Macro** TIME SYNC SLOTS Macro MiWi(TM) P2P Communication Protocol **CONNECTION INTERVAL Macro** MiWi and MiWi PRO Networking Protocols INDIRECT MESSAGE TIMEOUT CYCLE Macro MAX ROUTING FAILURE Macro

OPEN SOCKET POLL INTERVAL Macro **OPEN SOCKET TIMEOUT Macro** MiWi Mesh Networking Protocol BROADCAST RECORD_SIZE Macro BROADCAST RECORD TIMEOUT Macro MIWI ACK TIMEOUT Macro MiWi PRO Networking Protocol COMM INTERVAL Macro COMM RSSI THRESHOLD Macro ENABLE MIWI PRO ACKNOWLEDGEMENT Macro ENABLE ROUTING UPDATE Macro ENABLE_BROADCAST_TO_SLEEP_DEVICE Macro FA COMM INTERVAL Macro FA MAX NOISE THRESHOLD Macro FA WAIT TIMEOUT Macro FAMILY TREE BROADCAST Macro MIWI PRO ACK TIMEOUT Macro NUM COORDINATOR Macro PACKET RECORD SIZE Macro PACKET RECORD TIMEOUT Macro RANDOM DELAY RANGE Macro ROUTING UPDATE INTERVAL Macro ROUTING UPDATE EXPIRATION Macro ROUTING TABLE BROADCAST Macro

RF Transceivers

BANK_SIZE Macro KEY_SEQUENCE_NUMBER Macro SECURITY_LEVEL Macro SECURITY_KEY_00 Macro

MRF24J40 IEEE 802.15.4 Compliant 2.4GHz Transceiver

TURBO_MODE Macro

VERIFY_TRANSMIT Macro

SubGHz Transceivers

MRF49XA SubGHz Transceiver

BAND 915 Macro **CRYSTAL PPM Macro** DATA RATE 9600 Macro INFER DEST ADDRESS Macro MAX ALLOWED TX FAILURE Macro **RSSI THRESHOLD Macro** XTAL LD CAP Macro MRF89XA SubGHz Transceiver BAND 902 Macro DATA RATE 20 Macro ACK INFO SIZE Macro CCA RETRIES Macro CCA THRESHOLD Macro CCA TIMES Macro ENABLE ACK Macro ENABLE CCA Macro ENABLE RETRANSMISSION Macro FRAME COUNTER UPDATE INTERVAL Macro LNA GAIN Macro RETRANSMISSION TIMES Macro SOURCE ADDRESS ABSENT Macro **TX POWER Macro**

Library API

MiApp Interfaces

MiApp_BroadcastPacket Function

MiApp_ConnectionMode Function

MiApp_DiscardMessage Function

MiApp_EstablishConnection Function

MiApp_FlushTx Macro

MiApp_InitChannelHopping Function

MiApp_MessageAvailable Function

MiApp_NoiseDetection Function

MiApp_RemoveConnection Function

MiApp_ResyncConnection Function

MiApp_SearchConnection Function

MiApp_SetChannel Function

MiApp_StartConnection Function

MiApp_TransceiverPowerState Function

MiApp_UnicastAddress Function

MiApp_UnicastConnection Function

MiApp_WriteData Macro

Call Back Functions

MiApp_CB_AllowConnection Macro

MiApp_CB_RFDAcknowledgement Macro

MiMAC Interfaces

MiMAC_ChannelAssessment Function

MiMAC_DiscardPacket Function

MiMAC_Init Function

MiMAC_PowerState Function

MiMAC_ReceivedPacket Function

MiMAC_SendPacket Function

MiMAC_SetAltAddress Function

MiMAC_SetChannel Function

MiMAC_SetPower Function

Symbol Reference

Types

BOOL Type CHAR Type SHORT Type BYTE Type LONG Type WORD Type WORD_VAL Union DWORD Type DWORD VAL Union
CONNECTION_ENTRY Structure RECEIVED_MESSAGE Structure MAC_RECEIVED_PACKET Structure MAC_TRANS_PARAM Structure

Structs, Records, Enums

ACTIVE_SCAN_RESULT Structure CONNECTION_STATUS Union BYTE_BITS Union WORD BITS Union

Variables

ActiveScanResults Variable AdditionalNodeID Variable ConnectionTable Variable rxMessage Variable TxBuffer Variable myLongAddress Variable mySecurityKey Variable

