# RW BLE Device Information Service Interface Specification

Interface Specification
RW-BLE-PRF-DIS-IS
Version 8.0
2015-07-29

RivieraWaves
http://www.rivierawaves.com/

## Revision History

| Version | Date | Revision Description | Author |
|---|---|---|---|
| 1.0 | 2012-08-14 | Initial release | LT |
| 2.0 | 2012-12-03 | Client Multi-Instance API + Server rework | LT |
| 7.00 | 2014-06-30 | Update of API for BLE 4.1 | FB |
| 7.01 | 2014-11-27 | Minor API renaming | FB |
| 8.00 | 2015-07-29 | Update for BLE 4.2 | CM |

# Table of Contents

## Abbreviations

| Abbreviation | Original Terminology |
|---|---|
| API | Application Programming Interface |
| ATT | Attribute |
| BLE | Bluetooth Low Energy |
| DB | Database |
| DIS | Device Information Service |
| DISC | Device Information Service Client |
| DISS | Device information Service Server |
| FW | Firmware |
| GAP | Generic Access Profile |
| GATT | Generic Attribute Profile |
| IS | Interface Specification |
| PRF | Profile |
| RW | RivieraWaves |
| SVC | Service |

# 1 Overview

## 1.1 Document Overview

This document describes the non-standard interface of the RW BLE Device Information Service (DIS) implementation. Along this document, the interface messages will be referred to as API messages for the profile block(s).

Their description will include their utility and reason for implementation for a better understanding of the user and the developer that may one day need to interface them from a higher application.

## 1.2 Protocol Overview

The Bluetooth Low Energy Device Information Service enables the user to expose manufacturer and/or vendor information about a device. Within the profile, two roles can be supported: **Client** and S**erver**. The client must support the GAP Central Role and the Server, the GAP Peripheral role. The profile requires a connection to be established between the two devices for its functionality.

The various documents edited by the Bluetooth SIG GPA Working group present different use cases for this profile, their GATT, GAP and security, mandatory and optional requirements. The DIS specification has been adopted by the Bluetooth SIG on November 19$^{st}$ 2011 ([1]). Its related Test Specifications has been released at the same time and are referenced in [2].

The profile is implemented in the RW-BLE software stack as two tasks, one for each role. Each task has an API decided after the study of the profile specifications and test specifications, and it is considered to be minimalistic and designed for a future application which would combine the profile functionality with the device connectivity and security procedures.

## 1.3 Firmware Implementation Overview

Basically, if a device needs only be DIS Server, the firmware should be compiled with this role only, and inversely for the Client role.

The DIS is part of several profiles (Health Thermometer, Heart Rate …). Implemented as an independent task, it can also be used in a proprietary profile.

The Applications which will control the roles on end-products are responsible with creating the connection between the devices, using suggested advertising intervals and data, connection intervals, security levels, etc. The Profile implementation allows modulating the behavior depending on the final needs. Profile role enabling should be immediate after connection creation in order to allow correct profile behavior with the peer device.

# 2 Device Information Service Server (DISS)

This role is meant to be activated on the device that share information. It implies it is a GAP Peripheral. Please refer to "diss_task.h" for implementation of this API.

Device Information Service support multipoint and scatternet. It is a mono-instantiated task; application doesn't have to manage connection index.

## 2.1 Initialization / Database Creation

During the initialization phase of the device, to use the Device Information Service task, the DISS task has to be allocated and corresponding attribute database initialized, using GAPM API. Application has to send GAPM_PROFILE_TASK_ADD_CMD [3] with specific device required security level and following parameters.

**Parameters**:

| Type | Parameters | Description |
|---|---|---|
| uint16_t | features | Indicate characteristics that are supported. (see Table 1) |

All characteristics of the Device Information Service are optional. However, some profiles require the presence of several of these characteristics. Please refer to the specification of these profiles for more information.

| Value | Flag | Description |
|---|---|---|
| 0x0001 | DIS_MANUFACTURER_NAME_CHAR_SUP | Indicate if Manufacturer Name characteristic is supported |
| 0x0002 | DIS_MODEL_NB_STR_CHAR_SUP | Indicate if Model Number String characteristic is supported |
| 0x0004 | DIS_SERIAL_NB_STR_CHAR_SUP | Indicate if Serial Number String characteristic is supported |
| 0x0008 | DIS_HARD_REV_STR_CHAR_SUP | Indicate if Hardware Revision String characteristic is supported |
| 0x0010 | DIS_FIRM_REV_STR_CHAR_SUP | Indicate if Firmware Revision String characteristic is supported |
| 0x0020 | DIS_SW_REV_STR_CHAR_SUP | Indicate if Software Revision String characteristic is supported |
| 0x0040 | DIS_SYSTEM_ID_CHAR_SUP | Indicate if System ID characteristic is supported |
| 0x0080 | DIS_IEEE_CHAR_SUP | Indicate if IEEE 11073-20601 Regulatory Certification Data List characteristic is supported |
| 0x0100 | DIS_PNP_ID_CHAR_SUP | Indicate if PnP ID characteristic is supported |

