18051 WNW

Introduction to Wireless Networking (MiWi™ Protocol I)

Class Objectives

- Describe the physical layer options available in MiWi™ networks
- Describe the major services provided by the MiMAC layer to higher layers
- Be able to modify/debug a simple wireless application with the MiWi protocol Development Environment

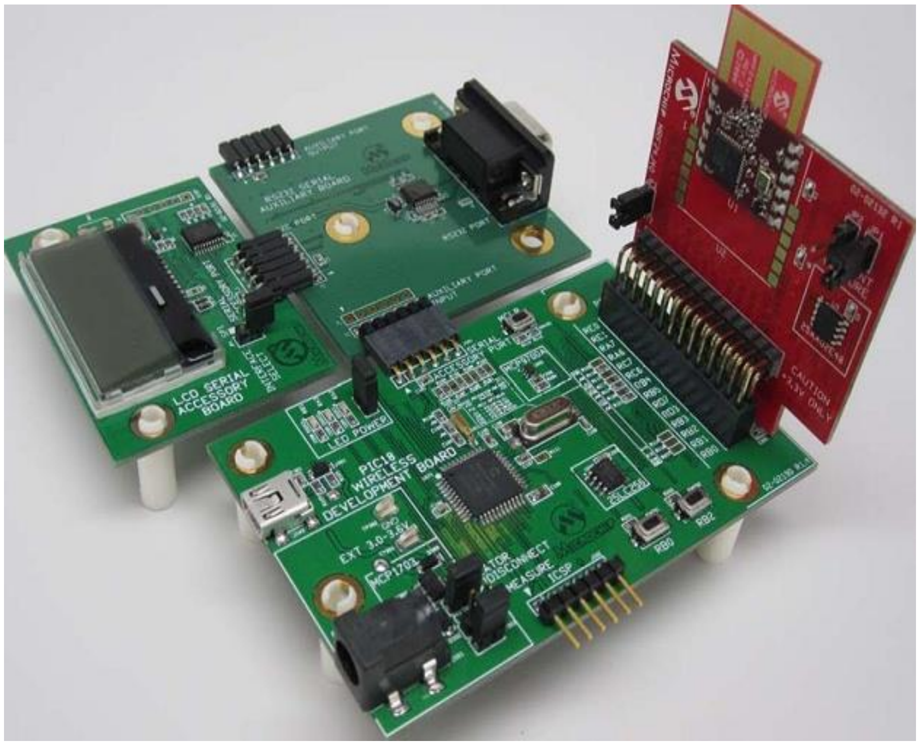
Agenda

- **Introduction to MiWi™ DE**
- **Microchip Wireless MAC (MiMAC)**
  - Lab 1 – Configure the Transceiver
  - Lab 2 – Configure Security
- **Overview of Microchip Wireless Protocols and Application Layer (MiApp)**
- **Overview of MiWi™ PRO advanced functionality**
- **Summary and References**

Class Folders

```
C:\Masters\18051
    \Labs
    \Literature
    \Presentations
    \Tools
```

Development Platform



- **8-Bit Wireless Development Kit (DM182015-1)**
  - Provides 2-RF Nodes
    - **PIC18F46J50 main board**
    - **MRF24J40 2.4GHz Radio PICtail™ Board**
    - **LCD Display board**
    - **RS-232 board**
- **ZENA™ Wireless Adapter (2.4GHz MRF24J40) (AC182015-1)**
- **MPLAB® ICD 3 (DV164035)**

Introduction to MiWi™ DE

Agenda

- Introduction to MiWi™ DE -

- MiWi DE At A Glance
- MiWi Stack vs. Standard Model
- MiWi DE Benefits
- Regulatory Considerations
- MiWi DE Transceiver Radio Highlights
- Debugging MiWi DE Applications

Agenda

- Introduction to MiWi™ DE -

- MiWi DE At A Glance
- MiWi Stack vs. Standard Model
- MiWi DE Benefits
- Regulatory Considerations
- MiWi DE Transceiver Radio Highlights
- Debugging MiWi DE Applications

MiWi™ DE At A Glance

- **What is MiWi DE?**
  - Microchip proprietary Wireless Development Environment:
    - **Agency-certified transceivers, Protocol Stack (Networking Protocols, MAC, Physical Layers)**
    - **Configuration and Debug tools**

MiWi™ DE At A Glance

- **Application Spaces**
  - Low-power, low-data-rate, cost-sensitive applications:
    - **Wireless sensor networks, mouse/joystick, toys & games, security/HVAC/lighting**

Microchip Wireless (MiWi™) Development Environment

Application Configuration (`ConfigApp.h`) → Wireless Application

Wireless Application → MiApp → Interchangeable Microchip Proprietary Wireless Protocols:

- MiWi™ P2P Protoc
- MiWi™ Mesh Protoc
- MiWi™ PRO Protocol

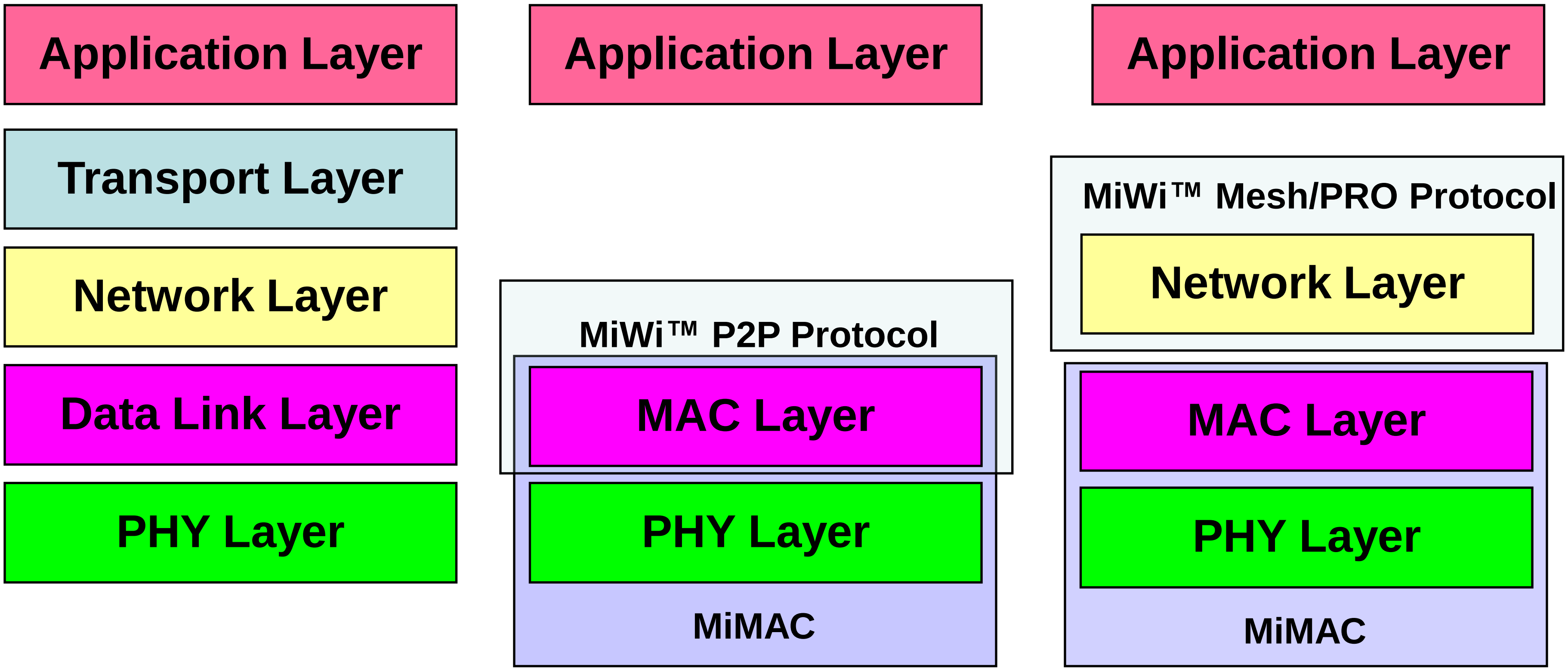
Network Protocol Configuration (`ConfigYYY.h`) → Interchangeable Microchip Proprietary Wireless Protocols

Interchangeable Microchip Proprietary Wireless Protocols → MiMAC → Microchip Interchangeable RF Transceivers:

- MRF24J40
- MRF49XA
- MRF89XA
- MRF24XA

MiMAC & Transceiver Configuration (`ConfigMRFxx.h`) → Microchip Interchangeable RF Transceivers

MiWi™ Protocol Stack



Standard Model

MiWi™ P2P Protocol

MiWi™ Mesh/PRO Protocol

MiWi™ DE Benefits

- Universal, Simple and Powerful Interface to Wireless Application Developers
- All Microchip Proprietary Wireless Protocols are Exchangeable
- All Microchip RF Transceivers are Exchangeable via MiMAC
- Minimize the Software Development Risk by Allowing Maximum Flexibility

Agenda

- Introduction to MiWi™ DE -

- MiWi DE At A Glance
- MiWi Stack vs. Standard Model
- MiWi DE Benefits
- **Regulatory Considerations**
- MiWi DE Transceiver Radio Highlights
- Debugging MiWi DE Applications

Regulatory Considerations

- **Regulations**
  - Regulations govern the operation of the device
    - **It's not what the technology can do, it's what the regulations will permit you to do**
  - Your product must conform to the regulations in the country you sell the device
  - Each country has their own regulations
  - International Telecommunications Union (ITU)

    http://www.itu.int/

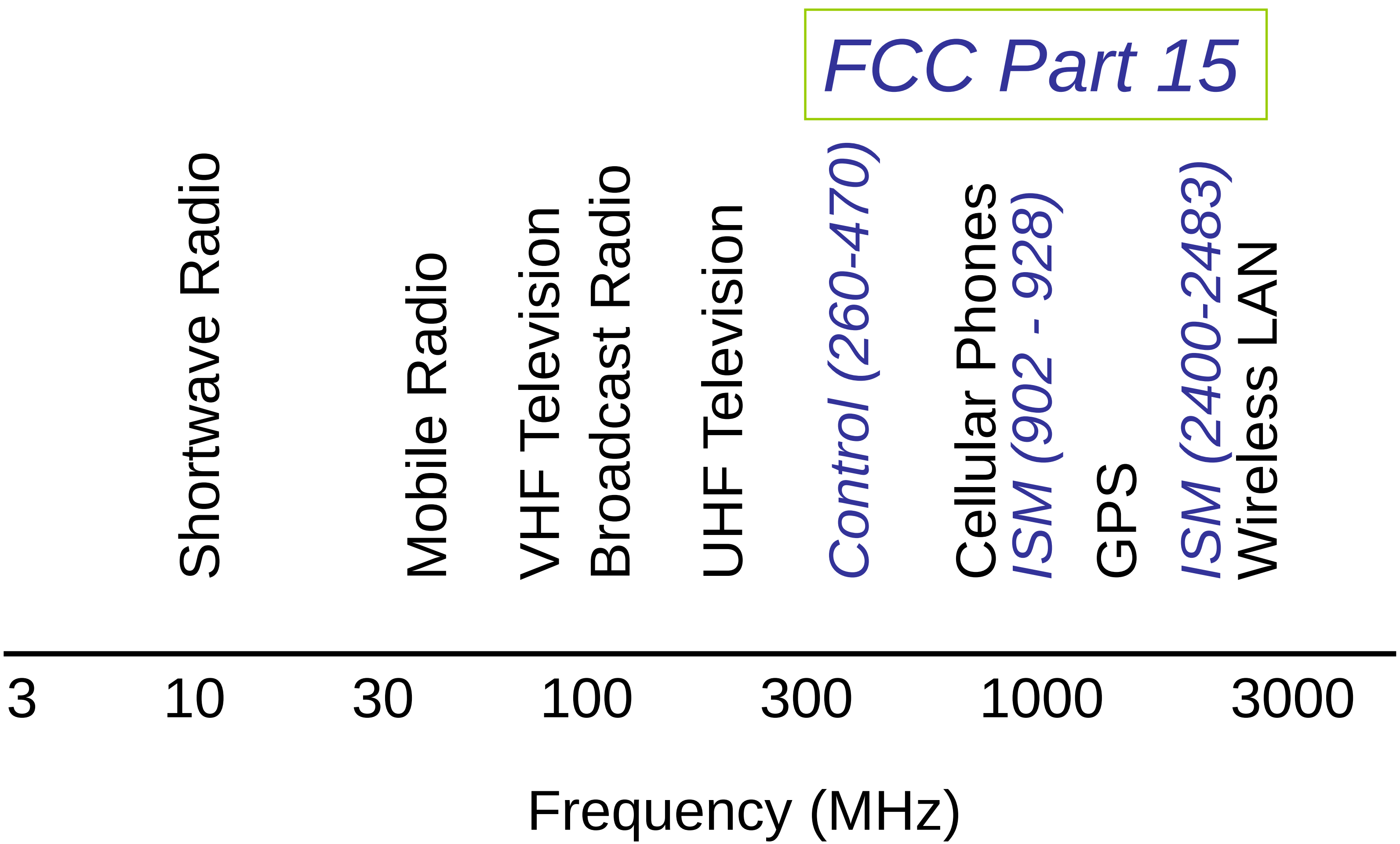
MICROCHIP MASTERs 2014

Regulatory Considerations

- **United States:**
  - U.S. Federal Communications Commission
    http://www.fcc.gov
  - Title 47 Part 15 Rules:
    http://www.fcc.gov/oet/info/rules/
- **Europe:**
  - ETSI EN 300 220-1
    http://www.etsi.org/
  - CEPT/ERC Recommendation 70-03
    http://www.cept.org/

Unlicensed Bands

(FCC Part 15)



Global ISM Bands

- **433MHz permitted everywhere except Japan**
- **2.4GHz permitted everywhere**

MICROCHIP MASTERs 2014

Considerations

- **Your product has to live with interference from other devices in the same band**
- **2.4 GHz Band**
  - Is the worldwide standard (good for regulatory compliance), but…
  - Shared by IEEE 802.11™ (WLAN), Bluetooth, and IEEE 802.15.4™ (WPAN) traffic

Product Support

- **MRF49XA**
  - 433/868/915MHz Support
  - No Module (reference design is available)
- **MRF24XA**
  - 2.4GHZ Support
  - No Module (reference design is available)
- **MRF89XA**
  - 868/915MHz Support
  - Agency certified module (FCC, IC, ETSI)
- **MRF24J40**
  - 2.4GHz Support
  - Agency certified module (FCC, IC, ETSI)

Agenda

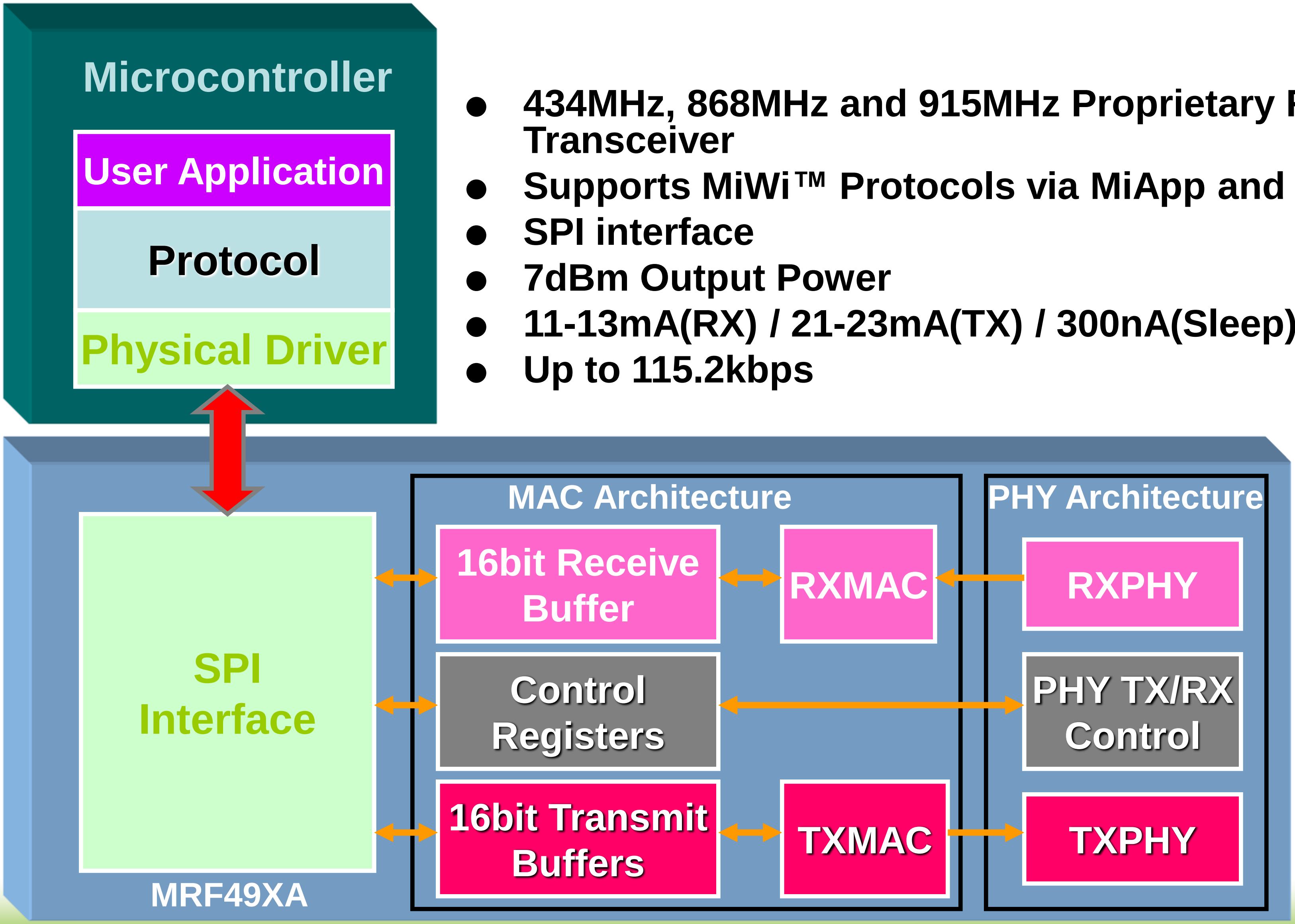
- Introduction to MiWi™ DE -

- MiWi DE At A Glance
- MiWi Stack vs. Standard Model
- MiWi DE Benefits
- Regulatory Considerations
- **MiWi DE Transceiver Radio Highlights**
- Debugging MiWi DE Applications