Definitions

ASSOCIATION_ACCESS_DENIED Macro ASSOCIATION_PAN_FULL Macro ASSOCIATION_SUCCESSFUL Macro CHANNEL_ASSESSMENT_CARRIER_SENSE Macro CHANNEL_ASSESSMENT_ENERGY_DETECT Macro CMD_CHANNEL_HOPPING Macro CMD_DATA_REQUEST Macro CMD_MAC_DATA_REQUEST Macro CMD_P2P_CONNECTION_REMOVAL_REQUEST Macro CMD_P2P_CONNECTION_REMOVAL_RESPONSE Macro CMD_P2P_CONNECTION_REQUEST Macro CMD_P2P_CONNECTION_REQUEST Macro CMD_P2P_CONNECTION_RESPONSE Macro CMD_P2P_CONNECTION_RESPONSE Macro CONN_MODE_DIRECT Macro DISABLE_ALL_CONN Macro

ENABLE ALL CONN Macro ENABLE CONSOLE Macro ENABLE PREV CONN Macro ERR INVALID INPUT Macro ERR RX FAIL Macro ERR TRX FAIL Macro ERR TX FAIL Macro IEEE 802 15 4 Macro **KEY SIZE Macro** LNA GAIN 0 DB Macro LNA GAIN N 14 DB Macro LNA GAIN N 20 DB Macro LNA GAIN N 6 DB Macro NOISE DETECT CS Macro NOISE DETECT ENERGY Macro PACKET TYPE ACK Macro PACKET TYPE COMMAND Macro PACKET TYPE DATA Macro PACKET TYPE MASK Macro PACKET TYPE RESERVE Macro POWER STATE DEEP SLEEP Macro POWER STATE OPERATE Macro POWER STATE SLEEP Macro POWER STATE WAKEUP Macro POWER STATE WAKEUP DR Macro **RSSI THRESHOLD 103 Macro** RSSI THRESHOLD 73 Macro **RSSI THRESHOLD 79 Macro** RSSI THRESHOLD 85 Macro **RSSI THRESHOLD 91 Macro RSSI THRESHOLD 97 Macro** SECURITY MASK Macro SOFTWARE CRC Macro

START CONN CS SCN Macro START CONN DIRECT Macro START CONN ENERGY SCN Macro SUCCESS Macro TX POWER 0 DB Macro TX POWER N 10 DB Macro TX POWER N 12 5 DB Macro TX POWER N 15 DB Macro TX POWER N 17 5 DB Macro TX_POWER_N_2_5_DB Macro TX POWER N 5 DB Macro TX POWER N 7 5 DB Macro XTAL LD CAP 10 Macro XTAL LD CAP 105 Macro XTAL LD CAP 11 Macro XTAL LD CAP 115 Macro XTAL LD CAP 12 Macro XTAL LD CAP 125 Macro XTAL LD CAP 13 Macro XTAL LD CAP 135 Macro XTAL LD CAP 14 Macro XTAL LD CAP 145 Macro XTAL_LD_CAP_15 Macro XTAL LD CAP 155 Macro XTAL LD CAP 16 Macro XTAL LD CAP 85 Macro XTAL LD CAP 9 Macro XTAL LD CAP 95 Macro XTEA ROUND Macro CMD TIME SYNCHRONIZATION NOTIFICATION Macro CMD TIME SYNCHRONIZATION REQUEST Macro DATA REQUEST ASSOCIATION RESPONSE Macro DATA REQUEST SHORT ADDRESSES Macro

EUI 1 Macro EUI 2 Macro EUI 3 Macro EUI 4 Macro EUI 5 Macro EUI 6 Macro EUI 7 Macro FRAME TYPE ACK Macro FRAME TYPE BEACON Macro FRAME TYPE COMMAND Macro FRAME TYPE DATA Macro FREQ BAND Macro MAC COMMAND ASSOCIATION REQUEST Macro MAC COMMAND ASSOCIATION RESPONSE Macro MAC COMMAND BEACON REQUEST Macro MAC COMMAND COORDINATOR REALIGNMENT Macro MAC COMMAND DATA REQUEST Macro MAC COMMAND DISASSOCIATION NOTIFICATION Macro MAC COMMAND ORPHAN NOTIFICATION Macro MAC COMMAND PAN ID CONFLICT NOTIFICATION Macro STATUS ACTIVE SCAN Macro STATUS_ENTRY_NOT_EXIST Macro STATUS EXISTS Macro STATUS NOT ENOUGH SPACE Macro STATUS NOT SAME PAN Macro STATUS SUCCESS Macro ACK REPORT TYPE Macro ROLE COORDINATOR Macro ROLE FFD END DEVICE Macro ROLE PAN COORDINATOR Macro **OPEN SOCKET REQUEST Macro OPEN SOCKET RESPONSE Macro** CHANNEL HOPPING REQUEST Macro

