Using BLE and LoRa with the RM1xx

RM1xx Series

Application Note *v1.0*

INTRODUCTION

The RM1xx modules simplify bridging BLE sensor devices with a LoRaWAN network. This application note demonstrates the simplicity of this effort by collecting temperature data from a BLE sensor and forwarding it over the LoRaWAN network to be processed.

REQUIREMENTS

- DVK-RM186 or DVK-RM191 board
- DVK-BL600 board
- RM1xx demo.ble.lora.rm1xx.sb application available at https://github.com/LairdCP/RM1xx-Applications
- BL600 temp.server.notify.bl600.sb application available at https://github.com/LairdCP/RM1xx-Applications
- UwTerminalX, found at https://github.com/LairdCP/UwTerminalX (v1.03 or later recommended)

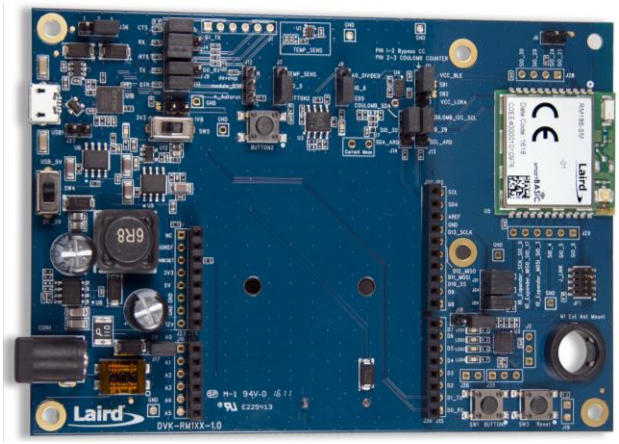

Figure 1: DVK-RM1xx

Figure 2:DVK-BL600

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

Overview – Applications

This guide focuses on two specific *smart*BASIC sample applications: **demo.ble.lora.rm1xx.sb** (which runs on the RM1xx) and **temp.server.notify.bl600.sb** (which runs on the BL600). These applications provide the following functions:

- **demo.ble.lora.rm1xx.sb**: Connect the RM1xx to the BLE sensor (BL600) then connect to the LoRaWAN network. Once both connections are established, the RM1xx accepts temperature notifications from the BL600 device and forwards the notification data through the LoRaWAN network to be parsed and handled by the LoRaWAN network servers.
- **temp.server.notify.bl600.sb**: Send BLE notifications to a GATT Client reporting the value of the temperature sensor on board the DVK-BL600.

The following sections detail a generalized procedure for compiling and loading applications in UwTerminalX, as well as how to test and use each of these sample applications.

Compiling and Loading Applications in *smart*BASIC

UwTerminalX is Laird's terminal software provided for use with *smart*BASIC radios. It includes complete functionality for terminal communication, as well as loading and compiling *smart*BASIC applications. This section covers the generalized procedure for compiling and loading an application. This can be applied to each of the applications mentioned in subsequent sections.

Note: These applications must all be stored in the same directory as the *rm1xx-defs.h* header file that is included in each module firmware release. RM1xx Series firmware releases can be found on the Software Downloads tab of the RM1xx Product Page.

Compiling / Loading to the RM1xx

To compile and load the demo application to the RM1xx Series modules, complete the following steps:

1. Download the applications to your PC.
2. Connect the RM1xx development board to your PC via the included USB micro cable.
3. Power your development board.
4. Launch UwTerminalX.
5. On the Update tab within the UwTerminalX pane, click **Check for Updates** to ensure you're using the latest version of UwTerminalX with support for the RM1xx Series.
6. In the Config tab, in the Device drop down, select either **RM186** or **RM191** based on your setup.
7. Select the correct port to which your development board is connected.
8. Click **OK** to advance to the Terminal tab.
9. Hit **Enter** on your keyboard. If you see the return *00*, you are connected successfully.
10. Right-click in the terminal window, and in the context menu click **XCompile + Load.**
11. In the file selector window, select the **demo.ble.lora.rm1xx.sb** application and click **Open.**
12. When the terminal displays *00*, the compiler has finished successfully (Figure 3).

