BLE Device Information Service

1.0

Features

- Custom Server in Peripheral Role operation
- Device Information Service usage
- DeepSleep mode demonstration
- LED status indication

General Description

This is a simple BLE example project that demonstrates how to configure and use BLE component APIs and an application layer callback. The Device Information Service is used as an example to demonstrate how to configure the BLE service characteristics in the BLE component.

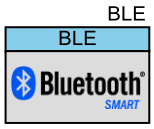
Development Kit Configuration

Build the project and program the hex file onto the PSoC4 CY8CKIT-042 BLE Pioneer Kit.

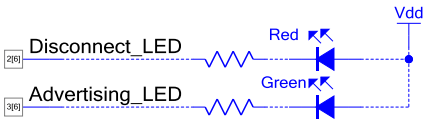
Project Configuration

The example project consists of the following components: BLE and digital output pin. The input pin is configured to the resistive pull up mode and used to wake the device from low power hibernate mode. The top design schematic is shown in **Figure 1**.

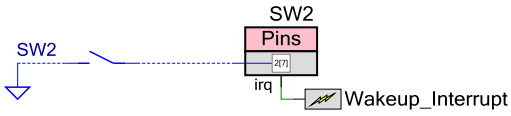
BLE Device Information Service Example Project



BLE component contains Device Information Service.



The red LED is used to indicate that the device is disconnected.
The green LED is used to indicate that the device is advertising.



The button is used to wake the device up from the hibernate mode.

Figure 1. Top design schematic

The BLE component is configured as the Custom Profile in Peripheral Role. Device Information Service is added with all the characteristics.

