

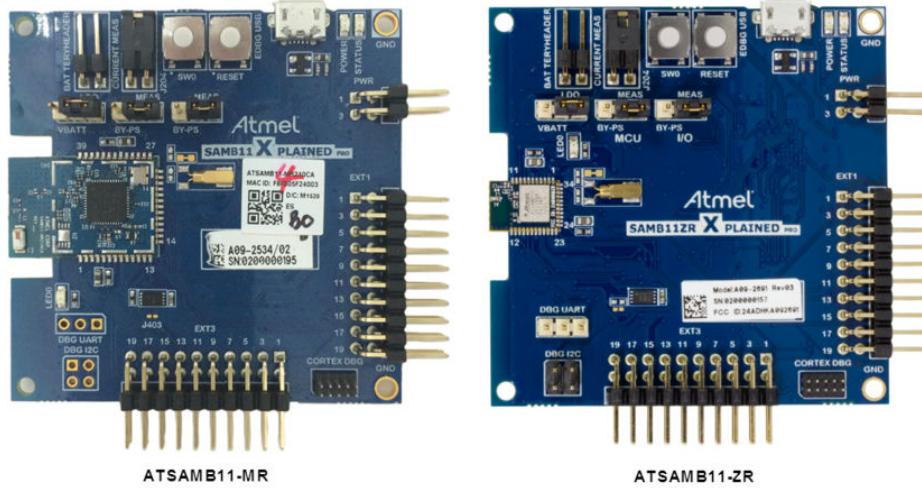
ATSAMB11 BluSDK SMART Example Profiles Application User's Guide

Introduction

This document describes how to set the ATSAMB11-MR/ZR evaluation boards for various example applications supported by the Advanced Software Framework (ASF). This document also provides the list of supported IDEs to be used in conjunction with the ATSAMB11-MR/ZR evaluation boards (see [Table 2-1](#)). The part number of the ATSAMB11-MR Xplained Pro (XPro) board is ATSAMB11MR-XPRO and the part number of the ATSAMB11-ZR Xplained Pro board is ATSAMB11ZR-XPRO.

Note: All example applications are included in the BluSDK SMART software package.

Figure 1. ATSAMB11 Xplained Pro Boards



Features

- Observer Application
- Proximity Monitor Application:
 - Device discovery and disconnection
 - Services and characteristics discovery
 - Services – Link Loss service, Immediate Alert service, and Tx Power service
 - Setting up Path Loss and Link Loss
 - Received Signal Strength Indicator (RSSI) sampling
- Proximity Reporter Application:
 - Advertisement
 - Pairing/bonding
 - Services – Link Loss service (mandatory), Immediate Alert service, and Tx Power service

- Apple® Notification Center Service (ANCS) Application
- Scan Parameters Service Application
- Time Information Profile Application:
 - Device discovery and disconnection
 - Pairing/bonding
 - BLE time client
- HID Mouse Device and HID Keyboard Device Applications:
 - Advertisement
 - Pairing
 - Services: HID service and Device Information service
 - Report mode (mouse)/Report mode (keyboard)
- Battery Service Application
- Simple Broadcaster Application
- Device Information Service
- Custom Serial Chat (CSC) Profile Application:
 - Device discovery and disconnection
 - Pairing/bonding
 - Send and receive messages
- Heart Rate Profile Application:
 - Advertisement
 - Pairing/bonding
 - Heart rate sensor measurements
 - Console display
- Blood Pressure Profile Application:
 - Advertisement
 - Pairing/bonding
 - Blood pressure measurements
- Find Me Profile Application:
 - Advertisement
 - Pairing/bonding
 - Find Me alerts
- Phone Alert Status Profile Application:
 - Advertisement
 - Pairing/bonding
 - Phone alert status
- Alert Notification Profile Application:
 - Device discovery and disconnection
 - Pairing/bonding
 - Alert notification service
 - Alert on incoming call
- Multi-Role Peripheral Multi-Connect Application:
 - Supports eight connections
- L2CAP Throughput Application:
 - L2CAP central

-
- L2CAP peripheral
 - Used LE L2CAP connection oriented channel for data communication
 - Health Thermometer Profile (HTP) Application:
 - Advertisement
 - Pairing/bonding
 - RSSI sampling
 - Health thermometer service
 - Health thermometer profile app for iOS/Android™
 - iBeacon Application:
 - RSSI sampling
 - Beacon advertising
 - iBeacon demo app for iOS/Android
 - AltBeacon Application:
 - AltBeacon advertising
 - AltBeacon demo app for iOS/Android
 - Eddystone Beacon Application:
 - Eddystone UID, URL, and TLM frame types
 - URL configuration service with optional lock
 - Beacon demo application for Android and iOS
 - Direct Test Mode (DTM) Application:
 - DTM setup procedure
 - Downloading DTM firmware
 - AT Command Application
 - Reset the link layer
 - Set device configuration
 - Start scan
 - Stop scan

Table of Contents

Introduction.....	1
Features.....	1
1. Functional Overview.....	6
1.1. Observer Application.....	6
1.2. Proximity Reporter Application.....	6
1.3. Proximity Monitor Application.....	7
1.4. ANCS Profile Application.....	7
1.5. Scan Parameters Service Application.....	7
1.6. Time Information Profile Application.....	8
1.7. HID Mouse Device or HID Keyboard Device Application.....	8
1.8. Battery Service Application.....	8
1.9. Simple Broadcaster Application.....	8
1.10. Device Information Service Application.....	9
1.11. Custom Serial Chat Profile Application.....	9
1.12. Heart Rate Profile Application.....	9
1.13. Blood Pressure Profile Application.....	10
1.14. Find Me Profile Application.....	11
1.15. Phone Alert Status Profile Application.....	11
1.16. Alert Notification Profile Application.....	11
1.17. Multi-Role Peripheral Multi-Connect Application.....	12
1.18. L2CAP Throughput Application.....	12
1.19. Health Thermometer Profile Application.....	12
1.20. iBeacon Application.....	13
1.21. AltBeacon Application.....	13
1.22. Eddystone Beacon Application.....	13
1.23. Direct Test Mode Application.....	13
1.24. AT Command Application.....	13
2. Supported Hardware Platforms and IDEs.....	15
3. Hardware Setup.....	16
4. Software Setup.....	17
4.1. Installation Steps.....	17
4.2. Build Procedure.....	17
4.3. Application Configuration.....	24
5. Application Demo.....	26
5.1. Demo Setup.....	26
5.2. Console Logging.....	28
5.3. Running the Demo.....	28
6. Adding a BLE Standard Service.....	77

7. Custom Serial Chat Service Specification.....	80
7.1. Service Declaration.....	80
7.2. Service Characteristic.....	80
7.3. Endpoint.....	80
7.4. Characteristic Descriptors.....	81
7.5. Sequence Flow Diagram.....	81
8. Customization of Keil Project.....	82
9. Memory Map.....	93
9.1. RAM Partition.....	93
9.2. Sample GCC Linker File.....	94
9.3. Sample Keil Scatter File.....	96
10. BluSDK SMART Software Architecture.....	97
11. Document Revision History.....	98
The Microchip Web Site.....	99
Customer Change Notification Service.....	99
Customer Support.....	99
Microchip Devices Code Protection Feature.....	99
Legal Notice.....	100
Trademarks.....	100
Quality Management System Certified by DNV.....	101
Worldwide Sales and Service.....	102

1. Functional Overview

This chapter describes the functional overview of all the applications that are pre-defined in Atmel Studio.

1.1 Observer Application

The Observer application is used for continuously listening to the advertisement data over the air. This application supports the following advertisement data types:

- Incomplete list of 16-bit service class UUIDs
- Complete list of 16-bit service class UUIDs
- Incomplete list of 32-bit service class UUIDs
- Complete list of 32-bit service class UUIDs
- Incomplete list of 128-bit service class UUIDs
- Complete list of 128-bit service class UUIDs
- Shortened local name
- Complete local name
- Appearance
- Manufacturer specific data
- Tx Power
- Advertisement interval

1.2 Proximity Reporter Application

The Proximity profile is defined by the Bluetooth® SIG to enable proximity monitoring between two Bluetooth Low Energy (BLE) devices. The Proximity Monitor (a Generic Attribute (GATT) client) configures the behavior of the peer Proximity Reporter device (a GATT server) based on the link conditions. The configuration includes setting the alert level, which triggers on the Link Loss or based on a different threshold of the Path Loss. The Path Loss determines the quality of the connection and it is derived out of the Received Signal Strength Indicator (RSSI) and transmits the power. The Proximity Monitor continuously evaluates the Path Loss and creates an immediate alert in the Proximity Reporter device when the Path Loss crosses threshold values.

On-Board LED Status

The on-board LED is configured to notify the user about the alerts received. The different alerts for the Link Loss and Immediate Alert services are explained in the following subsections.

Link Loss

On the Link Loss, the LED blinks according to the alert level set by the Proximity Monitor. The alert levels are:

- NO_ALERT for No alert level
- MILD_ALERT for Mild alert level
- HIGH_ALERT for High alert level

Based on the alert level configuration set by the Proximity Monitor, the LED blinks at different rates:

- If the alert level is “HIGH_ALERT” then the LED blinks faster (1 second interval)
- If the alert level is “MILD_ALERT” then the LED blinks moderately (2 second interval)
- If the alert level is “NO_ALERT” the LED must be off

Alert on Path Loss (Immediate Alert)

This alert is applicable when the “Immediate Alert” service is implemented. The example application relies on the Path Loss configuration done by the Proximity Monitor and notifies accordingly. The alert levels are:

- NO_ALERT for No alert level
- MILD_ALERT for Mild alert level
- HIGH_ALERT for High alert level

Based on the alert level configuration set by the Proximity Monitor, the LED blinks at different rates:

- If the alert level is “HIGH_ALERT” then the LED blinks faster (3 second interval)
- If the alert level is “MILD_ALERT” then the LED blinks moderately (5 second interval)
- If the alert level is “NO_ALERT” the LED must be off

1.3 Proximity Monitor Application

The Proximity profile is defined by the Bluetooth SIG to enable proximity monitoring between two BLE devices. The Proximity Monitor (a GATT client) configures the behavior of a peer Proximity Reporter device (a GATT server) based on the link conditions. The Proximity Monitor configures the desired behavior of the peer device through setting alert levels on the Link Loss and the Path Loss. In addition, it also maintains the connection with the Proximity Reporter and monitors the link quality of the connection based on RSSI reporting from the peer device.

1.4 ANCS Profile Application

The Apple Notification Center Service (ANCS) is used to enable a device to access notifications from an iOS device that exposes ANCS.

The ANCS profile defines the following roles:

- Notification Provider (NP) is a device that provides the iOS notification
- Notification Consumer (NC) is a device that receives the iOS notifications and notification- related data from Notification Provider

Incoming Call Notification

The programmed ATSAMB11-MR/ZR (Notification Consumer) must be paired with an iPhone® to display the received incoming call notification on a console.

The Bluetooth SIG defined Alert Notification profile provides similar functionality for Android devices. ANCS is a variant of the Alert Notification profile customized by Apple. For more details on Alert Notification, refer to the [Alert Notification Profile Application](#).

1.5 Scan Parameters Service Application

The Scan Parameter service is an example application that demonstrates how to retrieve scan interval window information from a peer device. The Scan Parameter service must be implemented on a peer device to retrieve scan interval information. This application implements a GATT server role. This application can be used for obtaining the updated scan interval window value by configuring the scan refresh characteristic for notification.

1.6 Time Information Profile Application

The Time Information Profile is an example application for a compatible Android/iPhone device for implementing the BLE time service, such as current time, date and day, and displaying it on the console.

The profile defines the following roles:

- Time client, a device in a peripheral role to read the time, date, and day information
- Time server, a device to provide the time-related information

Note: This application is supported in iOS 7.0 and above or a BLE compatible Android device which has the Microchip SmartConnect mobile application installed.

1.7 HID Mouse Device or HID Keyboard Device Application

The HID Over GATT Profile (HOGP) is defined by the Bluetooth SIG to enable HID services support over a BLE protocol stack using the GATT profile. This allows devices like a keyboard or mouse to implement HOGP and to connect with a compatible HOGP/BLE host device, such as mobile phone, tablet, TV, and so on.

The HID Mouse device or HID Keyboard device application supports the following characteristics:

- Protocol mode (mouse/keyboard)
- Report (mouse/keyboard)
- Report map (mouse/keyboard)
- HID information (mouse/keyboard)
- HID control point (mouse/keyboard)
- Boot mouse input report (mouse only)
- Boot keyboard input report (keyboard only)
- Boot keyboard output report (keyboard only)

This example application simulates a function of a mouse or keyboard. Once the connection procedure is implemented between a mobile phone and the ATSAMB11-MR/ZR board, the board can act as a mouse or a keyboard.

In the case of a HID Mouse device application, a mouse cursor, visible in the mobile screen, can be moved as per the predefined pattern by pressing the SW0 button on the board.

In the case of a HID keyboard device application, the predetermined text is sent to the mobile phone by pressing the SW0 button on the board. This can be viewed in any standard text editor in the mobile phone.

1.8 Battery Service Application

The Battery Service application is used for reporting the battery level of the device using the battery characteristics. Any application discovering the database can access the battery service instance during discovery services. This example application simulates the device battery level from 0% to 100%, with the step of 1% every second.

1.9 Simple Broadcaster Application

The Simple Broadcaster application is used for continuously broadcasting the advertisement data over the air. This application supports the following advertisement data types:

- Incomplete list of 16-bit service class UUIDs
- Complete list of 16-bit service class UUIDs
- Incomplete list of 32-bit service class UUIDs
- Complete list of 32-bit service class UUIDs
- Incomplete list of 128-bit service class UUIDs
- Complete list of 128-bit service class UUIDs
- Shortened local name
- Complete local name
- Appearance
- Manufacturer specific data

1.10 Device Information Service Application

The Device Information Service (DIS) application is used for providing a setup for the user to define and use the BLE DIS service. Any application discovering the database can access the DIS service instance during discovery services. This application supports the following characteristics:

- Manufacturer name string
- Model number string
- Serial number string
- Hardware revision string
- Firmware revision string
- Software revision string
- System ID
- IEEE® 11073-20601 regulatory certification data list
- PnP ID

1.11 Custom Serial Chat Profile Application

The Custom Serial Chat application is used for sending and receiving data between the boards (see, [Table 2-1](#)) and the Microchip SmartConnect mobile application. This is a custom profile example application, implemented over GATT. The user can send the information to the mobile phone using the console terminal that is configured with the board and vice versa.

Note: For more information on Custom Serial Chat service, refer to [Custom Serial Chat Service Specification](#).

1.12 Heart Rate Profile Application

The Heart Rate Profile application is used for enabling the collector device (GATT client) to connect and interact with a Heart Rate sensor (GATT server) to be used in the fitness applications. The Heart Rate sensor sends the heart rate measurement in bpm (beats per minute), energy expended in kJ (kilojoules), and R-R intervals in seconds. In addition to the Heart Rate service, this profile also implements the Device Information Service, which provides information about the Heart Rate sensor device.

The Heart Rate profile provided by Bluetooth SIG defines three characteristics for the exchange of heart rate parameters between the sensor and monitor. The characteristics of the profile are used to transfer heart rate parameters like bpm, R-R interval measurements, and other parameters like body sensor

location and energy expended values. The optional “Heart Rate Control Point characteristic” is used by the Heart Rate monitor to reset the energy expended in the Heart Rate sensor.

The Heart Rate sensor, which is the GATT server, holds the characteristics and sends the measurement values to the Heart Rate monitor.

- The Heart Rate, R-R interval, and energy expended are sent using the Heart Rate measurement characteristics
- The Heart Rate measurements are sent to the monitor on a value change if the monitor has enabled the notifications
- The body sensor location is read by the monitor by its body sensor location characteristic. The energy expended sent in the heart rate measurement can be reset by the monitor by writing to the Heart Rate control point characteristic

Note: The example application simulates the sensor measurements and sends them to the Heart Rate collector.

1.13 Blood Pressure Profile Application

The Blood Pressure Profile (BLP) application is used for connecting and interacting with a device with a blood pressure sensor to be used in a consumer and professional health care applications. This application enables the device to obtain blood pressure measurement and other data from a non-invasive blood pressure sensor that exposes the Blood Pressure service. For example, a nurse or doctor could use a non-invasive blood pressure sensor on a patient that sends blood pressure measurements to a laptop or other handheld device.

Blood Pressure Measurements

The blood pressure measurement characteristic can be used to send blood pressure measurements containing the following information:

- Flags field (containing units of blood pressure and used to show the presence of optional fields)
- Blood pressure measurement compound value field, depending upon the contents of the Flags field
- Timestamp (time of the measurement)
- Pulse Rate
- User ID
- Measurement status fields

The intermediate cuff pressure characteristic may be notified frequently during the course of a measurement, so that a receiving device can effectively update the display on its user interface during the measurement process.

When the client characteristic configuration descriptor is configured for indications and a blood pressure measurement is available, this characteristic is indicated while in a connection. When the client characteristic configuration descriptor is configured for indications and a blood pressure measurement is available, this characteristic is indicated while in a connection.

- The blood pressure measurement characteristic is used to send blood pressure measurements
- The intermediate cuff pressure characteristic is used to send current cuff pressure values to a device to display, while the measurement is in progress
- The blood pressure feature characteristic is used to describe the supported features of the blood pressure sensor

The ATSAMB11-MR/ZR simulates a blood pressure sensor (GATT server role) and sends simulated values to the blood pressure monitor (Microchip SmartConnect mobile application).

1.14 Find Me Profile Application

The Find Me Profile (FMP) application is used to define the device to create an alert signal behavior when a button is pressed on one device to cause an alerting signal on a peer device.

Find Me Target

The FMP defines the behavior when a button is pressed on a device to cause an immediate alert on a peer device. This can be used to allow users to find devices that have been misplaced.

The Find Me Target application, which is the GATT server, holds the alert level characteristics and waits for the Find Me locators alert and performs the following alert level characteristic:

- When the Find Me locator device wishes to cause an alert on the Find Me Target device, it writes the specific alert level (High, Mild and No alert) in the alert level characteristic.

1.15 Phone Alert Status Profile Application

The Phone Alert Status (PAS) profile is used to obtain the phone alert status exposed by the phone alert status service on a mobile phone. The alert status and ringer setting information of a mobile phone can be received and modified by the phone alert status service. The device can also use this profile to configure the ringer status on the mobile phone.

Phone Alert Status Notifications

This profile defines two roles:

- Phone alert server – device that originates the alerts
- Phone alert client – device that receives the alerts and alerts the user

The phone alert client (a GATT client) configuration is implemented on the ATSAM11-MR/ZR (see [Table 2-1](#)). The example application utilizes the SW0 button on the supported hardware platform to demonstrate the notification use-cases. A BLE compatible Android device that contains the Microchip SmartConnect mobile application provides the phone alert server functionality in this example. On the application, once the service is discovered and the user can click on the PAS service to enable the notifications.

1. After connecting with the mobile phone, press the SW0 button once to set the PAS server to “Silent” mode.
2. With the second SW0 button press, the device is set to “Mute” mode.
3. With the third SW0 button press, the device is return back to the “Normal” mode.
4. With the fourth SW0 button press, a “Read Characteristic” request is issued, that reads the characteristics of “Alert Status”, “Ringer Settings”, and “Ringer Control Point”.

Note: The PAS profile application is not supported in iOS devices. This example works only with BLE compatible Android devices that contain the Microchip SmartConnect mobile application.

1.16 Alert Notification Profile Application

The Alert Notification Profile allows a device to obtain information from a mobile phone about incoming calls, missed calls, and SMS/MMS messages. The information includes the caller ID for an incoming call or the sender ID for an email/SMS/MMS, but not the message text. This profile also enables the client device to get information about the number of unread messages on the server device.

Note: This example application only works with BLE compatible Android devices that contain the Microchip SmartConnect mobile application.

The Microchip SmartConnect mobile application is used for implementing the Alert Notification service and can be used for demonstrating an example application. This example application supports missed call alert notification and SMS alert notification.

The device implements the GATT client, which reads (or notifies) about the characteristic values received from the GATT server (the mobile phone). The device must be paired with an Android phone. A missed call or SMS alert notifications can be enabled/disabled, once connection is established. The Microchip SmartConnect application notifies a missed call or SMS alert, which are then displayed on the terminal console on the device side.

The “SW0” user button on the supported platform is programmed in such a way that each successive button press either enables or disables the notifications.

1.17 Multi-Role Peripheral Multi-Connect Application

The Multi-Role Peripheral Multi-Connect application demonstrates the ATSAMB11-MR/ZR to have eight simultaneous active connections. The ATSAMB11-MR/ZR supports multiple roles, such as GAP peripheral device with battery service and GAP central device with Find Me locator profile at the same time. It also supports multiple connections, such as GAP peripheral device with battery service that can connect with seven GAP central devices simultaneously.

The Multi-Role Peripheral Multi-Connect application initially starts advertising using connectable advertisement packets as a GAP peripheral and if any device sends a connection request, application gets connected to the remote device and exchanges the data on the established link. If the connection request from the device is not sent within a minute, then the application initiates to scan the devices and initiates a connection to the peripheral device, which advertises using connectable advertisement packets. The ATSAMB11-MR/ZR is exchanging the data as a GAP central once the link is established. Again, the Multi-Role application is started to advertise using connectable advertisement packets as a GAP peripheral and gets connected to the remote device, which sends a connection request and exchanges the data on the new link established. The process continues until the Multi-Connection application reaches eight connections.

1.18 L2CAP Throughput Application

The L2CAP Throughput example application supports the L2CAP central feature and the L2CAP peripheral feature.

1.19 Health Thermometer Profile Application

The Health Thermometer Profile (HTP) enables the data collection device to obtain data from a thermometer sensor that exposes the health thermometer service. The profile defines the following roles:

- Thermometer – Device to measure temperature
- Collector – Device to receive temperature measurement and other data from a thermometer

The thermometer implements only one Health Thermometer service in addition to the Device Information Service to display the information about the thermometer device. The current HTP application implements the following characteristics:

- Temperature measurement
- Intermediate temperature
- Measurement interval

1.20 iBeacon Application

The iBeacon application is used to advertise iBeacon specific packets that include UUID, major and minor numbers. Any beacon scanner application can be used for finding the beacon device. The iOS Microchip SmartConnect app can be used to find the beacon devices in the vicinity.

This profile defines the following roles:

- Monitor – Device (iOS/Android) to search for beacon packets
- Reporter – Device that continuously advertises the beacon packet as a part of advertisement data

1.21 AltBeacon Application

The AltBeacon application advertises packets that include MFG ID, Beacon code, Beacon ID, Reference RSSI, and MFG reserved value. Any AltBeacon scanner application can be used to find the AltBeacon device based on the beacon code. The supplied iOS demo app can be used to find the AltBeacon devices in the vicinity. The profile defines the following roles:

- Monitor – Device (iOS/Android) to search for AltBeacon packets
- Reporter – Device that continuously advertises the AltBeacon packet as part of advertisement data

1.22 Eddystone Beacon Application

Eddystone is an open Bluetooth Smart beacon format from Google that works across Android and iOS devices. The Microchip SmartConnect BLE BluSDK Smart software solution provides full support for this beacon format on the ATSAMB11-MR/ZR devices.

The Eddystone beacon application supports UID, URL, and TLM frame types. The application can be configured as follows using the APP_TYPE define:

- Set APP_TYPE to EDDYSTONE_UID_APP to send UID and TLM beacon frames at regular beacon intervals
- Set APP_TYPE to EDDYSTONE_URL_APP to send URL and TLM frames. This also supports the URL configuration service that enables the beacon to be configured dynamically from a mobile application

The Eddystone application is completely configurable using the `conf_eddystone.h` file. The #defines present in the `conf_eddystone.h` file are supplied with default values, which can be changed by the user to meet the requirements. In addition to this compile time configuration, frame fields like the UID value, URL, transmit power at 0 meters, and so on can be changed using the APIs provided in the `eddystone.h` file.

1.23 Direct Test Mode Application

The Direct Test mode (DTM) application is used to establish performance of Tx and Rx tests between two ATSAMB11-MR/ZR modules. The BLE Performance Analyzer is a performance analysis tool that is part of the Wireless Composer tool in Atmel Studio. This tool will be used at both ends (one assuming the role of a transmitter and the other the role of a receiver) for execution of tests. The BLE Performance Analyzer communicates to ATSAMB11-MR/ZR by using the DTM application running on the MCU.

1.24 AT Command Application

The following table provides the list of AT commands used.

Table 1-1. AT Command Set

Command	Description	Syntax
Reset	Reset the link layer	AT+RESET
Configuration	Set device configuration	AT+CFG_DEF
Start Scan	Start scan operation	AT+SCAN
Stop Scan	Stop an ongoing scan operation	AT+STOP

2. Supported Hardware Platforms and IDEs

The following table provides the supported hardware platforms and IDEs for the ATSAMB11-MR/ZR.

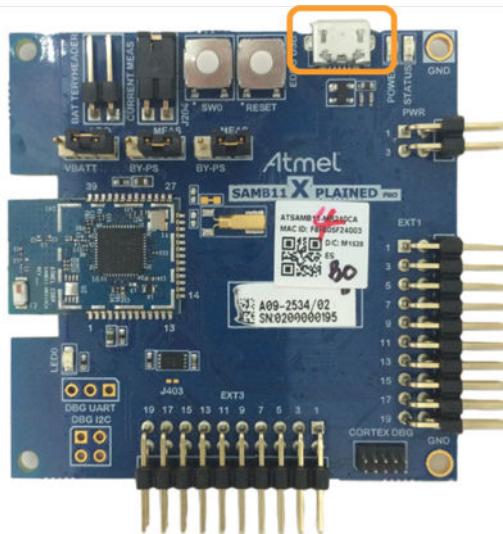
Table 2-1. BluSDK – Supported Hardware and IDEs

Platform	MCU	Supported IDEs
SAMB11MR-XPRO	ATSAMB11G18A	Atmel Studio v7.0 and Keil
SAMB11ZR-XPRO	ATSAMB11-ZR210CA	Atmel Studio v7.0 and Keil

3. Hardware Setup

The following figure illustrates the connection of the ATSAMB11-MR/ZR XPro board to the host PC using a micro-USB cable.

Figure 3-1. EDBG USB Port



4. Software Setup

4.1 Installation Steps

1. Download and install the [Atmel Studio](#) software.
2. Install the standalone [Advanced Software Framework \(ASF\)](#) package.
3. Keil IDE Installation – To use Keil IDE instead of Atmel Studio, perform the following:
 - 3.1. Download and install Keil MDK-ARM from <https://www.keil.com/download/product/>.
 - 3.2. Download and install Python from <https://www.python.org/downloads/>.

Note: When installing the Atmel Studio, the driver for SAMB11-MR/ZR XPRO is installed. Therefore, Atmel Studio must be installed to use the Keil compiler.

4. On the mobile phone, download and install the Microchip SmartConnect App on the mobile phone, available in the Apple Store for iPhone and in the Google Play™ Store for Android.

Note: Atmel Studio offers predefined example projects for the SAM B11 and SAM B11ZR XPro boards.

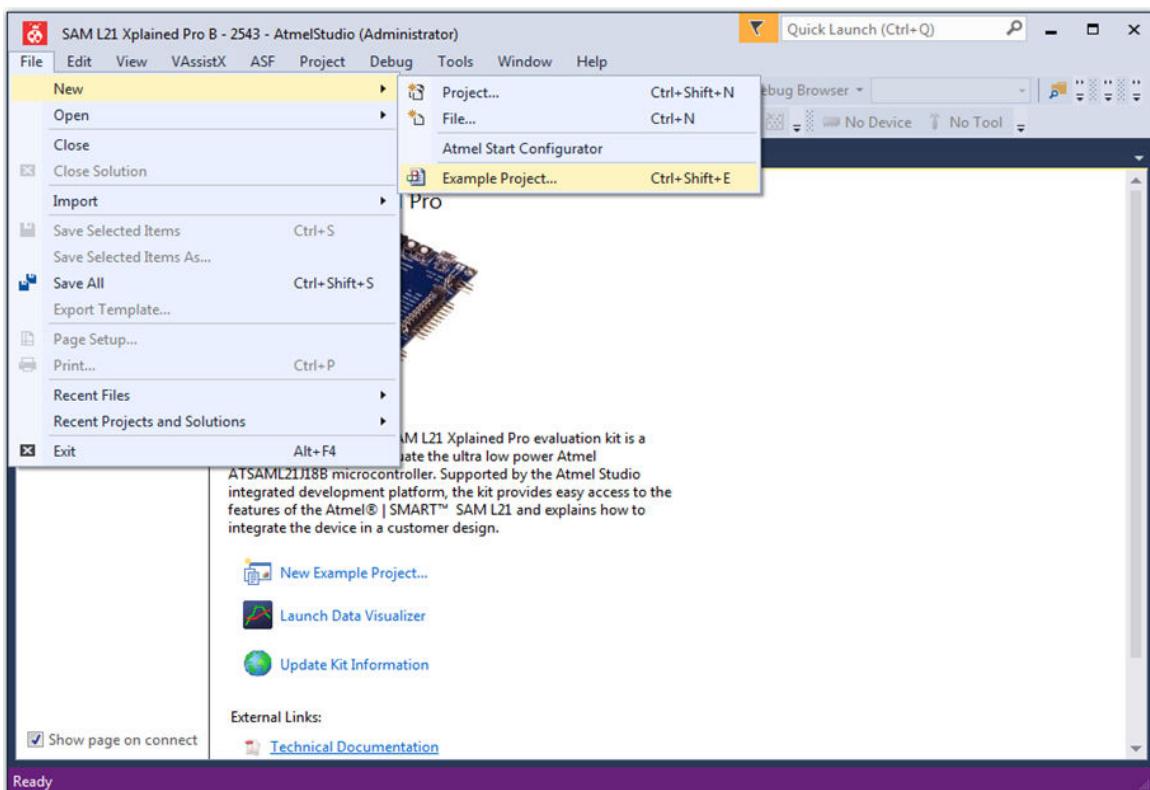
Note: For more information on the previous releases, refer to the *Atmel Studio Release Notes* available on the [Microchip web page](#).

4.2 Build Procedure

4.2.1 Build Procedure for Atmel Studio

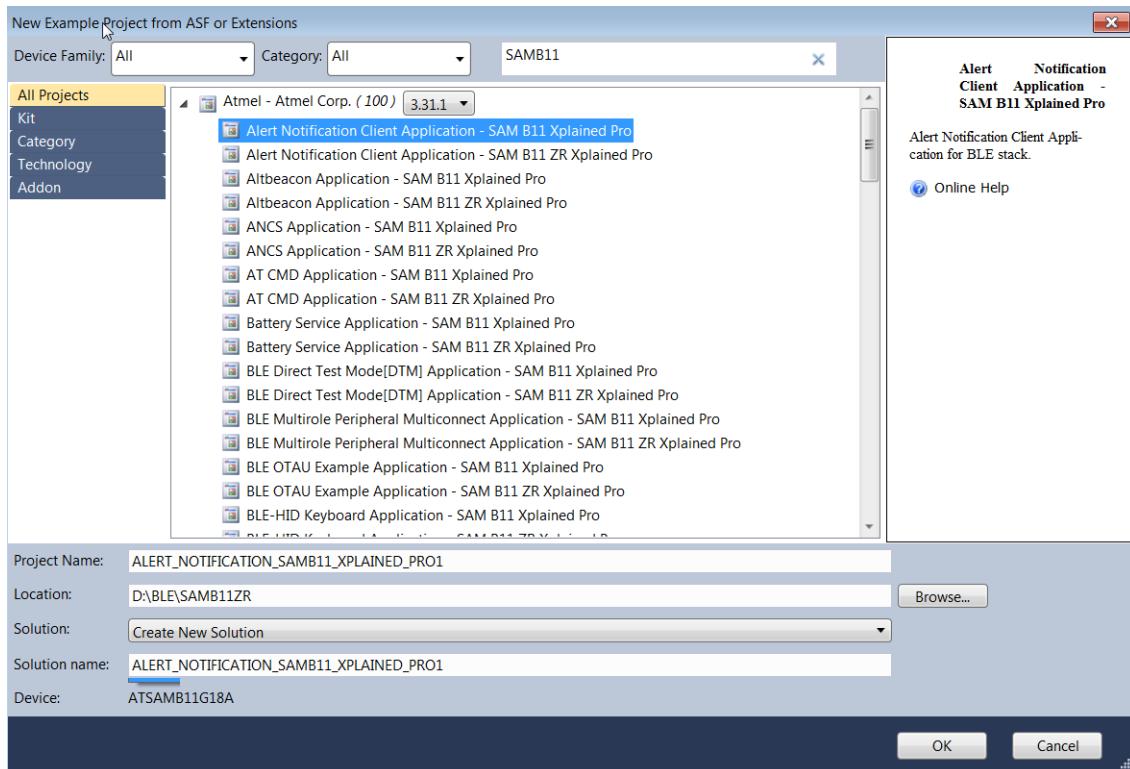
Perform the following steps to build an example project using Atmel Studio IDE. This example build procedure is developed using the SAMB11-MR/ZR Xplained Pro board, which is also valid for the other supported hardware platforms (see [Table 2-1](#)).