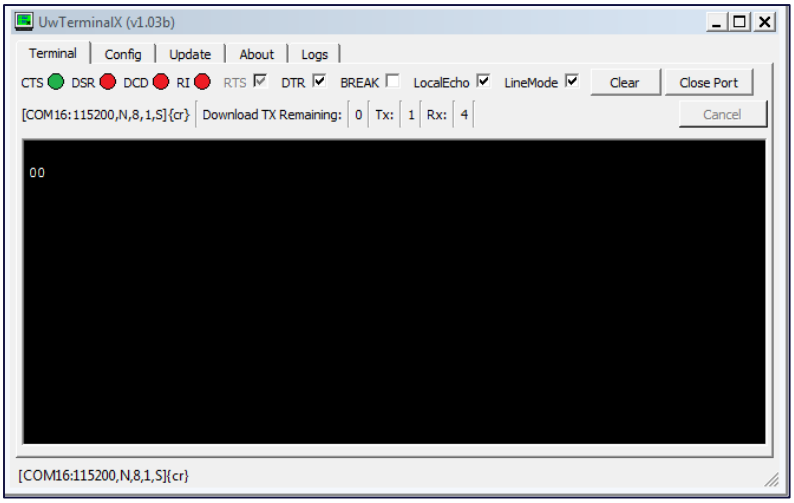

Figure 3: Terminal returns "00" for a successful operation

13. Type **at+dir** and press **Enter**. You should see your application in the file list.

> **Note:** When an application is compiled and loaded in UwTerminalX, the file name is truncated after the first period (.) in the filename. Thus, *demo.ble.lora.rm1xx.sb* becomes *demo*.

14. To run your application, type its filename and press **Enter.**

Compiling / Loading to the BL600

To compile and load the temperature server application to the BL600, follow the same procedure listed in the previous section with the following exceptions:

- Choose BL600/BL620 from the drop down box instead of RM186 or RM191.
- Use the application **temp.server.notify.bl600.sb** instead of demo.ble.lora.rm1xx.sb.
- Run the application when complete by typing its truncated name in UwTerminalX: **temp**

TEMP.SERVER.NOTIFY.BL600.SB

This application is designed to read the temperature data gathered from the onboard temperature sensor on the development board and send the measurement to a GATT Client using a BLE Notification.

DEMO.BLE.LORA.RM1XX.SB

When the **temp** application is running on the BL600 and the **demo** application is running on the RM1xx, they work together to produce the expected output as follows.

On startup, the application opens the GATT Client and starts scanning for recognizable BLE devices:

```
// Open the GATT client
rc=BleGattcOpen(0,0)

// Start scanning for BLE devices
rc=BleScanStart(0,0)
```

Once a BLE device is recognized and a connection is established, the application attempts to join the LoRaWAN network.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

```
// Join the LoRa network
rc=LORAMACJoin(LORAMAC_JOIN_BY_REQUEST)
```

The LoRaWAN connection parameters are expected to be configured prior to running this application. Details and examples of this configuration can be found in the Kerlink Gateway and Multitech Conduit Gateway application notes, found in the documentation tab of the RM1xx product page.

Once both the connection to the BLE device and LoRaWAN network are completed, the application will configure the BLE sensor device to send data using BLE Notifications. Each time a notification is received, the data will be passed to the LoRaWAN network using the LORAMACTxData() smartBASIC API.

In this instance, we are using a Multitech Conduit AEP gateway with NodeRED. Using NodeRED, we can configure the gateway to output all of the incoming LoRaWAN traffic to a debug terminal. In this way, we can see the readings from the DVK-BL600 temperature sensor as they are sent up to the LoRaWAN network. This screenshot shows data coming in in the debug frame on the right as the temperature sensor was subject to a cold shock. The temperature readings are only a couple of seconds apart and are reported in degrees Celcius.