EUI_ADDRESS_SEARCH_REQUEST Macro EUI_ADDRESS_SEARCH_RESPONSE Macro MIWI_ACK_REQ Macro RESYNCHRONIZATION_REQUEST Macro RESYNCHRONIZATION_RESPONSE Macro

Microchip My Application xx.yy - [Jan 1, 2009] Copyright $\ensuremath{\mathbb{C}}$ 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home

MiWi(TM) Development Environment Help

Contents | Index | Home

Index

 $\frac{18|A|B|C|D|E|F|H|!|K|L|M|N|O|P|R|S|T|V|W|}{X}$

<u>_CONNECTION_ENTRY structure</u> <u>_CONNECTION_STATUS union</u>

8

8 bit Wireless Development Kit Demos 8-bit Wireless Development Kit 8-bit Wireless Development Kit for MiWi 8-bit Wireless Development Kit for MiWi P2P 8-bit Wireless Development Kit for MiWi PRO

Α

ACK_INFO_SIZE macro ACK_REPORT_TYPE macro ACTIVE_SCAN_RESULT structure ACTIVE_SCAN_RESULT_SIZE macro ActiveScanResults variable ADDITIONAL_NODE_ID_SIZE macro AdditionalNodeID variable Application ASSOCIATION_ACCESS_DENIED macro ASSOCIATION_PAN_FULL macro ASSOCIATION_SUCCESSFUL macro

MiApp SetC MiApp_Start MiApp Tran function MiApp Unic MiApp Unic MiApp Write MiMAC Inter MiMAC Cha MiMAC Disc MiMAC Init MiMAC Pov MiMAC Rec MiMAC Ser MiMAC Set MiMAC Set MiMAC Set MiWi and Mi Protocols MiWi Mesh MiWi Mesh I MiWi P2P MiWi PRO MiWi PRO N MiWi PRO T MiWi(TM) P Protocol MIWI ACK

В

BAND_902 macro BAND_915 macro

BANK_SIZE macro	<u>MIWI_ACK_</u>
Basic Demos	<u>MIWI_PRO_</u>
BOOL type	<u>MRF24J40 I</u>
BROADCAST_RECORD_SIZE macro	<u>2.4GHz Trar</u>
BROADCAST_RECORD_TIMEOUT macro	<u>MRF24J40 r</u>
BYTE type	<u>MRF49XA m</u>
BYTE_BITS union	<u>MRF49XA S</u>
C	<u>MRF89XA n</u>
	<u>MRF89XA S</u>
Call Back Functions	MY_ADDRE
CCA_RETRIES macro	<u>MY_PAN_IC</u>
CCA_THRESHOLD macro	<u>myLongAdd</u>
<u>CCA_TIMES macro</u>	<u>mySecurityk</u>
CHANNEL_ASSESSMENT_CARRIER_SENSE macro	N
CHANNEL_ASSESSMENT_ENERGY_DETECT macro	
CHANNEL_HOPPING_REQUEST macro	Network Fre
CHAR type	New Feature
CMD_CHANNEL_HOPPING macro	NOISE_DE1
CMD_DATA_REQUEST macro	NOISE_DE1
CMD_MAC_DATA_REQUEST macro	NUM_COOF
CMD_P2P_CONNECTION_REMOVAL_REQUEST	<u>NWK_ROLE</u>
macro	<u>NWK_ROLE</u>
CMD_P2P_CONNECTION_REMOVAL_RESPONSE	0
CMD P2P CONNECTION REQUEST macro	OPEN SOC
CMD P2P CONNECTION RESPONSE macro	macro
CMD TIME SYNCHRONIZATION NOTIFICATION	OPEN SOC
macro	OPEN SOC
CMD_TIME_SYNCHRONIZATION_REQUEST macro	OPEN_SOC
COMM_INTERVAL macro	D
COMM_RSSI_THRESHOLD macro	Ρ
Configuring the Hardware	PACKET_RI
Configuring the Library	PACKET_RI
	PACKET_T