**Table 1: DIS Supported features**

## 2.2 DISS_SET_VALUE_REQ

**Parameters**:

| Type | Parameters | Description |
|---|---|---|
| uint8_t | value | Characteristic Code (see Table 2) |
| uint8_t | length | Value length |
| uint8_t[length] | data | Value data |

**Response:**

DISS_SET_VALUE_RSP: to inform about state of setting value

**Description**:

This message should be used to initialize any of the characteristic values before a connection with a peer device.

## 2.3 DISS_SET_VALUE_RSP

**Parameters**:

| Type | Parameters | Description |
|---|---|---|
| uint8_t | value | Characteristic Code (see Table 2) |
| uint8_t | status | Status of the request (see [5]) |

**Description**:

This message is triggered when DISS_SET_VALUE_REQ has been processed.

## 2.4 DISS_VALUE_REQ_IND

**Parameters**:

| Type | Parameters | Description |
|---|---|---|
| uint8_t | value | Characteristic Code (see Table 2) |

**Response**:

DISS_VALUE_CFM: triggered by application to provide

**Description**:

This API message is triggered by DISS task when peer device request a value not initialized by application.

This can be used to read a non-volatile or hardcoded value without reserving memory.

## 2.5 DISS_VALUE_CFM

**Parameters**:

| Type | Parameters | Description |
|---|---|---|
| uint8_t | value | Characteristic Code (see Table 2) |
| uint8_t | length | Value length |
| uint8_t[length] | data | Value data |

**Description**:

This API message should be sent by application when DISS_VALUE_REQ_IND is triggered by profile in order to provide the characteristic value to peer device.

# 3 Device Information Service Client (DISC)

This role is meant to be activated on the device that will locate the Server. It implies it is a GAP Central. The FW task for this role will discover the Device Information Service present on the peer Server, after establishing connection, and will allow reading different information about the device.

Please refer to “disc_task.h” for implementation of this API.

***Important Note***: The TASK_DISC task is multi-instantiated, one instance is created for each connection for which the profile will be enabled and each of these instances will have a different task ID. Thus, it is very important for the application to keep the source task ID of the DISC_ENABLE_CFM message to be able to communicate with the peer device linked to this task ID once it has been enabled.

The term TASK_DISC_IDX will be used in the rest of the document to refer to any instance of the Device Information Service Client Role Task. The term TASK_DISC will refer to the first instance of this task.

## 3.1 Initialization

During the initialization phase of the device, to use the Device Information Service Client task, the DISC task has to be allocated using GAPM API. Application has to send GAPM_PROFILE_TASK_ADD_CMD [3].

## 3.2 DISC_ENABLE_REQ

**Parameters**:

| Type | Parameters | Description |
|---|---|---|
| uint8_t | con_type | Connection type:<br>- 0: 1st discovery<br>- 1: normal connection, Reuse bond data |
| struct dis_disc_content | dis | Saved DIS information (bond data - see Table 3) |

**Response**: DISC_ENABLE_RSP

**Description**:

This API message is used for enabling the Client role of the Device Information Service. The Application sends it with connection index related to the peer device connected with the connection type and the previously saved discovered DIS details on peer.

The Task for this profile role will go from IDLE state to CONNECTED state.

The connection type may be *0 = Connection for discovery* or *1 = Normal connection*. This difference has been made and Application would handle it in order to not discover the DIS on the Server at every connection, but do it only once and keep the discovered details in the Client device between connections.

If it is a discovery type connection, the *dis* parameter is useless; it will be filled with 0's. Otherwise it will contain pertinent data which will be kept in the Client environment while enabled. It allows for the Application to not be aware of attribute details. For a normal connection, the response to this request is sent right away after saving the *dis* content in the environment. For a discovery connection, discovery of the peer DIS is started and the response will be sent at the end of the discovery with the discovered attribute details.

## 3.3 DISC_ENABLE_RSP

**Parameters**:

| Type | Parameters | Description |
|---|---|---|
| uint8_t | status | Status of the request (see [5]) |
| struct dis_disc_content | dis | Discovery result (bond data to keep for next connections - see Table 3) |

**Description**:

This API message is used by the Client to either send the discovery results of DIS on Server or confirm enabling of the Client role, or to simply confirm enabling of Client role if it is a normal connection and the DIS details are already known. An error may have also occurred and is signaled.