MiWi™ Transceiver Products

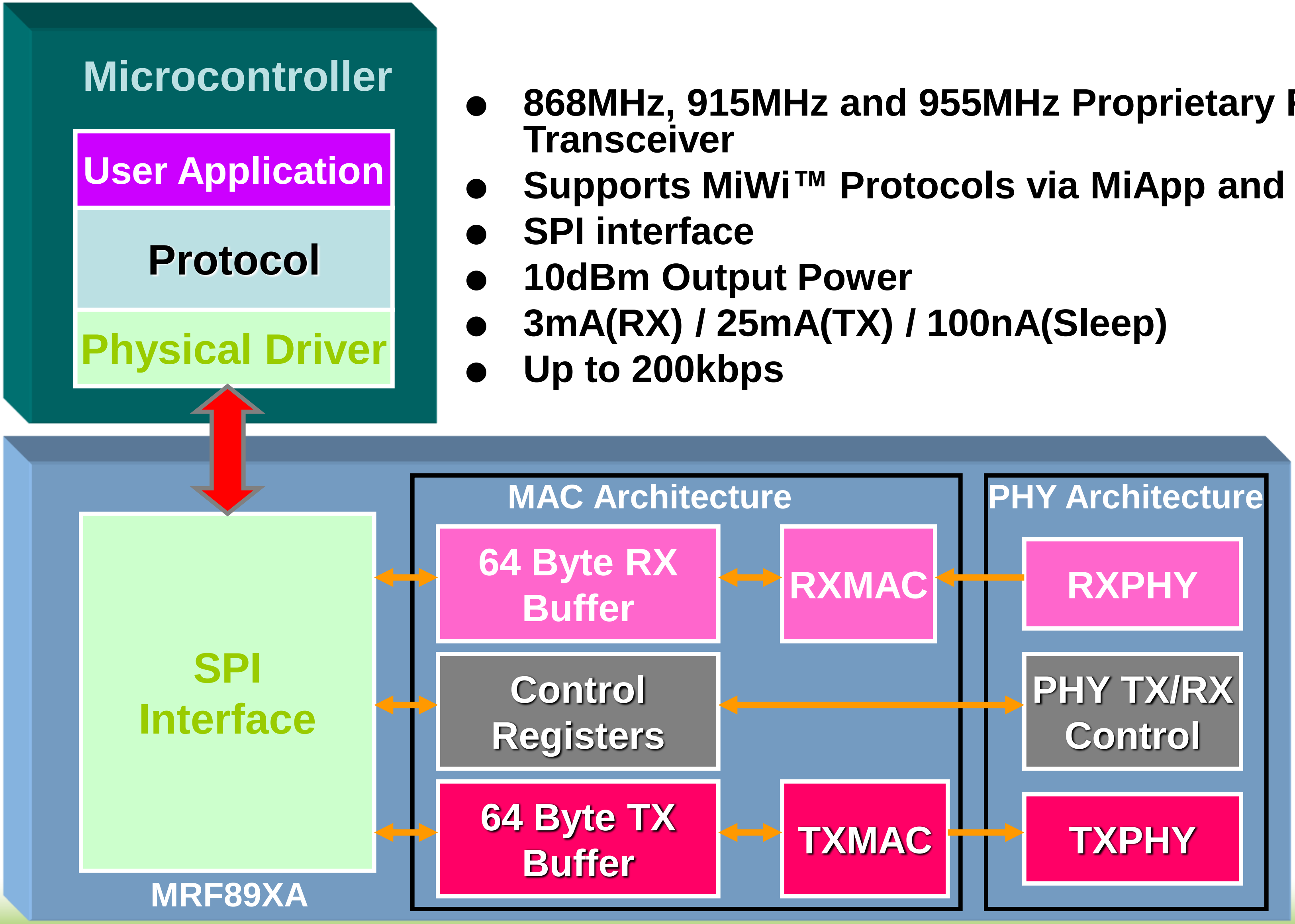
- **ISM Band - Sub-GHz**
  - MRF49XA — Transceiver IC
    - **433/868/915 MHz**
  - MRF89XA — Transceiver IC
    - **868/915 MHz**
  - MRF89XAM8A — Transceiver Module (868 MHz)
  - MRF89XAM9A — Transceiver Module (915 MHz)
- **ISM Band - 2.4GHz (IEEE 802.15.4™)**
  - MRF24J40 — Transceiver IC
  - MRF24J40MA — Transceiver Module (+0 dBm)
  - MRF24J40MB — Transceiver Module (+20 dBm)
  - MRF24J40MC — Transceiver Module (+20 dBm + ext. ant)
  - MRF24XA — Transceiver IC

MRF49XA Transceiver



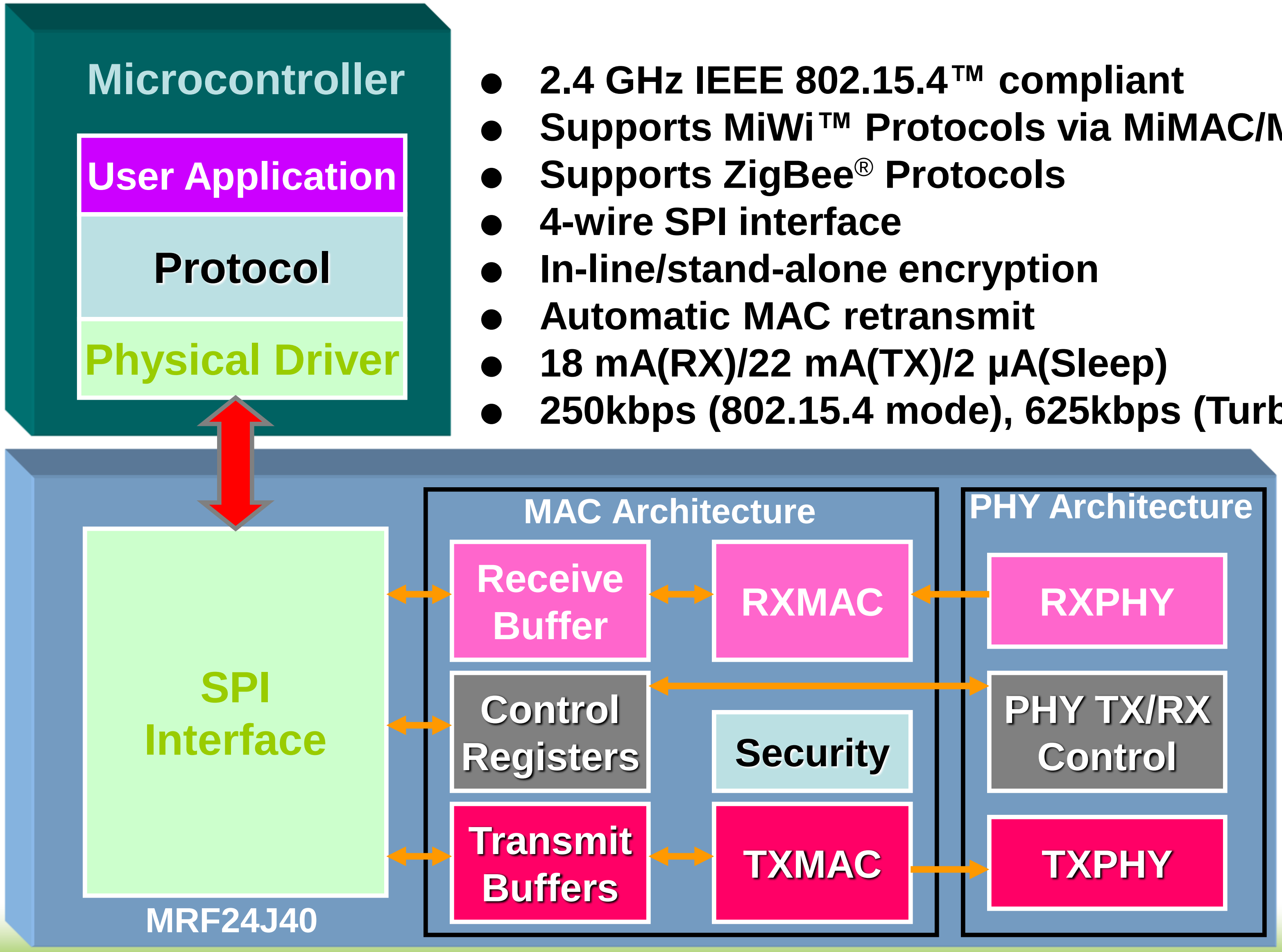
- 434MHz, 868MHz and 915MHz Proprietary RF Transceiver
- Supports MiWi™ Protocols via MiApp and MiMAC
- SPI interface
- 7dBm Output Power
- 11-13mA(RX) / 21-23mA(TX) / 300nA(Sleep)
- Up to 115.2kbps

MRF89XA Transceiver



- 868MHz, 915MHz and 955MHz Proprietary RF Transceiver
- Supports MiWi™ Protocols via MiApp and MiMAC
- SPI interface
- 10dBm Output Power
- 3mA(RX) / 25mA(TX) / 100nA(Sleep)
- Up to 200kbps

MRF24J40 Transceiver

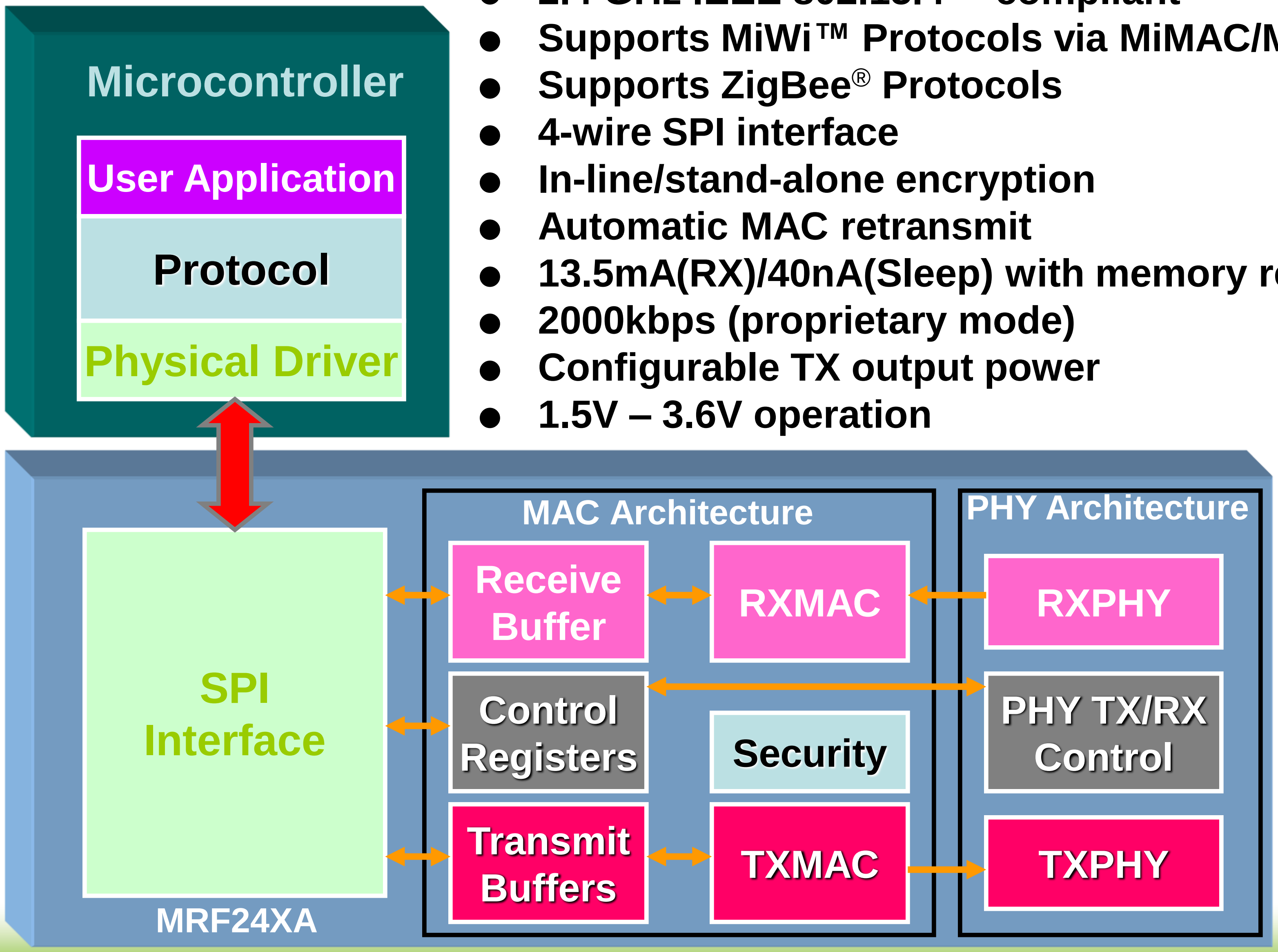


- 2.4 GHz IEEE 802.15.4™ compliant
- Supports MiWi™ Protocols via MiMAC/MiApp
- Supports ZigBee® Protocols
- 4-wire SPI interface
- In-line/stand-alone encryption
- Automatic MAC retransmit
- 18 mA(RX)/22 mA(TX)/2 µA(Sleep)
- 250kbps (802.15.4 mode), 625kbps (Turbo mode)

MRF24XA Transceiver

MICROCHIP
MASTERs 2014

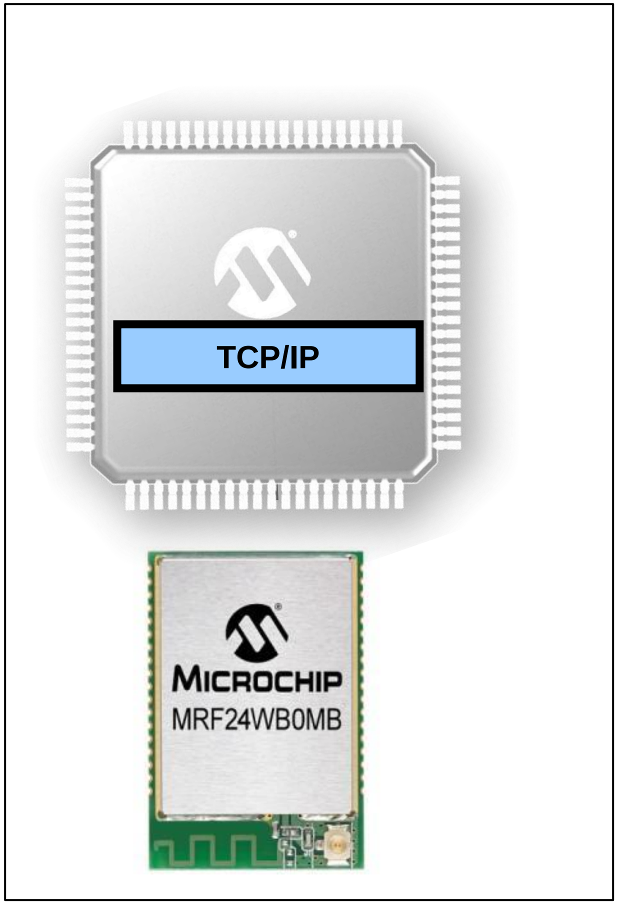
- 2.4 GHz IEEE 802.15.4™ compliant
- Supports MiWi™ Protocols via MiMAC/MiApp
- Supports ZigBee® Protocols
- 4-wire SPI interface
- In-line/stand-alone encryption
- Automatic MAC retransmit
- 13.5mA(RX)/40nA(Sleep) with memory retention
- 2000kbps (proprietary mode)
- Configurable TX output power
- 1.5V – 3.6V operation

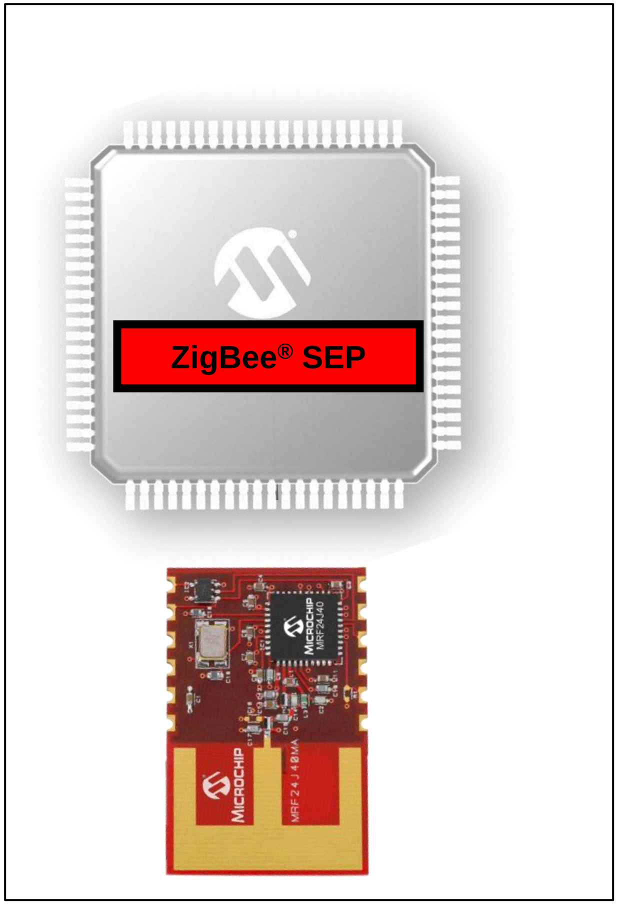


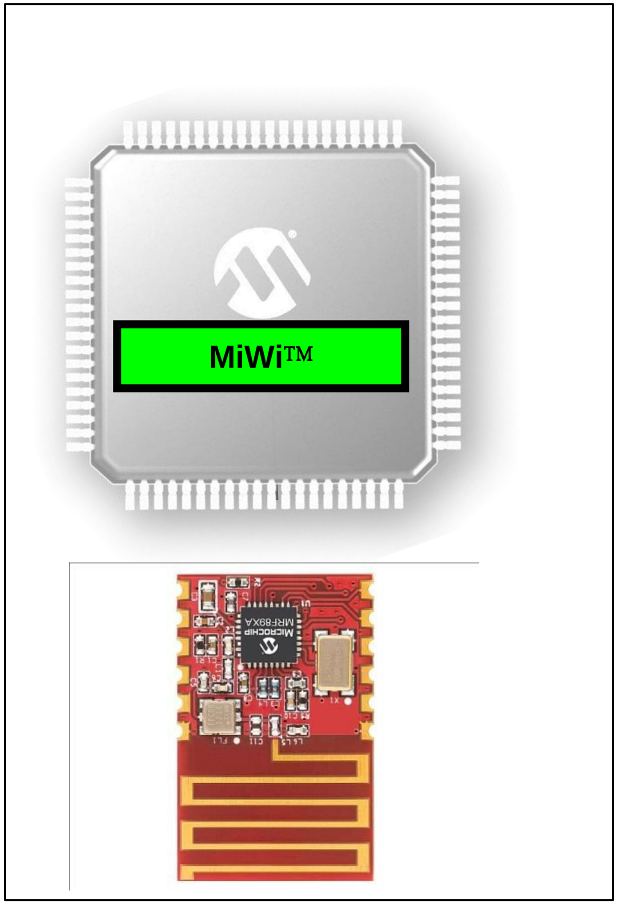
MRF24XA
Key Features and Benefits

Key Feature	Benefit
1.5V – 3.6V, 13.5mA Rx and 40nA Sleep Current	Low operating voltage and current enables longer battery life
Proprietary mode – Up to 2 Mbps	2Mbps frames can reduce radio ON-time of radio by a factor of 4 to 8 with respect to 250kbps
Inferred destination addressing	Destination address is computed as part of frame and is omitted from the frame header – thus saving overhead and save power
Configurable Tx output power	Output power can be configured from -17.5 to 0dBm depending on application requirements
Hardware security engine	Security algorithm implemented on hardware enables quick encryption and reduces the ON time of the MCU
SPI interface	Easy to interface most of the PIC® microcontrollers

