

WAP-2-023

BLE Connectivity Specification

Version 1.4

26th December 2024

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Change | Author | Description | Version |
| 17th July 2024 | Creation | Tom Hao | Initial draft with Windsor | 1.0 |
| 30th July 2024 | Add | Tom Hao | Add Brighton Red Light | 1.1 |
| 12th September 2024 | Modify | Tom Hao | Change Service Data fields of Windsor | 1.2 |
| 24th December 2024 | Modify | Tom Hao | Change “product params” | 1.3 |
| 26th December 2024 | Modify | Tom Hao | Change Service Data fields with device code and device state | 1.4 |

1. Scope

This document describes BLE advertising parameters of OneBase products. Hardware team should implement the BLE advertisement packets according to this, and software team should use this to filter OneBase products and extract information during BLE scanning.

We will introduce some common knowledge about BLE advertising and then discuss details of each OneBase hardware.

1. Common
   1. Roles of BLE communication

There are two main roles in BLE communication: peripheral and central.

Peripherals send radio signals in certain BLE channels repeatedly with a constant interval. Centrals would listen in these channels and receive these signals to discover the presence of peripherals. This process is called advertising and scanning.

Once discovered a peripheral, the central could establish a connection to it. Once connected with a central, the peripheral normally stops advertising, making itself invisible to other centrals.

OneBase hardware normally act as peripheral, and Smart phones with OneBase application normally act as central.

* 1. Structure of BLE advertisement data

There are two packets to carry advertisement data: advertising packet and scan response packet. Payload length is 31 bytes for each of them. The advertising packet would be sent by peripheral devices repeatedly, and after it’s discovered by central device, the central device would send a scan request packet to the peripheral, then the peripheral device would send a scan response packet. Whether to send scan request and get the scan response packet depends on the central device, which may not be able to control by application, so it’s better to put important information in advertising packet.

The payload of advertising or scan response packet is divided into fields, each filed contains three parts:

|  |  |  |
| --- | --- | --- |
| Length | Type | Content |
| 1 Byte | 1 Byte | n Bytes |

The value of Length byte equals n+1 (length of Type and Content). And length of the whole filed equals n+2.

The value of Type byte identifies what kind of information this field is about and how the content bytes should be interpreted. Available Type values are described in Bluetooth specification ([Assigned Numbers | Bluetooth® Technology Website](https://www.bluetooth.com/specifications/assigned-numbers/) 2.3 Common Data Types) and defined in components\softdevice\s140\headers\ble\_gap.h of Nordic nRF5 SDK. Here’s a list of some Type values we would use:

|  |  |
| --- | --- |
| Type value | Type name |
| 0x01 | Flags |
| 0x02 | Incomplete List of 16-bit Service or Service Class UUIDs |
| 0x03 | Complete List of 16-bit Service or Service Class UUIDs |
| 0x08 | Shortened Local Name |
| 0x09 | Complete Local Name |
| 0x16 | Service Data - 16-bit UUID |
| 0xFF | Manufacturer Specific Data |

1. Windsor / WAP-1-100
   1. Advertising interval

100ms

* 1. Tx power

4dBm

* 1. Advertisement fields

All fields described below should be put in advertising packet.

* Service UUIDs

|  |  |  |
| --- | --- | --- |
| Length | Type | Content |
| 0x03 | 0x03 | 0x42 0x4F |

This field is used by peripheral to advertise what services it provides. The content should be an array of 16bit UUIDs of little endian. It’s OK to be different than the services discovered after connection. We use 0x4F42 (ASCII codes of “OB” as in OneBase) here.

This field could be used as filter by iOS and Android BLE scanning API.

* Short Local Name

|  |  |  |
| --- | --- | --- |
| Length | Type | Content |
| 0x08 | 0x08 | 0x4F 0x6E 0x65 0x42 0x61 0x73 0x65 |

This field is name of the peripheral. Its content is a UTF-8 encoded string. Here we use ‘OneBase’.

* Service Data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Length | Type | Content | | |
| 0x11 | 0x16 | UUID | | 0x42 0x4F (same as Service UUID) |
| Data | Device Code | 2 Byte, Uint16, Big Endian |
| Device State | 4 Bytes, reserved |
| Chip Id | 8 Bytes (unique to nRF Chip) |

Service Data field is where we put product information. Its content should start with a 16bit service UUID followed by data bytes.

Device Code should be used by app to look up in database for model information.

Device State is reserved for session status and sensor values, set to all zeros for now.

* 1. Example advertising packet
* HEX Bytes (31 Bytes)

03 03 42 4F 08 08 4F 6E 65 42 61 73 65 11 16 42 4F 11 22 00 00 00 01 02 03 04 05 06 07 08 00

* Parsed

Service UUIDs: [0x4F42]

Name: OneBase

Service Data:

UUID: 0x4F42

Data:

Device Code: 0x1122

Device State: 0x00 0x00 0x00 0x00

Chip Id: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08

* 1. Future expanding

Currently, we only use advertising packet and used all 31 bytes. If we are going to add more data in advertisement, we could use scan response packet and “Manufacture Specific Data” type. If new data is important and should be in advertising packet, we can move “Short Local Name” to scan response packet.

1. Brighton / WAP-1-901

The BLE module of Brighton is developed by the manufacture, not by OneBase team.

* 1. Advertising interval

200ms

* 1. TX power

Unknown

* 1. Advertisement fields
* Flags

|  |  |  |
| --- | --- | --- |
| Length | Type | Content |
| 0x02 | 0x01 | 0x06 |

* Service UUIDs

|  |  |  |
| --- | --- | --- |
| Length | Type | Content |
| 0x03 | 0x02 | 0xF0 0xFF |

* Complete Local Name

|  |  |  |
| --- | --- | --- |
| Length | Type | Content |
| 0x13 | 0x09 | 0x4F 0x6E 0x65 0x42 0x61 0x73 0x65 0x4F 0x72 0x69 0x52 0x65 0x64 0x4C 0x69 0x67 0x68 0x74 |

We can change the complete local name with a command after connection.

* 1. Full advertising packet
* HEX Bytes (27 Bytes)

02 01 06 03 02 F0 FF 13 09 4F 6E 65 42 61 73 65 4F 72 69 52 65 64 4C 69 67 68 74

* Parsed

Flags: General discoverable, BR/EDR not supported

Service UUIDs: [0xFFF0]

Name: OneBaseOriRedLight