## 3.4 DISC_RD_CHAR_REQ

**Parameters**:

| Type | Parameters | Description |
|---|---|---|
| uint8_t | char_code | Characteristic Code (see Table 2) |

**Response**: DISC_RD_CHAR_RSP or DISC_ERROR_IND

**Description**:

This API message is used by the application to send a GATT_READ_CHAR_REQ with the parameters deduced from the *char_code*. Upon reception of this message, DISC checks whether the parameters are correct, then if the handle for the characteristic is valid (not 0x0000) and the request is sent to GATT.

When the peer has responded to GATT, and the response is routed to DISC, the DICS_RD_CHAR_RSP message will be generically built and the Application must be able to interpret it based on the read request it made. And error status is also possible either for the Read procedure or for the application request, in the second case, the DISC_ERROR_IND message is sent to Application.

No parsing intelligence of the received response is added in this API handler, so all the work of interpretation must be added in the Application depending of its request and use of the response.

## 3.5 DISC_RD_CHAR_RSP

**Parameters**:

| Type | Parameters | Description |
|---|---|---|
| uint16_t | handle | Attribute Handle |
| uint16_t | length | Value length |
| uint8_t | status | Status of the request (see [5]) |
| uint8_t[length] | data | Value data |

**Description**:

This API message is used by the Client role to inform the Application of a received read response. The status and the data from the read response are passed directly to Application, which must interpret them based on the request it made.

# 4 Miscellaneous

| Value | Flag | Description |
|---|---|---|
| 0x00 | DIS_MANUFACTURER_NAME_CHAR | Manufacturer Name |
| 0x01 | DIS_MODEL_NB_STR_CHAR | Model Number |
| 0x02 | DIS_SERIAL_NB_STR_CHAR | Serial Number |
| 0x03 | DIS_HARD_REV_STR_CHAR | HW Revision Number |
| 0x04 | DIS_FIRM_REV_STR_CHAR | FW Revision Number |
| 0x05 | DIS_SW_REV_STR_CHAR | SW Revision Number |
| 0x06 | DIS_SYSTEM_ID_CHAR | System Identifier Name |
| 0x07 | DIS_IEEE_CHAR | IEEE Certificate |
| 0x08 | DIS_PNP_ID_CHAR | Plug and Play Identifier |

**Table 2: DIS Value Type**

| Type | Parameters | Description |
|---|---|---|
| struct prf_svc | svc | Structure containing the start handle and the end handle of the service<br>/// start handle<br>uint16_t shdl;<br>/// end handle<br>uint16_t ehdl; |
| struct prf_char_inf | chars[DISC_CHAR_MAX] | Structure containing handle and properties of the alert level characteristic<br>/// Characteristic handle<br>uint16_t char_hdl;<br>/// Value handle<br>uint16_t val_hdl;<br>/// Characteristic properties<br>uint8_t prop;<br>/// Number of attributes<br>uint8_t char_ehdl_off; |

**Table 3: DIS Content (struct disc_dis_content)**

# References

| | | | | |
|---|---|---|---|---|
| **[1]** | **Title** | Device Information Service Specification | | |
| | **Reference** | DIS_SPEC_V11r00 | | |
| | **Version** | V11r00 | **Date** | 29/11/2011 |
| | **Source** | Bluetooth SIG – GPA Working Group | | |

| | | | | |
|---|---|---|---|---|
| **[2]** | **Title** | Device Information Service Test Specification | | |
| | **Reference** | DIS.TS.1.1.0 | | |
| | **Version** | 1.1.0 | **Date** | 29/11/2011 |
| | **Source** | Bluetooth SIG – GPA Working Group | | |

| | | | | |
|---|---|---|---|---|
| **[3]** | **Title** | GAP Interface Specification | | |
| | **Reference** | RW-BLE-GAP-IS | | |
| | **Version** | 7.00 | **Date** | 2014-06-30 |
| | **Source** | RivieraWaves SAS | | |

| | | | | |
|---|---|---|---|---|
| **[4]** | **Title** | GATT Interface Specification | | |
| | **Reference** | RW-BLE-GATT-IS | | |
| | **Version** | 7.00 | **Date** | 2014-06-30 |
| | **Source** | RivieraWaves SAS | | |

| | | | | |
|---|---|---|---|---|
| **[5]** | **Title** | RW BLE Host Error Code Interface Specification | | |
| | **Reference** | RW-BLE-HOST-ERR-CODE-IS | | |
| | **Version** | 7.00 | **Date** | 2014-06-30 |
| | **Source** | RivieraWaves SAS | | |