1. Open the Atmel Studio and select File > New > Example Project.

Figure 4-1. Creating a New Project

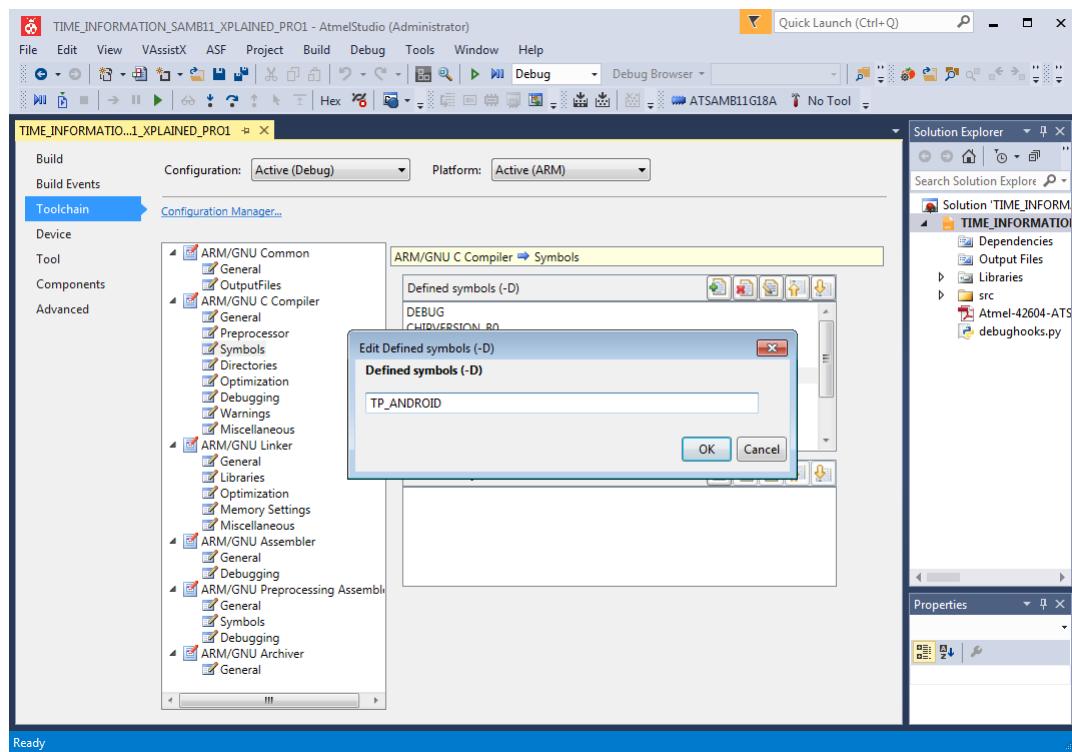
2. In the New Example Project from ASF or Extensions window:
 - 2.1. Enter “SAMB11” keyword in the search box, which lists all the supported examples for SAMB11-MR and SAMB11-ZR.
 - 2.2. Select the respective example application of the SAMB11-MR/ZR by expanding the “Atmel - Atmel Corp.” in the **All Projects** tab. This selection automatically populates the Project Name, Location, Solution, Solution Name, and Device.
 - 2.3. Click **OK**.

Figure 4-2. Searching for a Specific Application Example



3. Select the “Accept the License Agreement” check box and then click **Finish**.
 4. The Atmel Studio generates the project files for the selected application example that can be used in the SAMB11-MR/ZR Xplained Pro board.
 5. For the Time Information Profile application, the user must select the compiler symbol based on the following:
 - For Android devices: TP_ANDROID
 - For iOS devices: NTP_ANDROID
- Note:** This step is applicable only for the Time Information Profile application.

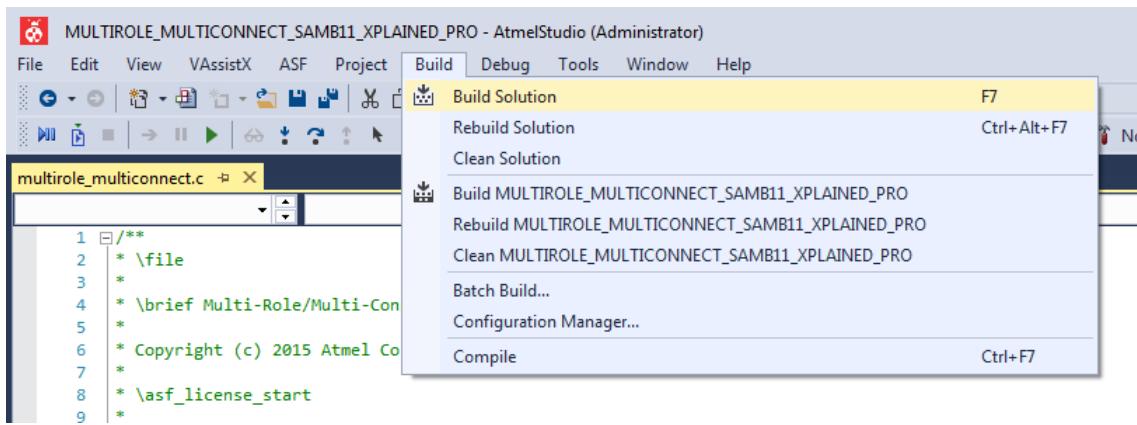
Figure 4-3. Configuring the UART Flow Compiler Symbols for the Time Information Profile



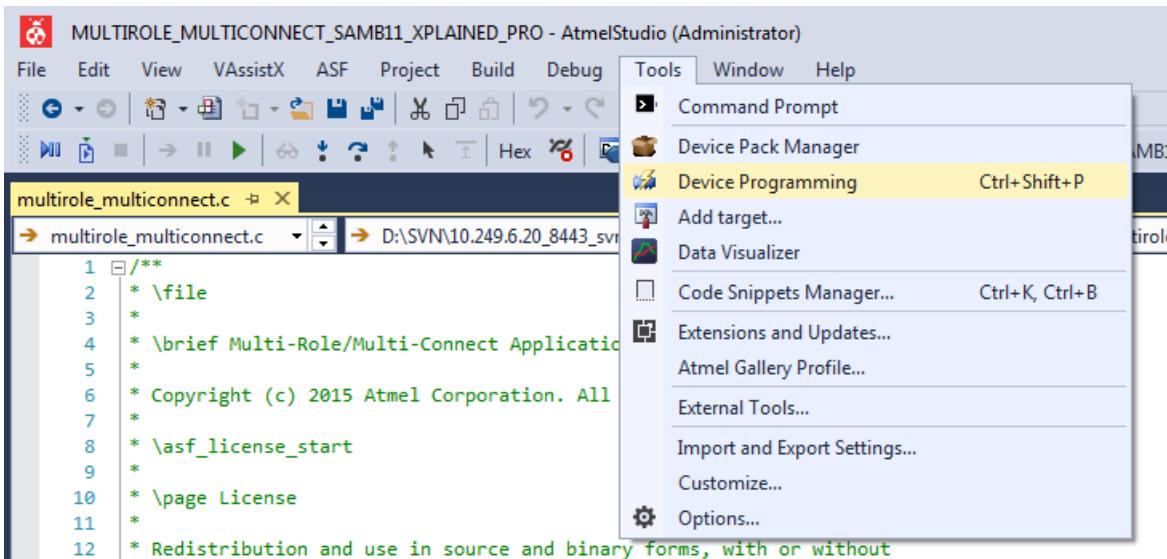
Note: iOS requires a device supporting the Time Information Profile to include the service solicitation advertisement type in the advertisement data. The above setting provides configuration to build the Time Information Profile for iOS or Android. The iOS natively supports Time Server and does not require a specific mobile application. To enable the devices that are displayed in the iOS BLE devices page, the service solicitation advertisement data type configuration is necessary.

- To build the solution, go to Build > Build Solution.

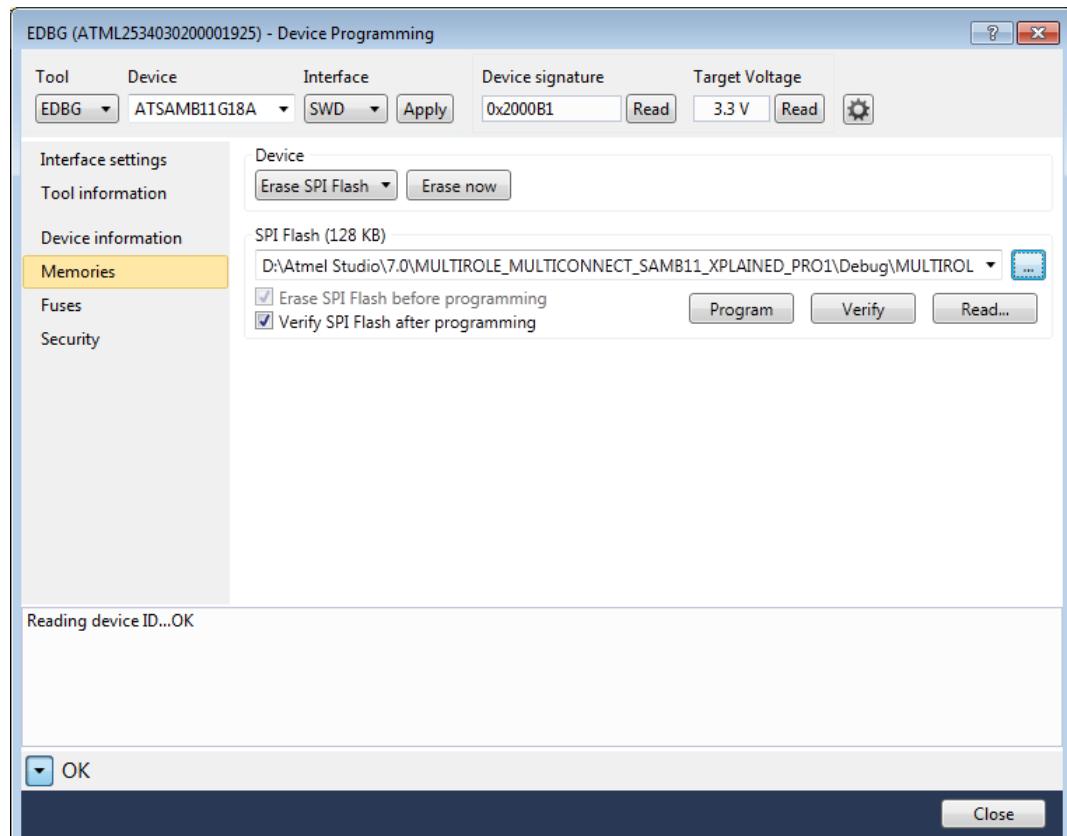
Figure 4-4. Building Solution for Selected Application Example



- The generated solution is downloaded into the SAMB11-MR/ZR XPro board through the USB cable. To program the board, go to Tools > Device Programming.

Figure 4-5. Selecting Device Programming

8. In the EDBG (XXXXXXXX) Device Programming window, perform the following steps:
 - 8.1. Select EDBG in **Tool**.
 - 8.2. Click **Apply** and then click **Read** to read the *Device Signature*.
 - 8.3. After reading the *Device*, click **Program** to program the device.

Figure 4-6. Embedded Debugger Device Programming Window

9. After flashing the example application into the SAMB11-MR/ZR Xpro board, it is ready to be used as a BLE device that supports the selected application example.

Note:

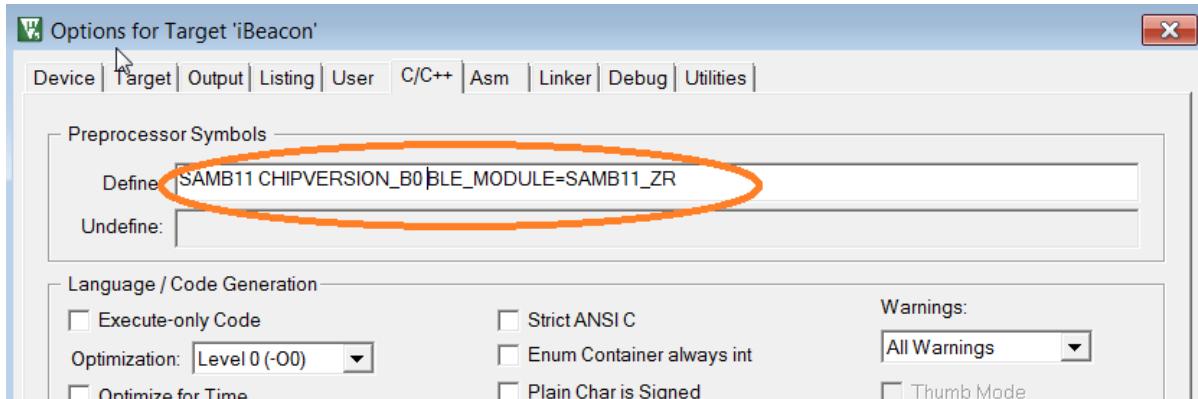
1. To run the profile application, refer to [Running the Demo](#).
2. In the case of Device Information, Simple Broadcaster and Observer application, refer to [Application Configuration](#).

4.2.2 Build Procedure for Keil

Perform the following steps to build an example project using Keil IDE. This example build procedure is developed using the SAMB11-MR/ZR Xplained Pro board. Applications for Keil IDE are available in the BluSDKSmart release package under \SDK. After unzipping the package, the example applications are available in the <release_dir>\apps folder.

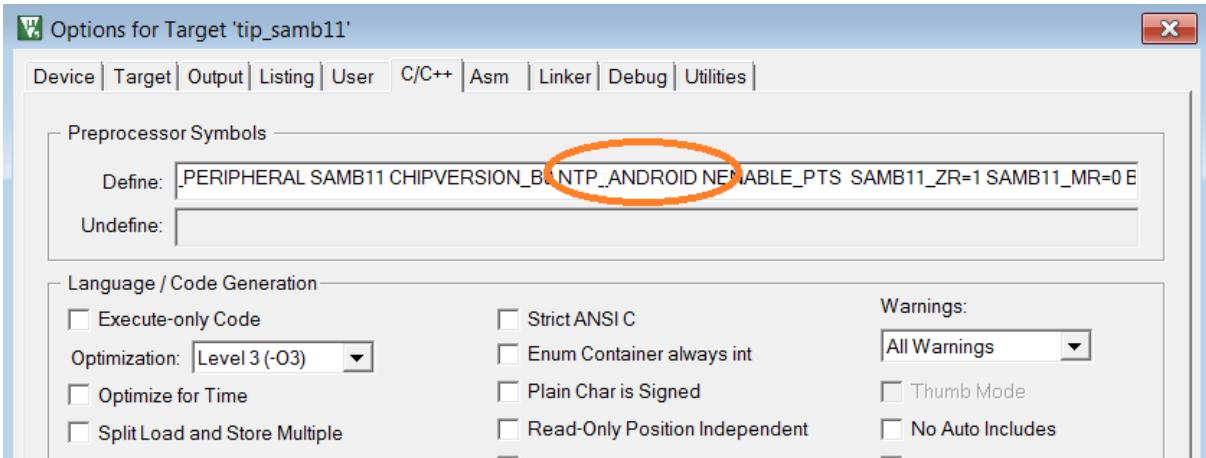
1. Open an intended sample application Keil project (*.uvprojx) from Keil IDE.
2. After opening the project, the following files are available in the Project tab:
 - xxxx_app.c (xxxx is replaced with the specific application name)
 - app_startup.s
 - ble_services
 - ble_profiles
 - services
 - libs (driver_lib.lib, ble_api.lib)
3. Set the appropriate build symbols (see the following figure):
 - For ATSAMB11-MR – BLE_MODULE=SAMB11_MR
 - For ATSAMB11-ZR: BLE_MODULE=SAMB11_ZR

Figure 4-7. Selecting SAMB11 Board Type



4. For the Time Information Profile application, the user must select the compiler symbol based on the following:
 - For Android devices: TP_ANDROID
 - For iOS devices: NTP_ANDROID

Note: This step is applicable only for the Time Information Profile application.

Figure 4-8. Compiler Symbol for Time Information Profile

Note: iOS requires the device supporting Time Profile to include service solicitation advertisement type in the advertisement data. The above setting provides configurability to build the Time Profile for iOS or for Android. iOS natively supports Time Server and does not require a specific mobile application. To enable devices to be displayed in the iOS BLE devices page, the service solicitation advertisement data type configuration is necessary.

5. Select Project > Rebuild all target files to compile the project.

Figure 4-9. Compiling the Project

6. Go to Flash > Download to download the application via the USB onto the SAMB11-MR/ZR XPro board.
7. After flashing the following messages are displayed in the build output section.

```

Wrote page 94 of 96 pages ....
Wrote page 95 of 96 pages ....
Wrote page 96 of 96 pages....
Writing Done...
Starting target application...O.K.
0x1001008f

```

8. Now the demo application is running on the SAMB11-MR/ZR XPro board.

Note:

1. To run the profile application, refer to [Running the Demo](#).
2. In the case of Device Information, Simple Broadcaster and Observer application, refer to [Application Configuration](#).
3. To create a custom project using Keil IDE, refer to [Customization of Keil Project](#).

4.3 Application Configuration

4.3.1 Configuration of Device Information Service Application

The list of macros that must be modified by user is as follows:

1. Before Connection:

The user must configure the default characteristics for the Device Information Service by modifying the following macros in `device_info.h`, as shown in the following figure.

Figure 4-10. Device Information Service Header File for Atmel Studio

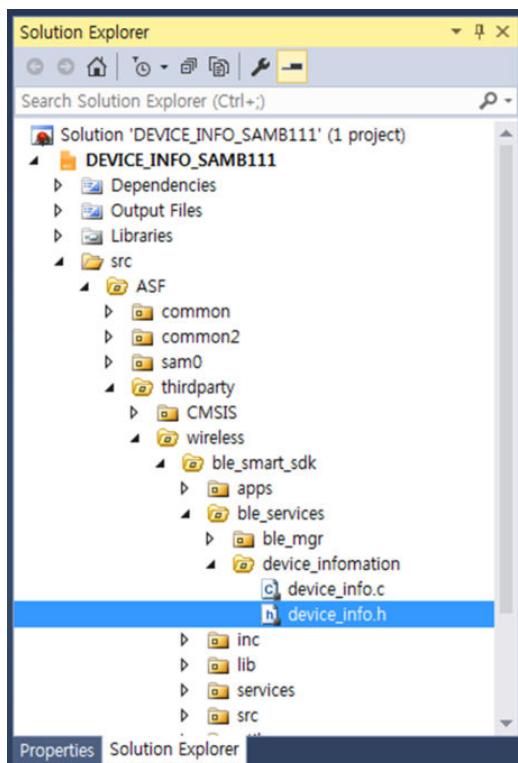
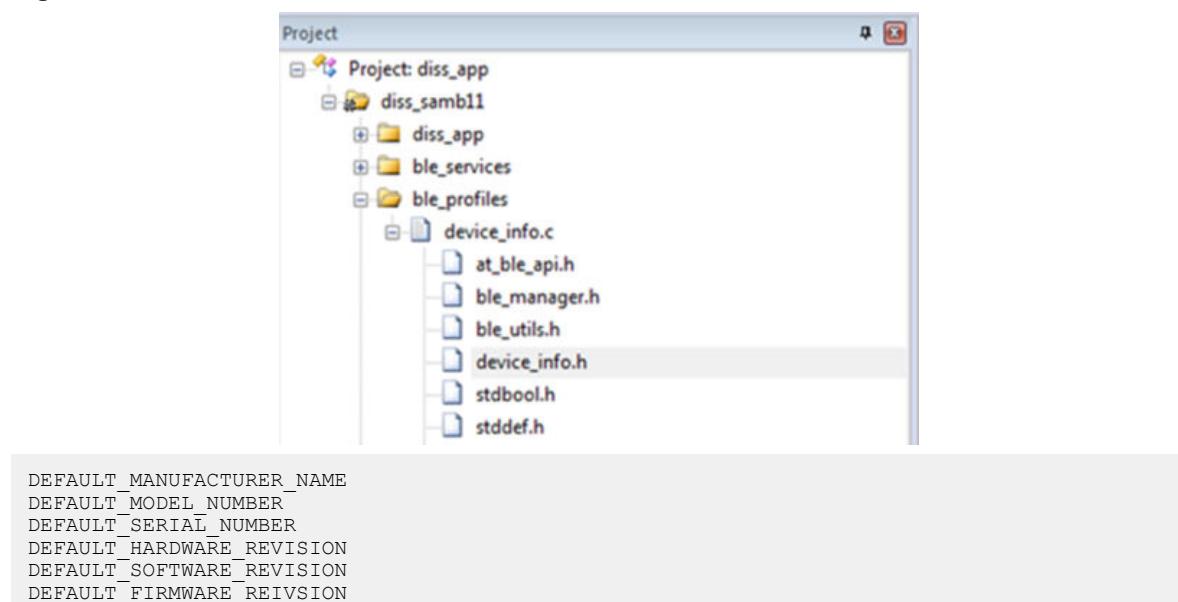


Figure 4-11. Device Information Service Header File for Keil



```

PNP_ID_VENDOR_ID_SOURCE
PNP_ID_VENDOR_ID
PNP_ID_PRODUCT_ID
PNP_ID_PRODUCT_VERSION
SYSTEM_ID_MANUFACTURER_ID
SYSTEM_ID_ORG_UNIQUE_ID

```

2. After connection:

The user can configure the values of characteristics for the Device Information Service using the following macros inside `device_info.h`.

```

UPDATE_MANUFACTURER_STRING
UPDATE_MODEL_NUMBER
UPDATE_SERIAL_NUMBER
UPDATE_HARDWARE_REVISION
UPDATE_FIRMWARE_REVISION
UPDATE_SOFTWARE_REVISION
UPDATE_SYSTEM_ID
UPDATE_PNP_ID
UPDATE_IEEE_REG_CERT_DATA_LIST

```

Note: After configuring the profile for Device Information Service, follow the steps mentioned in [Device Information Service Application](#).

4.3.2 Configuration of Simple Broadcaster Application

- Simple Broadcaster application advertises the default configuration provided as follows:
 - Non-connectable undirected advertisement event
 - Broadcasts data in advertisement data packets only
 - Broadcasts the following advertisement data types:
 - Complete list of 16-bit service class UUIDs
 - Complete local name
 - Appearance
- The configuration and advertisement data types listed above can be changed by using the macros provided in the `simple_broadcaster_app.h` file that is available in the `\src\config` directory.

Note: After configuration, follow the steps mentioned in [Simple Broadcaster Application](#).

4.3.3 Configuration of Observer Application

The default scanning parameters of Observer application are:

```

MAX_SCAN_DEVICE      (10)
SCAN_INTERVAL        (96)
SCAN_WINDOW          (96)
SCAN_TIMEOUT         (0x0000)
SCAN_TYPE            (AT_BLE_SCAN_ACTIVE)

```

These parameters can be modified as per the user requirement. They can be updated in the `ble_manager.h` file, which is available in the following directory: `\asf\thirdparty\wireless\ble_smart_sdk\ble_services\ble_mgr`

Note: After configuration, follow the steps in [Observer Application](#).

5. Application Demo

5.1 Demo Setup

The following figure shows how to setup the board and the Microchip SmartConnect App for demo purposes.

Figure 5-1. Demo Setup



Table 5-1. Demo Setup Details for Various Applications

Applications (Keywords)	BLE Node 1	BLE Node 2
Observer Application	Any BLE device can be used as Peripherals	Supported by the ATSAMB11-MR/ZR XPro board to act as an Observer application
Proximity Reporter Application	Supported by the Microchip SmartConnect application for iPhone/Android devices to act as a Proximity Monitor	Supported by the ATSAMB11-MR/ZR XPro board to act as a Proximity Reporter
Proximity Monitor Application	Supported by the ATSAMB11-MR/ZR XPro board to act as a Proximity Monitor	Supported by the ATSAMB11-MR/ZR XPro board to acts as a Proximity Reporter
ANCS Profile Application	Supported by the Microchip SmartConnect application for iPhone devices only, to act as a Notification Provider	Supported by the ATSAMB11-MR/ZR XPro board to act as a Notification Consumer
Scan Parameters Service Application	Supported by the Microchip SmartConnect application for iPhone/Android devices	Supported by the ATSAMB11-MR/ZR XPro board to act as Scan Parameter Service
Time Information Profile Application	Supported by the Microchip SmartConnect application for iPhone/Android devices to act as a Time server	Supported by the ATSAMB11-MR/ZR XPro board to act as Time client
HID Mouse Device Application	Supported by the Microchip SmartConnect application for Android devices only, to act as HOGP host	Supported by the ATSAMB11-MR/ZR XPro board to act as HID Mouse Device application

Applications (Keywords)	BLE Node 1	BLE Node 2
HID Keyboard Device Application	Supported by the Microchip SmartConnect application for Android devices only, to act as Notepad text editor app (HOGP Host role)	Supported by the ATSAMB11-MR/ZR XPro board to act as HID Keyboard Device application
Battery Service Application	Supported by the Microchip SmartConnect application for iPhone/Android devices	Supported by the ATSAMB11-MR/ZR XPro board to act as Battery Service application
Simple Broadcaster Application	Supported by the ATSAMB11-MR/ZR XPro board to act as Simple Broadcaster application	Supported by Scanner application on a mobile phone
Device Information Service	Supported by the Microchip SmartConnect application for iPhone/Android devices	Supported by the ATSAMB11-MR/ZR XPro board to act as Device Information Service application
Custom Serial Chat Profile Application	Supported by Custom Serial Chat (CSC) application for iPhone/Android to send and receive data	Supported by the ATSAMB11-MR/ZR XPro board with CSC application to send and receive data
Heart Rate Profile Application	Supported by the Microchip SmartConnect application for iPhone/Android devices to act as a Heart Rate Data Collector	Supported by the ATSAMB11-MR/ZR XPro board to act as Heart Rate Sensor
Blood Pressure Profile Application	Supported by the Microchip SmartConnect application for iPhone/Android devices to act as a Blood Pressure Monitor	Supported by the ATSAMB11-MR/ZR XPro board to act as Blood Pressure Sensor
Find Me Profile Application	Supported by the Microchip SmartConnect application for iPhone/Android devices to act as a Find Me Locator	Supported by the ATSAMB11-MR/ZR XPro board to act as Find Me Target
Phone Alert Status Profile Application	Supported by the Microchip SmartConnect application for Android devices only, to act as a Phone Alert Status server	Supported by the ATSAMB11-MR/ZR XPro board to act as Phone Alert Status client
Alert Notification Profile Application	Supported by the Microchip SmartConnect application for Android devices only, to act as a Notification Provider	Supported by the ATSAMB11-MR/ZR XPro board to act as Notification Consumer
Multi-Role Peripheral Multi-Connect Application	Supported by the ATSAMB11-MR/ZR XPro board and Microchip SmartConnect application to act as Peripheral or Central	

Applications (Keywords)	BLE Node 1	BLE Node 2
L2CAP Throughput Application	Supported by the ATSAMB11-MR/ZR XPro board to act as L2CAP Central	Supported by the ATSAMB11-MR/ZR XPro board to act as L2CAP Peripheral
Health Thermometer Profile (HTP) Application	Supported by the Microchip SmartConnect application for iPhone/Android devices to act as a Health Thermometer Collector	Supported by the ATSAMB11-MR/ZR XPro board to act as HTP application
iBeacon Application	Supported by the Microchip SmartConnect application for iPhone/Android devices to act as Beacon Monitor	Supported by the ATSAMB11-MR/ZR XPro board to act as Beacon Reporter
AltBeacon Application	Supported by the Microchip SmartConnect application for iPhone/Android devices act as a AltBeacon App (Monitor)	Supported by the ATSAMB11-MR/ZR XPro board to act as Reporter
Eddystone Beacon Application	Supported by the Microchip SmartConnect application for iPhone/Android devices	Supported by the ATSAMB11-MR/ZR XPro board to act as Eddystone Beacon Application
Direct Test Mode (DTM) Application	Supported by the ATSAMB11-MR/ZR XPro board to act as a Transmitter (Tx) Test Board. BLE performance analyzer tool connected with target board using COM port	Supported by the ATSAMB11-MR/ZR XPro board to act as a Receiver (Rx) Test Board. BLE performance analyzer tool connected with target board using COM port
AT Command Application	Supported by the ATSAMB11-MR/ZR XPro board connected with serial port monitor application (for example, TeraTerm) to act as AT Command Application.	Any BLE device can be used as Peripherals

5.2 Console Logging

For the purpose of debugging, a logging interface can be implemented in the applications.

The logging interface utilizes the same EDBG port that connects to a supported Xplained Pro (XPro) platform. A serial port monitor application (for example, TeraTerm) is opened and attached to the appropriate COM port enumerated by the device on the PC.

5.3 Running the Demo

Initializing the Device

Perform the following steps to initialize the device:

1. Open any Terminal Application (for example, TeraTerm). Select the COM port enumerated on the PC and set the following parameters:

- Baudrate 115200
 - Parity None
 - One Stop bit
 - One Start bit
 - No Hardware Handshake
2. Press the Reset button on the supported Xplained Pro (XPro) platforms (see [Table 2-1](#)).
 - Note:** The device is now ready to be used as selected application and starts to scan or advertise on the button press. This button must be pressed only when the “Press button” is displayed on the console log window. The same button is pressed to stop the device scan or advertise.
 3. The device is in advertising mode and the device initialization message is displayed on the console window.

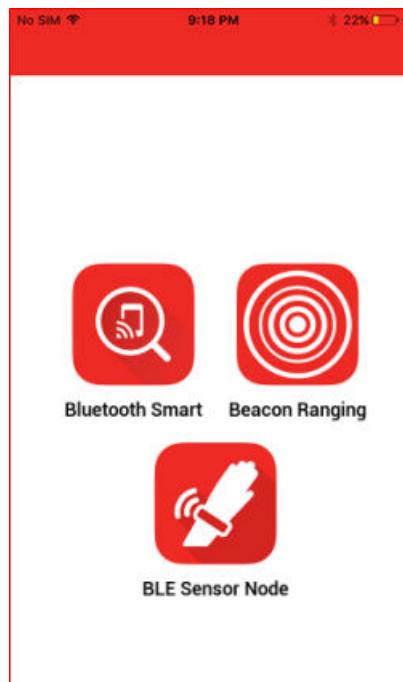
```
Initializing Application
Initializing SAMB11
BD Address:0xF8F005F23FFF, Address Type:0
BLE Started Advertisement
```

Pairing Procedure

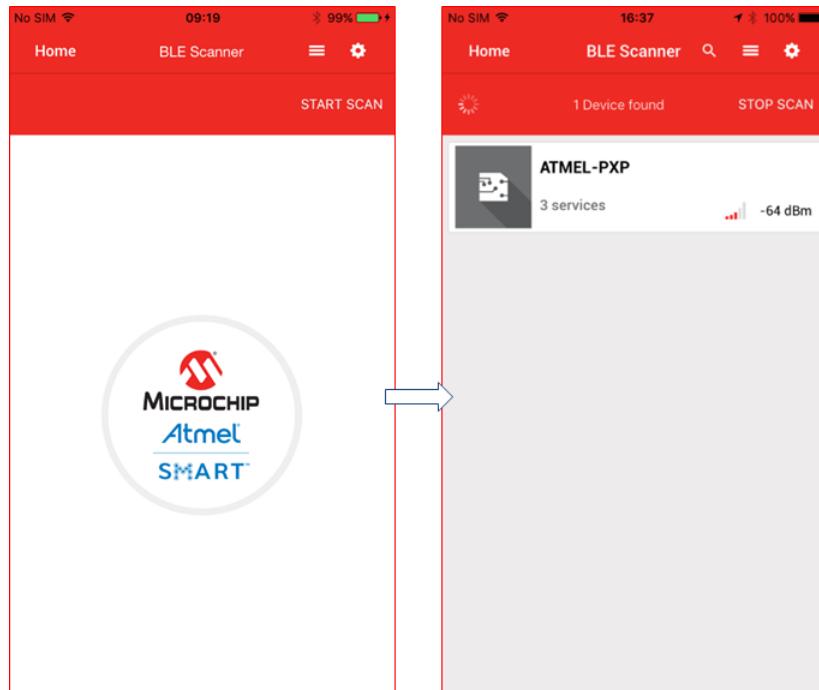
Perform the following steps to pair the device with the mobile phone application:

1. Open the Microchip SmartConnect application and click **Bluetooth Smart** in an application dashboard as illustrated in the following figure.
- Note:** With a Android mobile phone, ensure that the location service is enabled.

