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.

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in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates.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 unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Helda, JukeBlox, KEELOG, KEELOG logo, Kleer, LANcheck, LINK MD, maxXtulus, maxTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PiC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.


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

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark 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.

© 2017, Microchip Technology Incorporated, All Rights Reserved.

ISBN: 978-1-5224-2059-0
EU Declaration of Conformity

This declaration of conformity is issued by the manufacturer.
The development/evaluation tool is designed to be used for research and development in a laboratory environment. This
development/evaluation tool is not a Finished Appliance, nor is it intended for incorporation into Finished Appliances that are made
commercially available as single functional units to end users under EU EMC Directive 2004/108/EC and as supported by the European
This development/evaluation tool complies with EU RoHS2 Directive 2011/65/EU.
This development/evaluation tool, when incorporating wireless and radio-telecom functionality, is in compliance with the essential
requirement and other relevant provisions of the R&TTE Directive 1999/5/EC and the FCC rules as stated in the declaration of conformity
For information regarding the exclusive, limited warranties applicable to Microchip products, please see Microchip’s standard terms and
conditions of sale, which are printed on our sales documentation and available at www.microchip.com.
Signed for and on behalf of Microchip Technology Inc. at Chandler, Arizona, USA.

Rodger Richey  
Director of Development Tools

Date  4/4/17
**BM70 COMPACT DEMO BOARD (CDB) USER’S GUIDE**

# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>7</td>
</tr>
<tr>
<td>Chapter 1. Overview</td>
<td></td>
</tr>
<tr>
<td>1.1 Introduction</td>
<td>11</td>
</tr>
<tr>
<td>1.2 BM-70-CDB Description</td>
<td>11</td>
</tr>
<tr>
<td>1.3 Features</td>
<td>12</td>
</tr>
<tr>
<td>Chapter 2. Interface Description</td>
<td></td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>13</td>
</tr>
<tr>
<td>2.2 BM-70-CDB</td>
<td>13</td>
</tr>
<tr>
<td>2.3 Sensor Board</td>
<td>14</td>
</tr>
<tr>
<td>Chapter 3. Using BM70 Firmware on BM-70-CDB</td>
<td></td>
</tr>
<tr>
<td>3.1 Introduction</td>
<td>15</td>
</tr>
<tr>
<td>3.2 Connecting The BM-70-CDB To A Host PC</td>
<td>15</td>
</tr>
<tr>
<td>3.3 BM70 UART Transparent Connection Using Smartdata App</td>
<td>16</td>
</tr>
<tr>
<td>Chapter 4. Using RN4870 Firmware on BM-70-CDB</td>
<td></td>
</tr>
<tr>
<td>4.1 Introduction</td>
<td>19</td>
</tr>
<tr>
<td>4.2 Connecting The BM-70-CDB To A Host PC</td>
<td>19</td>
</tr>
<tr>
<td>4.3 Changing Settings Using ASCII Commands</td>
<td>22</td>
</tr>
<tr>
<td>4.4 Connecting To The RN4870 Using SmartDiscover APP</td>
<td>23</td>
</tr>
<tr>
<td>4.5 Creating Custom GATT Services</td>
<td>25</td>
</tr>
<tr>
<td>4.6 Accessing GATT Service Using UART Commands And SmartDiscover App</td>
<td>26</td>
</tr>
<tr>
<td>Chapter 5. RN4870 Sensor Board</td>
<td></td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>29</td>
</tr>
<tr>
<td>5.2 Reading Sensor Board Peripheral I/O Ports Using UART Commands</td>
<td>29</td>
</tr>
<tr>
<td>5.3 Configuring The RN4870 Module Settings</td>
<td>31</td>
</tr>
<tr>
<td>5.4 Sensor Board GATT Service</td>
<td>32</td>
</tr>
<tr>
<td>5.5 Transferring Sensor Data Into GATT Service Using Scripting</td>
<td>34</td>
</tr>
<tr>
<td>5.6 BLESensorApp Smartphone App</td>
<td>36</td>
</tr>
<tr>
<td>5.7 Provision Utility</td>
<td>38</td>
</tr>
<tr>
<td>Appendix A. Loading RN4870 Firmware</td>
<td></td>
</tr>
<tr>
<td>A.1 Introduction</td>
<td>43</td>
</tr>
<tr>
<td>Appendix B. Loading BM70 Firmware</td>
<td></td>
</tr>
<tr>
<td>B.1 Introduction</td>
<td>47</td>
</tr>
<tr>
<td>Appendix C. BM-70-CDB Example Configuration</td>
<td></td>
</tr>
<tr>
<td>C.1 Introduction</td>
<td>51</td>
</tr>
<tr>
<td>C.2 Requirements</td>
<td>51</td>
</tr>
</tbody>
</table>
C.3 Setting Configuration Parameters .......................................................... 52

Appendix D. Schematics and BOM
D.1 Introduction .................................................................................................. 59
D.2 BM-70-CDB Schematic ............................................................................... 60
D.3 BM-70-CDB BOM ....................................................................................... 61
D.4 RN4870 Sensor Board Schematic .............................................................. 62
D.5 RN4870 Sensor Board BOM ........................................................................ 62

Appendix E. Bluetooth Low Energy Primer
E.1 GAP Roles: Peripheral And Central ............................................................ 65
E.2 GATT Service: Client And Server ............................................................... 65

Appendix F. Sensor Board Script Text
F.1 Overview ...................................................................................................... 67
F.2 Sensor Board Script ..................................................................................... 67

Appendix G. Transparent UART
G.1 Introduction .................................................................................................. 69

Worldwide Sales and Service ............................................................................. 71
INTRODUCTION

This chapter contains general information that will be useful to know before using the BM70 Compact Demo Board (CDB). Items discussed in this chapter include:

• Document Layout
• Conventions Used in this Guide
• Recommended Reading
• The Microchip Website
• Development Systems Customer Change Notification Service
• Customer Support
• Document Revision History

DOCUMENT LAYOUT

This document describes how to use the BM70 Compact Demo Board (CDB) as a development tool to emulate and debug firmware on a target board. The document is organized as follows:

• Chapter 1. “Overview” describes the hardware and software requirements of the BM70 Compact Demo Board (CDB).
• Chapter 2. “Interface Description” provides the board and interface descriptions of the BM-70-CDB Board and the Sensor Board.
• Chapter 3. “Using BM70 Firmware on BM-70-CDB” provides the steps on how to establish a connection to the BM-70-CDB using the BM70 firmware.
• Chapter 4. “Using RN4870 Firmware on BM-70-CDB” provides the steps on how to establish a connection to the BM-70-CDB using the RN4870 firmware.
• Chapter 5. “RN4870 Sensor Board” demonstrates how to use the RN4870 Sensor Board with BM-70-CDB.
• Appendix A. “Loading RN4870 Firmware” provides the steps on how to load the RN4870 firmware to BM-70-CDB using the PC Tool isupdate.exe.
• Appendix B. “Loading BM70 Firmware” provides the steps on how to update the firmware for the BM70 using the PC Tool isupdate.exe.
• Appendix C. “BM-70-CDB Example Configuration” provides the hardware and software requirements to configure the BM70 module.
• Appendix D. “Schematics and BOM” provides the schematics and the Bill of Materials (BOM) for the BM-70-CDB and the Sensor Board.
• Appendix E. “Bluetooth Low Energy Primer” provides a brief summary of GAP roles and GATT services.
• Appendix F. “Sensor Board Script Text” provides an example of a script that can be used to control and communicate with the Sensor Board.
• Appendix G. “Transparent UART” provides an introduction to the Transparent UART Service.

CONVENTIONS USED IN THIS GUIDE

This manual uses the following documentation conventions:

<table>
<thead>
<tr>
<th>Description</th>
<th>Represents</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arial font:</td>
<td>Referenced books</td>
<td><strong>MPLAB® IDE User’s Guide</strong></td>
</tr>
<tr>
<td>Italic characters</td>
<td>Emphasized text</td>
<td>...is the only compiler...</td>
</tr>
<tr>
<td>Initial caps</td>
<td>A window</td>
<td>the Output window</td>
</tr>
<tr>
<td></td>
<td>A dialog</td>
<td>the Settings dialog</td>
</tr>
<tr>
<td></td>
<td>A menu selection</td>
<td>select Enable Programmer</td>
</tr>
<tr>
<td>Quotes</td>
<td>A field name in a window or dialog</td>
<td>“Save project before build”</td>
</tr>
<tr>
<td>Underlined, italic text with</td>
<td>A menu path</td>
<td><strong>File&gt;Save</strong></td>
</tr>
<tr>
<td>right angle bracket</td>
<td>A dialog button</td>
<td>Click <strong>OK</strong></td>
</tr>
<tr>
<td></td>
<td>A tab</td>
<td>Click the <strong>Power</strong> tab</td>
</tr>
<tr>
<td>N’Rnnnn</td>
<td>A number in verilog format, where N is the total number of digits, R is the radix and n is a digit.</td>
<td>4'b0010, 2'hF1</td>
</tr>
<tr>
<td>Text in angle brackets &lt; &gt;</td>
<td>A key on the keyboard</td>
<td>Press &lt;Enter&gt;, &lt;F1&gt;</td>
</tr>
<tr>
<td><strong>Courier New font:</strong></td>
<td>Sample source code</td>
<td>#define START</td>
</tr>
<tr>
<td></td>
<td>Filenames</td>
<td>autoexec.bat</td>
</tr>
<tr>
<td></td>
<td>File paths</td>
<td>c:\mcc18\h</td>
</tr>
<tr>
<td></td>
<td>Keywords</td>
<td>_asm, _endasm, static</td>
</tr>
<tr>
<td></td>
<td>Command-line options</td>
<td>-Opa+, -Opa-</td>
</tr>
<tr>
<td></td>
<td>Bit values</td>
<td>0, 1</td>
</tr>
<tr>
<td></td>
<td>Constants</td>
<td>0xFF, 'A'</td>
</tr>
<tr>
<td>Italic Courier New</td>
<td>A variable argument</td>
<td><em>file.o</em>, where <em>file</em> can be any valid filename</td>
</tr>
<tr>
<td>Square brackets [ ]</td>
<td>Optional arguments</td>
<td>mcc18 [options] file [options]</td>
</tr>
<tr>
<td>Curly brackets and pipe character: { }</td>
<td>Choice of mutually exclusive arguments; an OR selection</td>
<td>errorlevel {0</td>
</tr>
<tr>
<td>Ellipses...</td>
<td>Replaces repeated text</td>
<td>var_name [ , var_name ...]</td>
</tr>
<tr>
<td></td>
<td>Represents code supplied by user</td>
<td>void main (void) { ... }</td>
</tr>
</tbody>
</table>
RECOMMENDED READING

This user's guide describes how to use BM70 Compact Demo Board (CDB). Other useful document(s) are listed below. The following Microchip document(s) are recommended as supplemental reference resources.

BM70/71 Bluetooth® Low Energy Module Data Sheet (DS60001372)
This document provides the technical specifications for the BM70/71 module and is available for download from the Microchip website (www.microchip.com).

BM70/71 Bluetooth® Low Energy Module User's Guide (DS50002542)
This document describes how to use the BM70/71 module as a development tool to emulate and debug firmware on a target board.

THE MICROCHIP WEBSITE

Microchip provides online support via our website at www.microchip.com. This website is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the website contains the following information:

• Product Support – Data sheets and errata, application notes and sample programs, design resources, user’s guides and hardware support documents, latest software releases and archived software
• General Technical Support – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip consultant program member listing
• Business of Microchip – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events; and listings of Microchip sales offices, distributors and factory representatives

DEVELOPMENT SYSTEMS CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip’s customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip website at www.microchip.com, click on Customer Change Notification and follow the registration instructions.

The Development Systems product group categories are:

• Compilers – The latest information on Microchip C compilers and other language tools
• Emulators – The latest information on the Microchip MPLAB® REAL ICE™ in-circuit simulator
• In-Circuit Debuggers – The latest information on the Microchip in-circuit debugger, MPLAB ICD 3
• MPLAB X IDE – The latest information on Microchip MPLAB X IDE, the Windows® Integrated Development Environment for development systems tools
• Programmers – The latest information on Microchip programmers including the PICKit™ 3 development programmer
CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

• Distributor or Representative
• Local Sales Office
• Field Application Engineer (FAE)
• Technical Support

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the website at:

http://www.microchip.com/support.

DOCUMENT REVISION HISTORY

Revision A (August 2017)

This is the initial release of this document.
1.1 INTRODUCTION

This document describes the hardware and software requirements for the BM70 Compact Demo Board (CDB), part number: BM-70-CDB. The BM-70-CDB enables the designer to evaluate and demonstrate the capabilities of the BM70 module.

The BM-70-CDB includes the following key features:

- Integrated configuration and programming interface
- LED and push button for prototype user inputs
- MCP2200 USB to UART bridge for plug-and-play connectivity to host PC

An additional RN4870 Sensor Board is included to demonstrate peripheral access using Bluetooth Low Energy (BLE), refer to Figure 1-1. In addition to BM-70-CDB hardware, the following smartphone applications from Microchip are available on the Apple iTunes® and Google Play™ Store to demonstrate the capabilities of the BM70 module:

- SmartDiscover
- SmartData
- BLESensorApp

For more information on the BM70, refer to the “BM70/71 Bluetooth® Low Energy Module Data Sheet” (DS60001372) and the “BM70/71 Bluetooth® Low Energy Module User’s Guide” (DS50002542), which are available for download from the Microchip product web page at www.microchip.com/BM70.

1.2 BM-70-CDB DESCRIPTION

The BM-70-CDB is a versatile evaluation board for developing and prototyping Bluetooth Low Energy data applications. It can be powered via USB host or through the coin cell battery. The BM-70-CDB uses the BM70, a fully certified Bluetooth 4.2 Low Energy module. The BM-70-CDB provides a USB connectivity to connect to a host PC, a PC terminal utility and smartphone apps to evaluate both BLE serial data connections and custom BLE services.
1.3 FEATURES

The BM-70-CDB has the following features:

- Fully certified Bluetooth Low Energy 4.2 RF module
- On-Board Dual In-Line Package (DIP) switch block to set operating modes
- Embedded MCP2200 USB-UART bridge which provides programming interface to update firmware and configuration settings
Chapter 2. Interface Description

2.1 INTRODUCTION

This chapter provides details on the components and the interface for the following boards:

- BM-70-CDB
- Sensor Board

2.2 BM-70-CDB

The BM-70-CDB components are represented in Figure 2-1. The CDB can be used in Standalone mode powered by either an external USB or a coin cell battery.

FIGURE 2-1: BM-70-CDB

2.2.1 Interface Description

1. BM70 module
2. Status indicator LED (LD2); For blink rate descriptions, refer to the “BM70/71 Bluetooth® Low Energy Module Data Sheet” (DS60001372)
3. Push button for hardware Reset (SW1)
4. Current measurement test point (J8)
5. Header pins for coin cell battery connection (J6, close Pin 1 and Pin 2)
6. Header pins for USB power (J6, close Pin 2 and Pin 3)
7. MCP2200 - USB to UART bridge
8. Micro-USB connector
9. DIP Switch block (SW2)
   - Switch 1: P1_2 GPIO
   - Switch 2: P2_0 programming mode
   - Switch 3: LD2 status LED enable/disable
   - Switch 4: P35 GPIO

CN1, CN2, and CN3 are header pins to the BM70 module pins. The header pins are used to connect to the Sensor Board or as test points.

Note: While updating the firmware on BM-70-CDB, ensure that the J6 jumper (Pin 2 and Pin 3) and J8 jumper (current measurement test point) are closed. Do not interrupt jumpers J6 and J8 during firmware update, otherwise the firmware can be corrupted.

2.3 SENSOR BOARD

The BM-70-CDB kit comes with the Sensor Board to demonstrate the peripheral I/O capabilities of the BM70. The Sensor Board components are illustrated in Figure 2-2.

The Sensor Board is an accessory designed for the BM-70-CDB to demonstrate digital and analog I/O capabilities over Bluetooth Low Energy connections. In order to use the Sensor Board with the BM-70-CDB, configuration must be done as described in Chapter 5, “RN4870 Sensor Board”.

FIGURE 2-2: RN4870 SENSOR BOARD

2.3.1 Interface Descriptions
1. Test Point for ADC channels of the BM70 module
2. Variable Resistor to drive ADC0 input
3. DIP Switch block to connect or disconnect sensors from the BM70 module
4. LED connected to pin P22
5. Light Sensor connected to ADC1 input
6. Push Button connected to PIO2

Note: Turn ON all the switches on DIP switch block SW2 before running the demos in this document.
Chapter 3. Using BM70 Firmware on BM-70-CDB

3.1 INTRODUCTION

The simplest method to program the BM70 module on the BM-70-CDB is to connect it to a PC host that supports USB CDC virtual COM (serial) ports.

Using BM70 firmware on CDB, the user data can be transferred to a remote device over Transparent UART (Refer to Appendix G. “Transparent UART”). This section shows the procedure on how to connect to a smartphone via the BM70 and open a Transparent UART data pipe to communicate with the smartphone app.

To interact directly with the BM70 module, the following software are needed:

- PC Host supporting USB CDC virtual serial port
  - The BM-70-CDB uses the MCP2200 USB/UART bridge. The drivers can be downloaded from the Microchip website at www.microchip.com/MCP2200.
- Terminal Emulator Application
  - TeraTerm or CoolTerm is recommended.
- Microchip SmartData App for iOS®
  - Available on App Store® (for iOS).

Note 1: At the time of publication, the BM70 modules are released with firmware version 1.03. The latest firmware version for BM70 is available for download from the Microchip product web page at www.microchip.com/BM70.

2: Refer to Appendix B. “Loading BM70 Firmware” on how to program the firmware on BM70 and Appendix C. “BM-70-CDB Example Configuration” on how to configure BM70 after programming the firmware.