Module Concept

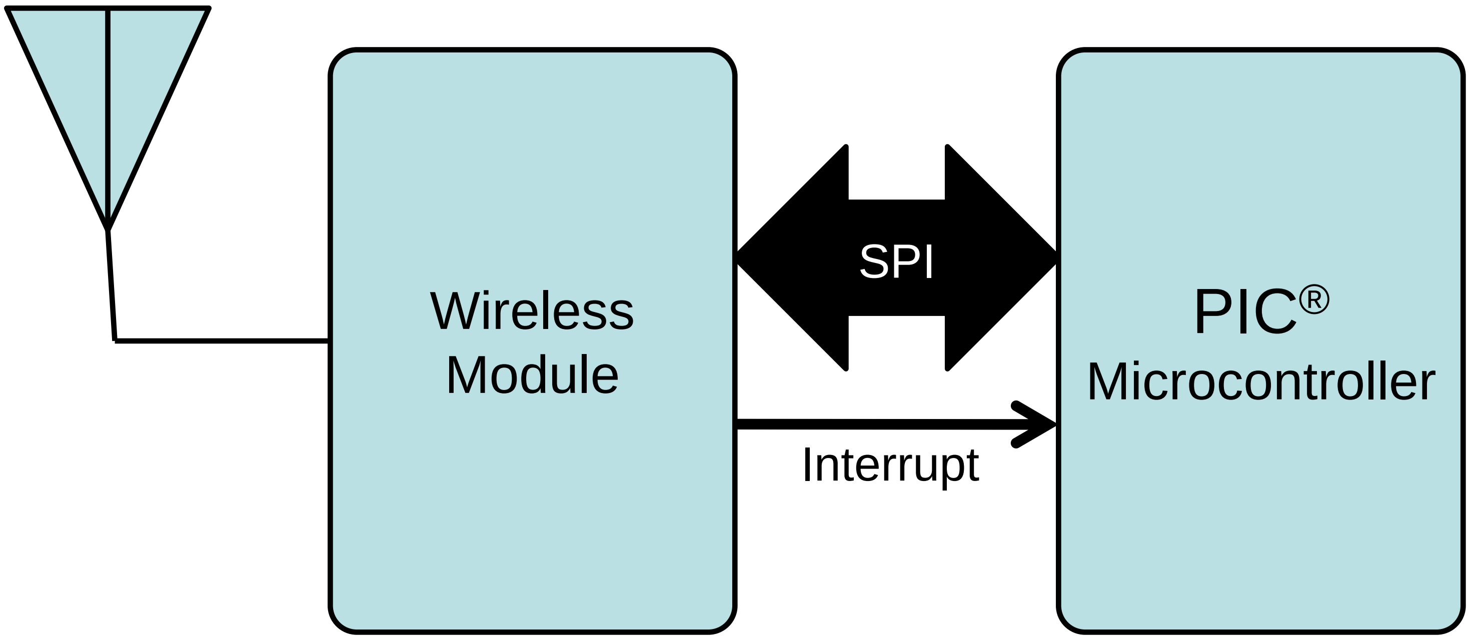






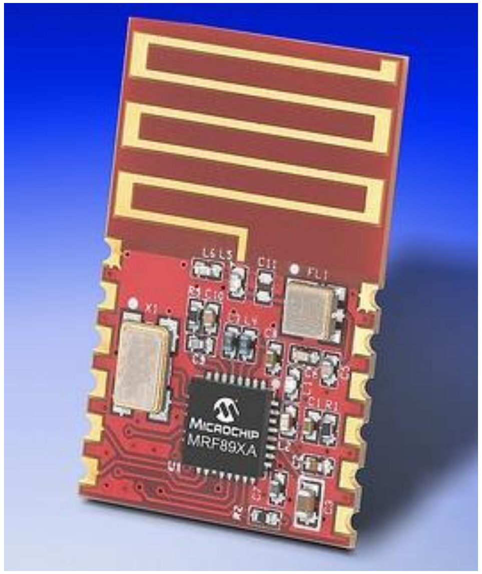
- RF design complete, optimized antenna
- Agency Certified
- Surface Mount

Module



MRF89XAMxA Module

- FCC/IC/ETSI Certified
- RF knowledge is NOT required
- No Certification cost
- Quick Time to Market
- Proven Performance



- MRF89XAM8A – 868MHz
- MRF89XAM9A – 915MHz

MRF24J40MA/MB/MC Modules

- **2.4 GHz IEEE 802.15.4™ Compatible**
- **FCC/IC/ETSI Certified**
- **Module features:**
  - Integrated PCB antenna
  - Matching circuit components
  - Surface-mountable PCB
  - MC and ME versions have external antenna
  - Tx Power / Rx Sensitivity
    - **MA = +0 dBm** — **MA = -95 dBm**
    - **MB = +20 dBm** — **MB = -102 dBm**
    - **MD = +20 dBm** — **MD = -104 dBm**
      - 77mA transmit for MD vs. 120mA for MB

MRF49XA Development Boards

- **MRF49XA PICtail™/PICtail Plus Daughter Boards**
  - Part # AC164137-1 (433 MHz),
  - Part # AC164137-2 (868/915 MHz),
  - Two Daughter Boards per package

MRF89XA Development Boards

- **MRF89XA PICtail™/PICtail Plus Daughter Boards**
  - Part # AC164138-1 (868 MHz)
  - Part # AC164138-2 (915 MHz)
  - Two Daughter Boards per package

MRF24J40 Development Boards

- **MRF24J40MA**
  - Part # AC164134-1

- **MRF24J40MB**
  - Part # AC164134-2

MRF24XA Development Boards

- **Development Support**:
  - MRF24XA PICtail™/ PICtail Plus Daughter Board
  - Part # AC164152-1, $24.95

- **Compatible with**
  - Explorer 16 board (16/32-bit)
  - PIC18 Explorer board (8-bit)

MRF24XA PICtail/ PICtail Plus

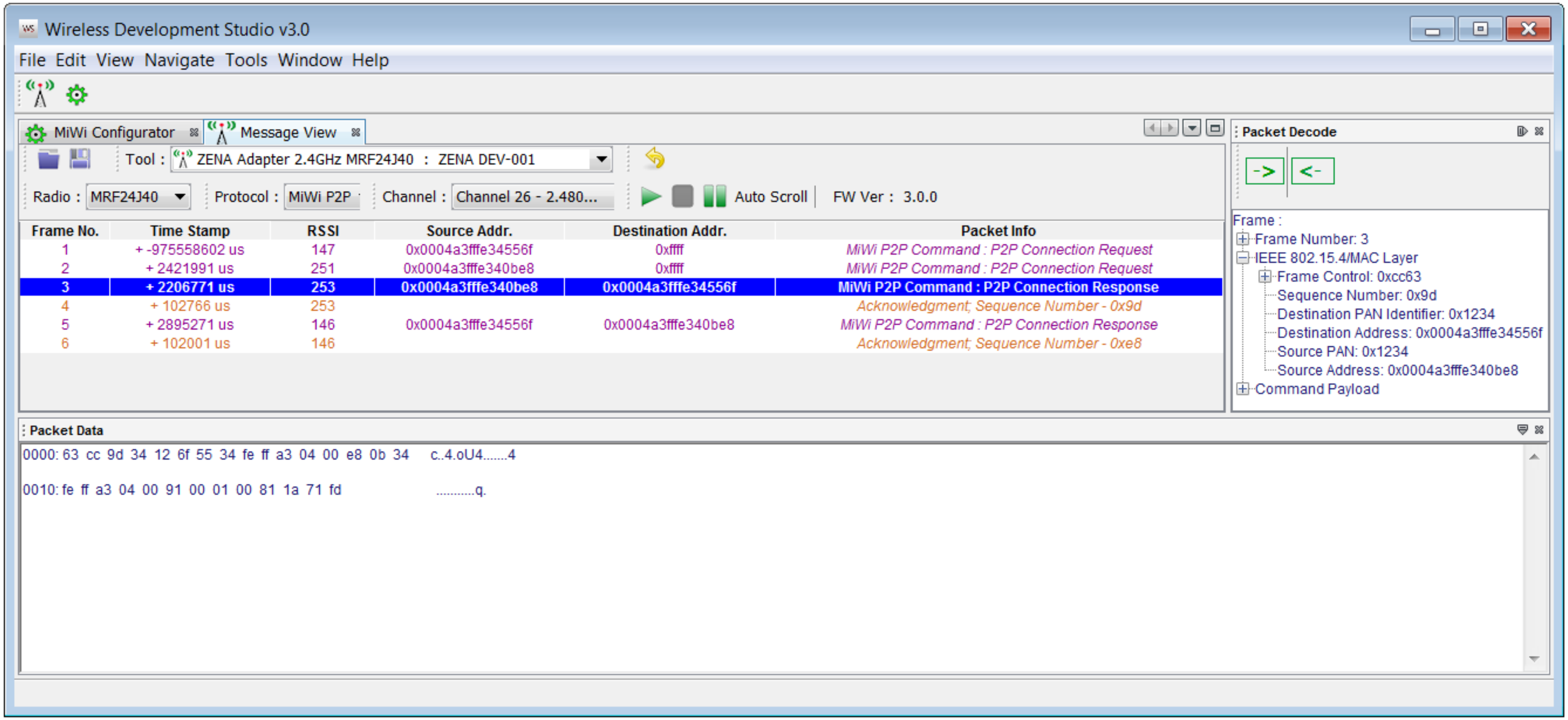
Agenda

- Introduction to MiWi™ DE -

- MiWi DE At A Glance
- MiWi Stack vs. Standard Model
- MiWi DE Benefits
- Regulatory Considerations
- MiWi DE Transceiver Radio Highlights
- **Debugging MiWi DE Applications**

Debugging MiWi™ Applications

- Wireless Development Studio (WDS) -

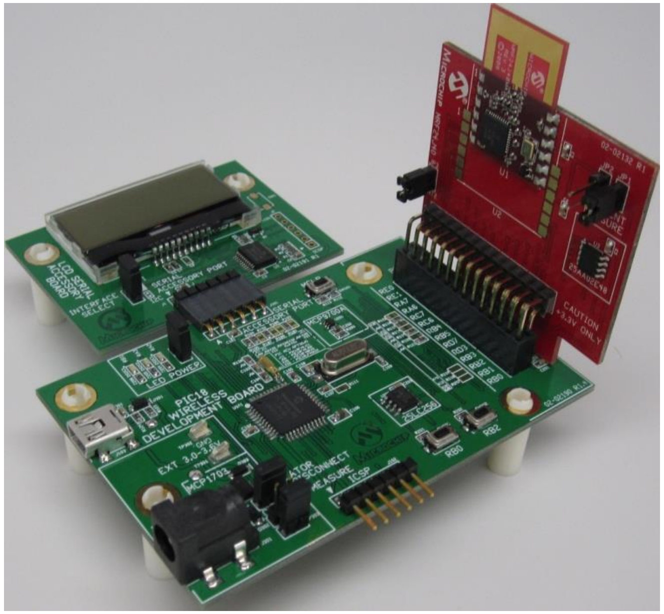


www.microchip.com/wds

Adapters

- **ZENA™ Wireless Adapter 2.4 GHz MRF24J40MA (AC182015-1)**

- **8-bit WDK Platform**
  - MRF24J40
  - MRF89XA (868/915)



Microchip Wireless MAC (MiMAC)

Foundation for Networking

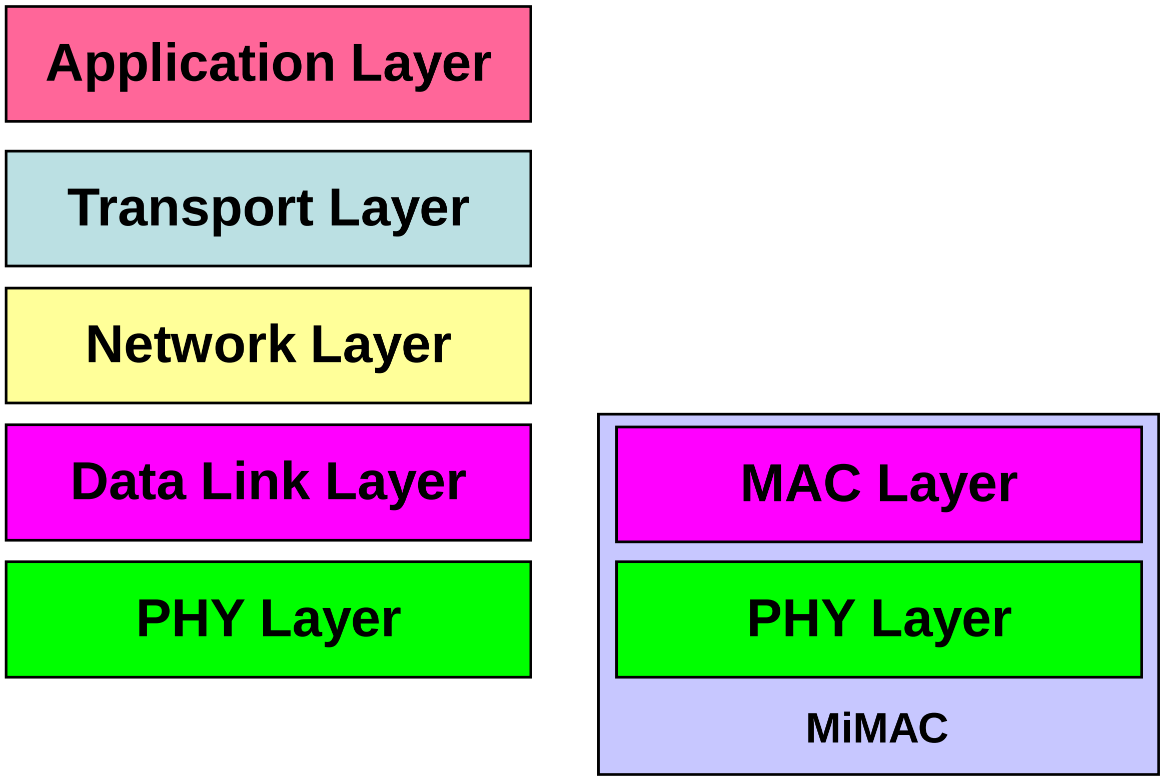
Agenda
- Microchip Wireless MAC (MiMAC)-

- **MiMAC Layers**
- **MAC Layer Functions**
- **MiMAC Physical Layer**
  - Lab 1 – Configure The Transceiver
- **Device Types and Roles**
- **Network Topologies & Addressing**
- **Channel Scanning**
- **Network Creation & Association**
- **MiMAC Data Transfer Model**
- **MiMAC Frame Structure**
- **Security**
  - Lab 2 – Configure Security
- **MiMAC APIs**

Agenda
- Microchip Wireless MAC (MiMAC)-

- **MiMAC Layers**
- **MAC Layer Functions**
- **MiMAC Physical Layer**
  - Lab 1 – Configure The Transceiver
- **Device Types and Roles**
- **Network Topologies & Addressing**
- **Channel Scanning**
- **Network Creation & Association**
- **MiMAC Data Transfer Model**
- **MiMAC Frame Structure**
- **Security**
  - Lab 2 – Configure Security
- **MiMAC APIs**

MiMAC Layers



OSI Model **MiMAC Function**

MAC Layer Functions/Services

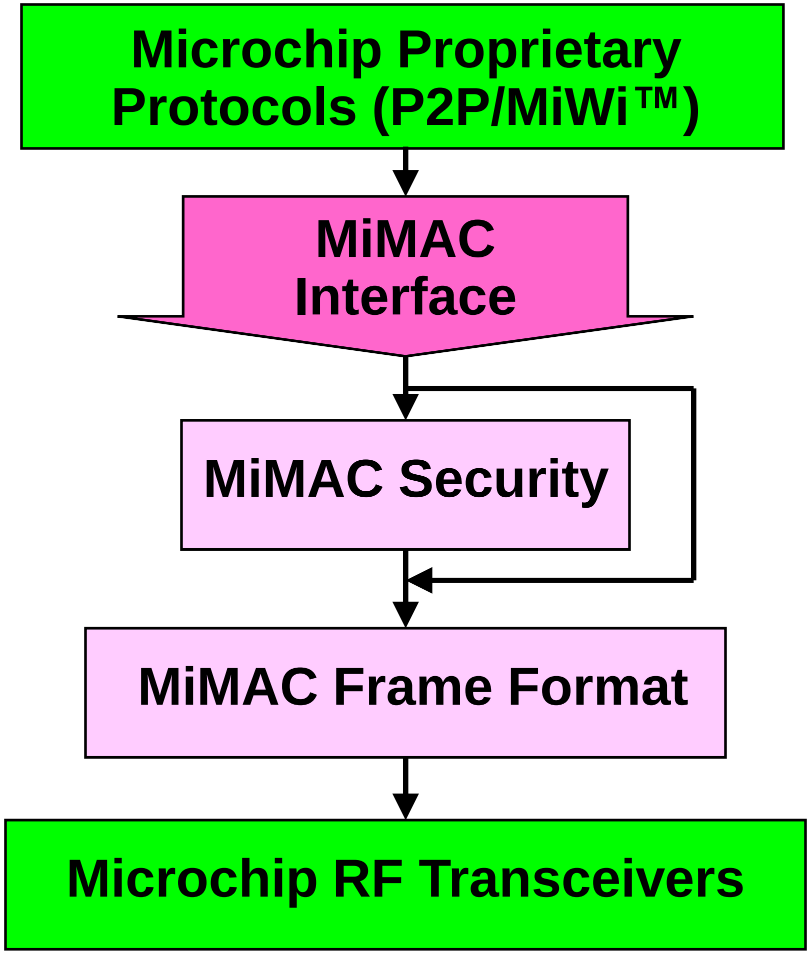
- Channel access (CSMA/CA)
- Frame validation
- Acknowledged frame delivery
- Security Mechanisms

Challenges in MAC Layer Design

- **Transceivers differ widely in features**
  - Some have a well defined MAC layer
    - **MRF24J40 and MRF24XA (IEEE 802.15.4™)**
      - Hardware-based. Includes frame format and security engine
  - Some do not
    - **MRF49XA andMRF89XA**
      - Require software-driven MAC function

MiMAC Overview

- **Microchip Media Access Control (MiMAC) Layer**
  - MiMAC Software-based Frame Format
  - MiMAC Software-based Security Module
  - MiMAC Programming Interface to Microchip Proprietary Protocols



MICROCHIP MASTERs 2014

MiMAC Physical Layer
- Selecting the Transceiver -

- **Application-layer macro**
- **`ConfigApp.h`**

ConfigApp.h

```
//------------------------------------------------------------------------
// Definition of RF Transceiver. ONLY ONE TRANSCEIVER CAN BE CHOSEN
//------------------------------------------------------------------------

#define MRF24J40

//#define MRF49XA

//#define MRF89XA
```

MiMAC Physical Layer
- Configurable Transceiver Parameters -

- **Key configuration parameters**
  - Frequency band
  - Enable Clear channel assessment (CCA) for CSMA-CA
  - Enable acknowledgement of transmissions
  - Data transmission rate
  - Security Mode, Key
- **Not all parameters are available across all transceivers**
  - Proprietary transceivers need software-based MAC services →more macro options are defined

MiMAC Physical Layer

- Configuring The Transceiver -

- **MiMAC-layer macro**
- **`ConfigMRFxxx.h`**

ConfigMRF89XA.h

```
…
#define BAND_902
//#define BAND_915
//#define BAND_863

#define DATA_RATE_20
//#define DATA_RATE_40
//#define DATA_RATE_50
//#define DATA_RATE_66
//#define DATA_RATE_100
//#define DATA_RATE_200

#define ENABLE_CCA
#define ENABLE_ACK
…
```

Configure The Transceiver

LAB 1

Lab 1

Configure The Transceiver

Objective

- **Learn how to reconfigure an existing MiWi™ project to use a different transceiver.**
- **Learn how to configure/use Wireless Development Studio (WDS) with the ZENA™ adapter to capture and filter packets.**

Lab 1

Configure The Transceiver

Procedure

- **Build/Run the base project using the default transceiver module (MRF24J40MA).**
  - Configure/Use WDS to capture packets.
- **Modify the project configuration settings to support a different module (MRF89XAM8A or MRF89XAM9A).**
  - Configure/Use WDS to capture packets.

