iBEACON SAMPLE APPS

BL600

Application Note *v1.2*

INTRODUCTION

The goal of this document includes the following:

- Explain how to setup the BL600 to function as an iBeacon.
- Detail how to test the BL600 for iBeacon functionality.

OVERVIEW

Bluetooth Low Energy (BLE) beacons are becoming ever more popular since Apple introduced its iBeacon specification but there is far more to BLE beacons than just Apples iBeacon. A beacon in BLE terms is simply a device that broadcasts a BLE advert that can provide location context for mobile device applications. iBeacons are currently the most well-known form of BLE beacon and are the subject of this document. At the end of the day, an iBeacon is just a BLE advert in a particular format that complies with the existing BT4.0 specification. Therefore all iBeacons are BLE beacons (adverts) but not all BLE beacons (adverts) are iBeacons. Other beacon formats include AltBeacon by Radius networks, URIBeacon from the Physical Web project, and Eddystone from Google. Anybody considering beacons in their application are advised to consider other beacon types alongside Apple's iBeacon. In particular, Eddystone deserves special attention as it adds considerable functionality and flexibility to a beacon application.

iBeacons consists of a BLE-enabled device that sends out an Apple, Inc. Manufacturer Specific Advertising Record in a BLE advert (AMSAD), formatted specifically to be interpreted by a scanning BLE central device.

The Bluetooth specification requires that the AMSAD consists of four or more octets. The first octet is the overall length, the second octet is always 0xFF, the third and fourth octets are the manufacturer assigned Company ID for Apple Inc., and the rest are the iBeacon data. Apple has specified the rest of the octets to be grouped as: [Datatype] [Length] [UUID] [Major] [Minor] [Calibration Power], where the UUID is 16 octets long, Major and Minor are two octets long, and the rest are single octets as shown in Figure 1:

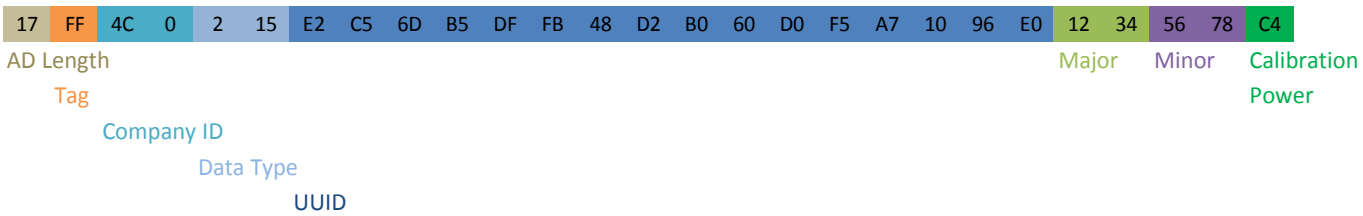


Figure 1: AMSAD graphical representation

The $autorun$.iBeacon smartBASIC sample app allows you to remotely configure the following parameters:

- UUID
- Major
- Minor
- Calibration Power
- Advertising Interval
- Advertising Timeout
- Time to remain connectable
- Actual Transmit Power Value

When not in a connection, the BL600 running $autorun$.beacon advertises with the AMSAD.

REQUIREMENTS

The following equipment and utilities are required:

- BL600 Devkit (firmware 1.2.54.0 or newer)
- Windows PC
- UwTerminal (Laird's terminal emulation utility)
- Android or iOS device with BLE capability
- A *smart*BASIC iBeacon program
- App for Android or iOS device for detecting iBeacons
  - For Android: Nordic nRF Master Control Panel.
  - For iOS: Michael krolls BLExplr

Laird currently provides two sample applications for demonstrating iBeacon functionality for the BL600 on our GitHub site.

- https://github.com/LairdCP/BL600-Applications/blob/master/iBeacon.minimal.sb
- https://github.com/LairdCP/BL600-Applications/blob/master/%24autorun%24.iBeacon.sb

iBeacon.minimal provides a straightforward simple implementation of iBeacons in *smart*BASIC that, when run, creates an iBeacon and starts advertising. It does include a GPIO-controlled function to wipe the file system to allow it to load a new *smart*BASIC program over the air (OTA) if required. It facilitates the module booting directly into VSP command mode and is particularly useful when working with Bx600 breakout boards which may not have access to a UART. Please refer the Command and Bridge Mode Operation BL600 extensions guide which is available from the BL600 product page. iBeacon.minimal is a standalone program which uses no libraries.

