NXP Semiconductors
Application Note

Document Number: BLEDAUG
Rev 4, 09/2016

Bluetooth® Low Energy Demo Applications User's Guide

Contents

1 Introduction

This document describes the Bluetooth® low energy stack enablement for the NXP Freedom development platform with the Kinetis KW41Z dual wireless mode system-on-chip (SoC). The initial chapters start with a presentation of the hardware and toolchain requirements.

Chapter 5 describes each demo application that can be found in the software development kit. It presents the profiles and services implemented and how to interact with them.

2 Bluetooth low energy

The Software Development Package provides a Bluetooth Low Energy v4.2-compliant host stack and controller implementation with a set of GATT-based profiles and services implemented on top. To demonstrate the device functionality, the following demo applications are implemented.

- BLE Relay Proxy
- BLE Shell App
- IPv6 Node and Router
- OTAP Client and Server
- HID Device (Mouse) and HID Host
- Low Power Temperature Sensor and Collector
- Wireless UART demo application
- Proximity Reporter
- Alert Notification Server
- Heart Rate Sensor
- Blood Pressure Sensor
- Glucose Sensor
- Health Thermometer
- Cycling Speed and Cadence Sensor
- Cycling Power Sensor
- Running Speed and Cadence Sensor

- Beacon Application
- Wireless Power Transfer Receiving and Transmitting Units

3 Hardware Configurations

3.1 Hardware requirements

- Freedom FRDM-KW41Z platform
- USB-KW41Z platform

3.2 Toolchain requirements

IAR Embedded Workbench is required. The BLE Stack demo applications were compiled and tested with IAR Embedded Workbench for ARM®.

3.3 Freedom FRDM-KW41Z Platform

The main target platform is the FRDM-KW41Z Freedom platform based on the KW41Z wireless, dual-mode SoC, which incorporates an ARM® Cortex®-M0+ core which can be configured to operate at various frequencies, up to 48 MHz. It has 512 KB of Flash and 128 KB of SRAM. A figure representing the platform is shown below. For detailed information about the platform, see the appropriate board user's guide.

The platform features a composite USB device called OpenSDA which serves as debugger interface and as a USB-to-serial converter via a virtual COM port application. Several firmware images can be programmed on the OpenSDA device, such as these.

- developer.mbed.org/handbook/CMSIS-DAP
- segger.com/opensda.html
- www.pemicro.com/opensda

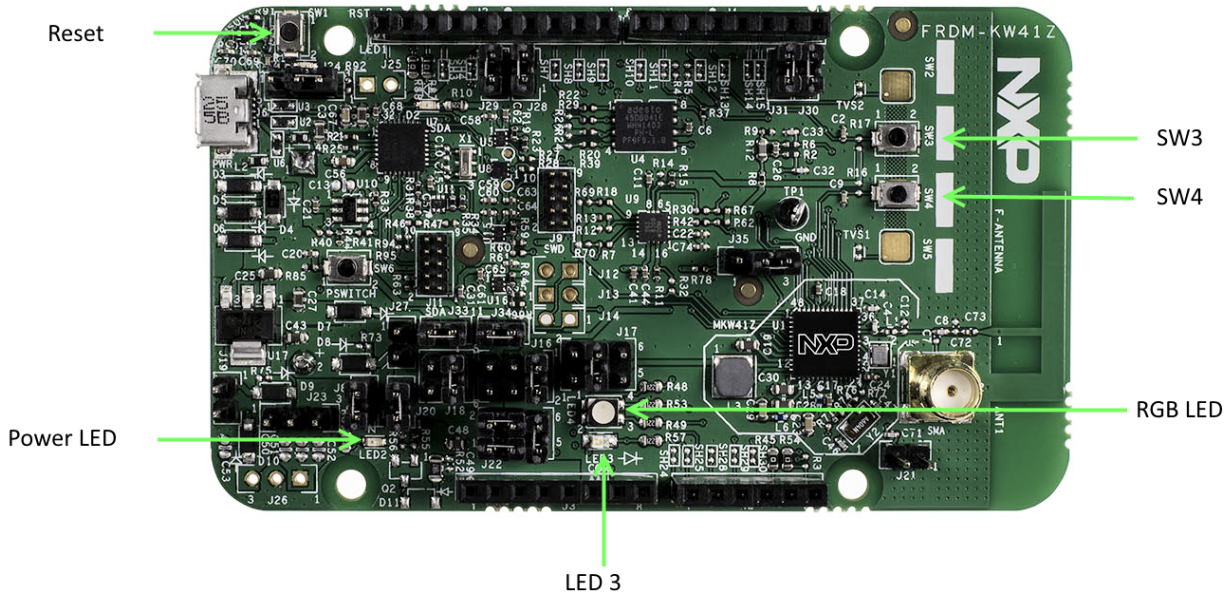

Figure 1. Freedom FRDM-KW41Z platform

3.4 USB-KW41Z platform

The USB-KW41Z board is mainly targeted for sniffer applications. It is based on the same KW41Z wireless, dual mode SoC. A figure representing the platform is shown below. For detailed information about the board, see the appropriate board user's guide.

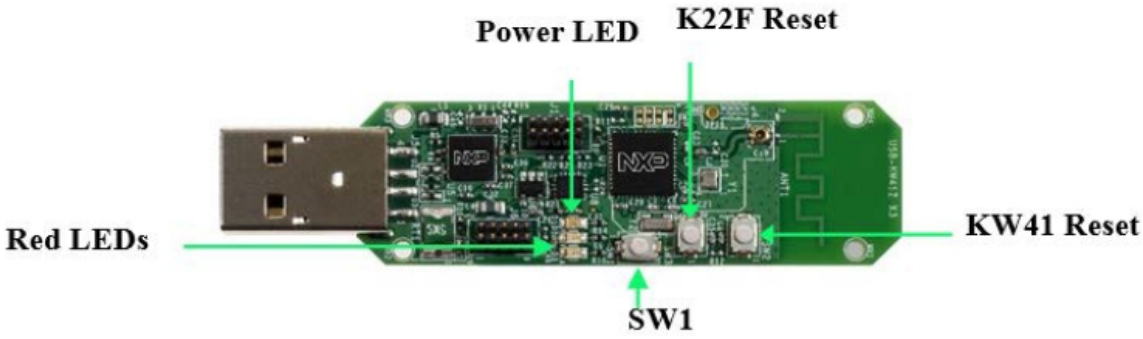

Figure 2. USB-KW41Z platform

4 Build and Run a Bluetooth LE Example Application

To open, build and run any example application, see the MKW41Z Bluetooth Low Energy Quick Start Guide document.

4.1 User Interface

The demo applications that implement the Battery Service expose the current battery level, as measured on the board, through the Battery Level characteristic. The value represents a percentage between 0 and 100. The value can be read from the device from a connected GATT client.

The demo applications that implement the Device Information Service expose different information regarding the current software, hardware, and firmware revisions. These values are used as an example and can be modified by the user when developing their product. The values can be read from the device from a connected GATT client.

4.2 Security

The examples that enable pairing always generates a default passkey of 999999 that has to be entered on the central device, in most cases a smartphone or tablet.

4.3 Testing Devices

To demonstrate the profile functionality, most of the scenarios require one FRDM-KW41Z platform and a Bluetooth Low Energy capable central device, usually a smartphone or a tablet that runs a compatible BLE application.

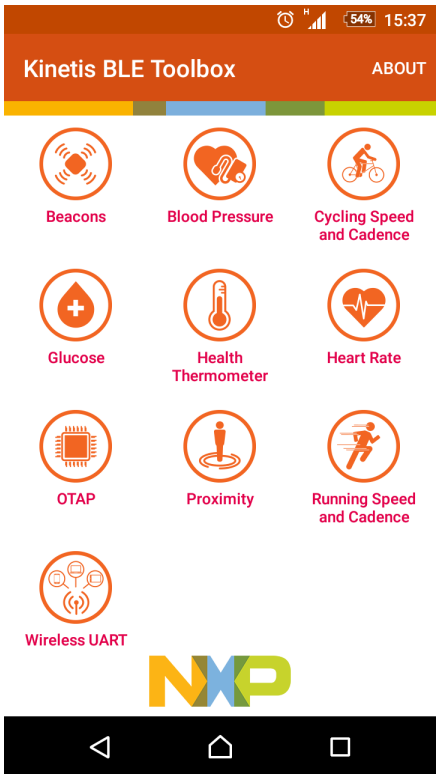

Figure 3. Kinetis BLE Toolbox

The recommended application is the Kinetis BLE Toolbox, which can be installed on Apple® iOS or Android™ OS handheld devices that support Bluetooth Low Energy. The application can be found on iTunes or Google Play Store.

Goolgle playstore

iTunes

Other demos can be run by using two FRDM-KW41Z platforms, one for the peripheral and one for the central role, for example IPv6 Node and Router, Low Power Temperature Sensor and Collector, Wireless UART, OTAP Client and Server, HID Host and Device.

To provide feedback and additional interaction, some examples make use of a shell console via the virtual COM port exposed by the OpenSDA device. To access the device, open a serial port terminal and connect it to the FRDM-KW41Z platform as shown in the figures below. For this example, Tera Term VT was used. The communication parameters are 115200 and 8N1.

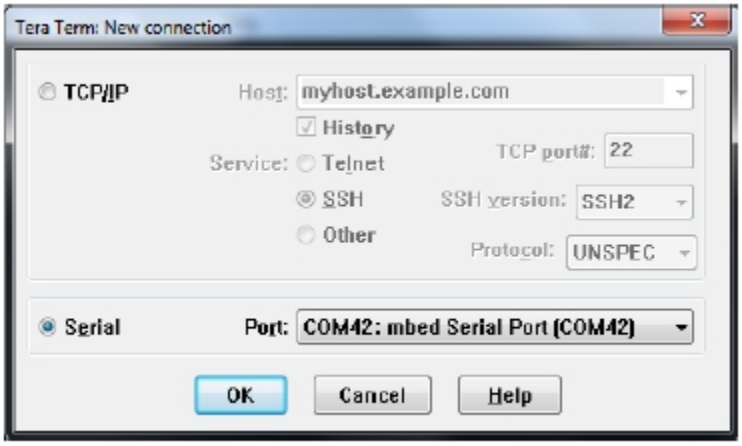

Figure 4. Tera Term – mbed serial port

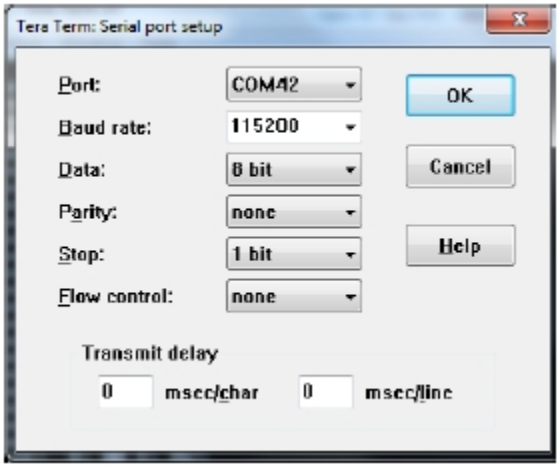

Figure 5. Tera Term – mbed serial port configuration

The start screen is displayed after the board is reset.

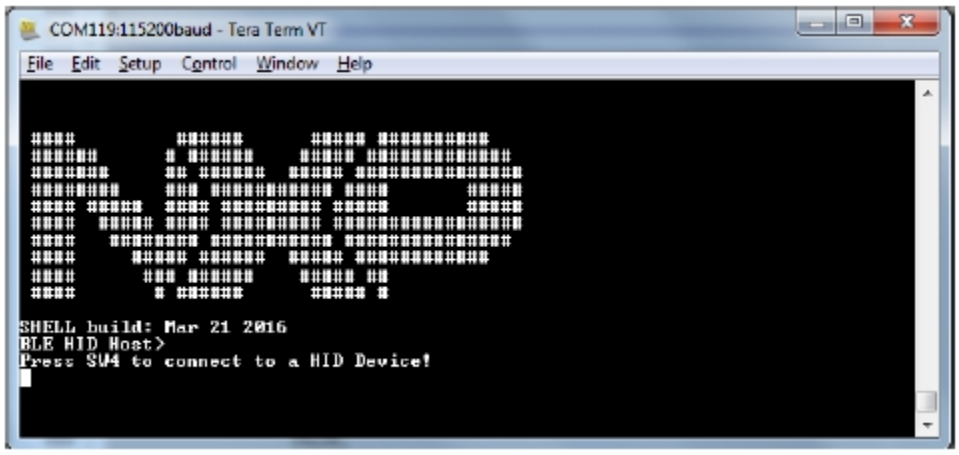

Figure 6. Tera Term – console start

4.4 Time client devices

Some applications implement the Current Time Service. To enable this feature, define gAppUseTimeService_d in the app_preinclude.h file as 1. If the Time Client is enabled, the device must synchronize with a Time Server to update its internal date/time to the current date/time. The Time Client synchronizes with a phone if you connect the device to it (pairing and bonding must be active).

5 BLE Stack and Demo Applications

5.1 Heart rate sensor

This section describes the implemented profiles and services, user interactions, and testing methods for the Heart Rate Sensor application.

5.1.1 Implemented Profile and Services

The Heart Rate Sensor application implements a GATT server and the following profile and services.

- Heart Rate Profile v1.0
- Heart Rate Service v1.0
- Battery Service v1.0
- Device Information Service v1.1

The application behaves as a GAP peripheral node. It enters the GAP General Discoverable Mode and waits for a GAP central node to connect. When the collector configures notifications, the sensor starts sending heart rate measurements every second.

5.1.2 User interface

After flashing the board, the sensor is put in deep sleep (all LEDs are off). Also, any attached debugger will lose its connection. To wake up and start advertising, press the SW4 button. When in GAP Discoverable Mode, the RGB LED is on. When the

central node connects to the peripheral, the RGB LED remains solid. The heart rate measurement values are generated randomly in the interval 40 to 200 bpm. The heart rate value format is set to 8 bits and the contact status is on.

5.1.3 Usage

The Heart Rate Sensor can be connected to any Bluetooth® Smart Ready products available on the market.

To make the Heart Rate Sensor visible, press the SW4 button to start sending advertisements. The sensor name “FSL_HRS” shows on the device when its scanning is active. If configured, the sensor notifies the application with heart rate measurements every second. Also, the battery level and various device information is exposed for reading. The Kinetis BLE Toolbox can be used to showcase the profile functionality, as shown in the image below.