CONN_MODE_DIRECT macro CONN_MODE_INDIRECT macro CONNECTION_ENTRY structure CONNECTION_INTERVAL macro CONNECTION_RETRY_TIMES macro CONNECTION_SIZE macro CONNECTION_SIZE macro CONNECTION_STATUS union ConnectionTable variable COUNTER_CRYSTAL_FREQ macro CRYSTAL_PPM macro

D

DATA RATE 20 macro PIC24 or dsl DATA RATE 9600 macro PIC32 for Mi PIC32 for Mi DATA REQUEST ASSOCIATION RESPONSE macro DATA REQUEST SHORT ADDRESSES macro PIC32 for Mi PICDEM Z Definitions PICDEM Z [Demo Source Code Project for MPLAB 8.x PICDEM Z [Demo Source Code Project for MPLAB X PICDEM Z [Demos **DISABLE ALL CONN macro** POWER ST <u>macro</u> DWORD type POWER ST **DWORD VAL union** POWER ST

PACKET T

PACKET T

PACKET T

PACKET T

PIC18 Explo

PIC18 Explo PIC18 Explo

PIC18 Explo

PIC24 or dsl

PIC24 or dsl

POWER ST

P2P

PRO

Ε

POWER ST **ENABLE ACK macro** macro ENABLE ACTIVE SCAN macro **Precompiled** ENABLE ALL CONN macro PROTOCOL ENABLE BROADCAST macro PROTOCOL ENABLE BROADCAST TO SLEEP DEVICE macro PROTOCOL ENABLE CCA macro **ENABLE CONSOLE macro** R ENABLE DUMP macro RANDOM E

ENABLE_ED_SCAN macro	RECEIVED
ENABLE_ENHANCED_DATA_REQUEST macro	Release Not
ENABLE_FREQUENCY_AGILITY macro	Required Ha
ENABLE_HAND_SHAKE macro	RESYNC_T
ENABLE_INDIRECT_MESSAGE macro	RESYNCHF
ENABLE_MIWI_PRO_ACKNOWLEDGEMENT macro	<u>macro</u>
ENABLE_NETWORK_FREEZER macro	<u>RESYNCHF</u>
ENABLE_PA_LNA macro	macro
ENABLE PREV CONN macro	<u>RETRANSN</u>
ENABLE RETRANSMISSION macro	<u>RF Transcei</u>
ENABLE ROUTING UPDATE macro	<u>RFD_DATA_</u>
ENABLE SECURITY macro	<u>RFD_WAKE</u>
ENABLE SLEEP macro	ROLE_COC
ENABLE TIME SYNC macro	ROLE_FFD
Enhanced Data Request	<u>ROLE_PAN</u>
ERR INVALID INPUT macro	<u>ROUTING_</u>
ERR RX FAIL macro	macro
ERR TRX FAIL macro	<u>ROUTING_</u> l
ERR TX FAIL macro	macro
EUI 0 macro	<u>ROUTING_</u>
EUI 1 macro	macro
EUI 2 macro	RSSI_THRE
EUI 3 macro	RSSI_THRE
EUL 4 macro	RSSI_THRE
EUL 5 macro	RSSI_THRE
EUL 6 macro	<u>RSSI_THRE</u>
EUL 7 macro	RSSI_THRE
EUL ADDRESS SEARCH REQUEST macro	<u>RSSI_THRE</u>
EUL ADDRESS SEARCH RESPONSE macro	Running Dei
Explorer 16	RX_BUFFEI
Explorer 16 Demo Board for MiWi	<u>rxMessage \</u>
Explorer 16 Demo Board for MiWi P2P	S
Explorer 16 Demo Board for MiWi PRO	
	<u>SECURITY</u>

п

F	<u>SECURITY</u>
F FA_BROADCAST_TIME macro FA_COMM_INTERVAL macro FA_MAX_NOISE_THRESHOLD macro FA_WAIT_TIMEOUT macro FAMILY_TREE_BROADCAST macro Feature Demo Firmware FRAME_COUNTER_UPDATE_INTERVAL macro FRAME_TYPE_ACK macro FRAME_TYPE_BEACON macro FRAME_TYPE_DATA macro FRAME_TYPE_DATA macro FREQ_BAND macro	SECURITY SHORT typ Simple Exa SOFTWAR SOURCE_A macro START_CC START_CC START_CC START_CC Macro STATUS_A STATUS_E STATUS_E STATUS_N
	macro

