INTRODUCTION

Most of the embedded applications require real-time communications to support their application environment. Wired standards of communication, such as RS232, RN422, RS485 or Ethernet are not easily implemented due to the infrastructure support required in the end application. Wi-Fi® and Bluetooth® have emerged as the standards of choice for connecting local embedded applications to the cloud through a router, smartphone or tablet (i.e. a service call).

Wi-Fi, based on IEEE 802.11 b/g/n/ac, is a standard feature in laptops, smart phones, smart machines and many other applications. Wi-Fi provides substantial bandwidth for data transfer; however, it is heavy in protocol stack and power consumption. Recent efforts to reduce power consumption in Wi-Fi, yielded low-energy Wi-Fi, suitable for man-to-man or man-to-machine embedded application designs.

Bluetooth is characterized by easy, temporary connectivity to smartphones and tablets, and is supported in many Android® and iOS® applications today. It provides a convenient cable replacement for applications, such as audio streaming and data synchronization between devices. Initially Bluetooth supported 1 Mbps data transfer (Bluetooth 1.2), and later the data transfer rate has increased to 3 Mbps with the Enhanced Data Rate version (Bluetooth 2.1 + EDR), and further advanced to a high-speed version (Bluetooth 3.0 + HS) to support large file transfers.

Many applications only need simple command and control, or a quick status from a sensor. Therefore, Bluetooth Low Energy (BLE) has evolved to support these low-duty cycle applications.

BLUETOOTH SMART COMMUNICATION

Bluetooth Low Energy, also known as Bluetooth Smart, is a hallmark in the Bluetooth 4.0 specification. BLE is intended for energy-constrained applications, such as sensors or disposable devices. BLE is intended for low-duty cycle devices that support low-data throughput and can operate for a longer duration compared to other protocols from a coin cell battery. Key benefits in implementing the technology are inexpensive silicon, light MCU processing requirements, and reduced memory is also quite suitable for applications related to the body area network (BAN) which represents a ‘connectivity bubble’ that moves along with the individual’s network.

BLE operates in the same spectrum band (i.e. 2.400 GHz to 2.4835 GHz ISM band) as Classic Bluetooth technology, but uses a different set of channels. Classic Bluetooth has 79 channels, each of 1 MHz wide, whereas Bluetooth Smart has 40 channels, each of 2 MHz wide. Within a channel, data is transmitted using Gaussian Frequency Shift Keying (GFSK) modulation technique, which is similar to Classic Bluetooth’s FSK modulation. The maximum over-the-air bit rate is 1 Mbps, and the maximum transmit power is 10 mW. For additional information related to Bluetooth and its specifications, refer to “Bluetooth Core Specification V4.0” from the following web site: http://www.bluetooth.org.

The Microchip RN4020 is a fully-certified Bluetooth Version 4.1 module for adding a low-power wireless capability to applications and products. The surface mounted module has the complete Bluetooth stack on-board and is controlled through simple ASCII commands over the UART interface. The RN4020 also includes most of the Bluetooth SIG standard profiles and Microchip’s Low-energy Data Profile (MLDP) for user specific serial data transfer over BLE. Developers can utilize the scripting feature to enable standalone operation without a Host microcontroller or processor. The RN4020 can be remotely controlled or updated over-the-air by another module over a secure connection, or can be controlled or updated through its UART interface.
Figure 1 illustrates the RN4020 module mounted on a PICtail™ board.

FIGURE 1: RN4020 MODULE MOUNTED ON A PICTAIL™ BOARD

This application note provides the following:
- Framework for any of the user application platform using RN4020 module and PIC24FJ series of microcontrollers.
- Specifically an interface of RN4020 module with PIC24FJ128GA010 microcontroller.
- Start up or reference code to command RN4020 module through PIC® microcontroller.
- Initial procedures or techniques for interfacing a PIC microcontroller and BLE module.