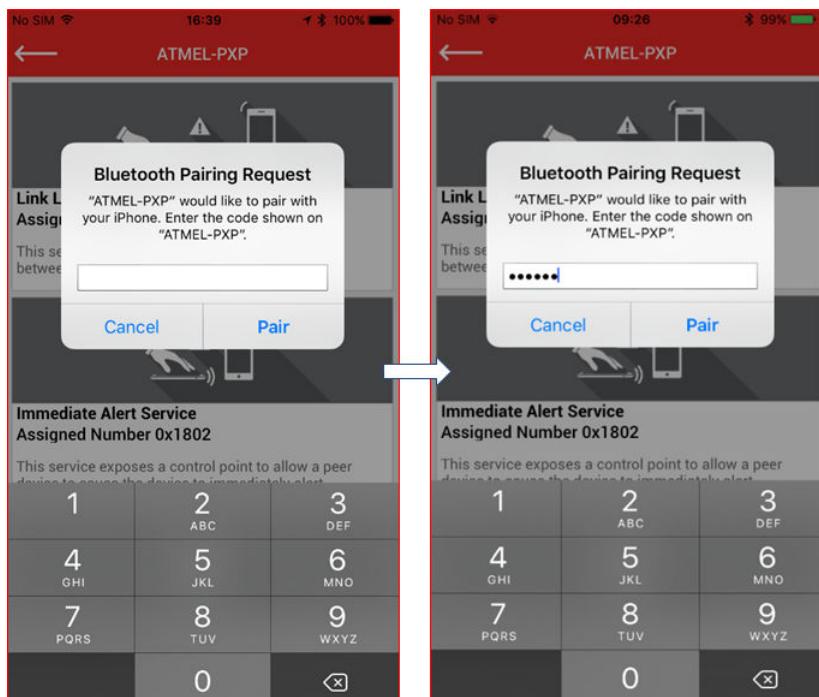
Figure 5-2. Dashboard of Microchip SmartConnect Application



2. To scan for the peripheral devices, click the **START SCAN** option available in scanning page. The device name (for example, ATTEL-PXP) is displayed among the list of scanned devices.

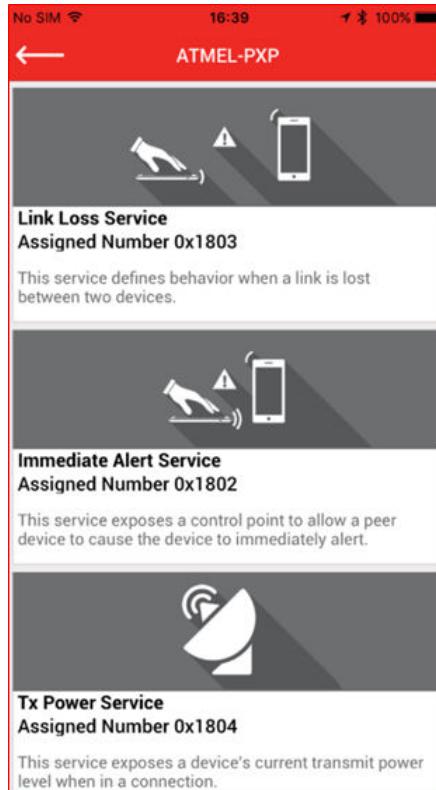
Figure 5-3. Scanning for Devices

3. Select the device name in the scan results, which initiates the pairing procedure. Enter the passkey “123456” on the Bluetooth Pairing Request window and click **Pair**. The mobile app displays “Successful connection” upon successful completion of pairing.

Figure 5-4. Pairing Request

4. On the device side, the console displays the successful completion of the pairing procedure.
5. On the Microchip SmartConnect app, the supported services are displayed for the device.

Figure 5-5. Display of Services Supported by the Application



5.3.1 Observer Application

Perform the following steps to run the Observer application demo:

1. Follow the steps (1 and 2) from [Initializing the Device](#).
Note: Observer Application Console Baudrate must be set to 921600.
2. Press the Reset button on the ATSAMB11-MR/ZR XPro board.
3. The device is now ready to be used as an Observer and starts to scan for nearby BLE devices.
4. The following figure shows example logs from the Observer application.

Figure 5-6. Observer Console Output

```

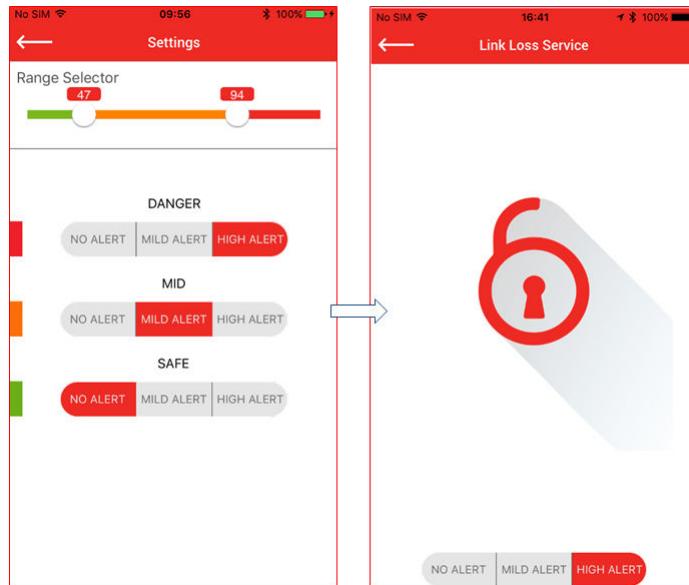
COM33:921600baud - Tera Term VT
File Edit Setup Control Window Help
Initializing SAMB11
SAMB11 Chip ID: 0x2000B1
BD Address: 0xP8F005F35DFA, Address Type:0
BluSmartSDK Firmware Version:6.1.6991
SAMB11 RF Version:0x10000000
Press button to start scanning..
Scanning...Please wait...
Scanning process initiated
Press button to stop scanning..
Scan Device Count:1
BD Addr :ed104dc09b77
RSSI :-45
Scan Device Count:2
BD Addr :c413440fb9c1
RSSI :-45
Scan Device Count:3
BD Addr :fb85e8bce818
RSSI :-42
Scan Device Count:4
BD Addr :f72709613df3
RSSI :-48
Scan Device Count:5
BD Addr :fd89c177d4eb
RSSI :-49
Scan Device Count:6
BD Addr :e785a3851ed3
RSSI :-55
Scan Device Count:7
BD Addr :fc471b2c6bd3
RSSI :-36
Scan Device Count:8
BD Addr :ffd4fa245023
RSSI :-42
Scan Device Count:9
BD Addr :ed104dc09b77
RSSI :-61
Scan Device Count:10
BD Addr :c413440fb9c1
RSSI :-59
Scan Device Count:11
BD Addr :e785a3851ed3
RSSI :-49
Scan Device Count:12
BD Addr :fd89c177d4eb
RSSI :-48
Scan Device Count:13
BD Addr :fc471b2c6bd3
RSSI :-35
Scan Device Count:14
BD Addr :ffd4fa245023
RSSI :-45
Scan Device Count:15
BD Addr :ed104dc09b77
RSSI :-47
Scan Device Count:16
BD Addr :c413440fb9c1

```

5.3.2 Proximity Reporter Application

Perform the following steps to run the Proximity Reporter application demo:

1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
2. Select the desired service (Link Loss or Immediate Alert) for alert level characteristics configuration. Choose a value from the following:
 - HIGH ALERT
 - MILD ALERT
 - NO ALERT

Figure 5-7. Configuring Alert Level Settings

- After configuration of desired alert levels, click **Immediate Alert** service and then move the mobile phone away from the Proximity reporter. Based on the separation distance, Path Loss is plotted on the zone radar (using RSSI values received from the Proximity Reporter). Based on the zone, the Proximity Monitor sends the corresponding alert level. The console log on the Proximity Reporter displays the corresponding alerts and on-board status LED behavior.

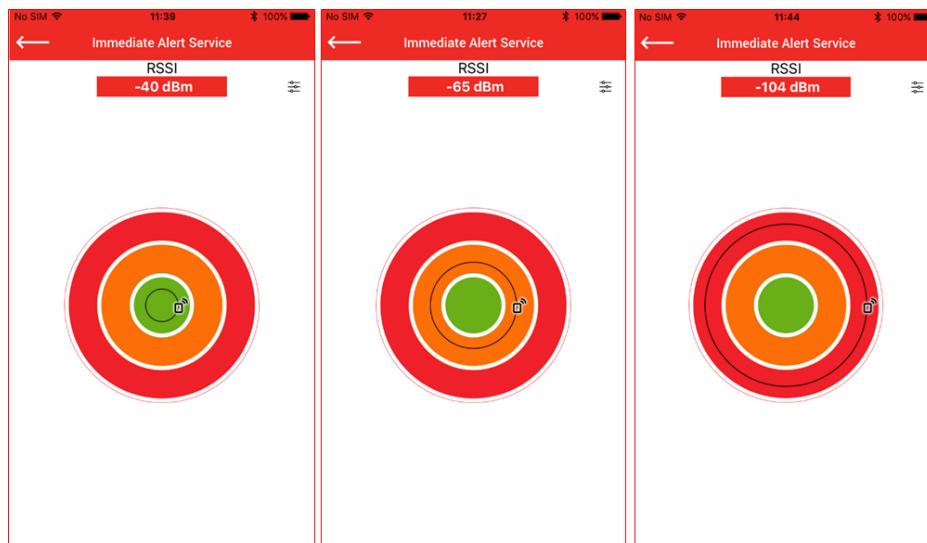
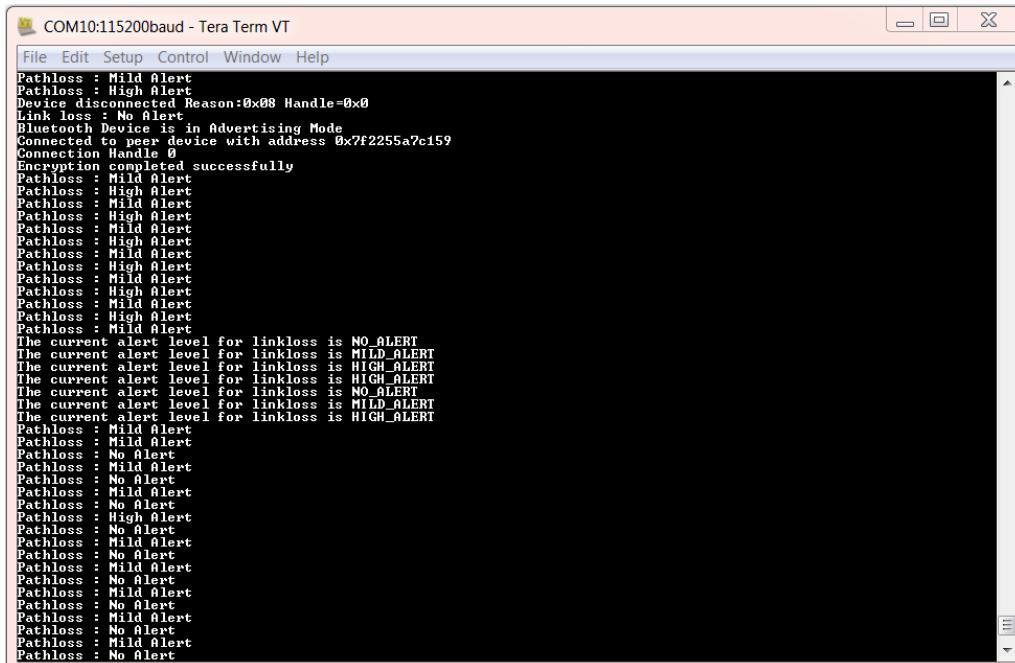
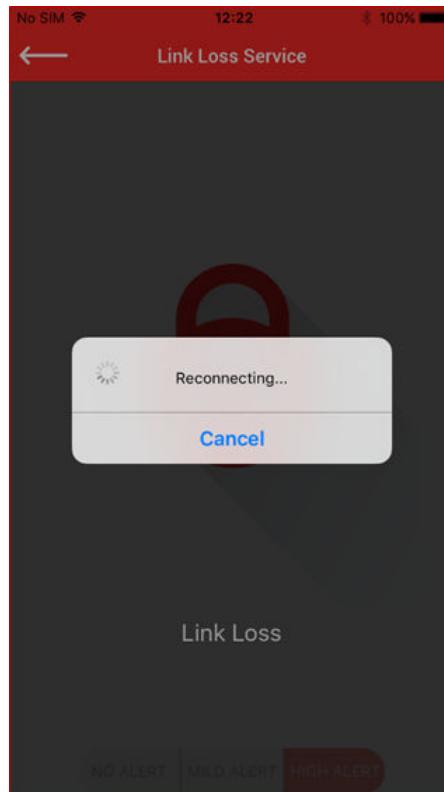
Figure 5-8. Proximity Reporter Path Loss Plot Across Safe, Mid, and Danger Zone

Figure 5-9. Proximity Reporter Path Loss Console Log Alerts Notification



- After configuration of desired alert levels, click on the **Link Loss** service and then move the mobile phone away from the reporter. Based on the separation distance, the Proximity Reporter receives the path loss notifications based on alert settings. Keep moving until the “Link Loss” pop-up appears. The console log on the Proximity Reporter displays the corresponding alerts and when Link Loss occurs, it reports disconnection and the on-board status LED behavior. The lock screen emulates a common use-case application where this Link Loss service is used (for example, Key Fob). When the user is in close proximity, the lock remains open. Subsequently, the user moving out of range can be triggered to close the lock.

Figure 5-10. Link Loss Pop-up on Proximity Monitor**Figure 5-11. Proximity Reporter Console Log for Link Loss**

```

Initializing Proximity Reporter Application
Initializing SAMB11
BD Address:0xF8F005F24004, Address Type:0
The Supported Services in Proximity Reporter are:
-> Link Loss Service
-> Immediate Alert Service
-> Tx Power Service
Bluetooth device is in Advertising Mode
Proximity Reporter Initializing Completed
Connected to peer device with address 0xf8f005f24002
Connection Handle 0
Peer device request pairing
Sending pairing response
Please Enter the following Pass-code(on other Device):123456
The current alert level for linkloss is 2
Pathloss : No Alert
Pathloss : No Alert
Pathloss : Mild Alert|
Device disconnected Reason:0x13 Handle=0x0
Link loss : High Alert
Bluetooth Device is in Advertising Mode

```

5. After Link Loss, the mobile application attempts to reconnect to the Proximity Reporter. The connection is re-established by moving the mobile phone closer to the reporter.

6. The Tx Power service is used to retrieve the Tx Power of the Proximity Reporter. Click the **Tx Power** service icon in the services screen. The Proximity Monitor reads the Tx Power value from the Proximity Reporter and displays the TX POWER LEVEL as shown in the following figure.

Figure 5-12. Proximity Monitor – Reading Tx Power Service



5.3.3 Proximity Monitor Application

Perform the following steps to run the Proximity Monitor application demo:

1. Connect one ATSAMB11-MR/ZR device loaded with Proximity Monitor example application code. Follow the steps (1 and 2) from [Initializing the Device](#).
2. Setup another ATSAMB11-MR/ZR device with Proximity Reporter application. Follow all the steps from [Initializing the Device](#). The device starts advertising.
3. Then the Proximity Monitor device starts scanning for available devices in the vicinity and displays their Bluetooth Device Address (BD) in the console window. The Proximity Reporter device found during the scan is displayed in the console log window (refer to the following figure). Select the index number of that device to establish connection with it.

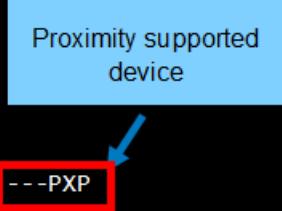
Figure 5-13. Proximity Monitor Connection with a Proximity Reporter

```

Initializing SAMB11
BD Address:0xF8F005F24002, Address Type:0
High Alert RSSI range: -91dBm and above
Mild Alert RSSI range: -70dBm to -90dBm
No Alert RSSI range: -69dBm and below
Initializing Proximity Monitor Application
Scanning...Please wait...
Info: Device found address [0] 0xF8F005F24004 ---PXP
Select Index number to Connect or [s] to scan
0
PXP Connect request sent
Connected to peer device with address 0xf8f005f24004
Connection Handle 0
Sending pairing response
link loss service discovered
Enter the Passkey(6-Digit) in Terminal:123456
Entered Pass-code:123456
Immediate Alert service discovered
Tx power service discovered
AT_BLE_CHARACTERISTIC_WRITE_CMD_CMP
AT_BLE_CHARACTERISTIC_WRITE_COMMAND : SUCCESS

Tx Power of device is 00 dBm
Starting timer
Rx Power(RSSI):-034 dBm---No Alert
Rx Power(RSSI):-036 dBm---No Alert
Rx Power(RSSI):-034 dBm---No Alert
Rx Power(RSSI):-033 dBm---No Alert

```



- When the connection is established, the Proximity Monitor sets the link loss alert value to “HIGH ALERT” at the Proximity Reporter device. The Proximity Monitor also monitors the path loss, if the Proximity Reporter device supports the optional “Immediate Alert” service and “Tx Power” service. The Proximity Reporter example application supports both of these optional services.

The default alert settings are as follows:

- For HIGH ALERT, set high alert RSSI to -91dBm and above. Alert status is indicated by LED, which must be ON.
- For MILD ALERT, set RSSI to -70dBm to -90dBm. Alert status is indicated by LED, which must be toggling.
- For NO ALERT, set RSSI to -69dBm and below. Alert status is indicated by LED, which must be OFF.

If the reporter device moves out of the proximity of the monitor device, the path loss crosses the threshold values and the corresponding alert value is set. The alert notification is displayed on the console as shown below.

Figure 5-14. Proximity Monitor Setting Alert Levels

```

Tx Power of device is 00 dBm
Starting timer
Rx Power(RSSI):-034 dBm--No Alert
Rx Power(RSSI):-036 dBm--No Alert
Rx Power(RSSI):-034 dBm--No Alert
Rx Power(RSSI):-070 dBm--Mild Alert!
Rx Power(RSSI):-070 dBm--Mild Alert!
Rx Power(RSSI):-070 dBm--Mild Alert!
Rx Power(RSSI):-056 dBm--No Alert
Rx Power(RSSI):-070 dBm--Mild Alert!

```

5.3.4 ANCS Application

Perform the following steps to run the ANCS application demo:

1. Follow the steps from [Initializing the Device](#).
2. From the Settings page of the iPhone, enable Bluetooth. The phone starts to scan for the devices. ATTEL-ANCS appears among the list of devices scanned. Click the **ATTEL-ANCS** to connect to the device.

Figure 5-15. ANCS Device Discovery in iPhone

3. When connected, the client side initiates the pairing request with the iPhone. The console log provides guidance for the user to enter the pass-key on the iPhone.

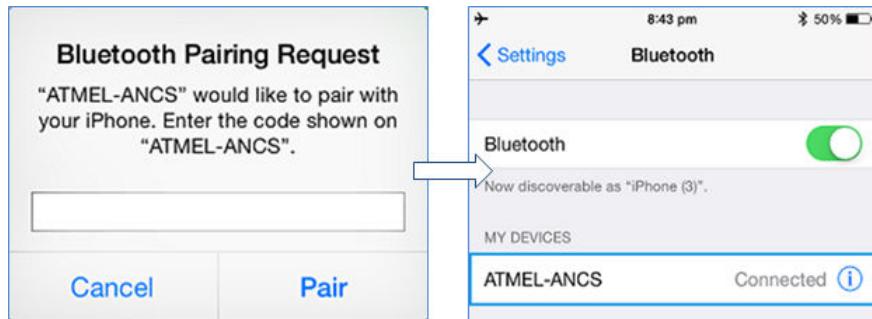
Figure 5-16. Console Display for Pairing in ANCS

```

ANCS Application
Initializing SAMB11
BD Address:0xF8F005F23FFF, Address Type:0
Device is in Advertising Mode
Peer device request pairing
Sending pairing response
Please Enter the following Pass-code(on other Device):123456
Pairing procedure completed successfully

```

4. In the Bluetooth Pairing Request window of the iPhone, enter the pass-key displayed in the console log and click **Pair**. After the device is connected, “ATTEL-ANCS” appears in the MY DEVICES section on the iPhone.

Figure 5-17. Pairing and Connecting iPhone to ATMEL-ANCS

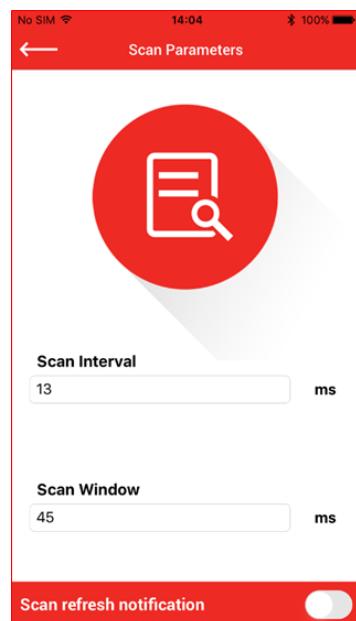
- Now, the user can initiate a mobile terminated call to the iPhone. When the iPhone receives a call, the corresponding incoming call alert is indicated on the device side console log window. Once the call is terminated, the device waits for a new alert to occur, as shown in the following screen.

Figure 5-18. Console Display for Notification Received as Incoming Call Alert

5.3.5 Scan Parameters Application

Perform the following steps to run the Scan Parameters application demo:

- Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
- When paired, the application displays the Scan Parameters and the Generic Information service.
- Click the **Scan Parameters** service. The user receives notification for scan refresh characteristic value. The user can disable the notification in the Scan Parameters page, as shown in the following figure.

Figure 5-19. Scan Refresh Characteristic Notification Options

- The user can set appropriate values for Scan Interval and Scan Window characteristics.
- The newly updated value of Scan Interval and Scan Window must be displayed on the console log of the device side as shown in the following figure.

Figure 5-20. Updated Scan Interval/Window Characteristic Value on Device

```

Initializing Scan Parameter Application
Initializing SAMB11
BD Address:0xF8F005F24004, Address Type:0
BLE Started Adv
Connected to peer device with address 0x6b2d2a1fe5e5
Connection Handle 0
Peer device request pairing
Sending pairing response
Please Enter the following Pass-code(on other Device):123456
AT_BLE_CHARACTERISTIC_CONFIGURATION_CHANGED
Pairing procedure completed successfully
LED On~~~
LED Toggle~~~
Scan Refresh Characteristic Value: 0
New scan interval window parameter
Scan Interval 3 ms
Scan Window 3 ms
LED Toggle~~~
Scan Refresh Characteristic Value: 0
New scan interval window parameter
Scan Interval 45 ms
Scan Window 3 ms

```

5.3.6 Time Information Profile Application

Perform the following steps to run the Time Information Profile application demo:

1. Follow the steps (1 through 4) from the [ANCS Application](#).
2. Press the SW0 button on the device to read the internally supported characteristic values from the iPhone.
3. The console log on the device side displays the values for all characteristics supported by iPhone internally.

Figure 5-21. Console Display – Date, Time, and Day Information

```

Current Time:[DD:MM:YYYY]: 13-09-2015 [HH:MM:SS]: 15:50:06 Day:SUN Fraction:67
Time Zone 22
DST Offset 00 Standard Time
Current Time:[DD:MM:YYYY]: 13-09-2015 [HH:MM:SS]: 15:50:08 Day:SUN Fraction:223
Time Zone 22
DST Offset 00 Standard Time

```

5.3.6.1 Running the Demo for Android devices

Perform the following steps to run the Time Information Profile application demo for Android devices:

1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
2. After the device is connected, the application displays Continuous Time Service, Next DST Change Service, and Reference Time Update Service.
3. Click on the services to read the characteristic values.
4. Press the SW0 button on the supported platform device to read the internally supported characteristic values from the Android device.
5. The console log on the device side displays the values for all the characteristics supported by the device.

Figure 5-22. Console Display - All Supported Characteristic Values

```

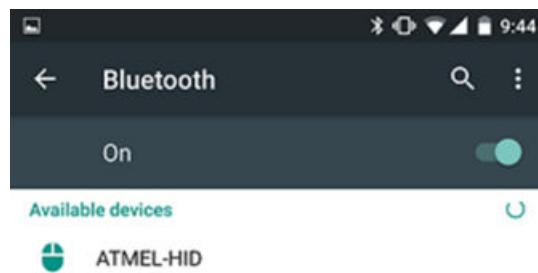
Current Time:[DD:MM:YYYY]: 13-09-2015 [HH:MM:SS]: 04:42:56 Day:SUN Fraction:00
Time Zone 22
DST Offset 00 Standard Time
Time Source = 6 Cellular Network
Accuracy = 255
Day Since Update = 255
Hour Since Update = 255
DST Time is Time:[DD:MM:YYYY]: 13-09-2015 [HH:MM:SS]: 04:42:56 DST Offset is :00
Source = 255
Result = 00

```

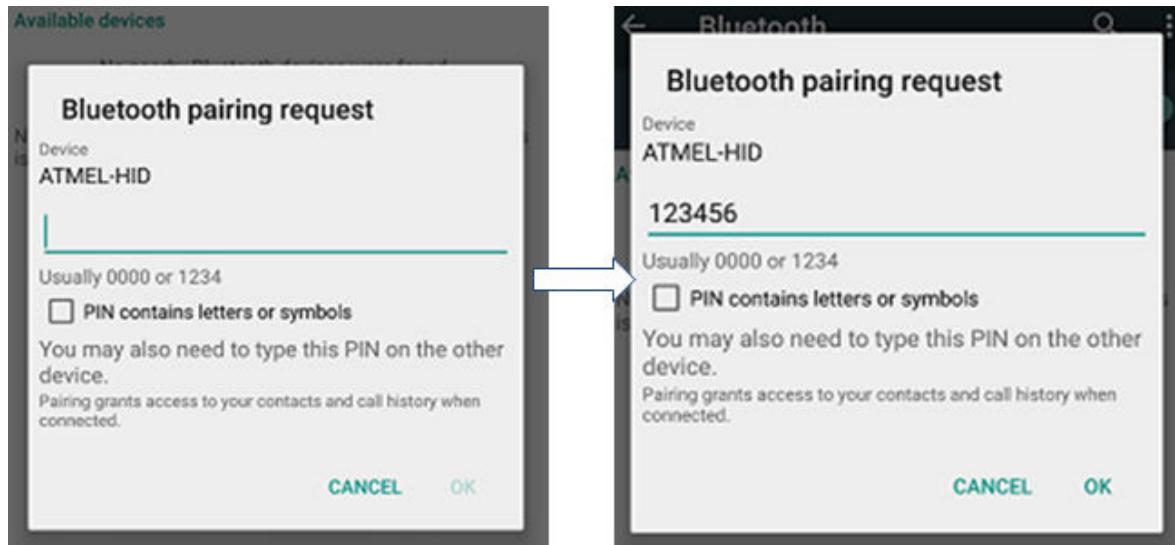
5.3.7 HID Mouse Device Application

Perform the following steps to run the HID Mouse Device application demo:

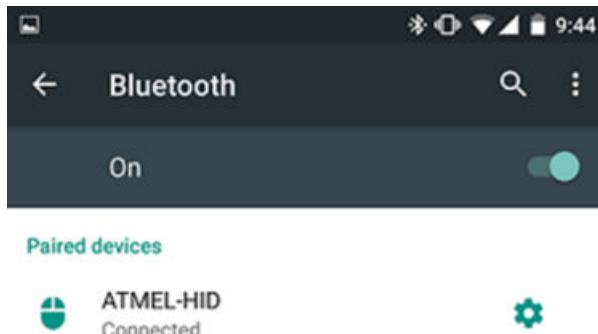
1. Follow the steps from [Initializing the Device](#).
2. In this demonstration, Android device supporting HOGP is used. The HOGP profile is natively supported in Android version 4.4 (Android KitKat) and higher. The mobile phone must include a Bluetooth chipset supporting Bluetooth 4.0 or higher. On the mobile phone, enable Bluetooth in the Settings page to scan for the devices. “ATMEL-HID” appears among the list of scanned devices. Select **ATMEL-HID** to connect to the supported platform device.

Figure 5-23. HID (Mouse) Device Discovery on Bluetooth Settings Page

3. Click **ATMEL-HID** to start the pairing procedure. A pop-up requesting for the pass-key appears. Enter the pass-key “123456” and click **Pair**.

Figure 5-24. Bluetooth Pairing Request

4. After the pairing is completed, the connected device is listed under Paired devices, as shown in the following figure.

Figure 5-25. Paired Devices

5. The HID device side for pairing and connection procedure is shown in the console log.

Figure 5-26. HID Mouse Device Console Log

```

Initializing HID Mouse Application
HID Profile Configured
Initializing SAMB11
BLE-chip id:0x0002000B0
BD Address:0xF8F005F0B299, Address Type:0
Device Started Advertisement
Connected to peer device with address 0x7b5bfcc5cf8d0
Connection Handle 0
Remote device request pairing
Sending pairing response
AT_BLE_CONN_PARAM_UPDATE
Pairing procedure completed successfully

```

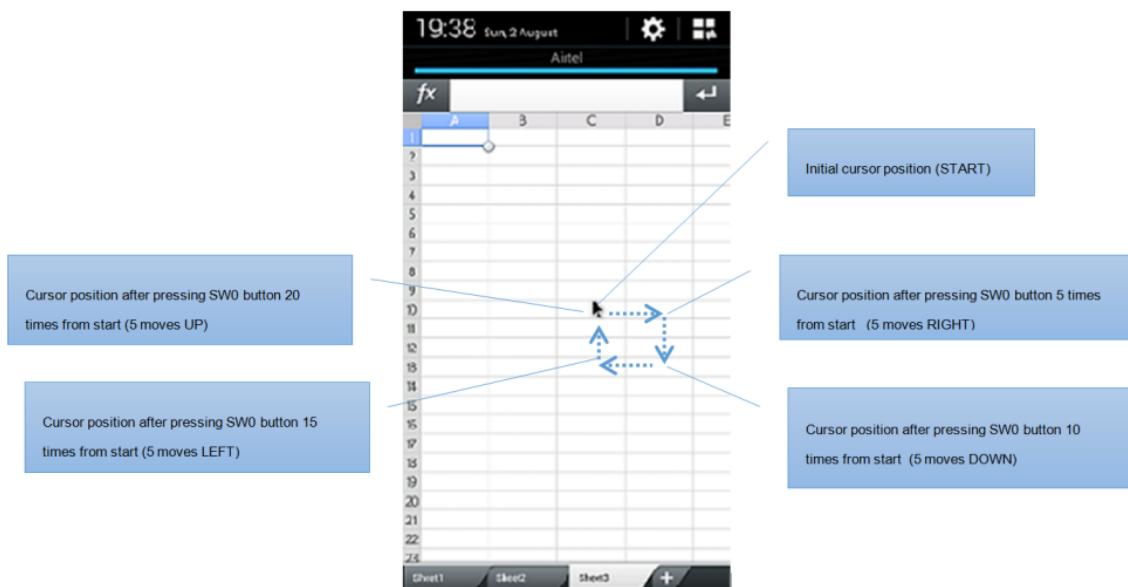
6. After the device is connected to the mobile phone, click on the SW0 button for simulating mouse movement.
7. For every button press, corresponding cursor movement is shown on the HID host as described below:
 - First 5 button presses – cursor moves right
 - Next 5 button presses – cursor moved down
 - Next 5 button presses – cursor moves left
 - Next 5 button presses – cursor moved up

The same sequence is repeated based on the user input. The console log is shown in the following screen.