Lab 1

Configure The Transceiver

Results

- **Chat Demo runs unchanged.**
  - Easy to change the transceiver.
  - No application-layer code re-write is required.

Lab 1

Configure The Transceiver

Conclusions

- Learned how to edit the MiMAC configuration for a project in order to change the transceiver.
- Learned how to configure and use Wireless Development Studio to filter and capture MiWi™ packets.

Agenda

- Microchip Wireless MAC (MiMAC)-

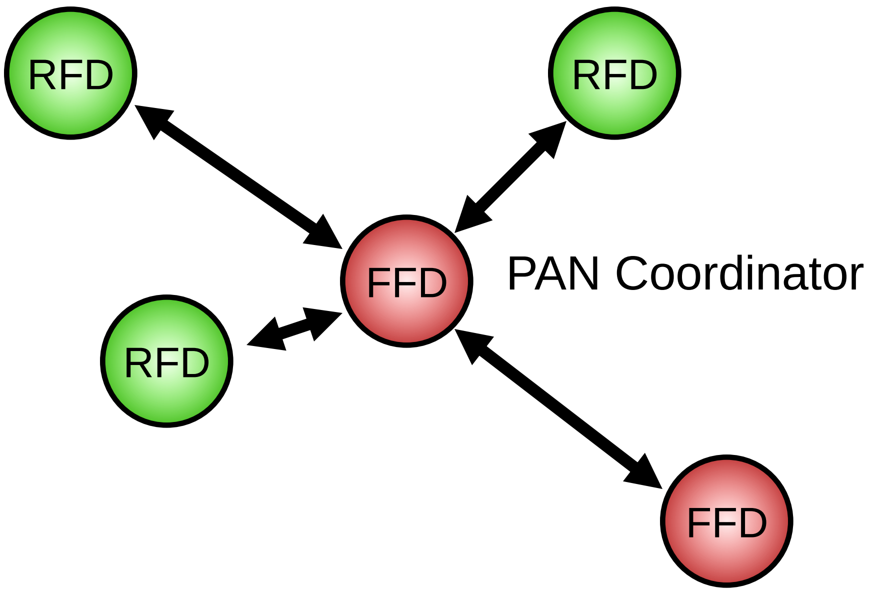
- MiMAC Layers
- MAC Layer Functions
- MiMAC Physical Layer
  - Lab 1 – Configure The Transceiver
- **Device Types and Roles**
- **Network Topologies & Addressing**
- Channel Scanning
- Network Creation & Association
- MiMAC Data Transfer Model
- MiMAC Frame Structure
- Security
  - Lab 2 – Configure Security
- MiMAC APIs

Device Types & Roles

- **Full Function Device – FFD**
  - PAN Coordinator (Starts a MiWi network)
  - Coordinator (Can act as a router)
  - Network Device (only produces data)
- **Reduced Function Device – RFD**
  - Network Device role only
  - Minimal resources (power and memory)
  - Lowest cost

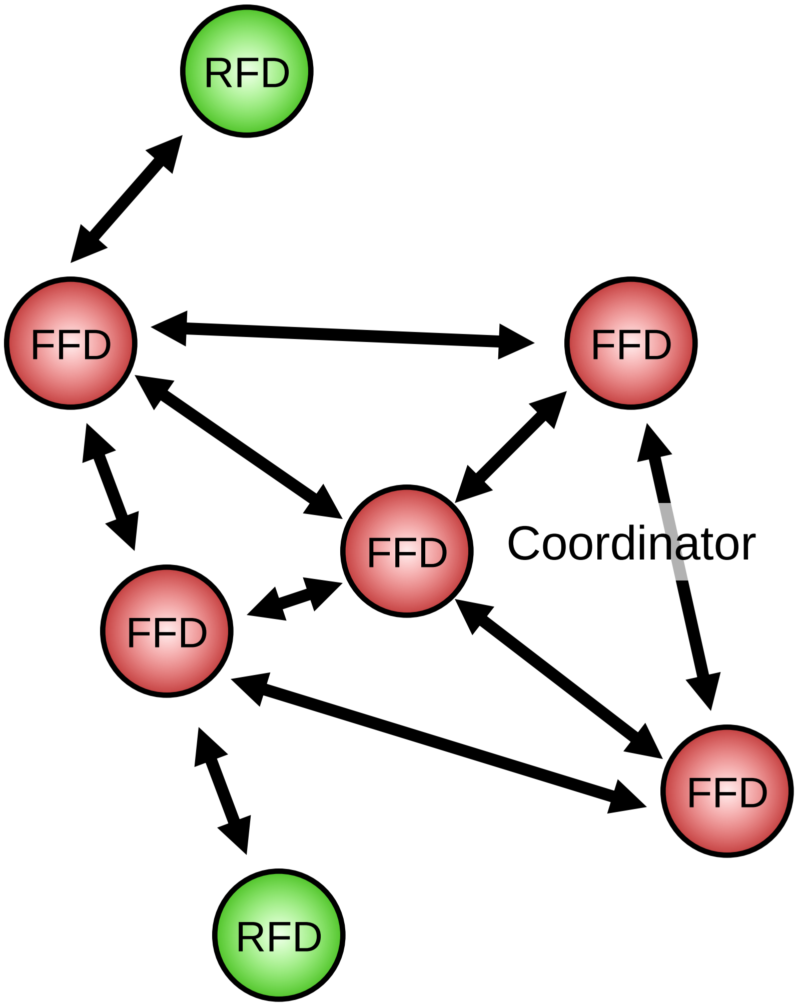
Network Topologies

- **Star Topology**
  - Low message latency
  - Centralized network control
  - Can only cover a limited area (single hop)

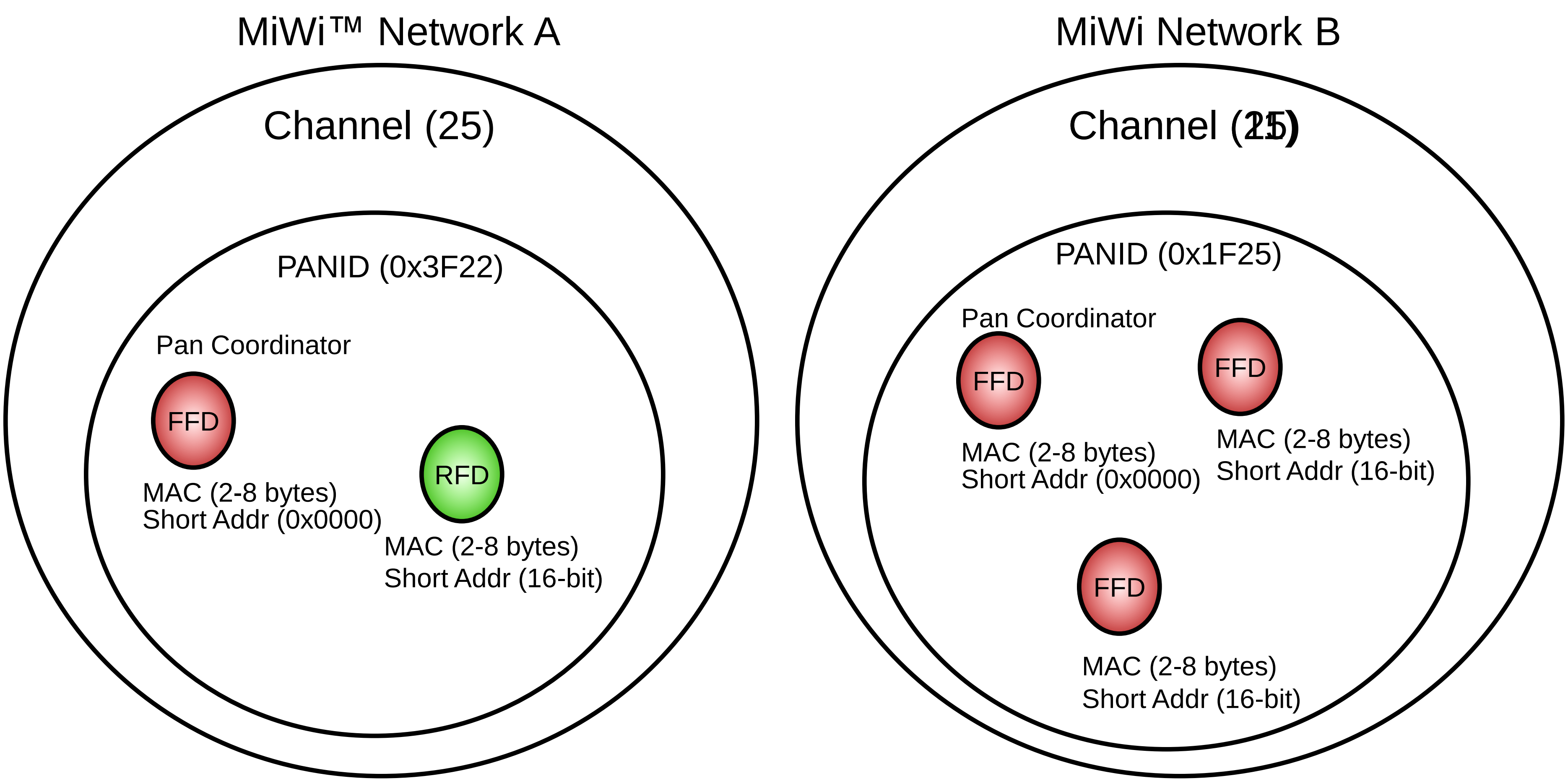


Network Topologies

- **Peer-to-Peer Topology**
  - Any device may communicate with any other in radio range (single-hop)
  - Upper stack layers can create mesh, cluster and cluster tree topologies



Addressing Basics



Can 2 networks use the same channel ?

…Yes, as long as the PANID is unique

Setting the Node Addresses

- **MAC Address (2-8 bytes)**
  - Assigned at manufacture.
  - Can be fixed at compile-time, dynamically programmed or can use IEEE EUI EEPROM (25AA02E48)
- **16-bit Short Address**
  - Dynamically assigned by a coordinator on joining a network (MiWi™ Mesh/PRO)
    - **Unique within a network**
    - **Node's Full Network Address = PANID:ShortAddr**

Setting the Channel, PANID

- **Procedure for setting the channel is defined by the application**
  - Can be fixed at compile-time, or dynamically determined via channel assessment procedure
    - **`BOOL MiApp_SetChannel(BYTE channel)`**
- **Procedure for setting the PANID is defined by the application**
  - Can be fixed at compile-time, or dynamically determined via timer value, dip switches, or user-input
  - **`ConfigApp.h`**
    - **`#define MY_PAN_ID 0x0134`**
    - **Set to 0xFFFF to define a different procedure for setting PANID**

Node ID

- An additional identifier that is exchanged between nodes on association and stored in each

ConfigApp.h

```
/**********************************************************************/
// ADDITIONAL_NODE_ID_SIZE defines the size of additional payload
// will be attached to the packets in the handshake process. The
// additional payload is defined by the application and declared in main.c
/**********************************************************************/
#define ADDITIONAL_NODE_ID_SIZE   6
```

main.c

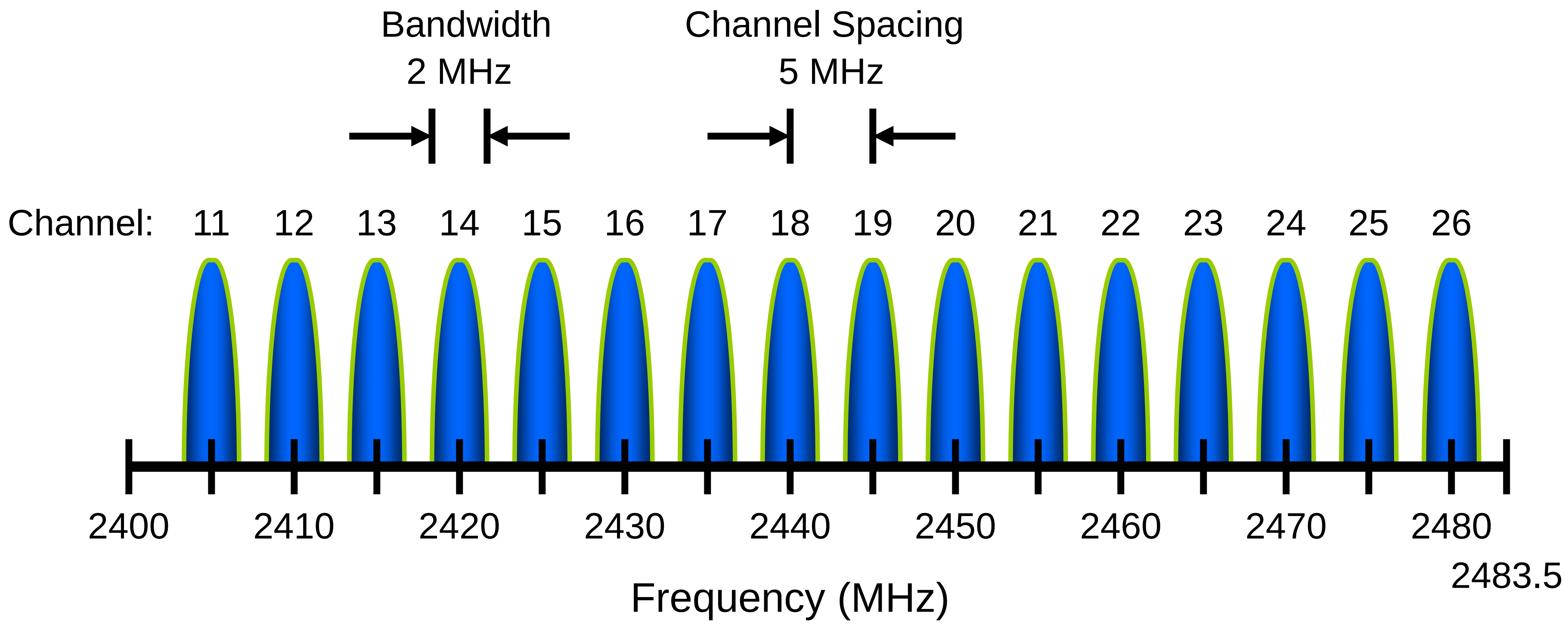
```
/**********************************************************************/
// Define AdditionalNodeID[] array
/**********************************************************************/
#if ADDITIONAL_NODE_ID_SIZE > 0
    BYTE AdditionalNodeID[ADDITIONAL_NODE_ID_SIZE]= {'S', 'W', 'I', 'T', 'C', 'H'};
#endif
```

Agenda
- Microchip Wireless MAC (MiMAC)-

- MiMAC Layers
- MAC Layer Functions
- MiMAC Physical Layer
  - Lab 1 – Configure The Transceiver
- Device Types and Roles
- Network Topologies & Addressing
- **Channel Scanning**
- **Network Creation & Association**
- MiMAC Data Transfer Model
- MiMAC Frame Structure
- Security
  - Lab 2 – Configure Security
- MiMAC APIs

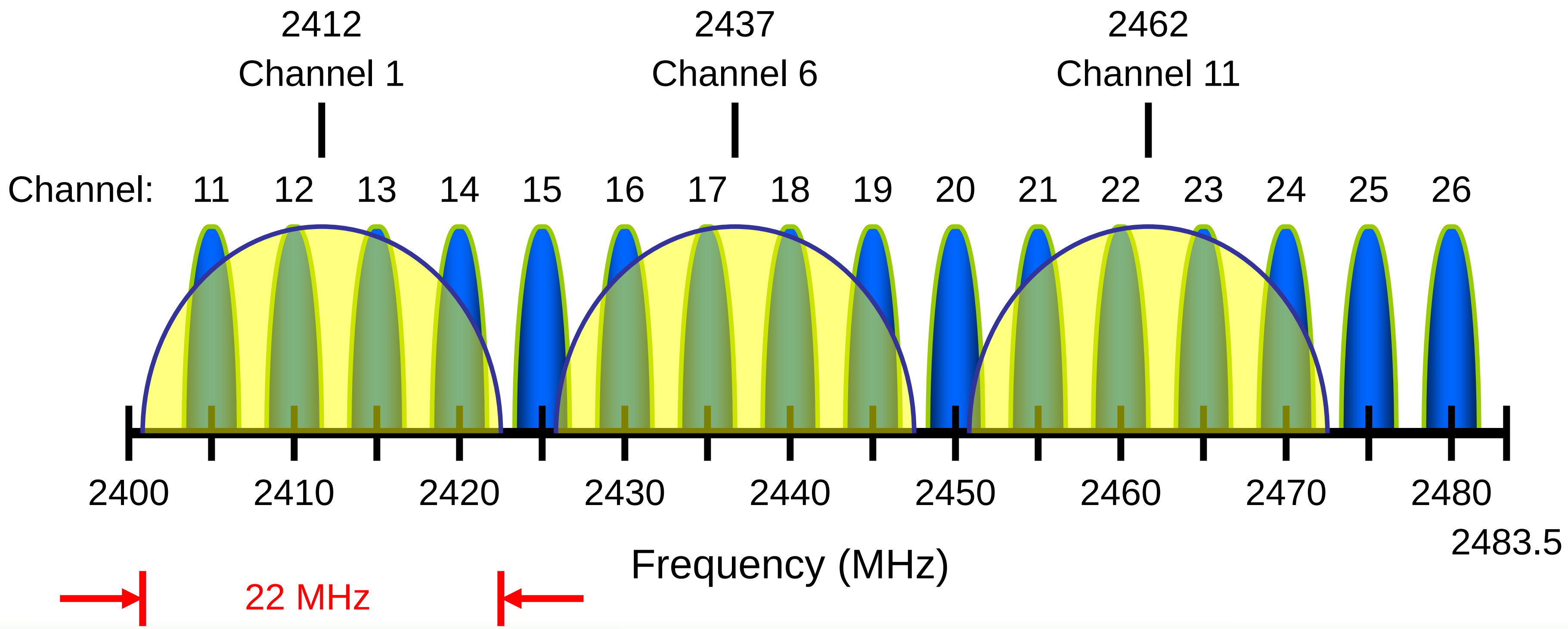
Channel Assignment IEEE 802.15.4™ Transceiver