Note: This application note is not intended to provide a complete understanding of the Bluetooth technology principles or using the commands related to the RN4020 module.
The primary purpose of this application note is to help users or developers to have a quick understanding of the interface requirements and process of communication using commands between the RN4020 modules and the PIC24 (16-bit) microcontroller over UART interface. Figure 2 illustrates the PIC24 MCU interface with the RN4020 module. The inputs or commands are given through the switches available on the Explorer 16 Development board, and the status is monitored through on-board LCDs and LEDs.

**FIGURE 2: SMART COMMUNICATION APPLICATION DIAGRAM**

**Application Demo Requirements**

This section describes the hardware and software/utilities required for the demo setup.

**HARDWARE REQUIREMENTS**

The following hardwares are used for the demo application:

- Two BLE RN4020 PICtail™/PICtail Plus daughter boards
- Two Explorer 16 Development boards
- Any one of the following Microchip development tools for programming and debugging: MPLAB® REAL ICE™, MPLAB® ICD 3, or PICkit™ 3
- Two Power supplies: 9V/0.75A or equivalent battery packs

The hardware interface of the RN4020 module with any of the PIC microcontroller is known as wireless node, as illustrated in Figure 3. The wireless nodes can be realized using a combination of the development boards and the RN4020 daughter boards.

**SOFTWARE/UTILITY REQUIREMENTS**

The demo application that is based on the RN4020 module is used to showcase communication between the BLE-based embedded nodes. In the demo application, the nodes can emulate a sensor, actuator systems, fitness device, healthcare gadget and so on.

Alternatively terminal emulator programs (for example, TeraTerm for Windows OS and CoolTerm for MAC OS) can be used to control and monitor the RN4020 module. This is to do some independent tests using the RN4020 PICtail card.

The application or demo source code related to this application note is available as MPLAB workspace and can be downloaded from the Microchip web site: http://www.microchip.com. The code is compiled using the Microchip XC16 compiler through MPLABX IDE. The demo uses commands which are initiated by the switches on one of the Explorer 16 Development board and the result is displayed on the another Explorer 16 Development board’s LCD.
Figure 3 illustrates pin to pin configuration used in the application demo code between the PIC24FJ128GA010 microcontroller and the RN4020 BLE module.

**FIGURE 3: MICROCONTROLLER TO BLE MODULE INTERFACE DIAGRAM**

HARDWARE DEMO SETUP
The RN4020 demo requires two wireless nodes. The demo setup is done by using two Explorer 16 Development boards with identical RN4020 module mounted on the PICtail board. Thus, two identical RN4020 module-based wireless nodes are used for this application demonstration. For more information on the RN4020 module, refer to the Microchip web site: http://www.microchip.com.

Explorer 16 Development Board and RN4020 Module Connections
The RN4020 module-based PICtail daughter board’s 30-pin PCB-edge connector (J3) is used to connect the Explorer 16 Development board’s PICtail plus connector. This connection supplies 3.3V power, two/four wire UART, Reset, wake and interrupt connections to the RN4020 module from the Microcontroller. Figure 4 illustrates the plug-in arrangement between the Explorer 16 Development board and the RN4020 module. For more information on using and programming the Explorer 16 Development board with RN4020 module, refer to the "RN4020 PICtail™/PICtail Plus Daughter Board User’s Guide" (DS50002265).
Figure 4 illustrates the RN4020 module is mounted on the Explorer 16 Development board.

**FIGURE 4: RN4020 MODULE MOUNTED ON EXPLORER 16 DEVELOPMENT BOARD**

---

**GETTING STARTED**

To set up the RN4020 BLE module as wireless node, perform the following actions:

1. Insert the RN4020 PICtail card into the Explorer 16 Development board’s 30-pin card edge connector. The Explorer 16 Development board would be a base board for the RN4020 PICtail Daughter card.

2. Display on the LCD of the Explorer 16 Development board can be used for configuring or monitoring the wireless terminals.

3. Plug-in the 9V power supply to the base.

4. Connect the programmer or debugger (MPLAB ICD3/PICKIT3) to the Explorer 16 Development board.

5. Open the application demo source code available on the web site.

6. In Release mode, compile the two source codes related to the application demo.

7. The generated .hex files can then be programmed into the two wireless Nodes, A and B, using the MPLAB ICD3 or any other programmer available with the user.