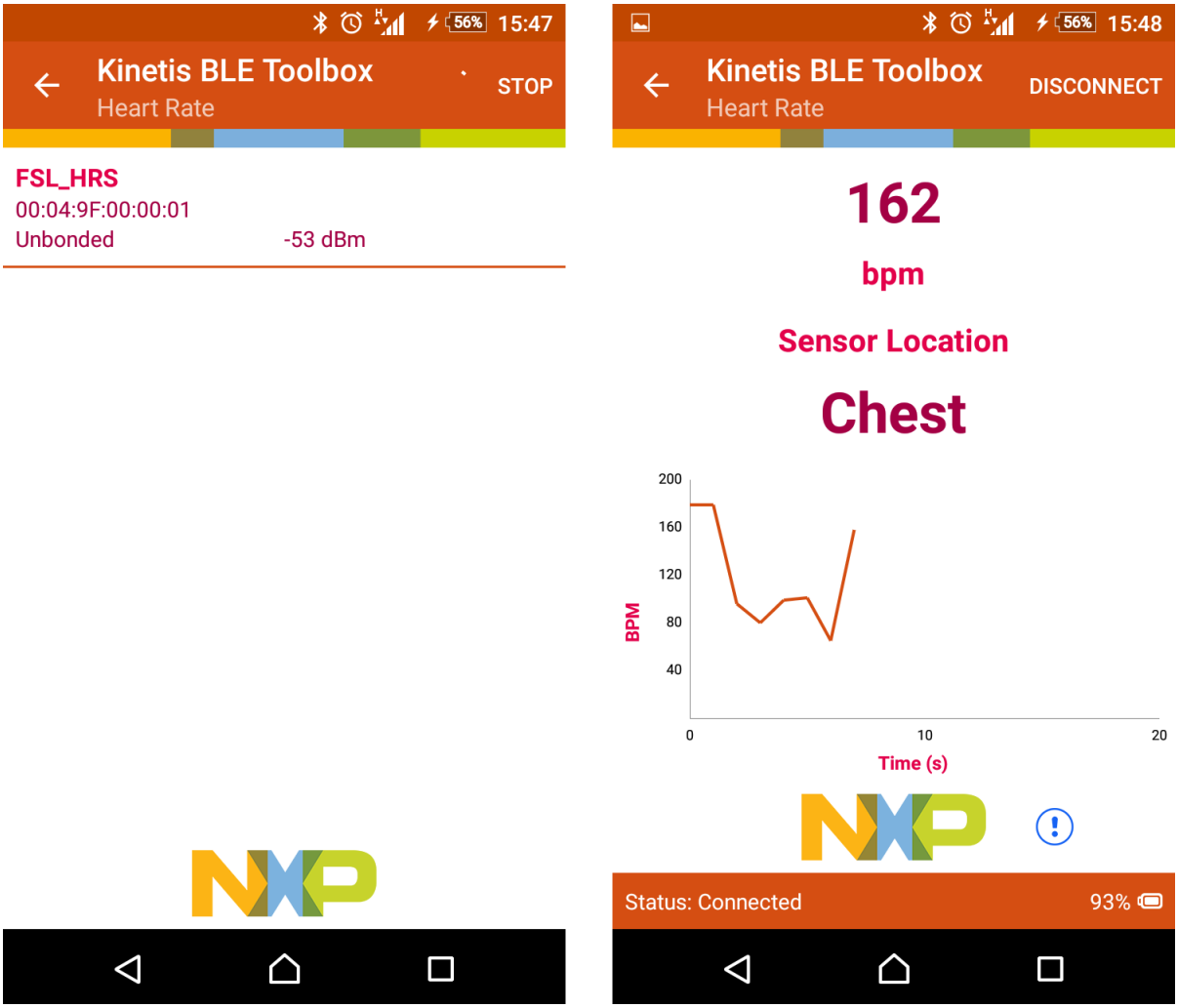

Figure 7. Kinetis BLE Toolbox Heart Rate Monitor Demo

5.2 Beacon

This section presents the user interactions and testing methods for the Beacon application.

5.2.1 Advertising data

The beacons are non-connectable advertising packets that are sent on the three advertising channels. The latter contains the following fields.

Company Identifier (2 bytes): 0x01FF (Freescale ID as defined by the Bluetooth SIG).

- Beacon Identifier (1 byte): 0xBC (Allows identifying an FSL Beacon alongside with Company identifier.
- UUID (16 bytes): Beacon sensor unique identifier.
- A (2 bytes): Beacon application data.
- B (2 bytes): Beacon application data.
- C (2 bytes): Beacon application data.
- RSSI at 1m (1 byte): Allows distance-based applications.

By default, the UUID value is a random value based on the unique identifier of the board.

5.2.2 User interface

After flashing the board, the beacon application starts advertising. When in GAP Discoverable Mode, LED3 is flashing. Hold the SW4 button pressed for 2-3 seconds anytime to stop advertising. In idle mode, all LEDs are flashing. To restart advertising, press the SW4 button.

5.2.3 Usage

The beacon can be tested with any Bluetooth® Smart Ready products available on the market. The Kinetis BLE Toolbox can also be used to showcase the profile functionality, as shown in the image below.

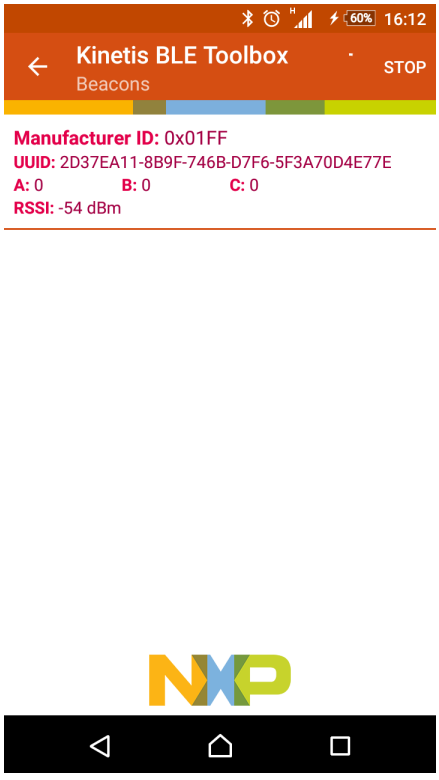

Figure 8. Kinetis BLE Toolbox Beacon Demo

5.3 Blood pressure sensor

This section describes implemented profiles and services, user interactions, and testing methods for the Blood Pressure Sensor application.

5.3.1 Implemented profile and services

The Blood Pressure Sensor application implements a GATT server and the following profile and services.

- Blood Pressure Profile v1.0
- Blood Pressure Service v1.0
- Battery Service v1.0
- Device Information Service v1.1
- Current Time Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect. When the collector configures notifications or indications, the sensor starts sending intermediate cuff pressure and blood pressure measurements. In a 5 second interval, the sensor sends 5 events (4 notifications with the intermediate cuff pressure measurement and one indication with the blood pressure measurement).

5.3.2 User interface

After flashing the board, the sensor is in idle mode (all LEDs flashing). To start advertising, press the SW4 button.

When in GAP Discoverable Mode, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters GAP Discoverable Mode.

The blood and cuff pressure measurement values are generated randomly on the device.

5.3.3 Usage

The blood pressure sensor can be connected to any Bluetooth® Smart Ready products available on the market.

To make the sensor visible, press the SW4 button to start sending advertisements, which causes LED3 to start flashing. The sensor name “FSL_BPS” shows on the device when its scanning is active. A solid LED3 indicates a successful connection between the 2 devices. Hold the SW4 button pressed for 2-3 seconds anytime to initiate disconnect.

If configured, the sensor notifies the application with intermediate cuff pressure each second and blood pressure measurements every 5 seconds. Also, the battery level and various device information is exposed for reading. The Kinetis BLE Toolbox can be used to showcase the profile functionality, as shown in the image below.

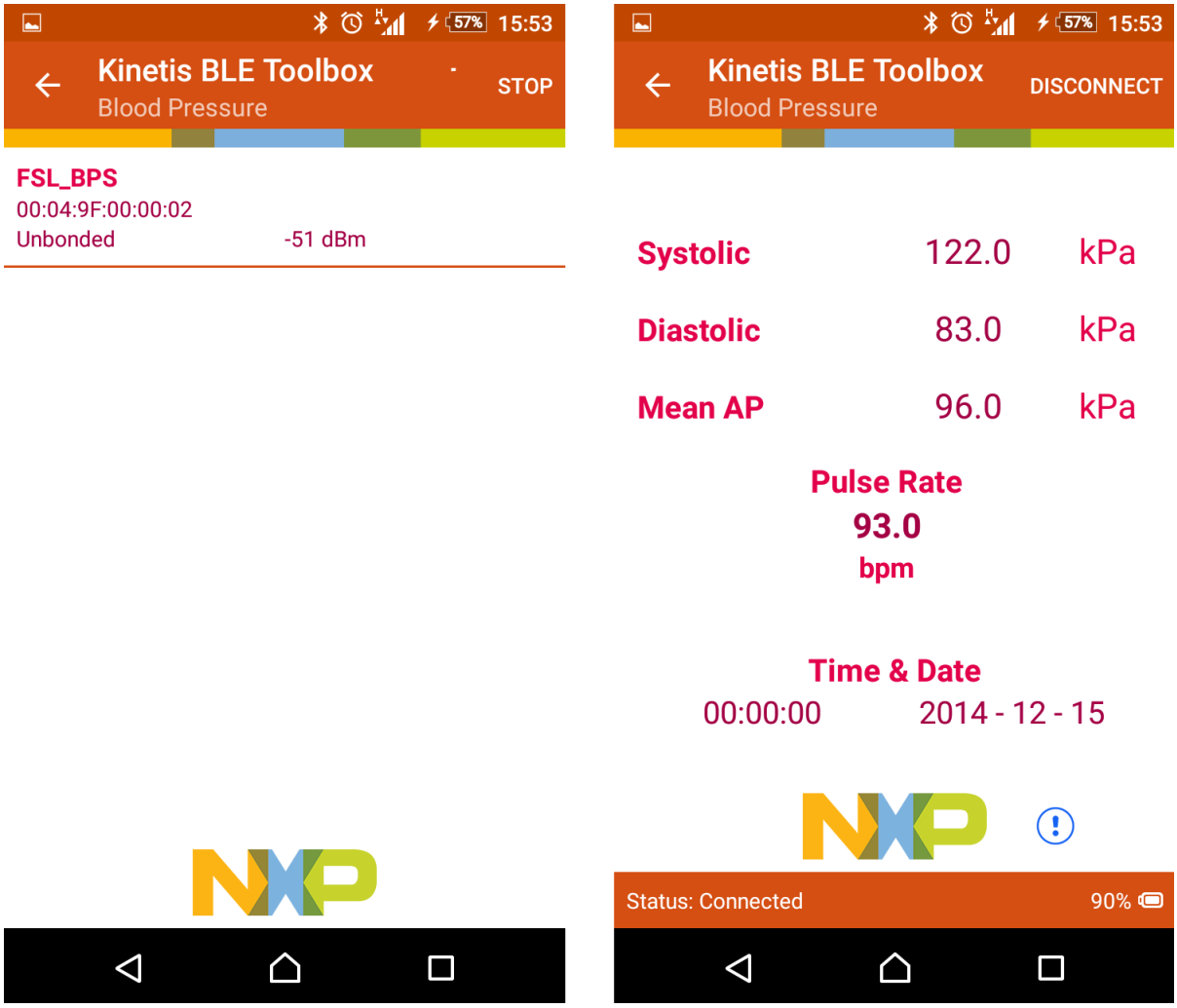

Figure 9. Kinetis BLE Toolbox Blood Pressure Demo

5.4 Glucose Sensor

This section describes implemented profiles and services, user interactions, and testing methods for the Glucose Sensor application.

5.4.1 Implemented profile and services

The Glucose Sensor application implements a GATT server and the following profile and services.

- Glucose Profile v1.0
- Glucose Service v1.0
- Battery Service v1.0
- Device Information Service v1.1
- Current Time Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect. When the collector configures notifications, the sensor can send glucose measurements. The latest three measurements are stored on the sensor device and can be retrieved by the collector.

5.4.2 User interface

After flashing the board, the sensor is in idle mode (all LEDs flashing). To start advertising, press the SW4 button. When in GAP Discoverable Mode, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters GAP Discoverable Mode. When in connection, pressing the SW4 button triggers a new glucose measurement. The glucose measurement values are generated randomly on the device.

5.4.3 Usage

The glucose sensor can be connected to any Bluetooth® Smart Ready products available on the market.

To make the sensor visible, press the SW4 button to start sending advertisements, which causes LED3 to start flashing. The sensor name “FSL_GLS” shows on the device when its scanning is active. A solid LED3 indicates a successful connection between the two devices. Hold the SW4 button pressed for 2-3 seconds anytime to initiate disconnect.

If notifications are configured, the sensor sends the data to the collector when SW4 button is pressed for less than 1 second. The measurement data is stored on the sensor. Also, the battery level and various device information is displayed for reading. The Kinetis BLE Toolbox can be used to showcase the profile functionality, as shown in the image below.