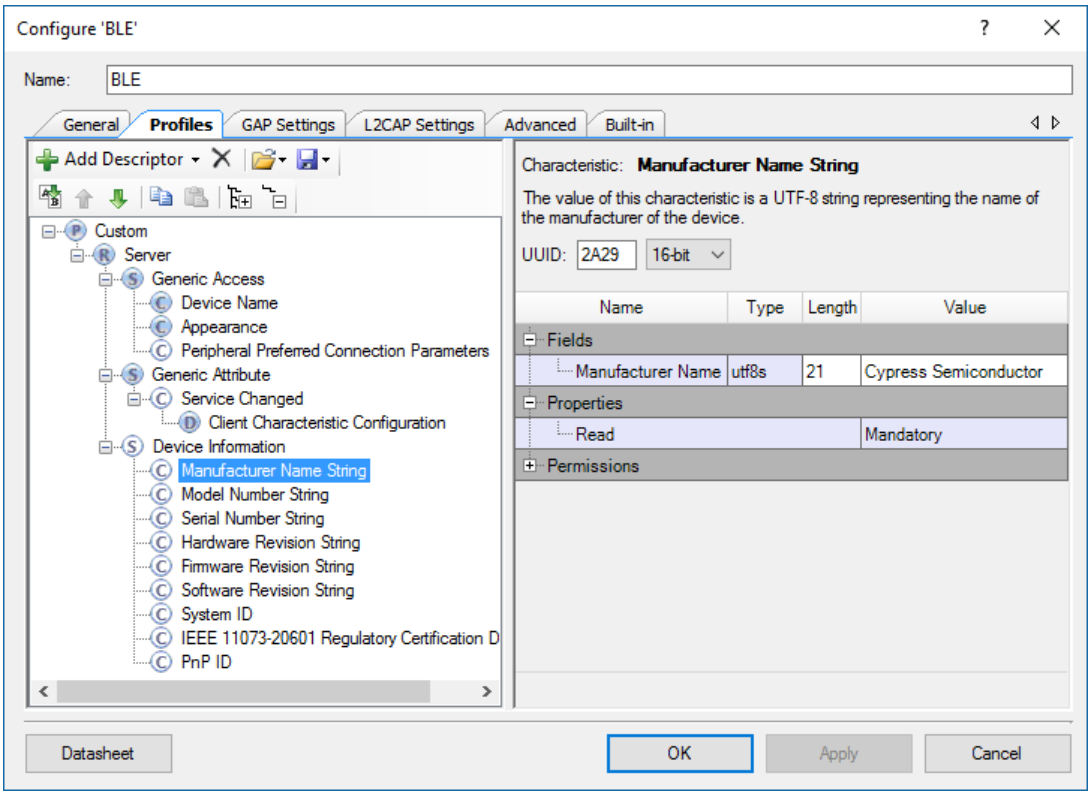


Figure 2. GATT settings

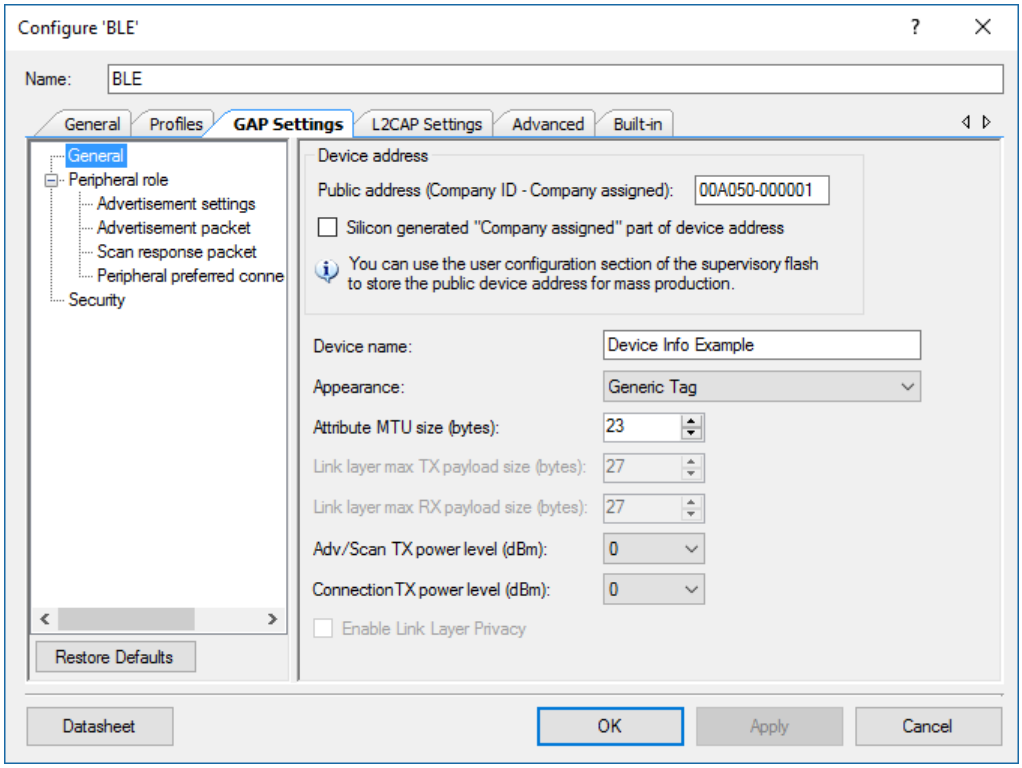


Figure 3. GAP settings

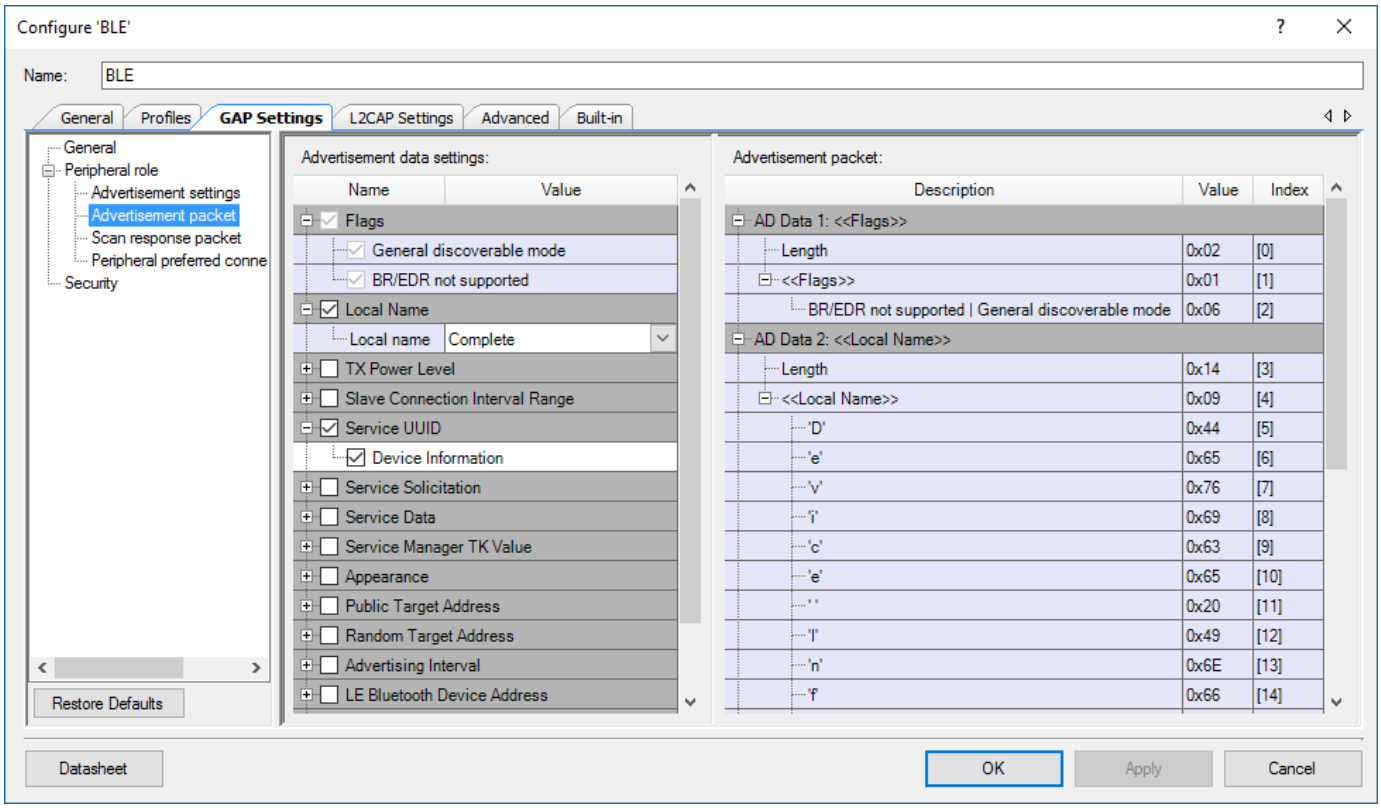


Figure 4. GAP settings -> Advertisement packet

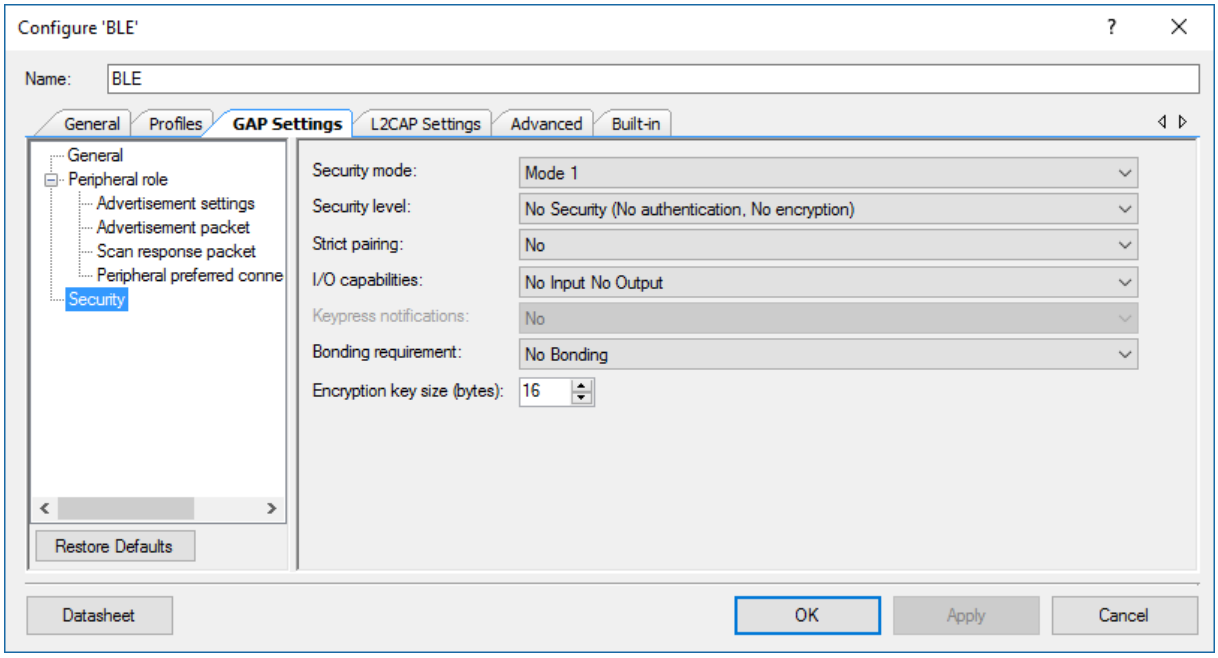


Figure 5. Security settings

Project Description

One callback function (AppCallBack()) is required to receive generic events from BLE Stack.

CyBle_GappStartAdvertisement() API is called after the CYBLE_EVT_STACK_ON event to start advertising with the packet shown in **Figure 4**.

To indicate that the device is advertising, the green LED is blinking. The red LED will be lighted on to indicate that no Client is connected to the device. When a Client is connected successfully, both red and green LEDs will turn off.

Expected Results

You can use CySmart mobile app as a Device Information Service client:

- Launch CySmart mobile app (Android/iOS), and swipe down to refresh list of found BLE devices.
- Connect to “Device Info Example” service and open “Device Information Service” service.
- Explore information about device:

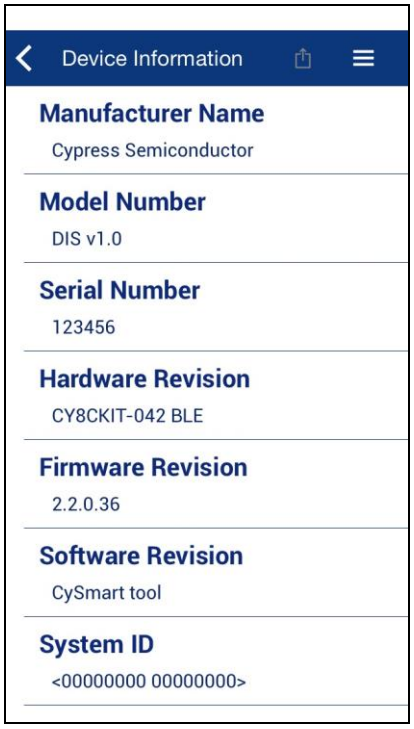


Figure 6. CySmart iOS app

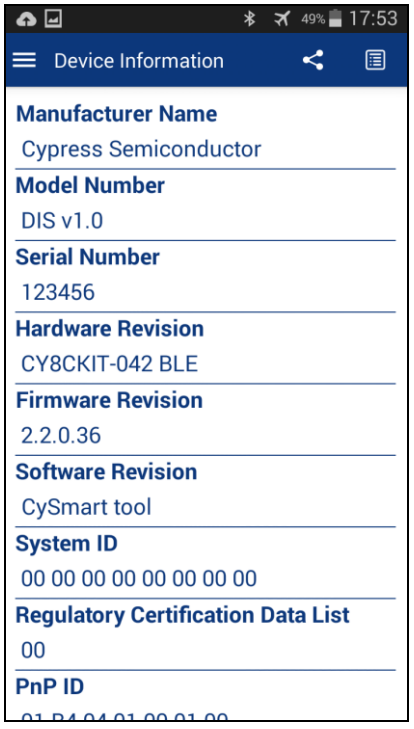


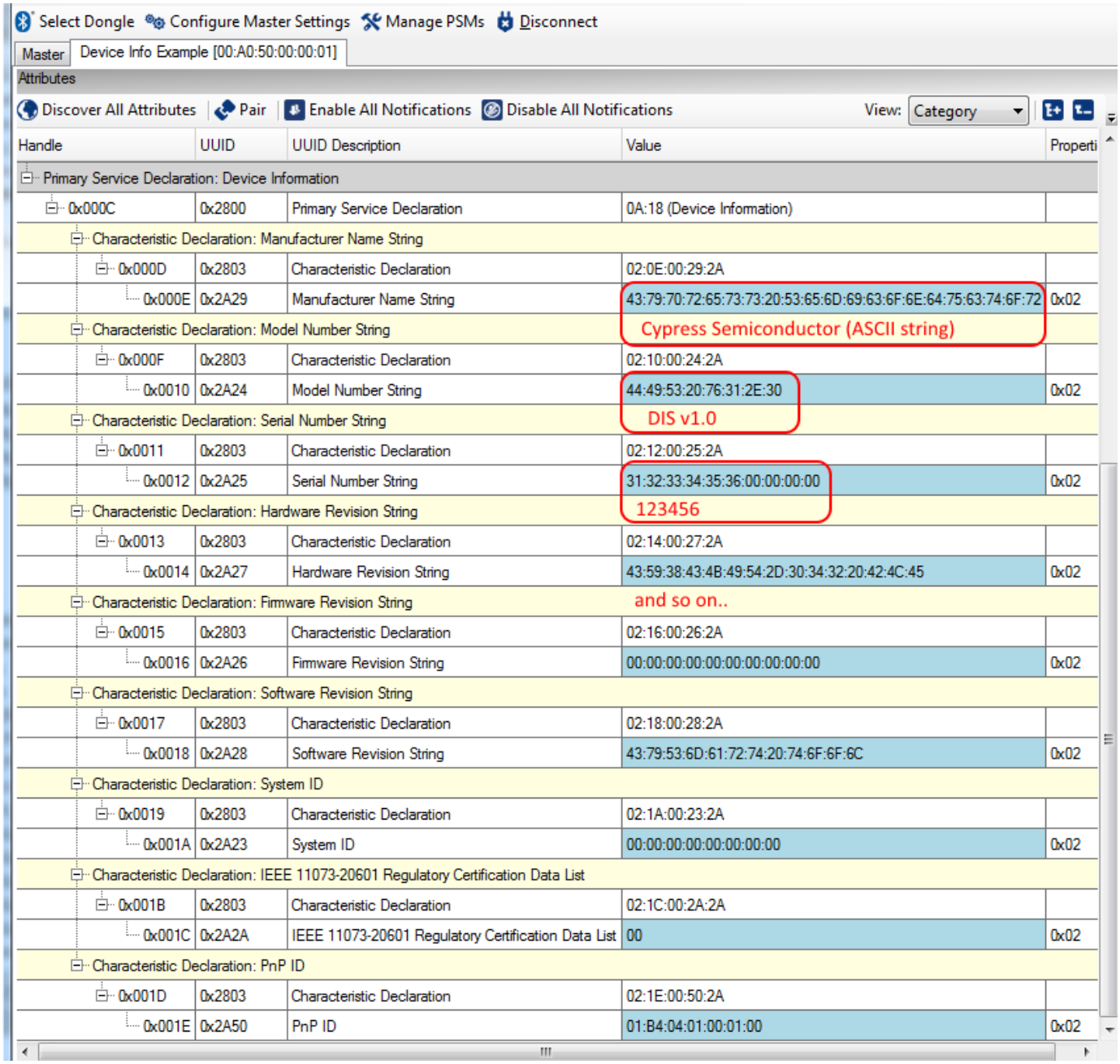
Figure 7. CySmart Android app

Also, the Device Information Service Example can be used together with CySmart app for Windows. No security procedures (pairing) required. For further instructions on how to use CySmart application, see CySmart User Guide.