Note: Users can use already generated .hex files which are available in the source code WinZip file.

---

For more information on programming and debugging with MPLAB ICD 3, refer to “MPLAB® ICD 3 User’s Guide for MPLAB X” (DS50002081) and for Explorer 16 Development Board, refer to “Explorer 16 Development Board User’s Guide” (DS51589).

**Appendix A** through **Appendix C** provide details of the source code, related files with description, and call graphs of the main() functions associated to Central and Peripheral nodes. However, users can further generate call graphs related to specific functions of the source code for their understanding.
RN4020 Demo Application

The execution of the RN4020 demo application involves the following two steps:

1. Configuring one node as Central and another node as Peripheral.
2. Switch ON or OFF to toggle LEDs (D7 and D8) on the Node B when switches (S3 and S6) on the Node A are pressed, and vice versa.

Configuring the Node A as Central

Using the demo code, the first RN4020 module (referred to as module A) is configured as Central (Node A). The following commands are used to configure the device:

1. Pull `WAKE_SW` high to enter command mode. (On the daughter board, this is the default state).
2. Open a terminal emulator that connects the serial port of the module A with the following parameters:
   - Baud rate: 115200
   - Data bits: 8
   - Parity: none
   - Stop bits: 1
   - Flow control: hardware
3. `SF, 1` // factory reset.
4. `SR, 92000000` // set device as central, support MLDP and enable UART flow control.
5. `R, 1` // reboot to make the changes effective.

Figure 5 illustrates the BLE Central (Node A) flowchart.
Figure 6 and Figure 7 illustrate the configuration and scanning of Central (Node A).

FIGURE 6: BLE CENTRAL NODE IN CONFIGURATION MODE
FIGURE 7: BLE CENTRAL NODE IN SCANNING MODE

Figure 8 illustrates the BLE Central node demo showing switches for scroll and select. Figure 9 illustrates the BLE Central node in the connected mode.

FIGURE 8: BLE CENTRAL NODE IN SCROLL AND SELECT MODE
FIGURE 9: BLE CENTRAL NODE IN CONNECTED MODE
Configuring the Node B as Peripheral

Using the demo code, the second RN4020 module (referred to as module B) is configured as Peripheral (Node B). The following commands are issued to configure the device:

1. Pull `WAKE_SW` high to enter Command mode. (On the daughter board, this is the default state).
2. Open a terminal emulator that connects the serial port of the module B with the following parameters:
   - Baud rate: 115200
   - Data bits: 8
   - Parity: none
   - Stop bits: 1
   - Flow control: hardware
3. `SF, 1` // factory reset.
4. `SR, 32000000` // set device as peripheral with automatic advertisement, and support for MLDP and flow control features.
5. `R, 1` // Reboot the device to make the changes effective.

Figure 10 illustrates the BLE Peripheral (Node B) flowchart.
Figure 11 and Figure 12 illustrate the configuration and auto advertising of the BLE Peripheral (Node B).

**FIGURE 11:** BLE PERIPHERAL NODE IN CONFIGURATION MODE

**FIGURE 12:** BLE PERIPHERAL NODE IN AUTO ADVERTISING MODE
Figure 13 illustrates the BLE Peripheral node when connected.

FIGURE 13: BLE PERIPHERAL NODE IN CONNECTED MODE
Connecting Two Devices

When module B is powered up, it automatically starts advertising because the auto advertisement feature is enabled using the "SR" command, and then the module A discovers the module B using the "F" command:

F // Start scan

The scan result displays three elements: MAC address, MAC address type, and device name (for example, 00035B0358E6, 0, MCHP-LE, -50).

Input the "X" command followed by an "E" command to stop scanning and establishing the connection:

X // Stop scanning
E, 0, 00035B0358E6 // Try to establish connection with the device of public MAC address, 0x00035B0358E6.

Figure 14 illustrates the application demo diagram using the terminal emulator interface.
MLDP Mode Details

MLDP mode is entered by setting the CMD/MLDP pin high, all data from the UART is sent to the peer device as a data stream. To exit MLDP mode, the CMD/MLDP pin must be set low so that the RN4020 module is returned to Command mode by outputting “CMD” to the UART. The CMD/MLDP pin (pin 8) is used to control the RN4020 module when an MLDP serial data service is used. For more information on MLDP Commands, refer to “RN4020 Bluetooth Low Energy Module User’s Guide” (DS70005191).