2.4 GHz PHY Channel Assignment



Channel Overlap

2.4 GHz PHY Channel Assignment
IEEE 802.15.4
vs.
IEEE 802.11 (United States)



Channel Assignment Proprietary Transceivers

- **Each ISM band is divided up to 32 channels**
- **Channel spacing (therefore number of channels <u>available</u>) depends on bandwidth of ISM band and data rate setting.**
  - See the transceiver header file for limits (eg. `MRF89XA.h`)
- **Setting the channel requires defensive coding**
  - `BOOL MiApp_SetChannel(BYTE channel)` returns `FALSE` if the channel parameter is not supported

Channel Assessment

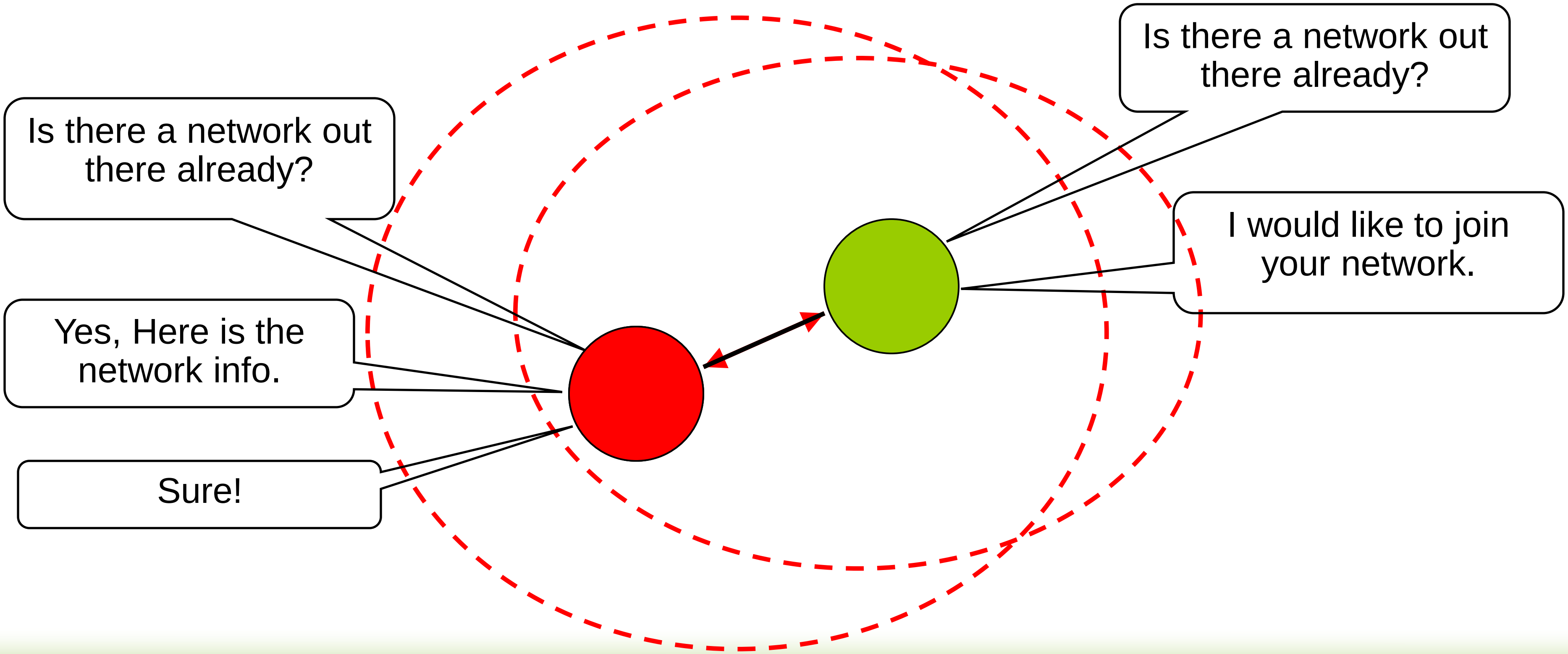
- **MiMAC enables channel scanning for upper layers**
  - Energy/Carrier Detection Scan
    - **Can be used by the upper layers in the PAN Coordinator to find a suitable channel for starting a network**
    - **`MiMAC_ChannelAssessment(AssessmentMode)`**

Starting a PAN

- **A PAN is started by**
  - an FFD
  - after an active channel scan
  - a suitable PAN identifier is selected
- **The algorithm for selecting a suitable PAN identifier (PANID) from the list of PAN descriptors returned from the active channel scan procedure is controlled by the application layer.**

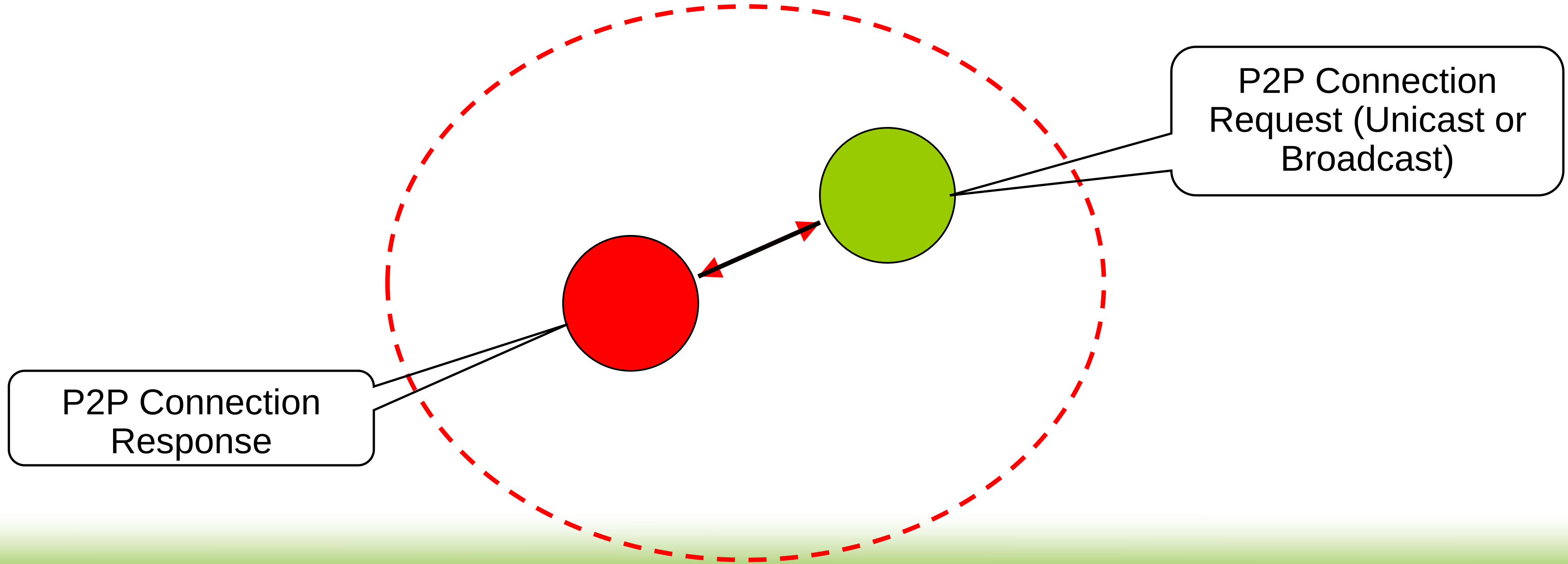
Example

- **Network Creation & Association**
  - MiWi™ Mesh/PRO:



Creating a P2P Connection

- **Designed for simplicity**



PAN Coordinator Selection

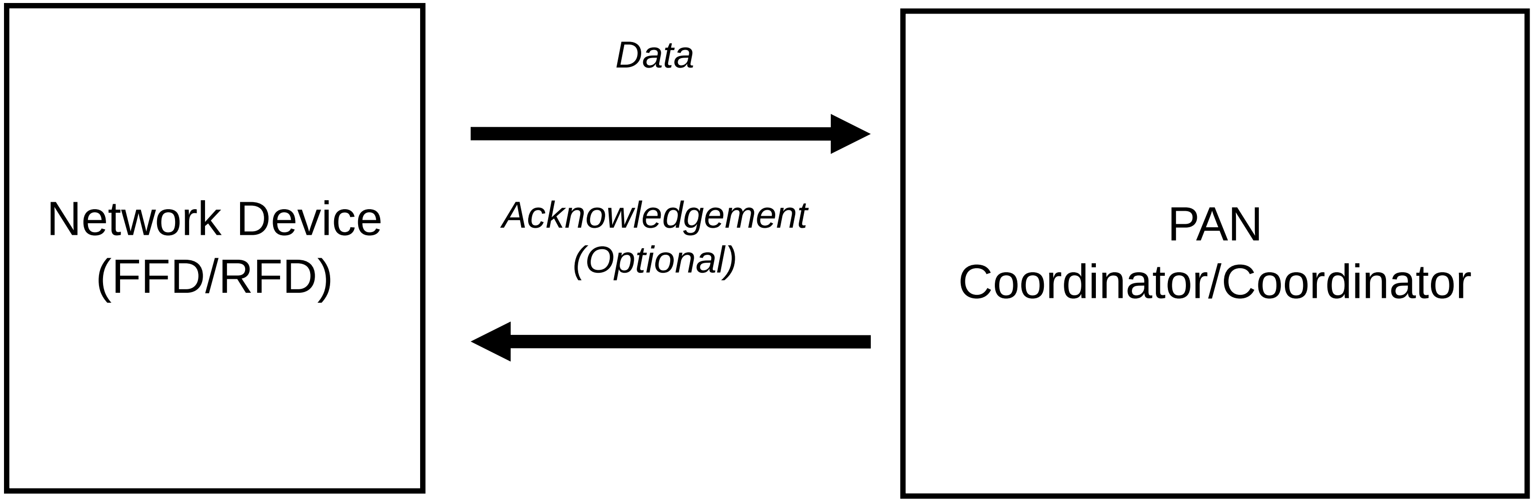
- **Some application-dependent scenarios:**
  - *Dedicated PAN coordinator*
    - **Ex. Security panel (FFD) + sensors (RFDs)**
  - *Event-determined PAN coordinator*
    - **Button press by a user – all devices are FFDs**
  - *Self-determined PAN coordinator*
    - ***Ad-hoc network formation – doesn’t matter who is the PAN coordinator – all devices are FFDs***

Agenda
- Microchip Wireless MAC (MiMAC)-

- MiMAC Layers
- MAC Layer Functions
- MiMAC Physical Layer
  - Lab 1 – Configure The Transceiver
- Device Types and Roles
- Network Topologies & Addressing
- Channel Scanning
- Network Creation & Association
- **MiMAC Data Transfer Model**
- **MiMAC Frame Structure**
- Security
  - Lab 2 – Configure Security
- MiMAC APIs

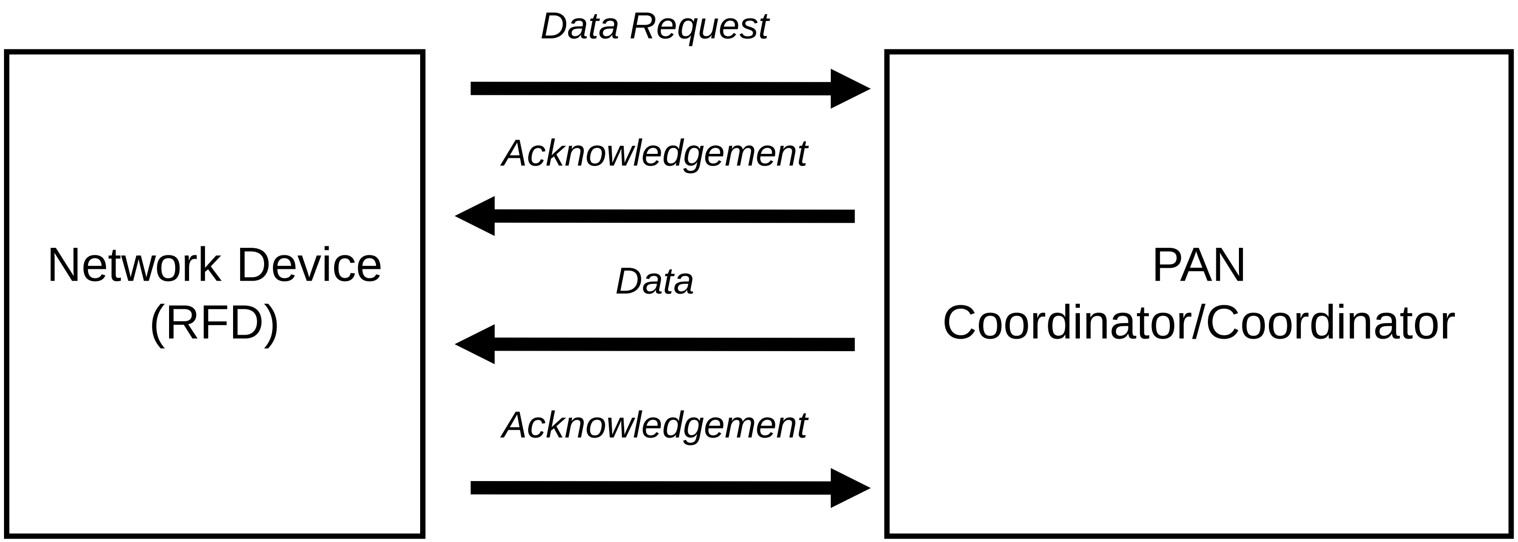
Data Transfer to a Coordinator/Device

- **<u>Direct</u> transmission mode**



Data Transfer to a Sleeping Device

- **Indirect transmission mode**
- **Network Device (RFD) polls for messages stored on coordinator**



MiMAC Frame Format

- **MiMAC Frame Format Requirements**
  - Provide Network Capabilities
    - **Essentially the Same Capability as IEEE 802.15.4**
  - Provide Concise Frame Format
    - **MAC Header Length (Overhead):**
      - IEEE 802.15.4: Minimum 9 Bytes

Frame Control	Seq Num	Dest PAN ID	Dest Address	Source Address
2 Bytes	1 Byte	2 Bytes	2 Bytes	2 Bytes

      - MiMAC: Typical 2 Bytes

Agenda

- Microchip Wireless MAC (MiMAC)-

- MiMAC Layers
- MAC Layer Functions
- MiMAC Physical Layer
  - Lab 1 – Configure The Transceiver
- Device Types and Roles
- Network Topologies & Addressing
- Channel Scanning
- Network Creation & Association
- MiMAC Data Transfer Model
- MiMAC Frame Structure
- **Security**
  - Lab 2 – Configure Security
- MiMAC APIs

MICROCHIP MASTERs 2014

Security Goals

- **Communicate sensitive data (Data Privacy/Confidentiality)**
  - Snooping or eavesdropping
- **Guarantee data is unmodified (Data Integrity)**
  - Tampering, “Man in the middle”
- **Assure source of data (Data Authenticity)**
  - Redirection, “Man in the middle”

MiMAC Security Overview

- **Uses on-chip hardware security module where available**
  - MRF24J40
  - Encryption Engine: AES-128
  - Additional Modes: CTR, CBC-MAC, CCM
- **Otherwise uses a software module**
  - MRF49XA, MRF89XA
  - Encryption Engine: XTEA-64, XTEA-128
  - Additional Modes: CTR, CBC-MAC, CCM
- **References: AN1283A, 1561_SEC, Schneier et al.**

Enabling Security

- **Application-layer macro**
- **`ConfigApp.h`**

ConfigApp.h

```
…
/*********************************************************************/
// ENABLE_SECURITY will enable the device to encrypt and decrypt
// information transferred
/*********************************************************************/
#define ENABLE_SECURITY
…
```

Defining The Security Mode

- MiMAC-layer macro
- `ConfigMRFxxx.h`

ConfigMRF89XA.h

```
…
/*********************************************************************/
// SECURITY_LEVEL defines the security mode used in the application.
/*********************************************************************/
#define SECURITY_LEVEL SEC_LEVEL_CCM_16
…
```

- See `Security.h` & `MRF24J40.h` for available macro settings

Defining the Encryption Key

- MiMAC-layer macro
- `ConfigMRFxxx.h`
- Use any tool to calculate the key
- For 64-bit key designs, use key 00-07

ConfigMRF89XA.h

```
#define SECURITY_KEY_00 0x00
#define SECURITY_KEY_01 0x01
#define SECURITY_KEY_02 0x02
#define SECURITY_KEY_03 0x03
#define SECURITY_KEY_04 0x04
#define SECURITY_KEY_05 0x05
#define SECURITY_KEY_06 0x06
#define SECURITY_KEY_07 0x07
#define SECURITY_KEY_08 0x08
#define SECURITY_KEY_09 0x09
#define SECURITY_KEY_10 0x0a
#define SECURITY_KEY_11 0x0b
#define SECURITY_KEY_12 0x0c
#define SECURITY_KEY_13 0x0d
#define SECURITY_KEY_14 0x0e
#define SECURITY_KEY_15 0x0f
…
```

Securing the Payload

- **Security settings are applied to the payload via an application-layer TX function parameter:**

```
BOOL   MiApp_BroadcastPacket( BOOL SecEn );
```

```
BOOL   MiApp_UnicastConnection( BYTE    ConnectionIndex,
                                BOOL    SecEn);
```

```
BOOL   MiApp_UnicastAddress( BYTE     *DestinationAddress,
                             BOOL      PermanentAddr,
                             BOOL      SecEn);
```

Configure Security

LAB 2

Lab 2
Configure Security

Objective

- **Learn how to reconfigure an existing MiWi™ project to use a specific security setting.**
- **Learn how to use Wireless Development Studio (WDS) to verify that packets are encrypted.**

Lab 2
Configure Security

Procedure

- **Procedure**
  - Modify your project's security configuration settings to secure the communications:
    - **Enable security**
    - **Define the security mode**
    - **Define the security key**
    - **Encrypt the transmitted payload**
  - Build, re-run.
  - Verify that communications are secured.