3.2 CONNECTING THE BM-70-CDB TO A HOST PC

To connect the BM-70-CDB to a host PC, perform the following steps:

1. Configure the BM-70-CDB as follows:
   - Pin 2 and Pin 3 on J6 are closed to select the USB power option
   - J8 is closed
   - Operating mode switch (Switch 2 on SW2) is in position 2 to enable application mode.

2. Connect the BM-70-CDB to the host PC using Micro-USB cable.
   - Verify that the virtual COM port is enumerated on the host PC. If the COM port does not enumerate, the MCP2200 drivers may need to be downloaded to the host PC. The drivers can be downloaded from the Microchip website at www.microchip.com/MCP2200.
   - Press Reset button (SW1) and verify that LD2 is flashing slowly.

3. Start the Terminal Emulator software. In this example, TeraTerm is used. Configure the enumerated COM port as shown in Figure 3-1.
3.3 BM70 UART TRANSPARENT CONNECTION USING SMARTDATA APP

To establish a UART Transparent connection on the BM-70-CDB using Bluetooth SmartData App, perform the following steps:

1. On your Apple device, download and install the SmartData App by Microchip Technology Inc. from iTunes App Store.

2. Open a serial port terminal to the BM70 module by following the steps in Section 3.2 “Connecting The BM-70-CDB To A Host PC”.

3. Turn ON Bluetooth and open SmartData App on the iOS device.

4. The SmartData automatically scans for BLE modules in the vicinity upon opening. From the list of scanned devices, Click “BLE-SPP” (or any user-defined name) from the scan list in SmartData App to connect to the module. Refer to Figure 3-2.
5. After the connection is established with the BM70 module, select the **Serial Data** tab in the SmartData App.

6. Enter example text in the first dialog box of the **Serial Data** tab and click **Send**. The example text displays in the serial terminal emulator on the host PC.

7. Type any data on the serial terminal of the BM70 to send data from BM70 to SmartData App. The data sent from the host PC is displayed in the Receive box of the SmartData app. Refer to **Figure 3-3** and **Figure 3-4**.
FIGURE 3-3: DATA EXCHANGE ON SMARTDATA APP WITH BM70 MODULE

FIGURE 3-4: DATA EXCHANGE ON BM70 UART TERMINAL EMULATOR WITH SMARTDATA APP
Chapter 4. Using RN4870 Firmware on BM-70-CDB

4.1 INTRODUCTION

The BM70 module on the BM-70-CDB can be loaded with the RN4870 firmware and operated as a RN4870 module for evaluation and development. Refer to Appendix A. “Loading RN4870 Firmware” for the procedure on how to load the RN4870 firmware onto the BM70 module.

The BM-70-CDB module is programmed with RN4870 firmware. In this section, the module is referred to as RN4870 and the hardware board as BM-70-CDB.

The simplest method to access the BM-70-CDB is to connect it to a PC host that supports USB CDC virtual COM ports. To interact directly with the BM-70-CDB, the following software are needed:

- PC Host supporting USB CDC virtual serial port
  - The BM-70-CDB uses MCP2200 USB to UART bridge. The drivers can be downloaded from the product web page at www.microchip.com/MCP2200.
- Terminal Emulator Application
  - TeraTerm or CoolTerm is recommended
- Microchip SmartDiscover App for iOS or Android
  - Available on App Store (for iOS) or Google Play Store (Android)
- RN4870 Sensor Board Support Package (RN4870-Sensor-Board-Support.zip)
  - This is a collection of files and utilities to assist in configuring the Sensor Board. The files can be downloaded from the Microchip product web page at www.microchip.com/RN4870.

4.2 CONNECTING THE BM-70-CDB TO A HOST PC

To connect the BM-70-CDB to a host PC, perform the following steps:

1. Configure the BM-70-CDB as follows:
   - Pin 2 and Pin 3 on J6 are closed to select USB power option
   - J8 is closed
   - Operating mode switch (Switch 2 on SW2) is in position 2 to enable application mode
   - Switch 3 on SW2 is in position 3 to enable LD2 (Blue LED)
2. Connect the BM-70-CDB to the host PC using Micro-USB cable and perform the following steps:
   - Verify that the virtual COM port is enumerated on the host PC. If the COM port does not enumerate, the MCP2200 drivers may be missing from the host PC. The drivers can be downloaded from the Microchip website at www.microchip.com/MCP2200.
   - Press Reset button (SW1) and verify that LD2 is flashing slowly
3. Start the Terminal Emulator software. In this example, TeraTerm is used. Configure the enumerated COM port as shown in Figure 4-1.
4. Enter Command mode by sending the command escape sequence $$$$.
Pressing the dollar sign ($) three times puts the RN4870 into Command mode and displays the CMD> prompt. Enable the 'local echo' feature by pressing +, which echoes the character entered by the user on the serial emulator window. The ECHO ON response is displayed as shown in Figure 4-2.

5. To display the basic configuration settings, press letter <D> followed by the <Enter> key (\r) as shown in Figure 4-3. Refer to the “RN4870/71 Bluetooth® Low Energy Module User’s Guide” (DS50002466) for a detailed explanation of the commands.
6. SmartDiscover App from Microchip can be used to demonstrate the RN4870 features. Refer to Figure 4-4 for an example of the SmartDiscover App icon. The iOS version of SmartDiscover App is referred in this document.

![SmartDiscover App Icon](image)

FIGURE 4-4: SMARTDISCOVER APP ICON

7. Once SmartDiscover App is opened on your smartphone, it discovers the RN4870 peripherals as shown in Figure 4-5.

![SmartDiscover App Discovering RN4870 Peripherals](image)

FIGURE 4-5: SMARTDISCOVER APP DISCOVERS RN4870 PERIPHERAL
4.3 CHANGING SETTINGS USING ASCII COMMANDS

The RN4870 firmware is shipped with a default configuration which includes two services (Transparent UART service and Device Info Service) as noted by the result of the previous command D (displays basic configuration). The list of services is indicated as Services = C0. Refer to the “RN4870/71 Bluetooth® Low Energy Module User’s Guide” (DS50002466) for a bitmap of services. The RN4870 can be discovered using Bluetooth SmartDiscover App. The SmartDiscover App is available in the iTunes Store and the Google Play Store.

To enable built-in services and connection to the RN4870, perform the following steps:

1. Connect the BM-70-CDB to the host PC USB port:
   - Using Terminal Emulator, open the COM port enumerated for the BM-70-CDB.
   - Type $$$ to enter Command mode.
   - Enter + to turn ON ECHO. Refer to Figure 4-2.

2. To change the default name, send the following commands:
   - S-,BLE to create a unique serialized name based on Bluetooth (BT) address of the device.
   - R,1 to reboot the module and enable the configuration commands to take effect. Refer to Figure 4-6.

3. Verify that the changes made using the S- command are saved by the reboot with the following steps:
   - Type $$$ to enter Command mode.
   - Enter + to turn ON local echo.
   - Issue command D to display the settings.
   - Verify that the name of the device is BLE-b1b0, where b1b0 are the low order bytes of the BT address. In this example, the Bluetooth is set to BLE-B1E4. Refer to Figure 4-7.
4.4 CONNECTING TO THE RN4870 USING SMARTDISCOVER APP

To connect the RN4870 using SmartDiscover App, perform the following steps:

1. Configure the RN4870 according to the steps listed in Section 4.3 “Changing Settings Using ASCII Commands”. Connect it to the host PC and type $$ to enter the Command mode.

2. Launch the SmartDiscover App. Refer to Figure 4-8 on how to demonstrate the SmartDiscover App on an iOS device.
3. Verify that the RN4870 device name, BLE-b1b0, is displayed. In this example, the device name is BLE-B1E4 as shown in Figure 4-8.

4. Click the RN4870 device from the list to initiate a connection.

5. Ensure that the connection is established. The GATT service view is illustrated in Figure 4-9. The connection is also verified by the %CONNECT,1,<MAC> status message returned by the RN4870/71 UART. The <MAC> is the address of the remote BT device that initiated the connection. The response in the terminal emulator is shown in Figure 4-10.

**FIGURE 4-9: GATT SERVICE VIEW WHEN CONNECTED**

![GATT Service View](image)

**FIGURE 4-10: CONNECTED STATUS MESSAGE**

![Connected Status Message](image)
4.5 CREATING CUSTOM GATT SERVICES

The RN4870 module supports custom (or private) GATT services. Public GATT services are defined by specifications published by the Bluetooth Special Interest Group (SIG). Private GATT services are defined by the user to host information stored in the GATT characteristics. For more information on GATT services, refer to Appendix E. “Bluetooth Low Energy Primer”.

To create a private GATT service, enter the configuration commands listed in Step 3 below. The commands can be entered as shown in this user guide, or copied from configuration files and pasted into the terminal emulator. The configuration text files are included in the RN4870 Sensor Board Support package that is available for download from the Microchip product web page at www.microchip.com/RN4870.

1. Connect the BM-70-CDB to the host PC USB port
   - Using Terminal Emulator, open the COM port to the RN4870
   - Type $$ to enter Command mode
   - Enter + to turn ON ECHO.

2. Set Factory Default values
   - Enter PZ to clear all services and characteristics
   - Enter SF,1
   - Verify that the module reboots after the command is entered.