After the access and characteristics in public services are verified, the MLDP service can be started. The MLDP service is built on top of the private service, but acts transparently by routing binary data read and write to a private characteristic through the UART. To use the MLDP service between two RN4020 devices, both the devices must enable MLDP with the “SR” command. The MLDP mode can only be started when two RN4020 modules are MLDP-enabled and are connected to each other. To start MLDP mode, perform the following actions:

1. Assert the CMD/MLDP pin to be high. The RN4020 module will acknowledge with “MLDP” string to indicate the start of the MLDP mode.
2. Once in the MLDP mode, any data from the UART will be sent to the peer device.
3. While receiving the MLDP data from the peer, if the AUTO_MLDP_DISABLE feature is disabled, the RN4020 module will automatically enter the MLDP mode; otherwise, all data will be ignored until the CMD/MLDP pin is set high to enter MLDP mode.
4. From the module A, assert the CMD/MLDP pin to be high and wait until “MLDP” is output to the UART. Provided module B shows “MLDP”, anything typed on the UART of the module A will be displayed on the UART of the module B, and vice versa.
5. Set the CMD/MLDP pin to be low on module B (WAKE_HW and CMD/MLDP pins have weak pull down resistors hence they will stay low if not pulled high).
6. On module A, the status change will be notified to the Host. However, module A is currently in MLDP mode and only output MLDP data is sent to the UART. Instead, PIO2 will be set high (the red LED (MLDP_EV) illuminates on the RN4020 PICtail Board) to indicate the pending status message to be sent over the UART to the Host microcontroller.
7. Once the CMD/MLDP pin is set low to enter command mode, the status message will be output to the UART. The maximum status message that can be buffered is 256 bytes.

Note: To exit MLDP mode, set the CMD/MLDP pin to be low and the module will acknowledge with the “CMD” string, which appears on the UART indicating that the RN4020 module is back in command mode. A Host microcontroller is required to interpret the command and responses available on the UART of the modules.
Running the Demo

As mentioned earlier, configure one node as Peripheral and another node as Central as required by the BLE platform for communication. Use the following procedure to program, connect, and test two wireless BLE nodes.

1. Program one board with the BLE_Peripheral.x.production.hex file and another board with the BLE_Central.x.production.hex file.
2. After two boards are programmed, the Peripheral device will be in the Auto-Advertising mode and the Central device will be in Scan mode.
3. The Central node has a LCD for monitoring and two switches for scrolling and selecting.
4. Scroll for the specific Peripheral node from the Central node using the switch SW3.
5. Select the Peripheral node using the switch SW6.
6. Upon selection, the Peripheral node will connect to the Central node and the green (CONN) LED on the RN4020 PICtail board turns ON, which indicates the connection.
7. The modules then switch to the MLDP mode.
8. Press the switch S3 or S6 from the Peripheral node or Central node. The LED D9 or D10 toggle on the Central or Peripheral node.

**Figure 15** illustrates the Central and Peripheral nodes are connected in the MLDP mode.

**FIGURE 15: CENTRAL AND PERIPHERAL NODES CONNECTED IN MLDP MODE**
APPENDIX A: SOURCE CODE

Software License Agreement

The software supplied herewith by Microchip Technology Incorporated (the “Company”) is intended and supplied to you, the Company’s customer, for use solely and exclusively with products manufactured by the Company.

The software is owned by the Company and/or its supplier, and is protected under applicable copyright laws. All rights are reserved. Any use in violation of the foregoing restrictions may subject the user to criminal sanctions under applicable laws, as well as to civil liability for the breach of the terms and conditions of this license.

THIS SOFTWARE IS PROVIDED IN AN “AS IS” CONDITION. NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.

All of the software covered in this application note is available as a single WinZip archive file. This archive can be downloaded from the Microchip corporate web site at: www.microchip.com
APPENDIX B: SOURCE CODE FILE LIST

Table 1 provides the source files that are used as part of the Central node (Node A).