Figure 5-27. HID Device Console Log for Movement

```
Mouse Right Movement
Mouse Down Movement
Mouse Down Movement
Mouse Down Movement
Mouse Down Movement
Mouse Left Movement
Mouse Left Movement
Mouse Left Movement
Mouse Left Movement
Mouse UP Movement
Mouse UP Movement
Mouse UP Movement
Mouse UP Movement
```

Figure 5-28. Mouse Movement Simulation



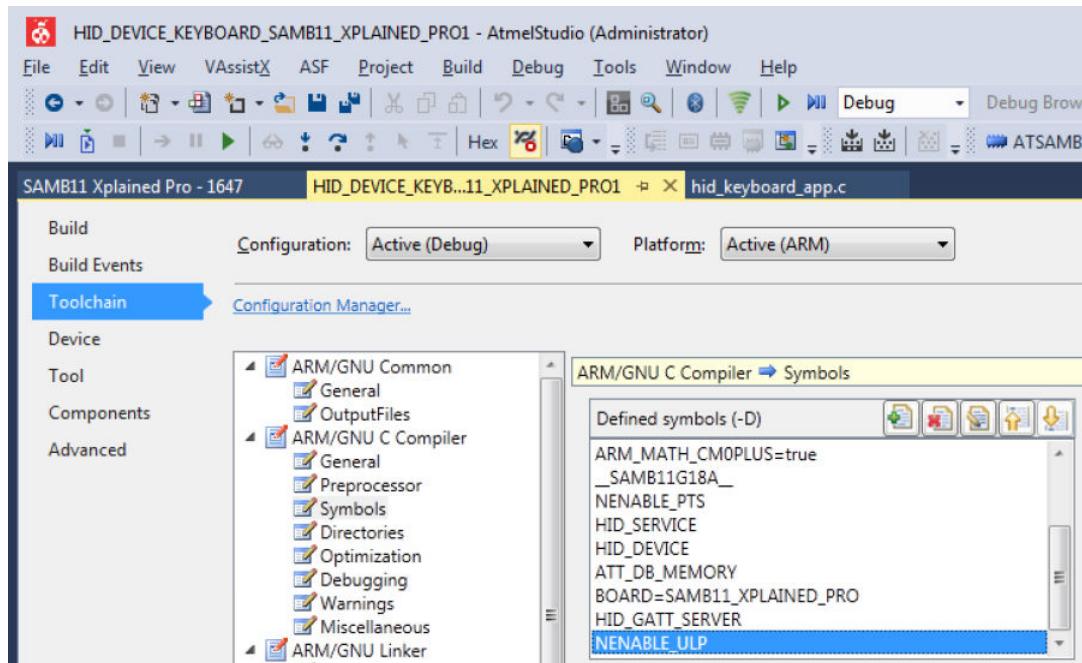
5.3.8 HID Keyboard Device Application

5.3.8.1 Disabling ULP Mode

Perform the following steps to disable the ULP mode for HID Keyboard device application.

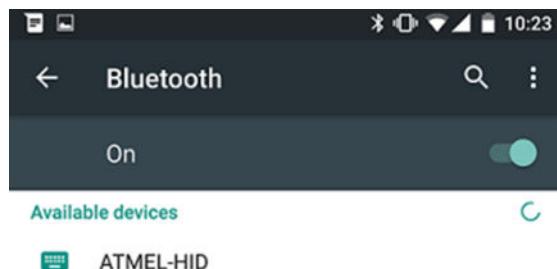
1. Before building the project, ensure that the “NENABLE_ULP” symbol is used, as shown in the following figure.

Figure 5-29. Selecting NENABLE_ULP Symbol to Disable ULP Mode



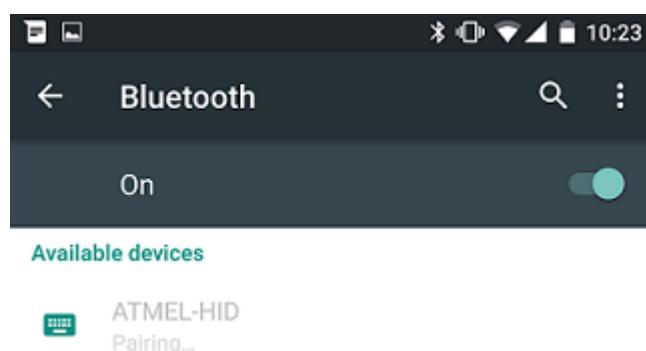
2. Follow the steps from [Initializing the Device](#).
3. In this demonstration, an Android device supporting HOGP is used. The HOGP profile is natively supported in Android version 4.4 (Android KitKat) and higher. The mobile phone must include a Bluetooth chip-set supporting Bluetooth 4.0 or higher. On the mobile phone, enable Bluetooth in the Settings page to scan for the devices. “ATMEL-HID” appears among the list of scanned devices. Select **ATMEL-HID** to connect to the supported platform device.

Figure 5-30. HID (Keyboard) Device Discovery on Bluetooth Settings Page



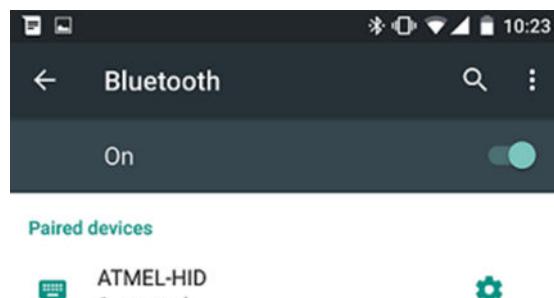
4. Click **ATMEL-HID** to initiate the pairing procedure.

Figure 5-31. Pairing Procedure with HID Device



- After the pairing is completed, the connected device is listed under Paired devices.

Figure 5-32. Paired Devices



- The HID device side for pairing and connection procedure is shown in the console log.

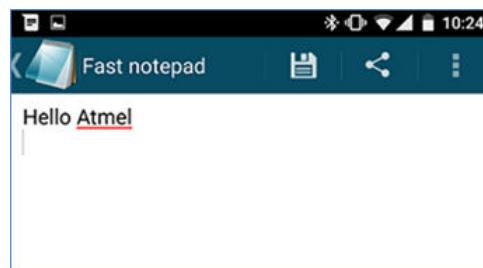
Figure 5-33. HID Keyboard Device Console Log

```

Initializing HID Keyboard Application
HID Profile configured
HID Keyboard Profile Application
BLE_MGR Intializing SAMB11
BLE_MGR BD Address: 0xA12345678912
HID Number of characteristic 7
HID Define service handle 16
HID Report Reference descriptor handle 24
HID Report Reference descriptor ID = 1 :: Type = 1
HID Descriptor Value set successfully
HID_DEVICE Device Started Advertisement|
BLE_MGR ble_connected_state_handler
BLE_MGR Connected to peer device with address 0x142312312112
BLE_MGR Connection Handle = 0
BLE_MGR Remote device request pairing
BLE_MGR Sending pairing response
BLE_MGR ble_pair_done_handle
BLE_MGR Pairing procedure completed successfully
Hid_keyboard_pair_done_callback done
  
```

- After the device connected, start any notepad application on the mobile phone.
- Click **SW0** button on the supported platform device.
- A letter for each press is shown in the application “Fast notepad”.
- The user can see a complete “Hello Atmel” in the application as shown in the following screen.

Figure 5-34. Message Displayed in the Application



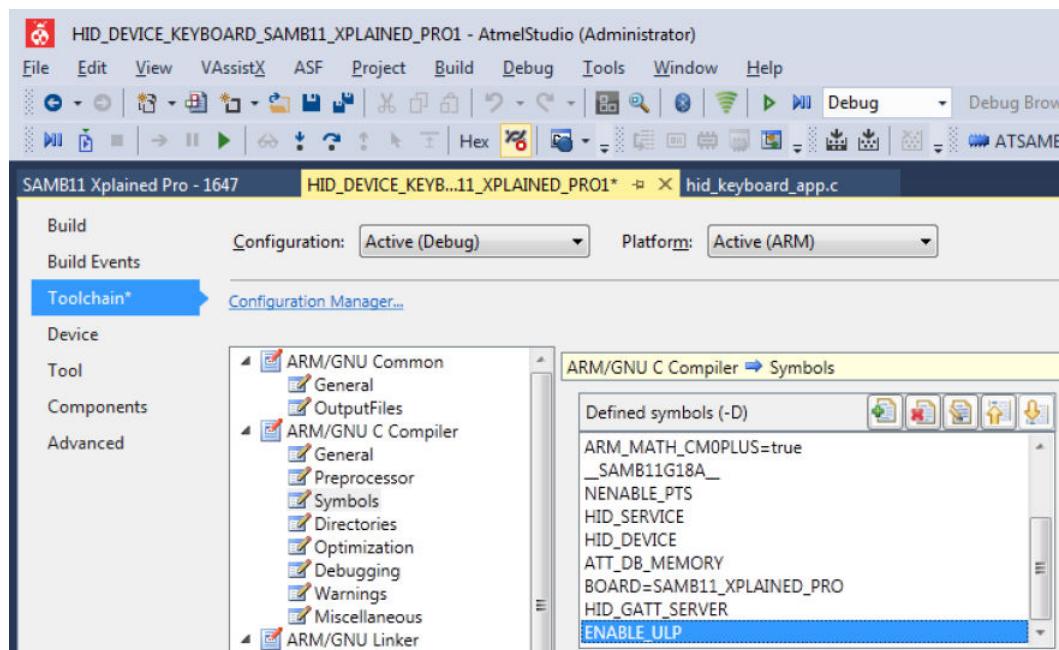
5.3.8.2 Enabling ULP Mode

Perform the following steps to enable the ULP mode for HID Keyboard device application.

Note: Entering into ULP mode with the following hardware changes is required only for SAMB11-MR Xplained Pro. For SAMB11-ZR Xplained Pro AON_GPIO0 is pulled down by default and connected to the SW0 button. Once the device is entered into ULP mode, use AON_GPIO to wake-up the device or use other wake-up sources (BLE Events, AON Timer).

1. Before building the project, ensure that the “ENABLE_ULP” symbol is used, as shown in the following figure.

Figure 5-35. Selecting ENABLE_ULP Symbol for ULP mode



2. Configure the device using cables and a switch, as shown in the following figure.

Figure 5-36. Cables and Switch

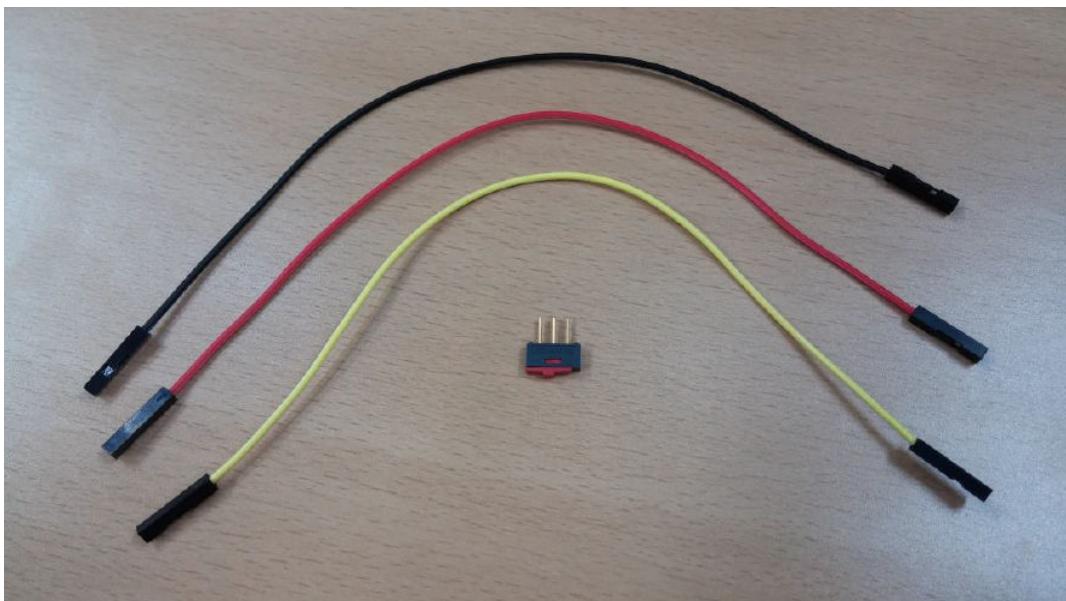
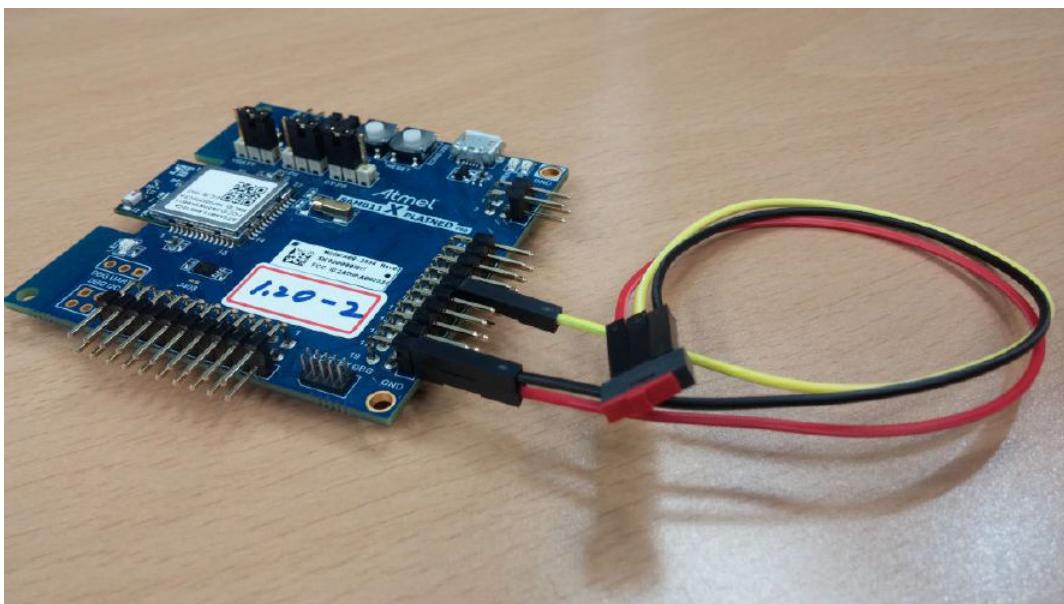


Figure 5-37. Configuring the ATSAMB11 to Control ULP Mode



3. Follow the steps (2 through 7) from [Disabling ULP Mode](#).
4. Adjust the switch to connect pin 9 (AON_GPIO_0) to pin 20 (VCC) to disable ULP mode.



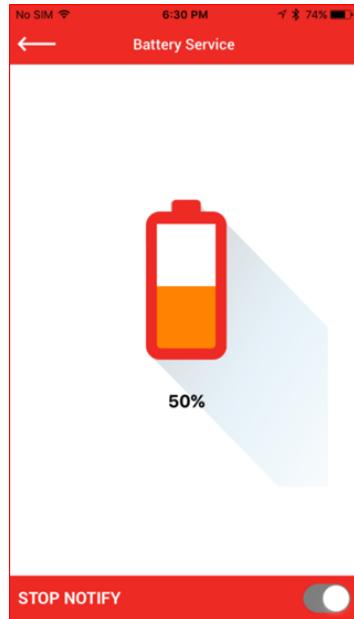
Important: This step must be performed to proceed with the next steps.

5. Click **SW0** button on the supported platform device.
6. A letter for each press shows in the application “Fast notepad”.
7. The user can see a complete “Hello Atmel” in the application, as shown in [Figure 5-34](#).
8. To enable the ULP mode again, adjust the switch to connect pin 9 (AON_GPIO_0) to pin 19 (GND).

5.3.9 Battery Service Application

Perform the following steps to run the Battery Service Application demo:

1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
2. When paired, the application displays the Battery Service and the Generic Information service.
3. Select “Battery Service” to receive notifications for the battery level characteristic. The user can stop receiving the notifications by disabling notifications, as shown in the following figure.

Figure 5-38. Battery Level Characteristic Notification Options

4. On the device side, the console log displays the periodic battery level updates.

```
Battery Level:0%
Battery Level:1%
Battery Level:2%
Battery Level:3%
Battery Level:4%
Battery Level:5%
```

5.3.10 Simple Broadcaster Application

Perform the following steps to run the Simple Broadcaster application demo:

1. Follow the steps (1 and 2) from [Initializing the Device](#).
2. The device is in advertising mode.
3. The following figure shows example logs from the Simple Broadcaster application.

Figure 5-39. Simple Broadcaster Console Display

```
Initializing Broadcaster Application
Initializing SAMB11
BD Address:0xF8F005F23FFF, Address Type:0
Advertisement type set to nonconnectable undirected
Complete name set
Appearance set
Complete list of service uuid16 set
BLE Broadcast data set success
Started Broadcasting
```

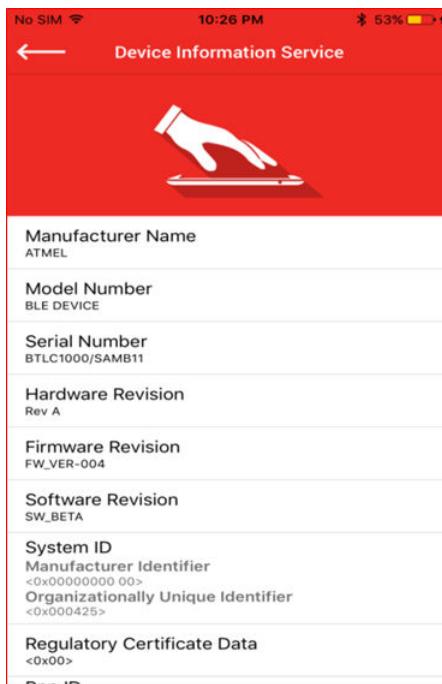
5.3.11 Device Information Service Application

Perform the following steps to run the Device Information Service application demo:

1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
2. When paired, the application displays the Device Information Service.

3. When the Device Information Service is selected, the device information service characteristics can be viewed, as shown in the following screen.

Figure 5-40. Display of Device Information Service Characteristics

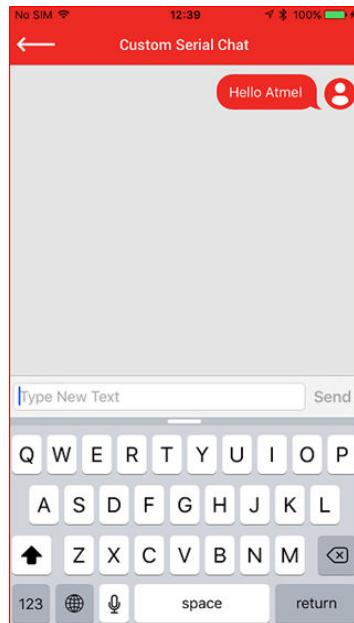


4. The page can be refreshed to get the updated characteristic value of all characteristics.

5.3.12 Custom Serial Chat Profile Application

Perform the following steps to run the Custom Serial Chat Profile application demo.

1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
2. Once pairing is complete, the Custom Serial Chat icon appears on service list page.
3. Click the **Custom Serial Chat** icon. The chat screen appears where the text that is to be sent to the remote device is to be entered, and the text coming from the remote device can be seen.
4. Chat text “Hello Atmel” and send to the remote device.

Figure 5-41. Sending Data to Device

5. The user can also write the text on the console for the device and press the ENTER key to transmit the chat text to the mobile application.

Figure 5-42. Console Log for Sending Data to Remote Device

```
Initializing Custom Serial Chat Application
Initializing SAMB11
BD Address:0xF8F005F23FFF, Address Type:0
Device Started Advertisement
Connected to peer device with address 0xcccf00710852
Connection Handle 0
Peer device request pairing
Sending pairing response
Please Enter the following Pass-code(on the Device):123456
Pairing procedure completed successfully

Hello Atmel

Hello Atmel
```

Figure 5-43. Chat Text Received from ATSAMB11

Note: For more information on Custom Serial Chat service, refer to [Custom Serial Chat Service Specification](#).

5.3.13 Heart Rate Profile Application

Perform the following steps to run the Heart Rate Profile application demo.

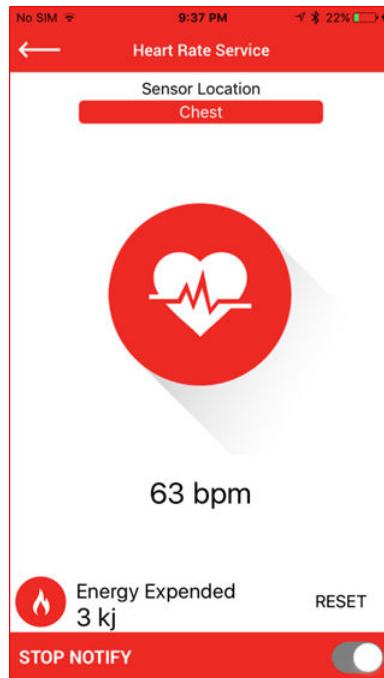
1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
2. After the device is connected, the application displays the Heart Rate and the Device Information Service.
3. When the notifications are enabled the HRM values are displayed, as shown in the console and the corresponding mobile app, the LED on SAMB11-MR/ZR board starts blinking while sending notifications.

```

Notification Enabled
Heart Rate: 50 bpm          RR Values:<100,300>msec User Status:Idle
Heart Rate: 51 bpm          RR Values:<500,700>msec User Status:Idle
Heart Rate: 52 bpm          RR Values:<900,1100>msec User Status:Idle
Heart Rate: 53 bpm          RR Values:<100,300>msec User Status:Idle
Heart Rate: 54 bpm          RR Values:<500,700>msec User Status:Idle
Heart Rate: 55 bpm          RR Values:<900,1100>msec User Status:Idle
Heart Rate: 56 bpm          RR Values:<100,300>msec User Status:Idle
Heart Rate: 57 bpm          RR Values:<500,700>msec User Status:Idle
Heart Rate: 58 bpm          RR Values:<900,1100>msec User Status:Idle
Heart Rate: 59 bpm          RR Values:<100,300>msec User Status:Idle
Energy Expended :3KJ

```

Figure 5-44. Displaying Heart Rate Measurements



4. When Stop Notify is disabled, the console logs display the notifications display as:
Notification Disabled
5. During the connection, the SW0 button is used to disconnect the connection. If no connection exists, the SW0 button is used to start advertisement.

5.3.14 Blood Pressure Profile Application

Perform the following steps to run the Blood Pressure Profile application demo.

1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
2. When the device is connected, the application displays Blood Pressure, Device Information Service and Generic Information.
3. Upon entering the Blood Pressure service page, the mobile application enables the notifications and indications for interim cuff pressure and blood pressure characteristics respectively. The blood pressure sensor device simulated by the device sends the current blood pressure values after receiving the indications enabling the request. The corresponding console logs and mobile application screen are shown in the following figure.

Figure 5-45. Console Log for Blood Pressure Measurements

```

Please Enter the following Pass-code(on other Device):123456
Pairing procedure completed successfully
Notifications enabled by the remote device for interim cuff pressure
Indications enabled by the remote device for blood pressure

Systolic      10 kpa
Diastolic     07 kpa
Map           08 kpa
Pulse rate    60 bpm

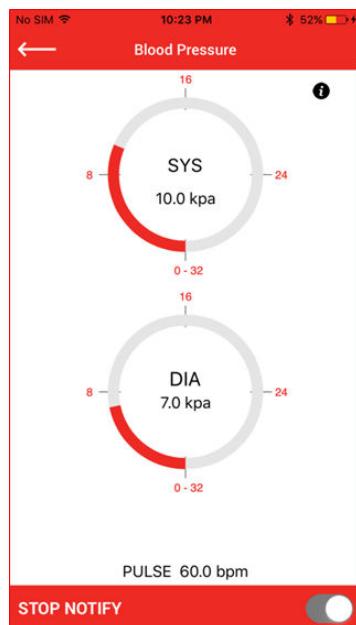
Press the button to receive the blood pressure parameters

Started sending Interim Cuff Pressure Values
Cuff pressure 80 mmhg
Cuff pressure 81 mmhg
Cuff pressure 82 mmhg
Cuff pressure 83 mmhg
Cuff pressure 84 mmhg
Cuff pressure 85 mmhg
Cuff pressure 86 mmhg
Cuff pressure 87 mmhg
Cuff pressure 88 mmhg

The Blood Pressure Values are:
Systolic      81 mmhg
Diastolic     61 mmhg
Map           71 mmhg
Pulserate    61 bpm

Press the button to receive the blood pressure parameters

```

Figure 5-46. Blood Pressure Service Page after Receiving BP Indications

- The SW0 button can be used on SAMB11-MR/ZR to receive updated blood pressure measurements. The blood pressure sensor first sends the interim cuff pressure values as notifications and then sends the final blood pressure measurements as indication. The blood pressure measurements sent by the blood pressure sensor are simulated values. The following figures demonstrate the scenario after the SW0 button press.

Figure 5-47. Console Log for Blood Pressure Values after Button Press

```

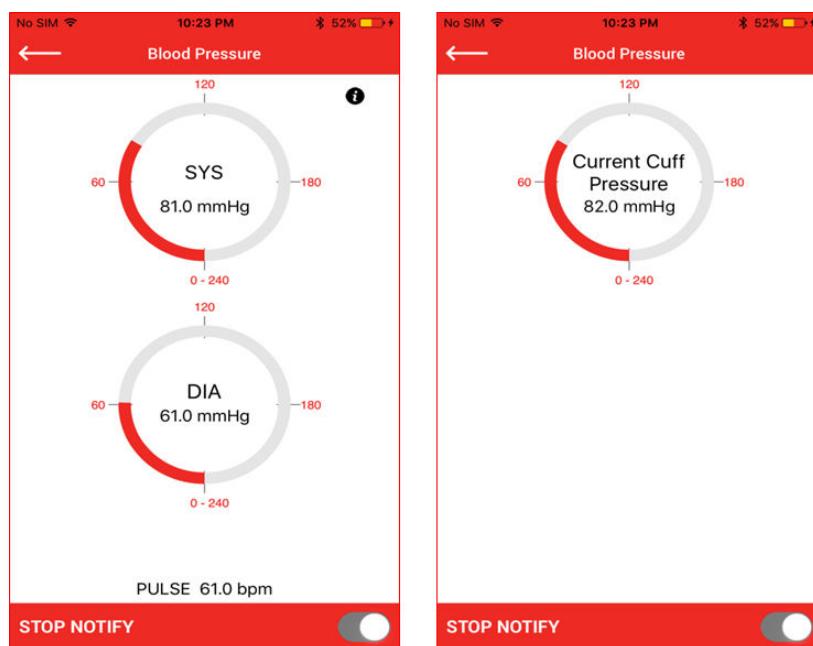
Started sending Interim Cuff Pressure Values
Cuff pressure 80 mmhg
Cuff pressure 81 mmhg
Cuff pressure 82 mmhg
Cuff pressure 83 mmhg
Cuff pressure 84 mmhg
Cuff pressure 85 mmhg
Cuff pressure 86 mmhg
Cuff pressure 87 mmhg
Cuff pressure 88 mmhg

The Blood Pressure Values are:
Systolic      81 mmhg
Diastolic     61 mmhg
Pulse         71 mmhg
Pulserate     61 bpm

Press the button to receive the blood pressure parameters■

```

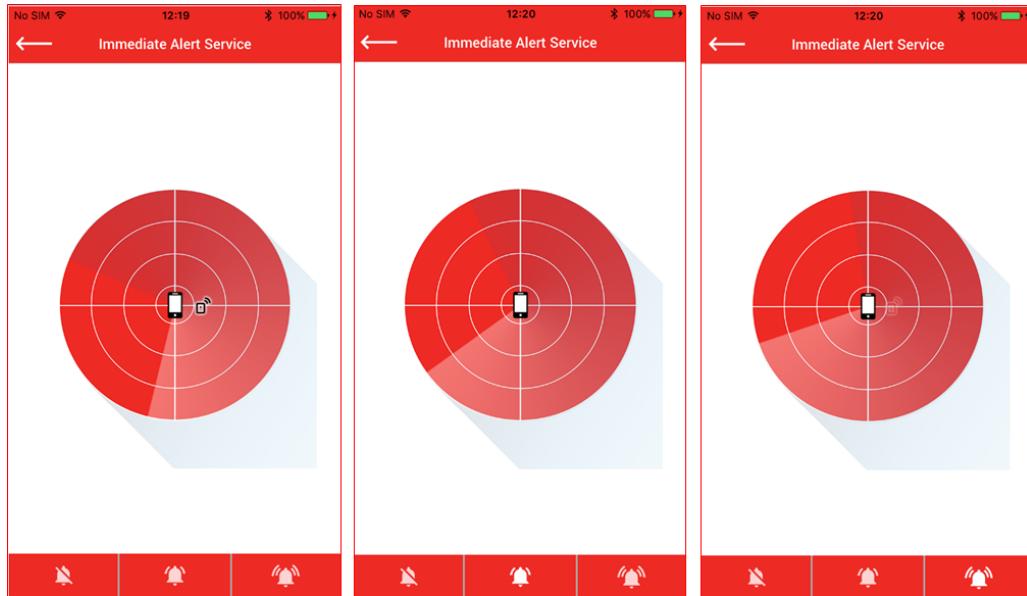
Figure 5-48. Blood Pressure Service Pages after Receiving Measurement Data on Button Press



5.3.15 Find Me Profile Application

Perform the following steps to run the Find Me Profile application demo.

- Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
- After the device is connected, the application displays a service page that includes Immediate Alert Service and Generic Information.
- Since the service level connection is established, notifications are shown based on the alert level settings as depicted in the following figures.

Figure 5-49. Sending Alerts to Find Me Target ATTEL-FMP

4. On the device side, the console log is displayed as:

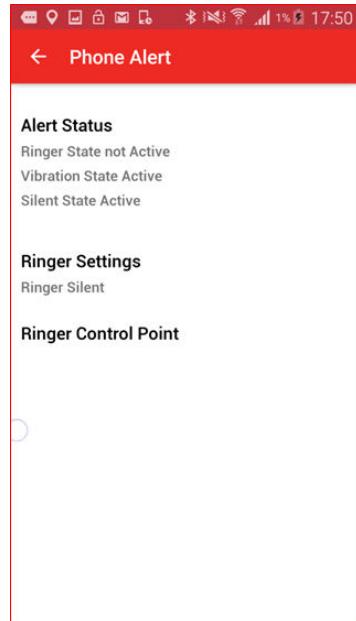
```
Find Me : Mild Alert
Find Me : High Alert
Find Me : No Alert
```

5.3.16 Phone Alert Status Application

Perform the following steps to run the Phone Alert Status application demo.

1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
2. After connection, the application displays the Phone Alert Status Service page.
3. The notifications are automatically enabled and the application reads the values of “Alert Status”, “Ringer Settings”, and “Ringer Control Point” characteristics. These are updated on the mobile application, as illustrated in the following figure.

Figure 5-50. Displaying the Characteristics of the Phone Alert Service



4. Press the SW0 button. The device is set to different modes (see [Phone Alert Status Profile Application](#)) by using the notifications and the corresponding console logs are displayed.

Figure 5-51. Phone Alert Status Console Log

```

button Pressed
Device to silent

AT_BLE_CHARACTERISTIC_WRITE_CMD_CMP
AT_BLE_CHARACTERISTIC_WRITE_COMMAND : SUCCESS

button Pressed
Device to Mute Once

AT_BLE_CHARACTERISTIC_WRITE_CMD_CMP
AT_BLE_CHARACTERISTIC_WRITE_COMMAND : SUCCESS

button Pressed
Device to cancel mute

AT_BLE_CHARACTERISTIC_WRITE_CMD_CMP
AT_BLE_CHARACTERISTIC_WRITE_COMMAND : SUCCESS

button Pressed
reading the alert status and ringer setting
pas_client_char_read_response_handler
Alert setting read :
pas_client_char_read_response_handler
Alert Status read:
Ringer State Active
Vibrate State Active
Display State Active

```

5.3.17 Alert Notification Profile Application

Perform the following steps to run the Alert Notification Profile application demo:

1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).

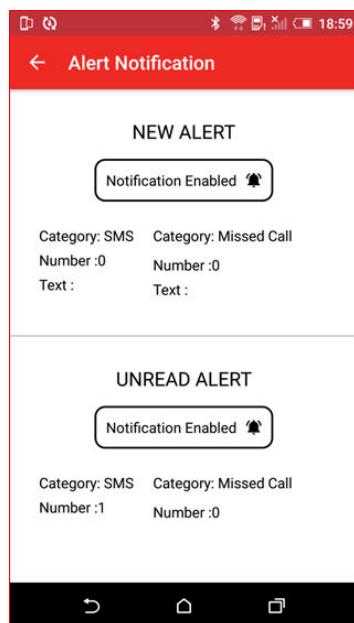
2. When connected, the application displays the Alert Notification service page. The console log displays the new and unread alert categories.