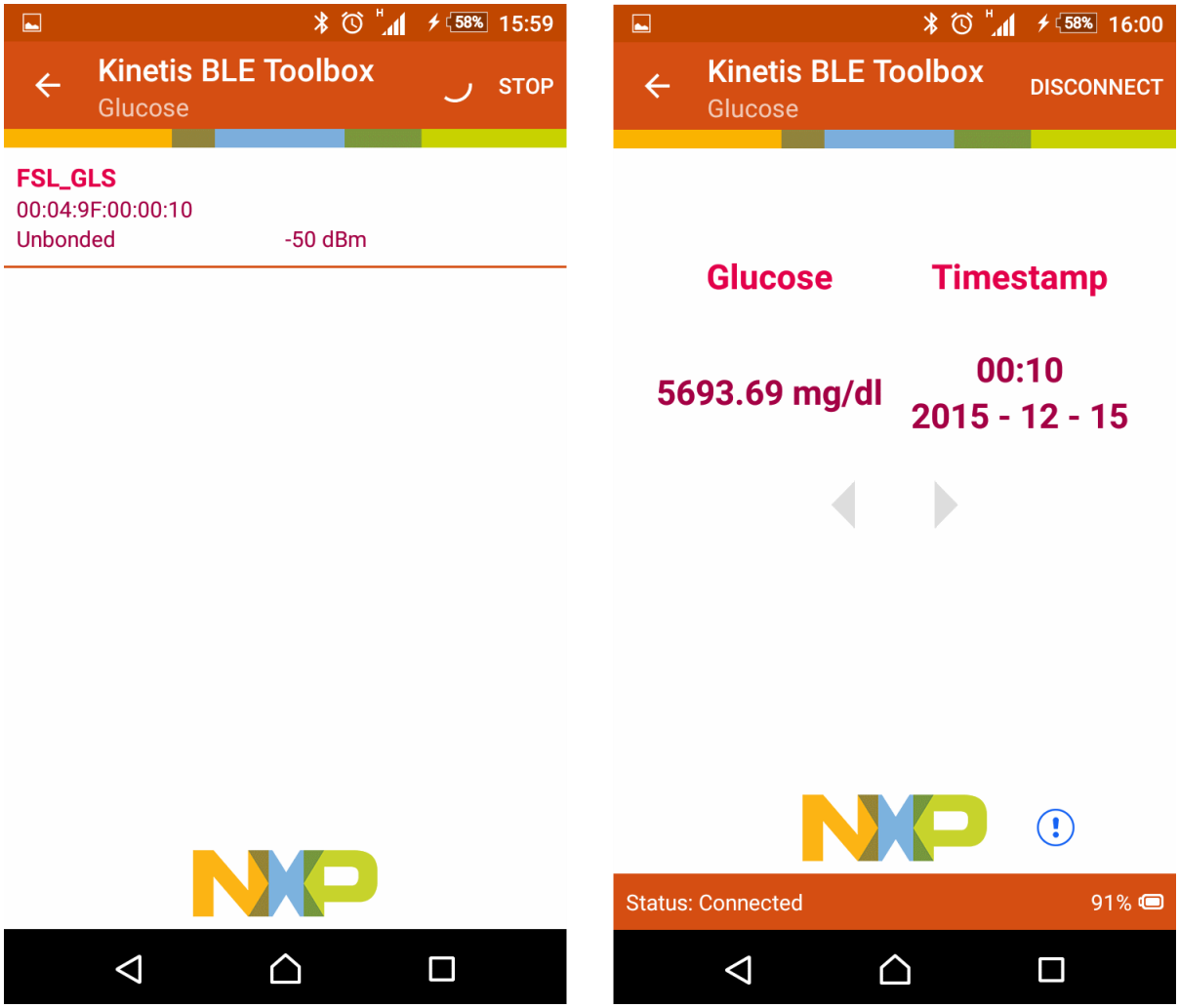

Figure 10. Kinetis BLE Toolbox Glucose Sensor Demo

5.5 Proximity Reporter

This section describes the implemented profiles and services, user interactions and testing methods for the Proximity Reporter application.

5.5.1 Implemented profile and services

The Proximity Reporter application implements a GATT server and the following profile and services.

- Proximity Profile v1.0.1
- Immediate Alert Service v1.0
- TX Power Service v1.0
- Link Loss Service v1.0.1
- Battery Service v1.0
- Device Information Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect.

The Immediate Alert Service shows the alert level control point. GATT clients can trigger an alert on the device. The TX Power Service exposes the TX Power level that is read from the BLE controller. The Link Loss Service exposes a configurable alert level. When the radio link with the central node is suddenly lost, the configured alert is triggered on the reporter node.

5.5.2 User interface

After flashing the board, the sensor is in idle mode (all LEDs flashing). To start advertising, press the SW4 button. When in GAP Discoverable Mode, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters GAP Discoverable Mode.

When a mild alert is triggered on the reporter, RGB LED starts flashing blue. When a high alert is triggered on the reporter, both RGB LED starts flashing green.

5.5.3 Usage

The proximity reporter can be connected to any Bluetooth® Smart Ready products available on the market. To make the sensor visible, press SW4 to start advertising which causes LED3 to start flashing. The sensor name “FSL_PXR” shows on the device when its scanning is active. A solid LED3 indicates a successful connection between the two devices. Hold the SW4 button pressed for 2-3 seconds anytime to initiate disconnect. When a link loss is detected or when an alert is written on the control point, the user interface exposes the alert level on the RGB LED as stated previously.

Also, the battery level and various device information is displayed for reading. The Kinetis BLE Toolbox can be used to showcase the profile functionality, as shown in the image below.

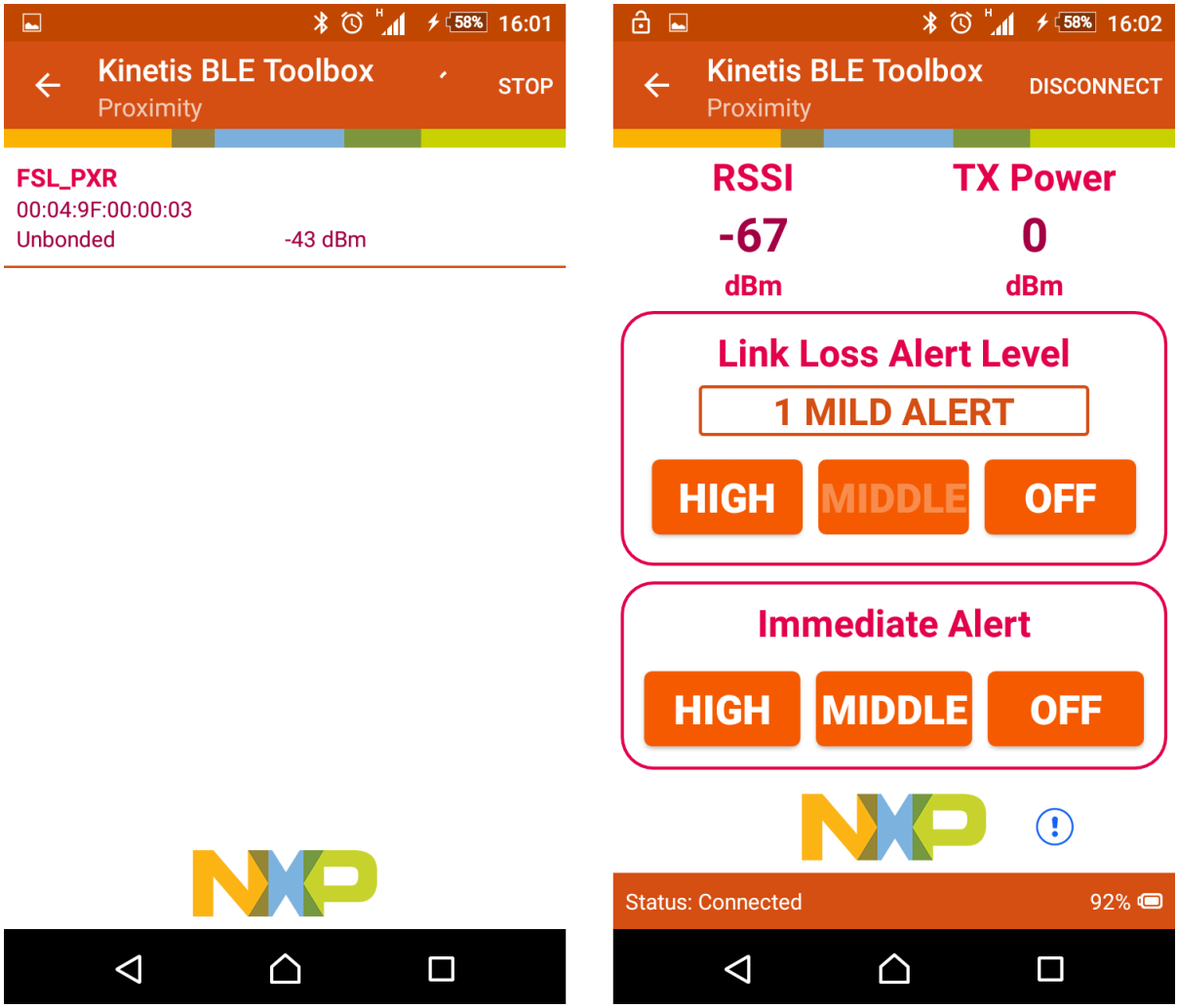

Figure 11. Kinetis BLE Toolbox Proximity Reporter Demo

5.6 HID Device (Mouse)

This section describes implemented profiles and services, user interactions, and testing methods for the HID mouse application.

5.6.1 Implemented profiles and services

The HID Device application implements a GATT server and the following profile and services.

- HID over GATT Profile v1.0
- Human Interface Device Service v1.0
- Battery Service v1.0
- Device Information Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect. Security on the services and bonding is enabled on this device.

When the GATT client configures notification, the application starts sending HID reports every two seconds with the movement of the MOUSE_STEP. The demo moves the cursor in a square pattern between AXIS_MIN and AXIS_MAX. The report contains 3 bytes, one for button status, one for X axis, and one for Y axis. The report descriptor matches the example in chapter E.10 from the USB Device Class Definition for Human Interface Devices (USB HID Specification), Version 1.11.

5.6.2 User interface

After flashing the board, the device is in idle mode (all LEDs flashing). To start advertising, press the SW4 button. When in GAP Discoverable Mode, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 pressed for 2-3 seconds. The node then re-enters GAP Discoverable Mode.

5.6.3 Usage

The HID mouse can be connected to any Bluetooth Smart Ready products available on the market that supports HID devices or to another FRDM-KW41Z platform running the HID Host example (setup steps detailed in the HID Host section).

To make the HID mouse visible, press the SW4 button to start sending advertisements, which causes LED3 to start flashing. The sensor name “FSL_HID” shows on the device when its scanning is active. A solid LED3 indicates a successful connection between the 2 devices. When prompted to enter the pin, type the 999999 passkey.

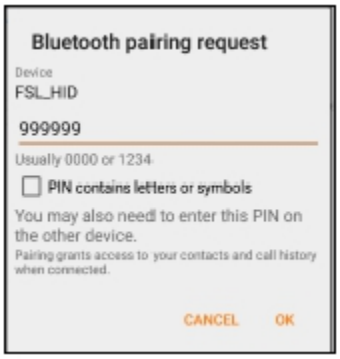

Figure 12. Enter PIN prompt on Android platform

When configured, the HID mouse starts sending HID report, which is configured as explained above, with notifications every 100 milliseconds. The mouse cursor shows a square pattern movement on the screen.

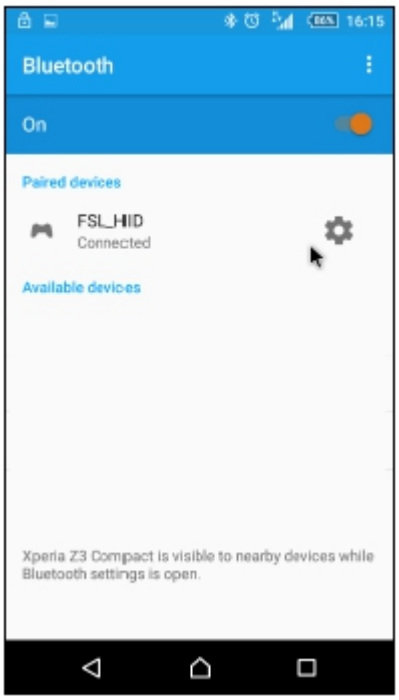

Figure 13. HID Mouse detected by Android platform

5.7 HID Host

This section presents the implemented profiles and services, user interactions, and testing methods for the HID Host application.

5.7.1 Implemented profiles and services

The HID Host application implements a GATT client or server for the following profile and service.

- HID over GATT Profile v1.0
- Battery Service v1.0
- Device Information Service v1.1

The application behaves as a GAP central node. It enters the GAP Limited Discovery Procedure and searches for HID devices to connect to. After connecting with the peripheral, it configures notifications and displays the received HID reports on a terminal connected to the UART port. The application uses pairing with bonding by default. When connected with the HID Device application, it sends the 999999 passcode to the host stack by default.

5.7.2 User interface

After flashing the board, the device is in idle mode (all LEDs flashing). To start scanning, press the SW4 button. When in GAP Limited Discovery Procedure, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters GAP Limited Discovery Procedure.

5.7.3 Usage

The application is built to work only with the HID Device application presented in the previous subchapter and it supports up to 2 peripherals connected at the same time.

1. Open a serial port terminal and connect it to board, in the same manner described in section 5.1.3. The start screen is displayed after the board is reset.
2. Press the SW4 button on the HID Host board to start scanning for devices. Do the same on the HID device board to make it enter discoverable mode. The host connects with the board after it sees it advertise the HID service, connects to it, and configures report notifications. The device then starts sending HID reports, as shown below.

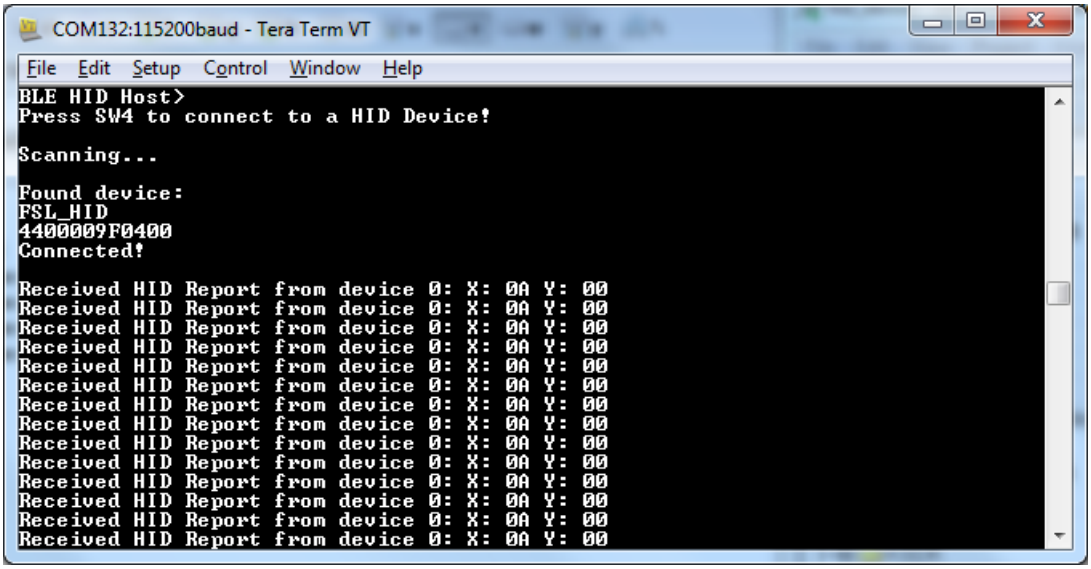

Figure 14. Tera Term – Output Console on HID Host with 1 peripheral connected

3. To connect a second HID device, press again the SW4 button on the HID Host board to start scanning for devices. Do the same on the second HID device board to make it enter discoverable mode. The host connects with the board after it sees it advertise the HID service, connects to it, and configures report notifications. The device then starts sending HID reports. The console will print reports from both devices, as shown below.