$autorun$.ibeacon provides the same basic iBeacon functionality but adds the ability to configure the iBeacon parameters over the air (OTA) using a suitable application on a smartphone. It uses a custom service and characteristics that allow new values for the ibeacon parameters to be set during a connection, which will then be used when the module starts advertising again. Note the use of library files.

Note: The UWTerminal download is available from the Laird Embedded Wireless Solutions Support site:
https://laird-ews-support.desk.com/?b_id=1945#software

This application note assumes you are familiar and with downloading *smart*BASIC applications into the BL600 using UWTerminal. If not, please refer to the module's user guide. All BL600 documentation is also available from the Laird Embedded Wireless Solutions Support site:
https://laird-ews-support.desk.com/?b_id=1945#docs

Product information can also be accessed from the BL600 product page on the Laird website:
http://www.lairdtech.com/products/bl600-series

Checking Firmware Version

To check the firmware version, attach the BL600 development kit to a USB port on your PC and run UWTerminal. Once the module is running, type **AT I 3** to display the firmware version. See Figure 2.

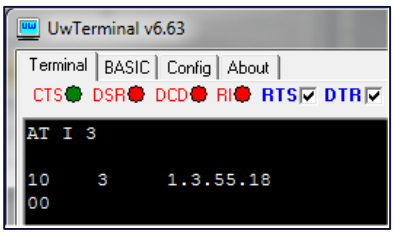


Figure 2: Firmware version display

If the firmware version is older than 1.2.54.0, then you must upload new firmware. Please contact a Laird representative for the latest engineering or production firmware.

BL600 Devkit Setup

After having downloaded the sample applications from our GitHub and loaded them onto the BL600, the ibeacon.minimal smartBASIC sample program can be run from UWterminal by entering *ibeacon* followed by return as shown in Figure 3. The program begins advertising in iBeacon format.

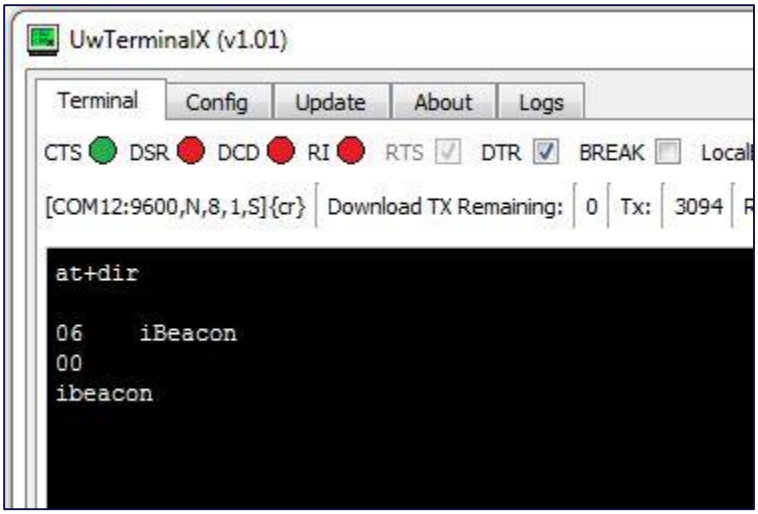


Figure 3: Running ibeacon.minimal

The $autorun$.ibeacon smartBASIC sample program is supplied as an autorun program, so the devkit must be configured if you want the program to run on startup. You can also manually run the program by entering its name **$autorun$** followed by return. Autorun is controlled by sio_28 which is controlled by switch 2-6 on CON12 of the development board. Please refer to the BL600 User Guide, available on the BL600 product page https://laird-ews-support.desk.com/?b_id=1945 for more information.

- CON12 (switch 2 or pins 2-6) MUST be CLOSED (in ON position) to allow a PC (using uWTerminal) to control nAutoRUN pin (SIO.28); with no jumper fitted to J6
- CON12 (switch 2 or pins 2-6) MUST be OPEN to allow nAutoRUN pin (SIO.28) to be controlled by J6 (with jumper fitted to J6 pin1-2 to select nAutoRUN)

To enable Autorun mode while connected to UwTerminal:

1. On the BL600 devkit, close DIP switch 2-6 on CON12. See Figure 4.
2. In UwTerminal, ensure that the DTR checkbox is ticked to enable autorun on startup. See Figure 5.