Figure 5-52. Alert Notification Categories

```
Alert Notification Profile Application
Initializing SAMB11
BD Address:0xF8F005F23FFF, Address Type:0
Device is in Advertising Mode
Connected to peer device with address 0xcccf00710852
Connection Handle 0
Peer device request pairing
Sending pairing response
Please Enter the following Pass-code(on other Device):123456
Pairing procedure completed successfully
```

3. Enable the notifications by using the SW0 button. The mobile application reflects the status as shown.

Figure 5-53. Alert Notification Screen on Microchip SmartConnect Application



4. The user can trigger a missed call to the Android device or send an SMS. The corresponding notification then gets displayed on the device side in the console logs.

Figure 5-54. Console Display for Missed Call Alert and SMS Alert Notifications

```

New Alert received
The no of new alerts are 1
The alert type is :
Missed Call alert
Unread alert received
The no of unread alerts are 0
New Alert received
The no of new alerts are 2
The alert type is :
Missed Call alert
Unread alert received
The no of unread alerts are 1
The alert type is :
Missed Call alert
New Alert received
The no of new alerts are 0
New Alert received
The no of new alerts are 0
Unread alert received
The no of unread alerts are 2
The alert type is :
Sms/Mms alert
Unread alert received
The no of unread alerts are 2
The alert type is :
Missed Call alert

```

5.3.18 Multi-Role Peripheral Multi-Connect Application

Perform the following steps to run the Multi-Role Peripheral Multi-Connect application demo.

- Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#). The device initially acts as a GAP Peripheral and starts advertisement with Battery Service UUID in the advertisement data. Perform all the steps from [Battery Service Application](#).
- The device starts scanning and displays the devices found, as shown in the following figure.

Figure 5-55. Multi-Role Peripheral Multi-Connect Application – Scanning Devices

```

Initializing Multi-Role Peripheral Multi-Connection Application
SAMB11 XPro Module:MR SAMB11-MR
Initializing SAMB11
SAMB11 Chip ID: 0x2000B1
BD Address:0xF8F005F36701, Address Type:0
BluSmartSDK Firmware Version:6.1.6991
SAMB11 RF Version:0x10000000
BLE Started Advertisement
Advertisement stopped due to timeout
Scanning...Please wait...
Info: Device found address [0] 0xFD8095FF6167
Info: Device found address [1] 0xF8F005F35B58 ---FMP
Select Index number to Connect or [s] to scan or [a] to advertise

```

- Set up another ATSAMB11-MR/ZR device with the Find Me application example loaded. Follow the steps (1 through 3) from [Running the Demo](#). The device starts advertising.
- The GAP Central (Find Me Locator) scans and then displays the list of all BLE devices that are advertising. Find Me Target devices (GATT server role) are indicated with tag “--FMP”. Select the appropriate index number for the Find Me Target. The GAP Central (Find Me Locator) connects to the selected peer device.

Figure 5-56. Connecting GAP Central (Find Me Locator) with GAP Peripheral (Find Me Target)

```
SAMB11 RF Version:0x10000000
BLE Started Advertisement
Advertisement stopped due to timeout
Scanning...Please wait...
Info: Device found address [0] 0xF8F005F35B58 ---FMP

Select Index number to Connect or [s] to scan or [a] to advertise
0
FMP Connect request sent
Connected to peer device with address 0xf8f005f35b58
Connection Handle 0
central device connected
BLE Started Advertisement
Enter the Passkey(6-Digit) in Terminal:123456
Entered Pass-code:123456
Pairing procedure completed successfully
Immediate Alert service discovered
Platform Event: Timer
RSSI update: Handle 0
Rx Power(RSSI):-043 dBm
---No Alert
Platform Event: FMP
Battery Level:0%
Platform Event: Timer Exit
```

- The ATSAMB11 as a GAP Central pairs with the connected peripheral. Then the ATSAMB11-MR/ZR acts as a GAP Peripheral by advertising with Battery Service UUID in the advertisement data. Now the ATSAMB11-MR/ZR sends alert levels as a GAP Central, sends battery level notifications to the device connected as a GAP Peripheral, and also starts advertising with Connectable advertisement packets.

Figure 5-57. GAP Peripheral (ATSAMB11) Connected to GAP Central (Mobile) and Transferring the Data

```
Rx Power(RSSI):-039 dBm---No Alert
Rx Power(RSSI):-042 dBm---No Alert
Rx Power(RSSI):-038 dBm---No Alert
Rx Power(RSSI):-039 dBm---No Alert
Rx Power(RSSI):-037 dBm---No Alert
Connected to peer device with address 0x712ff8994bf6
Connection Handle 1
Sending pairing response
Please Enter the following Pass-code(on other Device):123456
Rx Power(RSSI):-038 dBm---No Alert
Rx Power(RSSI):-036 dBm---No Alert
Rx Power(RSSI):-038 dBm---No Alert
Pairing procedure completed successfully
Battery Level:0%
Rx Power(RSSI):-037 dBm---No Alert
Battery Level:1%
Rx Power(RSSI):-036 dBm---No Alert
Battery Level:2%
Rx Power(RSSI):-037 dBm---No Alert
Battery Level:3%
```

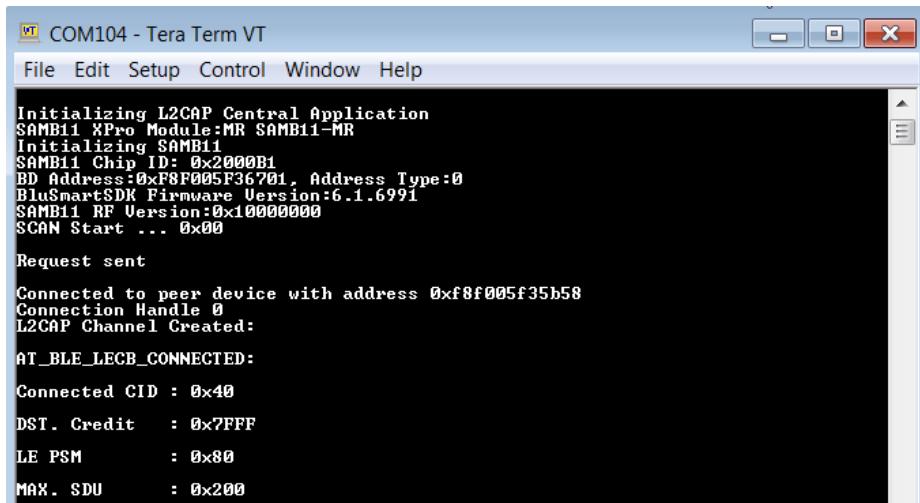
- The ATSAMB11-MR/ZR acting as a GAP Peripheral (BAS) can connect to seven GAP central devices (mobile devices through the Microchip SmartConnect application). Now, the ATSAMB11-MR/ZR continues to behave as Find Me Locator (GAP Central) and Battery Service Application (GAP Peripheral) simultaneously with eight active connections. Continuous data transfer happens on all the links by the ATSAMB11-MR/ZR and even if one link gets disconnected, the data transfer happens on the other links.

5.3.19 L2CAP Throughput Application

This demonstration requires two ATSAMB11-MR/ZR devices. Program one ATSAMB11-MR/ZR device with the L2CAP Peripheral and another one with the L2CAP Central application example. Perform the following steps to run the Throughput application demo:

1. Follow the steps (1 and 2) from [Initializing the Device](#) for both devices.
 2. The device initializes start-up.
 3. The Central device starts scanning and subsequently connects with the desired peripheral device.
- The following log shows that both devices connected to confirm the connection status.

Figure 5-58. L2CAP Central Connection with L2CAP Peripheral



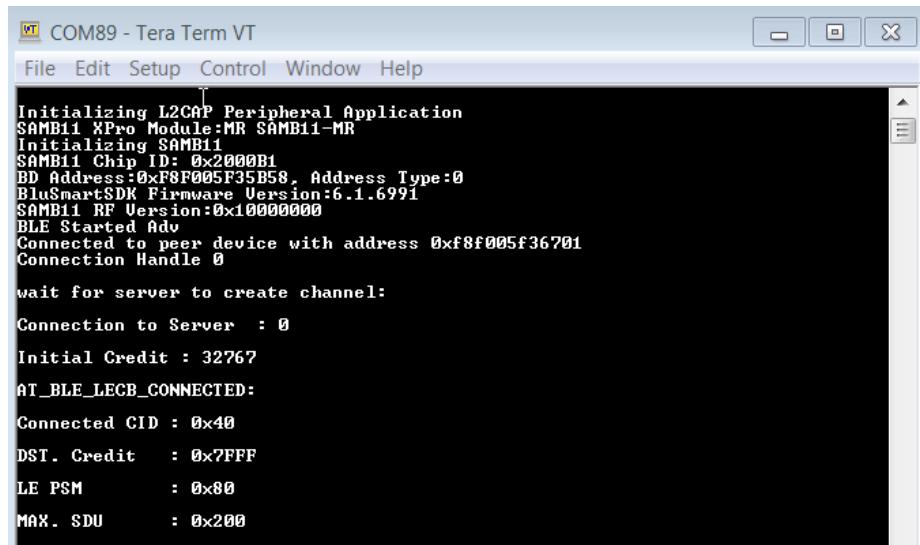
```
VT COM104 - Tera Term VT
File Edit Setup Control Window Help
Initializing L2CAP Central Application
SAMBI1 XPro Module:MR SAMBI1-MR
Initializing SAMBI1
SAMBI1 Chip ID: 0x2000B1
BD Address:0x8F005F36701, Address Type:0
BluSmartSDK Firmware Version:6.1.6991
SAMBI1 RF Version:0x10000000
SCAN Start ... 0x00

Request sent

Connected to peer device with address 0xf8f005f35b58
Connection Handle 0
L2CAP Channel Created:

AT_BLE_LECB_CONNECTED:
Connected CID : 0x40
DST. Credit   : 0x7FFF
LE PSM        : 0x80
MAX. SDU      : 0x200
```

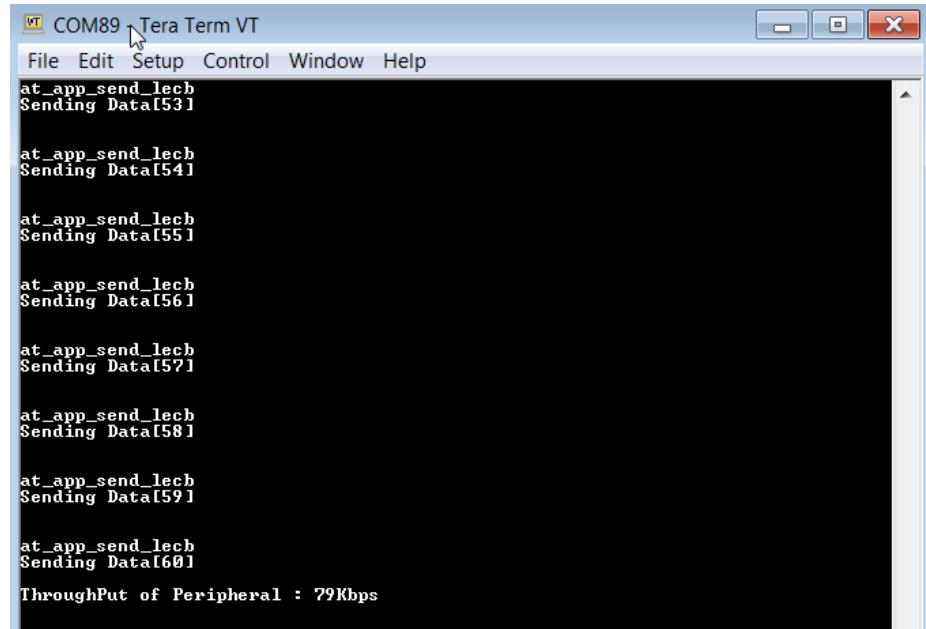
Figure 5-59. L2CAP Peripheral Connection with a L2CAP Central



```
VT COM89 - Tera Term VT
File Edit Setup Control Window Help
Initializing L2CAP Peripheral Application
SAMBI1 XPro Module:MR SAMBI1-MR
Initializing SAMBI1
SAMBI1 Chip ID: 0x2000B1
BD Address:0x8F005F35B58, Address Type:0
BluSmartSDK Firmware Version:6.1.6991
SAMBI1 RF Version:0x10000000
BLE Started Adv
Connected to peer device with address 0xf8f005f36701
Connection Handle 0

wait for server to create channel:
Connection to Server : 0
Initial Credit : 32767
AT_BLE_LECB_CONNECTED:
Connected CID : 0x40
DST. Credit   : 0x7FFF
LE PSM        : 0x80
MAX. SDU      : 0x200
```

4. Once the connection is established, the peripheral device keeps sending the specified data and the central device receives the same data in a given time. Eventually, calculated Throughput is displayed on the console for both central and peripheral.

Figure 5-60. L2CAP Peripheral Final Throughput Value


at_app_send_lecb
Sending Data[53]

at_app_send_lecb
Sending Data[54]

at_app_send_lecb
Sending Data[55]

at_app_send_lecb
Sending Data[56]

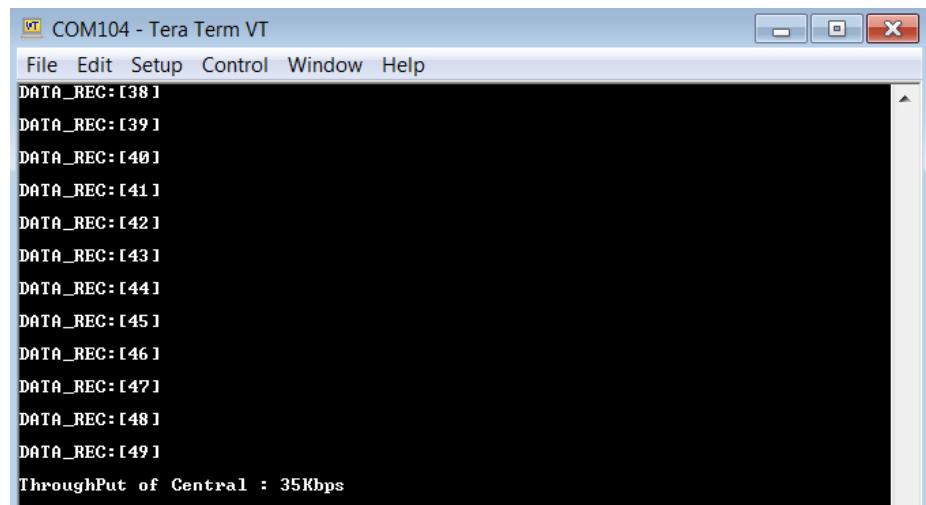
at_app_send_lecb
Sending Data[57]

at_app_send_lecb
Sending Data[58]

at_app_send_lecb
Sending Data[59]

at_app_send_lecb
Sending Data[60]

ThroughPut of Peripheral : 79Kbps

Figure 5-61. L2CAP Central Final Throughput Value


DATA_REC:[38]
DATA_REC:[39]
DATA_REC:[40]
DATA_REC:[41]
DATA_REC:[42]
DATA_REC:[43]
DATA_REC:[44]
DATA_REC:[45]
DATA_REC:[46]
DATA_REC:[47]
DATA_REC:[48]
DATA_REC:[49]

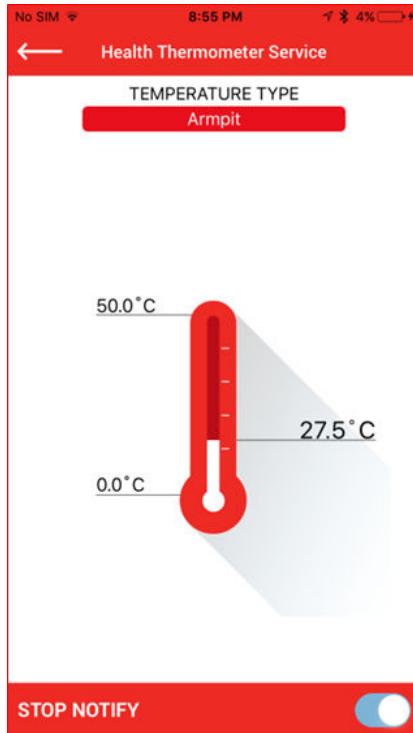
ThroughPut of Central : 35Kbps

5.3.20 Health Thermometer Profile Application

Perform the following steps to run the Health Thermometer Profile application demo.

1. Establish the connection between the device and mobile phone using the procedure listed in [Running the Demo](#).
2. When paired, the application displays the Health Thermometer Service and the Generic Information service.

Figure 5-62. Health Thermometer Connected Services



3. The temperature value, RSSI and the device name are displayed on the mobile application. Console log is shown in the following screen.

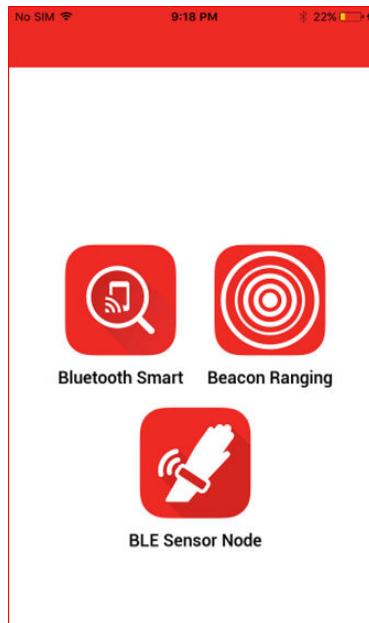
Figure 5-63. Console Log after Connection, Pairing, and with Notifications

- To change the body measurement location, press the user button (SW0) on the SAMB11-MR/ZR. The new value is updated in the application.
 - Going back to the scanning screen on the Microchip SmartConnect application disconnects the device with the mobile application.

5.3.21 iBeacon Application

Perform the following steps to run the iBeacon application demo.

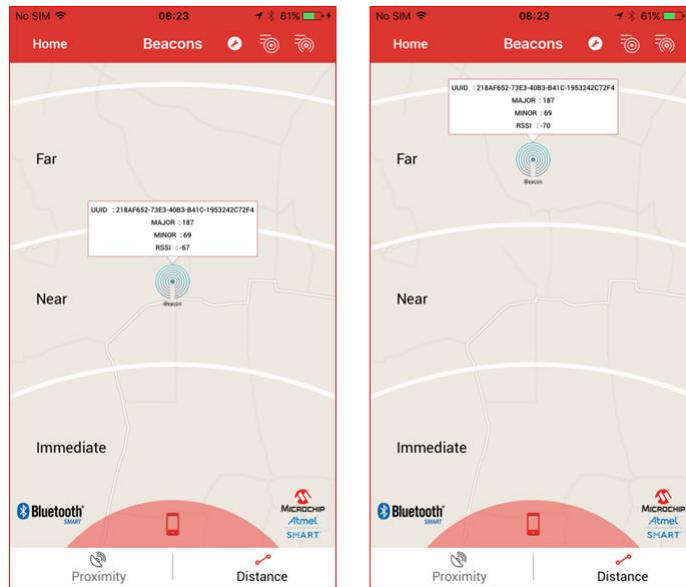
1. Follow the steps (1 and 2) from [Initializing the Device](#).
 2. Start the Microchip SmartConnect application on the iPhone/Android.

Figure 5-64. Beacon Radar Profile App Launch Screen

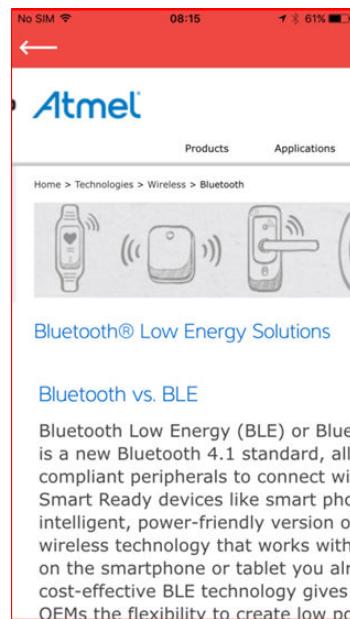
3. Click **Beacon Ranging**. The beacon application is launched to show the positioning of beacon device with respect to mobile phones and support following modes:
 - **Proximity** – used to display beacon specific information when the mobile device comes in close proximity to a given beacon. This mode also shows the corresponding product-related information configured for this particular beacon device.
 - **Distance** – used to indicate the distance between beacon device and the mobile.

Figure 5-65. Beacon Radar Application Initial Screen

4. Click **iBeacon** to check the Major, Minor and RSSI value. The RSSI value is automatically updated based on the movement of the scanner device, as shown in the following figure.

Figure 5-66. Beacon Radar Application in Distance Mode

- Inside the proximity mode, if the scanner device is very near to the beacon the product information can be seen when the user is in close proximity to a given beacon device. When the user moves away from the beacon device information content is no longer shown, indicating that the user has moved away from the beacon device. Optionally, the message can be closed by clicking on close.

Figure 5-67. Beacon Radar Application in Proximity Mode

5.3.22 AltBeacon Application

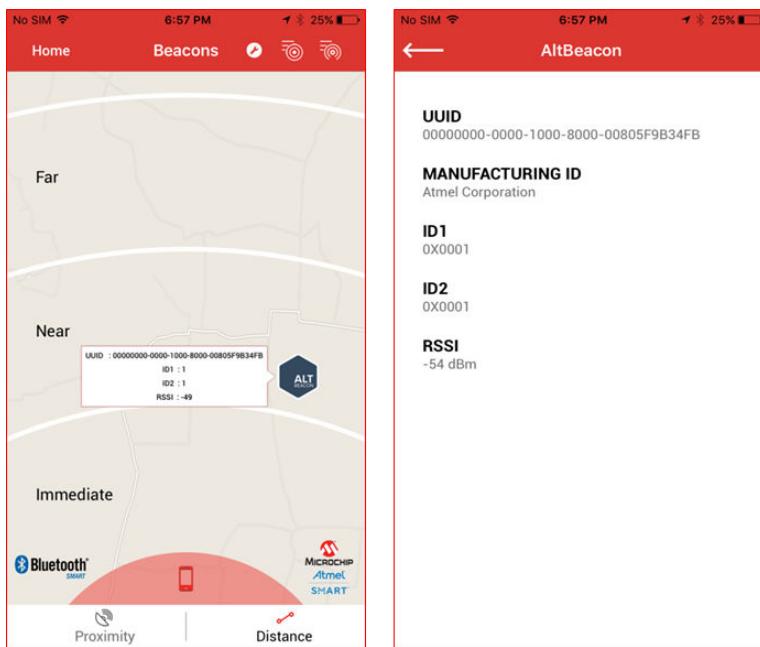
Perform the following steps to run the AltBeacon application demo.

- Follow the steps (1 and 2) from [Initializing the Device](#).
- The beacon application initialization is displayed in the console.

```
Initializing AltBeacon Application
BLE AltBeacon Advertisement started
```

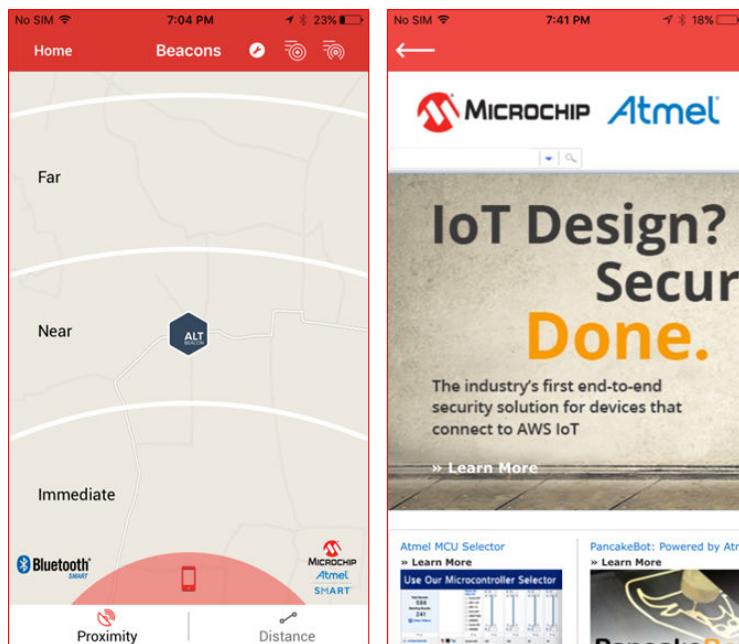
3. Start the Microchip SmartConnect application and click **Beacon Ranging** from the mobile phone (see Figure 5-64). In this demonstration, an iPhone is used to run the application.
4. Tap on the AltBeacon icon for Major, Minor and UUID Value. The RSSI values are automatically updated based on the movement of the scanner device. For more details about the AltBeacon device, tap on the pop-up message (which shows UUID, ID1 and ID2 values), as shown in the following figures.

Figure 5-68. AltBeacon Radar Application in Distance Mode



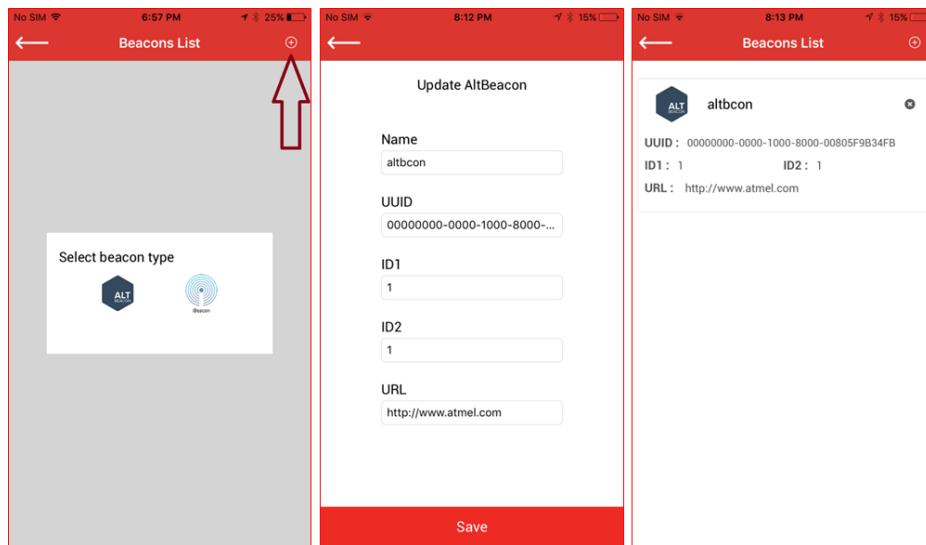
5. In proximity mode, the application opens the configured URL whenever the user comes in close proximity to the configured beacon device. When the user moves away from the beacon device the configured beacon is not shown, indicating the user moved away from the beacon device.

Figure 5-69. AltBeacon Radar Application in Proximity Mode



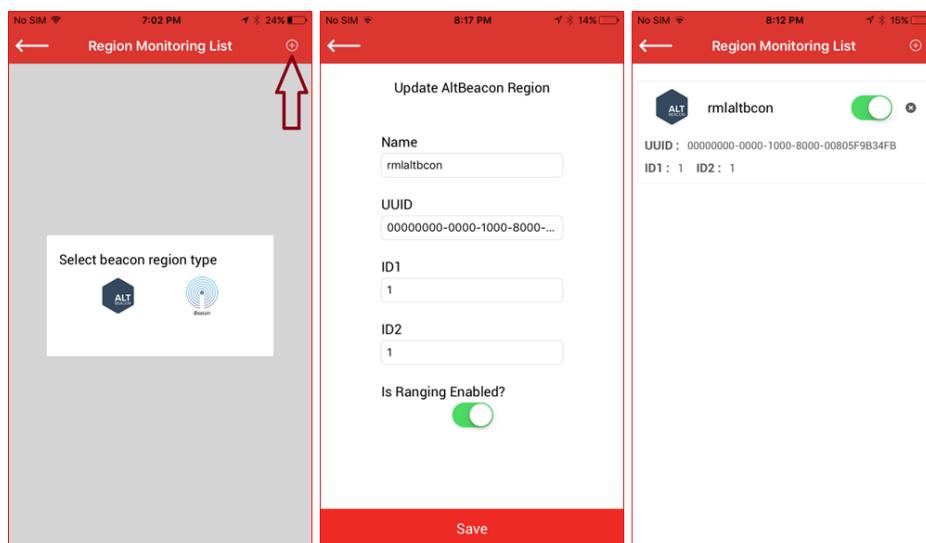
6. A new AltBeacon can be added to the Beacon list using the add button, as illustrated in the following figure.

Figure 5-70. Adding new beacon



7. A new AltBeacon can be added into the Region monitoring list using the add button, as illustrated in the following figure.

Figure 5-71. Adding new beacon in Region Monitoring List



Note: The Region Monitoring List is supported on iOS, and not on Android devices.

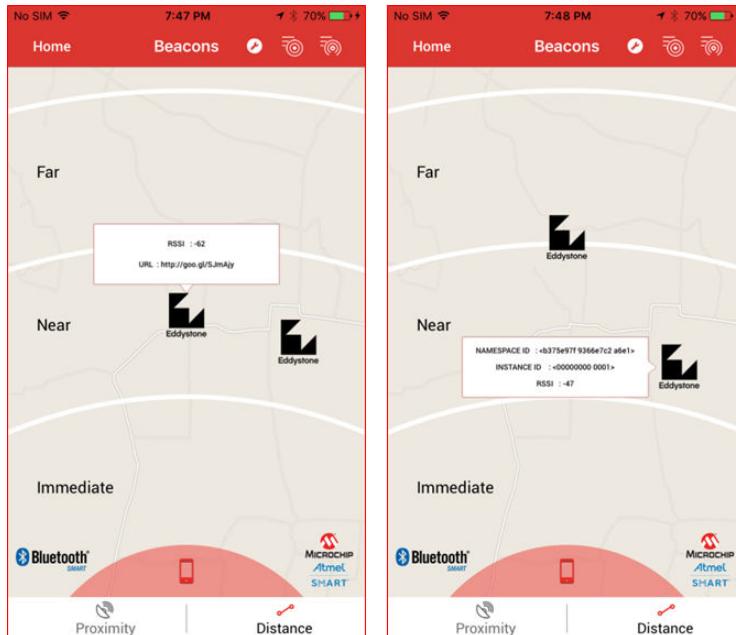
5.3.23 Eddystone Beacon Application

Perform the following steps to run the Eddystone Beacon application demo:

1. Follow the steps (1 and 2) from [Initializing the Device](#).
2. The beacon application initialization is displayed on the console.

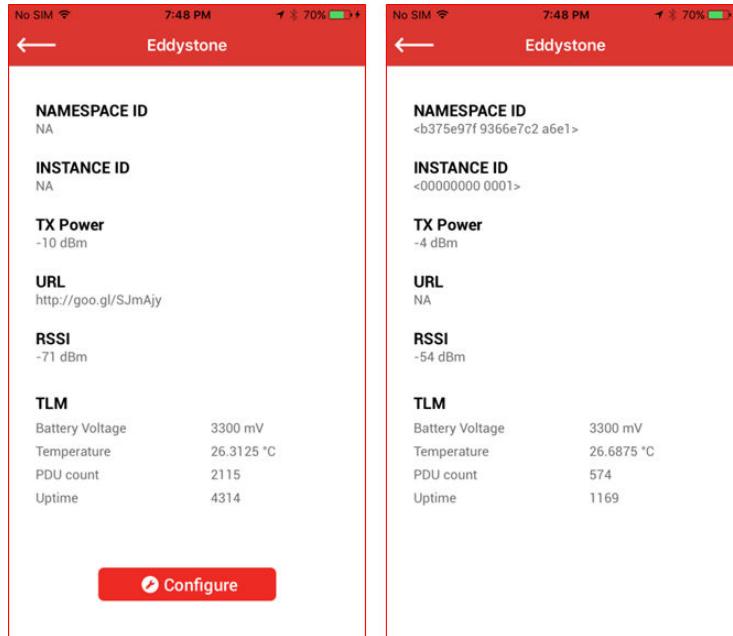
```
SAMB11 XPro Module:MR SAMB11-MR
Initializing SAMB11
SAMB11 Chip ID: 0x2000B1
BD Address:0xF8F005F36701, Address Type:0
BluSmartSDK Firmware Version:6.1.6991
SAMB11 RF Version:0x10000000
Eddystone beacon started
```

- Adv count: 1
Tx URL
3. Start the Microchip SmartConnect application and click **Beacon Ranging** from the mobile phone (see [Figure 5-64](#)). In this demonstration, an iPhone is used to run the application.
 4. Open the **Beacons** navigation tab to view the ranging screen. The Eddystone beacon device is shown on the ranging screen with the Eddystone icon. The position of the beacon is based on the strength of the signal received from RSSI. Click the **Beacon** icon to see a pop-up window showing the identity of the frame; in the case of the `EDDYSTONE_URL_APP`, the shortened URL value is shown and in the case of the `EDDYSTONE_UID_APP`, Namespace ID and Instance ID is shown.
- Figure 5-72. Eddystone Beacons (both UID and URL beacons) ranged by Microchip SmartConnect Application**



5. Click the beacon pop-up window to view detailed information. The detailed view shows UID/URL and telemetric information like battery voltage, beacon temperature, time since power-on, and so on. This telemetric information is obtained from the Eddystone-TLM frames which are interleaved with Eddystone identifying frames (UID/URL).