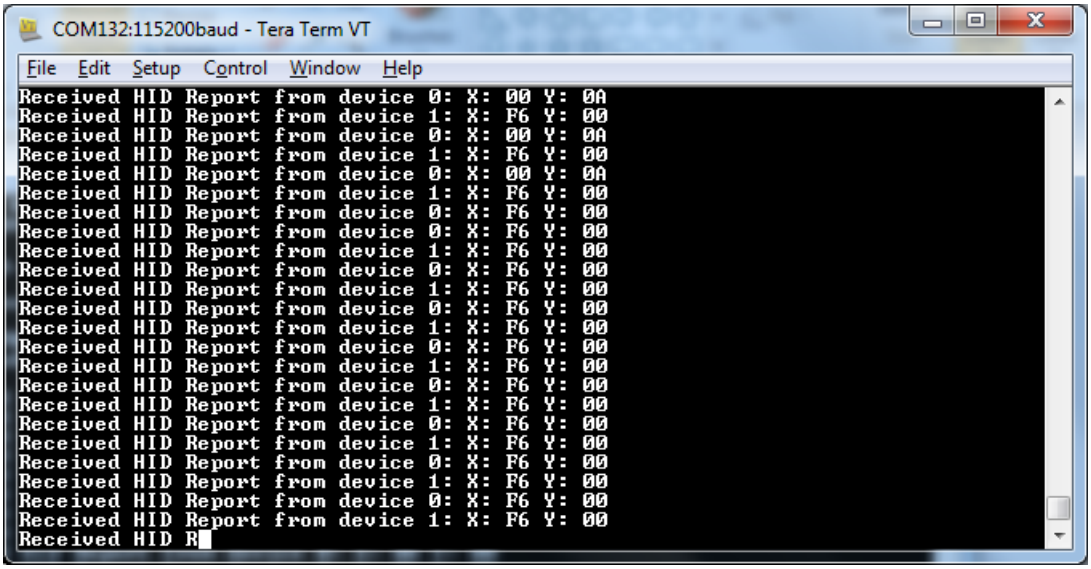

Figure 15. Tera Term – Output Console on HID Host with 2 peripherals connected

5.8 Cycling Speed and Cadence Sensor

This section describes the implemented profiles and services, user interactions, and testing methods for the Cycling Speed and Cadence Sensor application.

5.8.1 Implemented profiles and services

The Cycling Speed and Cadence application implements a GATT server and the following profile and services.

- Cycling Speed and Cadence Profile v1.0
- Cycling Speed and Cadence Service v1.0
- Battery Service v1.0
- Device Information Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect. When the collector configures notifications, the sensor starts sending speed and cadence measurements. The measurement values are generated randomly on the device.

5.8.2 User iterface

After flashing the board, the sensor is in idle mode (all LEDs flashing). To start advertising, press the SW4 button. When in GAP Discoverable Mode, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters GAP Discoverable Mode

5.8.3 Usage

The sensor can be connected to any Bluetooth Smart Ready products available on the market. To make the sensor visible, press the SW4 button to start sending advertisements, which causes LED3 to start flashing. The sensor name “FSL_CSCS” shows on the device when its scanning is active. A solid LED3 indicates a successful connection between the 2 devices. Hold the SW4 button pressed for 2-3 seconds anytime to initiate disconnect.

If configured, the sensor notifies the application with speed and cadence measurements every second. Also, the battery level and various device information is exposed for reading. The Kinetis BLE Toolbox can be used to showcase the profile functionality, as shown in the image below.

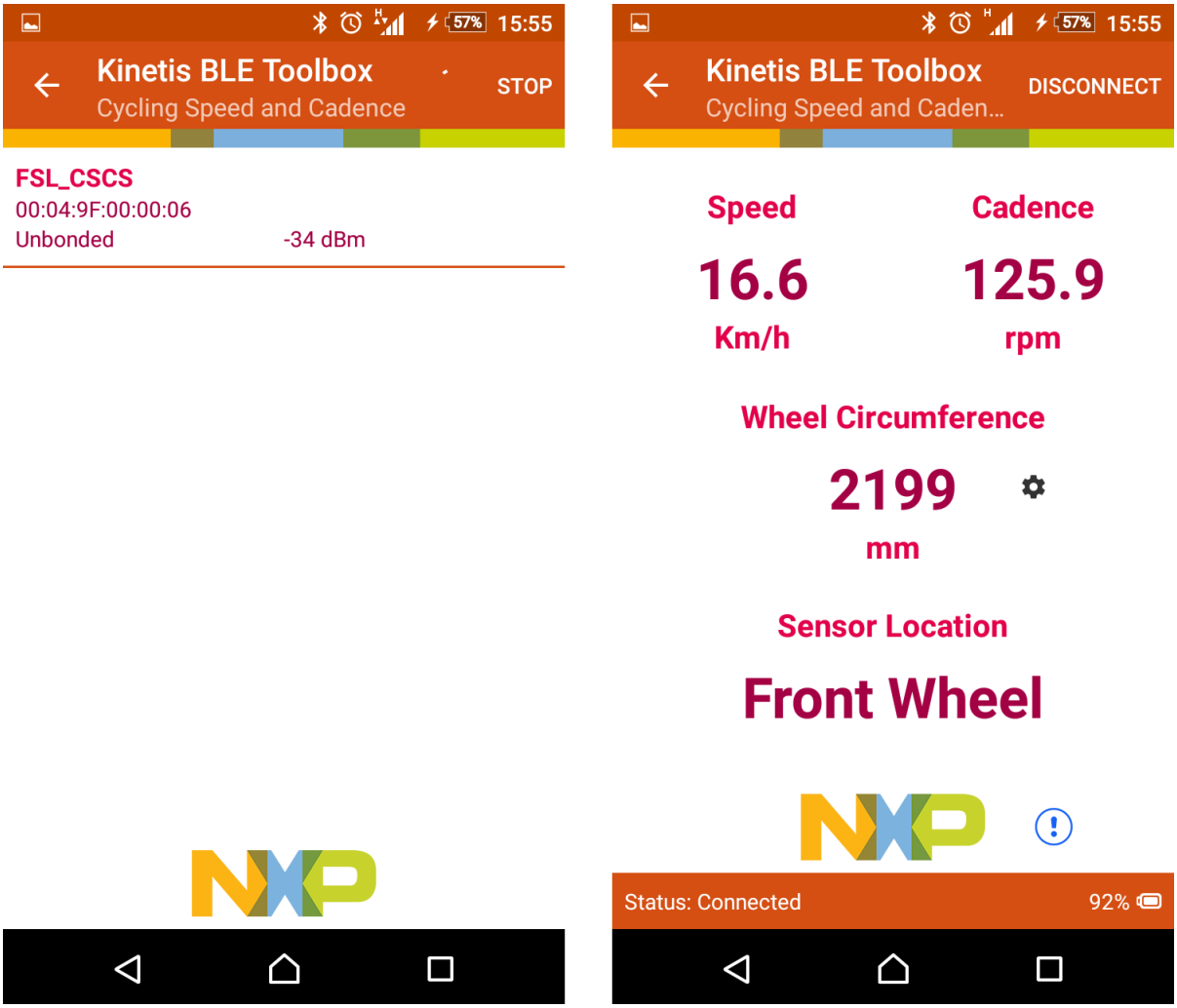

Figure 16. Kinetis BLE Toolbox Cycling Speed and Cadence Sensor Demo

5.9 Cycling Power Sensor

This section presents the implemented profiles and services, user interactions, and testing methods for the Cycling Power Sensor application.

5.9.1 Implemented profiles and services

The Cycling Power Sensor application implements a GATT server and the following profile and services.

- Cycling Power Profile v1.0
- Cycling Power Service v1.0
- Battery Service v1.0
- Device Information Service v1.1
- Current Time Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect. When the collector configures notifications and/or indications, the sensor starts sending power measurements. The measurement values are generated randomly on the device.

5.9.2 User interface

After flashing the board, the sensor is in idle mode (all LEDs flashing). To start advertising, press the SW4 button. When in GAP Discoverable Mode, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters GAP Discoverable Mode.

Because all the fields in the power measurement cannot fit into one single ATT frame, the user can toggle the optional fields by pressing the SW3 button.

5.9.3 Usage

The sensor can be connected to any Bluetooth Smart Ready products available on the market. To make the sensor visible, press the SW4 button to start sending advertisements, which causes LED3 to start flashing. The sensor name “FSL_CPS” shows on the device when its scanning is active. A solid LED3 indicates a successful connection between the 2 devices. Hold the SW4 button pressed for 2-3 seconds anytime to initiate disconnect. If configured, the sensor notifies the application with power measurements. Also, the battery level and various device information is displayed for reading.

To demonstrate profile functionality, any bike computer application (from Health & Fitness category) can be installed on Apple iOS or Android OS handheld devices that support Bluetooth Low Energy.

5.10 Running Speed and Cadence Sensor

This section describes the implemented profiles and services, user interactions, and testing methods for the Running Speed and Cadence Sensor application.

5.10.1 Implemented profiles and services

The Running Speed and Cadence application implements a GATT server and the following profile and services.

- Running Speed and Cadence Profile v1.0
- Running Speed and Cadence Service v1.0
- Battery Service v1.0
- Device Information Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect. When the collector configures notifications, the sensor starts sending speed and cadence measurements. The measurement values are generated randomly on the device.

5.10.2 User interface

After flashing the board, the sensor is in idle mode (all LEDs flashing). To start advertising, press the SW4 button. When in GAP Discoverable Mode, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters GAP Discoverable Mode.

When the node is connected, the user can toggle the running status (running/walking) by pressing the SW3 button. When the status is set to “walking” the RGB LED turns solid. When the status is set to “running”, the RGB LED is not lit.

5.10.3 Usage

The sensor can be connected to any Bluetooth® Smart Ready products available on the market. To make the sensor visible, press the SW4 button to start sending advertisements, which causes LED3 to start flashing. The sensor name “FSL_RSCS” shows on the device when its scanning is active. A solid LED3 indicates a successful connection between the two devices. Hold the SW4 button pressed for 2-3 seconds anytime to initiate disconnect.

If configured, the sensor notifies the application with speed and cadence measurements every second. Also, the battery level and various device information is exposed for reading. The Kinetis BLE Toolbox can be used to showcase the profile functionality, as shown in the image below.

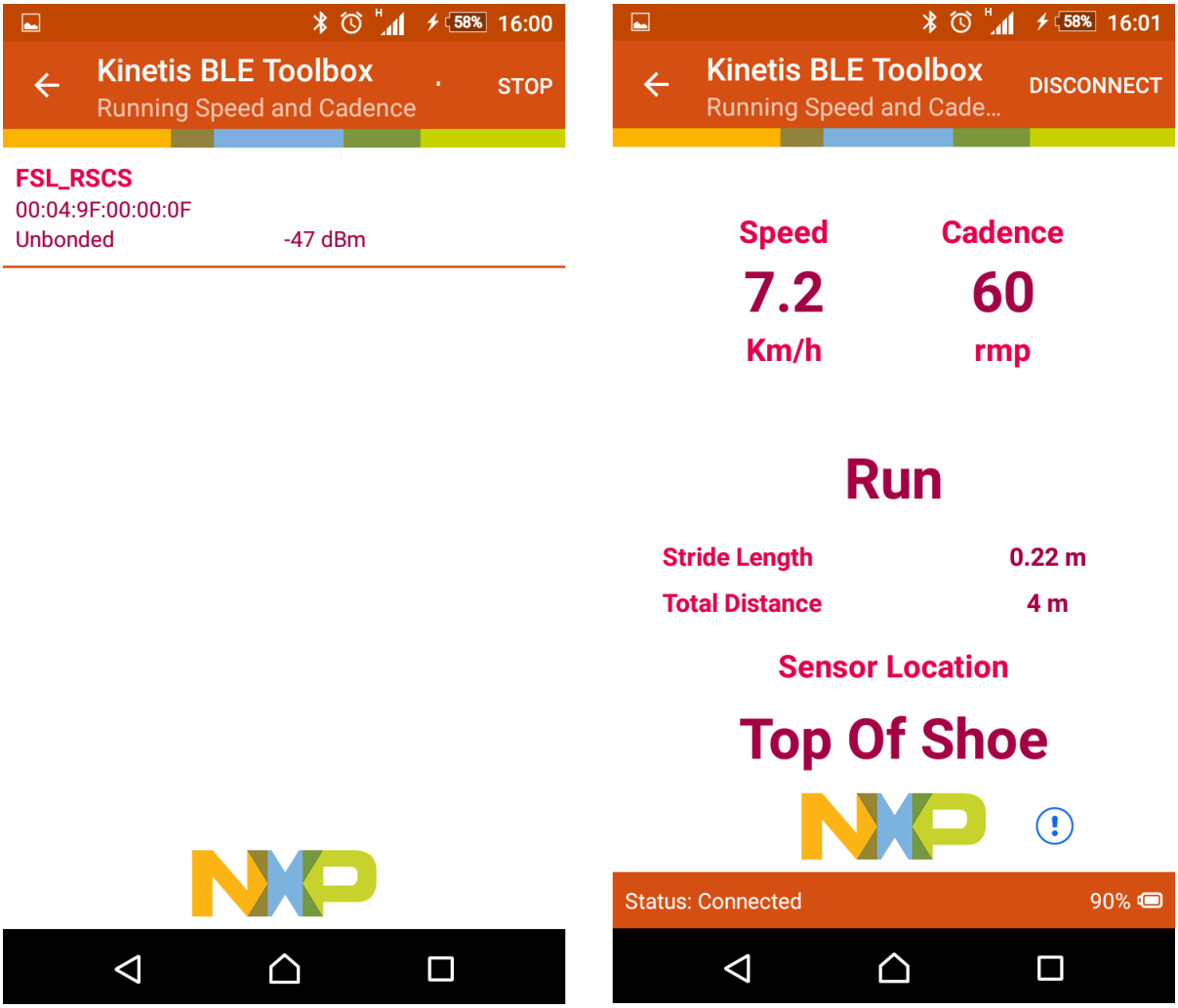

Figure 17. Kinetis BLE Toolbox Running Speed and Cadence Demo

5.11 Health Thermometer

This section presents the implemented profiles and services, user interactions, and testing methods for the Health Thermometer application.

5.11.1 Implemented profiles and services

The Health Thermometer application implements a GATT server and the following profile and services.

- Health Thermometer Profile v1.0
- Health Thermometer Service v1.0
- Battery Service v1.0
- Device Information Service v1.1
- Current Time Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect. When the collector configures notifications or indications, the sensor starts sending intermediate temperature and temperature measurements.

In a 5 second interval, the sensor sends 5 events (4 notifications with the intermediate temperature and one indication with the temperature measurement). The intermediate temperature and temperature measurements are generated randomly on the device.

5.11.2 User interface

After flashing the board, the sensor is in idle mode (all LEDs flashing). To start advertising, press SW4. When in GAP Discoverable Mode LED3 is flashing. When the central node connects to the peripheral LED3 turns solid. Hold the SW4 button pressed for 2-3 seconds anytime to initiate disconnect. The node re-enters the GAP Discoverable Mode.