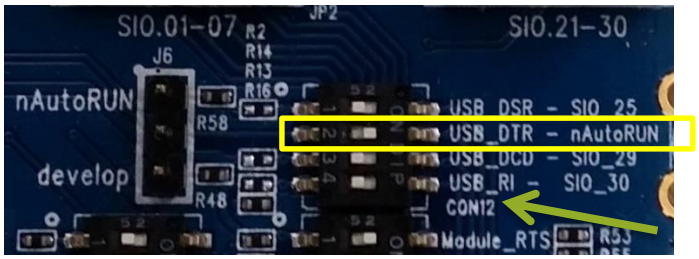


Figure 4: Closing DIP switch 2-6

Figure 5: Asserting DTR line in UwTerminal

To enable Autorun mode when not connected to UwTerminal:

1. On the BL600 devkit, place a jumper on the nAutoRun pin and the pin under it. See Figure 6.
2. Ensure that SW4 is in the DC position and SW5 is in the 3V3 position. If powering from a coin cell battery, ensure that SW6 is in the "CR2032" position. See Figure 7.

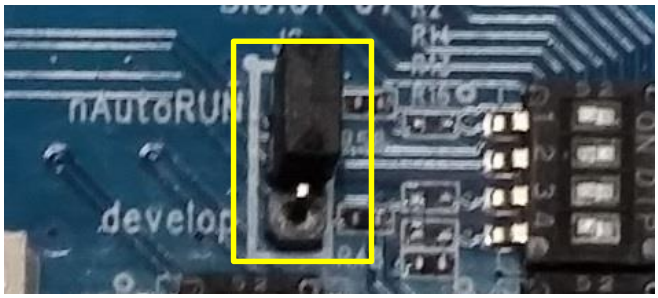


Figure 6: Jumper position for Autorun mode

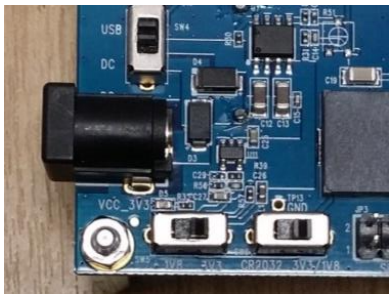


Figure 7: Setting SW4 SW5 and SW6

CONFIGURING $AUTORUN$.IBEACON OVER THE AIR

The custom iBeacon service used in $autorun$.ibeacon allows the user to configure the iBeacon and advertising parameters remotely via any BLE device that can read and write to characteristics such as a smartphone running Master Control Panel (MCP).

These are the general steps to change the value of **any** of the characteristics which may vary slightly depending on the app used:

1. Reset the module – it will remain connectable for 20 seconds.
2. Open nRF Master Control Panel (Android) or BLExplr (iOS) and connect to the iBeacon.
3. Select the iBeacon service by tapping its UUID (569a**1900**-b87f-490c-92cb-11ba5ea5167c).
4. Select the desired characteristic by tapping its UUID.

5. Supply a Hex value **.
6. Disconnect from the BL600 to update the iBeacon service.

See the appendix for a list of UUID's used to configure the iBeacon.

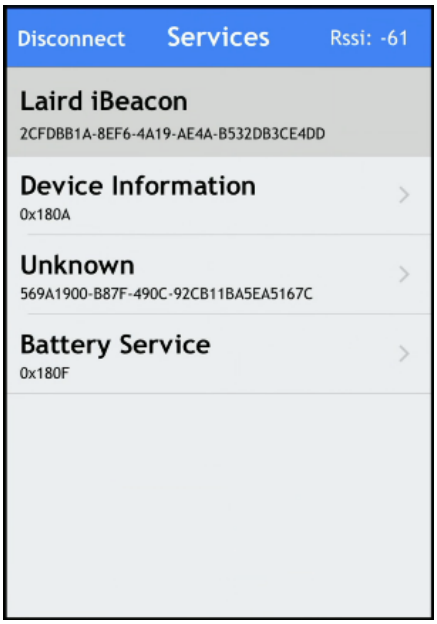


Figure 8: Services shown in BLExpir

Figure 9: Characteristics within service (BLExpir)