Figure 5-73. Detailed view of the Eddystone URL and UID beacon



6. In the EDDYSTONE_URL_APP, the detailed beacon information screen shows a Configure button. Click the **Configure** button. It requests that the user puts the beacon into Configuration mode. The SW0 hardware button present on the SAM B11 Xplained Pro board has to be long pressed (around 3 seconds) to enter into Configuration mode.
7. Connect to the beacon in Configuration mode as shown in [Figure 5-74](#). Once connected, the configurable beacon parameters are listed out as shown in [Figure 5-75](#).

Figure 5-74. Connecting to Beacon in Configuration Mode

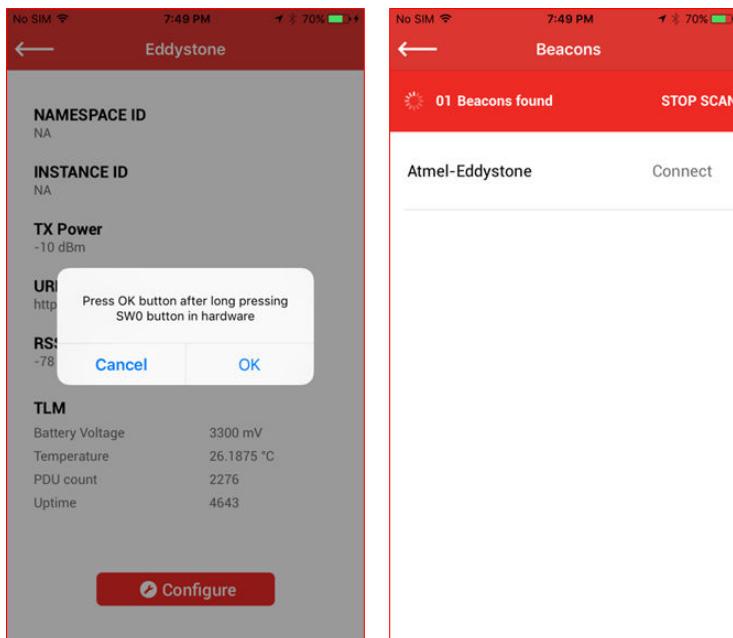
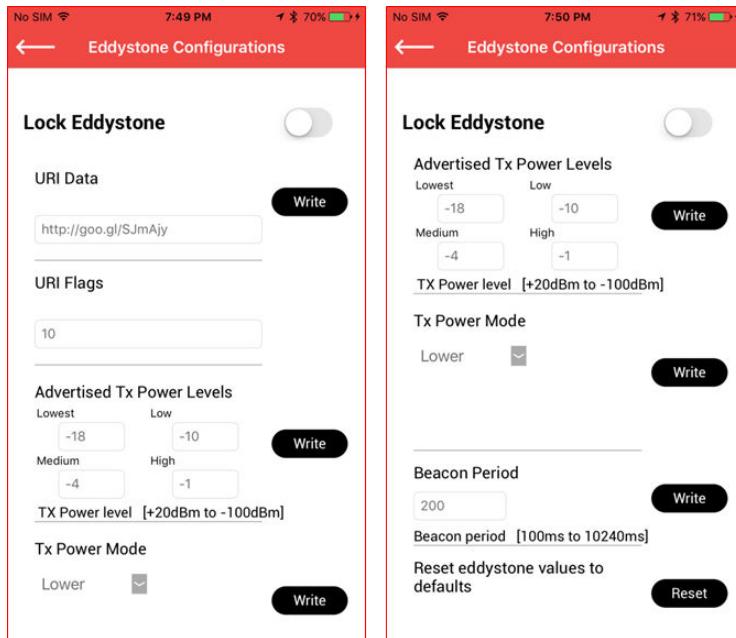


Figure 5-75. Beacon Configuration Screen

- Update the URL, Tx Power mode, beacon period, etc. and then save. Now, disconnect from the beacon and enter the ranging screen. Once disconnected, the beacon device (ATSAMB11-MR/ZR) enters into Beacon mode and start sending Eddystone URL frames with the updated values. The ranging console log screen shows the beacon with new URL value.

Figure 5-76. Eddystone Beacon Console Log

```
COM104 - Tera Term VT
File Edit Setup Control Window Help
SAMBI1 XPro Module:MR SAMBI1-MR
Initializing SAMBI1
SAMBI1 Chip ID: 0x2000B1
BD Address:0xF8F005F36701, Address Type:0
BluSmartSDK Firmware Version:6.1.6991
SAMBI1 RF Version:0x10000000
Eddystone beacon started
Adv count: 1
Tx URL
Adv count: 22
Tx ILM
Adv count: 24
Tx URL
Adv count: 45
Tx ILM
Adv count: 47
Tx URL
Adv count: 68
Tx ILM
Adv count: 70
Tx URL
Adv count: 91
Tx ILM
Adv count: 93
Tx URL
Eddystone-URL config mode
Connected to peer device with address 0x6353594b5df2
Connection Handle 0
Please Enter the following Pass-code<on other Device>:123456
Pairing procedure completed successfully
Paired
Device disconnected Reason:0x13 Handle=0x0
Eddystone beacon mode
Adv count: 336
Tx URL
Adv count: 357
Tx ILM
Adv count: 359
Tx URL
Adv count: 380
Tx ILM
```

- The beacon configuration page also provides a reset button that can set all the parameters to default factory settings.

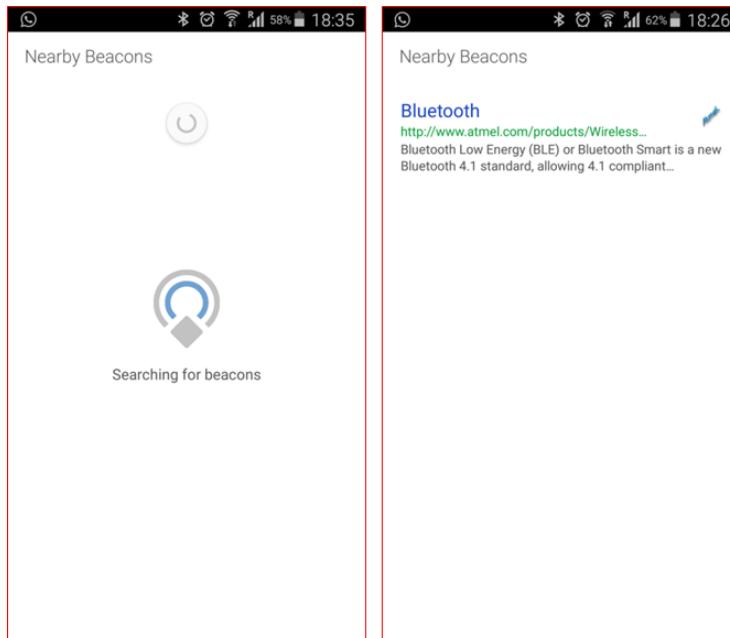
5.3.23.1 Demo with Physical Web Application

Eddystone is the backbone of the Physical Web initiative from Google. For more information on the Physical Web, refer to <https://google.github.io/physical-web/>.

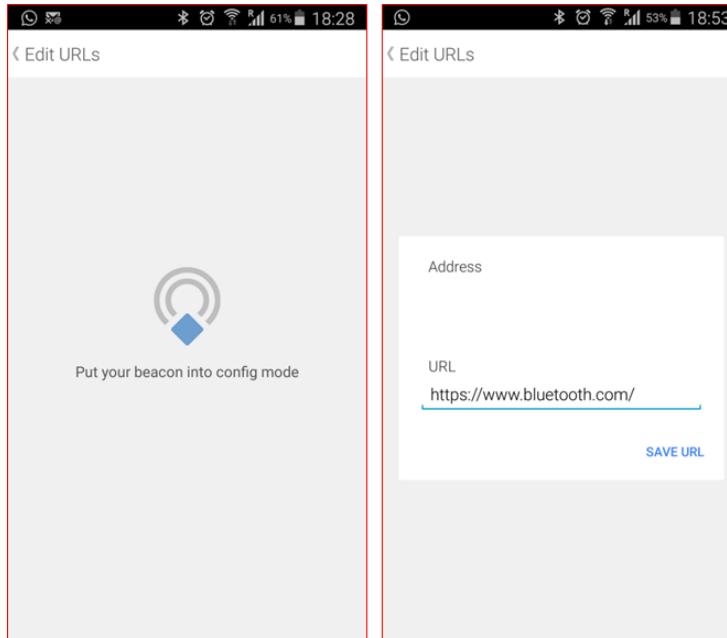
The following demo shows how the Eddystone application running on an ATSAMB11-MR/ZR device works seamlessly with the Physical Web Android application.

1. Install the Physical Web Android application in a BLE compatible android device.
2. Build and run the `EDDYSTONE_URL_APP` in the hardware setup.
3. Open the Physical Web app to detect the URL emitted by the beacon node, as shown in the following figure.

Figure 5-77. Physical Web App Detecting Eddystone-URL



4. Click the **Menu** button to open the “Edit URL” option; this requests that the user to put the beacon in Configuration mode. Pressing the SW0 button on the Xplained Pro board for 3 seconds (long press) puts the beacon device in Configuration mode.
5. The URL configuration window will pop-up once the Android device establishes connection with the beacon’s configuration service, as shown in the following figure. Change the URL value to a different one; make sure to use a shortened URL as the size of encoded URL is limited to 17 bytes. Google’s URL shortener can be used for this purpose <https://goo.gl/>.

Figure 5-78. URL Configuration on Physical Web App

5.3.24 Direct Test Mode Application

This demonstration requires two ATSAMB11-MR/ZR devices loaded with Direct Test Mode example application code. Perform the following steps to run the DTM with Performance Analyzer tool.

1. Start the performance analyzer in the Atmel Studio.

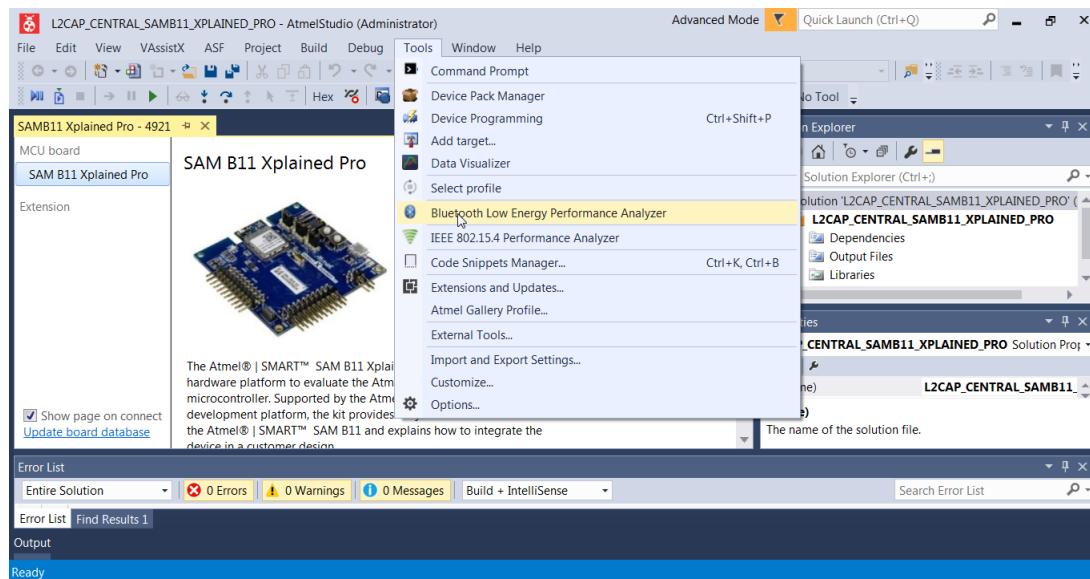
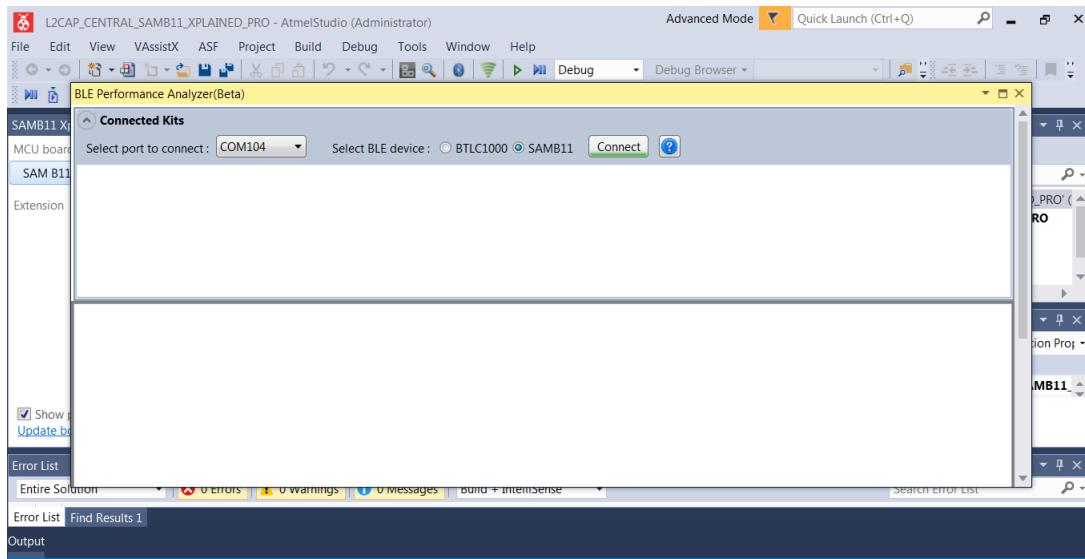
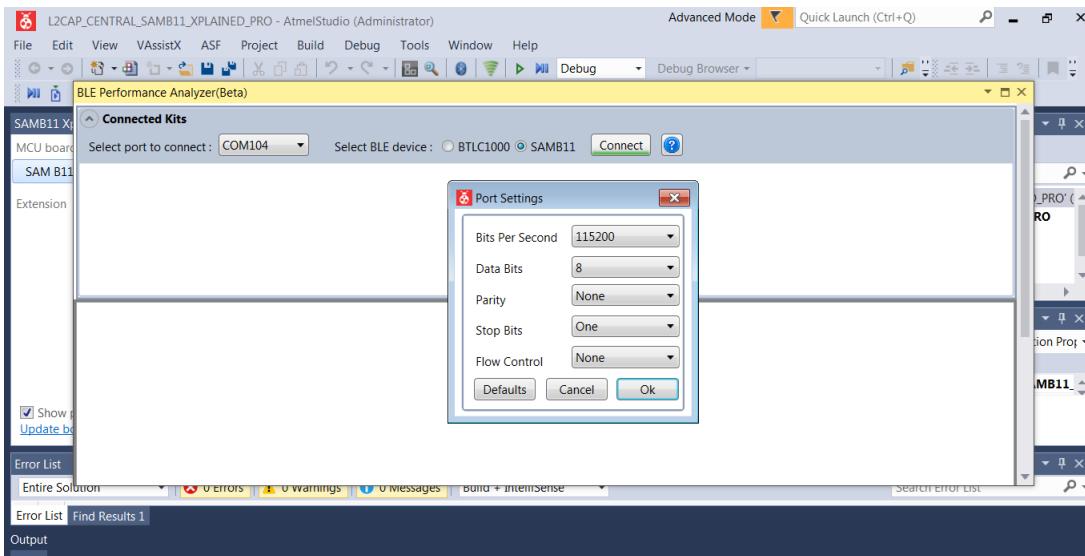
Figure 5-79. Selecting Studio Performance Analyzer Tool

Figure 5-80. BLE Performance Analyzer Tool Window

2. Next, initialize UART. Enter the COM port number and press “Init UART”. A successful initialization is indicated by receiving a chip response, as shown in the following figure.

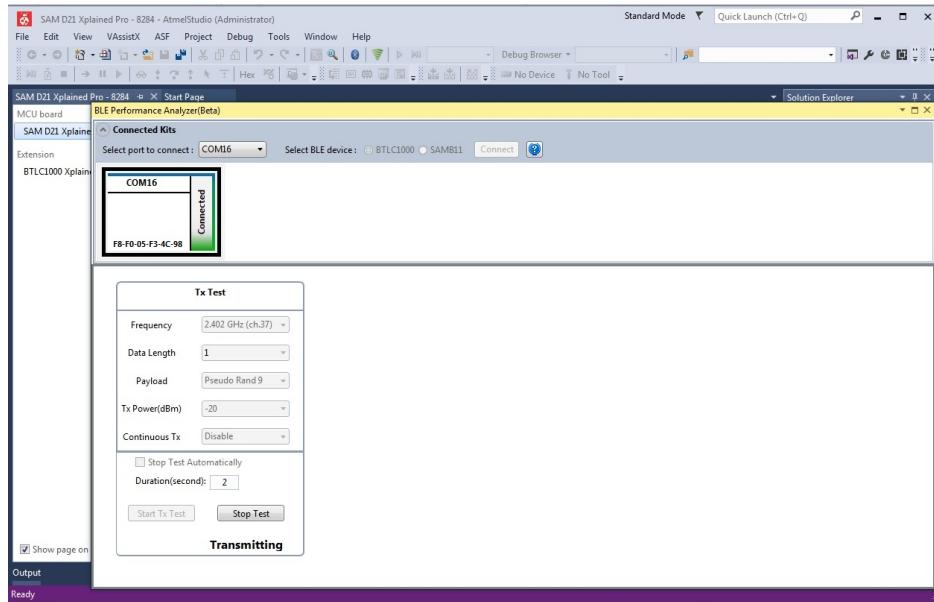


Tip: Check the COM port number from the Device Manager.

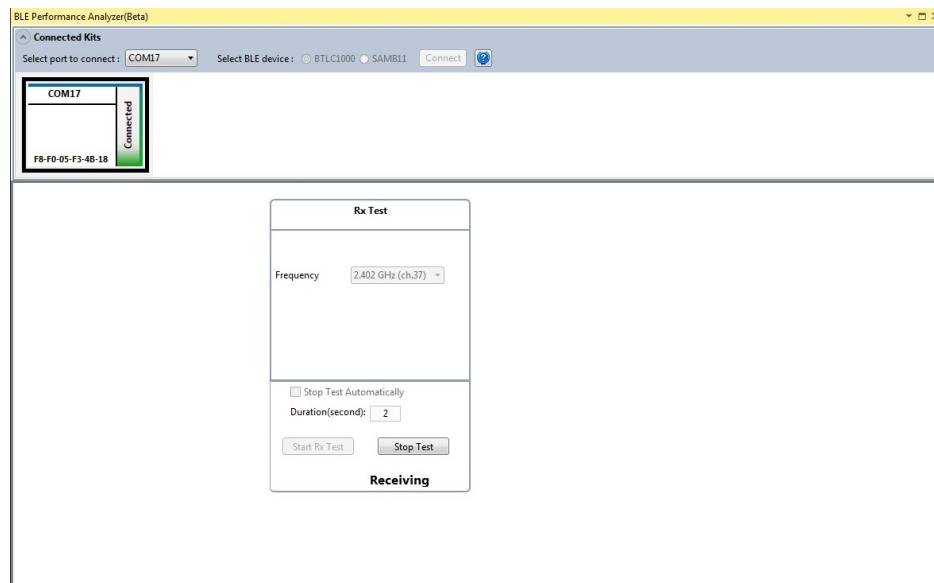
Figure 5-81. Initializing UART

3. Start the Direct Test Mode, configuring one board as Tx and the other one as Rx. Make sure to select the same RF Channel for both Rx and Tx during the test, and start the Rx test before the Tx test so that no packets are missed. The user must see non-zero packets received at the Rx side, indicating successful transmission and reception.

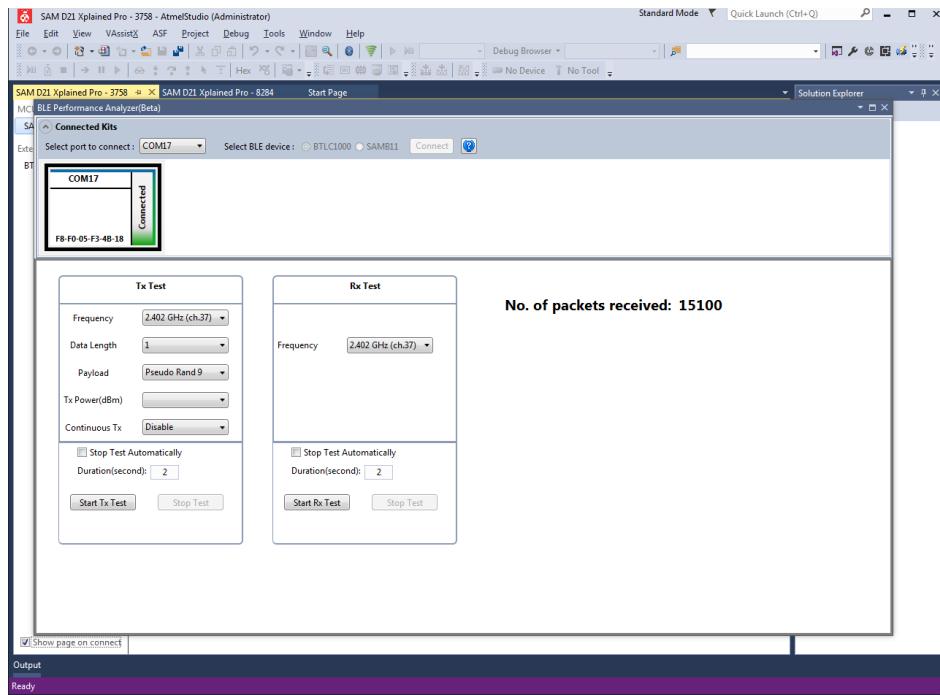
Note: Any side can be replaced by standard compliant test equipment.

Figure 5-82. Starting Tx Test

- On the Rx side, select the appropriate COM port with the same default settings. Open the window with both Tx and Rx options. Click **Start Rx Test** and ensure that the packets are transferred for a certain time period from the Rx device.

Figure 5-83. Starting Rx Test

- Click **Stop Test** to display the number of successful received packets.

Figure 5-84. Number of Packets Received

Important: The PER is calculated assuming that the transmitter side sends 1500 packets for testing using R&S CBT equipment. For peer testing, ignore the PER reading.

5.3.25 AT Command Application

This application supports the following AT commands on the console.

1. Follow Step 1 from [Initializing the Device](#).
2. Open any Terminal Application (for example, TeraTerm). Select the COM port enumerated on the PC and set the following parameters:
 - Baudrate 115200
 - Parity None
 - One Stop bit
 - One Start bit
 - No Hardware Handshake
3. RESET command – to reset the device, type “AT+RESET” and then press the <Enter> key.

Figure 5-85. RESET Command

```
UART Initialized
Device name    : Atmel BLE Device
Device Address : 0xF8F005F35D67, Type : 0

>>BLE Stack Initialized, use "AT+" for AT-Commands

#          CMD           Handler Ptr.
-----
01        RESET         100108b9
02        SCAN          10010917
03        STOP          10010941
04        CFG_DEF       100108cf
-----

AT+RESET
Searching for CMD handler...
Processing...Reset Handler
DONE
```

4. CFG_DEF command – to configure the device, type “AT+CFG_DEF” and then press the <Enter> key.

Figure 5-86. CFG_DEF Command

```
AT+CFG_DEF
Searching for CMD handler...
Processing... Config Handler
DONE
```

5. SCAN command – to scan the BLE device, type “AT+SCAN” and then press the <Enter> key.
After scanning for the device, the scan result is shown in the following figure.

Figure 5-87. SCAN Command

```
AT+SCAN
Searching for CMD handler...
Processing... Scan Handler
DONE
AT_BLE_SCAN_INFO:
    Device Addr.: 0xF8F005F35BE4
    AddrType     : 0x00
    RSSI        : -68
AT_BLE_SCAN_INFO:
    Device Addr.: 0xF8F005F35BE4
    AddrType     : 0x00
    RSSI        : -66
AT_BLE_SCAN_INFO:
    Device Addr.: 0xACEE9E19AE82
    AddrType     : 0x00
    RSSI        : -78
AT_BLE_SCAN_INFO:
    Device Addr.: 0xF8F005F35978
    AddrType     : 0x00
    RSSI        : -77
AT_BLE_SCAN_INFO:
    Device Addr.: 0xF8F005F35978
    AddrType     : 0x00
    RSSI        : -76
```

6. STOP command – to stop the command handler, type “AT+STOP” and then press the <Enter> key.

Figure 5-88. STOP Command

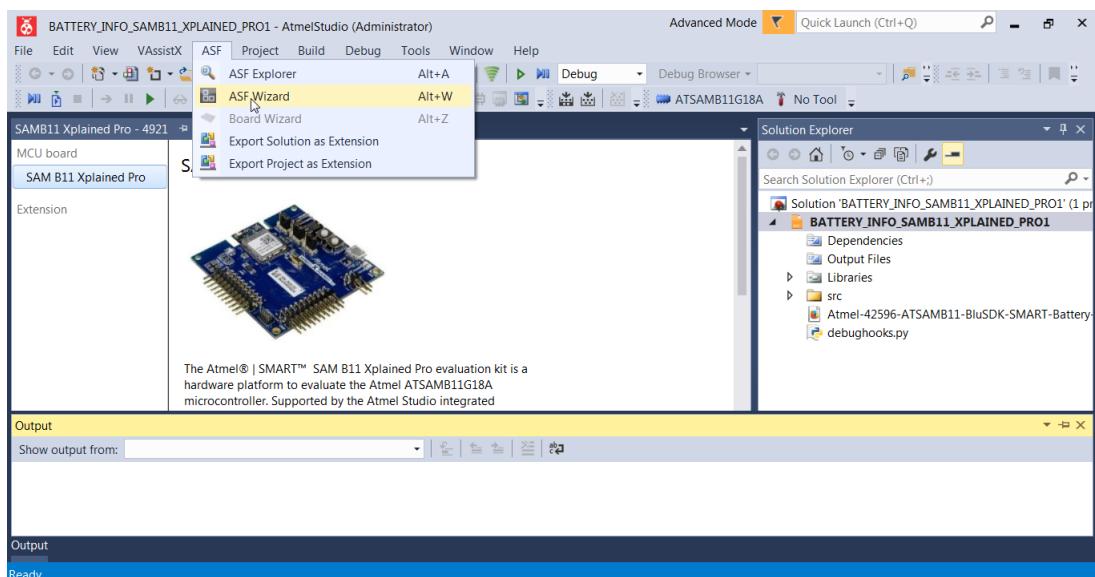
```
AT+STOP
Searching for CMD handler...
Processing... Stop Handler
DONE
```

6. Adding a BLE Standard Service

Another service such as the “Device Information Service” or “Battery Service” can be added to the application by using the ASF wizard as shown in the following steps.

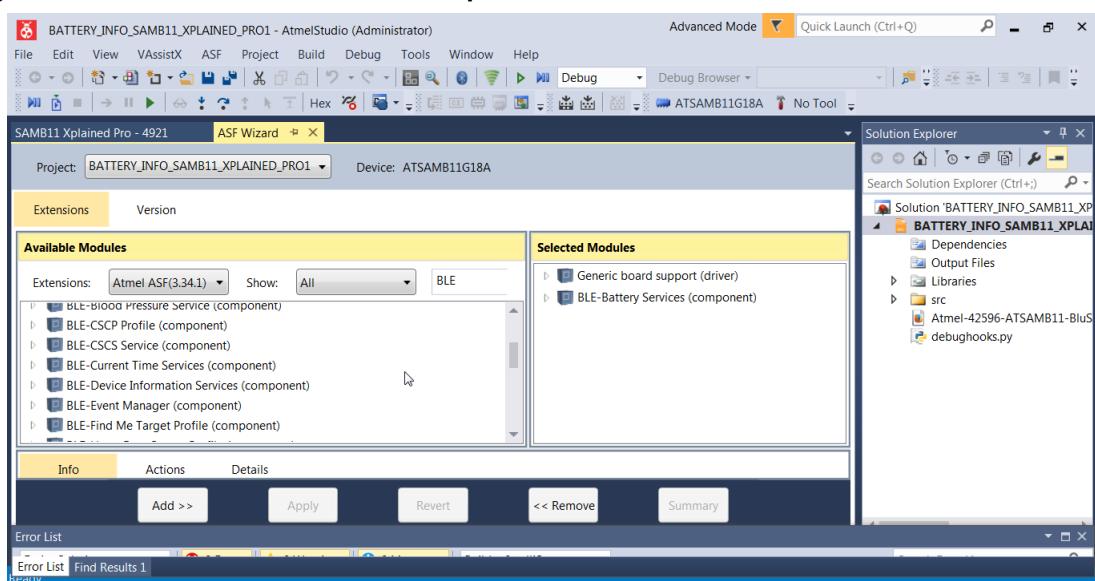
1. Go to the Atmel Studio ASF > ASF Wizard as shown in the following figure.

Figure 6-1. Invoking ASF Wizard



2. In the *ASF Wizard* window, enter “BLE” in the search box, as shown in the following figure.

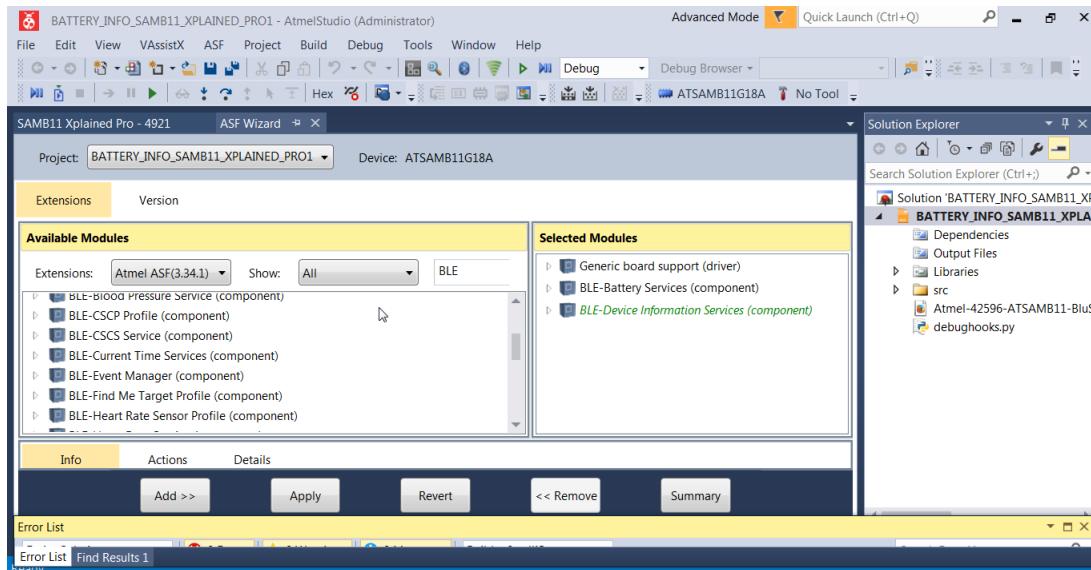
Figure 6-2. ASF BLE Services and Components Window



3. Select the required BLE Services/Profiles Component, as shown in the following figure.

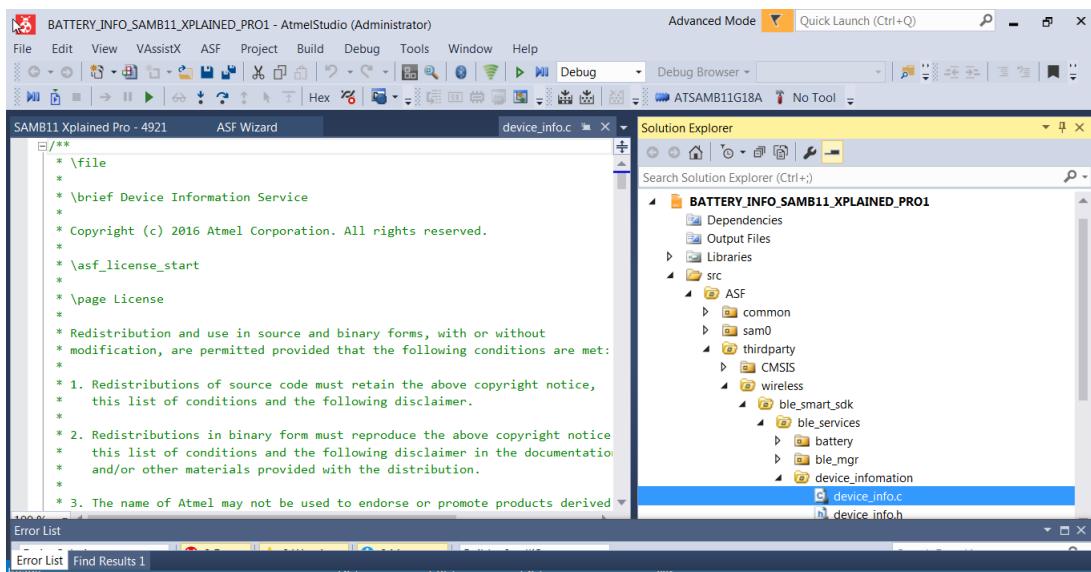
- 3.1. Select **Device Information Services**.
- 3.2. Click **Add > Apply > OK**.