5.11.3 Usage

The sensor can be connected to any Bluetooth® Smart Ready products available on the market. To make the sensor visible, press the SW4 button to start sending advertisements, which causes LED3 to start flashing. The sensor name “FSL_HTS” shows on the device when its scanning is active. A solid LED3 indicates a successful connection between the two devices. Hold the SW4 button pressed for 2-3 seconds anytime to initiate disconnect.

If configured, the sensor notifies the application with temperature measurements. Also, the battery level and various device information is exposed for reading. The Kinetis BLE Toolbox can be used to showcase the profile functionality, as shown in the image below.

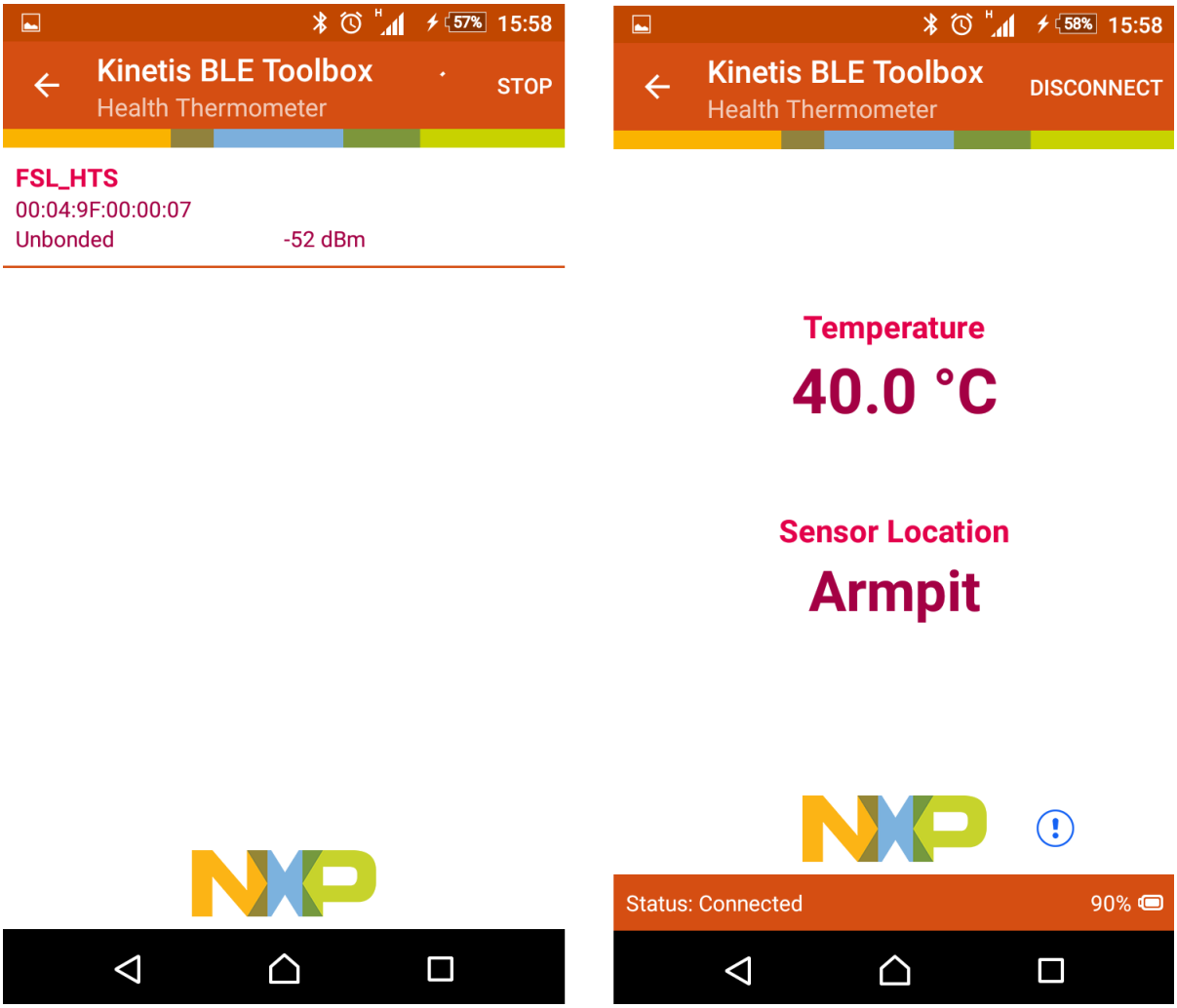

Figure 18. Kinetis BLE Toolbox Health Thermometer Demo

5.12 Low Power Temperature Sensor and Collector

This section describes the implemented profiles and services, user interactions, and testing methods for the temperature sensor application.

5.12.1 Implemented profiles and services

The Temperature Sensor application implements a GATT server, a custom profile and the following services.

- Temperature Service (UUID: 01ff0200-ba5e-f4ee-5ca1-eb1e5e4b1ce0)
- Battery Service v1.0
- Device Information Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect and configure notifications for the temperature value.

The Temperature service is a custom service that implements the Temperature characteristic (UUID: 0x2A6E) with a Characteristic Presentation Format descriptor (UUID: 0x2904), both defined by the Bluetooth SIG.

The Temperature Collector application implements a GATT client or server for the following profile and services.

- Temperature Service (UUID: 01ff0200-ba5e-f4ee-5ca1-eb1e5e4b1ce0)
- Battery Service v1.0
- Device Information Service v1.1

The application behaves as a GAP central node. It enters GAP Limited Discovery Procedure and searches for sensor devices to pair with. After pairing with the peripheral, it configures notifications and displays temperature values on a terminal connected to the UART port.

Both application uses pairing with bonding by default. When connected with the Low Power Temperature Sensor application, the collector sends the 999999 passcode to the host stack by default.

5.12.2 User interface

After flashing the board, both nodes enter DSM 3. All LEDs are off. When the node is awake and communicating, LED3 is on. To wake up the node press the SW4 button.

5.12.3 Usage

The setup requires two FRDM-KW41Z platforms, one for the temperature sensor and one for the temperature collector.

1. Open a serial port terminal and connect it to the temperature collector board, in the same manner as described in 5.1.3. The start screen is displayed after the board is reset. At first the LEDs are off on both devices.
2. To start advertising on the sensor, press the SW4 button and LED3 lights up. The sensor enters DSM1, which means that the MCU wakes up on any packet from the Link layer, in this case the connect request. If no connection is established in an interval of 10 seconds, the sensor stops advertising and enters DSM 3 again. LED3 turns off.
3. To start scanning on the collector, press the SW4 button and LED3 lights up. The device wakes up, scans and connects to a compatible sensor device. If no connection is established within 5 seconds, the collector stops scanning and enters DSM 3 again. LED3 turns off.
4. If the collector connects to a sensor node, it bonds (if no bond was previously made), does service discovery and configures notification (only the first time it connects with the sensor) and waits for notifications from the sensor for 5 seconds. If no data is sent, the node disconnects and re-enters DSM3. The sensor exits low power and sends a notification with the value of the temperature read through an ADC from the thermistor on the KW41Z.

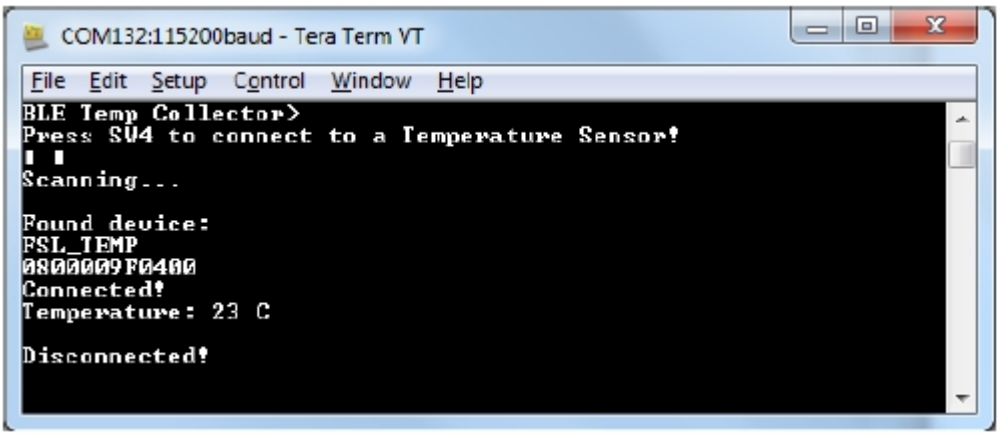

Figure 19. Tera Term – Output Console on Temperature Collector

5. Subsequent key pressing triggers other notifications for the collector. If no key is pressed in an interval of 3 seconds, the sensor node disconnects and re-enters DSM3.

5.13 IPv6 Router and Node

For information about the IPv6 Node and Router applications, see the *Bluetooth® Low Energy Transport for IPv6 Datagrams* (document BLEIP6UG).

5.14 Alert Notification Server

This section describes the implemented profiles and services, user interactions, and testing methods for the Alert Notification Server application.

5.14.1 Implemented profile and services

The Alert Notification application implements a GATT client or server for the following profile and services.

- Alert Notification Profile v1.0
- Alert Notification Service v1.0
- Battery Service v1.0
- Device Information Service v1.1
- Current Time Service v1.1
- Next DST Change Service v1.0
- Reference Time Update Service v1.0

The application behaves as a GAP central node. It enters the GAP Limited Discovery Procedure and searches for Alert Notification Clients or Time Clients to connect to. After connecting with the peripheral, it waits for notifications to be configured.

5.14.2 User interface

After flashing the board, the device is in idle mode (all LEDs flashing). To start scanning, press the SW4 button. When in GAP Limited Discovery Procedure, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters GAP Limited Discovery Procedure.

When in connection and if notifications are configured, pressing the SW3 button triggers a new alert notification (“New mail”) to be sent to the GATT client and adjusts the time of the device by +1 hour. Holding the SW3 button for 2-3 seconds triggers a new unread alert status notification (missed call) to be sent to the same GATT client.

5.14.3 Usage

The application can be tested against the Profile Tuning Suite (PTS) black-box testing tool from the Bluetooth SIG.

5.15 Wireless UART

This section describes the implemented profiles and services, user interactions, and testing methods for the Wireless UART application.

5.15.1 Implemented profile and services

The Wireless UART application implements both the GATT client and server for the custom Wireless UART profile and services.

- Wireless UART Service (UUID: 01ff0100-ba5e-f4ee-5ca1-eb1e5e4b1ce0)
- Battery Service v1.0
- Device Information Service v1.1

The Wireless UART service is a custom service that implements a custom writable ASCII Char characteristic (UUID: 01ff0101-ba5e-f4ee-5ca1-eb1e5e4b1ce0) that holds the character written by the peer device.

The application behaves at first as a GAP central node. It enters GAP Limited Discovery Procedure and searches for other Wireless UART devices to connect. If the node fails to find any peripherals within seconds, it changes its role to a GAP peripheral. It enters GAP General Discoverable Mode and waits for a GAP central node to connect.

5.15.2 User interface

After flashing the board, the device is in idle mode (all LEDs flashing). To start scanning, press the SW4 button. When in GAP Limited Discovery Procedure of GAP General Discoverable Mode, LED3 is flashing. When the node connects to a peer device, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters GAP Limited Discovery Procedure.

5.15.3 Usage

The application is built to work with another FRDM-KW41Z platform running the same example or with the Wireless UART from the Kinetis BLE Toolbox application. When testing with two boards do the following.

1. Open a serial port terminal and connect them to the two boards, in the same manner described in section 5.1.3. The start screen is blank after the board is reset.
2. Press the SW4 button on the first board to start scanning for devices. Do the same on the second board. After 3 seconds, the first board enters GAP General Discoverable Mode and the second board connects.
3. As soon as the LED3 turns solid on both devices, the user can start writing in one of the consoles. The text appears on the other terminal, as shown in the figure below.

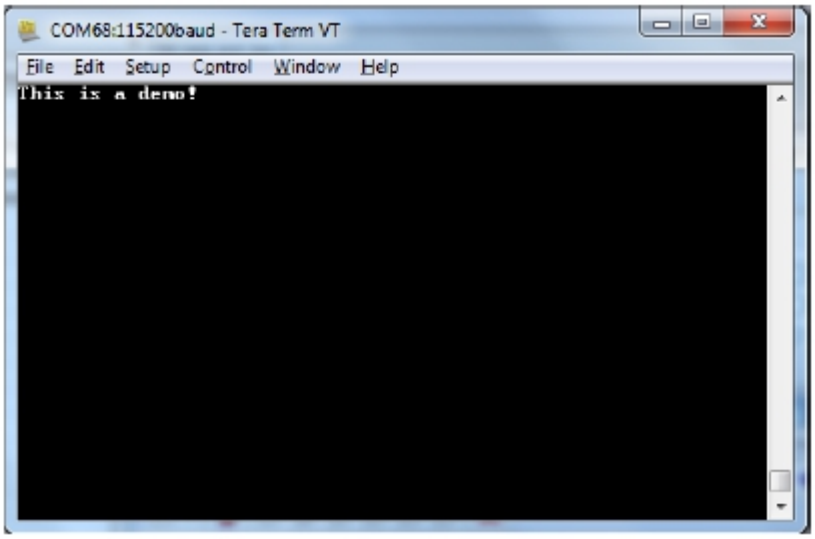

Figure 20. Tera Term – received text on Wireless UART

When testing with one board and the Kinetis BLE Toolbox do the following.

1. Open a serial port terminal and connect the board in the same manner described in section 5.1.3. The start screen is blank after the board is reset.
2. Press the SW4 button. After 5 seconds, the board enters GAP General Discoverable Mode and the Kinetis BLE Toolbox app can connect.

5.16 BLE Shell

This section describes the functionality, user interactions, and testing methods for the BLE Shell Application.

5.16.1 Implemented stack features

The BLE Shell Application implements a console application that allows the user to interact with a full feature Bluetooth Low Energy stack library. It implements All GAP roles and both GATT client and server. Enabling these roles can be done using shell commands.

5.16.2 Implemented profile and services

The application implements a dynamic GATT database. The user can add services at runtime and also erase the database contents. The database is always populated with the GAP and GATT services. These services cannot be erased. The user can dynamically add the following services.

- Heart Rate Service (UUID: 0x180D)
- Battery Service (UUID: 0x180F)
- Device Information Service (UUID: 0x180A)
- Internet Support Profile Service (0x1820)

5.16.3 User interface

After flashing the board, the device is in idle mode (all LEDs flashing). The interaction with the board is done entirely by using the shell commands via the serial communication terminal.

5.16.4 Usage

The application is built to work with any other BLE device. To showcase the functionality, two FRDM-KW41 platforms are used in the following setup.