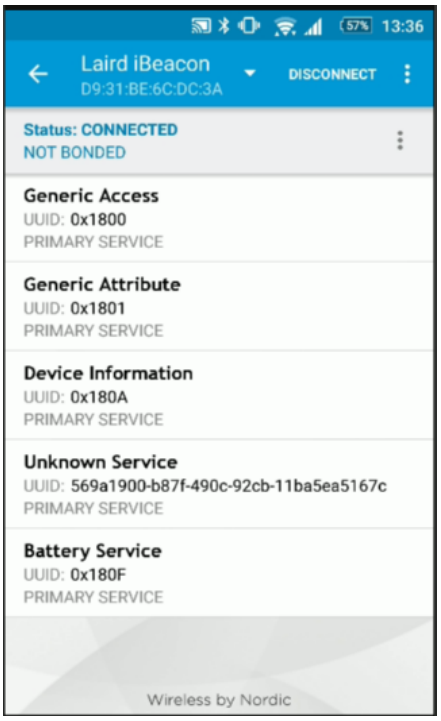


Figure 10: Services shown in MCP

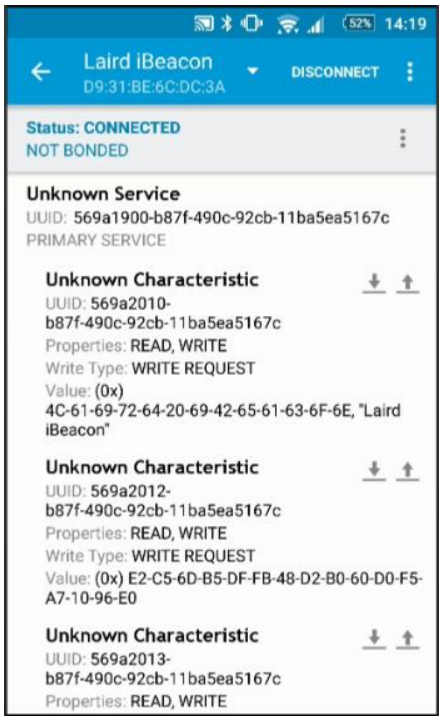


Figure 11: Characteristics within service (MCP)

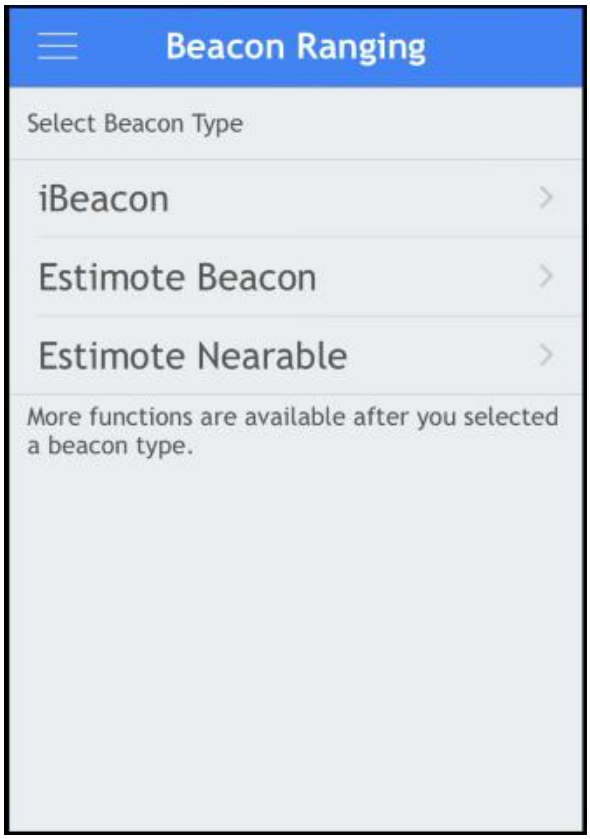
Note: When a characteristic is written to, its new value, if valid is saved in non-volatile memory once disconnected from the module.