3. Create the private GATT service with three characteristics by entering the following commands:
   - PS,4D6963726F636869702D524E34383730
   - PC,BF3FBD80063F11E59E690002A5D5C501,02,02
   - PC,BF3FBD80063F11E59E690002A5D5C502,02,02
   - PC,BF3FBD80063F11E59E690002A5D5C503,18,04

   These commands can be entered manually by typing each line followed by the <Enter> key from the file 3.4-GATT-service.txt, or each line copied and pasted one by one into the terminal emulator.

   Command PS creates the GATT service, identified by the 16-byte private UUID: 4D6963726F636869702D524E34383730. This command must be called before command PC.

   Command PC creates the private characteristics. Each characteristic is identified by the following UUIDs:
   - BF3FBD80063F11E59E690002A5D5C501,
   - BF3FBD80063F11E59E690002A5D5C502,
   - BF3FBD80063F11E59E690002A5D5C503.

   Command PC expects three parameters. The first parameter is the UUID, the second parameter is the characteristic property (refer to Table E-1 in Appendix E. “Bluetooth Low Energy Primer”), and the third parameter is the size of the data of the characteristic.

4. Reboot the module using command R,1 to ensure that the new GATT service takes effect.

5. Verify that the GATT service is correctly configured.
   - Enter Command mode ($$$) after rebooting the module.
   - Issue the LS command to list the GATT server services and their characteristics. The response is illustrated in Figure 4-11.
6. The result of the command LS is shown in Figure 4-11. A custom GATT service (UUID: 4D6963726F636869702D524E34383730) with three characteristics identified by low-order bytes C501, C502 and C503 from the 128-bit UUID. A 16-bit handle is assigned to each characteristic. Handles are used to reference and identify 128-bit characteristic UUIDs more efficiently in the GATT service. Note that there are two references for C503 characteristic. In the first case, in reference 0076, the property value 08 has the write property enabled (refer to Table E-1 in Appendix E. “Bluetooth Low Energy Primer”). Likewise, reference 0077 has the notification property 10 enabled. This means that to write a value to characteristic C503, reference 0076 is used. To enable Client notifications on this characteristic, reference 0077 is used.

4.6 ACCESSING GATT SERVICE USING UART COMMANDS AND SMARTDISCOVER APP

The following examples show how to read and write GATT characteristic values by using UART commands:

1. To write a value to the GATT Server characteristic C501, use the Server Handle Write (SHW) command with handle (for example, 0072) as first parameter, following by hex-byte values as shown in Figure 4-12.

2. To read a value from the GATT Server characteristic C501, use the Server Handle Read (SHR) command with handle (for example, 0072) as first parameter, following by hex-byte values as shown in Figure 4-12.

FIGURE 4-11: PRIVATE SERVICES DISPLAY FROM LS COMMAND

FIGURE 4-12: WRITING AND READING GATT VALUE BY HANDLE REFERENCE
3. It is also possible to access the GATT server over a Bluetooth Low Energy connection using SmartDiscover App. Launch the SmartDiscover App and connect to the RN4870 configured with the private GATT as shown in Section 4.5 “Creating Custom GATT Services”. In this example, the device “BLE-B1E4” is used. Follow the steps in Figure 4-13 to read the value of the GATT characteristic C501. In Step 1, click on the name of your device (BLE-B1E4 >), in Step 2 click on the UUID (BF3FB80-063F-11E5-96E9-0002A5D5C501 >), and in Step 3 click on Read. The characteristic value is read from the RN4870 into the SmartDiscover App.

FIGURE 4-13: USING SMARTDISCOVER TO READ GATT CHARACTERISTIC VALUE
Chapter 5. RN4870 Sensor Board

5.1 INTRODUCTION

This chapter demonstrates how to use RN4870 Sensor Board with BM-70-CDB which is loaded with RN4870 firmware. Once RN4870 is configured to enable the Sensor Board peripherals, the sensor values can be read locally using the RN4870 UART commands or from the BLESensor smartphone app over GATT. The configuration procedure is as follows:

1. Configure Peripheral I/O port to sensors using command SW.
2. Define a GATT service (Sensor Board) to hold the values from peripherals for a read access from a GATT client.
3. Similarly, when a GATT client writes a value to a GATT server characteristic in a peripheral (GAP), there must be a method for the peripheral (GAP) to write the value destined for characteristic into the peripheral port.
4. A GATT client on a Central (GAP) device is used to access the GATT characteristics in the peripheral (GAP) which is an RN4870 with Sensor Board. The GATT client for the Sensor Board is a smartphone app named BLESensorApp. It is possible to use another RN4870 as the GATT client to access the RN4870 Sensor Board peripherals.

The critical function of associating GATT characteristic with a peripheral is performed by the RN4870 scripting capability. The RN4870 can be manually configured by following the instructions in Section 4.2 “Connecting The BM-70-CDB To A Host PC” through Section 4.5 “Creating Custom GATT Services”. Alternatively, the RN4870 can also be configured by using the PC Utility (RN4870_SensorBoard_Cmd_Cmd.exe) as described in Section 5.7 “Provision Utility”.

5.2 READING SENSOR BOARD PERIPHERAL I/O PORTS USING UART COMMANDS

The RN4870 has digital and analog peripheral inputs and outputs. For a detailed description of the I/O interface, refer to the “RN4870/71 Bluetooth® 4.2 Low Energy Module Data Sheet” (DS50002489), and for the configuration details, refer to Section 2.4.24 and Section 2.6.5 through Section 2.6.8 of the “RN4870/71 Bluetooth® Low Energy Module User's Guide” (DS50002466). Peripheral I/O can be accessed through direct UART commands or over Bluetooth Low Energy connection by associating a GATT characteristic with a peripheral input or output.

This section provides the procedure on how to read and write peripherals on the Sensor Board using the BM70 and the terminal emulator. The pins on the BM70 of the BM-70-CDB that are connected to the corresponding peripherals on the Sensor Board are indicated in Figure 5-1.

1. Connect the Sensor Board to the BM-70-CDB as shown in Figure 1-1.
2. Set the module to Factory default using the SF,1 command to set the peripherals to default.
   - Using Terminal Emulator, open the COM port to the RN4870 module.
   - Type $$ to enter Command mode.
   - Enter SF,1 and verify that the module reboots after entering the command.
3. Enter the following sequence of commands to configure the RN4870 I/O pins as illustrated in Figure 5-1:
   - Type $$$ to enter Command mode.
   - Enter + to enable local echo.
   - Enter SW,01,00 for ADC input port 01 used by the light sensor.
   - Enter SW,02,00 for ADC input port 02 used by the potentiometer.
   - Enter SW,03,00 to configure digital I/O port connected to the LED.
   - Enter SW,04,09 to configure digital I/O trigger connected to the push button.
   - Enter R,1 to reboot the module for the settings to take effect.

**FIGURE 5-1: RN4870 PINS USED BY SENSOR BOARD INTERFACE**

4. After the I/O ports are configured and the module is rebooted, enter the commands shown in Figure 5-2 to read and write peripheral I/O.
5.3 CONFIGURING THE RN4870 MODULE SETTINGS

The RN4870 and the Sensor Board can be configured by entering the commands manually. Alternatively, the configuration commands are located in Appendix F. “Sensor Board Script Text”. Each command can be copied to the clipboard via <CTRL+C> and sent to the RN4870 via TeraTerm using the <ALT+R> (Paste) command.

1. Connect the BM-70-CDB to the host PC USB port.
   - Using Terminal Emulator, open the COM port to the RN4870.
   - Type $$$ to enter Command mode.
   - Enter + to turn ON local echo.
   - Enter WP to stop any running script.
   - Enter WC to clear any previous script.
   - Enter PZ to clear any previous user defined GATT service.
   - Enter SF, 1 to set the module into factory default and verify that the module reboots after the command is entered.
   - Type $$$ to re-enter Command mode.
   - Enter + to turn ON local echo.

2. Configure Module Features.
   - Enter S-, RN4870 to serialize Bluetooth name.
   - Enter SS, 80 to enable Device Information Profile.
   - Enter SR, 4040 to enable scripting start on PWR_ON event and disable command prompt.
   - Enter SW, 01, 00 for ADC input port 01 used by the light sensor.
   - Enter SW, 02, 00 for ADC input port 02 used by the potentiometer.
   - Enter SW, 03, 00 to configure digital I/O port connected to the LED.
   - Enter SW, 04, 09 to configure digital I/O trigger connected to the push button.
   - Enter R, 1 to reboot the module.
   - Type $$$ to re-enter Command mode.
- Enter + to turn ON local echo.
- Enter command D to display settings as shown in Figure 5-3. Command GW is used to return to the current I/O configuration.