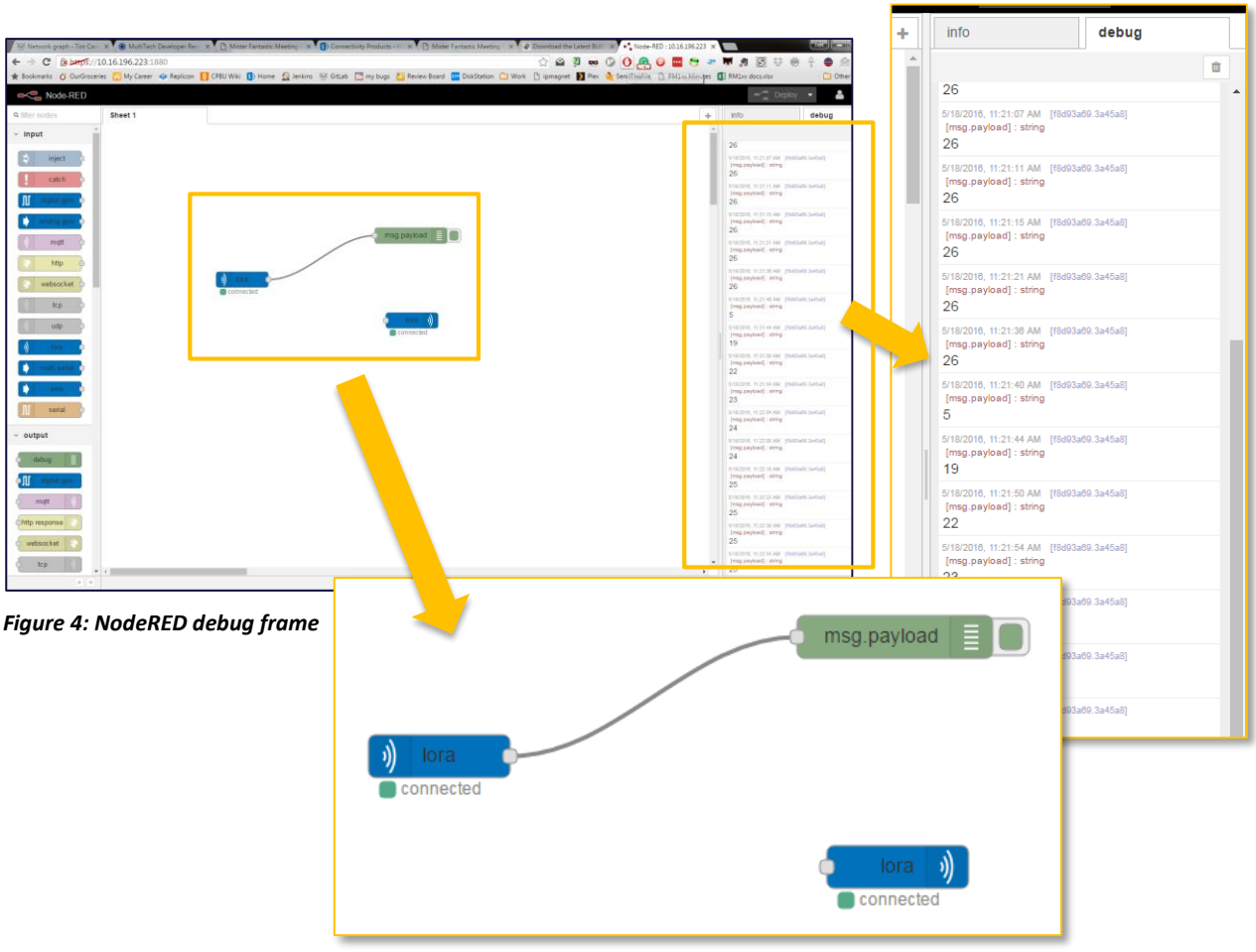

Figure 4: NodeRED debug frame

The corresponding UwTerminalX output from the RM1xx is shown in the following text box.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

```
00
ati 3

10        3        18.4.0.27
00
demo

OK
>LAIRD BL600
Found the BL600. Connecting ...
01C2C2FF508963, RSSI: -48
Advertising data [hex]:
  Length: 1, Type: 0x1, Value: 06


 --- Connect: (0001FF00) handle=1
Conn Interval 90000
Conn Supervision Timeout 500000
Conn Slave Latency 0

Attempting to join the LoRa network

EVFINDCHAR(hConn=0001FF00,hIncUuid=00000000,hVal=0,Props=00000000)
Conn Interval 990000
Conn Supervision Timeout 4000000
Conn Slave Latency 0

Joined the LoRa network
Enable notification for BL600.

EVATTRWRITE(hConn=0001FF00,handle=15,status=0)returned = 0

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3236)LoRa TX Complete Event
LoRa RX Complete Event
```

```
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=35)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3136)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3139)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3230)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3231)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3232)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3232)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3233)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3233)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3233)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3234)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3234)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3234)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3234)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3234)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3234)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3234)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3234)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3234)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
```


Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

```
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)LoRa TX Complete Event
LoRa RX Complete Event

EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
EVATTRNOTIFY()
 >BleGattcNotifyRead(hConn=0001FF00,handle=14,Dumped=0,data=3235)
```

Conclusion

With a few simple lines of smartBASIC code, it is possible to relay BLE sensor data to a LoRaWAN network and process it in multiple ways.

References

- User Guide – RM1xx Series *smart*BASIC Extensions (documentation tab of RM1xx product page)
- User Guide - *smart*BASIC Core Functionality (documentation tab of RM1xx product page)
- Application Note – Connecting to Multitech Conduit Gateway (documentation tab of RM1xx product page)
- Application Note – Connecting to Kerlink Gateway (documentation tab of RM1xx product page)

Revision History

Version	Date	Notes	Approver
1.0	20 May 2016	Initial Release	Tim Carney