1. Open a serial port terminal and connect them to the two boards, in the same manner described in section 5.1.3. The start screen is displayed after the board is reset. All LEDs are flashing on both devices.
2. Configure one of the devices as a GAP peripheral and a Heart Rate server. Change name to HRS. Start advertising on this device.

```
Kinetis BLE Shell>gap devicename HRS
--> GATTDB Event: Attribute Written
HRS>gap advdata 1 6
--> GAP Event: Advertising data successfully set.
HRSvgap advdata 8 HRS
--> GAP Event: Advertising data successfully set.
HRS>gap advstart
--> GAP Event: Advertising started.
```

```
HRS>gattdb addservice 0x180D
--> Heart Rate
- Heart Rate Measurement Value Handle: 14
- CCCD Handle: 15
- Body Sensor Location Value Handle: 17
- Heart Rate Control Point Value Handle: 19
--> GATTDB Event: Service Added in database.
```

3. Configure the other device as a GAP central. Change name as Collector. Start scanning and connect to the HRS device by selecting the corresponding device index. In the example below, the HRS device is device number 2.

```
Kinetis BLE Shell>gap devicename Collector
--> GATTDB Event: Attribute Written
Collector>gap scanstart
-> GAP Event: Scan started.
Collector>
--> GAP Event: Found device 0 : 880F102F500E 0 dBm
--> GAP Event: Found device 1 : FSL_CSCS 00049F000006 0 dBm
--> GAP Event: Found device 2 : HRS 00049F0000FF 0 dBm
Collector>gap connect 2
-> GAP Event: Scan stopped.
Collector>
--> GAP Event: Connected
```

4. Optionally, the devices can be paired. On the collector initiate the pairing.

```
Collector>gap pair
--> Pairing...
--> GAP Event: Passkey is 792910
```

5. On the HRS, enter the pin displayed by the collector, as shown below.

```
HRS>
--> GAP Event: PIN required
HRS>gap enterpin 792910
--> GAP Event: Device Paired.
```

6. On the Collector, start service discovery. The device discovers the GAP, GATT and Heart Rate services.

```
Collector>gatt discover -all
--> Discovered primary services: 3
--> Generic Access Start Handle: 1 End Handle: 7
- Device Name Value Handle: 3
- Appearance Value Handle: 5
- Peripheral Preferred Connection Parameters Value Handle: 7
--> Generic Attribute Start Handle: 8 End Handle: 11
- Service Changed Value Handle: 10
- Client Characteristic Configuration Descriptor Handle: 11
--> Heart Rate Start Handle: 12 End Handle: 19
- Heart Rate Measurement Value Handle: 14
- Client Characteristic Configuration Descriptor Handle: 15
- Body Sensor Location Value Handle: 17
- Heart Rate Control Point Value Handle: 19
```

7. Configure the HRS to send notifications by writing the CCCD from the Collector. Send a GATT write command with value 1 to the CCCD handle discovered, 15.

```
Collector>gatt write 15 0x0001
```

8. Send heart rate measurement notifications from the HRS device by using the value handle obtained after adding the service in the previous step.

```
HRS>gatt notify 14
```

9. A notification appears on the Collector console.

```
Collector>
--> GATT Event: Received Notification
Handle: 14
Value: B400
```

5.17 Over the Air Programming (OTAP)

This section describes the implemented profiles and services, user interactions, and testing methods for the BLE OTAP application.

5.17.1 Implemented profile and services

The BLE OTAP applications implement the GATT client and server for the custom BLE OTAP profile and service.

- BLE OTAP Service (UUID: 01ff5550-ba5e-f4ee-5ca1-eb1e5e4b1ce0)

The BLE OTAP Service is a custom service which has 2 characteristics.

- OTAP Control Point Characteristic (UUID: 01ff5551-ba5e-f4ee-5ca1-eb1e5e4b1ce0). This characteristic can be written and indicated to exchange OTAP Commands between the OTAP Server and the OTAP Client. Data chunks are not transferred using this characteristic.
- OTAP Data Characteristic (UUID: 01ff5552-ba5e-f4ee-5ca1-eb1e5e4b1ce0). This characteristic can be written without response by the OTAP Server to transfer image file data chunks to the OTAP Client only when an image block transfer is requested via the ATT transfer method. Data chunks can also be transferred via the L2CAP credit based PSM channels method.

The demo runs using 4 applications: an OTAP Client embedded application, an OTAP Bootloader embedded application, an OTAP Server embedded application, and an OTAP Server PC application. The OTAP Client embedded application has two versions, an ATT version and a L2CAP version each using a different transfer method.

The embedded OTAP Server application is a GAP Central application which scans for devices advertising the BLE OTAP service. After it finds one, it connects to it and configures the OTAP Control Point CCC Descriptor to receive ATT Indications from the device then it waits for OTAP commands from this device.

Once commands start arriving from the OTAP Client via ATT Indications the OTAP Server relays them via serial interface to a PC application which responds. The responses are then sent back to the OTAP Client by writing the OTAP Control Point Characteristic. The embedded OTAP Server application effectively acts as a relay between the OTAP Client to which the image is sent over the air and the OTAP Server PC application which has an OTAP image file constructed using a binary .srec image or a .bin image.

The OTAP Client is a GAP Peripheral which advertises the BLE OTAP Service and waits for a connection from an OTAP Server. After an OTAP Server connects, the OTAP Client waits for it to write the OTAP Control Point CCCD and then starts

sending commands via ATT Indications. If the OTAP Client is configured to ask the data transfer via the L2CAP CoC PSM, it registers and tries to connect a predetermined L2CAP PSM before sending any commands to the OTAP Server.

5.17.2 User interface

After flashing two boards with the OTAP Server and OTAP Client applications respectively, the devices are in idle mode (all LEDs flashing). To start advertising, press the SW4 button on the OTAP Client. To start scanning, press the SW4 button on the OTAP Server. After the two devices connect and start exchanging commands. LED1 becomes solid on the OTAP Server and on the OTAP Client.

After the embedded applications are flashed to the boards the OTAP Server PC application must be started. The application creates an OTAP image file using the provided executable .srec or .bin file, connects to the embedded OTAP Server via the configured serial interface and waits for commands. The application shows details about the creation of the image file and allows the configuration of the OTAP upgrade image file header. A log view is present where the interactions between the OTAP Client and the OTAP Server are shown.

5.17.3 Usage with test tool for Connectivity products

This is a list of requirements.

- Test Tool for Connectivity Products 12.5.0 or newer – Test Tool on www.nxp.com
- Serial COM port drivers – these are board-specific

These are the steps to run the application.

1. Flash the OTAP Server onto a KW4x board, and the OTAP Bootloader and the OTAP Client to another KW4x board. Make sure the board running the OTAP Server is connected to your PC and your PC has appropriate drivers for the USB to serial device on that board.

   NOTE

   The OTAP Bootloader must be programmed separately into the MCU, before programming the OTAP Client application.

2. Create the application to send over the air. The executable must be provided in the .srec or .bin format. The .srec format executable can be obtained by using the IAR Output Converter and setting the output format to Motorola as shown below. The created .srec application image must be offset to begin after the Bootloader region. To offset the application copy the settings from the Linker->Config tab of the otap_client_l2cap_credit or otap_client_att example applications project properties. For more details, see the *BLE Application Developer's Guide* (BLEADG).

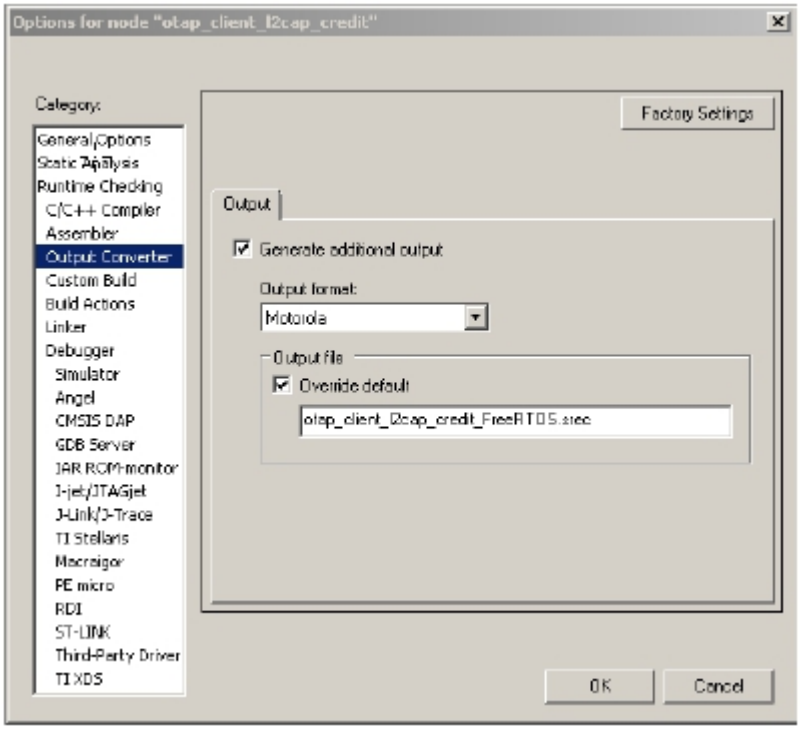

Figure 21. Output Converter Dialog

NOTE

For the .bin file format the Binary option must be selected from the Output format drop-down menu.

3. Start the Test Tool application. If you have the proper drivers installed and the OTAP Server board is connected to the PC, then its corresponding serial port shows up in the USB/UART Active devices view on the Start Page of the Test Tool software as shown below.

Figure 22. Test Tool Start Page - OTAP Server USB/UART

4. Go to the OTA Updates menu in Test Tool and choose the OTAP BLE option to open the BLE over-the-air update window.

Figure 23. Test Tool OTAP BLE

5. Load the image file into the application and configure the image file header and start the OTAP Server

   - In the Image File Information box of the OTAP BLE application from Test Tool press the “Browse” button and go to the .srec or .bin file containing the image to be sent to the OTAP Client. The .srec or .bin file and the OTA Header configuration options from the same box is used by the application to build the OTAP Image File which is sent over-the-air. The default values of the OTA Header configuration work out of the box for the OTAP demo applications. For details about these configuration options, see the BLE Application Developer’s Guide document (BLEADG). After the .srec or .bin file is chosen a pop-up window asks to choose the target processor (this is used to correctly configure the Sector Bitmap sub-element of the OTAP Image File). Choose the correct processor and press “OK”.

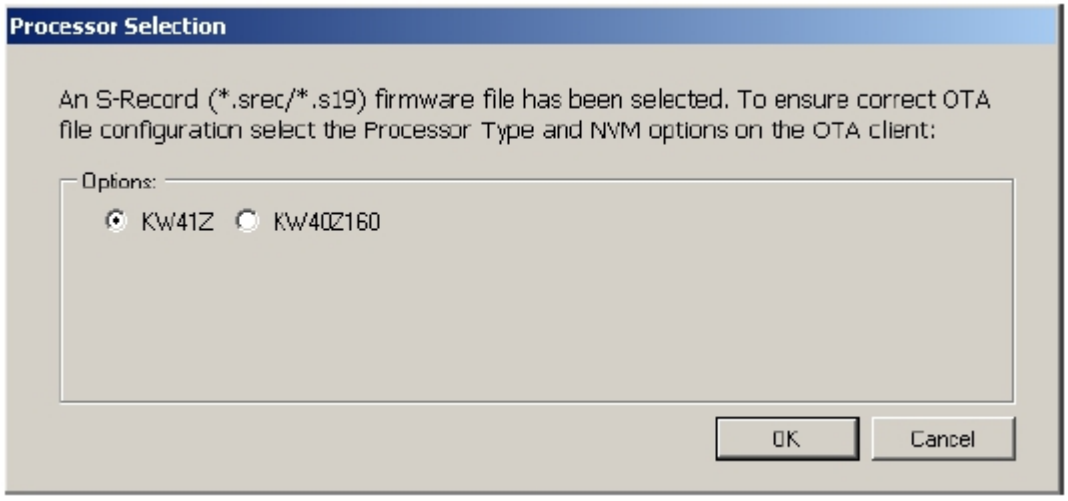

Figure 24. Test Tool OTAP BLE Processor Selection

   - After the image is loaded go to the OTA Server Image Loading box, select the correct COM Port for the OTAP Server board and the default baudrate of 115200 and press the “Connect to OTAP Server Device” button. If the connection is successful then the Message Log shows this. If the image is loaded before connecting to the OATP Server COM Port, then the application’s OTAP Server starts automatically. If the connection to the COM Port is established before the image is loaded, then the “Start OTAP” button needs to be pressed to start the application’s OTAP Server. For details, see the annotated screenshot below.

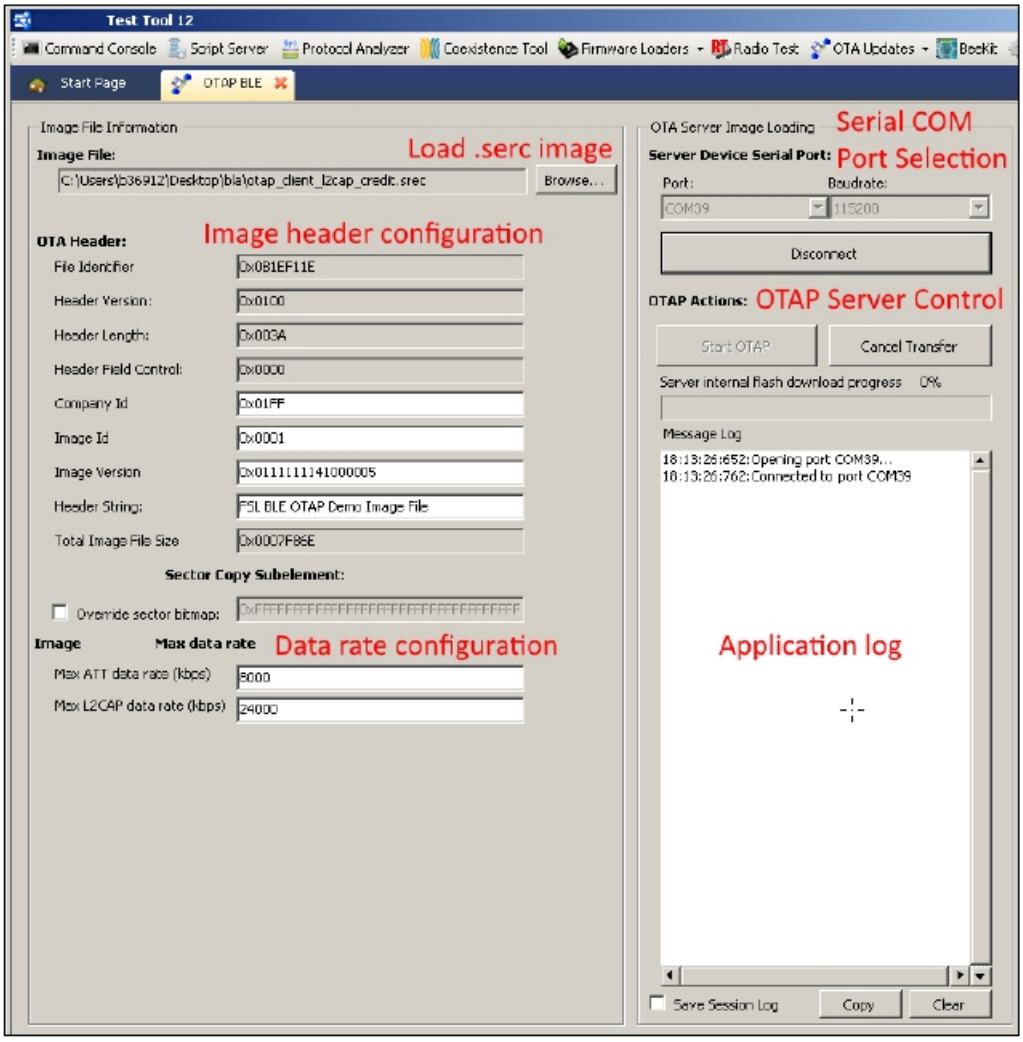

Figure 25. Test Tool BLE OTAP Application Overview

- Before starting the image transfer process, the data rate must be configured for each transfer method (ATT or L2CAP CoC). The image chunks of a block are sent over the serial interface and over-the-air without waiting for confirmation and if the data rate is not configured correctly errors can appear in the transfer process which can slow it down significantly. The optimal data rate depends on multiple factors like: distance between boards, type of antenna, performance of the RF circuitry between the radio and antenna, type and level of noise in the environment, speed of the storage medium in which the image is saved on the OTAP Client, serial driver delay between PC and OTAP Server board and other factors. If the data rate is too high, then the OTAP Client receives a new chunk before it can process the previous one and it sends an “Unexpected Chunk Sequence Number” error and restart the transfer of the current block from where it left off. If the channel is too noisy the transmitter can be flooded and some chunks may not reach the client triggering a similar type of error. The default data rate values should work for most configurations.