**FIGURE 5-3: RN4870 MODULE SETTINGS CONFIGURATION**

![Figure 5-3: RN4870 Module Settings Configuration](image)

### 5.4 SENSOR BOARD GATT SERVICE

The data from the RN4870 peripheral I/O are stored in the characteristics of a custom GATT service accessed by the Sensor Board BLE Client App. To create the Sensor Board GATT Service and characteristics, enter the following commands while in Command mode:

1. `PS,AD11CF40063F11E5BE3E002A5D5C51B` to create a 128-bit UUID to identify the GATT Service.
2. `PC,BF3FBD80063F11E59E69002A5D5C501,10,02` to declare a two-byte characteristic with notify property enabled. This characteristic stores the value of the ADC channel used for light sensor.
3. `PC,BF3FBD80063F11E59E69002A5D5C502,10,02` to declare a two-byte characteristic with notify property enabled. This characteristic stores the value of the ADC channel used for potentiometer.
4. `PC,BF3FBD80063F11E59E69002A5D5C503,18,14` to declare a 20-byte characteristic used to manage both the push button state (virtual LED in App) and the LED blink rate on the Sensor Board. Both notify and write properties are enabled for this characteristic.
5. `PC,BF3FBD80063F11E59E690002A5D5C504,10,02` to declare a two-byte characteristic with notify property enabled. This characteristic stores the value of the ADC channel used for the RN4870 internal temperature sensor.

6. `PC,BF3FBD80063F11E59E690002A5D5C505,10,02` to declare a two-byte characteristic with notify property enabled. This characteristic stores the value of the ADC channel used for the RN4870 internal battery voltage sensor.

7. Command `R,1` to reboot the RN4870 module.

8. Command `$$` to enter Command mode.

9. Command `+` to enable local echo.

10. Command `LS` (list services) to display Sensor Board GATT Service as shown in Figure 5-4.

**Figure 5-4:** SENSOR BOARD GATT SERVICE DECLARED IN RN4870

Figure 5-4 illustrates that a custom GATT service has been created in the RN4870. A 16-bit handle is assigned to each characteristic where each characteristic has a value handle and a property handle. The characteristic value and the properties are accessed by a short and more efficient 16-bit handle instead of the 128-bit UUID value. For example, two handles 0072 and 0073 are associated with the GATT characteristic `BF3FBD80063F11E59E690002A5D5C501`. A GATT client such as a smartphone App uses 0072 to read/write characteristic values, whereas 0073 is used to enable or disable the notify property.

The next step is to transfer the sensor values from the peripheral I/O into the GATT characteristics. This is done with the RN4870 scripting feature as described in Section 5.5 “Transferring Sensor Data Into GATT Service Using Scripting”.
5.5 TRANSFERRING SENSOR DATA INTO GATT SERVICE USING SCRIPTING

For more information about the Sensor Board GATT service, refer to Section 5.4 “Sensor Board GATT Service”. To populate the characteristics with data from the Sensor Board peripherals, and conversely transfer values received from the GAP client into peripheral I/O channels, the RN4870 scripting feature is used. For more details on the scripting capabilities, refer to the “RN4870/71 Bluetooth® Low Energy Module User’s Guide” (DS50002466).

The script that drives peripheral values to and from the GATT service is written into the Nonvolatile Memory (NVM) of the RN4870 via the UART.

First, put the module into Script Entry mode. The script is a series of text commands separated by carriage returns. The actual script is displayed in Appendix F. “Sensor Board Script Text”. It can be copied to the clipboard and pasted in UART using TeraTerm. Similarly, the PC Utility can load it into RN4870.

To manually load the Sensor Board script, perform the following steps:

1. Connect the BM-70-CDB to the host PC USB port.
   - Using Terminal Emulator, open the COM port to the RN4870 module.
   - Type $$$ to enter Command mode.
   - Enter + to turn ON ECHO.
2. Enter Script Entry mode.
   - Enter command WW to put the RN4870 into Script Entry mode. This command does not return an AOK response.
   - Copy the script text from Appendix F. “Sensor Board Script Text” into the clipboard using <Ctrl+C>.
   - Enter the script in TeraTerm using the <Alt+R> (Paste) command. Figure 5-5 shows a dialog window showing the script text from the clipboard.

   FIGURE 5-5: SENSOR BOARD SCRIPT DISPLAYED FROM CLIPBOARD

   - Press OK to load the script into the RN4870.
   - After the script is entered into the RN4870, exit Script Entry mode by pressing the <ESC> key. An AOK response is sent to the terminal. The script entry is shown in Figure 5-6.
- Enter command $R,1$ to reboot the RN4870. After reboot, the Sensor Board is ready to use. Figure 5-7 shows the script event handlers are seen on the TeraTerm display.
5.6 BLESENSORAPP SMARTPHONE APP

The RN4870 Sensor Board is a BLE Peripheral that advertises a GATT service to BLE Central devices. In this demonstration, the BLE Central device is a smartphone application named BLESensorApp. It is available to download for iOS and Android platforms via iTunes App Store, and Google Play Store, respectively. Search for the "Microchip BLESensorApp", and download the App matching the icon in Figure 5-8.

1. Ensure that the Sensor Board is attached to the BM-70-CDB. Refer to Figure 1-1.
2. Verify the Switch 2 in SW2 is in position 2: application mode.
3. Power the BM-70-CDB via USB
   - Connect the BM-70-CDB to power up
   - Verify that LD2 blinks blue with long interval.
4. Launch the BLESensorApp from the smartphone. In this example, the iOS version is used as shown in Figure 5-9.
5. The BLESensorApp attempts to connect to the first RN4870 Sensor Board it discovers. Once connected, the main user interface of the BLESensorApp is displayed. This auto connection is only possible when location access permission is enabled for BLESensorApp.
6. If location access permission is not enabled, then BLESensorApp displays the list of available RN4870 Sensor Boards in the vicinity. The connection can be initiated by clicking the desired device as shown in Figure 5-9.
FIGURE 5-10: BLE DATA FLOW FROM SENSOR BOARD PERIPHERAL I/O TO BLESSENSORAPP

Figure 5-10 illustrates the data flow between the RN4870 Sensor Peripheral I/O and the smartphone app. The data streaming from the Sensor Board to the App is performed by GATT Notifications. The script polls the sensor values, such as Switch, Light Meter, and writes the values into the appropriate GATT Characteristic. If the value changes, a GATT Notification is sent to the BLESensorApp where the App updates the Graphical User Interface (GUI).

BLE data can also be transferred from the BLESensorApp to the RN4870, and later processed by the script to write a received value into the Peripheral output channel. In the Sensor Board demo, slider control is used on the App to select a blink rate value. The value is written to GATT characteristic in the RN4870. The sensor script monitors the Characteristic for incoming data then extracts the value which is subsequently written to the Pulse-With Modulation (PWM) output peripheral to control LED blink rate.

For more information on the BLE GATT data transfers, refer to Appendix E. “Bluetooth Low Energy Primer”.

5.7 PROVISION UTILITY

The RN4870 Provision Utility can be used to provision or configure the RN4870 Bluetooth Low Energy (BLE) module from a Windows Host PC using a wired connection over UART interface. The RN4870 Provision Utility is used to program the RN4870 with the configuration and script to work with the Sensor Board and the BLESensorAPP.

The utility comes with a few prebuilt provision options that can be selected using the Provision File drop-down menu. Alternatively, a custom configuration XML file can be imported into the utility to provision the RN4870 module.

Detailed instructions for provisioning the RN4870 module and creating custom configuration XML file are provided in the RN4870 Provision Utility Help files. Examples of the Provision XML file are in the Examples folder.
Download the RN4870 Provision Utility from www.microchip.com/RN4870. Extract the contents into a separate folder on the host PC. The contents of the extracted folder include the RN4870 Provision Utility executable, the RN4870 Provision Utility Help files and the example configuration XML files. This utility requires the MCP2200 USB/UART CDC drivers. The driver is also available from the web page. Install the driver if needed.

**Note:** While the provisioning is in progress, the RN4870 must not be disconnected and the process must not be interrupted. If the provisioning process is interrupted, the RN4870 module may not be configured as desired.

To configure the RN4870 with a Sensor Board plugged in, perform the following steps:

1. Ensure that the Sensor Board is attached to the RN4870 (BM-70-CDB). Refer to Figure 1-1.
2. Verify the Switch 2 in SW2 is in position 2: application mode.
3. Connect the RN4870 to the host PC using the Micro-USB cable and verify that the LD2 blinks blue with long interval.
4. Wait for the RN4870 board to successfully enumerate as a Serial Port device.
5. After the RN4870 board enumerates as a Serial Port device, open the Device Manager on the Windows PC and note down the COM port number assigned to the RN4870 board under 'Ports'.
6. Run the RN4870 Provision Utility (RN4870ProvisionUtility.exe) on the Windows PC. Refer to Figure 5-11.

**FIGURE 5-11: CONFIGURATION UTILITY MAIN WINDOW**

7. Ensure that the RN4870 board is still connected to the Windows PC and successfully enumerated. Click **Scan** button to scan for all the UART COM ports available on the Windows PC.

**Note:** Ensure that the COM port on which the RN4870 board is enumerated is not opened by another application like a Serial Terminal application.

8. Click the drop-down menu of the UART COM Port and select the correct COM port that is assigned to the RN4870 board to be provisioned. Refer to Figure 5-12.
9. A set of prebuilt provision options for the RN4870 module are provided. Choose the RN4870_Sensors_Demo_1.xml XML file from the Provision File drop-down menu as shown in Figure 5-13.