Η

Hardware Sets HARDWARE SPI macro

I

IEEE 802 15 4 macro **INDIRECT MESSAGE SIZE macro INDIRECT MESSAGE TIMEOUT macro** INDIRECT MESSAGE TIMEOUT CYCLE macro **INFER DEST ADDRESS macro** Introduction

Κ

KEY SEQUENCE NUMBER macro KEY SIZE macro

Library API

e I E <u>4</u>)|)[)| C Ν <u>></u> (<u>illaciu</u> STATUS NC STATUS SL Structs, Rec SubGHz Tra SUCCESS r SW License

Т

TARGET SI **Time Synchi** TIME_SYNC TURBO MC TX BUFFEF TX POWER TX POWER TX POWER TX POWER

LNA_GAIN macro	TX_POWER TX_POWER
INA GAIN N 20 DB macro	
INA GAIN N 6 DB macro	TX POWER
LONG type	TxBuffer var
	Types
Μ	
MAC_COMMAND_ASSOCIATION_REQUEST macro	V
MAC_COMMAND_ASSOCIATION_RESPONSE macro	<u>Variables</u>
MAC_COMMAND_BEACON_REQUEST macro	<u>VERIFY_TR</u>
MAC_COMMAND_COORDINATOR_REALIGNMENT	10/
macro	vv
MAC_COMMAND_DATA_REQUEST macro	<u>Wireless Prc</u>
MAC_COMMAND_DISASSOCIATION_NOTIFICATION	WORD type
macro	WORD_BIT:
MAC_COMMAND_ORPHAN_NOTIFICATION macro	WORD_VAL
MAC_COMMAND_PAN_ID_CONFLICT_NOTIFICATION	Y
macro	^
MAC_RECEIVED_PACKET structure	<u>XTAL_LD_C</u>
MAC_TRANS_PARAM structure	VTAL ID C
	<u>XIAL_LD_C</u>
MAX_ALLOWED_TX_FAILURE macro	XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro	XTAL_LD_C XTAL_LD_C XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro MiApp Interfaces	XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro MiApp Interfaces MiApp_BroadcastPacket function	XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro MiApp Interfaces MiApp_BroadcastPacket function MiApp_CB_AllowConnection macro	XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro MiApp Interfaces MiApp_BroadcastPacket function MiApp_CB_AllowConnection macro MiApp_CB_RFDAcknowledgement macro	XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro MiApp Interfaces MiApp_BroadcastPacket function MiApp_CB_AllowConnection macro MiApp_CB_RFDAcknowledgement macro MiApp_ConnectionMode function	XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro MiApp Interfaces MiApp_BroadcastPacket function MiApp_CB_AllowConnection macro MiApp_CB_RFDAcknowledgement macro MiApp_ConnectionMode function MiApp_DiscardMessage function	XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro MiApp Interfaces MiApp_BroadcastPacket function MiApp_CB_AllowConnection macro MiApp_CB_RFDAcknowledgement macro MiApp_ConnectionMode function MiApp_DiscardMessage function MiApp_EstablishConnection function	XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro MiApp Interfaces MiApp_BroadcastPacket function MiApp_CB_AllowConnection macro MiApp_CB_RFDAcknowledgement macro MiApp_ConnectionMode function MiApp_DiscardMessage function MiApp_EstablishConnection function MiApp_FlushTx macro	XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C
MAX_ALLOWED_TX_FAILURE macro MAX_ROUTING_FAILURE macro MiApp Interfaces MiApp_BroadcastPacket function MiApp_CB_AllowConnection macro MiApp_CB_RFDAcknowledgement macro MiApp_CB_RFDAcknowledgement macro MiApp_DiscardMessage function MiApp_DiscardMessage function MiApp_EstablishConnection function MiApp_FlushTx macro MiApp_InitChannelHopping function	XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C XTAL_LD_C

MiApp_NoiseDetection function MiApp_RemoveConnection function MiApp_ResyncConnection function MiApp_SearchConnection function XTAL_LD_C XTAL_LD_C XTAL_LD_C XTEA_ROU

Microchip My Application xx.yy - [Jan 1, 2009] Copyright © 2009 Microchip Technology, Inc. All rights reserved.

Contents | Index | Home