Lab 2
Configure Security

Results

- **Easy to apply and use the MiWi™ security modes.**

Lab 2
Configure Security

Conclusions

- **Learned how to edit the MiMAC configuration for a project in order to configure the security mode.**

Agenda
- Microchip Wireless MAC (MiMAC)-

- MiMAC Layers
- MAC Layer Functions
- MiMAC Physical Layer
  - Lab 1 – Configure The Transceiver
- Device Types and Roles
- Network Topologies & Addressing
- Channel Scanning
- Network Creation & Association
- MiMAC Data Transfer Model
- MiMAC Frame Structure
- Security
  - Lab 2 – Configure Security
- **MiMAC APIs**

MiMAC Programming Interface

- **MiMAC API Details:**
  - Configuration File for Individual Transceivers
    - **`ConfigMRFxxx.h`**
  - 9 Function Calls from Microchip Proprietary Protocol Layers
    - **Configuration**
    - **TX/RX Operation**
    - **Special Functionalities (Sleep, Energy Scan)**
- **Reference:**
  - AN1283A, **`MiWi DE Help.chm`**

Latency vs. MiMAC Operations

- **Channel Assessment (seconds)**
  - Typically done once during initialization
  - Possible concern for frequency-agile applications
- **Sending Data (milliseconds) – depends on:**
  - SPI data rate
  - RF Data Rate
  - Length of packet
  - Application of security mode to the data
  - Acknowledge enabled?
  - CCA Enabled?
  - # of Re-transmissions

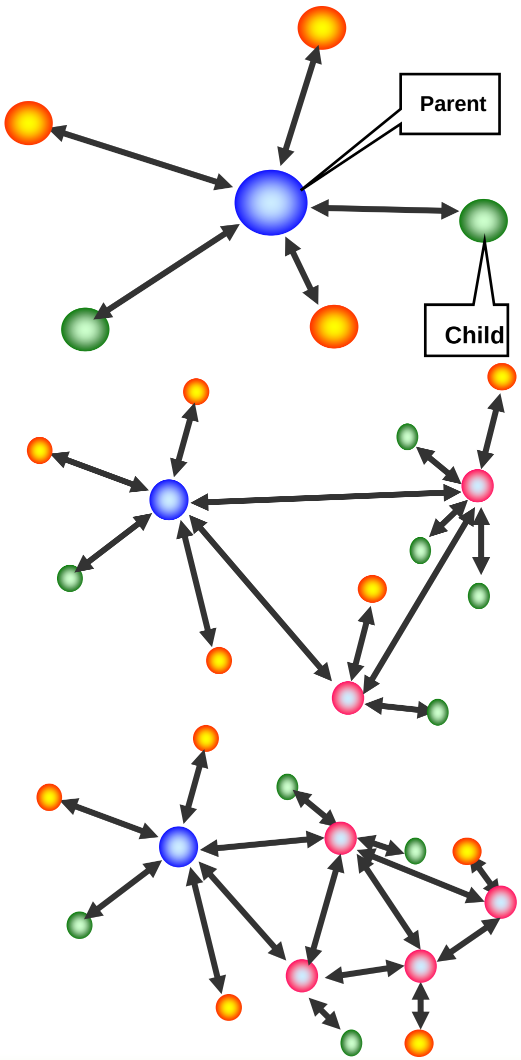
Overview of Microchip Wireless Protocols and Application Layer (MiApp)

Real Networking

- **Like most MAC standards, a "network" layer is not specified in MiMAC**
  - MiMAC Star/P2P topologies are single hop (radio range)
- **The Network & Upper Layers:**
  - Responsible for network formation,
  - End to end (source to destination) packet delivery
  - "Network" Addressing and assignment
  - Routing and route discovery
  - Adding and removing remote devices

Microchip Wireless Protocols

- **Microchip Current Provides Three Proprietary Networking Protocols:**
  - MiWi™ P2P Protocol
    - **MiMAC Star/P2P Topology**
    - **One Hop, No Routing**
  - MiWi Protocol
    - **Star/Small Cluster Tree Topology**
      - Adds FFDs as Proxies/Routers
    - **Up to 4 Hops. Mesh Routing**
  - MiWi PRO Protocol
    - **Star/Cluster Tree Topology**
      - Adds FFDs as Proxies/Routers
    - **Up to 65 Hops. Mesh Routing**



Selecting the Networking Protocol

- **Application-layer macro**
- **`ConfigApp.h`**

ConfigApp.h

```
…
//------------------------------------------------------------------------
// Definition of Protocol Stack. ONLY ONE PROTOCOL STACK CAN BE CHOSEN
//------------------------------------------------------------------------

#define PROTOCOL_P2P

//#define PROTOCOL_MIWI
//#define PROTOCOL_MIWI_PRO
…
```

MICROCHIP MASTERs 2014

Configuring the Networking Protocol

- **Networking-layer macros**
- **`ConfigP2P.h, ConfigMiWi.h`**

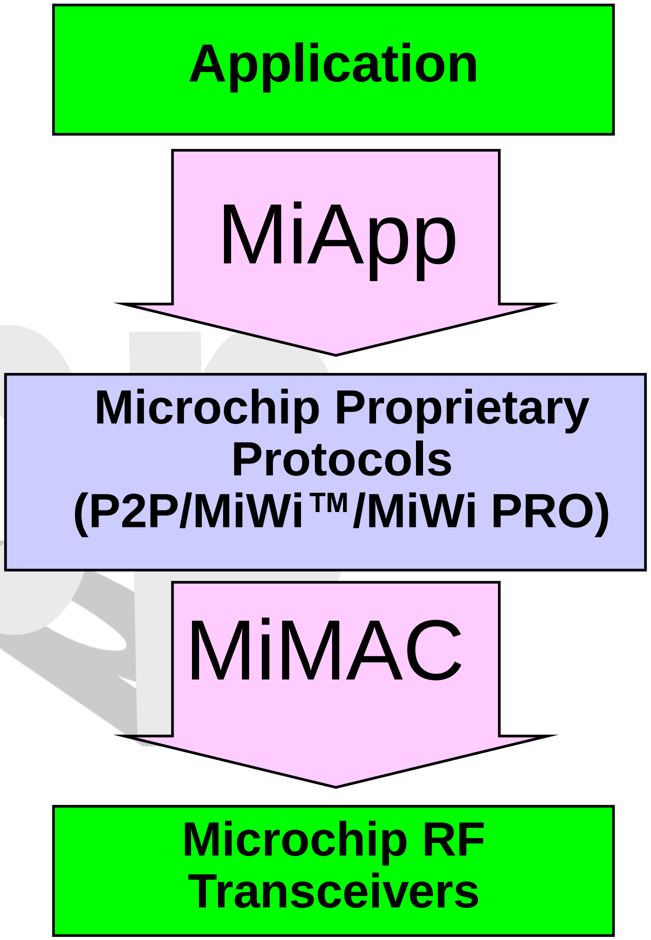
ConfigP2P.h

```
…
/*********************************************************************/
// 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
/*********************************************************************/
#define ACTIVE_SCAN_RESULT_SIZE     4

/*********************************************************************/
// INDIRECT_MESSAGE_SIZE defines the maximum number of packets that
// the device can store for the sleeping device(s)
/*********************************************************************/
#define INDIRECT_MESSAGE_SIZE   2
…
```

MiApp Definition

- **MiApp Defines Interfaces Between Application and Microchip Proprietary Protocols**
  - Configuration File
  - Programming Interfaces



MiApp APIs

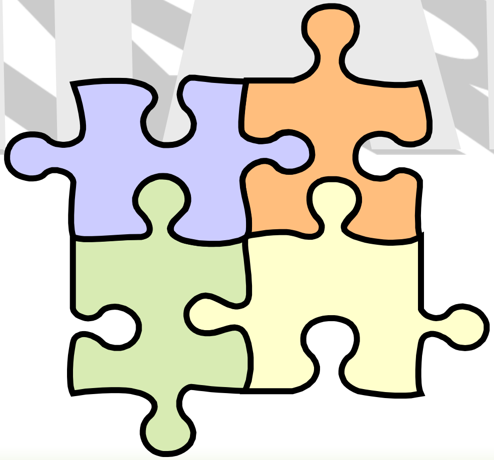
- **Four Categories of APIs**
  - Configuration
  - Connection
  - TX/RX Operation
  - Special Functionalities

MiApp APIs (Cont.)

- **Configuration**
  - Configuration File
  - Configuration Functions

```
BOOL    MiApp_ProtocolInit(BOOL bNetworkFreezer);
```

```
BOOL    MiApp_SetChannel(BYTE Channel);
```



MiApp APIs (Cont.)

- **Connecting**
  - Connection Mode
  - Establish Connection
  - Remove Connection

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

```
BOOL    MiApp_StartConnection( BYTE Mode,
                               BYTE ScanDuration,
                               DWORD ChannelMap);
```

```
BYTE    MiApp_SearchConnection( BYTE     ScanDuration,
                                DWORD    ChannelMap);
```

```
BYTE    MiApp_EstablishConnection( BYTE    ActiveScanIndex,
                                   BYTE    Mode);
```

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

MiApp APIs (Cont.)

- **TX/RX Operation**
  - Transmitting
    - **Use Global Buffer for TX**
    - **Broadcast**
    - **Unicast**

```
void    MiApp_FlushTx( void );
```

```
void    MiApp_WriteData( BYTE OneByteToSend );
```

```
BOOL    MiApp_BroadcastPacket( BOOL SecEn );
```

```
BOOL    MiApp_UnicastConnection( BYTE    ConnectionIndex,
                                 BOOL    SecEn);
```

```
BOOL    MiApp_UnicastAddress( BYTE      *DestinationAddress,
                              BOOL      PermanentAddr,
                              BOOL      SecEn);
```

MiApp APIs (Cont.)

- **Special Functionalities**
  - Transceiver Sleep/Wake up
  - Noise Detection
  - Frequency Hopping

```
BYTE    MiApp_TransceiverPowerState(BYTE Mode);
```

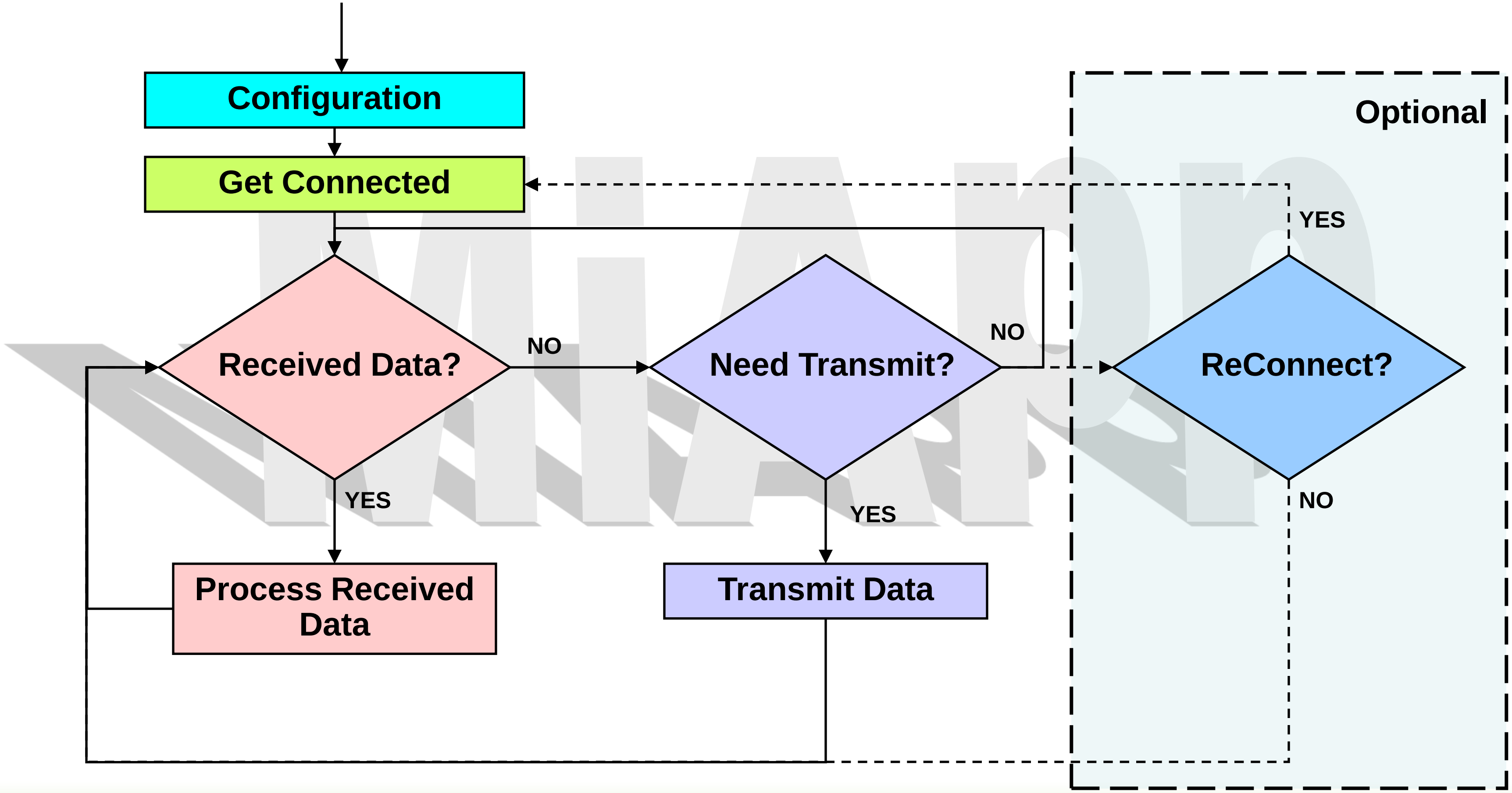
```
BYTE    MiApp_NoiseDetection(  DWORD    ChannelMap,
                               BYTE     ScanDuration,
                               BYTE     DetectionMode,
                               BYTE     *NoiseLevel);
```

```
BOOL    MiApp_InitChannelHopping(  DWORD ChannelMap );
```

```
BOOL MiApp_ResyncConnection( BYTE     ConnectionIndex,
                             DWORD    ChannelMap);
```

MiApp

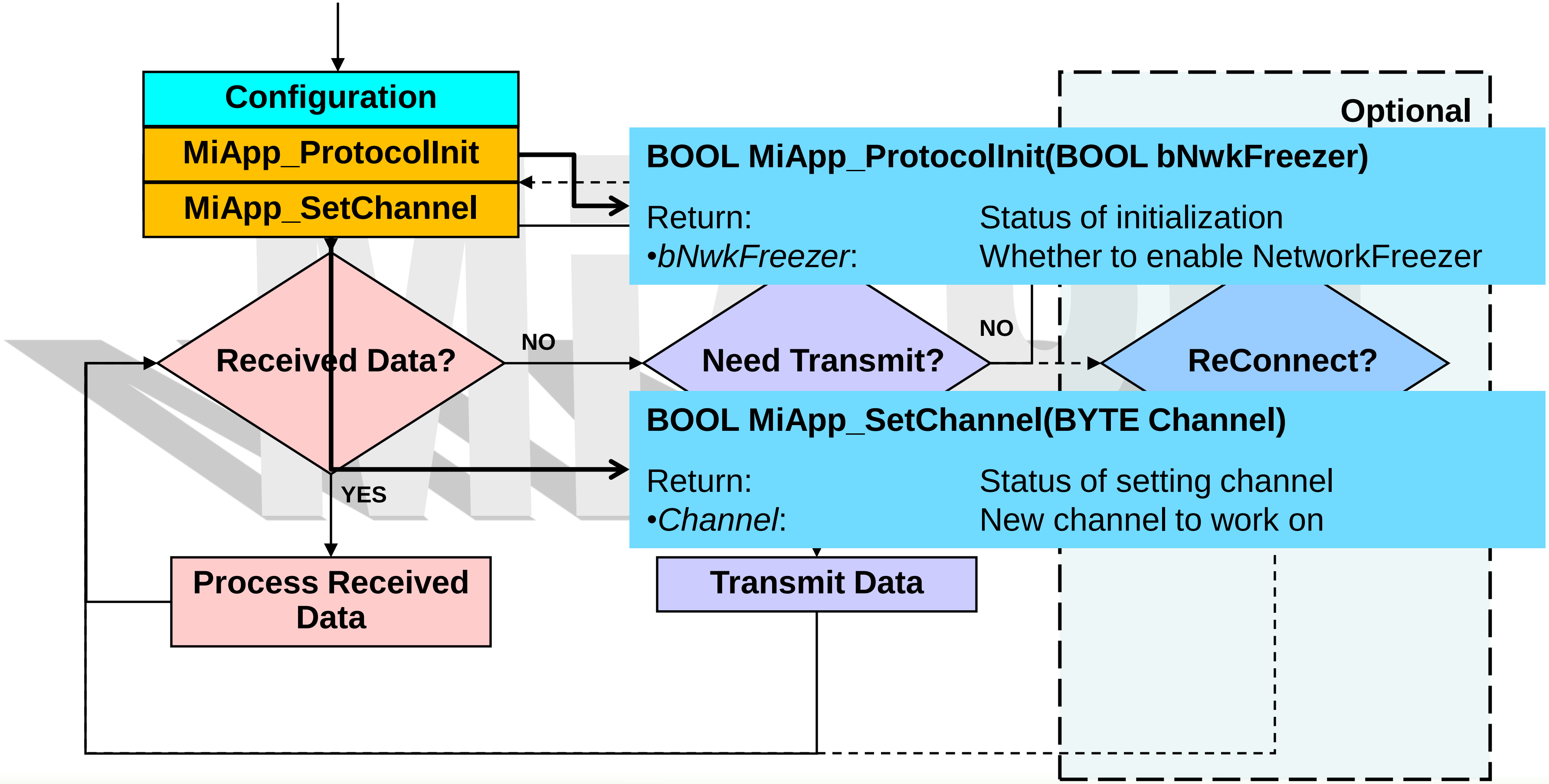
- **Typical Wireless Applications:**



MICROCHIP
MASTERs 2014

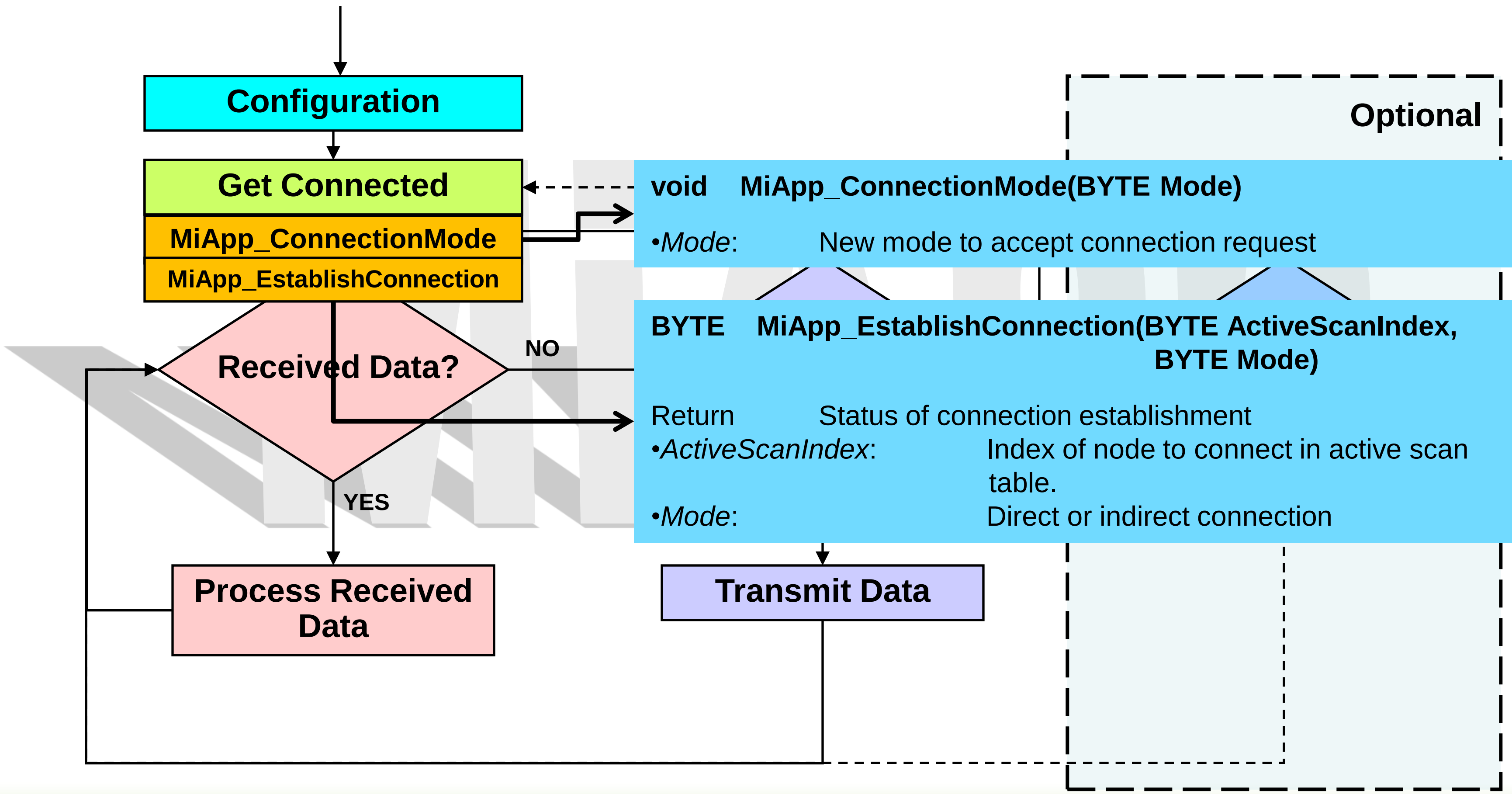
MiApp

- **Typical Wireless Applications:**



MiApp

- **Typical Wireless Applications:**



void MiApp_ConnectionMode(BYTE Mode)