**Note:** Use only a valid BLE Provision XML (.xml) file.

10. With the RN4870 board still connected, the COM port and the RN4870_Sensors_Demo_1.xml file selected, click Provision button to start the RN4870 provisioning process. The progress bar shows the progress of the provisioning process. Refer to Figure 5-14.
11. After the provisioning process is completed, an information dialog window pops up to inform about the successful provisioning of the RN4870 module. Refer to Figure 5-15.

**FIGURE 5-15: CONFIGURATION UTILITY: PROVISIONING SUCCESSFUL**

*Note:* After the provisioning is successful, the RN4870 automatically reboots and retains the provisioned configuration.
Appendix A. Loading RN4870 Firmware

A.1 INTRODUCTION

The RN4870 firmware can be loaded into BM-70-CDB using a PC Tool isupdate.exe over the USB port. The latest RN4870 firmware images and the isupdate.exe tool are available from the product web page at www.microchip.com/RN4870.

To update the firmware on the BM-70-CDB, perform the following steps:

1. Download the firmware zip file from the product web page and extract the contents. The zip file contains the isupdate.exe utility and a folder including the firmware images.
2. Connect the BM-70-CDB to the host PC using the Micro-USB cable.
3. Verify that USB power is selected in J6 (close Pin 2 and Pin 3) and J8 (current measurement test point) is closed.
4. Verify that Switch 2 on SW2 is set to ON position.
5. Press SW1 (Reset) button and verify that LD2 is showing a non-flashing blue that indicates the RN4870 is in programming mode.
6. Launch the isupdate.exe application.
   - Select the COM port used by the CDB.
   - Verify that other settings (baud rate, memory type, and address) are set as shown in Figure A-1.

![Figure A-1: isupdate.exe Firmware Update Utility](image-url)
7. Click the **Connect** button as shown in Figure A-1.
8. Verify the "Port connect -> COMxx" is displayed in the text box, as shown in Figure A-2.

**FIGURE A-2: MODULE CONNECTTED FROM ISUPDATE.EXE**

9. Click the **Browse** button to select the firmware files that must be flashed on the BM70 module.
10. Click the **Update** button to flash the selected image.
11. Verify that the firmware update is successfully completed. The “End of Write Memory” message is displayed as shown in Figure A-3.

**FIGURE A-3: FIRMWARE SUCCESSFULLY UPDATED**
12. Click the **Disconnect** button to close the COM port. The "port disconnect" message is displayed as indicated in **Figure A-4**.

**FIGURE A-4:** DISCONNECT ISUPDATE.EXE FROM THE MODULE

13. Set Switch 2 on SW2 to position 2 for application mode.
14. Open TeraTerm and connect to the module.
15. Press SW1 to reboot the module. Ensure that the **%REBOOT%** message is displayed.
16. Enter Command mode by sending **$$** escape sequence.
17. Enter the **V** command and verify firmware version as illustrated in **Figure A-5**.

**Note:** While uploading the firmware on BM-70-CDB, ensure that the J6 jumper (Pin 2 and Pin 3) and J8 jumper (current measurement test point) are closed. Do not interrupt jumpers J6 and J8 during firmware update, otherwise the firmware can be corrupted.

**FIGURE A-5:** VERIFY FIRMWARE VERSION
Appendix B. Loading BM70 Firmware

B.1 INTRODUCTION

The BM70 firmware can be updated using a PC Tool isupdate.exe over the USB port. The latest BM70 firmware images and the isupdate.exe tool are available from the Microchip product web page at www.microchip.com/BM70.

To update the firmware on the BM-70-CDB, perform the following steps:

1. Download the firmware zip file from the product web page and extract the contents. The zip file contains the isupdate.exe utility and a folder including the firmware images.
2. Connect the BM-70-CDB to the host PC using the Micro-USB cable.
3. Verify that USB power is selected in J6 (close Pin 2 and Pin 3) and J8 (current measurement test point) is closed.
4. Verify that Switch 2 on SW2 is set to ON position.
5. Press SW1 (Reset) button and verify that LD2 is showing a non-flashing blue that indicates the BM70 is in programming mode.
6. Launch the isupdate.exe application.
   - Select the COM port used by the CDB.
   - Verify that other settings (baud rate, memory type, and address) are set as shown in Figure B-1.

FIGURE B-1: isupdate.exe FIRMWARE UPDATE UTILITY
7. Click the **Connect** button as shown in *Figure B-1*. 
8. Verify the "Port connect -> COMxx" is displayed in the text box as shown in *Figure B-2*.

**FIGURE B-2: MODULE CONNECTED FROM ISUPDATE.EXE**

9. Click the **Browse** button to select the firmware files that must be flashed on the BM70 module.
10. Click the **Update** button to flash the selected image.
11. Verify that the firmware update is successfully completed. The "End of Write Memory" message is displayed as shown in *Figure B-3*. 

12. Click the **Disconnect** button to close the COM port. The "port disconnect" message is displayed as indicated in Figure B-4.
13. Set Switch 2 on SW2 to position 2 for application mode.
14. Press SW1 to reboot the module.

**Note:** While uploading the firmware on BM-70-CDB, ensure that the J6 jumper (Pin 1 and Pin3) and J8 jumper (current measurement test point) are closed. Do not interrupt jumpers J6 and J8 during firmware update, otherwise the firmware can be corrupted.
Appendix C. BM-70-CDB Example Configuration

C.1 INTRODUCTION

This appendix shows the procedure for updating the configuration parameters of the BM70 module using the BM-70-CDB.

C.2 REQUIREMENTS

C.2.1 Hardware Requirements

- BM-70-CDB
- Bluetooth-enabled smartphone
  - iPhone 4S or later version (it must support BLE) or
  - Android device running Android 4.3 or later version
- Windows host PC with USB port
- Micro-USB cable

C.2.2 Software Requirements

Download and install the latest firmware and corresponding tools (Firmware_SoftwareTools_Vx_xx.zip) from the Microchip product web page at www.microchip.com/BM70.

- Firmware update tool (BM7xBLE_IS187x_FlashUpdateTool.exe)
- Firmware hex images (*.H00, *.H01, *.H02, *.H03)
- Configuration tool (IS187x_102_BLEDK3_UI v100.xxx.exe)

Note 1: Ensure that the latest firmware is downloaded to the IS1870 IC on the BM-70-CDB. Version 1.03 is the initial BLE firmware version programmed into BM70 modules in Microchip's manufacturing line at the time of releasing this product to market. The latest firmware is made available from the Microchip website at www.microchip.com/BM70.

2: Ensure that the exact version of the Configuration tool (UI Tool) matches the specific firmware version programmed into the IS1870 IC's Flash memory is used to configure the module. For ease of use, the UI tool, firmware and firmware update utility are all provided in a single zip file for each version of firmware released by Microchip to the website. The latest firmware for the BM70 module is not compatible with older versions of the Configuration tool. To ensure that the correct version of firmware and tools are being used together, a Firmware_Software_Vx_xx.zip file is provided. This archive contains the firmware along with the compatible software tools for this specific firmware version.

3: In the following example, the BM-70-CDB with firmware version 1.06 and Configuration tool (IS187x_102_BLEDK3_UI v100.132.exe) is used.
C.3 SETTING CONFIGURATION PARAMETERS

The Configuration tool or User Interface (UI) tool is a PC-based program, which enables the user to change the configuration parameters of the BM70 (For hardware details on the configuration, refer to Chapter 2. “Interface Description”). The following list provides some examples of the parameter settings that can be updated on the BM70 module using the UI tool (For the description of features, refer to Chapter 1. “Overview”):

- Device name
- UART settings
- BLE connection settings
- Add or edit GATT service table

To update the UI parameter settings, perform the following steps:

1. Open the UI tool and click Load, refer to Figure C-1. The Loading Option window is displayed.

   Note: Download and unzip the contents of the Firmware_Software-Tools_Vx_xx.zip file, which is available from the Microchip product web page at www.microchip.com/BM70. The configuration tool is part of this zip file content and located under the sub-folder "ConfigurationTool". In this demonstration, the IS187x_102_BLEDK3_UI v100.132.exe tool is used. This tool version corresponds to firmware version v1.06. Information is shown by the file name Firmware_SoftwareTools_Vx_xx.zip, where "Vx_xx" is the firmware version.

   FIGURE C-1: CONFIGURATION TOOL WINDOW
2. In the Loading Option window, click **Load Text File** to load default configuration parameters. Refer to **Figure C-2**.

**Figure C-2: Loading Option Window**

3. From the Open dialog, select the default configuration parameter text file (provided with the `Firmware_SoftwareTools_Vx_xx.zip` file under the "ConfigurationTool" sub-folder) and then click **Open**. Refer to **Figure C-3**.

**Figure C-3: OPEN DIALOG BOX TO SELECT DEFAULT CONFIGURATION**
4. From the Configuration Tool window, click **Edit** to start editing the default parameters. Refer to Figure C-4.