Figure 6-3. Adding BLE - Device Information Service and its Component



- The newly added BLE service component must be available in the following directory: `src\thirdparty\wireless\ble_smart_sdk\ble_services\device_information`, as shown in the following figure.

Figure 6-4. Hierarchy of Newly Added Service Component



- If required in the application use the APIs, as mentioned in the Device Information Service (`device_info.h`), for incorporating this functionality.

```
/**@brief Update the DIS characteristic value after defining the services using
dis_primary_service_define
*
* @param[in] dis_serv dis service instance
* @param[in] info_type dis characteristic type to be updated
* @param[in] info_data data need to be updated
* @return @ref AT_BLE_SUCCESS operation completed successfully.
* @return @ref AT_BLE_FAILURE Generic error.
*/
at_ble_status_t dis_info_update(dis_gatt_service_handler_t *dis_serv , dis_info_type
info_type,
dis_info data* info_data, at_ble_handle_t conn_handle);
/**@brief DIS service and characteristic initialization (Called only once by user).
```

```
*  
* @param[in] device_info_serv dis service instance  
*  
* @return none  
*/  
void dis_init_service(dis_gatt_service_handler_t *device_info_serv );  
/**@brief Register a dis service instance inside stack.  
*  
* @param[in] dis_primary_service dis service instance  
*  
* @return @ref AT_BLE_SUCCESS operation completed successfully  
* @return @ref AT_BLE_FAILURE Generic error.  
*/  
at_ble_status_t dis_primary_service_define(dis_gatt_service_handler_t *dis_primary_service);
```

7. Custom Serial Chat Service Specification

7.1 Service Declaration

The Custom Serial Chat profile consists of a custom serial chat service. Both the mobile app and the host (ATSAMB11-MR/ZR) need to expose this service. The custom serial chat service is instantiated as a primary service.

The UUID value assigned to custom serial chat service is:

`fd5abba0-3935-11e5-85a6-0002a5d5c51b`.

7.2 Service Characteristic

The following characteristics are exposed in the Custom Serial Chat service. Only one instance of each characteristic is permitted within this service.

Table 7-1. Custom Serial Chat Service Characteristics

Characteristic Name	Requirement	Mandatory Properties	Security Permission
Endpoint	M	Notify	Depend on BLE_PAIR_ENABLE macro
Client characteristic configuration descriptor	M	Read, Write	None

Note:

1. The security permission depends on the `BLE_PAIR_ENABLE` macro, defined inside the `ble_manager.h`.
2. If `BLE_PAIR_ENABLE` is set true, then the security permission of the Endpoint characteristic is readable with authentication and writable with authentication.
3. If `BLE_PAIR_ENABLE` is set false, then the security permission of Endpoint characteristic is none.

7.3 Endpoint

The Endpoint characteristic is used to transmit the chat data provided by the user on the terminal (device side) and on the mobile chat screen (mobile side).

The UUID value assigned to Endpoint characteristic is `fd5abba1-3935-11e5-85a6-0002a5d5c51b`.

7.3.1 Characteristic Behavior

When the client characteristic configuration descriptor is configured for the notification by a remote device, the user can send chat text message to the remote device.

Note: The chat text is sent as a notification from the sender (mobile app or ATSAMB11-MR/ZR based device). Hence, the client characteristic configuration descriptor is always configured for notifications (in the Custom Serial Chat service instance on the mobile application and host).

7.4 Characteristic Descriptors

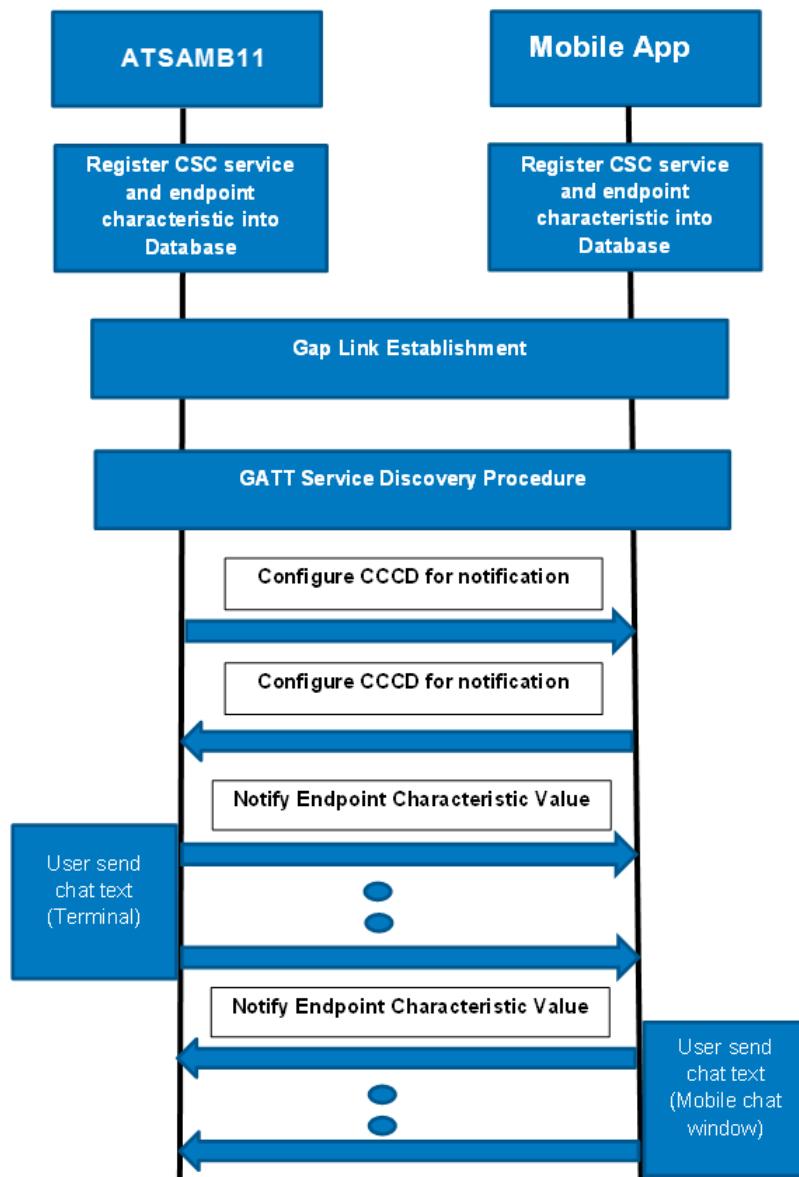
7.4.1 Client Characteristic Configuration Descriptor

The client characteristic configuration descriptor is included in the Endpoint characteristic.

7.5 Sequence Flow Diagram

The following figure illustrates the sequence flow diagram of the Custom Serial Chat profile.

Figure 7-1. Sequence Flow Diagram

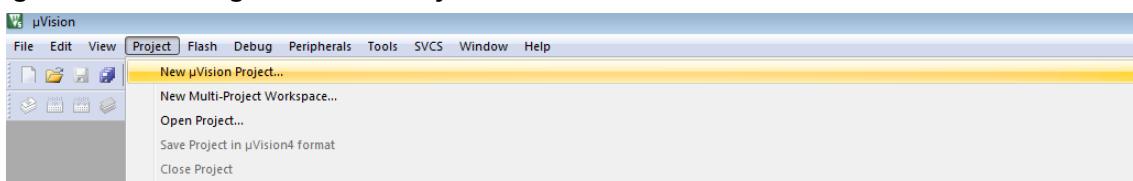


8. Customization of Keil Project

This section provides the step-by-step procedure to setup a Keil project from scratch to run any application on the SAMB11-MR/ZR XPro board.

- To create a new Keil Project, go to Project, select the **Project Name** (for example, iBeacon_SAMB11), and then click **Save**.

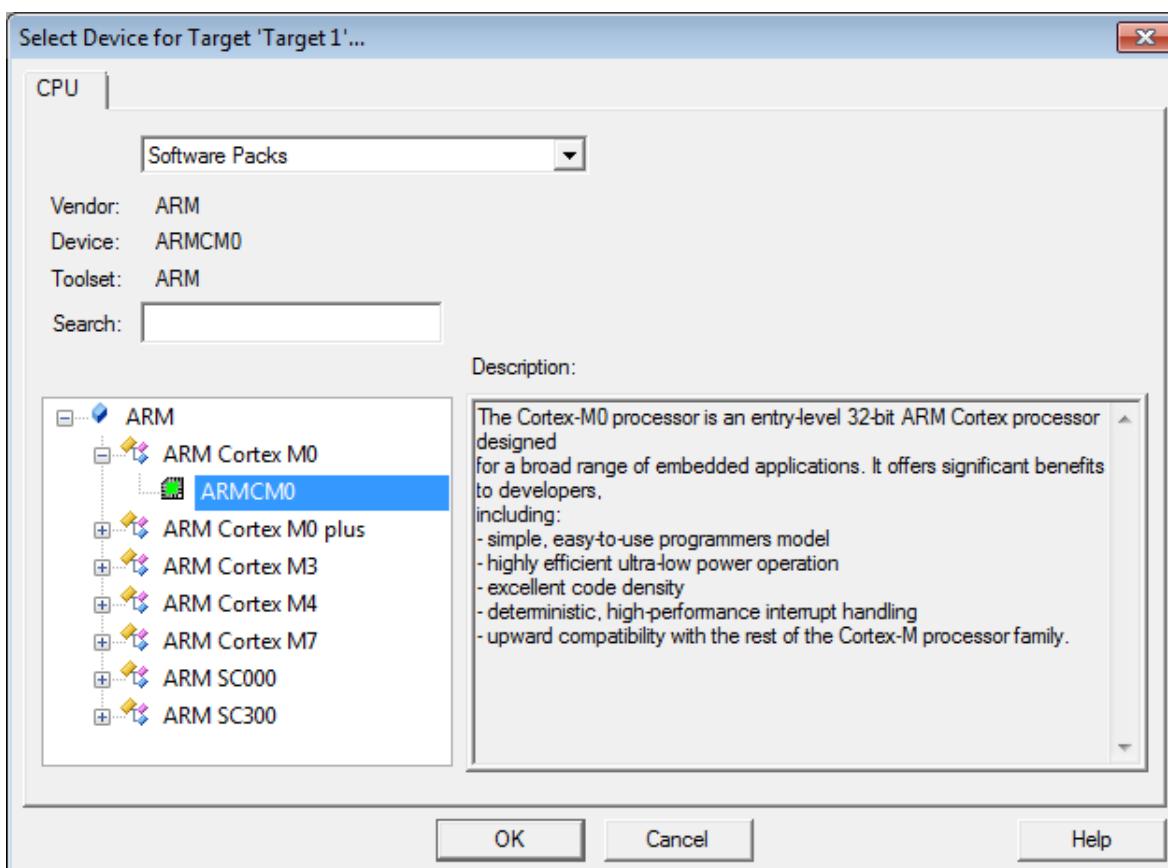
Figure 8-1. Creating a New Keil Project



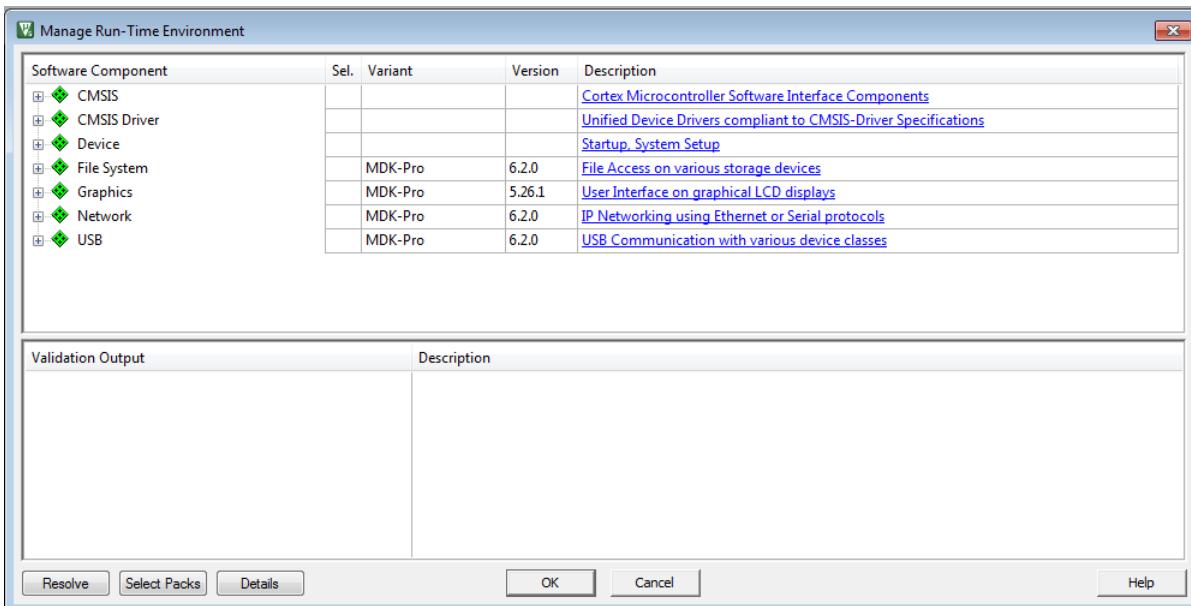
- The Select Device for Target window is displayed, as shown in the following figure.

- Select **ARMCM0** Device for Target.
- Click **OK**.

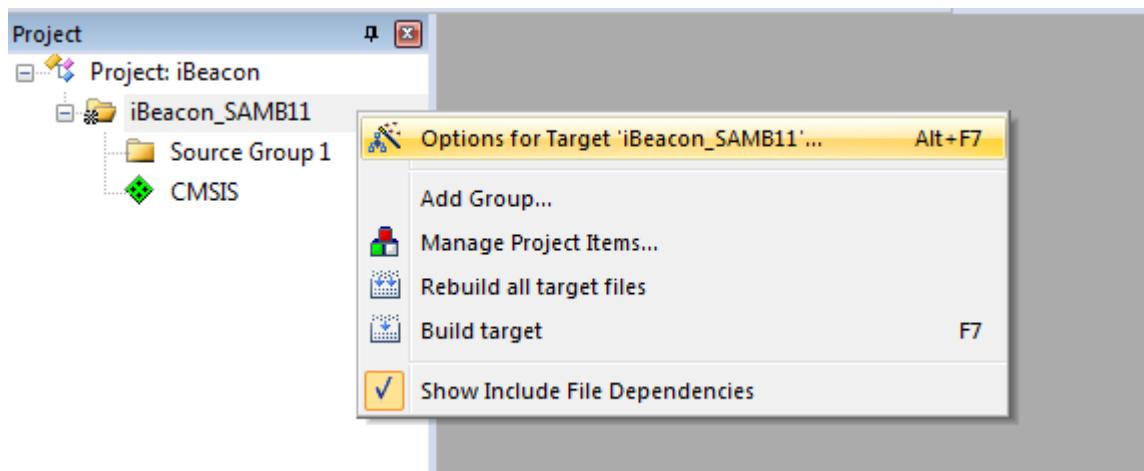
Figure 8-2. Select Device for Target



- A dialog box is displayed with a list of Software Components. Do not select any software components for the target device. Click **OK**.

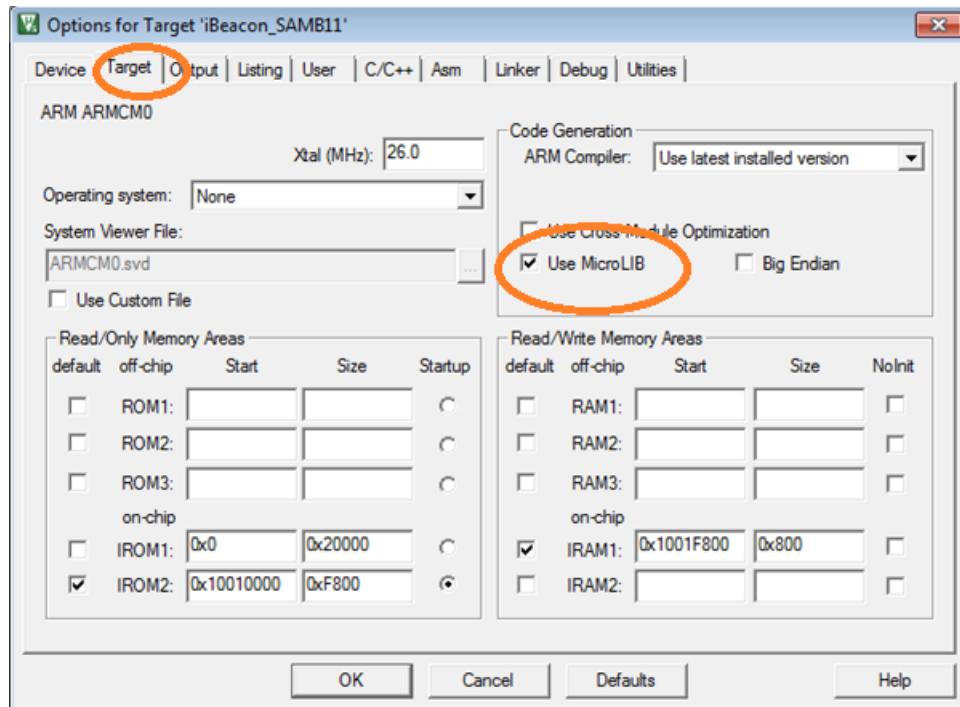
Figure 8-3. Software Component option selection

4. After creating the Project, right click on the **Project** to open the setting page. Select **Options for Target**, as shown in the following figure.

Figure 8-4. Options for Target

- 4.1. In the Options for Target window, select the **Target** tab and set the following:

- Select “Use MicroLIB” check box
- For this sample project, a customized scatter file is used. Using a similar scatter file instead of the memory map from this page is recommended.

Figure 8-5. Target Settings4.2. In the **Output** tab, set the following:

- Provide a name to the executable the application. In this demo “iBeacon_SAMB11” is used as a sample project.
- Be sure to check the “Create HEX File” check box.

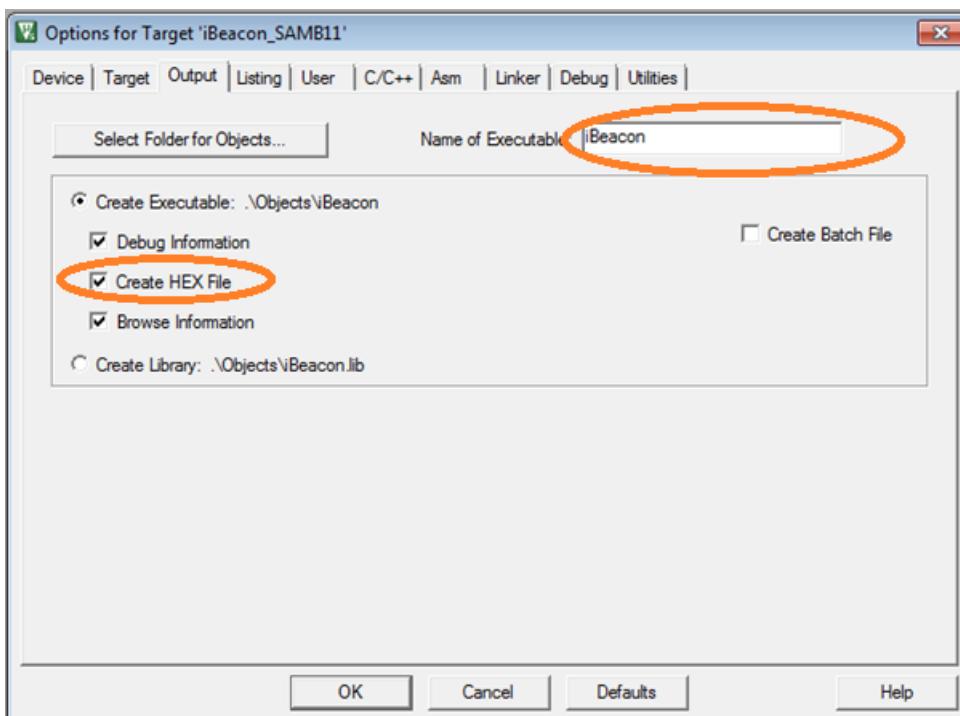
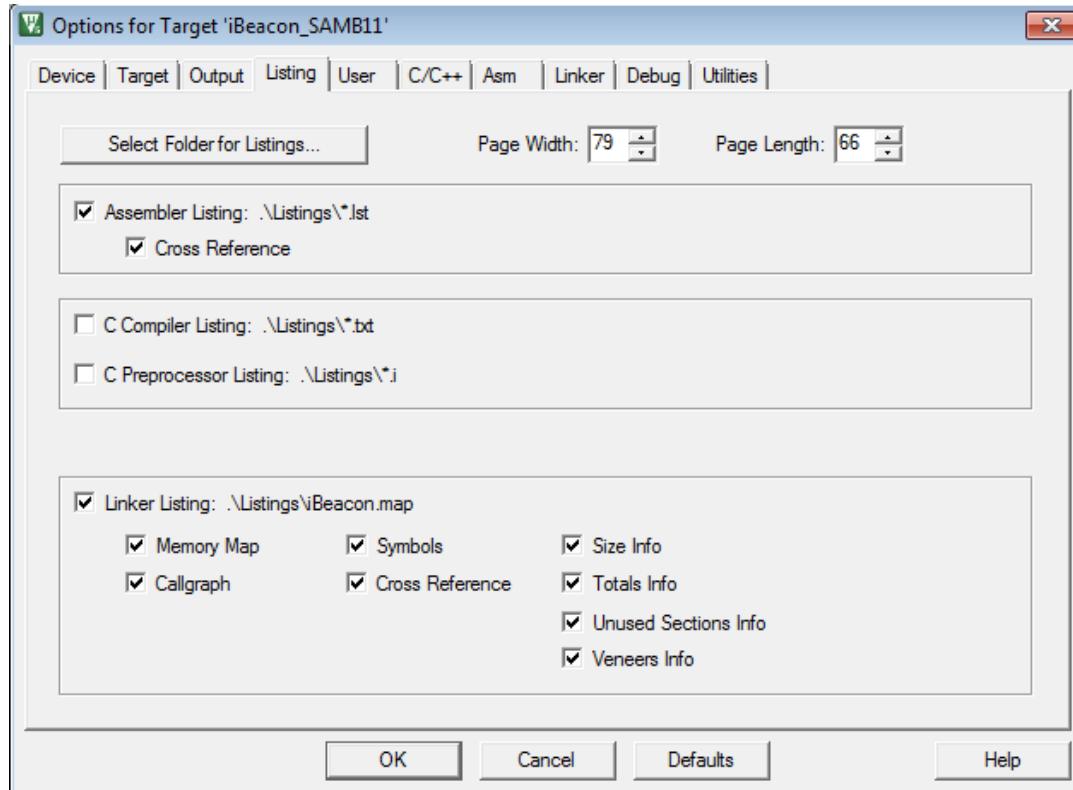
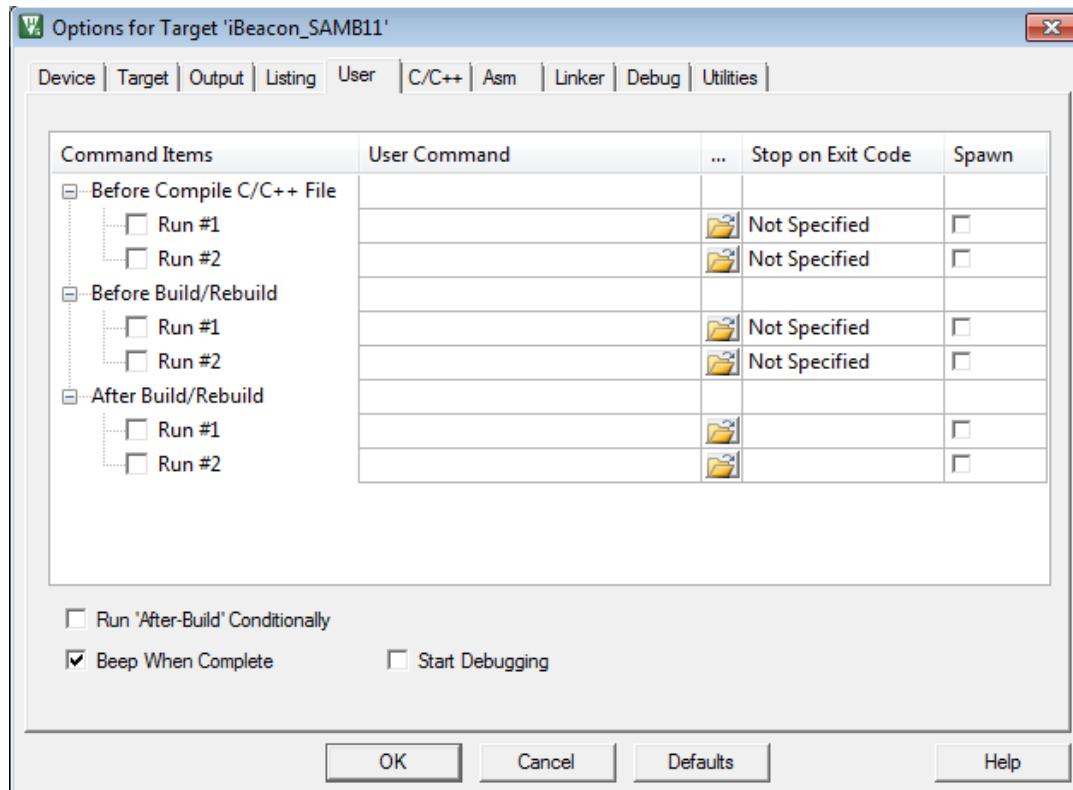
Figure 8-6. Output Settings4.3. In the **Listing** tab, set the parameters (default options), as shown in the following figure.

Figure 8-7. Listing Settings

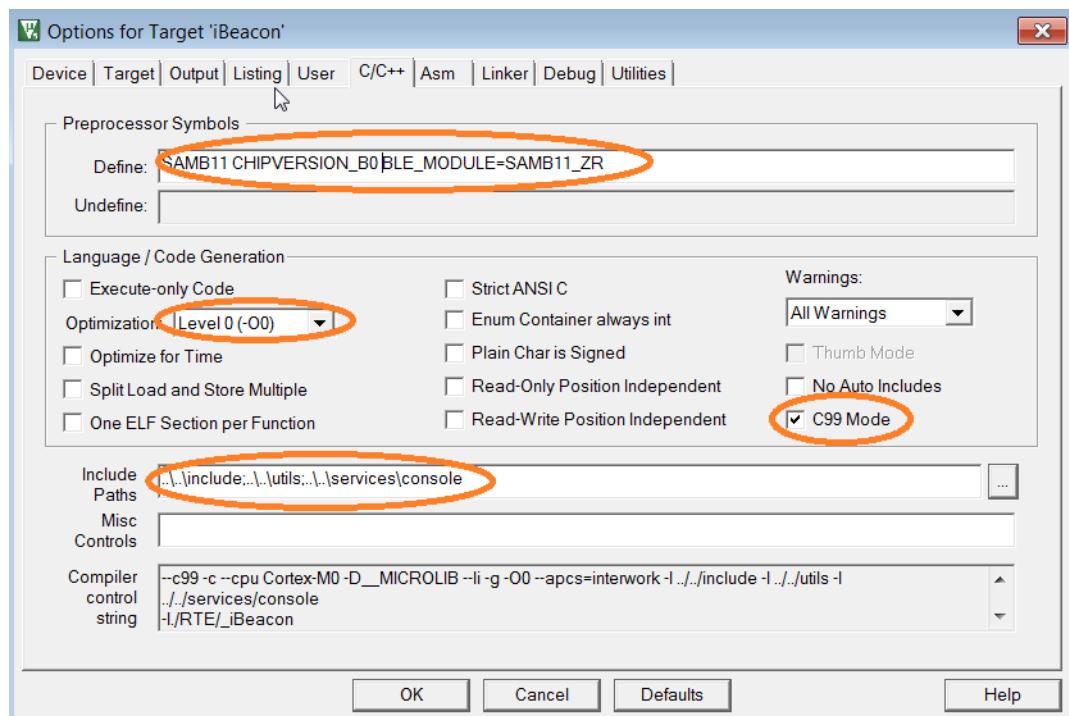
- 4.4. In the **User** tab, set the user settings (default selection), as shown in the following figure.

Figure 8-8. User Settings

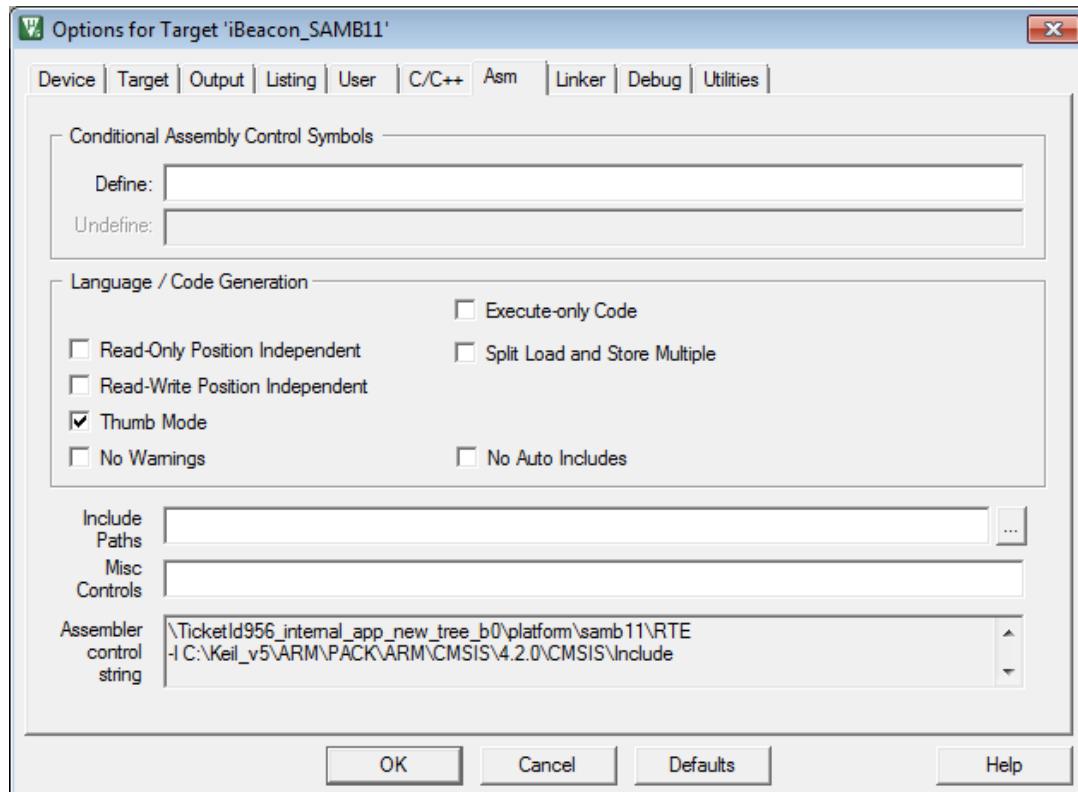
4.5. In the **C/C++** tab, set the following:

- Define the following preprocessor macros, as shown in the following figure.
 - SAMB11
 - CHIPVERSION_B0
 - For SAMB11-MR module define as, BLE_MODULE=SAMB11_MR
 - For SAMB11-ZR module defines as, BLE_MODULE=SAMB11_ZR
 - Select the preferred **Optimization** level from the drop-down list
 - Select **C99 Mode** check box
 - In the **Include paths**, enter “..\\..\\include;”, “..\\..\\utils;”and “..\\..\\services\\console”.
- Note:**
1. These paths contain the required header files for ble_api library and ATSAMB11 drivers.
 2. Make sure to correctly refer to the included folder in the release package.