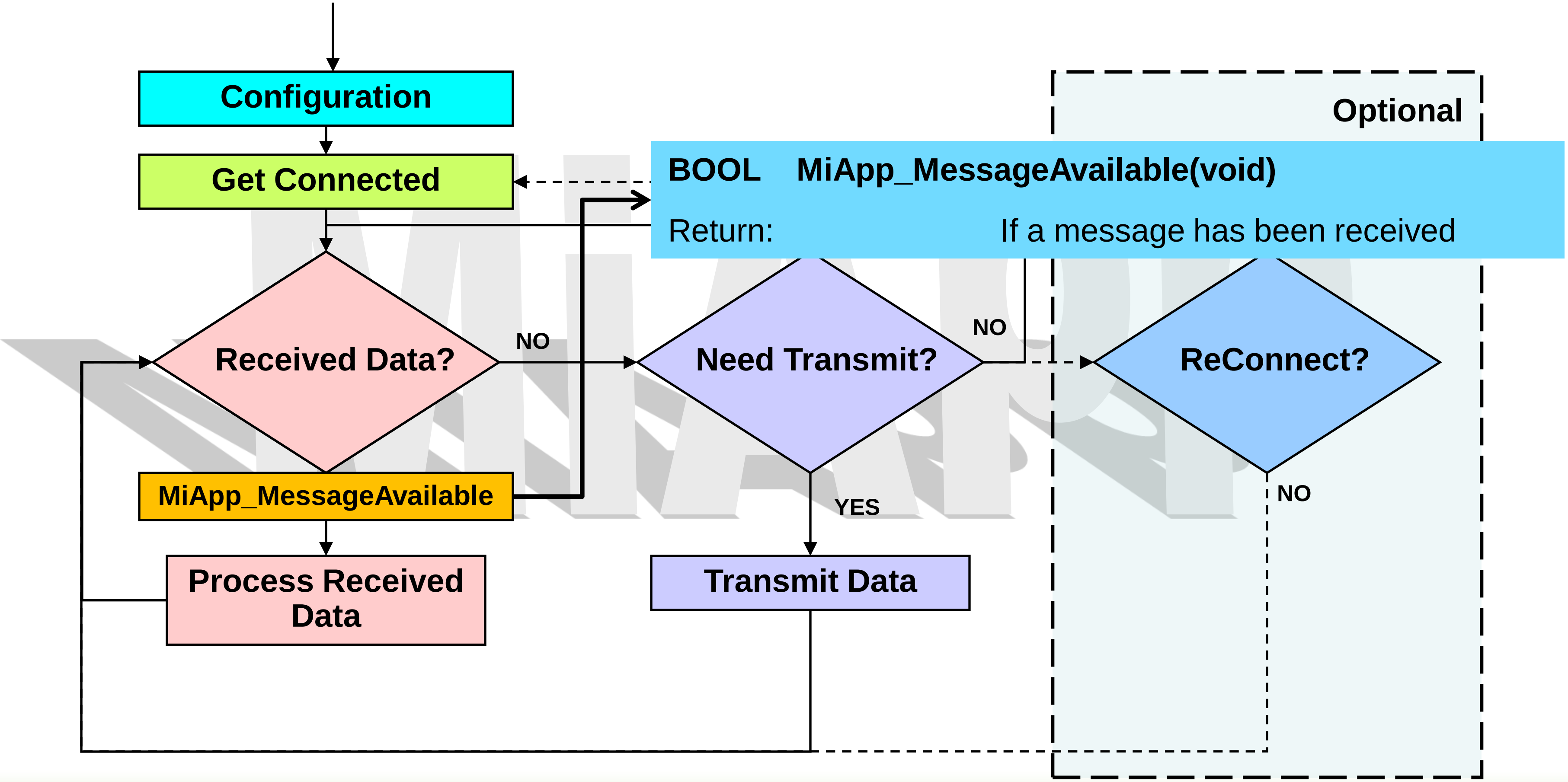
•*Mode*: New mode to accept connection request

BYTE MiApp_EstablishConnection(BYTE ActiveScanIndex, BYTE Mode)

Return Status of connection establishment
•*ActiveScanIndex*: Index of node to connect in active scan table.
•*Mode*: Direct or indirect connection

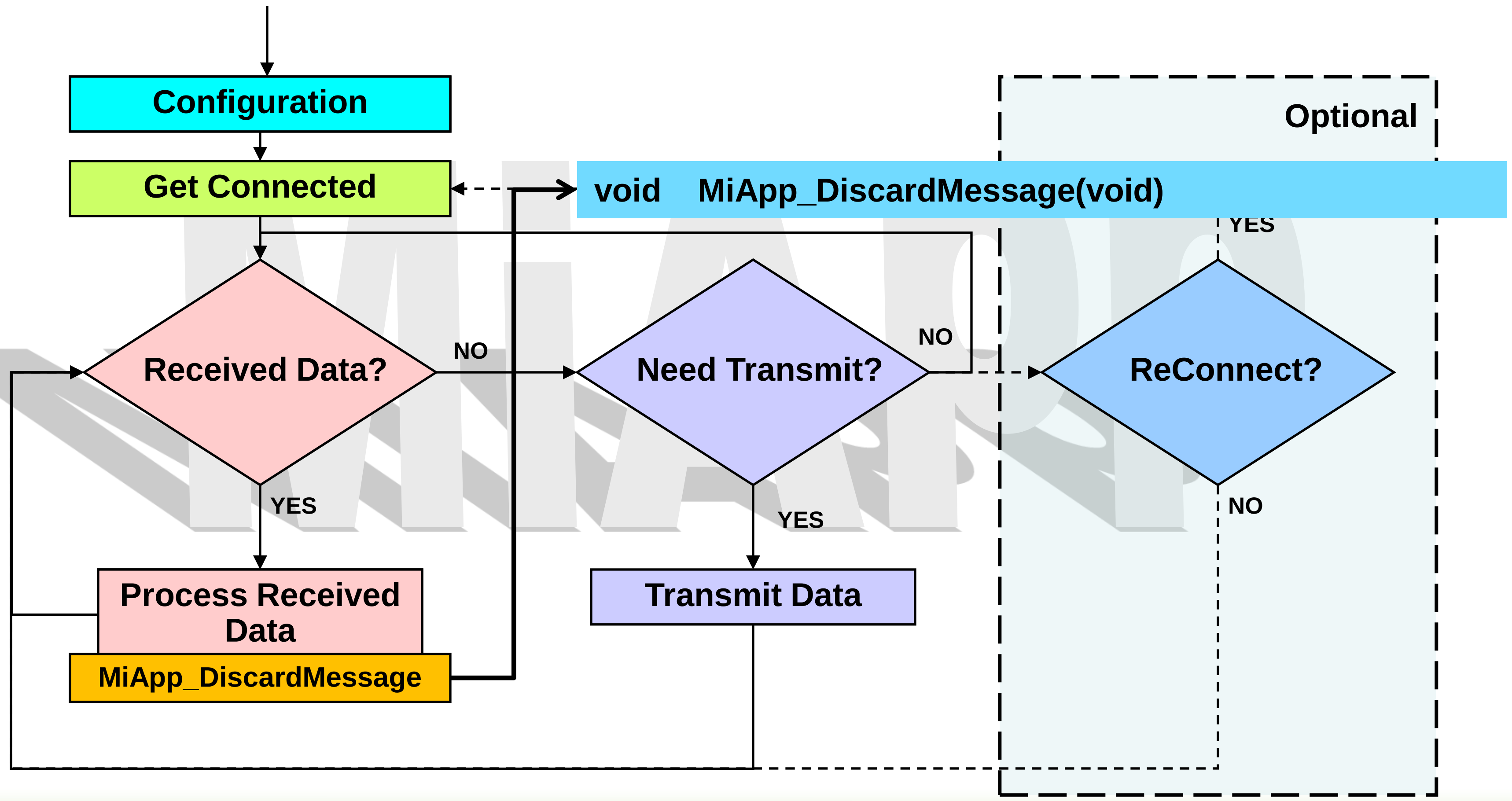
MiApp

- **Typical Wireless Applications:**



MiApp

- **Typical Wireless Applications:**



MiApp

- **Typical W**

BOOL MiApp_UnicastAddress(BYTE *Addr, BOOL PermAddr, BOOL SecEn)

Return:	Transmission status
Addr:	Pointer to the destination address
PermAddr:	Address the destination as permanent
SecEn:	Whether to encrypt transmission data

BOOL MiApp_UnicastConnection(BYTE ConnIndex, BOOL SecEn)

Return:	Transmission status
ConnectionIndex:	The index of the dest addr in connection table
SecEn:	Whether to encrypt transmission data

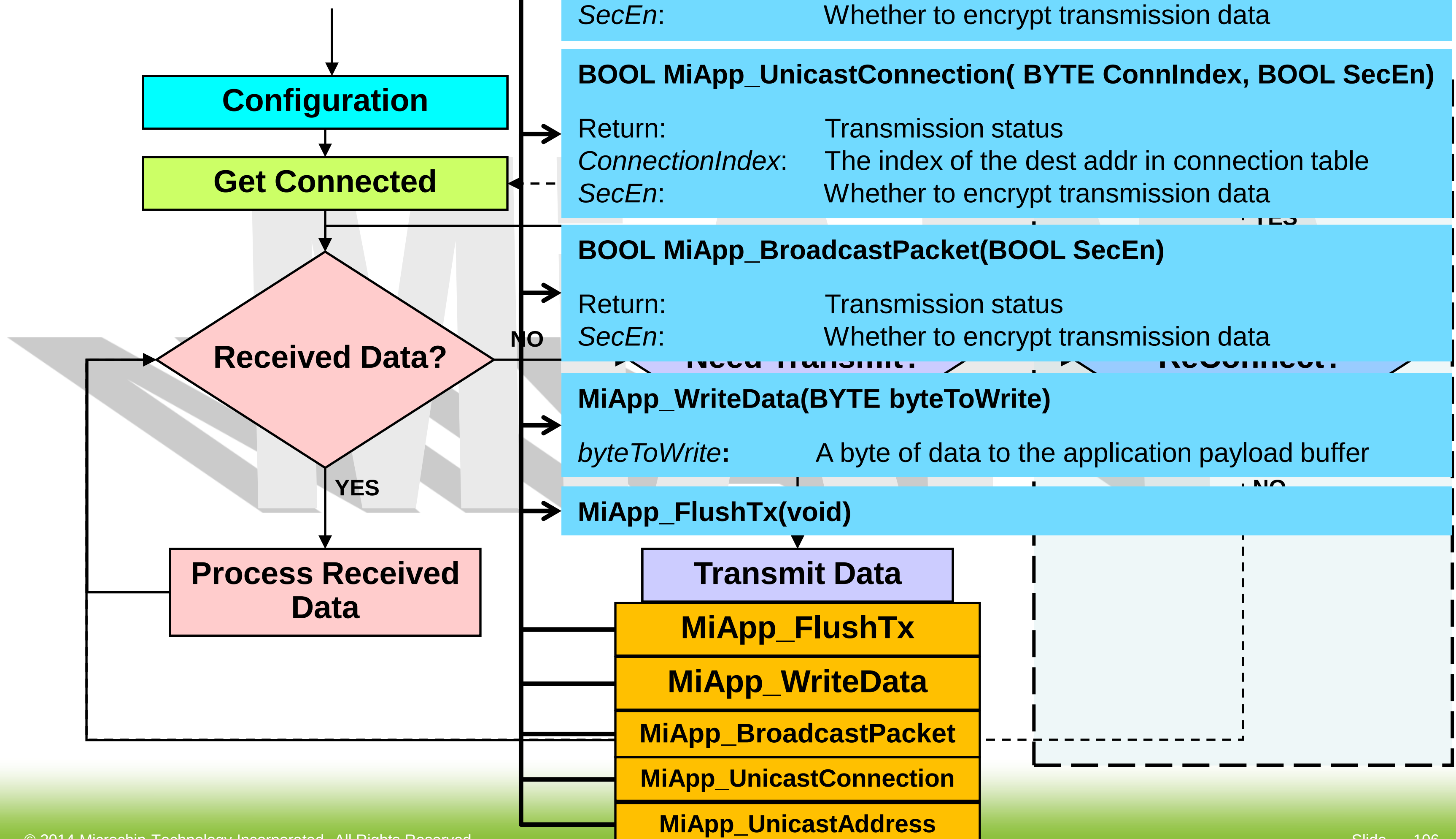
BOOL MiApp_BroadcastPacket(BOOL SecEn)

Return:	Transmission status
SecEn:	Whether to encrypt transmission data

MiApp_WriteData(BYTE byteToWrite)

byteToWrite:	A byte of data to the application payload buffer

MiApp_FlushTx(void)



Typical Wireless Application

```
void main(void)
{
1       // Configuration
        MiApp_ProtocolInit(FALSE);
2       MiApp_SetChannel(25);

        // Get Connected
3       MiApp_ConnectionMode(ENABLE_ALL_CONN);
4       MiApp_EstablishConnection(0xFF, CONN_MODE_DIRECT);

5       while(1)
        {
            // Receive Data
6           if( MiApp_MessageAvailable() )
            {
7               LED = RxMessage.PayLoad[0];
8               MiApp_DiscardMessage();
            }
```

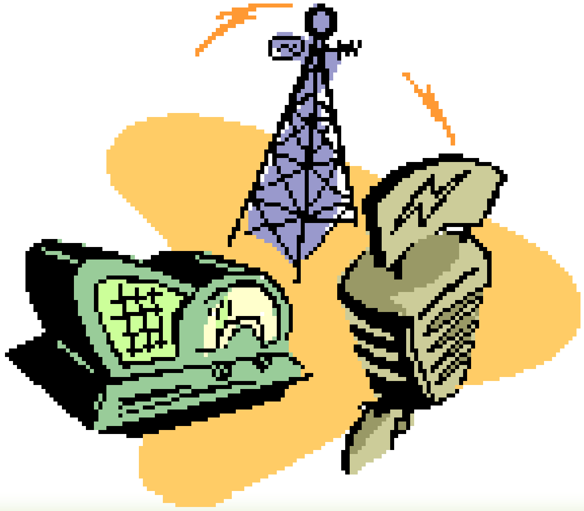
Simple Application

9 else if (ButtonPressed())
{
// Transmit LIGHT_ON to Peer
10 MiApp_FlushTx();
11 MiApp_WriteData(LIGHT_ON);
12 MiApp_UnicastConnection(0, TRUE);
}
}
}

MiApp

MiWi™ PRO: Expanding for Greater Coverage

- Enhanced Broadcast
- Enhanced Network Routing
- Enhanced Frequency Agility



MiWi™ PRO: Expanding for Greater Coverage

- **Support Multicast in Network Layer**
  - 0xFFFF All Devices
  - 0xFFFE All Non-Sleeping Devices
  - 0xFFFD All Coordinators
- **Broadcast Tracking**
  - Do not Rebroadcast Second Time
  - Eliminate Unnecessary Messages to Pollute the Spectrum

MiWi™ PRO: Expanding for Greater Coverage

- **Short Address Assignment**
  - Parent's number
    - **Reserved for Coordinator**
  - RxOffWhenIdle
    - **1 bit**
  - Child's Number
    - **7 bits**