**FIGURE C-4: CONFIGURATION TOOL WINDOW – EDIT OPTION**

5. From the Main Feature window, click **BLEDK** and then **OK**. Refer to Figure C-5.

**FIGURE C-5: MAIN FEATURE WINDOW**

6. The UI tool displays a window with various configuration options (tabs) in a tabular format. Click the **System Setup** tab, and type “BM70_BLE” (or any user-defined name) in the Name fragment box. Refer to Figure C-6.

**Note:** Click the **Help** button to get information related to UI parameters.
7. Click the **LE Mode Setup** tab. In the Advertising Data Setting section, select **Device Name** to advertise the device name as shown in **Figure C-7**. This ensures that the name fragment is included in the advertising packet.
8. Click **Finish**. The Configuration Tool main window is displayed. Refer to Figure C-8.

9. From the Configuration Tool window, perform one of these actions:
   - Click **Save** to save the selected parameter settings as `.txt` or `.hex` files (used for later production programming).
   - Click **Export** to export a log file along with the parameters to a `.txt` file. The log file contains the configuration parameters settings which are edited.
10. To program the configuration parameters on the BM70 module, perform the following steps:
   - Connect the BM-70-CDB to the host PC using the Micro-USB cable.
   - Verify that the USB power is selected in J6 (close Pin 2 and Pin 3) and J8 (current measurement test point) is closed.
   - Verify that the Switch 2 on SW2 is set to ON position.
   - Press SW1 (Reset) button and verify that LD2 is showing a non-flashing blue that indicates the BM70 is in programming mode.
   - Go to the Configuration Tool window and click **Write** to program the settings into the internal memory of the BM70 module. Refer to Figure C-9.

![WRITE CONFIGURATION](image)

- The Read/Write Flash window is displayed. Select the values for COM Port and Baudrate (must be 115200 for the configuration to succeed), and then click **Write**. Refer to Figure C-10.

![COM SETTINGS TO WRITE CONFIGURATION](image)
A message box appears displaying the message “Write the UI configuration settings to EFLASH! Are you sure?”. Click Yes to write the configuration. Refer to Figure C-11.

**FIGURE C-11: WRITE EFLASH ALERT**

A message box appears displaying the message "Write Flash Finish". Click OK to finish. Refer to Figure C-12.

**FIGURE C-12: WRITE EFLASH FINISH**
D.1 INTRODUCTION

This appendix provides the schematics and the Bill of Materials (BOM) for the BM-70-CDB:

- BM-70-CDB Schematic
- BM-70-CDB BOM
- RN4870 Sensor Board Schematic
- RN4870 Sensor Board BOM
D.2 BM-70-CDB SCHEMATIC

Figure D-1 shows the schematic for the BM-70-CDB.

FIGURE D-1: BM-70-CDB SCHEMATIC
D.3 BM-70-CDB BOM

Table D-1 shows the Bill of Materials for the BM-70-CDB.

<table>
<thead>
<tr>
<th>Qty</th>
<th>Designator</th>
<th>Description</th>
<th>Manufacturer</th>
<th>Manufacturer Part Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B1</td>
<td>BATT HOLDER COIN 1 TH CR2450</td>
<td>Keystone Electronics</td>
<td>1053</td>
</tr>
<tr>
<td>2</td>
<td>C1, C2</td>
<td>CAP CER 12pF 50V 1% NP0 SMD 0402</td>
<td>Murata Electronics North America</td>
<td>GRM1555C1H120FA01D</td>
</tr>
<tr>
<td>3</td>
<td>C3, C4, C7</td>
<td>CAP CER 0.1uF 10V 10% X5R SMD 0402</td>
<td>KEMET</td>
<td>C0402C104K8ACTU</td>
</tr>
<tr>
<td>2</td>
<td>C5, C6</td>
<td>CAP CER 1uF 10V 10% X5R SMD 0402</td>
<td>Murata Electronics North America</td>
<td>GRM155R61A105KE15D</td>
</tr>
<tr>
<td>1</td>
<td>C8</td>
<td>CAP CER 1000pF 25V 5% COG SMD 0402</td>
<td>TDK</td>
<td>C1005COG1E102J</td>
</tr>
<tr>
<td>1</td>
<td>C9</td>
<td>CAP CER 47uF 6.3V 20% X5R SMD 0603</td>
<td>Murata Electronics North America</td>
<td>GRM188R60J476ME15D</td>
</tr>
<tr>
<td>3</td>
<td>J1, J2, J3</td>
<td>CON HDR-2.54 Male 1x9 Gold 5.84MH TH VERT</td>
<td>Samtec</td>
<td>TSW-109-07-G-S</td>
</tr>
<tr>
<td>1</td>
<td>J5</td>
<td>CON USB2.0 MICRO-B FEMALE TH/SMD R/A</td>
<td>FCI</td>
<td>10118194-0001LF</td>
</tr>
<tr>
<td>1</td>
<td>J6</td>
<td>CON HDR-2.54 Male 1x3 Gold 6.0MH TH VERT</td>
<td>JAMECO VALUEPRO</td>
<td>7000-1X3SG-R</td>
</tr>
<tr>
<td>1</td>
<td>J8</td>
<td>CON HDR-2.54 Male 1x2 Gold 5.84MH TH VERT</td>
<td>FCI</td>
<td>77311-118-02LF</td>
</tr>
<tr>
<td>1</td>
<td>LD2</td>
<td>DIO LED BLUE 3.2V 20mA 140mcd Clear SMD 1206</td>
<td>Dialight</td>
<td>598-8291-107F</td>
</tr>
<tr>
<td>2</td>
<td>R1, R5</td>
<td>RES TKF 4.7K 1% 1/10W 0402</td>
<td>KOA Speer</td>
<td>RK73H1ETTP4701F</td>
</tr>
<tr>
<td>1</td>
<td>R2</td>
<td>RES TKF 470R 5% 1/16W SMD 0402</td>
<td>Panasonic</td>
<td>ERJ-2GEJ471X</td>
</tr>
<tr>
<td>1</td>
<td>R4</td>
<td>RES TF 10k 0.5% 1/16W SMD 0402</td>
<td>SUSUMU</td>
<td>RR0510P-103-D</td>
</tr>
<tr>
<td>1</td>
<td>R6</td>
<td>RES TKF 1M 5% 1/16W SMD 0402</td>
<td>Yageo</td>
<td>RC0402JR-071ML</td>
</tr>
<tr>
<td>1</td>
<td>SW1</td>
<td>SWITCH TACT SPST 12V 50mA RS-282G05A3-SM RT</td>
<td>C&amp;K Components</td>
<td>RS-282G05A3-SM RT</td>
</tr>
<tr>
<td>1</td>
<td>SW2</td>
<td>SWITCH DIP 4 SPST 24V 25mA 418121270804 SMD</td>
<td>Wurth Electronics Inc.</td>
<td>418121270804</td>
</tr>
<tr>
<td>1</td>
<td>X1</td>
<td>RESONATOR 12MHz 0.1% SMD CSTCE-G</td>
<td>Murata Electronics North America</td>
<td>CSTCE12M0G15L99-R0</td>
</tr>
<tr>
<td>1</td>
<td>U1</td>
<td>MCHP RF BLUETOOTH BM70BLES1FC2 MODULE-33</td>
<td>Microchip Technology Inc.</td>
<td>BM70BLES1FC2</td>
</tr>
<tr>
<td>1</td>
<td>U2</td>
<td>MCHP INTERFACE USB UART MCP2200-I/SS SSOP-20</td>
<td>Microchip Technology Inc.</td>
<td>MCP2200T-I/SS</td>
</tr>
<tr>
<td>1</td>
<td>U3</td>
<td>MCHP ANALOG LDO 3.3V MCP1700T-3302E/TT SOT-23-3</td>
<td>Microchip Technology Inc.</td>
<td>MCP1700T-3302E/TT</td>
</tr>
<tr>
<td>1</td>
<td>U4</td>
<td>MCHP ANALOG VOLATGE DETECTOR 1.9V MCP112T-195I/TT SOT-23-3</td>
<td>Microchip Technology Inc.</td>
<td>MCP112T-195I/TT</td>
</tr>
</tbody>
</table>
D.4 RN4870 SENSOR BOARD SCHEMATIC

Figure D-2 shows the schematic for the RN4870 Sensor Board.

**Figure D-2: RN4870 SENSOR BOARD SCHEMATIC**

![Schematic Diagram]

D.5 RN4870 SENSOR BOARD BOM

Table D-2 shows the Bill of Materials for the RN4870 Sensor Board.