6. Start the embedded applications by pressing SW4 first on the OTAP Client and then on the OTAP Server. The transfer progress and transfer-related messages and/or errors are shown in the application window. The duration of the transfer depends on the size of the image and the chosen data rate and transfer method.

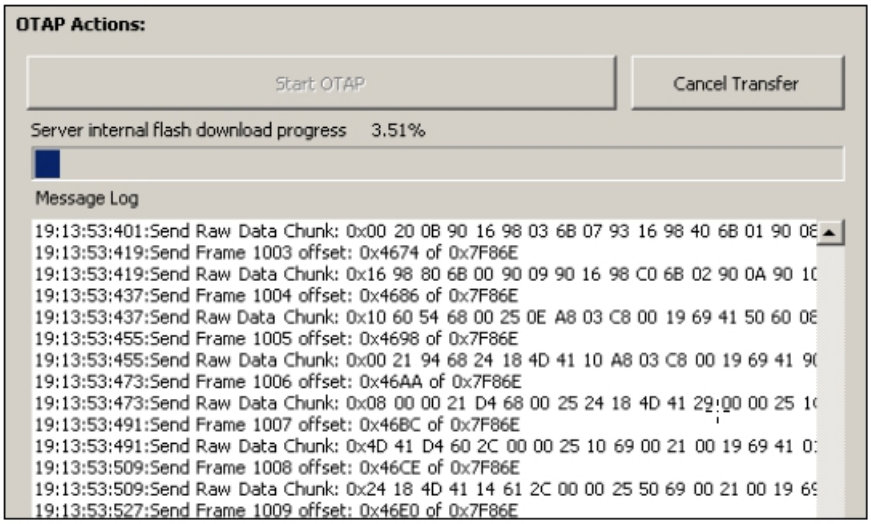

Figure 26. Test Tool OTAP BLE Image Transfer in Progress

7. After all the blocks are sent the OTAP Client send an Image Transfer Complete command to the OTAP Server. When this command is received by the PC Application, it displays a “Sent Image with Success” message in the log window.

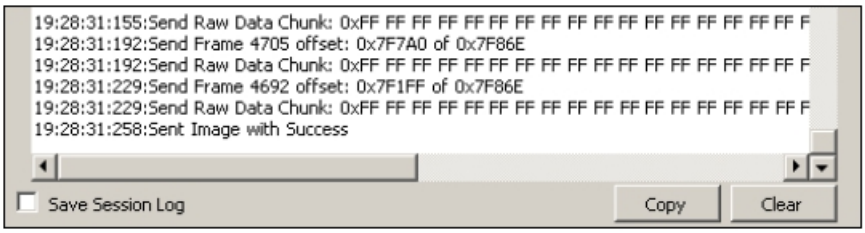

Figure 27. Test Tool OTAP BLE Image Transfer Completed

8. After the image transfer is complete the OTAP Client triggers the bootloader and resets the MCU. The bootloader takes about 30 seconds to flash a 512 KB image on a FRDM-KW41 platform. After this time passes the MCU resets again and runs the new image.

5.17.4 Usage with Kinetis BLE toolbox

This is the list of requirements.

- Mobile device running Android platform of iOS with hardware and software supporting Bluetooth 4.0 and later
- Kinetis BLE Toolbox application – download from the specific application store for your device

To run the application do the following.

1. Flash the OTAP Client ATT applications to a KW4x board. The Kinetis BLE Toolbox only supports the ATT OTAP Client
2. Create the application to send over the air in .srec format. Follow the instructions from the previous section on how to do this. Remember to include the Bootloader in its appropriate section in your application.
3. Start the Kinetis BLE Toolbox application on your mobile device and start the OTAP Tool. The application starts scanning.
4. Press SW4 on the KW4x board to start Advertising on the embedded OTAP Client application. The device should show up in the list of scanned devices. Touch the device in the scan list to connect to and the application performs service discovery and displays some information shown in the figures below.

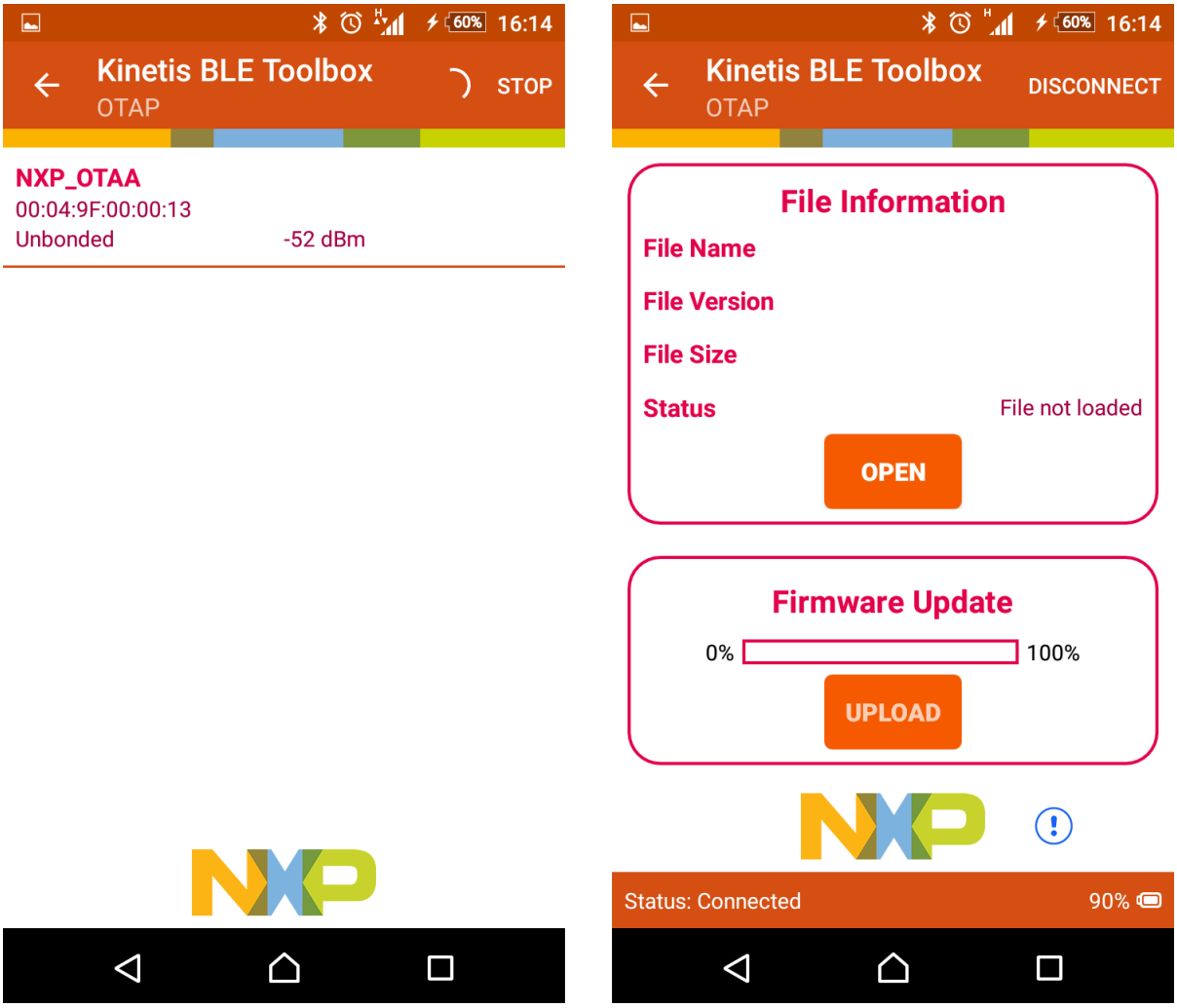

Figure 28. Kinetis BLE Toolbox - OTAP Tool Scanning and Discovery

5. Press the “Open” button and load the .srec file to be sent over-the-air. Once the file is loaded some information about it is displayed. Press the “Upload” button to start the image transfer process. A progress bar is shown while the image transfer is ongoing. The successful transfer is signaled by the progress bar reaching 100%. This is shown in the figures below.

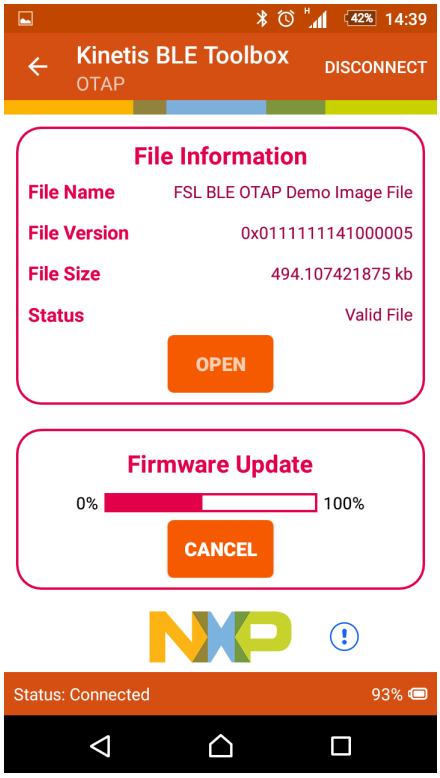

Figure 29. Kinetis BLE Toolbox - OTAP Image Transfer

6. After the image transfer is complete the OTAP Client triggers the bootloader and resets the MCU. The bootloader takes about 30 seconds to flash a 512 KB image on a FRDM-KW41 platform. After this time passes the MCU resets again and runs the new image.

5.18 Hybrid (Dual-Mode) Bluetooth Low Energy Heart Rate Sensor and IEEE 802.15.4 Coordinator Demo Application

This section describes the implemented profiles and services, user interactions, and testing methods for the Hybrid BLE Heart Rate Sensor/IEEE® 802.15.4 Coordinator demo application.

5.18.1 Implemented profile and services

The BLE Heart Rate Sensor implements a GATT server and the following profile and services.

- Heart Rate Profile v1.0
- Heart Rate Service v1.0
- Battery Service v1.0
- Device Information Service v1.1

The application behaves as a GAP peripheral node. It enters GAP General Discoverable Mode and waits for a GAP central node to connect. When the collector configures notifications, the sensor starts sending heart rate measurements every second.

The IEEE 802.15.4 part of the application is an IEEE 802.15.4 Coordinator running in background, in parallel with the BLE sensor application.

5.18.2 User interface

The IEEE 802.15.4 thread starts automatically at device startup in the role of an IEEE 802.15.4 Coordinator. It chooses the channel based on Clear Channel Assesment (CCA). No user interaction is needed.

The BLE Heart Rate Sensor user interface is identically with the standalone After flashing the board, the sensor is in idle mode (all LEDs flashing). To start advertising, press the SW4 button. When in GAP Discoverable Mode, the LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid. To disconnect the node, hold the SW4 button pressed for 2-3 seconds. The node then re-enters the GAP Discoverable Mode.

5.18.3 Testing method

The IEEE 802.15.4 Coordinator is basically a “MyWirelessApp Coordinator” demo (for more information check the 802154MPDAUG document), and responds to requests from other IEEE 802.15.4 devices (Active Scan, Association, Data Requests). Any IEEE 802.15.4 sniffer can be used to observe the over-the-air activity on the channel that the coordinator started.

At the same time, the BLE Heart Rate Sensor part of the application runs independently based on user interaction. The Heart Rate Sensor can be connected to any Bluetooth Smart Ready product available on the market. To make the Heart Rate Sensor visible, press the SW4 button to start sending advertisements, which causes LED3 to start flashing. The sensor name “FSL_HYBRID_HRS” shows on the device when its scanning is active. A solid LED1 indicates a successful connection between the 2 devices. Hold the SW4 button pressed for 2-3 seconds anytime to get the sensor to initiate disconnect.

If configured, the sensor notifies the application with heart rate measurements every second. Also, the battery level and various device information is exposed for reading. The Kinetis BLE Toolbox can be used to showcase the Heart Rate profile functionality.

5.19 Relay Proxy

This section describes the functionality, user interactions, and testing methods for the BLE Relay Proxy Application.

5.19.1 Implemented stack features

The BLE Relay Proxy implements a GAP dual-role application that allows 2 simultaneous connections, one with a GAP Peripheral and the other with a GAP Central device. It also supports both GATT client and server. When the two connections are active, the device will relay data from the GAP Peripheral to the GAP Central and vice-versa.

5.19.2 Implemented services

The device will discover services on the GAP Peripheral and clone the database locally by using the dynamic GATT database feature. The application acts afterwards as a proxy for that device and will respond to service discovery queries made by the remote GAP Central device. The application supports all services defined by the Bluetooth SIG. Connection with the GAP Peripheral that contains the database to be cloned is made based on the specified service in the following macro from app.h: gAppProxySelectService_d. By default, the application will check for GAP Peripherals that support the Heart Rate Service (0x180D).