0	**0**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**

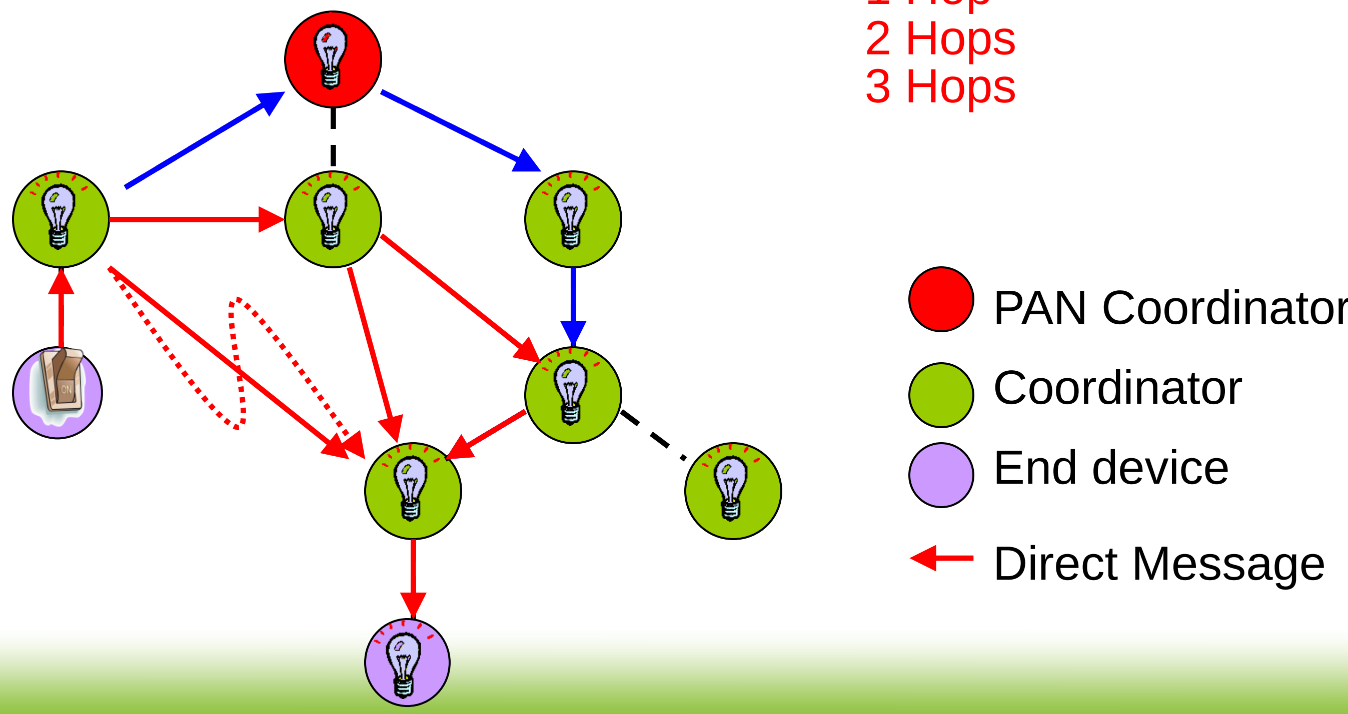
Reserved **Parent's Number** **Child's Number**

RxOffWhenIdel

MiWi™ PRO: Expanding for Greater Coverage

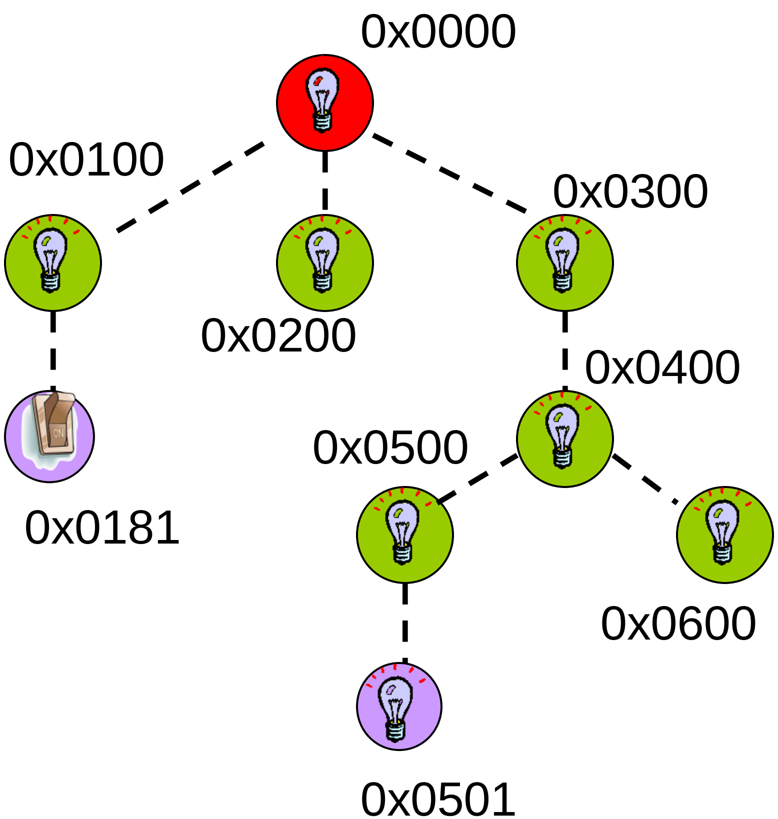
- **MiWi PRO Routing Mechanism**
  - Old Rules: Only Coordinator is able to Route Messages; Route to Destination's Parent for End Device; Route to the Neighbor Coordinator; Route to Parent/Child
  - New Rules:
    - **Find Hops for Tree Routing N_{tree}** 4 Hops
    - **Check if N hops can reach Destination with Mesh Routing (N Starts with1). If Cannot Reach, Increase N, until N_{tree}**
    - **Use Tree Routing**

1 Hop
2 Hops
3 Hops



MiWi™ PRO: Expanding for Greater Coverage

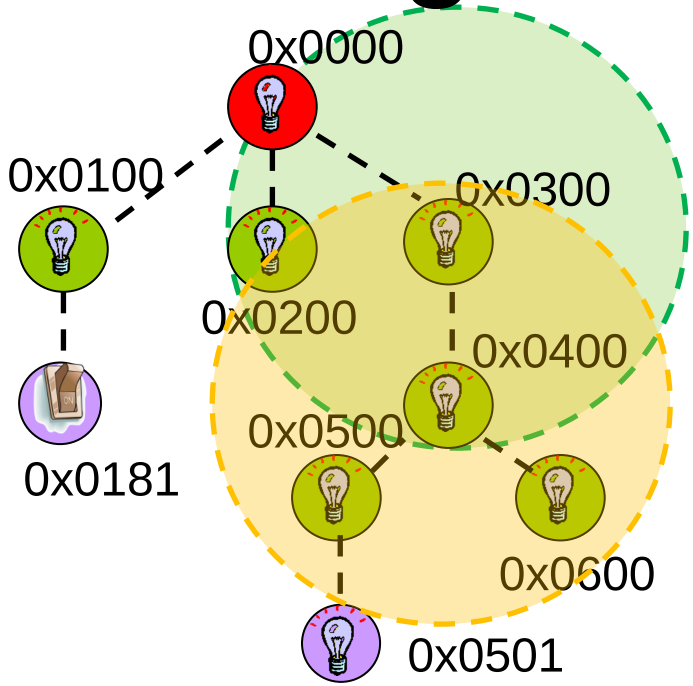
- **Family Info**



Higher Byte of Coordinator Address (Implied by Table Index)	Higher Byte of Parent Address for Coordinator	Description
00	0x00	PAN Coordinator 0x0000 has no Parent, point its parent to itself
01	0x00	Coordinator 0x0100's Parent is 0x0000
02	0x00	Coordinator 0x0200's Parent is 0x0000
03	0x00	Coordinator 0x0300's Parent is 0x0000
04	0x03	Coordinator 0x0400's Parent is 0x0300
05	0x04	Coordinator 0x0500's Parent is 0x0400
06	0x04	Coordinator 0x0600's Parent is 0x0400
07	0xFF	Coordinator 0x0700 does not exist yet
……	……	
n	0xFF	Coordinator 0xn00 does not exist yet

MiWi™ PRO: Expanding for Greater Coverage

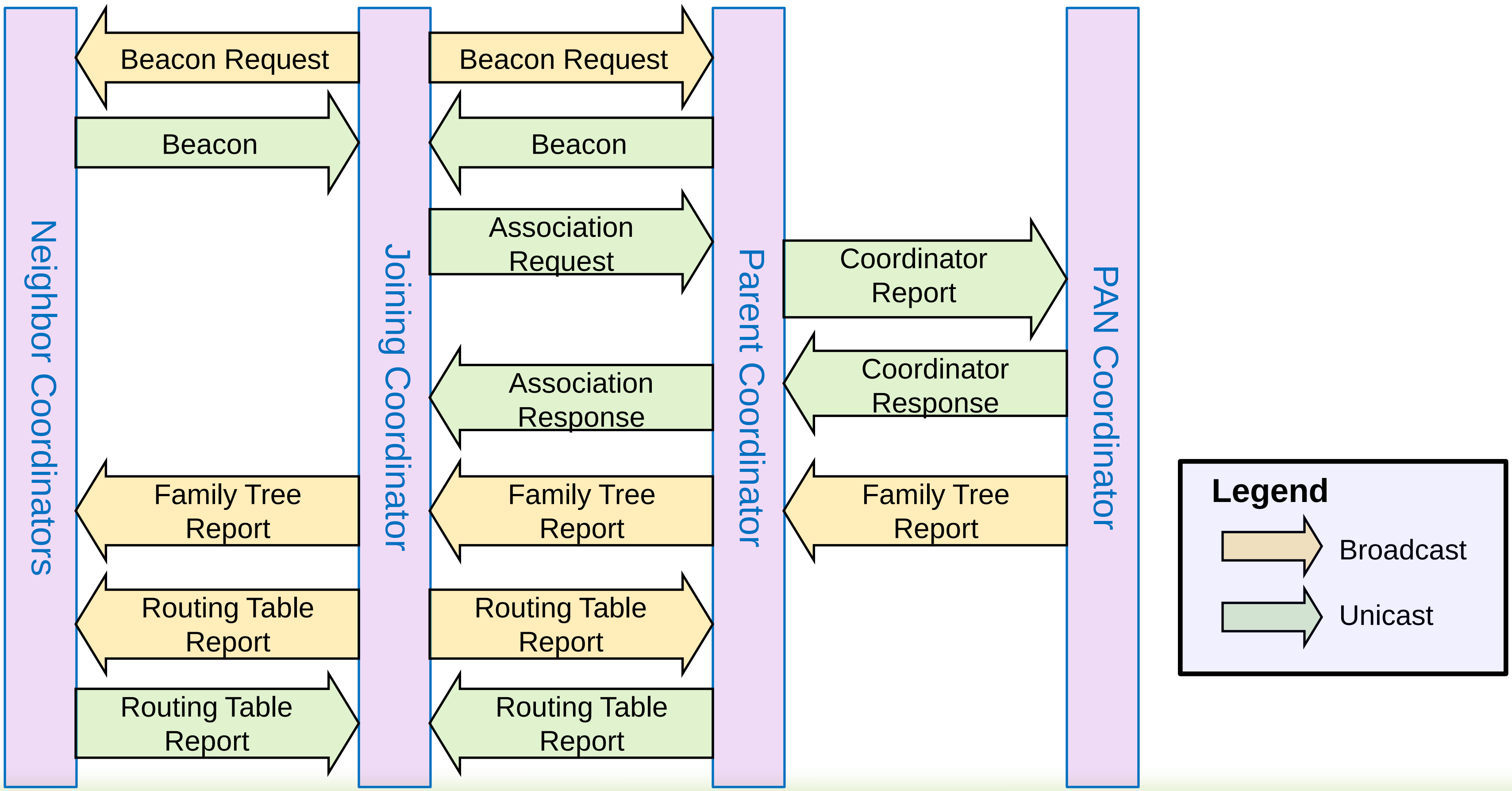
- **Neighbor Table**
  - Bit Map to Indicate Known Coordinators
- **Neighbor's Neighbor Table**
  - Bit Map from Each Neighbor to Indicate Their Known Coordinators



Neighbor Table of Coordinator 0x0400
0b01101100 (I Know Coordinator 0200, 0300, 0500 and 0600)
Neighbor's Neighbor Table of Coordinator 0x0300 on 0x0400
0b00010101 (My Neighbor 0x0300 Know Coordinator 0000, 0200 and 0400)

MiWi™ PRO: Expanding for Greater Coverage

- **MiWi PRO Joining Procedure**
  - **Network Freezer Mandatory**

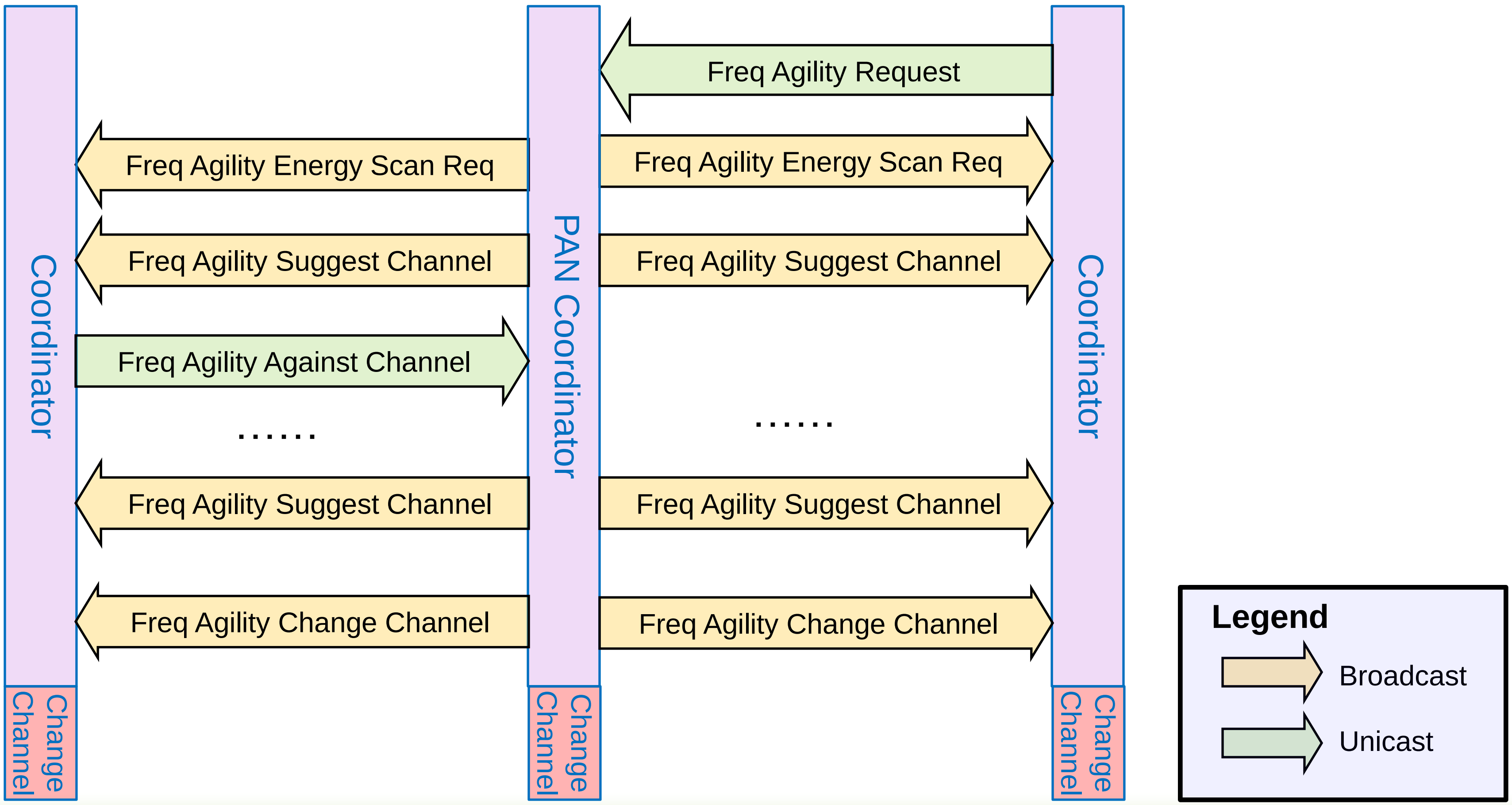


MiWi™ PRO: Expanding for Greater Coverage

- **MiWi P2P and MiWi Protocol Depends on PAN Coordinator to do Frequency Agility**
  - It's Simple
  - It's Doable: The Network Covers a Small Area
- **MiWi PRO Must Handle Frequency Agility Differently**
  - Noise Distribution of PAN Coordinator may NOT Represent the Whole Network
  - Voting System Developed for MiWi PRO Frequency Agility

MiWi™ PRO: Expanding for Greater Coverage

- **MiWi PRO Frequency Agility**



Summary and References

Summary

- We learned about the MiWi™ DE transceiver options available and how to configure them.
- We learned about the services provided to the upper layers by MiMAC and how to configure them.
- We reconfigured and debugged a simple wireless application using MiWi™ DE.
- We learned the basics of the MiWi™ PRO addressing and functionality

References

- **MiWi™ Application Design Page:**

  www.microchip.com/miwi
- **Links to:**
  - Microchip Application Libraries
    - **MiWi Stack source code and demo projects**
  - Development Platforms & Transceiver Cards:
    - **Explorer 16 (PIC24/PIC32)**
    - **PIC18 Explorer (PIC18)**
    - **8-Bit Wireless Development Kit (PIC18)**
  - Application Notes

References

- **Relevant Application Notes**
  - MiMAC Application Note (AN1283)
  - MiApp Application Note (AN1284)
  - MiWi™ P2P Application Note (AN1204)
  - MiWi Application Note (AN1066)
  - MiWi PRO Application Note (AN1371)

References

- **IEEE 802.15.4™-2003**

  http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf

- **IEEE OUI**

  https://standards.ieee.org/regauth/oui/forms/OUI-form.shtml

- **MiWi™ Protocol**

  http://www.microchip.com/MiWi

References

- **Security**
  - AN1283a
  - Schneier, B. “ Applied Cryptography” 2nd Ed.
    ISBN: 978-0-471-11709-4
  - Ferguson, N. et al. “Cryptography Engineering – Design Principles & Practical Applications”
    ISBN: 978-0-470-47424-2
  - Paar, C. et al. “Understanding Cryptography – A Textbook for Students and Practitioners”
    ISBN: 978-3-642-04100-6

Thank You!

MASTERs 2014

LEGAL NOTICE

SOFTWARE:

You may use Microchip software exclusively with Microchip products. Further, use of Microchip software is subject to the copyright notices, disclaimers, and any license terms accompanying such software, whether set forth at the install of each program or posted in a header or text file.

Notwithstanding the above, certain components of software offered by Microchip and 3rd parties may be covered by "open source" software licenses – which include licenses that require that the distributor make the software available in source code format. To the extent required by such open source software licenses, the terms of such license will govern.

NOTICE & DISCLAIMER:

These materials and accompanying information (including, for example, any software, and references to 3rd party companies and 3rd party websites) are for informational purposes only and provided "AS IS." Microchip assumes no responsibility for statements made by 3rd party companies, or materials or information that such 3rd parties may provide.

MICROCHIP DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, INCLUDING ANY IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY DIRECT OR INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND RELATED TO THESE MATERIALS OR ACCOMPANYING INFORMATION PROVIDED TO YOU BY MICROCHIP OR OTHER THIRD PARTIES, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBLITY OF SUCH DAMAGES OR THE DAMAGES ARE FORESEEABLE. PLEASE BE AWARE THAT IMPLEMENTATION OF INTELLECTUAL PROPERTY PRESENTED HERE MAY REQUIRE A LICENSE FROM THIRD PARTIES.

TRADEMARKS:

The Microchip name and logo, the Microchip logo, dsPIC, FlashFlex, flexPWR, JukeBlox, KeeLoq, KeeLoq logo, Kleer, LANCheck, MediaLB, MOST, MOST logo, MPLAB, OptoLyzer, PIC, PICSTART, PIC^{32} logo, RightTouch, SpyNIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

The Embedded Control Solutions Company and mTouch are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, ECAN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, KleerNet, KleerNet logo, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, RightTouch logo, REAL ICE, SQI, Serial Quad I/O, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

GestIC is a registered trademarks of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.