**Table D-2: RN4870 SENSOR BOARD BOM**

<table>
<thead>
<tr>
<th>Qty</th>
<th>Reference</th>
<th>Description</th>
<th>Manufacturer</th>
<th>Manufacturer Part Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>C1, C2</td>
<td>CAP CER 0.010 µF 25V 10% X7R SMD 0603</td>
<td>Yageo</td>
<td>CC0603KRX7R8BB103</td>
</tr>
<tr>
<td>1</td>
<td>C3</td>
<td>CAP CER 1 µF 50V 10% X5R SMD 0603</td>
<td>Taiyo Yuden</td>
<td>LMK107BJ105KA-T</td>
</tr>
<tr>
<td>3</td>
<td>J1, J2, J3</td>
<td>CON HDR 2.54 Female 1x9 Gold 7MH TH VERT</td>
<td>Preci-Dip</td>
<td>801-87-009-10-001101</td>
</tr>
<tr>
<td>1</td>
<td>LD1</td>
<td>DIO LED GREEN 2V 30 mA 35 mcd Clear SMD 0603</td>
<td>Lite-On Inc.</td>
<td>LTST-C191KGKT</td>
</tr>
<tr>
<td>1</td>
<td>P1</td>
<td>RES Variable CC 20K 10% 1/2W TH 3386P1-xxxT Knob</td>
<td>Bourns Inc.</td>
<td>3386P-1-203TLF</td>
</tr>
<tr>
<td>1</td>
<td>R2</td>
<td>RES TKF 12k 1% 1/10W SMD 0603</td>
<td>Yageo</td>
<td>RC0603FR-0712KL</td>
</tr>
<tr>
<td>1</td>
<td>R3</td>
<td>RES TKF 100R 1% 1/10W SMD 0603</td>
<td>Panasonic</td>
<td>ERJ-3EKF1000V</td>
</tr>
<tr>
<td>1</td>
<td>S1</td>
<td>SENSOR PHOTOCELL 27-60KOHM</td>
<td>Advanced Photonix Inc.</td>
<td>PDV-P8104</td>
</tr>
</tbody>
</table>
## TABLE D-2: RN4870 SENSOR BOARD BOM (CONTINUED)

<table>
<thead>
<tr>
<th>Qty</th>
<th>Reference</th>
<th>Description</th>
<th>Manufacturer</th>
<th>Manufacturer Part Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SW1</td>
<td>SWITCH TACT SPST 12V 50 mA PTS645SM43SMTR92 LFS SMD</td>
<td>C&amp;K Components</td>
<td>PTS645SM43SMTR92 LFS</td>
</tr>
<tr>
<td>1</td>
<td>SW2</td>
<td>SWITCH DIP 4-POS SLIDE SMD 6V</td>
<td>Copal Electronics Inc.</td>
<td>CHS-04TB</td>
</tr>
<tr>
<td>3</td>
<td>TP1, TP2, TP3</td>
<td>MISC, TEST POINT MULTI PURPOSE MINI BLACK</td>
<td>Keystone</td>
<td>5001</td>
</tr>
</tbody>
</table>
Appendix E. Bluetooth Low Energy Primer

E.1 GAP ROLES: PERIPHERAL AND CENTRAL

When two Bluetooth Low Energy (BLE) devices want to be connected, one device must be in Central role and the other in Peripheral role. The Peripheral device advertises to show its connectable status, while Central device scans service advertisements, and if needed, initiates a connection to the Peripheral device. Once connected, either end of the connection can choose to bond. Once bonded, all security related keys are saved and security process are waived when reconnecting. Bonded peripheral device can only perform direct advertise, therefore, unable to connect to a device other than its bonded peer.

E.2 GATT SERVICE: CLIENT AND SERVER

Similar to Bluetooth Classic, BLE uses the concept of profiles to ensure interoperability between different devices. As illustrated in Figure E-1, BLE profiles are collection of services. All BLE services are built on top of Generic Attribute Profile (GATT) where GATT defines accessibility of attributes called characteristics. The main functionality of the BLE profiles, therefore, is built around the characteristics. For those devices that maintain the value of characteristics in a service, such device is the Server of the service. On the other hand, devices that acquire data from their peer are called Client.

Each service and its characteristics can be identified by their Universally Unique Identifier (UUID). The UUID can be a short form (16-bit) or long form (128-bit). As specified by Bluetooth Core Specifications, all Bluetooth SIG adopted (public) services and characteristics have short UUID (16-bit), while user defined private UUIDs are in long form (128-bit). For the details of Bluetooth SIG adopted services and characteristics, please refer to https://developer.bluetooth.org/gatt/profiles/Pages/ProfilesHome.aspx.

The accessibility of each characteristic is defined by a 8-bit characteristic property in bitmap format, as shown in Table E-1.

**TABLE E-1: CHARACTERISTIC PROPERTIES**

<table>
<thead>
<tr>
<th>Property</th>
<th>Bitmap Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extended Property(1)</td>
<td>0x80</td>
<td>Additional property available</td>
</tr>
<tr>
<td>Authenticated Write(1)</td>
<td>0x40</td>
<td>Write characteristic with authentication from Client to Server</td>
</tr>
<tr>
<td>Indicate</td>
<td>0x20</td>
<td>Indicate value of characteristic with acknowledgment from Server to Client</td>
</tr>
<tr>
<td>Notify</td>
<td>0x10</td>
<td>Notify value of characteristic without acknowledgment from Server to Client</td>
</tr>
<tr>
<td>Write</td>
<td>0x08</td>
<td>Write value of characteristic with acknowledgment from Client to Server</td>
</tr>
<tr>
<td>Write without response</td>
<td>0x04</td>
<td>Write value of characteristic without acknowledgment from Client to Server</td>
</tr>
<tr>
<td>Read</td>
<td>0x02</td>
<td>Read value of characteristic. Value is sent from Server to Client</td>
</tr>
<tr>
<td>Broadcast(1)</td>
<td>0x01</td>
<td>Broadcast value of characteristic</td>
</tr>
</tbody>
</table>

**Note 1:** These features are supported in the future firmware releases.
As shown in Figure E-1, the GATT client can access the characteristics in the GATT server in the Peripheral device. When connected, the GATT client reads the GATT server service and characteristic UUIDs. The characteristic values can be accessed by the GATT client using Write, Read, Indication and Notifications.

Write-REQ enables the GATT client to update characteristic values on the Peripheral's GATT server. The write requests can be performed using RN4870 CHW and CUW commands. Refer to Section 4.6 “Accessing GATT Service Using UART Commands And SmartDiscover App” for more information on GATT characteristic access commands.

Write-CMD performs an unacknowledged write from a GATT client to the GATT server. This is allowed for Transparent UART Write procedure on the RN4870 when SR,0100 command is used.

A GATT client sends Read-REQ read a characteristic value on the Peripheral's GATT server. The read requests can be performed using RN4870 CHR and CUR commands. Notifications and Indications are unsolicited updates sent from the GATT server to the GATT client. The GATT client must enable the Notification and Indication on a characteristic to receive the updates. On the RN4870 module, this is done by using the CUW or CHW command to write non-zero value to the Notification Characteristic. When RN4870 in Client mode receives a notification, the %WC,hhhh,ddddddd% message is returned on UART in Command mode.

FIGURE E-1: GATT SERVICE
Appendix F. Sensor Board Script Text

F.1 OVERVIEW

This appendix provides a configuration text example for the Sensor Board Script.

F.2 SENSOR BOARD SCRIPT

To configure the Sensor Board, enter the following sequence of commands:

@PW_ON
IA, Z
IA, 01, 06
IA, FF, CD00FE14AD11CF40063F11E5BE3E002A5D5C51B
%0078 = ?FUNC1
?FUNC1
[,1,1,$PM1,$PM2
@DISCON
SM, 1, 0000
SM, 2, 0000
SM, 3, 0000
R, 1
@CONN
SM, 1, 0002
SM, 2, 0000A
SM, 3, 0000A
@TMR3
$VAR2 = @, 1
SHW, 0075, $VAR2
SM, 3, 0000A
@TMR2
$VAR2 = @, 0
SHW, 0072, $VAR2
SM, 2, 0000A
@TMR1
$VAR2 = @, 5
SHW, 007B, $VAR2
$VAR2 = @, 4
SHW, 007E, $VAR2
SM, 1, 0002
@PIO1L
SHW, 0078, 0001
@PIO1H
SHW, 0078, 0000
NOTES:
Appendix G. Transparent UART

G.1 INTRODUCTION

The Transparent UART Service is instantiated as a Primary Service. The service UUID of the Transparent UART Service is set to 49535343-FE7D-4AE5-8FA9-9FAFD205E455. The Transparent UART Service contains the following data Characteristics:

• Transparent UART Transmit (TX) Characteristic
• Transparent UART Receive (RX) Characteristic

The Transparent UART TX Characteristic is used for data transmission by the Server or the Client. Once the Client Characteristic Configuration Descriptor (CCCD) of Transparent UART TX Characteristic is enabled, the Server sends data to the Client using the Notify property. The Client can also send data to the Server using the Write/Write without response properties.

The Transparent UART RX Characteristic is used for data transmission by the Client. The Client can send data to the Server using the Write/Write without response properties.

Table G-1 shows the UUIDs and the properties of the data characteristics.

<table>
<thead>
<tr>
<th>Characteristic Name</th>
<th>UUID</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transparent UART TX</td>
<td>49535343-1E4D-4BD9-BA61-23C647249616</td>
<td>Notify, Write, Write without response</td>
</tr>
<tr>
<td>Transparent UART RX</td>
<td>49535343-8841-43F4-A8D4-ECBE34729BB3</td>
<td>Write, Write without response</td>
</tr>
</tbody>
</table>