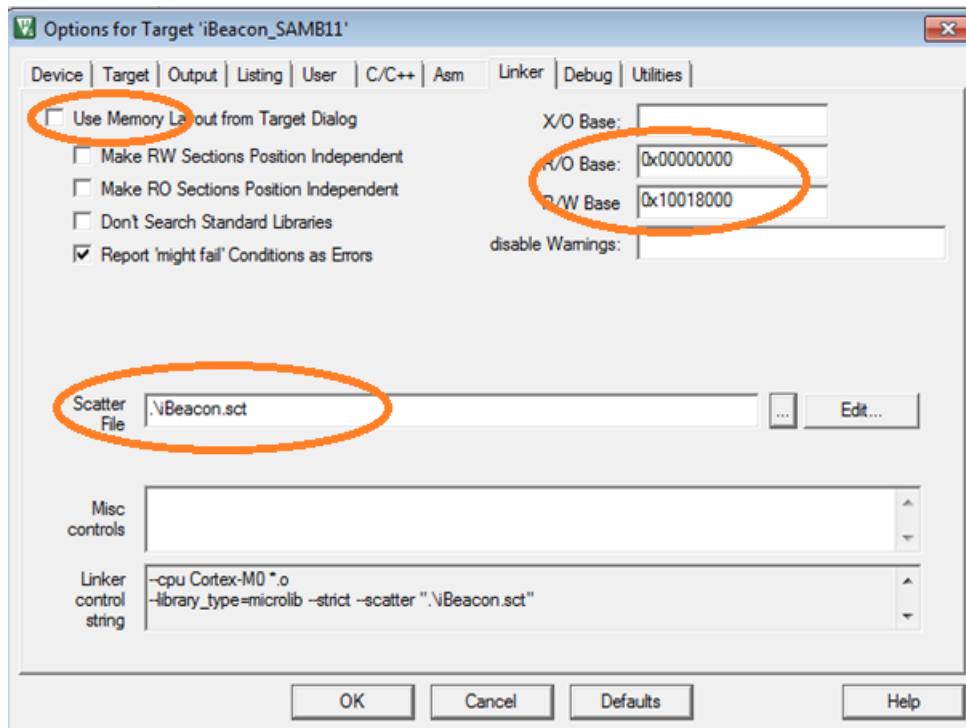
Figure 8-9. C/C++ Settings



4.6. In the **ASM** tab, check the “Thumb Mode” check box, as shown in the following figure.

Figure 8-10. ASM Settings

- 4.7. In the **Linker** tab, configure the following parameters:
- Uncheck **Use Memory Layout from Target Dialog**
 - In **R/O Base** enter **0x00000000**
 - In **R/W Base** enter **0x10018000**
 - Browse to add the manual scatter file “*iBeacon.sct*” or “*iBeacon_samb11.sct*” or “*app_manual.sct*” in scatter file path
- Note:** For more details on sample scatter file, refer to [Sample Keil Scatter File](#).

Figure 8-11. Linker Settings

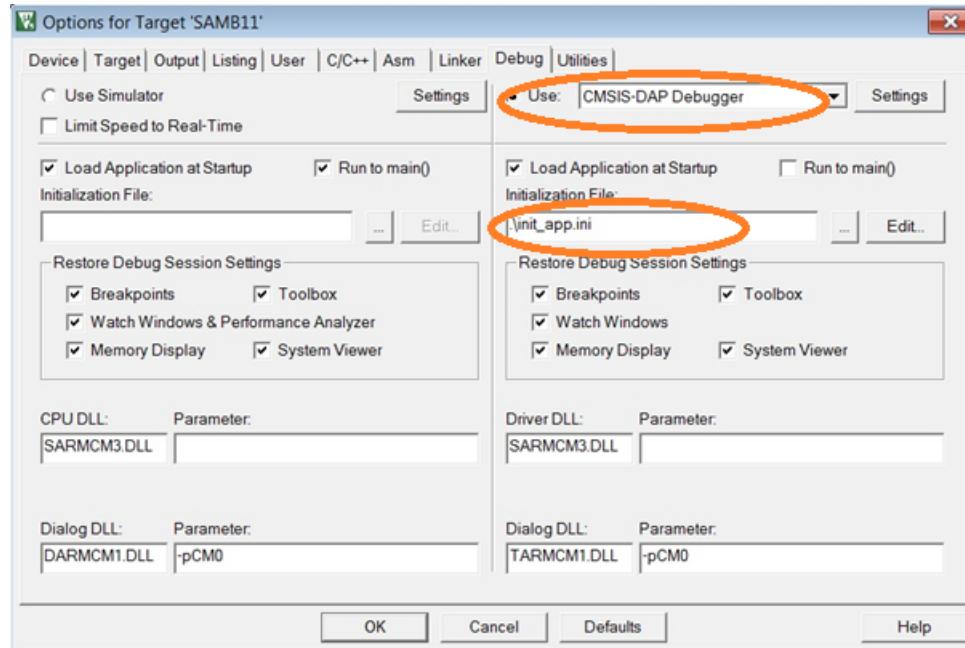
4.8. In the **Debug** tab, configure the following:

- Select **CMSIS-DAP Debugger**
- In the Initialization File edit box, enter “init_app.ini”.

Note:

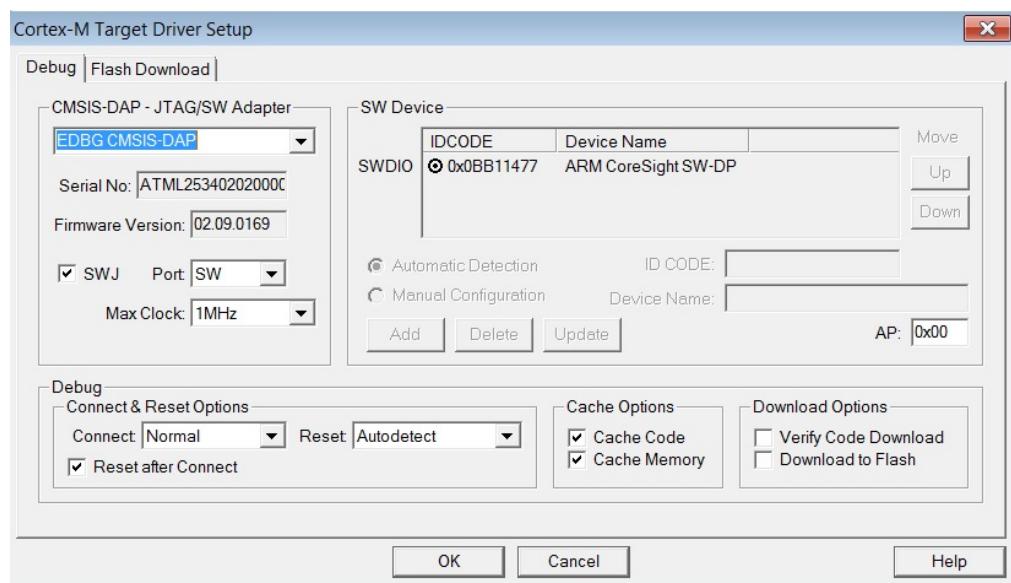
1. This is the initialization file for linking the application and patches to the bootrom code.
2. This initialization file is available in the sample application directory (<release_dir>\ apps\xxxx).

Figure 8-12. Debug Settings



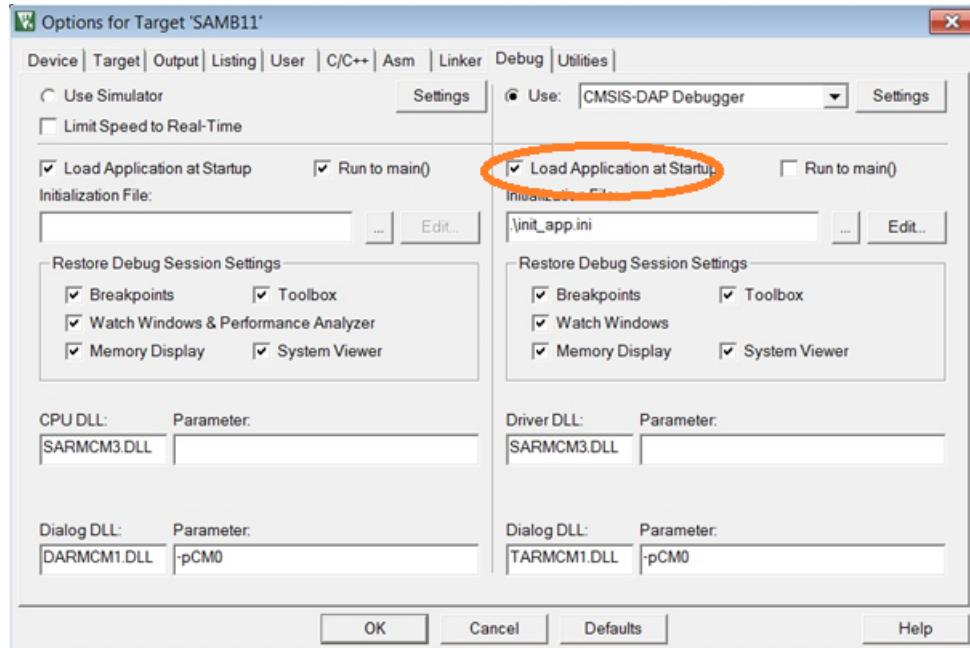
- Connect the SAMB11-MR/ZR XPro to PC and select the “Settings” button from **CMSIS-DAP Debugger**.
- In Port, select **SW** from the drop-down and then click **OK**.

Figure 8-13. SW Interface Selection for Debug Port



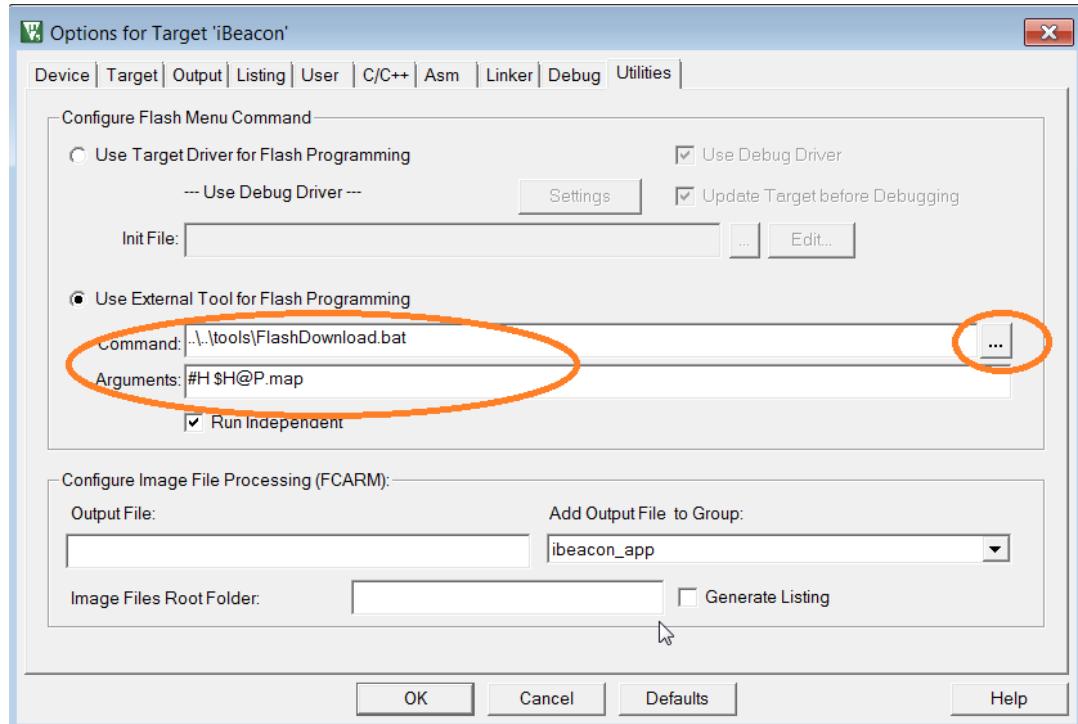
- Check the “Load Application at Startup” check box, as shown in the following figure.

Figure 8-14. Debug Selection

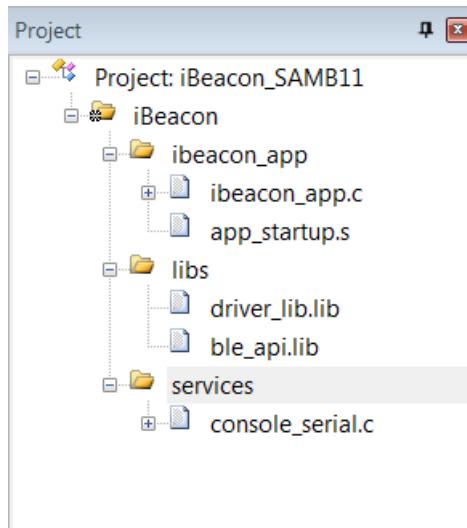


4.9. In the **Utilities** tab, set the following:

- Enable **Use External Tool for the Flash Programming** option
 - In the Command section, browse and select the FlashDownload.bat provided in the <release_dir>\tools\ directory.
 - In the Argument edit box, enter #H \$H@P.map. This is used to pass the application map file and the application hex file in to the image creator tool.
 - Tools directory has download.py (a Python® file) and other tools that create the out.img and download it to the SPI flash available on the SAMB11-MR/ZR.
- Note:** This tool uses Python. Be sure to install Python 2.7.3 or later.
- Click **OK**.

Figure 8-15. Utilities Setting

- Add the following source files and library to the Project, as shown in the following figure. In this example, beacon app is used.
 - Sample **ibeacon_app ibeacon_app.c**
 - Provided sample **app_startup.s**
 - console_serial.c** ... located in <release_dir>\services\console\b11
 - ble_api.lib** ... located in <release_dir>\lib
 - driver_lib.lib** ... located in <release_dir>\lib

Figure 8-16. Adding Files to Project

- After adding the files in to the project, build the solution by pressing <F7> or go to Project > Build Target.

7. To start the Debug session, perform the following:

- Before starting the Debug session, erase the SPI Flash in the ATSAMB11-MR/ZR:
 - Open the file "<release_dir>\tools\EraseFlash.bat" to run. An erase procedure starts.
 - After the erase is completed, the following output message is displayed.

Figure 8-17. Output Message

```
Converting 4109 to generic DAP error
Chip ID is 0x2000b0
Erasing Flash ...
Erase Complete ...
O.K.
```

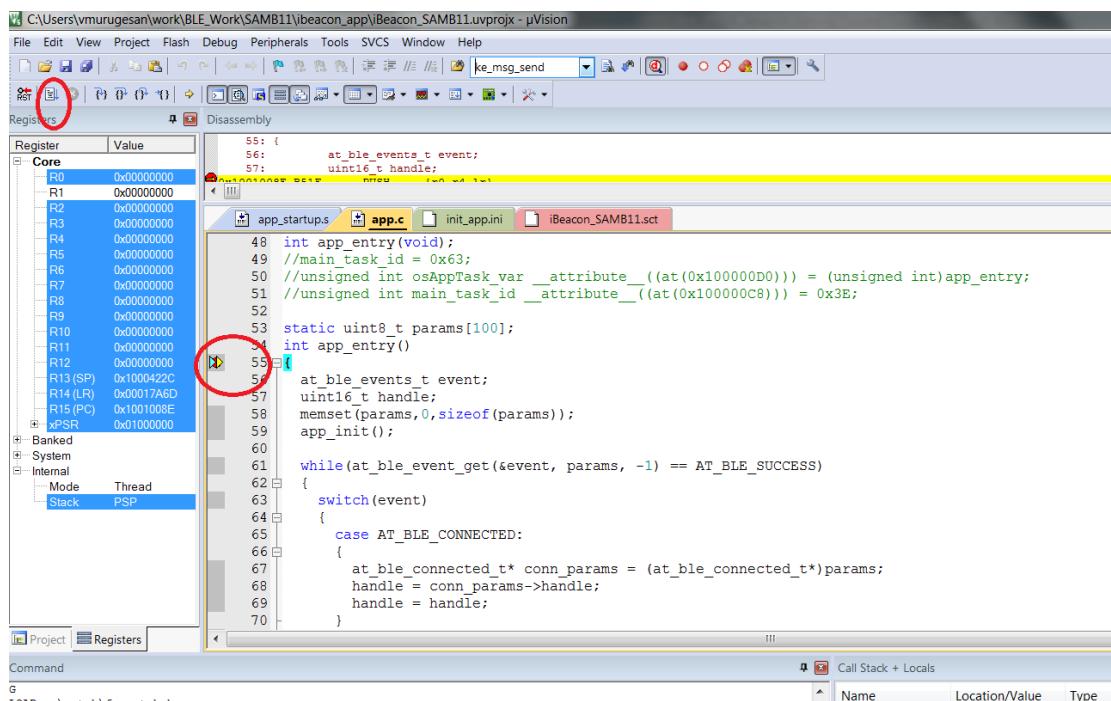
- After compiling, click the button shown in the following figure to start the debug session.

Figure 8-18. Starting Debug Session



- To start the Code execution, perform the following:
 - After loading the sample code in to the RAM location, the program stops at the entry function of the application, as shown in the following figure.
 - Next, click **RUN** to start the application.

Figure 8-19. Running the Application



- Now the demo application is running on the ATSAMB11-MR/ZR XPro board.

9. Memory Map

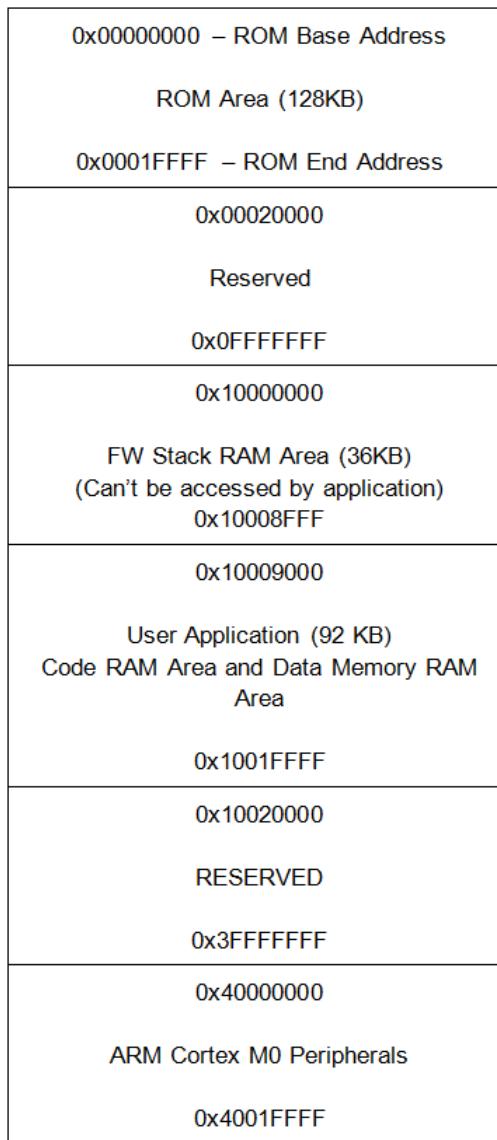
The ATSAMB11-MR/ZR has 128 KB ROM and 128 KB RAM.

Figure 9-1. ATSAMB11-MR/ZR Memory Map



9.1 RAM Partition

The following figure illustrates the ATSAMB11-MR/ZR RAM partition.

Figure 9-2. ATSAMB11-MR/ZR RAM Partition

9.2 Sample GCC Linker File

The following is a sample GCC Linker file.

ram - This defines the user application section as defined in Figure 9.2. The user application code and data resides on the same memory section. The ORIGIN and LENGTH of this section is limited as per Figure 9-2.

Iram - This section is not applicable for SAMB11. The "Iram" section can be commented and the LENGTH defined in the "Iram" section can add into LENGTH of existing "ram" section.

```

OUTPUT_FORMAT("elf32-littlearm", "elf32-littlearm", "elf32-littlearm")
OUTPUT_ARCH(arm)
SEARCH_DIR(..)

/* Memory Spaces Definitions */
MEMORY
{

```

```

    ram    (rwx) : ORIGIN = 0x10009000, LENGTH = 0x00016800
    lram   (rwx) : ORIGIN = 0x1001F800, LENGTH = 0x00000800
}

/* The stack size used by the application. NOTE: you need to adjust according to your
application. */
STACK_SIZE = DEFINED(STACK_SIZE) ? STACK_SIZE : DEFINED(__stack_size__) ? __stack_size__ :
0x800;

/* Section Definitions */
SECTIONS
{
    .text :
    {
        . = ALIGN(4);
        _sfixed = .;
        KEEP(*(.text.main .text.app_entry))
        *(.text .text.* .gnu.linkonce.t.*)
        *(.glue_7t) *(.glue_7)
        *(.rodata .rodata*.gnu.linkonce.r.*)
        *(.ARM.extab*.gnu.linkonce.armextab.*)

        /* Support C constructors, and C destructors in both user code
           and the C library. This also provides support for C++ code. */
        . = ALIGN(4);
        KEEP(*(.init))
        . = ALIGN(4);
        __preinit_array_start = .;
        KEEP (*(.preinit_array))
        __preinit_array_end = .;

        . = ALIGN(4);
        __init_array_start = .;
        KEEP (*(SORT(.init_array.*)))
        KEEP (*(.init_array))
        __init_array_end = .;

        . = ALIGN(4);
        KEEP (*crtbegin.o(.ctors))
        KEEP (*(EXCLUDE_FILE (*crtend.o) .ctors))
        KEEP (*(SORT(.ctors.*)))
        KEEP (*crtend.o(.ctors))

        . = ALIGN(4);
        KEEP(*(.fini))

        . = ALIGN(4);
        __fini_array_start = .;
        KEEP (*(.fini_array))
        KEEP (*(SORT(.fini_array.*)))
        __fini_array_end = .;

        KEEP (*crtbegin.o(.dtors))
        KEEP (*(EXCLUDE_FILE (*crtend.o) .dtors))
        KEEP (*(SORT(.dtors.*)))
        KEEP (*crtend.o(.dtors))

        . = ALIGN(4);
        _efixed = .;           /* End of text section */
    } > ram

    . = ALIGN(4);
    _etext = .;

    .relocate : AT (_etext)
    {
        . = ALIGN(4);
        _srelocate = .;
        *(.ramfunc .ramfunc.*);
        *(.data .data.*);
        . = ALIGN(4);
        _erelocate = .;
    } > ram

    /* .bss section which is used for uninitialized data */
    .bss (NOLOAD) :
    {
        . = ALIGN(4);

```

```

_sbss = . ;
_szero = . ;
*(.bss .bss.*)
*(COMMON)
. = ALIGN(4);
_ebss = . ;
_ezero = . ;
} > ram

/* stack section */
.stack (NOLOAD):
{
    . = ALIGN(8);
    _sstack = .;
    . = . + STACK_SIZE;
    . = ALIGN(8);
    _estack = .;
} > ram

/* .ARM.exidx is sorted, so has to go in its own output section. */
PROVIDE_HIDDEN (__exidx_start = .);
.ARM.exidx :
{
    *(.ARM.exidx* .gnu.linkonce.armexidx.*)
} > ram
PROVIDE_HIDDEN (__exidx_end = .);

. = ALIGN(4);
_end = . ;
}

```

9.3 Sample Keil Scatter File

The following is the sample scatter file for Keil. The application code in `app.c` is loaded into `LR_IRAM2` region.

`LR_IRAM2` - This defines the user application section as defined in Figure 9-2. The user application code and data resides on the same memory section. The start address and size of this section is limited as per Figure 9-2.

```

LR_IRAM2 0x10009000 0x00017000 {
    RW_IRAM2 0x10009000 0x00017000 { ; load address = execution address
        app.o(+RO +RW +ZI)
        .ANY (+RO +RW +ZI)
    }
}

```

If the application is created in some other file name, be sure to edit the `app.c` file with correct object files. The application file name must be updated in case of a different file name.

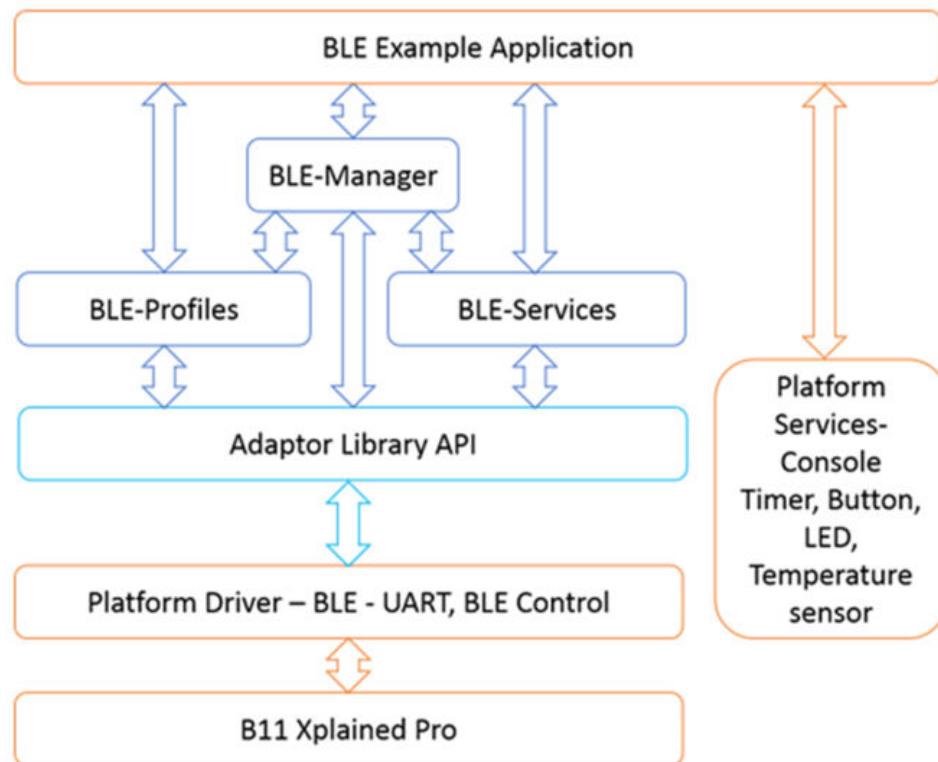
Note: The `app.c` must contain the `int app_entry(void)` function and also must have

```
#define APP_STACK_SIZE      (2048)
volatile unsigned char     app_stack_patch[APP_STACK_SIZE];
```

10. BluSDK SMART Software Architecture

The following diagram illustrates the various layers in the BluSDK SMART Architecture for implementing various applications.

Figure 10-1. BluSDK SMART Software Architecture



11. Document Revision History

Rev A - 09/2017

Section	Changes
Document	Initial Release

The Microchip Web Site

Microchip provides online support via our web site at <http://www.microchip.com/>. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site 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 (FAQ), 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, listings of Microchip sales offices, distributors and factory representatives

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 web site at <http://www.microchip.com/>. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

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 web site at: <http://www.microchip.com/support>

Microchip Devices Code Protection Feature

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.

Legal Notice

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, Heldo, JukeBlox, KeeLoq, KeeLoq logo, Kleer, LANCheck, LINK MD, maXStylus, 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.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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

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, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-5224-2137-5

Quality Management System Certified by DNV

ISO/TS 16949

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



Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/ support Web Address: www.microchip.com	Asia Pacific Office Suites 3707-14, 37th Floor Tower 6, The Gateway Harbour City, Kowloon Hong Kong Tel: 852-2943-5100 Fax: 852-2401-3431 Australia - Sydney Tel: 61-2-9868-6733 Fax: 61-2-9868-6755 China - Beijing Tel: 86-10-8569-7000 Fax: 86-10-8528-2104 China - Chengdu Tel: 86-28-8665-5511 Fax: 86-28-8665-7889 China - Chongqing Tel: 86-23-8980-9588 Fax: 86-23-8980-9500 China - Dongguan Tel: 86-769-8702-9880 China - Guangzhou Tel: 86-20-8755-8029 China - Hangzhou Tel: 86-571-8792-8115 Fax: 86-571-8792-8116 China - Hong Kong SAR Tel: 852-2943-5100 Fax: 852-2401-3431 China - Nanjing Tel: 86-25-8473-2460 Fax: 86-25-8473-2470 China - Qingdao Tel: 86-532-8502-7355 Fax: 86-532-8502-7205 China - Shanghai Tel: 86-21-3326-8000 Fax: 86-21-3326-8021 China - Shenyang Tel: 86-24-2334-2829 Fax: 86-24-2334-2393 China - Shenzhen Tel: 86-755-8864-2200 Fax: 86-755-8203-1760 China - Wuhan Tel: 86-27-5980-5300 Fax: 86-27-5980-5118 China - Xian Tel: 86-29-8833-7252 Fax: 86-29-8833-7256	China - Xiamen Tel: 86-592-2388138 Fax: 86-592-2388130 China - Zhuhai Tel: 86-756-3210040 Fax: 86-756-3210049 India - Bangalore Tel: 91-80-3090-4444 Fax: 91-80-3090-4123 India - New Delhi Tel: 91-11-4160-8631 Fax: 91-11-4160-8632 India - Pune Tel: 91-20-3019-1500 Japan - Osaka Tel: 81-6-6152-7160 Fax: 81-6-6152-9310 Japan - Tokyo Tel: 81-3-6880- 3770 Fax: 81-3-6880-3771 Korea - Daegu Tel: 82-53-744-4301 Fax: 82-53-744-4302 Korea - Seoul Tel: 82-2-554-7200 Fax: 82-2-558-5932 or 82-2-558-5934 Malaysia - Kuala Lumpur Tel: 60-3-6201-9857 Fax: 60-3-6201-9859 Malaysia - Penang Tel: 60-4-227-8870 Fax: 60-4-227-4068 Philippines - Manila Tel: 63-2-634-9065 Fax: 63-2-634-9069 Singapore Tel: 65-6334-8870 Fax: 65-6334-8850 Taiwan - Hsin Chu Tel: 886-3-5778-366 Fax: 886-3-5770-955 Taiwan - Kaohsiung Tel: 886-7-213-7830 Taiwan - Taipei Tel: 886-2-2508-8600 Fax: 886-2-2508-0102 Thailand - Bangkok Tel: 66-2-694-1351 Fax: 66-2-694-1350	Austria - Wels Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 Denmark - Copenhagen Tel: 45-4450-2828 Fax: 45-4485-2829 Finland - Espoo Tel: 358-9-4520-820 France - Paris Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 France - Saint Cloud Tel: 33-1-30-60-70-00 Germany - Garching Tel: 49-8931-9700 Germany - Haan Tel: 49-2129-3766400 Germany - Heilbronn Tel: 49-7131-67-3636 Germany - Karlsruhe Tel: 49-721-625370 Germany - Munich Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 Germany - Rosenheim Tel: 49-8031-354-560 Israel - Ra'anana Tel: 972-9-744-7705 Italy - Milan Tel: 39-0331-742611 Fax: 39-0331-466781 Italy - Padova Tel: 39-049-7625286 Netherlands - Drunen Tel: 31-416-690399 Fax: 31-416-690340 Norway - Trondheim Tel: 47-7289-7561 Poland - Warsaw Tel: 48-22-3325737 Romania - Bucharest Tel: 40-21-407-87-50 Spain - Madrid Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 Sweden - Gothenberg Tel: 46-31-704-60-40 Sweden - Stockholm Tel: 46-8-5090-4654 UK - Wokingham Tel: 44-118-921-5800 Fax: 44-118-921-5820