### TABLE 1: CENTRAL NODE SOURCE FILES

<table>
<thead>
<tr>
<th>File Name</th>
<th>File Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Central</td>
<td>.c and .h</td>
<td>files Central command state machine</td>
</tr>
<tr>
<td>Central_string</td>
<td>.c and .h</td>
<td>files Command and response strings required for Central node</td>
</tr>
<tr>
<td>Config</td>
<td>.h file</td>
<td>files PIC24F device configurations</td>
</tr>
<tr>
<td>Config_fuse</td>
<td>.c file</td>
<td>files Config fuses used to configure PIC24F</td>
</tr>
<tr>
<td>LCD</td>
<td>.c and .h</td>
<td>files LCD interface</td>
</tr>
<tr>
<td>Main</td>
<td>.c and .h</td>
<td>files Initialization of PIC24F device</td>
</tr>
<tr>
<td>UART</td>
<td>.c and .h</td>
<td>files UART driver interface for RN4020</td>
</tr>
</tbody>
</table>

Table 2 provides the source files that are used as part of the peripheral node (Node B).

### TABLE 2: PERIPHERAL NODE SOURCE FILES

<table>
<thead>
<tr>
<th>File Name</th>
<th>File Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC</td>
<td>.c and .h</td>
<td>files Not used for this demo application</td>
</tr>
<tr>
<td>Config</td>
<td>.h file</td>
<td>files PIC24F device configurations</td>
</tr>
<tr>
<td>Config_fuse</td>
<td>.c file</td>
<td>files Config fuses used to configure PIC24F</td>
</tr>
<tr>
<td>LCD</td>
<td>.c and .h</td>
<td>files LCD interface</td>
</tr>
<tr>
<td>Main</td>
<td>.c and .h</td>
<td>files Initialization of PIC24F device</td>
</tr>
<tr>
<td>Peripheral</td>
<td>.c and .h</td>
<td>files Peripheral command state machine</td>
</tr>
<tr>
<td>Peripheral_string</td>
<td>.c and .h</td>
<td>files Command and response strings required for Peripheral node</td>
</tr>
<tr>
<td>UART</td>
<td>.c and .h</td>
<td>files UART driver interface for RN4020</td>
</tr>
</tbody>
</table>
APPENDIX C: SOURCE CODE CALL GRAPHS

Figure C-1 illustrates the functions used by the Central main() program in the application code.

FIGURE C-1: CENTRAL NODE CALL GRAPH
Figure C-2 illustrates the functions used by the Peripheral main() program in the application code.

FIGURE C-2: PERIPHERAL NODE CALL GRAPH
APPENDIX D: REFERENCED SOURCES

This appendix provides information on the list of resources that are referenced in this application note.

Microchip Technology Inc. Resources

- "RN4020 Bluetooth Low Energy Module Data Sheet" (DS50002279)
- "RN4020 Bluetooth Low Energy Module User’s Guide" (DS70005191)
- "RN4020 PICtail™/PICtail Plus Board User’s Guide" (DS50002265)
- "Explorer 16 Development Board User’s Guide" (DS50001589)
- "PIC24FJ128GA010 Family Data sheet" (DS39747F)
- "MPLAB ICD 3 USER’S GUIDE FOR MPLAB X IDE" (DS50002081)

Other Resources

- Bluetooth 4.1 GATT Definitions Browser: https://developer.bluetooth.org/gatt/Pages/Definition-Browser.aspx

CONCLUSION

This application note is designed to enable Microchip Bluetooth customers to acquire basic understanding of the Bluetooth Low Energy (BLE) and to work with Microchip BLE RN4020 module. This application note also provides sample codes for enabling the RN4020 modules as a Central node and Peripheral node through 16-bit PIC microcontroller. The interface and code examples can be used further as a framework for any of the user applications or projects.
Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip’s Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

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, PIC32 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.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

QUALITY MANAGEMENT SYSTEM CERTIFIED BY DNV
ISO/TS 16949

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company’s quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001:2000 certified.