The simple example how to use CySmart Windows application as Device Information Service client is the next:

- Connect the CySmart BLE dongle to a USB port on the PC.
- Launch **CySmart** app and select connected dongle in the dialog window.
- Reset the development kit to start advertising by pressing **SW1** button.
- Click **Start Scan** button to discover available devices.
- Select **Device Info Example** in the list of available devices and connect to it.
- Click **Discover All Attributes**, and **Read All Characteristics** in CySmart app.

Observe the Device Information Service characteristics values read with example data:


Select Dongle
Configure Master Settings
Manage PSMs
Disconnect
Master
Device Info Example [00:A0:50:00:00:01]
Attributes
Discover All Attributes
Pair
Enable All Notifications
Disable All Notifications
View: Category
Handle | UUID | UUID Description | Value | Properti
Primary Service Declaration: Device Information
0x000C | 0x2800 | Primary Service Declaration | 0A:18 (Device Information)
Characteristic Declaration: Manufacturer Name String
0x000D | 0x2803 | Characteristic Declaration | 02:0E:00:29:2A
0x000E | 0x2A29 | Manufacturer Name String | 43:79:70:72:65:73:73:20:53:65:6D:69:63:6F:6E:64:75:63:74:6F:72 | 0x02
Cypress Semiconductor (ASCII string)
Characteristic Declaration: Model Number String
0x000F | 0x2803 | Characteristic Declaration | 02:10:00:24:2A
0x0010 | 0x2A24 | Model Number String | 44:49:53:20:76:31:2E:30 | 0x02
DIS v1.0
Characteristic Declaration: Serial Number String
0x0011 | 0x2803 | Characteristic Declaration | 02:12:00:25:2A
0x0012 | 0x2A25 | Serial Number String | 31:32:33:34:35:36:00:00:00:00 | 0x02
123456
Characteristic Declaration: Hardware Revision String
0x0013 | 0x2803 | Characteristic Declaration | 02:14:00:27:2A
0x0014 | 0x2A27 | Hardware Revision String | 43:59:38:43:4B:49:54:2D:30:34:32:20:42:4C:45 | 0x02
Characteristic Declaration: Firmware Revision String
and so on..
0x0015 | 0x2803 | Characteristic Declaration | 02:16:00:26:2A
0x0016 | 0x2A26 | Firmware Revision String | 00:00:00:00:00:00:00:00:00:00 | 0x02
Characteristic Declaration: Software Revision String
0x0017 | 0x2803 | Characteristic Declaration | 02:18:00:28:2A
0x0018 | 0x2A28 | Software Revision String | 43:79:53:6D:61:72:74:20:74:6F:6F:6C | 0x02
Characteristic Declaration: System ID
0x0019 | 0x2803 | Characteristic Declaration | 02:1A:00:23:2A
0x001A | 0x2A23 | System ID | 00:00:00:00:00:00:00:00 | 0x02
Characteristic Declaration: IEEE 11073-20601 Regulatory Certification Data List
0x001B | 0x2803 | Characteristic Declaration | 02:1C:00:2A:2A
0x001C | 0x2A2A | IEEE 11073-20601 Regulatory Certification Data List | 00 | 0x02
Characteristic Declaration: PnP ID
0x001D | 0x2803 | Characteristic Declaration | 02:1E:00:50:2A
0x001E | 0x2A50 | PnP ID | 01:B4:04:01:00:01:00 | 0x02