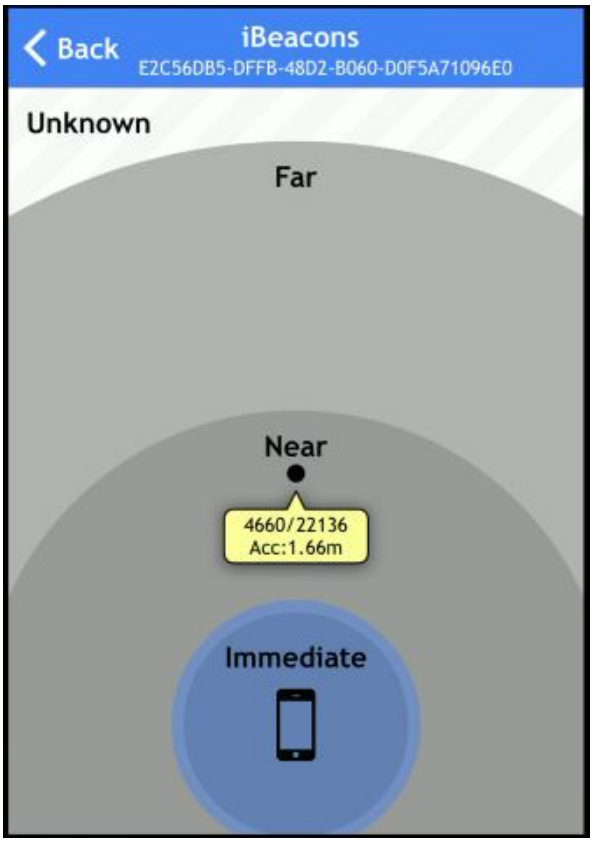
* The value of the Remain Connectable Time characteristic determines how long the module will advertise with type 0 – Connectable. After this time, adverts are restarted with type 2 – Not connectable. The value can be changed by following the steps in the section **Configuring the iBeacon**. The UUID of this characteristic is 569a2018-b87f-490c-92cb-11ba5ea5167c.

** With the exception of the Device Name characteristic (569a2010-b87f-490c-92cb-11ba5ea5167c) where a value can be supplied in Hex or ASCII, all characteristic values shall be supplied in Hex.

TESTING IBEACON FUNCTIONALITY ON IOS

On iOS, various iBeacon programs are available but here we are using BLExplr's beacon ranging mode to show our iBeacon in operation. After running one of the iBeacon sample programs it should appear in the beacon ranging part of the app. Note that this app and others may by default only recognize certain iBeacon UUID's set in the manufacturer specific data type.





TESTING IBEACON FUNCTIONALITY ON ANDROID

On Android, various apps can be used to test iBeacon functionality. Here we are using Nordics Master Control panel (MCP). Initiate a scan after having run ibeacon.minimal and you should see the BLE advert complete with the iBeacon fields displayed. By selecting the raw option, you can display the datatypes that make up the iBeacon advert. Note the 0xff data type which contains the iBeacon data (Figure 12).

Figure 12: iBeacon advert datatypes

ADDITIONAL INFORMATION

Apple developers iBeacon page https://developer.apple.com/ibeacon/

iOS BLExplr (Michael Kroll) on the Apple App store https://itunes.apple.com/gb/app/blexplr/id524018027?mt=8

Android Master Control Panel (Nordic Semiconductor ASA) on Google Play
https://itunes.apple.com/gb/app/blexplr/id524018027?mt=8

Documentation, product information, and software downloads are available from the Embedded Wireless Solutions Support Center: https://laird-ews-support.desk.com/?b_id=1945

Product information can also be accessed from the BL600 product page on the Laird website:
http://www.lairdtech.com/products/bl600-series

More information on how to create BLE beacons in *smart*BASIC can be found in the Beacons for smartBASIC Walkthrough document.

APPENDIX

UUIDs	
Format: 569aABCD-b87f-490c-92cb-11ba5ea5167c	
Where 0xABCD is a 16 bit offset from the Laird Base UUID	

- Laird Base:	569a0000-b87f-490c-92cb-11ba5ea5167c
- iBeacon Service:	569a1900-b87f-490c-92cb-11ba5ea5167c
- Device Name Char:	569a2010-b87f-490c-92cb-11ba5ea5167c
- Format Char:	569a2011-b87f-490c-92cb-11ba5ea5167c
- iBeacon UUID Char:	569a2012-b87f-490c-92cb-11ba5ea5167c
- Major Char:	569a2013-b87f-490c-92cb-11ba5ea5167c
- Minor Char:	569a2014-b87f-490c-92cb-11ba5ea5167c
- Advertising Interval Char:	569a2015-b87f-490c-92cb-11ba5ea5167c
- Advertising Time out Char:	569a2016-b87f-490c-92cb-11ba5ea5167c
- Calibrated RSSI Value Char:	569a2017-b87f-490c-92cb-11ba5ea5167c
- Remain Connectible Time Char:	569a2018-b87f-490c-92cb-11ba5ea5167c
Adopted Characteristics: (16 bit value as offset to Bluetooth SIG Base UUID)	
- Tx Power Level:	0x2A07

REVISION HISTORY

Version	Date	Notes	Approved By
1.0	03 Dec 2014	Initial Release	David Davis/Jonathan Kaye
1.1	12 Jan 2015	Updated *Additional Information* links	Sue White
1.2	05 Jan 2016	Added information to cover multiple versions of iBeacon programs	Mark Duncombe