5.19.3 User interface

After flashing the board, the sensor is in idle mode (all LEDs flashing). To start scanning, press the SW4 button. When in GAP Limited Discovery Procedure, LED3 is flashing. When the central node connects to the peripheral, LED3 turns solid.

After service discovery is done and the dynamic GATT database is created, the application enters GAP Discoverable Mode and LED3 starts flashing. When the central node connects to the peripheral, LED3 turns back solid. To disconnect the node from all connections, hold the SW4 button pressed for 2-3 seconds.

5.19.4 Usage

The application is built to work with any other two devices that implement a GAP Peripheral and a GAP Central. The out of the box example describes the BLE Relay Proxy connecting with a Heart Rate Sensor application from the same SDK and a smartphone or a tablet running the Kinetis BLE Toolbox application.

1. Download a Heart Rate Sensor application on a FRDM-KW41Z and start it as described in the previous chapter.
2. Press SW4 on the BLE Relay Proxy application. The node will connect with the Heart Rate Sensor and do Service Discovery. LED3 is solid. After the database is recreated locally, the node will start advertising with the same data as the original device. LED3 starts flashing.
3. Start Kinetis BLE Toolbox and connect to the BLE Relay Proxy application as with a normal Heart Rate Sensor application.
4. All GATT Read and Write operations made by the Kinetis BLE Toolbox are relayed by the node to the initial Heart Rate Sensor. All GATT Notifications are relayed to the Kinetis BLE Toolbox application.

5.20 Wireless Power Transfer System Receiving Unit and Transmitting Unit

This section describes the implemented profiles and services, user interactions, and testing methods for the wireless power transfer receiving unit (WPT PRU) and transmitting unit (WPT PTU).

5.20.1 Implemented profile and services

The WPT PRU and PTU applications implement a GATT server and a GATT client for the following profile and service:

- A4WP Wireless Power Transfer System specification v1.3
- Wireless Power Transfer (WPT) Service v1.3

The WPT PRU application behaves as a GAP peripheral node. It enters GAP Limited Discoverable Mode using as advertising data the WPT service UUID (0xFFFE) and the ATT database handle corresponding to the service declaration.

The WPT PTU application behaves as a GAP central node. It scans for PRU devices by looking for the WPT Service UUID, it connects to a PRU device, performs a procedure known as device registration and allows or disallows the device to start charging. The device registration procedure should take less than 500 milliseconds and for this reason the profile uses security mode 1, level 1 and no service discovery is performed. All handles are computed using the offset handle found in the advertising data. Device charging state and other measurable values are simulated in software using random numbers (except for device temperature included in the dynamic parameters).

Both application use pairing with bonding by default. Both applications require a terminal to output application specific information. A part of the application information is provided using onboard LED's. All actions are triggered using the push-button events. The events have the following meaning:

- Short press: pressing a button and releasing it under 1 second.
- Long press: pressing a button and releasing after 1 second, but before 8 seconds are passed.
- Very long press: pressing a button and releasing it after 8 seconds.

5.20.2 User interface

After flashing the boards, the PRU device enters a “Null” state, and the PTU enters a “Power Save” state. A short press on the SW4 button of the PTU device simulates a device configuration and the presence of a PRU device which requires charging. For this reason, the PTU device starts a scanning procedure. A short press on the SW4 of the PRU device simulates sufficient power to start the advertising procedure. During both scanning and advertising procedures, LED3 will flash. If the PRU device does not connect to the PTU for 10 seconds, the advertising is stopped and LED3 turns off. In this case the PRU device returns to the “Null” state. After the PTU connects to the PRU, LED3 is turned on.

After the devices connect, the PTU performs device registration and determines if it has sufficient power to charge the PRU. If registration is successful, the push-buttons events change their meaning as follows:

- Short press on the PTU device triggers another scanning procedure in case another device is supported.
- Short press on the PRU device prints the dynamic parameters in the console.
- Long press on the PTU device prints information regarding the connected devices and maximum power remaining.
- Long press on the PRU device simulates a charge complete event. After the PRU is considered charged, the PTU will disconnect from it. After disconnection, the PRU device enters in a “Boot” state and requires a very long press event to return to “Null” state.
- Very long press on the PTU device simulates an error case and disconnects all connected PRU devices.
- Very long press on the PRU device simulates power removal (taking a device outside the charge area).

If the PRU device is allowed to charge, it simulates charging by printing information in the console and by turning LED4 on.

5.20.3 Usage

The setup requires two or three FRDM-KW41Z platforms, one for PTU device and one or two for PRU devices.

1. Open serial port terminals for all the platforms. The start screen is displayed after the board is reset. At first the LEDs are flashing on all devices.
2. Press SW4 on the PTU device and on a PRU device. LED3 should start flashing on both devices until the connection is established or until the PRU device times out.
3. After the registration procedure is finished the PTU device will display a success message and will simulate the procedure for providing power to the PRU.

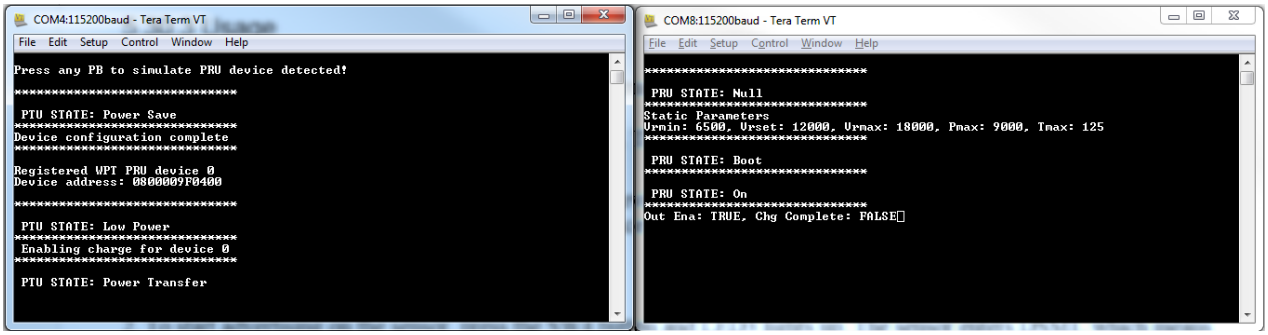

Figure 30. PTU and PRU after connecting, registering and enabling charge

4. Short press SW4 on the PRU device again to print most recent dynamic parameters.
5. Long press on SW4 on the PTU device to get system information.

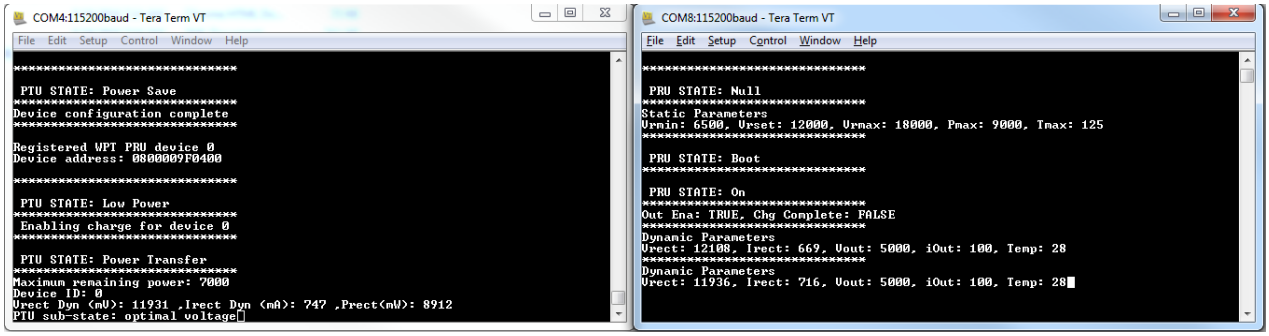

Figure 31. PTU information and PRU dynamic parameters display

6. Long press SW4 on the PRU device to simulate a charge complete event.

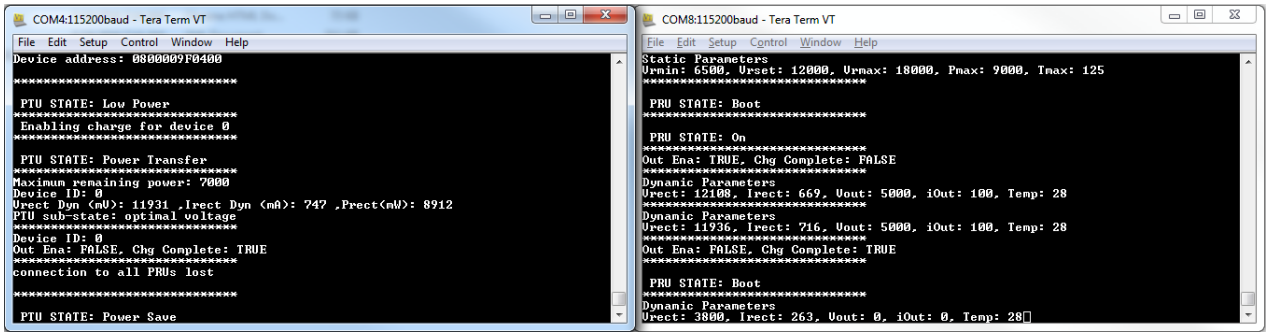

Figure 32. PRU device charged

7. Very long press SW4 on the PRU device to simulate power removed and return it into a “Null” state.

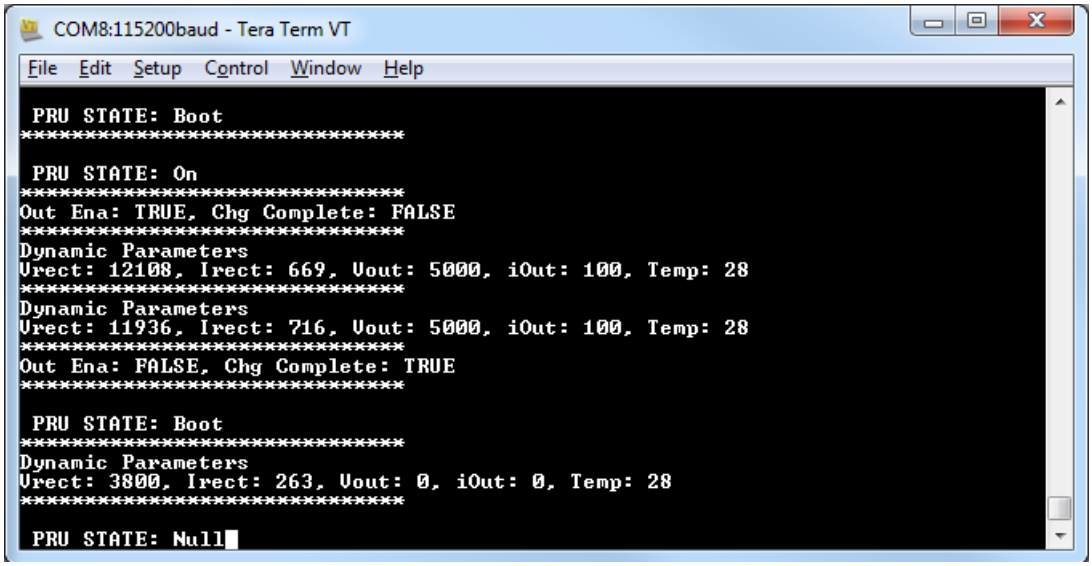

Figure 33. PRU power removed

8. If a second PRU device is used, SW4 can be short pressed on both the PTU and on the new PRU after step 3 was finished for the previous PRU. Also, to test the power sharing procedure support, the gAdjustPowerCommandSupport_d macro should be defined as TRUE on at least one of the PRU devices before flashing the firmware. If the above macro is set to TRUE a scenario is designed so that two PRU devices combined require more power than the PTU device can provide, but the PTU device attempts a power sharing procedure so that it can adjust the power draw for at least one of the PRU’s.

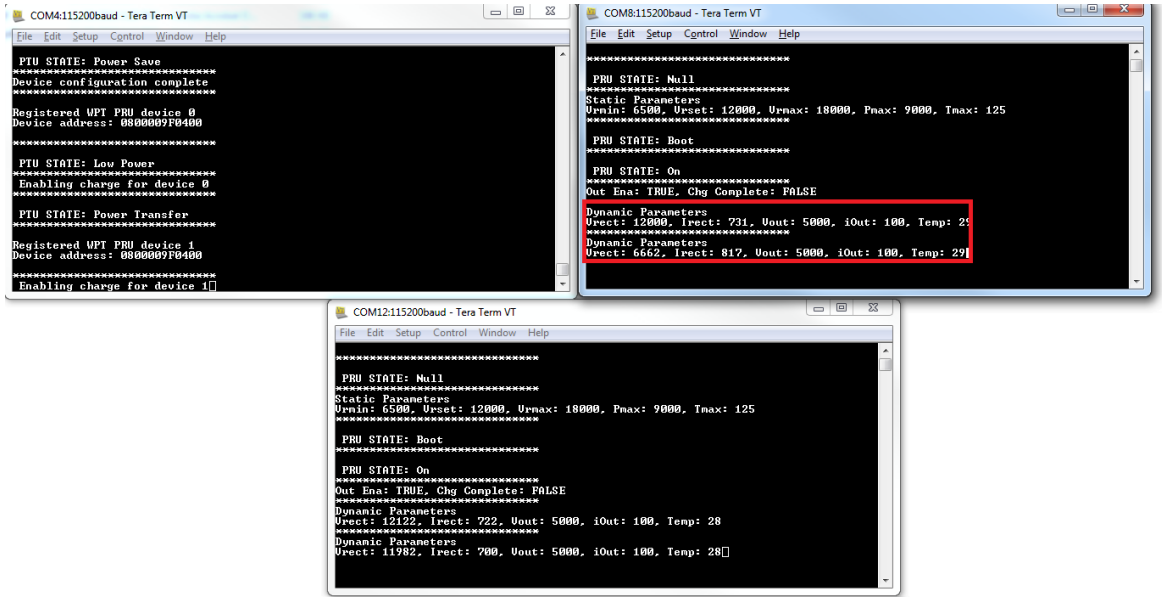

Figure 34. Power sharing support on one PRU

6 Revision History

This table summarizes revisions to this document.

Table 1. Revision history

Revision number	Date	Substantive changes
0	06/2015	Initial release
1	10/2015	Added new applications
2	04/2016	Added BLE Shell App Added KW41Z UI section Updated IPSP section Added interaction with Kinetis BLE Toolbox Application Changed OTAP demo application: removed instructions for use with the OTAP Python script for PC, added instructions for use with the Test Tool for Connectivity products BLE OTAP tool, and added instructions for use with the Kinetis BLE Toolbox mobile application
3	07/2016	Added BLE Relay Proxy application
4	09/2016	Updated for KW41Z

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support